ITERATION 2

OUR CLIENT:

EPI-USE

groupelephant.com
BEYOND CORPORATE PURPOSE

Group 35

CANVAS

NGO PROJECT PORTFOLIO MANAGEMENT SYSTEM

This iteration outlines the updated functional requirements for **EPI-USE** and **groupelephant.com,** Use Case Diagram of the system, Logical Use Case Narratives, Functional Decomposition, ERD, Context and Activity Diagrams, UML Class Diagram and a Technical Design.



Amahle Khoza

u22539702@tuks.co.za

+27 71 023 1872



Stephan Bredenhann

u23667436@tuks.co.za

+27 71 566 0997



Christopher Bailey

u24661092@tuks.co.za

+27 71 365 9963



Nicholas Bailey

u24671691@tuks.co.za

+27 71 610 6952



Alexander Steven

u22496786@tuks.co.za

+27 72 370 0225

Table of Contents

Table of Figures 8

Table of Tables 14

1. Document Introduction 21

2. Logical Design 22

2.1 Introduction 22

2.2 Updated Functional Requirements Lists 22

2.2.1 User Subsystem 22

2.2.2 Admin Subsystem 22

2.2.3 Inventory Subsystem 23

2.2.4 Group Subsystem 23

2.2.5 Warehouse Subsystem 24

2.2.6 Vendor Subsystem 24

2.2.7 Project Management Subsystem 24

2.2.8 K9 Subsystem 25

2.2.9 Rhino Subsystem 26

2.2.10 Elephant Subsystem 26

2.2.11 Message Subsystem 26

2.2.12 Report Subsystem 26

2.2.13 Machine Learning Subsystem 27

2.2.14. System Subsystem 27

2.2.15 Budget Subsystem 27

2.3 Functional Requirements List Descriptions 28

2.3.1 User Subsystem 28

2.3.2 Admin Subsystem 37

2.3.3 Inventory Subsystem 53

2.3.4 Group Subsystem 63

2.3.5 Warehouse Subsystem 71

2.3.6 Vendor Subsystem 76

2.3.7 Project Subsystem 87

2.3.8 K9 Subsystem 111

2.3.9 Rhino Subsystem 127

2.3.10. Elephant Subsystem 132

2.3.11 Message Subsystem 137
2.3.12 Report Subsystem 139
2.3.13 Machine Learning Subsystem 148
2.3.14 System Subsystem 149
2.3.15. Budget Subsystem 152
2.4 Functional Requirements List Modifications 160
2.5 Use Case Diagram 166
2.5.1 Use Case Diagram - User Subsystem 166
2.5.2 Use Case Diagram – Admin Subsystem 167
2.5.3 Use Case Diagram – Inventory Management Subsystem 168
2.5.4 Use Case Diagram - Group Subsystem 169
2.5.5 Use Case Diagram - Warehouse Subsystem 170
2.5.6 Use Case Diagram – Vendor Subsystem 171
2.5.7 Use Case Diagram – Project Management Subsystem 172
2.5.8 Use Case Diagram – K9 Subsystem 172
2.5.9 Use Case Diagram – Rhino Subsystem 173
2.5.10 Use Case Diagram – Elephant Subsystem 174
2.5.11 Use Case Diagram – Message Subsystem 175
2.5.12 Use Case Diagram – Report Subsystem 176
2.5.13 Use Case Diagram – Machine Learning Subsystem 177
2.5.14 Use Case Diagram – System Subsystem 178
2.5.15 Use Case Diagram – Budget Subsystem 179
2.6 Logical Entity Relationship Diagram 180
2.7 Decomposition Diagram 181
2.8 Logical Use Case Narratives 185
2.8.1 User Subsystem Narratives 185
2.8.2 Admin Subsystem Narratives 202
2.8.3 Inventory Subsystem Narratives 234
2.8.4 Group Subsystem Narratives 254
2.8.5 Warehouse Management Subsystem Narratives 270
2.8.6 Vendor Subsystem Narratives 280
2.8.7 Project Subsystem Narratives 303
2.8.8 K9 Subsystem Narratives 352

2.8.9 Rhino Subsystem Narratives 382
2.8.10 Elephant Subsystem Narratives 392
2.8.11 Messages Subsystem Narratives 403
2.8.12 Report Subsystem Narratives 407
2.8.13 Machine Learning Subsystem 425
2.8.14 System Subsystem Narratives 427
2.8.15 Budget Subsystem Narratives 433
2.9 Primitive Data Flow Diagrams 449
2.9.1 User Subsystems Primitives 449
2.9.2 Admin Subsystem Primitives 458
2.9.3 Inventory Subsystem Primitives 474
2.9.4 Group Subsystem Primitives 484
2.9.5 Warehouse Subsystem Primitives 492
2.9.6 Vendor Subsystem Primitives 497
2.9.7 Project Subsystem Primitives 507
2.9.8 K9 Subsystem Primitives 532
2.9.9 Rhino Subsystem Primitives 548
2.9.10 Elephant Subsystem Primitives 553
2.9.11 Message Subsystem Primitives 558
2.9.12 Report Subsystem Primitives 560
2.9.13 Machine Learning Subsystem Primitives 568
1.9.14 System Subsystem Primitives 569
2.9.15 Budget Subsystem Primitives 572
2.10 Activity Diagrams 579
2.10.1 User Subsystem Activity Diagrams 579
2.10.2 Admin Subsystem Activity Diagrams 587
2.10.3 Inventory Subsystem Activity Diagrams 603
2.10.4 Group Subsystem Activity Diagrams 613
2.10.5 Warehouse Subsystem Activity Diagrams 621
2.10.6 Vendor Subsystem Activity Diagrams 626
2.10.7 Project Management Subsystem Activity Diagrams 637
2.10.8 K9 Subsystem Activity Diagrams 661
2.10.9 Rhino Subsystem Activity Diagrams 677

2.10.10 Elephant Subsystem Activity Diagrams 682
2.10.11 Message Subsystem Activity Diagrams 687
2.10.12 Report Subsystem Activity Diagrams 689
2.10.13 Machine Learning Subsystem Activity Diagrams 698
2.10.14 System Subsystem Activity Diagrams 699
2.10.15 Budget Subsystem Activity Diagrams 702
2.11 Context Diagrams 710
2.11.1 User Subsystem Context Diagram 710
2.11.2 Admin Subsystem Context Diagram 711
2.11.3 Inventory Subsystem Context Diagram 712
2.11.4 Group Subsystem Context Diagram 713
2.11.5 Warehouse Subsystem Context Diagram 714
2.11.6 Vendor Subsystem Context Diagram 715
2.11.7 Project Management Subsystem Context Diagram 716
2.11.8 K9 Subsystem Context Diagram 717
2.11.9 Rhino Subsystem Context Diagram 718
2.11.10 Elephant Subsystem Context Diagram 719
2.11.11 Messages Subsystem Context Diagram 720
2.11.12 Report Subsystem Context Diagram 721
2.11.13 Machine Learning Subsystem Context Diagram 722
2.11.14 System Subsystem Context Diagram 723
2.11.15 Budget Subsystem Context Diagram 724
2.13 Validation of Functional Specifications Against the Business Requirements 726
2.13.1 User Subsystem 726
2.13.2 Admin Subsystem 726
2.13.3 Inventory Subsystem 727
2.13.4 Group Subsystem 727
2.13.5 Warehouse Subsystem 727
2.13.6 Vendor Subsystem 728
2.13.7 Project Management Subsystem 728
2.13.8 K9 Subsystem 729
2.13.9 Rhino Subsystem 729
2.13.10 Elephant Subsystem 730

2.13.11 Message Subsystem 730
2.13.12 Report Subsystem 730
2.13.13 Machine Learning Subsystem 731
2.13.14 System Subsystem 731
2.13.15 Budget Subsystem 731
2.14 Conclusion 732
3. Technical Design 733
3.1 Introduction 733
3.2 Proposed Design Principles 734
3.3 Example Wireframes 736
3.3.1 Sign-In 736
3.3.2 Navbar 737
3.3.3 Rhino CRUD 738
3.3.4 Report 747
3.3.5 Messaging 748
3.3.6 Calendar 749
3.3.7 Dashboard 750
3.4 Technical Specifications 751
3.4.1 Introduction 751
3.4.2 ASP.NET Core Web API (C#) 751
3.4.3 Angular Framework 753
3.4.4 Microsoft SQL Server (MSSQL) 754
3.4.5 JSON Web Tokens (JWT) for Authentication & Authorisation 755
3.4.6 Secure Password Hashing 755
3.4.7 Heatmap.js for Geographic Visualisations 756
3.4.8 Microsoft Azure for Cloud Hosting & Deployment 756
3.4.9 SendGrid (Email Notification Service) 757
3.4.10 scikit-learn (Artificial Intelligence and Predictive Analytics) 758
3.4.11 Google Maps Platform API 759
3.4.12 Conclusion 760
3.5 Complexity Matrix 761
3.6 Conclusion 764
5. Sign-off by Client 765

5. Sign-off by Team..... 766

Table of Figures

Figure 1: User Subsystem - Use Case Diagram 166
Figure 2: Admin Subsystem - Use Case Diagram 167
Figure 3: Inventory Subsystem - Use Case Diagram 168
Figure 4:Group Subsystem - Use Case Diagram 169
Figure 5: Warehouse Subsystem - Use Case Diagram 170
Figure 6:Vendor Subsystem - Use Case Diagram 171
Figure 7: Vendor Management Subsystem - Use Case Diagram 172
Figure 8: K9 Subsystem - Use Case Diagram 172
Figure 9: Rhino Subsystem - Use Case Diagram 173
Figure 10: Elephant Subsystem - Use Case Diagram 174
Figure 11: Message Subsystem - Use Case Diagram 175
Figure 12: Report Subsystem - Use Case Diagram 176
Figure 13: Machine Learning Subsystem - Use Case Diagram 177
Figure 14: System Subsystem - Use Case Diagram 178
Figure 15: Budget Subsystem - Use Case Diagram 179
Figure 16: Entity Relationship Diagram 180
Figure 17 Decomposition Diagram 181
Figure 18: 1.1 User Login - Primitive Data Flow Diagram 449
Figure 19: 1.2 User Logout - Primitive Data Flow Diagram 450
Figure 20: 1.3 Reset Password - Primitive Data Flow Diagram 451
Figure 21: 1.4 Create User - Primitive Data Flow Diagram 452
Figure 22: 1.5 Update User - Primitive Data Flow Diagram 453
Figure 23: 1.6 View User - Primitive Data Flow Diagram 454
Figure 24: 1.7 Reactivate User - Primitive Data Flow Diagram 455
Figure 25: 1.8 Deactivate User - Primitive Data Flow Diagram 456
Figure 26: 1.9 View FAQ - Primitive Data Flow Diagram 457
Figure 27: 2.1 Create User Profile - Primitive Data Flow Diagram 458
Figure 28: 2.2 Update User Primitive Data Flow Diagram 459
Figure 30 2.3 View User Profile - Primitive Data Flow Diagram 460
Figure 31 2.4 Search User Profile - Primitive Data Flow Diagram 461
Figure 32 2.5 Add FAQ - Primitive Data Flow Diagram 462
Figure 33 2.6 Update FAQ - Primitive Data Flow Diagram 463
Figure 34 2.7 Remove FAQ - Primitive Data Flow Diagram 464
Figure 35 2.8 Create Role - Primitive Data Flow Diagram 465
Figure 36 2.9 View Role - Primitive Data Flow Diagram 466
Figure 37 2.10 Update Role - Primitive Data Flow Diagram 467
Figure 38 2.11 Reactivate Role - Primitive Data Flow Diagram 468
Figure 39 2.12 Deactivate Role - Primitive Data Flow Diagram 469
Figure 40 2.13 Create Role Permission - Primitive Data Flow Diagram 470
Figure 41 2.14 Update Role Permission - Primitive Data Flow Diagram 471
Figure 42 2.15 View Role Permission - Primitive Data Flow Diagram 472

Figure 43 2.16 Deactivate Role Permission - Primitive Data Flow Diagram 473
Figure 44 3.1 Receive Inventory Item - Primitive Data Flow Diagram 474
Figure 45 3.2 View Inventory - Primitive Data Flow Diagram 475
Figure 46 3.3 Write-Off Inventory - Primitive Data Flow Diagram 476
Figure 47 3.4 Inventory Stock Take - Primitive Data Flow Diagram 477
Figure 48 3.5 Allocate Inventory - Primitive Data Flow Diagram 478
Figure 49 3.6 Create Inventory Item Type - Primitive Data Flow Diagram 479
Figure 50 3.7 View Inventory Item Type - Primitive Data Flow Diagram 480
Figure 51 3.8 View Inventory Item Types - Primitive Data Flow Diagram 481
Figure 52 3.9 Update Inventory Item Type - Primitive Data Flow Diagram 482
Figure 53 3.10 Deactivate Inventory Item Type - Primitive Data Flow Diagram 483
Figure 54 4.1 Create Group - Primitive Data Flow Diagram 484
Figure 55 4.2 View Groups - Primitive Data Flow Diagram 485
Figure 56 4.3 Update Group - Primitive Data Flow Diagram 486
Figure 57 4.4 Deactivate Group - Primitive Data Flow Diagram 487
Figure 58 4.5 Create Skill - Primitive Data Flow Diagram 488
Figure 59 4.6 View Skills - Primitive Data Flow Diagram 489
Figure 60 4.7 Update Skill - Primitive Data Flow Diagram 490
Figure 61 4.8 Deactivate Skill - Primitive Data Flow Diagram 491
Figure 62 5.1 Create Warehouse - Primitive Data Flow Diagram 492
Figure 63 5.2 View Warehouse - Primitive Data Flow Diagram 493
Figure 64 5.3 Update Warehouse - Primitive Data Flow Diagram 494
Figure 65 5.4 Reactivate Warehouse - Primitive Data Flow Diagram 495
Figure 66 5.5 Deactivate Warehouse - Primitive Data Flow Diagram 496
Figure 67 6.1 Create Vendor - Primitive Data Flow Diagram 497
Figure 68 6.2 View Vendor - Primitive Data Flow Diagram 498
Figure 69 6.3 View Vendors - Primitive Data Flow Diagram 499
Figure 70 6.4 Edit Vendor - Primitive Data Flow Diagram 500
Figure 71 6.5 Reactivate Vendor - Primitive Data Flow Diagram 501
Figure 72 6.6 Deactivate Vendor - Primitive Data Flow Diagram 502
Figure 73 6.7 Create Vendor Type - Primitive Data Flow Diagram 503
Figure 74 6.8 Update Vendor Type - Primitive Data Flow Diagram 504
Figure 75 6.9 View Vendor Types - Primitive Data Flow Diagram 505
Figure 76 6.10 Reactivate Vendor Type - Primitive Data Flow Diagram 506
Figure 77 6.11 Deactivate Vendor Type - Primitive Data Flow Diagram 507
Figure 78 7.1 Add Project - Primitive Data Flow Diagram 508
Figure 799 7.2 View Project - Primitive Data Flow Diagram 509
Figure 808 7.11 View Indemnity Form - Primitive Data Flow Diagram 518
Figure 81 8.1 Add Dog - Primitive Data Flow Diagram 532
Figure 82 8.2 View Dog - Primitive Data Flow Diagram 533
Figure 83 8.3 View Dogs - Primitive Data Flow Diagram 534
Figure 84 8.4 Update Dog - Primitive Data Flow Diagram 535
Figure 85 8.5 Deactivate Dog - Primitive Data Flow Diagram 536
Figure 86 8.6 Assign User to Dog - Primitive Data Flow Diagram 537

Figure 87 8.7 View User-Dog Assignment - Primitive Data Flow Diagram 538
Figure 88 8.8 Update User-Dog Assignment - Primitive Data Flow Diagram 539
Figure 89 8.9 Reactivate User-Dog Assignment - Primitive Data Flow Diagram 540
Figure 90 8.10 Deactivate User-Dog Assignment - Primitive Data Flow Diagram 541
Figure 91 8.11 Add K9 Role - Primitive Data Flow Diagram 542
Figure 92 8.12 Update K9 Role - Primitive Data Flow Diagram 543
Figure 93 8.13 View K9 Role - Primitive Data Flow Diagram 544
Figure 94 8.14 Reactivate K9 Role - Primitive Data Flow Diagram 545
Figure 95 8.15 Deactivate Role - Primitive Data Flow Diagram 546
Figure 96 8.16 View K9 Ancestry - Primitive Data Flow Diagram 547
Figure 97 9.1 Add Rhino - Primitive Data Flow Diagram 548
Figure 98 9.2 View Rhino - Primitive Data Flow Diagram 549
Figure 99 9.3 View Rhinos - Primitive Data Flow Diagram 550
Figure 100 9.4 Update Rhino - Primitive Data Flow Diagram 551
Figure 101 9.5 Deactivate Rhino - Primitive Data Flow Diagram 552
Figure 102 10.1 Add Elephant - Primitive Data Flow Diagram 553
Figure 103 10.2 View Elephant - Primitive Data Flow Diagram 554
Figure 104 10.3 View Elephants - Primitive Data Flow Diagram 555
Figure 105 10.4 Update Elephant - Primitive Data Flow Diagram 556
Figure 106 10.5 Deactivate Elephant - Primitive Data Flow Diagram 557
Figure 107: 12.1 Generate Custom Report - Primitive Data Flow Diagram 560
Figure 108: 12.2 View Custom Report - Primitive Data Flow Diagram 561
Figure 109: 12.3 Update Custom Report - Primitive Data Flow Diagram 562
Figure 110: 12.4 Remove Custom Report - Primitive Data Flow Diagram 563
Figure 111: 12.5 View Bike Impact Report - Primitive Data Flow Diagram 564
Figure 112: 12.6 View Budget Report - Primitive Data Flow Diagram 565
Figure 113: 12.7 View Rhino Report - Primitive Data Flow Diagram 566
Figure 114: 12.8 View Elephant Report - Primitive Data Flow Diagram 567
Figure 115: 13.1 View Audit Log - Primitive Data Flow Diagram 568
Figure 116: 14.1 Backup System - Primitive Data Flow Diagram 569
Figure 117: 14.2 Restore System - Primitive Data Flow Diagram 570
Figure 118: 14.3 View Audit Log - Primitive Data Flow Diagram 571
Figure 119 15.1 Create Project Budget - Primitive Data Flow Diagram 572
Figure 120 15.2 View Project Budget - Primitive Data Flow Diagram 573
Figure 121 15.3 View Project Budgets - Primitive Data Flow Diagram 573
Figure 122 15.4 Update Project Budget - Primitive Data Flow Diagram 574
Figure 123 15.5 Log An Expense - Primitive Data Flow Diagram 575
Figure 124 15.6 Update Expense - Primitive Data Flow Diagram 576
Figure 125 15.7 View Expense - Primitive Data Flow Diagram 577
Figure 126 15.8 Remove Expense - Primitive Data Flow Diagram 578
Figure 127 1.1 User Login - Activity Diagram 579
Figure 128 1.2 User Logout - Activity Diagram 580
Figure 129 1.3 Reset Password - Activity Diagram 581
Figure 130 1.4 Create User - Activity Diagram 582

Figure 131 1.5 Update User - Activity Diagram 583
Figure 132 1.6 View User - Activity Diagram 584
Figure 133 1.7 Reactivate User - Activity Diagram 585
Figure 134 1.8 Deactivate User - Activity Diagram 586
Figure 135 2.1 Create User Profile - Activity Diagram 587
Figure 136 2.2 Update User Profile - Activity Diagram 588
Figure 137 2.3 View User Profile - Activity Diagram 589
Figure 138 2.4 Deactivate User Profile - Activity Diagram 590
Figure 139 2.5 Add FAQ - Activity Diagram 591
Figure 140 2.6 Update FAQ - Activity Diagram 592
Figure 141 2.7 Remove FAQ - Activity Diagram 593
Figure 142 2.8 Create Role - Activity Diagram 594
Figure 143 2.9 View Role - Activity Diagram 595
Figure 144 2.10 Update Role - Activity Diagram 596
Figure 145 2.11 Reactivate Role - Activity Diagram 597
Figure 146 2.12 Deactivate Role - Activity Diagram 598
Figure 147 2.13 Create Role Permission - Activity Diagram 599
Figure 148 2.14 Update Role Permission - Activity Diagram 600
Figure 149 2.15 View Role Permission - Activity Diagram 601
Figure 150 2.16 Deactivate Role Permission - Activity Diagram 602
Figure 151 3.1 Receive Inventory Item - Activity Diagram 603
Figure 152 3.2 View Inventory - Activity Diagram 604
Figure 153 3.3 Write Off Inventory - Activity Diagram 605
Figure 154 3.4 Inventory Stock Take - Activity Diagram 606
Figure 155 3.5 Allocate Inventory - Activity Diagram 607
Figure 156 3.6 Create Inventory Item Type - Activity Diagram 608
Figure 157 3.7 View Inventory Item Type - Activity Diagram 609
Figure 158 3.8 View Inventory Item Types - Activity Diagram 610
Figure 159 3.9 Update Inventory Item Type - Activity Diagram 611
Figure 160 3.10 Deactivate Inventory Item Type - Activity Diagram 612
Figure 161 4.1 Create Group - Activity Diagram 613
Figure 162 4.2 View Group - Activity Diagram 614
Figure 163 4.3 Update Group - Activity Diagram 615
Figure 164 4.4 Deactivate Group - Activity Diagram 616
Figure 165 4.5 Create Skill - Activity Diagram 617
Figure 166 4.6 View Skills - Activity Diagram 618
Figure 167 4.7 Update Skill - Activity Diagram 619
Figure 168 4.8 Deactivate Skill - Activity Diagram 620
Figure 169 5.1 Create Warehouse - Activity Diagram 621
Figure 170 5.2 View Warehouse - Activity Diagram1 622
Figure 171 5.3 Update Warehouse - Activity Diagram 623
Figure 172 5.4 Reactivate Warehouse - Activity Diagram 624
Figure 173 5.5 Deactivate Warehouse - Activity Diagram 625
Figure 174 6.1 Create Vendor - Activity Diagram 626

Figure 175 6.2 View Vendor - Activity Diagram 627
Figure 176 6.3 View Vendors - Activity Diagram 628
Figure 177 6.4 Edit Vendor - Activity Diagram 629
Figure 178 6.5 Reactivate Vendor - Activity Diagram 630
Figure 179 6.6 Deactivate Vendor - Activity Diagram 631
Figure 180 6.7 Create Vendor Type - Activity Diagram 632
Figure 181 6.8 Update Vendor Type - Activity Diagram 633
Figure 182 6.9 View Vendor Type - Activity Diagram 634
Figure 183 6.10 Reactivate Vendor Type - Activity Diagram 635
Figure 184 6.11 Deactivate Vendor Type - Activity Diagram 636
Figure 185 8.5 Deactivate Dog - Activity Diagram 665
Figure 186 8.6 Assign User to Dog - Activity Diagram 666
Figure 187 8.7 View User-Dog Assignment - Activity Diagram 667
Figure 188 8.8 Update User-Dog Assigment - Activity Diagram 668
Figure 189 8.9 Reactivate User-Dog Assignment - Activity Diagram 669
Figure 190 8.10 Deactivate User-Dog Assignment - Activity Diagram 670
Figure 191 8.11 Add K9 Role - Activity Diagram 671
Figure 192 8.12 Update K9 Role - Activity Diagram 672
Figure 193 8.13 View K9 Role - Activity Diagram 673
Figure 194 8.14 Reactivate K9 Role - Activity Diagram 674
Figure 195 8.15 Deactivate K9 Role - Activity Diagram 675
Figure 196 8.16 View K9 Ancestry - Activity Diagram 676
Figure 197 9.1 Add Rhino - Activity Diagram 677
Figure 198 9.2 View Rhino - Activity Diagram 678
Figure 199 9.3 View Rhinos - Activity Diagram 679
Figure 200 9.4 Update Rhino - Activity Diagram 680
Figure 201 9.5 Deactivate Rhino - Activity Diagram 681
Figure 202 10.1 Add Elephant - Activity Diagram 682
Figure 203 10.2 View Elephant - Activity Diagram 683
Figure 204 10.3 View Elephants - Activity Diagram 684
Figure 205 10.4 Update Elephant - Activity Diagram 685
Figure 206 10.5 Deactivate Elephant - Activity Diagram 686
Figure 207 11.1 Create Message - Activity Diagram 687
Figure 208 11.2 View Messages - Activity Diagram 688
Figure 209 12.1 Generate Custom Report - Activity Diagram 689
Figure 210 12.2 View Custom Report - Activity Diagram 690
Figure 211 12.3 Update Custom Report - Activity Diagram 691
Figure 212 12.4 Remove Custom Report - Activity Diagram 692
Figure 213 12.5 View Bike Impact Report - Activity Diagram 693
Figure 214 12.6 View Budget Report - Activity Diagram 694
Figure 215 View Rhino Report - Activity Diagram 695
Figure 216 12.8 View Elephant Report - Activity Diagram 696
Figure 217 12.9 View K9 Report - Activity Diagram 697
Figure 218 13.1 View Machine Learning Dashboard - Activity Diagram 698

Figure 219 14.1 Backup System - Activity Diagram 699
Figure 220 14.2 Restore System - Activity Diagram 700
Figure 221 14.3 View Audit Log - Activity Diagram 701
Figure 222 15.1 Create Project Budget - Activity Diagram 702
Figure 223 15.2 View Project Budget - Activity Diagram 703
Figure 224 15.3 View Project Budgets - Activity Diagram 704
Figure 225 15.4 Update Project Budget - Activity Diagram 705
Figure 226 15.5 Log An Expense - Activity Diagram 706
Figure 227 15.6 Update An Expense - Activity Diagram 707
Figure 228 15.7 View Expense - Activity Diagram 708
Figure 229 15.8 Remove Expense - Activity Diagram 709
Figure 230: User Subsystem Context Diagram 710
Figure 231 Admin Subsystem Context Diagram 711
Figure 232: Inventory Subsystem Context Diagram 712
Figure 233: Group Subsystem Context Diagram 713
Figure 234 Warehouse Subsystem Context Diagram 714
Figure 235: Vendor Subsystem Context Diagram 715
Figure 236: Project Management Subsystem Context Diagram 716
Figure 237: K9 Subsystem Context Diagram 717
Figure 238: Rhino Subsystem Context Diagram 718
Figure 239: Elephant Subsystem Context Diagram 719
Figure 240: Messages Subsystem Context Diagram 720
Figure 241: Report Subsystem Context Diagram 721
Figure 242: Machine Learning Subsystem Context Diagram 722
Figure 243: System Subsystem Context Diagram 723
Figure 244: Budget Subsystem Context Diagram 724
Figure 245 Class Diagram 725
Figure 246 Sign-In Screen 736
Figure 247: Navbar 737
Figure 248 Rhino View Screen 738
Figure 249 Add Rhino Form 740
Figure 250 View Rhino Screen 742
Figure 251 Update Rhino Form 744
Figure 252 Success Message 746
Figure 253 Reports Screen 747
Figure 254 Messaging Screen 748
Figure 255 Calendar Screen 749
Figure 256 Dashboard Screen 750
Figure 257 ASP.NET Core Architecture 752
Figure 258 Angular Architecture 754

Table of Tables

Table 1: 1.1 User Login - Description 28
Table 2: 1.2 User Logout Description 29
Table 3: 1.3 Reset Password – Description 30
Table 4: 1.4 Create User - Description 31
Table 5: 1.5 Update User - Description 32
Table 6: 1.6 View User - Description 33
Table 7: 1.7 Reactivate User - Description 34
Table 8: 1.8 Deactivate User - Description 35
Table 9: 1.9 View FAQ - Description 36
Table 10: 2.1 Create User Profile - Description 37
Table 11: 2.2 Update User Profile - Description 38
Table 12: 2.3 View User Profile - Description 39
Table 13: 2.4 Deactivate User Profile - Description 40
Table 14: 2.5 Add FAQ - Description 41
Table 15: 2.6 Update FAQ - Description 42
Table 16: 2.7 Remove FAQ - Description 43
Table 17: 2.8 Create Role - Description 44
Table 18: 2.9 View Role - Description 45
Table 19: 2.10 Update Role - Description 46
Table 20:2.11 Reactivate Role - Description 47
Table 21: 2.12 Deactivate Role - Description 48
Table 22: 2.13 Create Role Permission - Description 49
Table 23: 2.14 Update Role Permission - Description 50
Table 24: 2.15 View Role Permission - Description 51
Table 25: 2.16 Deactivate Role Permission - Description 52
Table 26: 3.1 Receive Inventory Item - Description 53
Table 27: 3.2 View Inventory - Description 54
Table 28: 3.3 Write-Off Inventory - Description 55
Table 29: 3.4 Inventory Stock Take - Description 56
Table 30: 3.5 Allocate Inventory - Description 57
Table 31: 3.6 Create Inventory Item Type - Description 58
Table 32: 3.7 View Inventory Item Type - Description 59
Table 33:3.8 View Inventory Item Types - Description 60
Table 34: 3.9 Update Inventory Item Type - Description 61
Table 35: 3.10 Deactivate Inventory Item Type - Description 62
Table 36: 4.1 Create Group – Description 63
Table 37:4.2 View Group – Description 64
Table 38: 4.3 Update Group – Description 65
Table 39: 4.4 Deactivate Group - Description 66
Table 40: 4.5 Create Skill - Description 67
Table 41: 4.6 View Skill - Description 68
Table 42: 4.7 Update Skill - Description 69

Table 43: 4.8 Deactivate Skill - Description 70
Table 44 :5.1 Create Warehouse - Description 71
Table 45: 5.2 View Warehouse - Description 72
Table 46: 5.3 Update Warehouse - Description 73
Table 47 5.4: Reactivate Warehouse - Description 74
Table 48: 5.5 Deactivate Warehouse - Description 75
Table 49: 6.1 Create Vendor - Description 76
Table 50 6.2: View Vendor - Description 77
Table 51: 6.3 View Vendors - Description 78
Table 52: 6.4 Edit Vendor - Description 79
Table 53: 6.5 Reactivate Vendor - Description 80
Table 54: 6.6 Deactivate Vendor - Description 81
Table 55: 6.7 Create Vendor Type - Description 82
Table 56: 6.8 Update Vendor Type - Description 83
Table 57: 6.8 View Vendor Types - Description 84
Table 58: 6.10 Reactivate Vendor Type - Description 85
Table 59: 6.11 Deactivate Vendor Type - Description 86
Table 60: 7.1 Create Project - Description 87
Table 61: 7.2 View Project - Description 88
Table 62: 7.3 Update Project - Description 89
Table 63: 7.4 Deactivate Project - Description 90
Table 64: 7.5 Create Project Type - Description 91
Table 65: 7.6 View Project Type - Description 92
Table 66: 7.7 Update Project Type - Description 93
Table 67: 7.8 Deactivate Project Type - Description 94
Table 68: 7.9 Add Indemnity Form - Description 95
Table 69: 7.10 Update Indemnity Form - Description 96
Table 70: 7.11 View Indemnity Form - Description 97
Table 71: 7.12 Remove Indemnity Form - Description 98
Table 72: 7.13 Capture Indemnity Form - Description 99
Table 73: 7.14 Capture Project Feedback - Description 100
Table 74: 7.15 View Project Feedback - Description 101
Table 75: 7.16 Create Impact Feedback Form - Description 102
Table 76: 7.17 View Impact Forms - Description 103
Table 77: 7.18 Update Impact Form - Description 104
Table 78: 7.19 Deactivate Impact Form - Description 105
Table 79: 7.20 Capture Impact Form - Description 106
Table 80: 7.21 View Impact Feedback - Description 107
Table 81: 7.22 Add Photo - Description 108
Table 82: 7.23 View Photo - Description 109
Table 83: 7.24 Delete Photo - Description 110
Table 84: 8.1 Add Dog - Description 111
Table 85: 8.2 View Dog - Description 112
Table 86: 8.3 View All Dogs- Description 113

Table 87: 8.4 Update Dog - Description 114
Table 88: 8.5 Deactivate Dog - Description 115
Table 89: 8.6 Assign User to Dog - Description 116
Table 90: 8.7 View User-Dog Assignment - Description 117
Table 91: 8.8 Update User-Dog Assignment - Description 118
Table 92: 8.9 Reactivate User-Dog Assignment - Description 119
Table 93: 8.10 Deactivate User-Dog Assignment - Description 120
Table 94: 8.11 Add K9 Role - Description 121
Table 95: 8.12 Update K9 Role - Description 122
Table 96: 8.13 View K9 Role - Description 123
Table 97: 8.14 Reactivate K9 Role - Description 124
Table 98: 8.15 Deactivate K9 Role - Description 125
Table 99: 8.16 View K9 Ancestry - Description 126
Table 100: 9.1 Add Rhino - Description 127
Table 101:9.2 View Rhino - Description 128
Table 102: 9.3 View Rhinos - Description 129
Table 103: 9.4 Update Rhino - Description 130
Table 104: 9.5 9.5 Deactivate Rhino - Description 131
Table 105: 10.1 Add Elephant - Description 132
Table 106: 10.2 View Elephant - Description 133
Table 107: 10.3 View Elephants - Description 134
Table 108: 10.4 Update Elephant - Description 135
Table 109: 10.5 Deactivate Elephant - Description 136
Table 110: 11.1 Create Message - Description 137
Table 111: 11.2 View Message - Description 138
Table 112: 12.1 Generate Custom Report - Description 139
Table 113: 12.2 View Custom Report - Description 140
Table 114: 12.3 Update Custom Report - Description 141
Table 115: 12.4 Remove Custom Report - Description 142
Table 116: 12.5 View Bike Impact Report - Description 143
Table 117: 12.6 View Budget Report - Description 144
Table 118: 12.7 View Rhino Report - Description 145
Table 119: 12.8 View Elephant Report - Description 146
Table 120: 12.9 View K9 Report - Description 147
Table 121: 13.1 View Machine Learning Dashboard - Description 148
Table 122: 14.1 Backup System - Description 149
Table 123: 14.2 Restore System - Description 150
Table 124: 14.3 View Audit Log - Description 151
Table 125: 15.1 Create Project Budget - Description 152
Table 126: 15.2 View Project Budget - Description 153
Table 127: 15.3 View Project Budgets - Description 154
Table 128: 15.4 Update Project Budget - Description 155
Table 129: 15.5 Log An Expense - Description 156
Table 130: 15.6 Update An Expense - Description 157

Table 131: 15.7 View An Expense - Description 158
Table 132: 15.8 Remove An Expense - Description 159
Table 133: Functional Requirements List Modifications 165
Table 134: 1.1 User Login - Logical Use Case Narrative 186
Table 135: 1.2 User Logout - Logical Use Case Narrative 187
Table 136: 1.3 Reset Password - Logical Use Case Narrative 189
Table 137: 1.4 Create User - Logical Use Case Narrative 191
Table 138: 1.5 Update User - Logical Use Case Narrative 193
Table 139: 1.6 View User - Logical Use Case Narrative 195
Table 140: 1.7 Reactivate User - Logical Use Case Narrative 197
Table 141: 1.8 Deactivate User - Logical Use Case Narrative 199
Table 142: 1.9 View FAQ - Logical Use Case Narrative 201
Table 143: 2.1 Create User Profile - Logical Use Case Narrative 203
Table 144 2.2 Update User Profile - Logical Use Case Narrative 205
Table 145: 2.3 View User Profile - Logical Use Case Narrative 207
Table 146: 2.4 Deactivate User Profile - Logical Use Case Narrative 209
Table 147: 2.5 Add FAQ - Logical Use Case Narrative 211
Table 148: 2.6 Update FAQ - Logical Use Case Narrative 213
Table 149: 2.7 Remove FAQ - Logical Use Case Narrative 215
Table 150: 2.8 Create Role - Logical Use Case Narrative 217
Table 151: 2.9 View Role - Logical Use Case Narrative 219
Table 152: 2.10 Update Role - Logical Use Case Narrative 221
Table 153: 2.11 Reactivate Role - Logical Use Case Narrative 223
Table 154: 2.12 Deactivate Role - Logical Use Case Narrative 225
Table 155: 2.13 Create Role Permission - Logical Use Case Narrative 227
Table 156: 2.14 Update Role Permission - Logical Use Case Narrative 229
Table 157: 2.15 View Role Permission - Logical Use Case Narrative 231
`Table 158: 2.16 Deactivate Role Permission - Logical Use Case Narrative 233
Table 159: 3.1 Receive Inventory Item - Logical Use Case Narrative 235
Table 160: 3.2 View Inventory - Logical Use Case Narrative 237
Table 161: 3.3 Write-Off Inventory - Logical Use Case Narrative 239
Table 162: 3.4 Inventory Stock Take - Logical Use Case Narrative 241
Table 163: 3.5 Allocate Inventory - Logical Use Case Narrative 243
Table 164: 3.6 Create Inventory Item Type - Logical Use Case Narrative 245
Table 165: 3.7 View Inventory Item Type - Logical Use Case Narrative 247
Table 166: 3.8 View Inventory Item Types - Logical Use Case Narrative 249
Table 167: 3.9 Update Inventory Item Type - Logical Use Case Narrative 251
Table 168: 3.10 Deactivate Inventory Item Type - Logical Use Case Narrative 253
Table 169: 4.1 Create Group - Logical Use Case Narrative 255
Table 170: 4.2 View Groups - Logical Use Case Narrative 257
Table 171: 4.3 Update Group - Logical Use Case Narrative 259
Table 172: 4.4 Deactivate Group - Logical Use Case Narrative 261
Table 173: 4.5 Create Skill - Logical Use Case Narrative 263
Table 174: 4.6 View Skills - Logical Use Case Narrative 265

Table 175: 4.7 Update Skill - Logical Use Case Narrative 267
Table 176: 4.8 Deactivate Skill - Logical Use Case Narrative 269
Table 177: 5.1 Create Warehouse - Logical Use Case Narrative 271
Table 178: 5.2 View Warehouse - Logical Use Case Narrative 273
Table 179: 5.3 Update Warehouse - Logical Use Case Narrative 275
Table 180: 5.4 Reactivate Warehouse - Logical Use Case Narrative 277
Table 181: 5.5 Deactivate Warehouse - Logical Use Case Narrative 279
Table 182: 6.1 Create Vendor - Logical Use Case Narrative 282
Table 183: 6.2 Create Vendor - Logical Use Case Narrative 284
Table 184: 6.3 Create Vendor - Logical Use Case Narrative 286
Table 185: 6.4 Edit Vendor - Logical Use Case Narrative 288
Table 186: 6.5 Reactivate Vendor - Logical Use Case Narrative 290
Table 187: 6.6 Deactivate Vendor - Logical Use Case Narrative 292
Table 188: 6.7 Create Vendor Type - Logical Use Case Narrative 294
Table 189: 6.8 Update Vendor Types - Logical Use Case Narrative 296
Table 190: 6.9 View Vendor Types - Logical Use Case Narrative 298
Table 191: 6.10 Reactivate Vendor Type - Logical Use Case Narrative 300
Table 192 :6.11 Deactivate Vendor Type - Logical Use Case Narrative 302
Table 193: 7.1 Create Project - Logical Use Case Narrative 305
Table 194: 7.2 View Project - Logical Use Case Narrative 307
Table 195: 7.3 Update Project - Logical Use Case Narrative 310
Table 196: 7.4 Deactivate Project - Logical Use Case Narrative 312
Table 197: 7.5 Create Project Type - Logical Use Case Narrative 314
Table 198: 7.6 View Project Type - Logical Use Case Narrative 316
Table 199: 7.7 Update Project Type - Logical Use Case Narrative 318
Table 200: 7.8 Deactivate Project Type - Logical Use Case Narrative 320
Table 201: 7.9 Add Indemnity Form - Logical Use Case Narrative 322
Table 202: 7.10 Update Indemnity Form - Logical Use Case Narrative 324
Table 203: 7.11 View Indemnity Form - Logical Use Case Narrative 326
Table 204: 7.12 Remove Indemnity Form - Logical Use Case Narrative 328
Table 205: 7.13 Remove Indemnity Form - Logical Use Case Narrative 330
Table 206: 7.14 Capture Project Feedback - Logical Use Case Narrative 332
Table 207: 7.15 View Project Feedback - Logical Use Case Narrative 333
Table 208: 7.16 Create Impact Feedback Form - Logical Use Case Narrative 335
Table 209: 7.17 View Impact Forms - Logical Use Case Narrative 337
Table 210: 7.18 Update Impact Form - Logical Use Case Narrative 339
Table 211: 7.19 Deactivate Impact Form - Logical Use Case Narrative 341
Table 212: 7.20 Capture Impact Form - Logical Use Case Narrative 343
Table 213: 7.21 View Impact Feedback - Logical Use Case Narrative 345
Table 214: 7.22 Add Photo - Logical Use Case Narrative 347
Table 215: 7.23 View Photo - Logical Use Case Narrative 349
Table 216: 7.24 Delete Photo - Logical Use Case Narrative 351
Table 217: 8.1 Add Dog - Logical Use Case Narrative 353
Table 218: 8.2 View Dog - Logical Use Case Narrative 355

Table 219: 8.3 View Dogs - Logical Use Case Narrative 357
Table 220: 8.4 Update Dog - Logical Use Case Narrative 359
Table 221: 8.5 Deactivate Dog - Logical Use Case Narrative 361
Table 222: 8.6 Assign User to Dog - Logical Use Case Narrative 363
Table 223: 8.7 View User-Dog Assignment - Logical Use Case Narrative 365
Table 224: 8.8 Update User-Dog Assignment - Logical Use Case Narrative 367
Table 225: 8.9 Reactivate User-Dog Assignment - Logical Use Case Narrative 369
Table 226: 8.10 Deactivate User-Dog Assignment - Logical Use Case Narrative 371
Table 227: 8.11 Add K9 Role - Logical Use Case Narrative 373
Table 228: 8.12 Update K9 Role - Logical Use Case Narrative 375
Table 229: 8.13 View K9 Roles - Logical Use Case Narrative 377
Table 230: 8.14 Reactivate K9 Role - Logical Use Case Narrative 379
Table 231: 8.15 Deactivate K9 Role - Logical Use Case Narrative 381
Table 232: 9.1 Add Rhino - Logical Use Case Narrative 383
Table 233: 9.2 View Rhino - Logical Use Case Narrative 385
Table 234: 9.3 View Rhino - Logical Use Case Narrative 387
Table 235: 9.4 Update Rhino - Logical Use Case Narrative 389
Table 236: 9.5 Deactivate Rhino - Logical Use Case Narrative 391
Table 237: 10.1 Add Elephant - Logical Use Case Narrative 393
Table 238: 10.2 View Elephant - Logical Use Case Narrative 395
Table 239: 10.3 View Elephants - Logical Use Case Narrative 397
Table 240: 10.4 Update Elephant - Logical Use Case Narrative 400
Table 241: 10.5 Deactivate Elephant - Logical Use Case Narrative 402
Table 242: 11.1 Create Message - Logical Use Case Narrative 404
Table 243: 11.2 View Messages - Logical Use Case Narrative 406
Table 244: 12.1 Generate Custom Report - Logical Use Case Narrative 408
Table 245: 12.2 View Custom Report - Logical Use Case Narrative 410
Table 246: 12.3 Update Custom Report - Logical Use Case Narrative 412
Table 247: 12.4 Remove Custom Report - Logical Use Case Narrative 414
Table 248: 12.5 View Bike Impact Report - Logical Use Case Narrative 416
Table 249: 12.6 View Budget Report - Logical Use Case Narrative 418
Table 250: 12.7 View Rhino Report - Logical Use Case Narrative 420
Table 251: 12.8 View Elephant Report - Logical Use Case Narrative 422
Table 252: 12.9 View K9 Report - Logical Use Case Narrative 424
Table 253: 13.1 View Machine Learning Dashboard - Logical Use Case Narrative 426
Table 254: 14.1 Backup System - Logical Use Case Narrative 428
Table 255: 14.2 Restore System - Logical Use Case Narrative 430
Table 256: 14.3 View Audit Log - Logical Use Case Narrative 432
Table 257: 15.1 Create Project Budget - Logical Use Case Narrative 434
Table 258: 15.2 View Project Budget - Logical Use Case Narrative 436
Table 259: 15.3 View Project Budgets - Logical Use Case Narrative 438
Table 260: 15.4 Update Project Budget - Logical Use Case Narrative 440
Table 261: 15.5 Log An Expense UCD 442
Table 262: 15.6 Update An Expense - Logical Use Case Narrative 444

Table 263: 15.7 View Expense - Logical Use Case Narrative 446
Table 264: 15.8 Remove Expense UCD 448
Table 265: User Subsystem Validation 726
Table 266: Admin Subsystem Validation 726
Table 267: Inventory Subsystem Validation 727
Table 268: Group Subsystem Validation 727
Table 269: Warehouse Subsystem Validation 728
Table 270: Vendor Subsystem Validation 728
Table 271:Project Management Subsystem Validation 729
Table 272: K9 Subsystem Validation 729
Table 273: Rhino Subsystem Validation 730
Table 274: Elephant Subsystem Validation 730
Table 275: Message Subsystem Validation 730
Table 276: Report Subsystem Validation 730
Table 277: Machine Learning Subsystem Validation 731
Table 278: System Subsystem Validation 731
Table 279: Budget Subsystem Validation 731
Table 280 Design Principles Table 735
Table 281 Complexity Matrix 763

1. Document Introduction

This document contains a list of our updated functional requirements together with their descriptions, the use case diagram, the ERD, our functional decomposition diagram, logical use case narratives, logical context diagram, logical primitive level diagrams, activity diagrams, validation, UML class diagram, proposed designed principles, example wireframes, technical specifications, and an updated complexity matrix.

2. Logical Design

2.1 Introduction

The suggested system's logical design is presented in this section of the document.
Logical use case diagrams, table relationship diagram (ERD), functional decomposition diagram, and a new functional requirements list are all included. Narratives, the UML class diagram, activity diagrams, logical primitive level diagrams, logical context diagrams, and validation.

2.2 Updated Functional Requirements Lists

2.2.1 User Subsystem

2.2.1.1 User Login

2.2.1.2 User Logout

2.2.1.3 Reset Password

2.2.1.4 Create User

2.2.1.5 Update User

2.2.1.6 View User

2.2.1.7 Reactivate User

2.2.1.8 Deactivate User

2.2.1.9 View FAQ

2.2.2 Admin Subsystem

2.2.2.1 Create User Profile

2.2.2.2 Update User Profile

2.2.2.3 View User Profile

2.2.2.4 Deactivate User Profile

2.2.2.5 Add FAQ

2.2.2.6 Update FAQ

2.2.2.7 Remove FAQ

2.2.2.8 Create Role

2.2.2.9 View Role

2.2.2.10 Update Role

2.2.2.11 Reactivate Role

2.2.2.12 Deactivate Role

2.2.2.13 Create Role Permission

2.2.2.14 Update Role Permission

2.2.2.15 View Role Permission

2.2.2.16 Deactivate Role Permission

2.2.3 Inventory Subsystem

2.2.3.1 Receive Inventory Item

2.2.3.2 View Inventory

2.2.3.3 Write-Off Inventory

2.2.3.4 Inventory Stock Take

2.2.3.5 Allocate Inventory

2.2.3.6 Create Inventory Item Type

2.2.3.7 View Inventory Item Type

2.2.3.8 View Inventory Item Types

2.2.3.9 Update Inventory Item Type

2.2.3.10 Deactivate Inventory Item Type

2.2.4 Group Subsystem

2.2.4.1 Create Group

2.2.4.2 View Groups

2.2.4.3 Update Group

2.2.4.4 Deactivate Group

2.2.4.5 Create Skill

2.2.4.6 View Skills

2.2.4.7 Update Skill

2.2.4.8 Deactivate Skill

2.2.5 Warehouse Subsystem

2.2.5.1 Create Warehouse

2.2.5.2 View Warehouse

2.2.5.3 Update Warehouse

2.2.5.4 Reactivate Warehouse

2.2.5.5 Deactivate Warehouse

2.2.6 Vendor Subsystem

2.2.6.1 Create Vendor

2.2.6.2 View Vendor

2.2.6.3 View Vendors

2.2.6.4 Edit Vendor

2.2.6.5 Reactivate Vendor

2.2.6.6 Deactivate Vendor

2.2.6.7 Create Vendor Type

2.2.6.8 Update Vendor Type

2.2.6.9 View Vendor Types

2.2.6.10 Reactivate Vendor Type

2.2.6.11 Deactivate Vendor Type

2.2.7 Project Management Subsystem

2.2.7.1 Create Project

2.2.7.2 View Project

2.2.7.3 Update Project

2.2.7.4 Deactivate Project

2.2.7.5 Create Project Type

2.2.7.6 View Project Type

2.2.7.7 Update Project Type

2.2.7.8 Deactivate Project Type

2.2.7.9 Add Indemnity Form

2.2.7.10 Update Indemnity Form

2.2.7.11 View Indemnity Form

2.2.7.12 Remove Indemnity Form

2.2.7.13 Capture Indemnity Form

2.2.7.14 Capture Project Feedback

2.2.7.15 View Project Feedback

2.2.7.16 Create Impact Feedback Form

2.2.7.17 View Impact Feedback Form

2.2.7.18 Update Impact Feedback Form

2.2.7.19 Deactivate Impact Feedback Form

2.2.7.20 Capture Impact Feedback

2.2.7.21 View Impact Feedback

2.2.7.22 Add Photo

2.2.7.23 View Photo

2.2.7.24 Delete Photo

2.2.8 K9 Subsystem

2.2.8.1 Add Dog

2.2.8.2 View Dog

2.2.8.3 View Dogs

2.2.8.4 Update Dog

2.2.8.5 Deactivate Dog

2.2.8.6 Assign User to Dog

2.2.8.7 View User-Dog Assignment

2.2.8.8 Update User-Dog Assignment

2.2.8.9 Reactivate User-Dog Assignment

2.2.8.10 Deactivate User-Dog Assignment

2.2.8.11 Add K9 Role

2.2.8.12 Update K9 Role

2.2.8.13 View K9 Roles

2.2.8.14 Reactivate K9 Role

2.2.8.15 Deactivate K9 Role

2.2.8.16 View K9 Ancestry

2.2.9 Rhino Subsystem

2.2.9.1 Add Rhino

2.2.9.2 View Rhino

2.2.9.3 View Rhinos

2.2.9.4 Update Rhino

2.2.9.5 Deactivate Rhino

2.2.10 Elephant Subsystem

2.2.10.1 Add Elephant

2.2.10.2 View Elephant

2.2.10.3 View Elephants

2.2.10.4 Update Elephant

2.2.10.5 Deactivate Elephant

2.2.11 Message Subsystem

2.2.11.1 Create Message

2.2.11.2 View Messages

2.2.12 Report Subsystem

2.2.12.1 Generate Custom Report

2.2.12.2 View Custom Report

2.2.12.3 Update Custom Report

2.2.12.4 Remove Custom Report

2.2.12.5 View Bike Impact Report

2.2.12.6 View Budget Report

2.2.12.7 View Rhino Report

2.2.12.8 View Elephant Report

2.2.12.9 View K9 Report

2.2.13 Machine Learning Subsystem

2.2.13.1 View Machine Learning Dashboard

2.2.14. System Subsystem

2.2.14.1 Backup System

2.2.14.2 Restore System

2.2.14.3 View Audit Log

2.2.15 Budget Subsystem

2.2.15.1 Create Project Budget

2.2.15.2 View Project Budget

2.2.15.3 View Project Budgets

2.2.15.4 Update Project Budget

2.2.15.5 Log An Expense

2.2.15.6 Update An Expense

2.2.15.7 View Expense

2.2.15.8 Remove Expense

2.3 Functional Requirements List Descriptions

2.3.1 User Subsystem

Functional Requirement	Explanation
Requirement number:	1.1
Requirement name (use case name):	User Login
Requirement short description:	A User logs into the system using their email address and password.
Requirement detailed description and constraints:	A user logs in to the system using their email address and password. The system compares the email address with the associated hashed password in the database; upon successful authentication the user is directed to the appropriate dashboard based on their role and associated permission level. **Constraints:** • The user should already have created an account on the system. If not, they must sign up. • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Secure Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 1: 1.1 User Login - Description

Functional Requirement	Explanation
Requirement number:	1.2
Requirement name (use case name):	User Logout
Requirement short description:	A user can log out of the system.
Requirement detailed description and constraints:	A user can log out of the system, thus terminating their session, ensuring that no one with physical access to their device can access the system. Once they have completed this action, they are directed to the Login page. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Secure Access.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 2: 1.2 User Logout Description

Functional Requirement	Explanation
Requirement number:	1.3
Requirement name (use case name):	Reset Password
Requirement short description:	A user could reset their password if they either forgot it or suspect that their credentials might have leaked on another service.
Requirement detailed description and constraints:	A user can reset their password if they either forgot it or suspect that their credentials have leaked on another service. As the system does not save passwords in plain text, but rather a hashed format, the user will be prompted to enter their email address, to which a reset link will be sent. Once they have clicked the link, they have access to a field where they can enter a new password. The password hash is then saved in the database, replacing the previous version. Once these actions have been completed, the user will be redirected to the Login page. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • The password cannot be the same as the previous password, if this edge case occurs, the system will notify that the passwords are the same. • The user should already have created an account on the system. If not, they must sign up.
Business rules applicable to this requirement	• Secure Access. • Ability to recover Account.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 3: 1.3 Reset Password – Description

Functional Requirement	Explanation
Requirement number:	1.4
Requirement name (use case name):	Create User
Requirement short description:	A person must be able to create a user profile by signing up to the system.
Requirement detailed description and constraints:	A person must be able to create a user profile by signing up to the system. This entails providing login credentials (email address/password), basic personal information such as gender and age, location and selecting skills they possess. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • The new user must have a valid email address.
Business rules applicable to this requirement	• Secure Access.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 4: 1.4 Create User - Description

Functional Requirement	Explanation
Requirement number:	1.5
Requirement name (use case name):	Update User
Requirement short description:	A user must be able to edit their personal user profile.
Requirement detailed description and constraints:	A user must be able to edit their personal user profile. This could entail changing details like location or the skills they possess. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • The user cannot change their age, as this is automatically updated by the system.
Business rules applicable to this requirement	• Secure Access.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 5: 1.5 Update User - Description

Functional Requirement	Explanation
Requirement number:	1.6
Requirement name (use case name):	View User
Requirement short description:	A user must be able to view their personal user profile.
Requirement detailed description and constraints:	A user must be able to edit their personal user profile. Administrators should have the ability to view the personal profiles of all users This could entail changing details like location or the skills they possess. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • Only Administrators can view the full profile details of other users.
Business rules applicable to this requirement	• Secure Access.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 6: 1.6 View User - Description

Functional Requirement	Explanation
Requirement number:	1.7
Requirement name (use case name):	Reactivate User
Requirement short description:	An administrator must be able to reactivate an account.
Requirement detailed description and constraints:	An administrator should be able to reactivate any deactivated user accounts. Once an account has been reactivated, a user can login to the system using their credentials and will appear on the list of active users again. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Secure Access. • Data Retention.
Revision date and Revision number:	2026-03-28 Version 1.0
Criticality/Priority:	High

Table 7: 1.7 Reactivate User - Description

Functional Requirement	Explanation
Requirement number:	1.8
Requirement name (use case name):	Deactivate User
Requirement short description:	A user must be able to deactivate their personal user profile.
Requirement detailed description and constraints:	A user must be able to deactivate their personal user profile. Once they have selected the option to deactivate their profile, they will be asked to confirm. If they confirm, the profile will be deactivated, and they will be directed to the login page. Data regarding their involvement in projects will be kept for accurate records on historical data, but they will no longer appear on the users list. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Secure Access. • Data Retention.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 8: 1.8 Deactivate User - Description

Functional Requirement	Explanation
Requirement number:	1.9
Requirement name (use case name):	View FAQ
Requirement short description:	A user must be able to click a help button that takes them to a context aware FAQ page.
Requirement detailed description and constraints:	A user must be able to click a help button that takes them to a context aware FAQ page. If the button is pressed on the home page, it will direct the user to the full FAQ document, but if it is pressed while using a subsystem, it will direct the user to the appropriate section as set up by the admin using the FAQ Type use cases. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Access to a help function.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 9: 1.9 View FAQ - Description

2.3.2 Admin Subsystem

Functional Requirement	Explanation
Requirement number:	2.1
Requirement name (use case name):	Create User Profile
Requirement short description:	A Super Admin must be able to create a new user profile in the system.
Requirement detailed description and constraints:	Super Admin can input necessary details to create a new user. The system validates entries before asking for confirmation. Once confirmed, the user profile is saved. **Constraints:** • Internet connection required. • Mandatory fields must be completed before submission.
Business rules applicable to this requirement	• Only the Super Admin can create user profiles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 10: 2.1 Create User Profile - Description

Functional Requirement	Explanation
Requirement number:	2.2
Requirement name (use case name):	Update User Profile
Requirement short description:	A Super Admin must be able to update existing user profiles.
Requirement detailed description and constraints:	The Super Admin can modify user information. System validates changes before confirmation. Once confirmed, updates are applied. **Constraints:** • Internet connection required. • Cannot change another user's password without proper authorization.
Business rules applicable to this requirement	• Only the Super Admin can create user profiles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 11: 2.2 Update User Profile - Description

Functional Requirement	Explanation
Requirement number:	2.3
Requirement name (use case name):	View User Profile
Requirement short description:	The Super Admin must be able to view user profiles.
Requirement detailed description and constraints:	The Super Admin can search and view details of existing users. No changes are allowed in this view. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can view any profiles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 12: 2.3 View User Profile - Description

Functional Requirement	Explanation
Requirement number:	2.4
Requirement name (use case name):	Deactivate User Profile
Requirement short description:	The Super Admin can deactivate user profiles.
Requirement detailed description and constraints:	Super Admin can disable a profile temporarily. The system confirms the action before applying it. **Constraints:** • Internet connection required. • Action must be logged in the system audit trail.
Business rules applicable to this requirement	• Only the Super Admin can deactivate profiles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 13: 2.4 Deactivate User Profile - Description

Functional Requirement	Explanation
Requirement number:	2.5
Requirement name (use case name):	Add FAQ
Requirement short description:	The Super Admin can add new FAQs to the system.
Requirement detailed description and constraints:	The Super Admin input question and answer details. System validates before confirmation. Once confirmed, FAQ is added. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can add FAQs.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 14: 2.5 Add FAQ - Description

Functional Requirement	Explanation
Requirement number:	2.6
Requirement name (use case name):	Update FAQ
Requirement short description:	The Super Admin can update existing FAQs.
Requirement detailed description and constraints:	The Super Admin can edit FAQ content, which is validated before confirmation. Once confirmed, updates are applied. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can update FAQs.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 15: 2.6 Update FAQ - Description

Functional Requirement	Explanation
Requirement number:	2.7
Requirement name (use case name):	Remove FAQ
Requirement short description:	The Super Admin can remove FAQs from the system.
Requirement detailed description and constraints:	The Super Admin can select FAQ to delete. System asks for confirmation before removal. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can remove FAQs from the system.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 16: 2.7 Remove FAQ - Description

Functional Requirement	Explanation
Requirement number:	2.8
Requirement name (use case name):	Create Role
Requirement short description:	The Super Admin can create new roles in the system.
Requirement detailed description and constraints:	The Super Admin enter role name and permissions. System validates input before confirmation. Once confirmed, role is created. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• The Super Admin can create roles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 17: 2.8 Create Role - Description

Functional Requirement	Explanation
Requirement number:	2.9
Requirement name (use case name):	View Role
Requirement short description:	The Super Admin can view details of the existing roles.
Requirement detailed description and constraints:	The Super Admin can search and see role details. No changes allowed in view mode. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can view roles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 18: 2.9 View Role - Description

Functional Requirement	Explanation
Requirement number:	2.10
Requirement name (use case name):	Update Role
Requirement short description:	The Super Admin can update existing roles.
Requirement detailed description and constraints:	The Super Admin can modify role names or permissions. System validates changes before confirmation. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• The Super Admin can update roles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 19: 2.10 Update Role - Description

Functional Requirement	Explanation
Requirement number:	2.11
Requirement name (use case name):	Reactivate Role
Requirement short description:	The Super Admin can reactivate deactivated roles.
Requirement detailed description and constraints:	The Super Admin selects a deactivated role. System confirms before reactivation. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can reactivate a role.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 20:2.11 Reactivate Role - Description

Functional Requirement	Explanation
Requirement number:	2.12
Requirement name (use case name):	Deactivate Role
Requirement short description:	The Super Admin can deactivate roles.
Requirement detailed description and constraints:	The Super Admin selects a role. System asks for confirmation before applying deactivation. **Constraints:** • Internet connection required. • Action must be logged.
Business rules applicable to this requirement	• Only the Super Admin can deactivate roles.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 21: 2.12 Deactivate Role - Description

Functional Requirement	Explanation
Requirement number:	2.13
Requirement name (use case name):	Create Role Permission
Requirement short description:	The Super Admin can create permissions for roles.
Requirement detailed description and constraints:	The Super Admin enters permission details. System validates input before confirmation. Once confirmed, permission is added. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can create role permissions.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 22: 2.13 Create Role Permission - Description

Functional Requirement	Explanation
Requirement number:	2.14
Requirement name (use case name):	Update Role Permission
Requirement short description:	The Super Admin modifies existing role permissions. System validates changes before confirmation.
Requirement detailed description and constraints:	The Super Admin modifies existing role permissions. System validates changes before confirmation. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• The Super Admin can update role permissions.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 23: 2.14 Update Role Permission - Description

Functional Requirement	Explanation
Requirement number:	2.15
Requirement name (use case name):	View Role Permission
Requirement short description:	The Super Admin can view permissions assigned to roles.
Requirement detailed description and constraints:	The Super Admin can search and view role permissions. No edits allowed. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• The Super Admin can view role permissions.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 24: 2.15 View Role Permission - Description

Functional Requirement	Explanation
Requirement number:	2.16
Requirement name (use case name):	Deactivate Role Permission
Requirement short description:	The Super Admin can deactivate permissions assigned to roles.
Requirement detailed description and constraints:	The Super Admin selects a permission to deactivate. System asks for confirmation before applying. **Constraints:** • Internet connection required.
Business rules applicable to this requirement	• Only the Super Admin can deactivate role permissions.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 25: 2.16 Deactivate Role Permission - Description

2.3.3 Inventory Subsystem

Functional Requirement	Explanation
Requirement number:	3.1
Requirement name (use case name):	Receive Inventory Item
Requirement short description:	A user must be able to add an item(s) to the system's inventory.
Requirement detailed description and constraints:	A user must be able to add an item(s) to the system's inventory once the item is received from a vendor. The system will allow the admin to enter details regarding the inventory received. The system will validate the details entered before asking for confirmation. Once confirmed the system will add the inventory to the system. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with the required permissions can receive inventory.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 26: 3.1 Receive Inventory Item - Description

Functional Requirement	Explanation
Requirement number:	3.2
Requirement name (use case name):	View Inventory
Requirement short description:	A user must be able to view all inventory on the system.
Requirement detailed description and constraints:	This use case describes the event in which the user views the inventory on the system. The system will retrieve and display all the inventory on the system as well as allowing the user to filter. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with the required permissions can receive inventory.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 27: 3.2 View Inventory - Description

Functional Requirement	Explanation
Requirement number:	3.3
Requirement name (use case name):	Write-Off Inventory.
Requirement short description:	A user with appropriate permissions must be able to write off an inventory item(s).
Requirement detailed description and constraints:	A user with appropriate permissions must be able to edit the details of an inventory item(s). The system will display all the inventory on the system and allow the user to write-off inventory. The system will ask for confirmation before the inventory items are written off. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with the required permissions can receive inventory.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 28: 3.3 Write-Off Inventory - Description

Functional Requirement	Explanation
Requirement number:	3.4
Requirement name (use case name):	Inventory Stock Take.
Requirement short description:	A user with appropriate permissions must be able to perform a stock take.
Requirement detailed description and constraints:	A user with appropriate permissions must be able to perform a stock take. The system will allow the user to select a warehouse and view its inventory. The user will perform a stock take and enter the stock count. The system will determine the difference and capture the difference to adjust the inventory. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with the required permissions can receive perform a stock take.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 29: 3.4 Inventory Stock Take - Description

Functional Requirement	Explanation
Requirement number:	3.5
Requirement name (use case name):	Allocate Inventory
Requirement short description:	A user with the appropriate role must be able to allocate inventory to a project.
Requirement detailed description and constraints:	A user with the appropriate role must be able to allocate inventory to a project. The system will display all the inventory and allow the user to allocate inventory to an active project. The system will validate inputted data and ask for confirmation before allocating inventory. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with the required permissions can receive perform a stock take.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 30: 3.5 Allocate Inventory - Description

Functional Requirement	Explanation
Requirement number:	3.6
Requirement name (use case name):	Create Inventory Item Type
Requirement short description:	A user with the appropriate permissions must be able to create a new inventory item type.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to create a new inventory item type that will then be selectable when adding inventory items. Information including name, description and security level. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can create an inventory item type.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 31: 3.6 Create Inventory Item Type - Description

Functional Requirement	Explanation
Requirement number:	3.7
Requirement name (use case name):	View Inventory Item Type
Requirement short description:	A user must be able to view an inventory item type.
Requirement detailed description and constraints:	A user must be able to view an inventory item type. The system will retrieve and display all the inventory item types on the system before the user selects an inventory item type to view. Information including name, description and security level must be displayed. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view an inventory item type.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 32: 3.7 View Inventory Item Type - Description

Functional Requirement	Explanation
Requirement number:	3.8
Requirement name (use case name):	View Inventory Item Types
Requirement short description:	A user must be able to view all inventory item types.
Requirement detailed description and constraints:	A user must be able to view all inventory item types. The system will retrieve and display all the inventory item types on the system before displaying them to the user. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view all inventory item types.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 33:3.8 View Inventory Item Types - Description

Functional Requirement	Explanation
Requirement number:	3.9
Requirement name (use case name):	Update Inventory Item Type
Requirement short description:	A user with the appropriate permissions must be able to update an inventory item type.
Requirement detailed description and constraints:	A user with the appropriate permissions must be update inventory item type that will then be selectable when adding inventory items. Information including name, description and security level can be changed. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can update an inventory item type.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 34: 3.9 Update Inventory Item Type - Description

Functional Requirement	Explanation
Requirement number:	3.10
Requirement name (use case name):	Deactivate Inventory Item Type
Requirement short description:	A user with the appropriate permissions must be able to deactivate an inventory item type.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to deactivate an inventory item type if the type is no longer needed by the organization due to a change in policy or project scope. Data regarding previous use of the inventory item type will be kept for accurate record keeping, but the type will no longer appear when selecting an inventory item type. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can deactivate an inventory item type.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 35: 3.10 Deactivate Inventory Item Type - Description

2.3.4 Group Subsystem

Functional Requirement	Explanation
Requirement number:	4.1
Requirement name (use case name):	Create Group
Requirement short description:	The system must allow a user with appropriate permissions to create a group for a project.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to create a group that is associated with a specific project. A project can have multiple groups within it and group members can see information related to them as well as communicate with each other. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Each group must be linked with an existing project on the system. • Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 36: 4.1 Create Group – Description

Functional Requirement	Explanation
Requirement number:	4.2
Requirement name (use case name):	View Groups
Requirement short description:	The system must allow users to view the details of the group they are in for a project.
Requirement detailed description and constraints:	The system must allow users on the system to view a group's details. When a user selects to view a group, the system will display information such as the group name, the project it belongs to and the other members. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Users can only view groups they are assigned too. • Users must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 37:4.2 View Group – Description

Functional Requirement	Explanation
Requirement number:	4.3
Requirement name (use case name):	Update Group
Requirement short description:	The system must allow a user with appropriate permissions to update the details of a group.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update a group's details within a project. The editable details will include the group name and ability to add or remove members. Once the user clicks save, the system will validate the updated data and save the changes. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 38: 4.3 Update Group – Description

Functional Requirement	Explanation
Requirement number:	4.4
Requirement name (use case name):	Deactivate Group
Requirement short description:	The system must allow a user with appropriate permissions to deactivate a group.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to deactivate a group within a project when it is no longer used or required. The system will confirm the deactivation and upon confirmation, the group will be deactivated. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 39: 4.4 Deactivate Group - Description

Functional Requirement	Explanation
Requirement number:	4.5
Requirement name (use case name):	Create Skill
Requirement short description:	The system must allow a user with appropriate permissions to create a new skill.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to create new skills that can be assigned to volunteers and staff members. These skills can be used to help assign volunteers or staff members to appropriate teams. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	Medium

Table 40: 4.5 Create Skill - Description

Functional Requirement	Explanation
Requirement number:	4.6
Requirement name (use case name):	View Skills
Requirement short description:	The system must allow users to view skills stored on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view the skills that have been created on the system. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Users must be logged into the system. • Skills must be displayed accurately.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	Medium

Table 41: 4.6 View Skill - Description

Functional Requirement	Explanation
Requirement number:	4.7
Requirement name (use case name):	Update Skill
Requirement short description:	The system must allow a user with appropriate permissions to update skills stored on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update the skills on the system such as correcting the name or spelling. Once the user clicks save, the system will validate the updated data and save the changes. This will ensure that skill information remains accurate. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	Medium

Table 42: 4.7 Update Skill - Description

Functional Requirement	Explanation
Requirement number:	4.8
Requirement name (use case name):	Deactivate Skill
Requirement short description:	The system must allow a user with appropriate permissions to deactivate a skill stored on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update the skills on the system such as correcting the name or spelling. Once the user clicks save, the system will validate the updated data and save the changes. This will ensure that skill information remains accurate. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	Medium

Table 43: 4.8 Deactivate Skill - Description

2.3.5 Warehouse Subsystem

Functional Requirement	Explanation
Requirement number:	5.1
Requirement name (use case name):	Create Warehouse
Requirement short description:	The Super Admin can create new warehouses.
Requirement detailed description and constraints:	The Super Admin can input warehouse name, location, and capacity. System validates details before confirmation. **Constraints:** • Internet connection required. • Location field cannot be empty. • Capacity must be a positive number.
Business rules applicable to this requirement	• Only users with warehouse admin permissions can create warehouses.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 44 :5.1 Create Warehouse - Description

Functional Requirement	Explanation
Requirement number:	5.2
Requirement name (use case name):	View Warehouse
Requirement short description:	The Super Admin can view warehouse details.
Requirement detailed description and constraints:	The Super Admin can search and view warehouses. No edits allowed in view mode. **Constraints:** • Internet connection required. • Access limited to users with warehouse admin privileges.
Business rules applicable to this requirement	• Only warehouse admin users can view warehouses.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 45: 5.2 View Warehouse - Description

Functional Requirement	Explanation
Requirement number:	5.3
Requirement name (use case name):	Update Warehouse
Requirement short description:	The Super Admin can update warehouse details.
Requirement detailed description and constraints:	The Super Admin modifies warehouse information. System validates changes before confirmation. **Constraints:** • Internet connection required. • Capacity updates must be positive numbers. • Changes must be logged.
Business rules applicable to this requirement	• Only warehouse admin users can update warehouses.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 46: 5.3 Update Warehouse - Description

Functional Requirement	Explanation
Requirement number:	5.4
Requirement name (use case name):	Reactivate Warehouse
Requirement short description:	The Super Admin can reactivate deactivated warehouses.
Requirement detailed description and constraints:	The Super Admin selects a deactivated warehouse. System asks for confirmation before reactivation. **Constraints:** • Internet connection required. • Action must be logged.
Business rules applicable to this requirement	• Only the warehouse admin user can reactivate warehouses.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 47 5.4: Reactivate Warehouse - Description

Functional Requirement	Explanation
Requirement number:	5.5
Requirement name (use case name):	Deactivate Warehouse
Requirement short description:	The Super Admin can deactivate warehouses.
Requirement detailed description and constraints:	The Super Admin selects a warehouse. System asks for confirmation before applying deactivation. **Constraints:** • Internet connection required. • Action must be logged.
Business rules applicable to this requirement	• Only warehouse admin users can deactivate warehouses.
Revision date and Revision number:	2026-03-28 Version 2.0
Criticality/Priority:	High

Table 48: 5.5 Deactivate Warehouse - Description

2.3.6 Vendor Subsystem

Functional Requirement	Explanation
Requirement number:	6.1
Requirement name (use case name):	Create Vendor
Requirement short description:	A user must be able to add a vendor to the system.
Requirement detailed description and constraints:	A user must be able to add a vendor to the system by adding information including vendor name, type, email, phone, contact person, credit facility (if applicable), address, rating and photo. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can add a vendor.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 49: 6.1 Create Vendor - Description

Functional Requirement	Explanation
Requirement number:	6.2
Requirement name (use case name):	View Vendor
Requirement short description:	A user must be able to view any given vendor.
Requirement detailed description and constraints:	A user must be able to view any given vendor. The system will then retrieve and display all relevant information regarding the vendor. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view a vendor.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 50 6.2: View Vendor - Description

Functional Requirement	Explanation
Requirement number:	6.3
Requirement name (use case name):	View Vendors
Requirement short description:	A user must be able to view all vendors.
Requirement detailed description and constraints:	A user must be able to view all vendors on the system. The system will retrieve and display all vendors and their information for the user. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view all vendor.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 51: 6.3 View Vendors - Description

Functional Requirement	Explanation
Requirement number:	6.4
Requirement name (use case name):	Edit Vendor
Requirement short description:	A user must be able to edit vendor information
Requirement detailed description and constraints:	A user must be able to edit a vendor on the system by changing information including vendor name, type, email, phone, contact person, credit facility (if applicable), address, rating and photo. **Constraints:** • The user requires an Internet connection to access this feature. • The vendor must already exist on the system.
Business rules applicable to this requirement	• Only a user with appropriate permissions can edit a vendor.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 52: 6.4 Edit Vendor - Description

Functional Requirement	Explanation
Requirement number:	6.5
Requirement name (use case name):	Reactivate Vendor
Requirement short description:	A user must be able to reactivate a vendor.
Requirement detailed description and constraints:	A user must be able to reactivate a vendor on the system. The system will display the vendor and allow the user to reactivate the vendor. The system will ask the user to confirm the reactivation before the vendor is reactivated **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can reactivate a vendor.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 53: 6.5 Reactivate Vendor - Description

Functional Requirement	Explanation
Requirement number:	6.6
Requirement name (use case name):	Deactivate Vendor
Requirement short description:	A user must be able to deactivate any given vendor.
Requirement detailed description and constraints:	A user must be able to deactivate any given vendor in the case that the vendor is no longer in use, or the vendor no longer exists. The system will display the vendor and allow the user to deactivate the vendor. The system will ask the user to confirm the deactivation before the vendor is deactivated. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can deactivate a vendor.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 54: 6.6 Deactivate Vendor - Description

Functional Requirement	Explanation
Requirement number:	6.7
Requirement name (use case name):	Create Vendor Type
Requirement short description:	A user with the appropriate permissions must be able to create a type that vendors can be categorised under.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to create a type that vendors can be categorised under. These types will be used when updating and creating vendors. It also ensures that vendors can be easily filtered. The details will include type name, description and status. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Vendor management capabilities. • Use of Preferred vendors. • Role based access
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 55: 6.7 Create Vendor Type - Description

Functional Requirement	Explanation
Requirement number:	6.8
Requirement name (use case name):	Update Vendor Type
Requirement short description:	A user with the appropriate permissions must be able to update a vendor type.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to update a vendor type. This includes changes to the type name and description. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature. • Vendor type must already exist on the system.
Business rules applicable to this requirement	• Vendor management capabilities. • Use of Preferred vendors. • Role based access
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 56: 6.8 Update Vendor Type - Description

Functional Requirement	Explanation
Requirement number:	6.9
Requirement name (use case name):	View Vendor Types
Requirement short description:	A user with the appropriate permissions must be able to view the vendor types.
Requirement detailed description and constraints:	A user must be able to view the vendor types and then view all the available information regarding the vendor types. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Vendor management capabilities. • Use of Preferred vendors. • Role based access
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 57: 6.8 View Vendor Types - Description

Functional Requirement	Explanation
Requirement number:	6.10
Requirement name (use case name):	Reactivate Vendor Type
Requirement short description:	A user with the appropriate permissions must be able to view the vendor types.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to reactivate a vendor type when it is required again by the organization due to a change in policy or project scope. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature. Vendor type must already exist on the system.
Business rules applicable to this requirement	• Vendor management capabilities. • Use of Preferred vendors. • Role based access
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 58: 6.10 Reactivate Vendor Type - Description

Functional Requirement	Explanation
Requirement number:	6.11
Requirement name (use case name):	Deactivate Vendor Type
Requirement short description:	A user with the appropriate permissions must be able to deactivate a vendor type.
Requirement detailed description and constraints:	A user with the appropriate permissions must be able to deactivate a vendor type if the type is no longer needed by the organization due to a change in policy or project scope. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature. • Vendor type must already exist on the system.
Business rules applicable to this requirement	• Vendor management capabilities. • Use of Preferred vendors. • Role based access
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 59: 6.11 Deactivate Vendor Type - Description

2.3.7 Project Subsystem

Functional Requirement	Explanation
Requirement number:	7.1
Requirement name (use case name):	Create Project
Requirement short description:	A Super-Admin creates a new project representing an NGO goal by inputting project details and team assignments.
Requirement detailed description and constraints:	A Super-Admin will create a new project that will represent a new NGO goal. The system will open a form where the Super-Admin will input the project: name, dates, staff, teams, milestones and deadlines to help plan and organise the project. The system will capture all the information and alert all the relevant staff and create the schedule. **Constraints:** • The staff have all been previously added to the system. • The staff roles have all been previously added to the system. • The project type has been added to the system previously. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 60: 7.1 Create Project - Description

Functional Requirement	Explanation
Requirement number:	7.2
Requirement name (use case name):	View Project
Requirement short description:	A user clicks on a specific project to see all its details.
Requirement detailed description and constraints:	A user clicks on a specific project to see all its details, like the name, staff, dates, project type, milestones, and deadline. They see this in the project's nav bar option. The system retrieves the details and displays them in sections relative to the user's role. **Constraints:** • The user must already have a pre-existing account on the system. The user is logged in. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 61: 7.2 View Project - Description

Functional Requirement	Explanation
Requirement number:	7.3
Requirement name (use case name):	Update Project
Requirement short description:	A Super-Admin will edit an existing project already saved in the system.
Requirement detailed description and constraints:	A Super-Admin will edit an existing project already saved in the system. The system will open the project's form where the Super-Admin will be able to edit the project: name, dates, staff, teams, milestones and deadlines. The system will capture all the updated information and alert all the relevant staff. **Constraints:** • The project has already been created. • The staff have all been previously added to the system. • The staff roles have all been previously added to the system. • The project type has been added to the system previously. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 62: 7.3 Update Project - Description

Functional Requirement	Explanation
Requirement number:	7.4
Requirement name (use case name):	Deactivate Project
Requirement short description:	A super admin can deactivate projects after they are completed or no longer support goals.
Requirement detailed description and constraints:	A super admin can deactivate projects after they are completed or if the project no longer supports the NPO's organisational goals. The system will load the form and ask the user for confirmation before deactivating the project and notifying the admin of the status change. **Constraints:** • The project is already created and saved on the system. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 63: 7.4 Deactivate Project - Description

Functional Requirement	Explanation
Requirement number:	7.5
Requirement name (use case name):	Create Project Type
Requirement short description:	The super admin creates a new category for projects.
Requirement detailed description and constraints:	The super admin would like to create a new category for projects that will represent a type of project. The system will display a form where the admin can type the name of the new project type, which the system then verifies and saves. **Constraints:** • The user must already have a pre-existing account on the system. • The user has super admin privileges. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 64: 7.5 Create Project Type - Description

Functional Requirement	Explanation
Requirement number:	7.6
Requirement name (use case name):	View Project Type
Requirement short description:	The super admin can see, edit, or deactivate project types.
Requirement detailed description and constraints:	The super admin can click "View Project types" to be able to see, edit, or deactivate project types to better categorise the projects being completed. The system retrieves recorded project types and displays them in a table for the super admin to view. **Constraints:** • The user must already have a pre-existing account on the system. • The user must have super admin privileges. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 65: 7.6 View Project Type - Description

Functional Requirement	Explanation
Requirement number:	7.7
Requirement name (use case name):	Update Project Type
Requirement short description:	A super admin wants to change the name of a project type.
Requirement detailed description and constraints:	A super admin wants to change the name of a project type so users can better understand what those types of projects represent. The system retrieves the recorded project type and displays the name as an editable field for the admin to update. **Constraints:** • The project type has already been added and captured onto the system. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 66: 7.7 Update Project Type - Description

Functional Requirement	Explanation
Requirement number:	7.8
Requirement name (use case name):	Deactivate Project Type
Requirement short description:	The super admin deactivates a project type no longer being used.
Requirement detailed description and constraints:	The super admin deactivates a project type and it will remove it from the displayed table of project types. This would be because a project type is no longer being used. The system loads the project type name for confirmation before deactivating it. **Constraints:** • The project type has already been captured on the system. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 67: 7.8 Deactivate Project Type - Description

Functional Requirement	Explanation
Requirement number:	7.9
Requirement name (use case name):	Add Indemnity Form
Requirement short description:	The super admin can add a new Indemnity form to projects.
Requirement detailed description and constraints:	The super admin can add a new Indemnity form that can be added to projects to make form completions and capturing easier. The system shows fields to place a name for the form and a field to upload a pdf file or paste text. **Constraints:** • The admin is signed in and the form hasn't been added yet. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 68: 7.9 Add Indemnity Form - Description

Functional Requirement	Explanation
Requirement number:	7.10
Requirement name (use case name):	Update Indemnity Form
Requirement short description:	The super admin can update an existing Indemnity form.
Requirement detailed description and constraints:	The super admin can update an existing Indemnity form that can be added to projects to make form completions and capturing easier. The system displays editable fields where the admin can change the name, text, or pdf file of the form. **Constraints:** • The admin is signed in and the form hasn't been added yet. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 69: 7.10 Update Indemnity Form - Description

Functional Requirement	Explanation
Requirement number:	7.11
Requirement name (use case name):	View Indemnity Form
Requirement short description:	The super admin can see, edit, add or deactivate Indemnity Forms.
Requirement detailed description and constraints:	The super admin can click "View Indemnity Form" to be able to see, edit, add or deactivate Indemnity Forms to better help the projects. The system retrieves the recorded forms and displays them in a table for the admin to manage. **Constraints:** • The user must already have a pre-existing account on the system. • The user must have super admin privileges. • Must be completed before project day. • Must be user-friendly and easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 70: 7.11 View Indemnity Form - Description

Functional Requirement	Explanation
Requirement number:	7.12
Requirement name (use case name):	Remove Indemnity Form
Requirement short description:	The super admin can remove an existing indemnity form no longer being used.
Requirement detailed description and constraints:	The super admin can remove an existing indemnity form that is no longer being used to reduce clutter. The system loads the form content for deactivation confirmation and notifies the user once the form is removed from the table. **Constraints:** • The admin is signed in and the form hasn't been added yet. • Must be completed before project day. • Must be user-friendly and easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 71: 7.12 Remove Indemnity Form - Description

Functional Requirement	Explanation
Requirement number:	7.13
Requirement name (use case name):	Capture Indemnity Form
Requirement short description:	A super admin can attach an indemnity form to a project.
Requirement detailed description and constraints:	A super admin can attach an indemnity form to a project which will make it so that anyone who has been added to that project must complete the compulsory indemnity form in order to view the project details. The system captures the selection and requires signatures from staff. **Constraints:** • The super admin must already have a pre-existing account on the system. • Must be completed before project day. • Must be user-friendly and easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 72: 7.13 Capture Indemnity Form - Description

Functional Requirement	Explanation
Requirement number:	7.14
Requirement name (use case name):	Capture Project Feedback
Requirement short description:	A super admin can attach a project feedback form to a project.
Requirement detailed description and constraints:	A super admin can attach a project feedback form to a project which will make it so that anyone who has been added to that project will be sent a form 1 week after the project commences asking them for a rating and comments. The system captures the selection for all involved staff. **Constraints:** • The project assigned to the feedback form must have commenced. • Must be completed before project day. • Must be user-friendly and easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 73: 7.14 Capture Project Feedback - Description

Functional Requirement	Explanation
Requirement number:	7.15
Requirement name (use case name):	View Project Feedback
Requirement short description:	A super admin can view project feedback form ratings and comments.
Requirement detailed description and constraints:	A super admin can view all of the past projects and see the feedback forms ratings and comments for each project. The system retrieves the records for feedback and displays it neatly for the admin to read. **Constraints:** • The projects have been saved on the system and not been deactivated. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 74: 7.15 View Project Feedback - Description

Functional Requirement	Explanation
Requirement number:	7.16
Requirement name (use case name):	Create Impact Feedback Form
Requirement short description:	The super admin can add a new impact form for community members.
Requirement detailed description and constraints:	The super admin can add a new impact form that can be added to projects to make form completions and capturing easier for community members. The system shows fields to name the form and buttons to add and define question types. **Constraints:** • The admin creating the form has permissions to create impact forms already. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 75: 7.16 Create Impact Feedback Form - Description

Functional Requirement	Explanation
Requirement number:	7.17
Requirement name (use case name):	View Impact Forms
Requirement short description:	A super admin can view project impact form questions and answers.
Requirement detailed description and constraints:	A super admin can view all of the past projects and see the impact forms questions and answers and see what project they are assigned to. The system retrieves records for impact data and displays it neatly for the admin. **Constraints:** • The projects have been saved on the system and not been deactivated. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 76: 7.17 View Impact Forms - Description

Functional Requirement	Explanation
Requirement number:	7.18
Requirement name (use case name):	Update Impact Form
Requirement short description:	The super admin can update an existing impact form.
Requirement detailed description and constraints:	The super admin can update an existing impact form that can be added to projects to make form completions and capturing easier. The system shows fields for the form name and the questions currently on the form as editable content. **Constraints:** • The admin is signed in, and the form hasn't been added yet. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 77: 7.18 Update Impact Form - Description

Functional Requirement	Explanation
Requirement number:	7.19
Requirement name (use case name):	Deactivate Impact Form
Requirement short description:	The admin can deactivate an impact form from the list of all forms.
Requirement detailed description and constraints:	The admin can deactivate an impact form from the list of all the forms being displayed; this can remove unused forms and clear clutter from the system. The system loads the form name for deactivation confirmation before removing it from view. **Constraints:** • The impact form has already been created and is displayed on the list. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 78: 7.19 Deactivate Impact Form - Description

Functional Requirement	Explanation
Requirement number:	7.20
Requirement name (use case name):	Capture Impact Form
Requirement short description:	The super admin can include an impact form and send it to a community partner.
Requirement detailed description and constraints:	When creating or editing a project the super admin can check the "Include impact form" button and choose from the list or create a new one. The system will capture an email address or phone number to send the form to the community partner. **Constraints:** • The impact form has already been captured in the system. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 79: 7.20 Capture Impact Form - Description

Functional Requirement	Explanation
Requirement number:	7.21
Requirement name (use case name):	View Impact Feedback
Requirement short description:	A super admin can view all of the project's impact results.
Requirement detailed description and constraints:	A super admin can view all of the project's impact results and see the impact form questions and answers, the assigned project, and which community partner answered. The system retrieves and displays these records neatly. **Constraints:** • The impact forms have been saved on the system and not been deactivated. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 80: 7.21 View Impact Feedback - Description

Functional Requirement	Explanation
Requirement number:	7.22
Requirement name (use case name):	Add Photo
Requirement short description:	A user can upload multiple photos to a project's album.
Requirement detailed description and constraints:	A user can click on a project to see all photos from that project, then click the add photos button to upload multiple photos to that album. The system loads a page where images can be dragged, dropped, or uploaded from the device. **Constraints:** • The project's photo group has already been added, and the user has access to that project. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 81: 7.22 Add Photo - Description

Functional Requirement	Explanation
Requirement number:	7.23
Requirement name (use case name):	View Photo
Requirement short description:	A user can see all photos from a specific project.
Requirement detailed description and constraints:	A user can click on a project to see all photos from that project; they can then click the project to see all photos in that album. All projects are displayed in a table with their names and a representative project photo. **Constraints:** • All the project's photo group has already been added and the user has access to those projects. • Must be completed before project day. • Must be user-friendly and easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 82: 7.23 View Photo - Description

Functional Requirement	Explanation
Requirement number:	7.24
Requirement name (use case name):	Delete Photo
Requirement short description:	A user can remove photos from a project's image folder.
Requirement detailed description and constraints:	A user can click on a project to see all photos and then click the delete photos button; the system then displays an "X" button on photos they uploaded to remove them. The system captures the selection and updates the album. **Constraints:** • The project's photo group has already been added and the user has access to that project. • The user is deleting photo's they uploaded or are a super admin. • Must be completed before project day. • Must be easy to use.
Business rules applicable to this requirement	• Secure Access • Role Based Access
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	High

Table 83: 7.24 Delete Photo - Description

2.3.8 K9 Subsystem

Functional Requirement	Explanation
Requirement number:	8.1
Requirement name (use case name):	Add Dog
Requirement short description:	The system must allow a user with appropriate permissions to add a dog to the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to capture and store the details of new dogs used in K9 operations. The details will include the dog's name, breed, age, gender, role, status, handler and parents. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • Each dog must be assigned a handler when applicable. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 84: 8.1 Add Dog - Description

Functional Requirement	Explanation
Requirement number:	8.2
Requirement name (use case name):	View Dog
Requirement short description:	The system must allow a user with appropriate permissions to view a dog on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view the details of each dog used in K9 operations. This will help users to monitor and manage the K9 unit. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • Dog information must be stored and displayed accurately. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 85: 8.2 View Dog - Description

Functional Requirement	Explanation
Requirement number:	8.3
Requirement name (use case name):	View Dogs
Requirement short description:	The system must allow a user with appropriate permissions to view the dogs on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view the dogs used in K9 operations. This will help users to monitor and manage the K9 unit. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-21 Version 1.0
Criticality/Priority:	High

Table 86: 8.3 View All Dogs- Description

Functional Requirement	Explanation
Requirement number:	8.4
Requirement name (use case name):	Update Dog
Requirement short description:	The system must allow a user with appropriate permissions to update a dog on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update the details of an existing dog used in the K9 operations. The editable details will include the dog's name, breed, age, gender, role, and parents. Once the user clicks save, the system will validate the updated data and save the changes. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 87: 8.4 Update Dog - Description

Functional Requirement	Explanation
Requirement number:	8.5
Requirement name (use case name):	Deactivate Dog
Requirement short description:	The system must allow a user with appropriate permissions to deactivate a dog on the system.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to deactivate a dog on the system. This could occur when the dog becomes retired and no longer participating in anti-poaching activities. The system will confirm to deactivate the dog and upon confirmation, the dog will be deactivated. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-21 Version 1.0
Criticality/Priority:	High

Table 88: 8.5 Deactivate Dog - Description

Functional Requirement	Explanation
Requirement number:	8.6
Requirement name (use case name):	Assign User to Dog
Requirement short description:	The system must allow a user with appropriate permissions to assign a user to a dog.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to assign a registered handler to a dog on the system. This will help track who is responsible for the dog and working with the dog during patrols. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • The handler must be registered on the system. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 89: 8.6 Assign User to Dog - Description

Functional Requirement	Explanation
Requirement number:	8.7
Requirement name (use case name):	View User-Dog Assignment
Requirement short description:	The system must allow a user with appropriate permissions to view the assignment of users to dogs.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view which handlers are assigned to which dogs on the system. This will track responsibilities, ensure accountability, and manage K9 operations effectively. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • All assignments must be stored and displayed accurately. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 90: 8.7 View User-Dog Assignment - Description

Functional Requirement	Explanation
Requirement number:	8.8
Requirement name (use case name):	Update User-Dog Assignment
Requirement short description:	The system must allow a user with appropriate permissions to update the assignment of users to dogs.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update a dog's handler from one handler to another if needed. Once the user clicks save, the system will validate the updated data and save the changes. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • The new handler must be registered on the system. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 91: 8.8 Update User-Dog Assignment - Description

Functional Requirement	Explanation
Requirement number:	8.9
Requirement name (use case name):	Reactivate User-Dog Assignment
Requirement short description:	The system must allow a user with appropriate permissions to reactivate the assignment of users to dogs.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to reactivate a previous assignment between a handler and a dog. The system will confirm the reactivation and upon confirmation, the user-dog assignment will be reactivated. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-21 Version 1.0
Criticality/Priority:	High

Table 92: 8.9 Reactivate User-Dog Assignment - Description

Functional Requirement	Explanation
Requirement number:	8.10
Requirement name (use case name):	Deactivate User-Dog Assignment
Requirement short description:	The system must allow a user with appropriate permissions to deactivate the assignment of users to dogs.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to deactivate an existing assignment between a handler and a dog. The system will confirm the deactivation and upon confirmation, the user-dog assignment will be deactivated. This will remove the handler from being responsible for the dog. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 93: 8.10 Deactivate User-Dog Assignment - Description

Functional Requirement	Explanation
Requirement number:	8.11
Requirement name (use case name):	Add K9 Role
Requirement short description:	The system must allow a user with appropriate permissions to add a K9 role.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to add a new K9 role. These roles will define the responsibilities of each dog including tracking or attacking. These roles help to assign dogs to activities that match their abilities. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 94: 8.11 Add K9 Role - Description

Functional Requirement	Explanation
Requirement number:	8.12
Requirement name (use case name):	Update K9 Role
Requirement short description:	The system must allow a user with appropriate permissions to update a K9 role.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to update an existing K9's role on the system. The editable details will include the name or spelling of the role. Once the user clicks save, the system will validate the updated data and save the changes. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 95: 8.12 Update K9 Role - Description

Functional Requirement	Explanation
Requirement number:	8.13
Requirement name (use case name):	View K9 Roles
Requirement short description:	The system must allow a user with appropriate permissions to view K9 roles.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view a list of K9 roles on the system. These roles describe the responsibilities of each dog such as a tracking dog or attacking dog. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • K9 roles must be displayed accurately. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 96: 8.13 View K9 Role - Description

Functional Requirement	Explanation
Requirement number:	8.14
Requirement name (use case name):	Reactivate K9 Role
Requirement short description:	The system must allow a user with appropriate permissions to reactivate a K9 role.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to reactivate a K9 role on the system when it is required again. The system will confirm the reactivation and upon confirmation, the K9 role will be reactivated. Constraints: • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-21 Version 1.0
Criticality/Priority:	High

Table 97: 8.14 Reactivate K9 Role - Description

Functional Requirement	Explanation
Requirement number:	8.15
Requirement name (use case name):	Deactivate K9 Role
Requirement short description:	The system must allow a user with appropriate permissions to remove a K9 role.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to deactivate an existing K9 role from the system when it is no longer required. The system will confirm the deactivation and upon confirmation, the K9 role will be deactivated. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 98: 8.15 Deactivate K9 Role - Description

Functional Requirement	Explanation
Requirement number:	8.16
Requirement name (use case name):	View K9 Ancestry
Requirement short description:	The system must allow a user with appropriate permissions to view the ancestry of a K9.
Requirement detailed description and constraints:	The system must allow a user with appropriate permissions to view a dog's family tree showing the parentage and lineage of the dog. Constraints: • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Role based access. • User must be logged into the system.
Revision date and Revision number:	2026-03-21 Version 1.0
Criticality/Priority:	High

Table 99: 8.16 View K9 Ancestry - Description

2.3.9 Rhino Subsystem

Functional Requirement	Explanation
Requirement number:	9.1
Requirement name (use case name):	Add Rhino
Requirement short description:	The system allows users with appropriate permissions to add a rhino.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to store information about rhinos. The user will enter the required details including name, species, weight, date of birth, gender, horn status, status, last seen, identification notes, approximate location (kept vague for security reasons) and photo. **Constraints:** • Fields must pass input validation before being captured. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can add a rhino. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 100: 9.1 Add Rhino - Description

Functional Requirement	Explanation
Requirement number:	9.2
Requirement name (use case name):	View Rhino
Requirement short description:	The system allows users with appropriate permissions to view a rhino.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to access and view information about a rhino stored on the system. **Constraints:** • The rhino must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view a rhino's details. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 101:9.2 View Rhino - Description

Functional Requirement	Explanation
Requirement number:	9.3
Requirement name (use case name):	View Rhinos
Requirement short description:	The system allows users with appropriate permissions to view all rhinos.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to access and view all rhinos stored on the system. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view all rhinos. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 102: 9.3 View Rhinos - Description

Functional Requirement	Explanation
Requirement number:	9.4
Requirement name (use case name):	Update Rhino
Requirement short description:	The system allows users with appropriate permissions to update rhino details.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to update rhino details that are stored on the system. The system will display the rhino's current details and allow the user to change them. This includes name, species, weight, date of birth, gender, horn status, status, last seen, identification notes, approximate location (kept vague for security reasons) and photo. **Constraints:** • The rhino must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can update a rhino. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 103: 9.4 Update Rhino - Description

Functional Requirement	Explanation
Requirement number:	9.5
Requirement name (use case name):	Deactivate Rhino
Requirement short description:	The system allows users with appropriate permissions to remove a rhino; historic data will be kept.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to remove a rhino from the system. The user will get a confirmation prompt before the rhino is removed from the system. Historic rhino data will be kept. **Constraints:** • The rhino must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can deactivate a rhino. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 104: 9.5 9.5 Deactivate Rhino - Description

2.3.10. Elephant Subsystem

Functional Requirement	Explanation
Requirement number:	10.1
Requirement name (use case name):	Add Elephant
Requirement short description:	The system allows users with appropriate permissions to add an elephant.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to store information about elephants. The user will enter the required details including name, species, weight, date of birth, gender, tusk status, status, last seen, identification notes, approximate location (kept vague for security reasons) and photo. **Constraints:** • Fields must pass input validation before being captured. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can add an elephant. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 105: 10.1 Add Elephant - Description

Functional Requirement	Explanation
Requirement number:	10.2
Requirement name (use case name):	View Elephant
Requirement short description:	The system allows users with appropriate permissions to view an elephant's details.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to access and view information about an elephant stored on the system. **Constraints:** • The elephant must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view an elephant's details. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 106: 10.2 View Elephant - Description

Functional Requirement	Explanation
Requirement number:	10.3
Requirement name (use case name):	View Elephants
Requirement short description:	The system allows users with appropriate permissions to view all elephants.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to access and view all elephants stored on the system. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can view all elephants. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 107: 10.3 View Elephants - Description

Functional Requirement	Explanation
Requirement number:	10.4
Requirement name (use case name):	Update Elephant
Requirement short description:	The system allows users with appropriate permissions to update an elephant's details.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to update elephant details that are stored on the system. The system will display the elephant's current details and allow the user to change them. This includes the name, species, weight, date of birth, gender, tusk status, status, last seen, identification notes, approximate location (kept vague for security reasons) and photo. **Constraints:** • The elephant must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can update an elephant. • User must be logged in.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 108: 10.4 Update Elephant - Description

Functional Requirement	Explanation
Requirement number:	10.5
Requirement name (use case name):	Deactivate Elephant
Requirement short description:	The system allows users with appropriate permissions to deactivate an elephant; historic data will be kept.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to remove an elephant or elephant from the system. The user will get a confirmation prompt before the rhino or elephant is removed from the system. Historic data will be kept. **Constraints:** • Only users with appropriate permissions can remove an elephant. • The elephant must already exist on the system. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Only a user with appropriate permissions can remove an elephant. • User must be logged in.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 109: 10.5 Deactivate Elephant - Description

2.3.11 Message Subsystem

Functional Requirement	Explanation
Requirement number:	11.1
Requirement name (use case name):	Create Message
Requirement short description:	The system must allow a user to send a message on a group.
Requirement detailed description and constraints:	The system must allow a user to send a message on a group to which they are assigned. The user will type out the message they wish to send and then send the message to a group. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• User must be logged into the system.
Revision date and Revision number:	2026-03-24 Version 2.0
Criticality/Priority:	High

Table 110: 11.1 Create Message - Description

Functional Requirement	Explanation
Requirement number:	11.2
Requirement name (use case name):	View Messages
Requirement short description:	The system must allow a user to view the messages on a group.
Requirement detailed description and constraints:	The system must allow a user to view a message on a group to which they are assigned. The user selects a group and will be able to see the messages on the group. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• User must be logged into the system.
Revision date and Revision number:	2026-03-04 Version 1.0
Criticality/Priority:	High

Table 111: 11.2 View Message - Description

2.3.12 Report Subsystem

Functional Requirement	Explanation
Requirement number:	12.1
Requirement name (use case name):	Generate Custom Report
Requirement short description:	The system allows a user with appropriate permissions to generate a custom report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to generate a custom report. The user can select fields, and the system will generate a report using those fields. **Constraints:** • Only users with appropriate permissions can generate custom reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 112: 12.1 Generate Custom Report - Description

Functional Requirement	Explanation
Requirement number:	12.2
Requirement name (use case name):	View Custom Report
Requirement short description:	The system allows a user with appropriate permissions to view a custom report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view a custom report. The user can select and view a custom report that has been created. **Constraints:** • Only users with appropriate permissions can view custom reports. • Report must already exist. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 113: 12.2 View Custom Report - Description

Functional Requirement	Explanation
Requirement number:	12.3
Requirement name (use case name):	Update Custom Report
Requirement short description:	The system allows a user with appropriate permissions to update a custom report.
Requirement detailed and constraints:	The system will allow users with appropriate permissions to update a custom report. The system will display the report's current fields and allow the user to change them. **Constraints:** • Only users with appropriate permissions can update custom reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 114: 12.3 Update Custom Report - Description

Functional Requirement	Explanation
Requirement number:	12.4
Requirement name (use case name):	Remove Custom Report
Requirement short description:	The system allows a user with appropriate permissions to remove a custom report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to remove a custom report and its configuration from the database. **Constraints:** • Only users with appropriate permissions can remove custom reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-29 Version 3.0
Criticality/Priority:	High

Table 115: 12.4 Remove Custom Report - Description

Functional Requirement	Explanation
Requirement number:	12.5
Requirement name (use case name):	View Bike Impact Report
Requirement short description:	The system allows a user with appropriate permissions to view a bike impact report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view a bike impact report. The user Selects a date range, and the pre-structured report is generated for them to view. **Constraints:** • Only users with appropriate permissions can view bike impact reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-29 Version 3.0
Criticality/Priority:	High

Table 116: 12.5 View Bike Impact Report - Description

Functional Requirement	Explanation
Requirement number:	12.6
Requirement name (use case name):	View Budget Report
Requirement short description:	The system allows a user with appropriate permissions to view a budget report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view a budget report. The user Selects a date range, and the pre-structured report is generated for them to view. **Constraints:** • Only users with appropriate permissions can view budget reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-29 Version 3.0
Criticality/Priority:	High

Table 117: 12.6 View Budget Report - Description

Functional Requirement	Explanation
Requirement number:	12.7
Requirement name (use case name):	View Rhino Report
Requirement short description:	The system allows a user with appropriate permissions to view a rhino report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view a rhino report. The user Selects a date range, and the pre-structured report is generated for them to view. **Constraints:** • Only users with appropriate permissions can view rhino reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-29 Version 3.0
Criticality/Priority:	High

Table 118: 12.7 View Rhino Report - Description

Functional Requirement	Explanation
Requirement number:	12.8
Requirement name (use case name):	View Elephant Report
Requirement short description:	The system allows a user with appropriate permissions to view an elephant report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view an elephant report. The user Selects a date range, and the pre-structured report is generated for them to view. **Constraints:** • Only users with appropriate permissions can view elephant reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-29 Version 3.0
Criticality/Priority:	High

Table 119: 12.8 View Elephant Report - Description

Functional Requirement	Explanation
Requirement number:	12.9
Requirement name (use case name):	View K9 Report
Requirement short description:	The system allows a user with appropriate permissions to view a K9 report.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions to view a K9 report. The user Selects a date range, and the pre-structured report is generated for them to view. **Constraints:** • Only users with appropriate permissions can view K9 reports. • Users must be logged in.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 120: 12.9 View K9 Report - Description

2.3.13 Machine Learning Subsystem

Functional Requirement	Explanation
Requirement number:	13.1
Requirement name (use case name):	View Machine Learning Dashboard
Requirement short description:	A user with the appropriate role should be able to view all current predictions.
Requirement detailed description and constraints:	A user with the appropriate role should be able to view all current predictions. Predictions will appear dynamically as users complete tasks and will also be viewable in the Machine Learning Dashboard. Once a user opens the dashboard, the model will give all predictions based on current data. This will include team predictions for ongoing projects, budget predictions for ongoing projects and risk predictions for all ongoing projects **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Intelligent Business Insights.
Revision date and Revision number:	2026-03-30 Version 1.0
Criticality/Priority:	Medium

Table 121: 13.1 View Machine Learning Dashboard - Description

2.3.14 System Subsystem

Functional Requirement	Explanation
Requirement number:	14.1
Requirement name (use case name):	Backup System
Requirement short description:	The system must allow a user with appropriate permissions to back up the system.
Requirement detailed description and constraints:	The system will allow users with appropriate permissions make a backup of the system data to ensure that information can be recovered in case of a system failure or data loss. The system backup can be done manually or automatically at scheduled intervals. **Constraints:** • Only users with appropriate permissions can perform system backups.
Business rules applicable to this requirement	• Role-based access. • Ability to backup database.
Revision date and Revision number:	2026-03-05 Version 2.0
Criticality/Priority:	High

Table 122: 14.1 Backup System - Description

Functional Requirement	Explanation
Requirement number:	14.2
Requirement name (use case name):	Restore System
Requirement short description:	The user should be able to restore data on the system.
Requirement detailed description and constraints:	The user should be able to restore data on the system to a previously backed up version. All users will be logged out by the system before the operation takes place. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • Only a Super Administrator can access this feature.
Business rules applicable to this requirement	• Role-based access. • Ability to backup database.
Revision date and Revision number:	2026-03-04 Version 2.0
Criticality/Priority:	Medium

Table 123: 14.2 Restore System - Description

Functional Requirement	Explanation
Requirement number:	14.3
Requirement name (use case name):	View Audit Log
Requirement short description:	An administrator should be able to view the audit log
Requirement detailed description and constraints:	An administrator should be able to view the audit log to see who made which important changes when. This allows them to easily identify the source of misuse. **Constraints:** • Time constraint: the use case must be completed before the Project Day in October 2026. • User skill constraint: the use case must be easy to understand and use, as some users will have limited technology abilities. • The user requires an Internet connection to access this feature. • Only an Administrator can access this feature.
Business rules applicable to this requirement	• Role-based access.
Revision date and Revision number:	2026-03-30 Version 1.0
Criticality/Priority:	Medium

Table 124: 14.3 View Audit Log - Description

2.3.15. Budget Subsystem

Functional Requirement	Explanation
Requirement number:	15.1
Requirement name (use case name):	Create Project Budget
Requirement short description:	A user with the appropriate permissions should be able to add a project budget.
Requirement detailed description and constraints:	A user with the appropriate permissions should be able to add a project budget to any active project. The user will select an active project and be able to enter a budget. The system will validate inputted data and ask for confirmation before creating the project budget. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 125: 15.1 Create Project Budget - Description

Functional Requirement	Explanation
Requirement number:	15.2
Requirement name (use case name):	View Project Budget
Requirement short description:	A user with the appropriate permissions should be able to view a project budget.
Requirement detailed description and constraints:	A user with the appropriate permissions should be able to view a project budget. The system will allow the user to select and view a budget's details including expenses. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 126: 15.2 View Project Budget - Description

Functional Requirement	Explanation
Requirement number:	15.3
Requirement name (use case name):	View Project Budgets
Requirement short description:	A user with the appropriate permissions should be able to view all project budgets.
Requirement detailed description and constraints:	A user with the appropriate permissions should be able to view all project budgets. The system will retrieve all project budgets on the system and display them to the user. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 127: 15.3 View Project Budgets - Description

Functional Requirement	Explanation
Requirement number:	15.4
Requirement name (use case name):	Update Project Budget
Requirement short description:	A user with the appropriate permissions should be able to update a project budget.
Requirement detailed description and constraints:	A user with the appropriate permissions should be able to update a project budget in the edge case that a mistake was made when entering the details, or the projects financing has changed. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 128: 15.4 Update Project Budget - Description

Functional Requirement	Explanation
Requirement number:	15.5
Requirement name (use case name):	Log An Expense
Requirement short description:	A user with appropriate permissions should be able to log an expense.
Requirement detailed description and constraints:	A user with appropriate permissions should be able to log an expense. The system will display the project's budget and allow the user to add an expense by entering the expenses details and the system will validate the inputted data before asking for confirmation **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 129: 15.5 Log An Expense - Description

Functional Requirement	Explanation
Requirement number:	15.6
Requirement name (use case name):	Update An Expense
Requirement short description:	A user with appropriate permissions should be able to update an expense.
Requirement detailed description and constraints:	A user with appropriate permissions should be able to update an expense in the edge case that information is entered incorrectly. The system will display the expense's details and allow the user to update its details. The user will enter the updated details and the system will validate the updated details before asking the user to confirm the updated details **Constraints:** • The user requires an Internet connection to access this feature. • The expense should already exist on the system.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 130: 15.6 Update An Expense - Description

Functional Requirement	Explanation
Requirement number:	15.7
Requirement name (use case name):	View An Expense
Requirement short description:	A user with appropriate permissions should be able to view an expense.
Requirement detailed description and constraints:	A user with appropriate permissions should be able to view an expense. The system will display the project budget and allow the user to select an expense to view. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 131: 15.7 View An Expense - Description

Functional Requirement	Explanation
Requirement number:	15.8
Requirement name (use case name):	Remove An Expense
Requirement short description:	A user with the appropriate permissions should be able to remove an expense.
Requirement detailed description and constraints:	A user with the appropriate permissions should be able to remove an expense. The system will display the projects budget and expenses and allow the user to remove an expense. The system will ask for confirmation before the expense is removed. **Constraints:** • The user requires an Internet connection to access this feature.
Business rules applicable to this requirement	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Revision date and Revision number:	2026-03-29 Version 2.0
Criticality/Priority:	High

Table 132: 15.8 Remove An Expense - Description

2.4 Functional Requirements List Modifications

Previous Requirement	Current Requirement	Reason For Modification
1.1 User Login	1.1 User Login	-
1.2 User Logout	1.2 User Logout	-
1.3 Reset Password	1.3 Reset Password	-
1.4 Create User	1.4 Create User	-
1.5 Update User	1.5 Update User	-
1.6 View User	1.6 View User	-
	1.7 Reactivate User	Added to allow administrators to reactivate accounts.
1.7 Deactivate User	1.8 Deactivate User	Numbering change
1.8 View FAQ	1.9 View FAQ	Numbering change
2.1 Create Role	2.1 Create User Profile	Numbering change and made Admin and Role one subsystem
2.2 View Role	2.2 Update User Profile	Numbering change and made Admin and Role one subsystem
2.3 Update Role	2.3 View User Profile	Numbering change and made Admin and Role one subsystem
2.4 Delete Role	2.4 Deactivate User Profile	Numbering change and made Admin and Role one subsystem
2.5 Create Role Permission	2.5 Add FAQ	Numbering change and made Admin and Role one subsystem
2.6 Update Role Permission	2.6 Update FAQ	Numbering change and made Admin and Role one subsystem
2.7 Deactivate Role Permission	2.7 Remove FAQ	Numbering change and made Admin and Role one subsystem
-	2.8 Create Role	Numbering change
-	2.9 View Role	Numbering change
-	2.10 Update Role	Numbering change
-	2.11 Reactivate Role	Added to allow administrator to reactivate role
-	2.12 Deactivate Role	Added to allow administrator to deactivate role
-	2.13 Create Role Permission	Numbering change
-	2.14 Update Role Permission	Numbering change
-	2.15 View Role Permission	Numbering change
-	2.16 Deactivate Role Permission	Numbering change
3.1 Create Inventory Item	3.1 Receive Inventory	Changed name for more clarity on what the use case does.
3.2 View Inventory Item	3.2 View Inventory	Changed name as the use case views all inventory on the system.

Previous Requirement	Current Requirement	Reason For Modification
3.3 Update Inventory Item	[REMOVED]	Inventory items do not need to be updated. It will just be tracked.
3.4 Remove Inventory Item	3.3 Write-Off Inventory	Numbering change as well as changing name to write-off for better inventory tracking.
	3.4 Inventory Stock Take	Added to allow the super admin better track inventory stock.
3.5 Allocate Inventory Item	3.5 Allocate Inventory	Name changes for better clarity.
3.6 Create Inventory Item Type	3.6 Create Inventory Item Type	-
3.7 View Inventory Item Type	3.7 View Inventory item Type	-
	3.8 View Inventory Item Types	Allows the super admin to view all the inventory item types on the system.
3.8 Update Inventory Item Type	3.9 Update Inventory Item Type	Numbering change
3.9 Deactivate Inventory Item Type	3.10 Deactivate Inventory Item Type	Numbering change
4.1 Create Group	4.2 Create Group	-
4.2 View Group	4.2 View Groups	Name change because the system displays all the groups for a project at once.
4.3 Update Group	4.3 Update Group	Changed description to better match the logical narrative.
4.4 Deactivate Group	4.4 Deactivate Group	-
4.5 Create Skill	4.5 Create Skill	-
4.6 View Skill	4.6 View Skills	Name change because the system displays all the skills in a table at once. Also changed the description to better match the logical narrative.
4.7 Update Skill	4.7 Update Skill	-
4.8 Deactivate Skill	4.8 Deactivate Skill	-
5.1 Create Warehouse	5.1 Create Warehouse	-
5.2 View Warehouse	5.2 View Warehouse	-
5.3 Update Warehouse	5.3 Update Warehouse	-
5.4 Reactivate Warehouse	5.4 Reactivate Warehouse	Added to allow the Super Admin to reactivate warehouse.
5.5 Calculate Total Warehouse Content Value	5.5 Deactivate Warehouse	Removed

Previous Requirement	Current Requirement	Reason For Modification
-	5.5 Deactivate Warehouse	Number change
18.1 Create Vendor	6.1 Create Vendor	Numbering change
18.4 View Vendor	6.2 View Vendor	Numbering change
	6.3 View Vendors	Allow super admins and admins to view all vendors.
18.2 Edit Vendor	6.4 Edit Vendor	Numbering change
	6.5 Reactivate Vendor	Added to allow super admins to reactivate previously deactivated vendors on the system.
18.3 Deactivate Vendor	6.6 Deactivate Vendor	Numbering change
18.5 Create Vendor Type	6.7 Create Vendor Type	Number change and changed description to better match the logical narrative.
18.6 Update Vendor Type	6.8 Update Vendor Type	Number change and changed description to better match the logical narrative.
18.7 View Vendor Type	6.9 View Vendor Types	Number change and changed description to better match the logical narrative.
[ADDED]	6.10 Reactivate Vendor Type	Added after Iteration 1 because if you can deactivate a vendor type, you should be able to reactivate it.
18.8 Deactivate Vendor Type	6.11 Deactivate Vendor Type	Number change and changed description to better match the logical narrative.
9.1 Add Dog	8.1 Add Dog	Number change and changed description to better match the logical narrative.
9.2 View Dog	8.2 View Dog	Number change and changed description to better match the logical narrative.
[ADDED]	8.3 View Dogs	Added after Iteration 1 because before you can view a single dog, all the dogs are shown.
9.3 Update Dog	8.4 Update Dog	Number change.
9.4 Remove Dog	8.5 Deactivate Dog	Number and name change. Also changed the description to better match the logical narrative.
9.5 Assign User to Dog	8.6 Assign User to Dog	Number change.
9.6 View User-Dog Assignment	8.7 View User-Dog Assignment	Number change.
9.7 Update User-Dog Assignment	8.8 Update User-Dog Assignment	Number change.
[ADDED]	8.9 Reactivate User-Dog Assignment	Added after Iteration 1 because if you can deactivate a user-dog assignment, you should be able to reactivate it.
9.8 Deactivate User-Dog Assignment	8.10 Deactivate User-Dog Assignment	Number change.
9.9 Add K9 Role	8.11 Add K9 Role	Number change.

Previous Requirement	Current Requirement	Reason For Modification
9.10 Update K9 Role	8.12 Update K9 Role	Number change.
9.11 View K9 Role	8.13 View K9 Role	Name change because the system displays all the K9 roles in a table at once.
[ADDED]	8.14 Reactivate K9 Role	Added after Iteration 1 because if you can deactivate a K9 role, you should be able to reactivate it.
9.12 Deactivate K9 Role	8.15 Deactivate K9 Role	Number change.
[ADDED]	8.16 View Ancestry	Added after Iteration 1 because the group wanted to be able to view a dog's ancestry which can help with deciding the dog's role.
10.1 Add Rhino	9.1 Add Rhino	Numbering change
10.2 View Rhino	9.2 View Rhino	Numbering change
	9.3 View Rhinos	Allow the super admin to view all the rhinos on the system.
10.3 Update Rhino	9.4 Update Rhino	Numbering change
10.4 Deactivate Rhino	9.5 Deactivate Rhino	Numbering change
11.1 Capture Bike Location	[REMOVED]	Removed as bike tracking was a low priority and not a requirement for the client.
11.2 Deactivate Bike Location	[REMOVED]	Removed as bike tracking was a low priority and not a requirement for the client.
11.3 View Bike Location	[REMOVED]	Removed as bike tracking was a low priority and not a requirement for the client.
11.1 Add Elephant	10.1 Add Elephant	Numbering change
11.2 View Elephant	10.2 View Elephant	Numbering change
	10.3 View Elephants	Allow the super admin to view all the elephants on the system.
11.3 Update Elephant	10.4 Update Elephant	Numbering change
11.4 Deactivate Elephant	10.5 Deactivate Elephant	Numbering change
[ADDED]	11.1 Create Message	Added after Iteration 1 to allow the users of the system to send messages on groups.
[ADDED]	11.2 View Messages	Added after Iteration 1 to allow the users of the system to view messages on groups.
17.1 Generate Custom Report	12.1 Generate Custom Report	Numbering change
17.2 View Custom Report	12.2 View Custom Report	Numbering change
17.3 Update Custom Report	12.3 Update Custom Report	Numbering change
17.4 Deactivate Custom Report	12.4 Remove Custom Report	Changed to remove, as it is not necessary to keep historic data of custom reports.

Previous Requirement	Current Requirement	Reason For Modification
17.5 Generate Bike Impact report	[REMOVED]	Integrated use case into View Bike Impact Report use case.
17.6 View Bike Impact report	12.5 View Bike Impact report	Numbering change
17.7 Generate Budget Report	[REMOVED]	Integrated use case into View Budget Report use case.
17.8 View Budget Report	12.6 View Budget Report	Numbering change
17.9 Generate Rhino Report	[REMOVED]	Integrated use case into View Rhino Report use case.
17.10 View Rhino Report	12.7 View Rhino Report	Numbering change
17.11 Generate Elephant Report	[REMOVED]	Integrated use case into View Elephant Report use case.
17.12 View Elephant Report	12.8 View Elephant Report	Numbering change
17.13 Generate K9 Report	[REMOVED]	Integrated use case into View K9 Report use case.
17.14 View K9 Report	12.9 View K9 Report	Numbering change
17.15 Export Report	[REMOVED]	Integrated into view report use cases.
19.1 Create Project Risk Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
19.2 View Project Risk Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
19.3 Create Team Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
19.4 View Team Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
19.5 Create Budget Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
19.6 View Budget Prediction	[REMOVED]	Integrated into other use cases and available via the Machine Learning Dashboard.
	13.1 View Machine Learning Dashboard	Allow users to see all current ML predictions in one place.
20.1 Backup System	14.1 Backup System	Numbering Change
20.2 Restore System	14.2 Restore System	Numbering Change
	14.3 View Audit Log	Added to allow administrators to easily view all audit log transactions.
13.1 Create Project Budget	15.1 Create Project Budget	Numbering change
13.3 View Project Budget	15.2 View Project Budget	Numbering change
	15.3 View Project Budgets	Allow the super admin to view all project budgets on the system.

Previous Requirement	Current Requirement	Reason For Modification
13.2 Update Project Budget	15.4 Update Project Budget	Numbering change
13.4 Deactivate Project Budget	[REMOVED]	No longer used as the budget will be deactivated when a project is deactivated.
13.5 Log An Expense	15.5 Log An Expense	Numbering change
13.6 Update An Expense	15.6 Update An Expense	Numbering change
13.7 View Expense	15.7 View Expense	Numbering change
13.8 Deactivate AN Expense	15.8 Remove An Expense	Numbering change as well as changed to remove as an expense should be removed if not used and no record needs to be kept.

Table 133: Functional Requirements List Modifications

2.5 Use Case Diagram

2.5.1 Use Case Diagram - User Subsystem

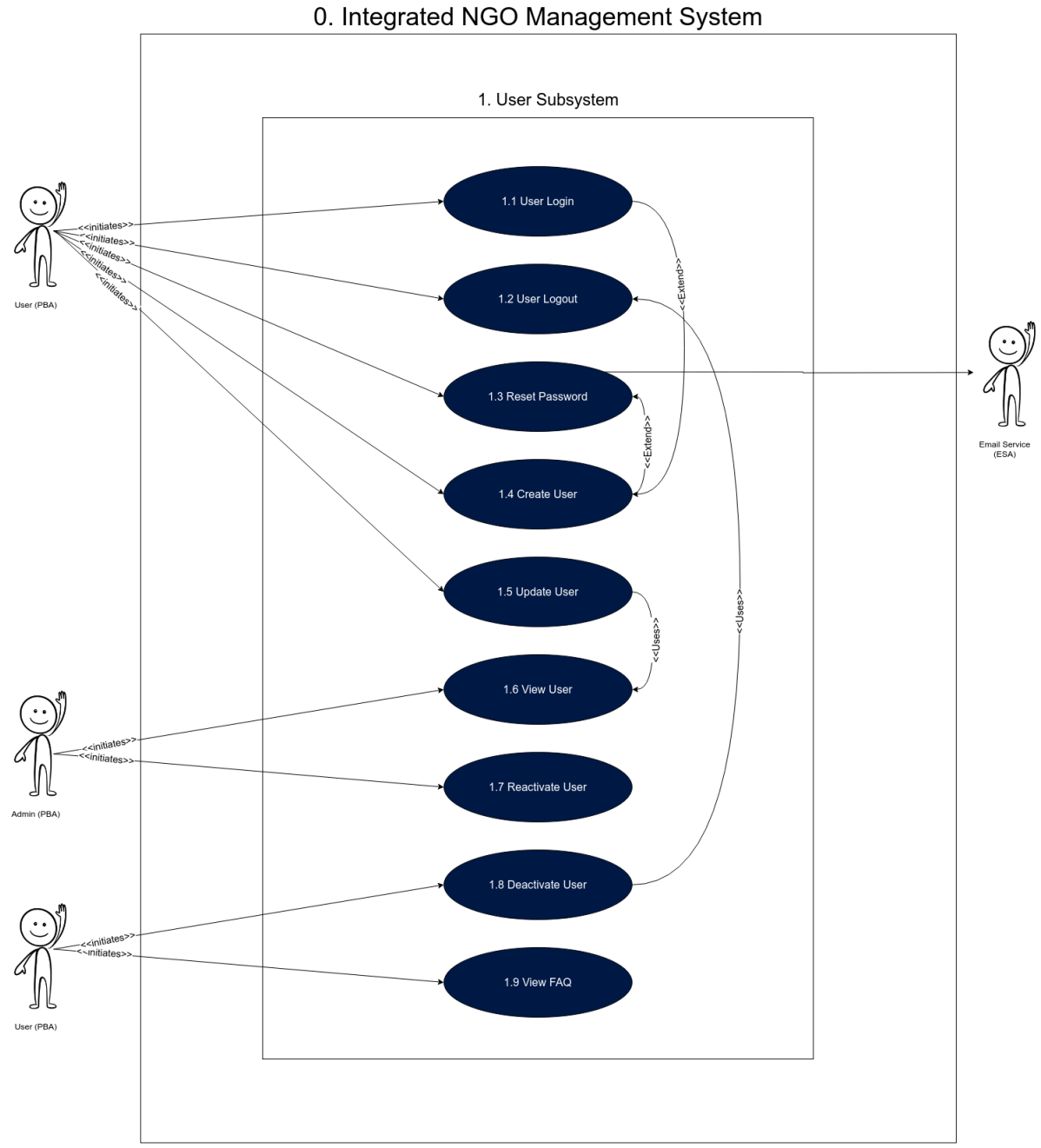


Figure 1: User Subsystem - Use Case Diagram

2.5.2 Use Case Diagram – Admin Subsystem

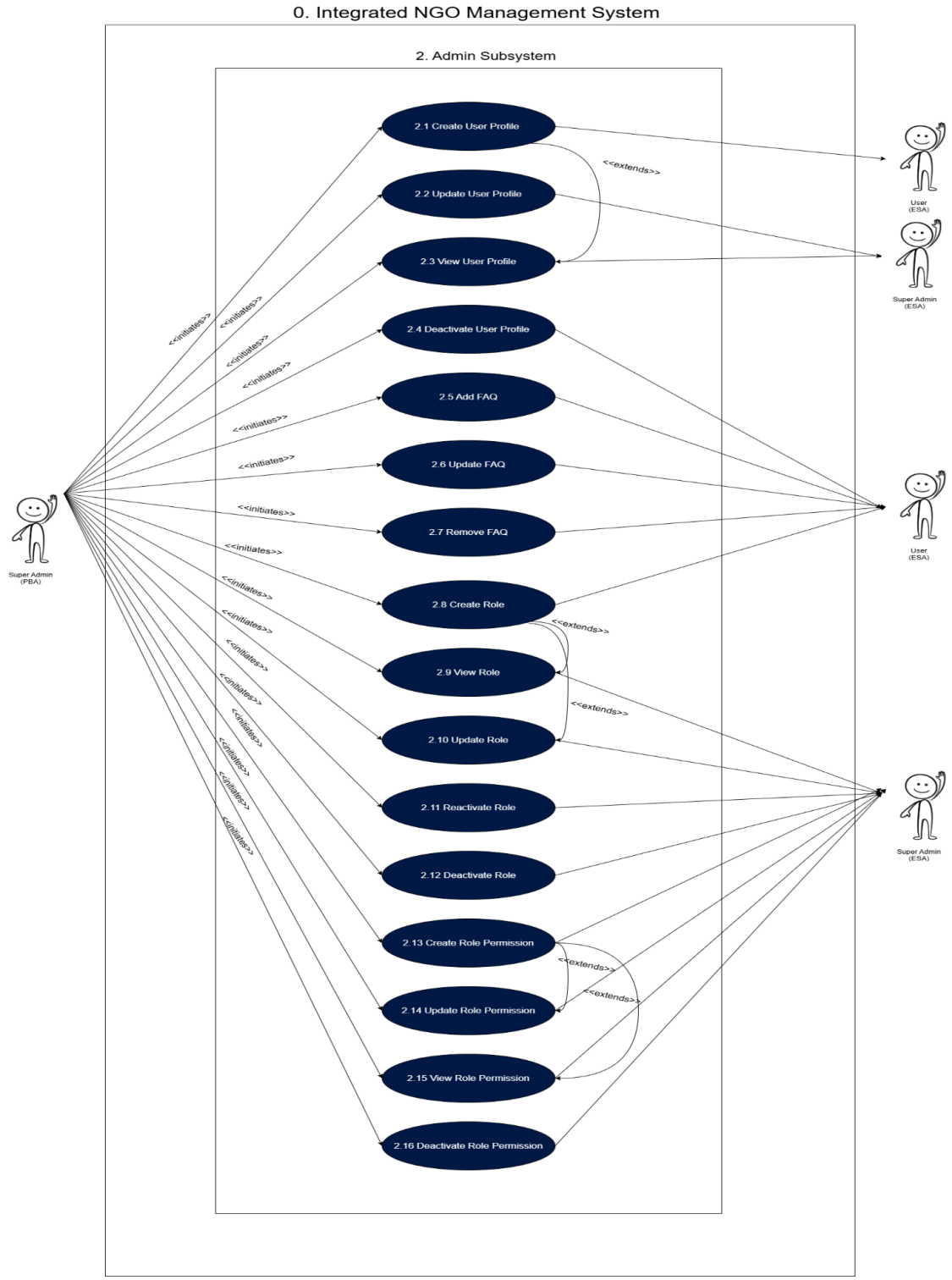


Figure 2: Admin Subsystem - Use Case Diagram

2.5.3 Use Case Diagram – Inventory Management Subsystem

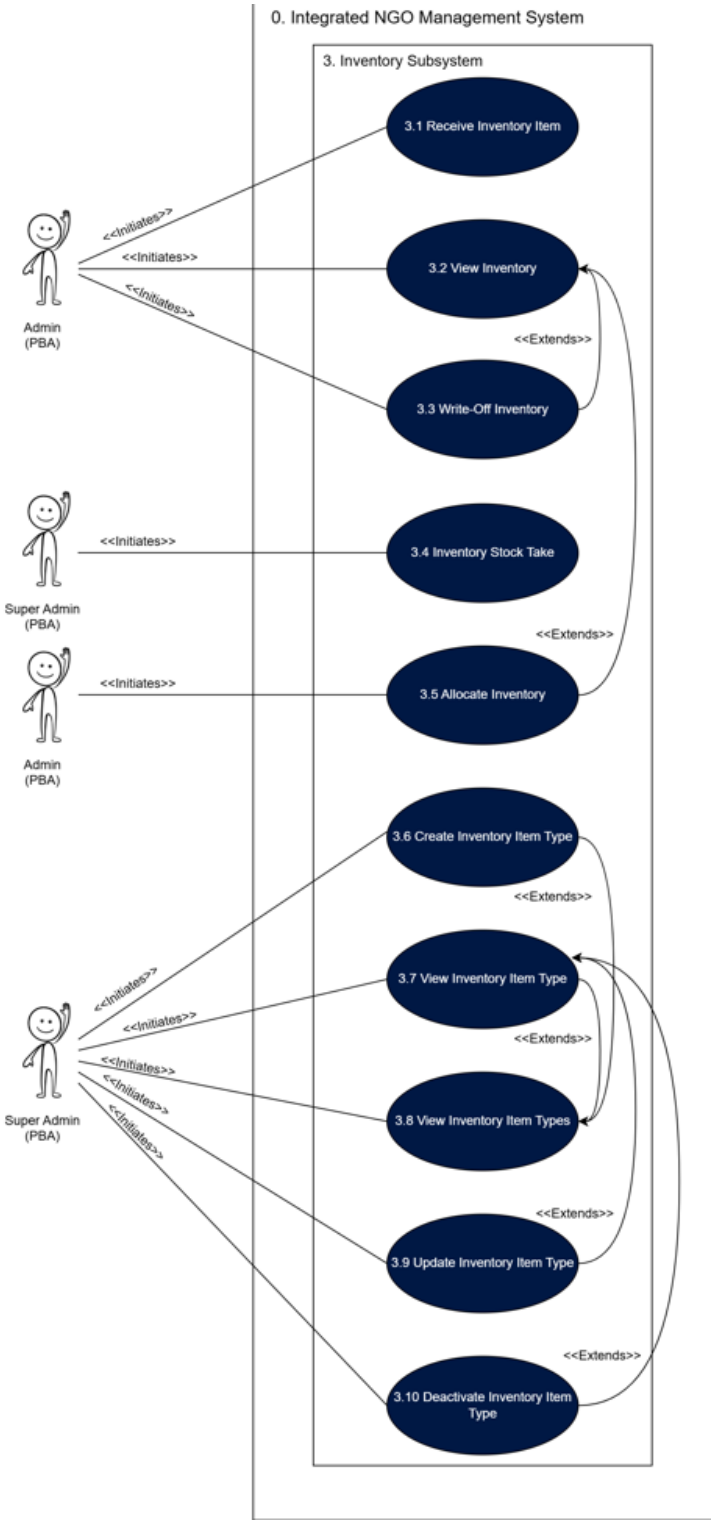


Figure 3: Inventory Subsystem - Use Case Diagram

2.5.4 Use Case Diagram - Group Subsystem

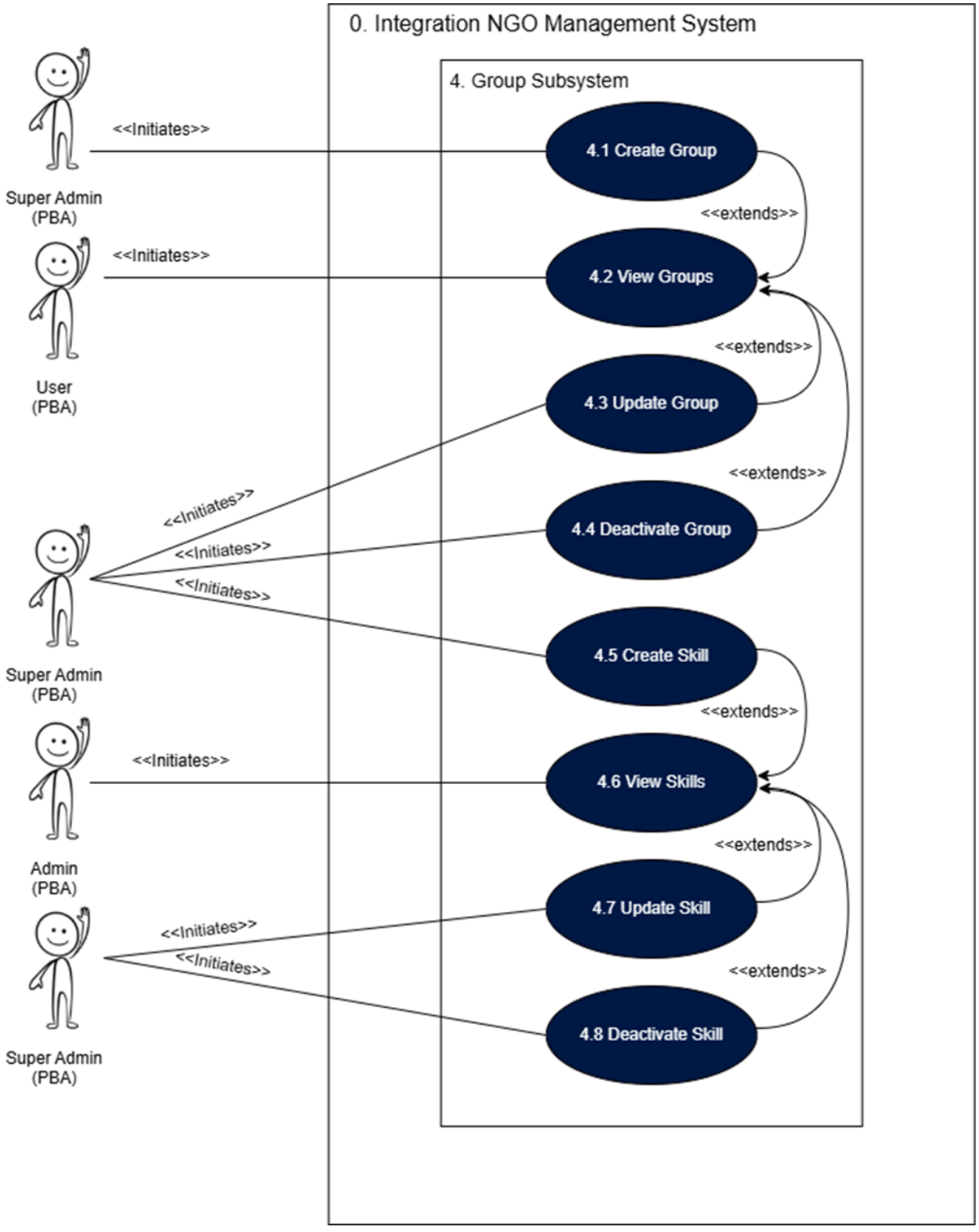


Figure 4:Group Subsystem - Use Case Diagram

2.5.5 Use Case Diagram - Warehouse Subsystem

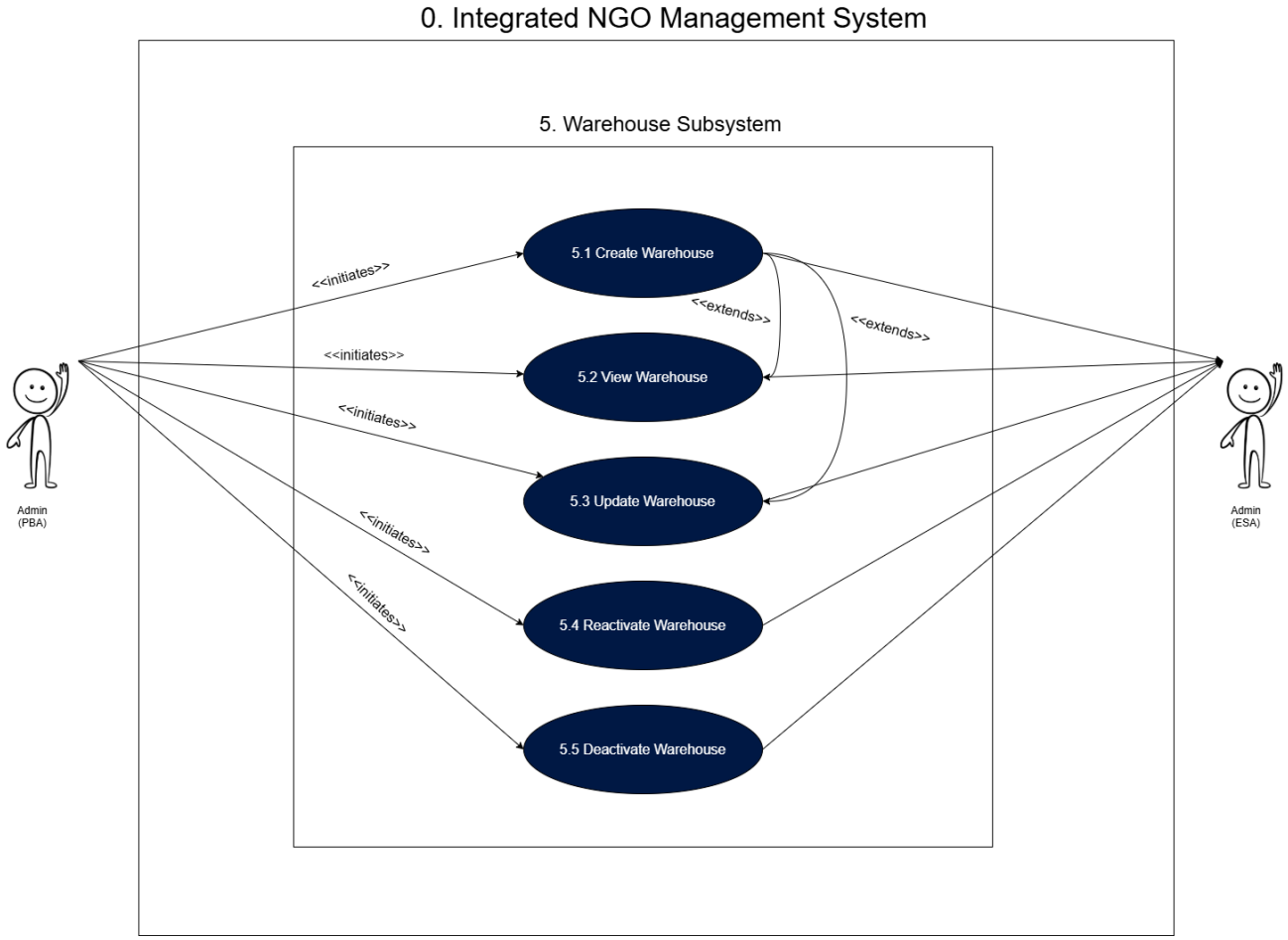


Figure 5: Warehouse Subsystem - Use Case Diagram

2.5.6 Use Case Diagram – Vendor Subsystem

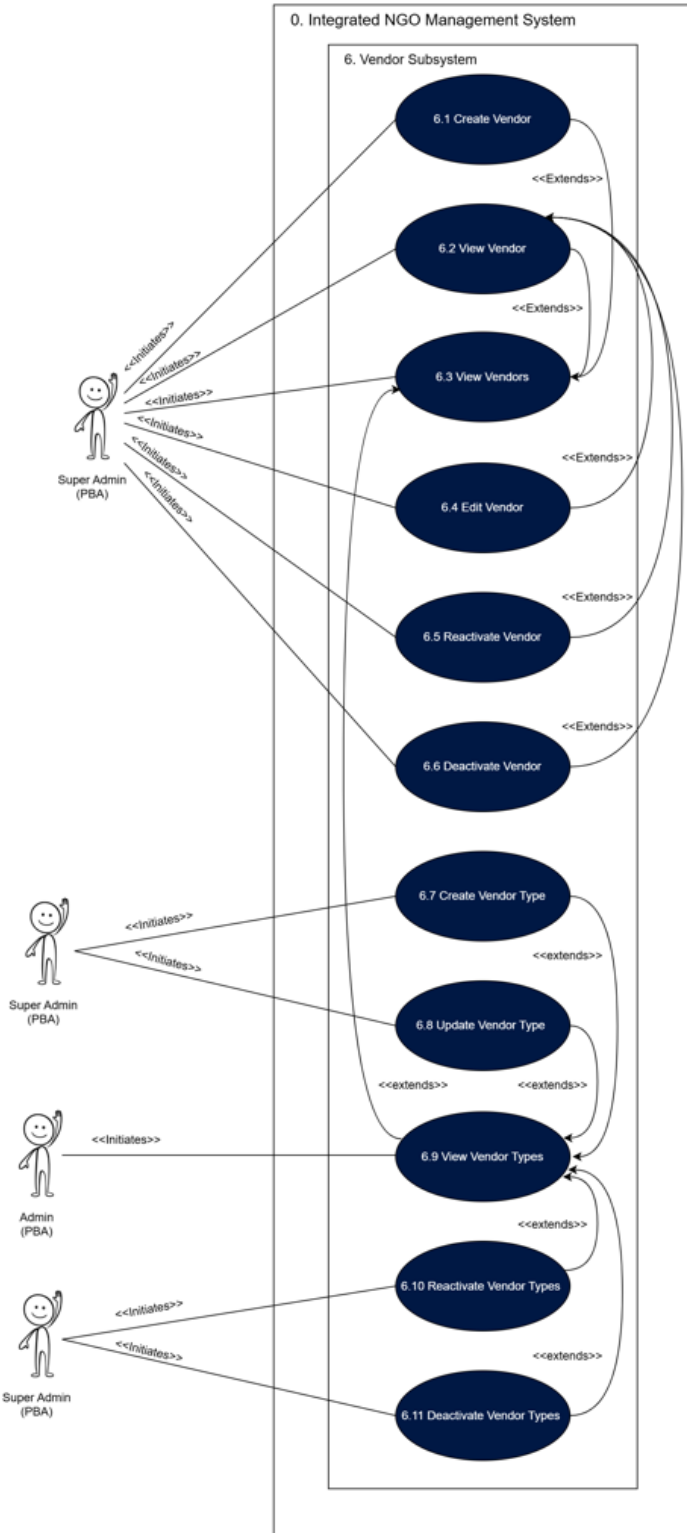


Figure 6:Vendor Subsystem - Use Case Diagram

2.5.7 Use Case Diagram – Project Management Subsystem



Figure 7: Vendor Management Subsystem - Use Case Diagram

2.5.8 Use Case Diagram – K9 Subsystem

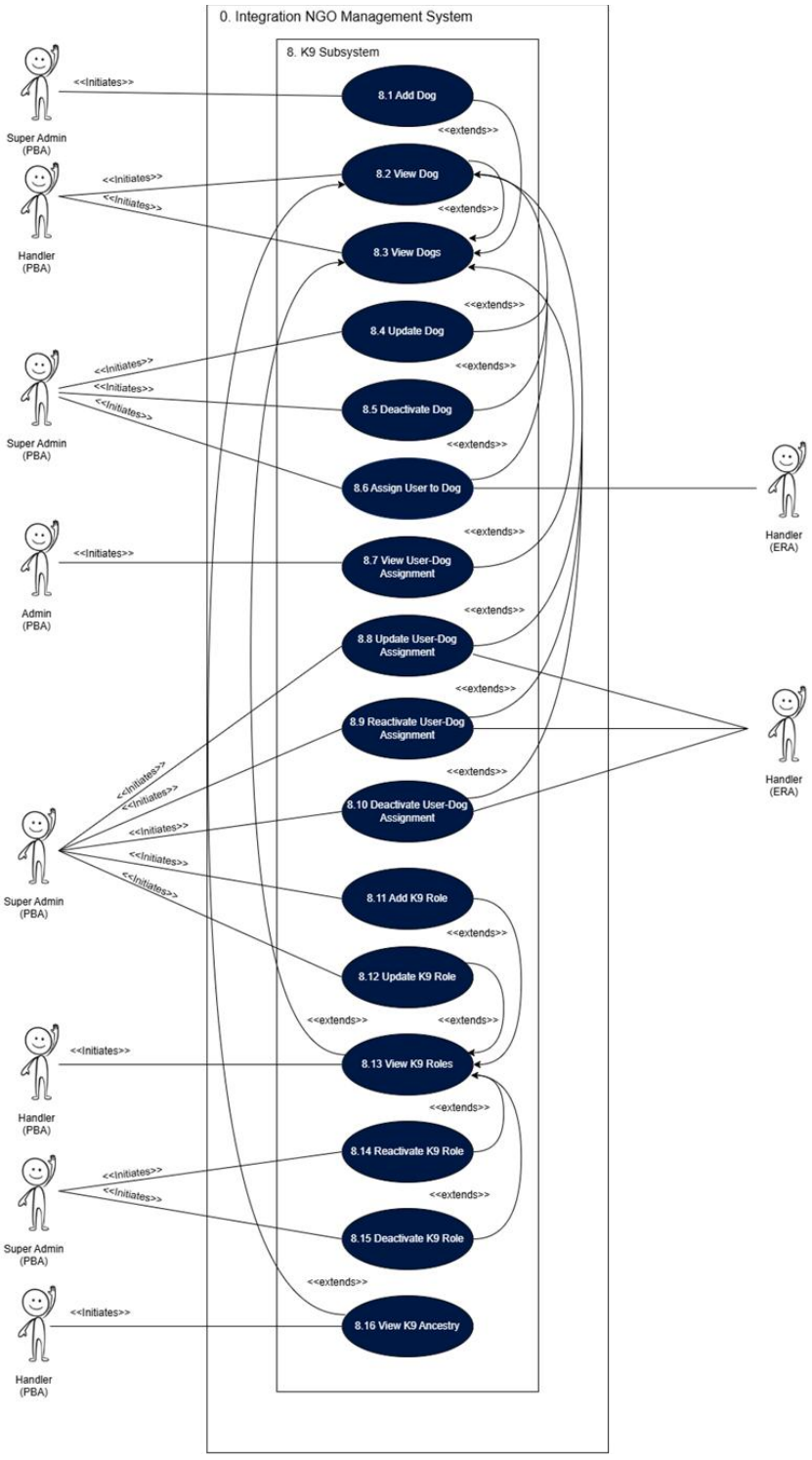


Figure 8: K9 Subsystem - Use Case Diagram

2.5.9 Use Case Diagram – Rhino Subsystem

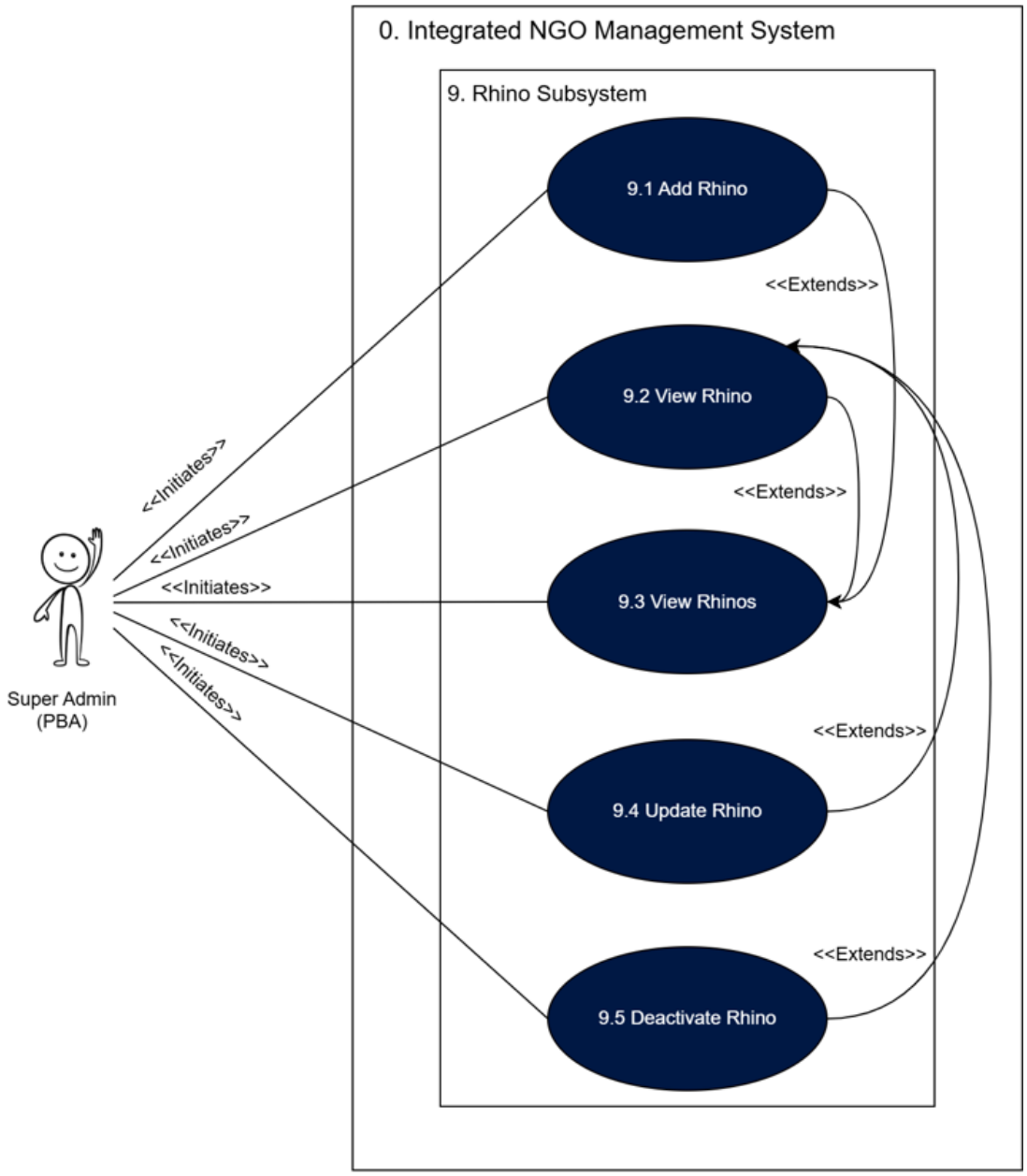


Figure 9: Rhino Subsystem - Use Case Diagram

2.5.10 Use Case Diagram – Elephant Subsystem

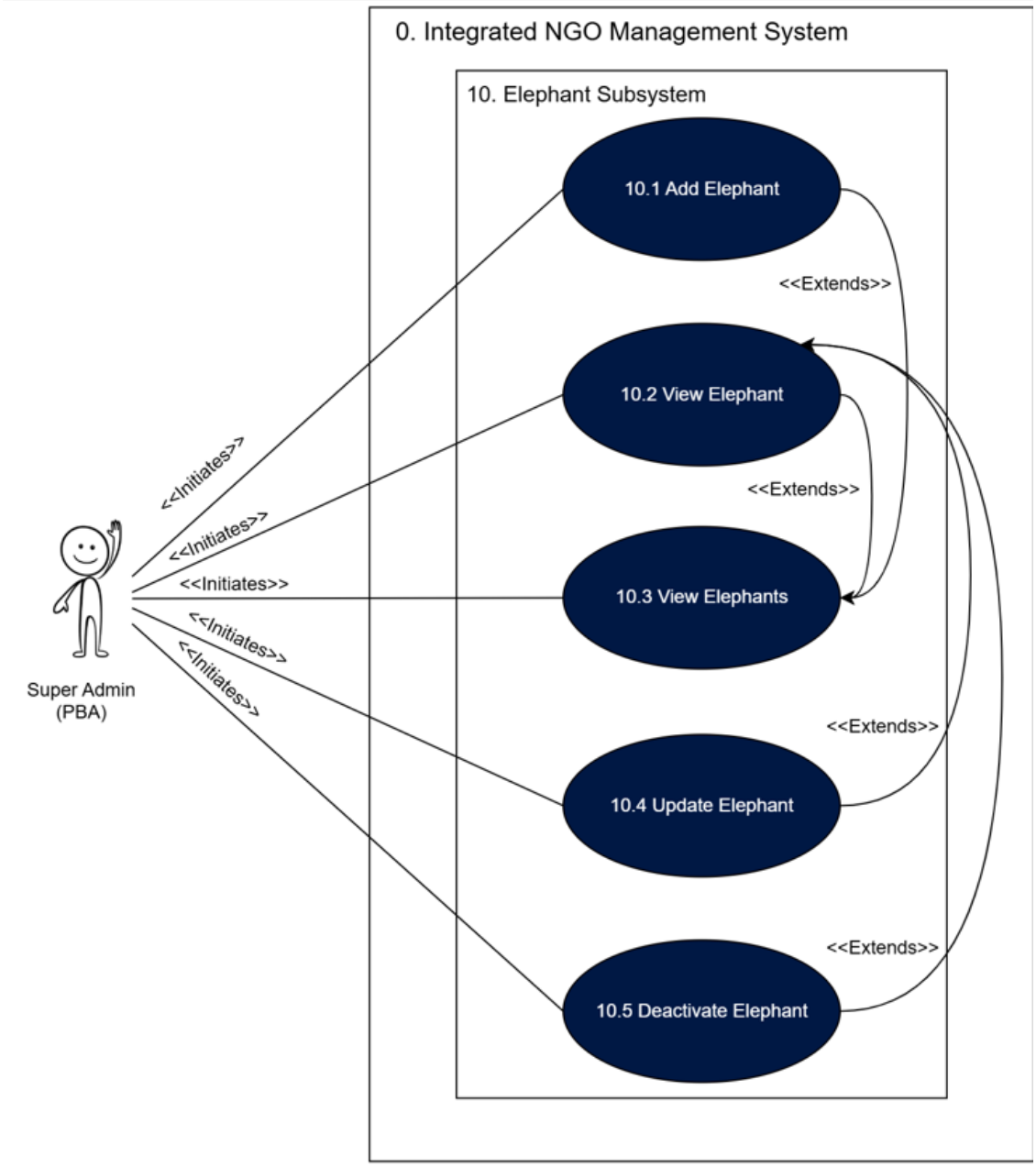


Figure 10: Elephant Subsystem - Use Case Diagram

2.5.11 Use Case Diagram – Message Subsystem

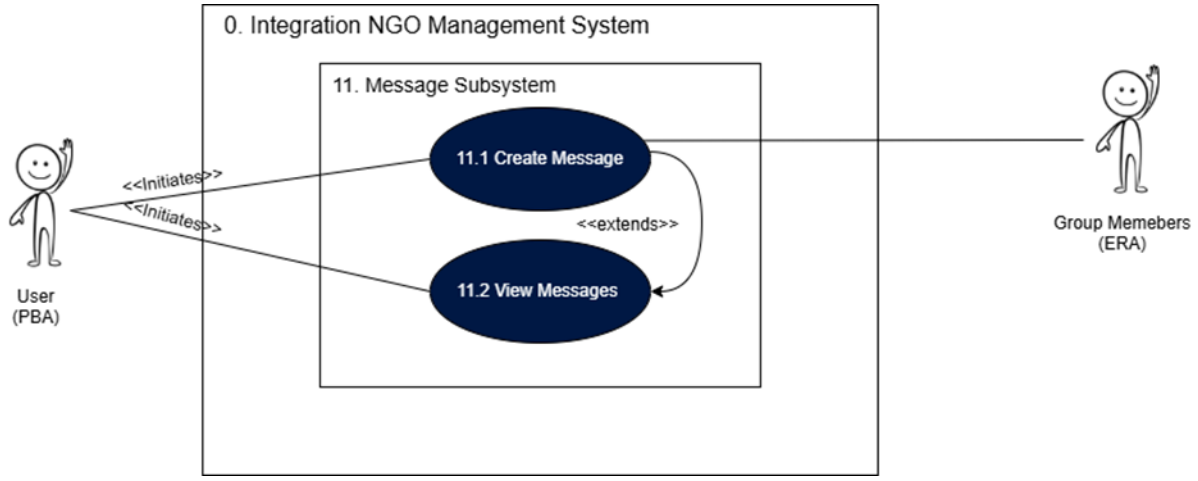


Figure 11: Message Subsystem - Use Case Diagram

2.5.12 Use Case Diagram – Report Subsystem

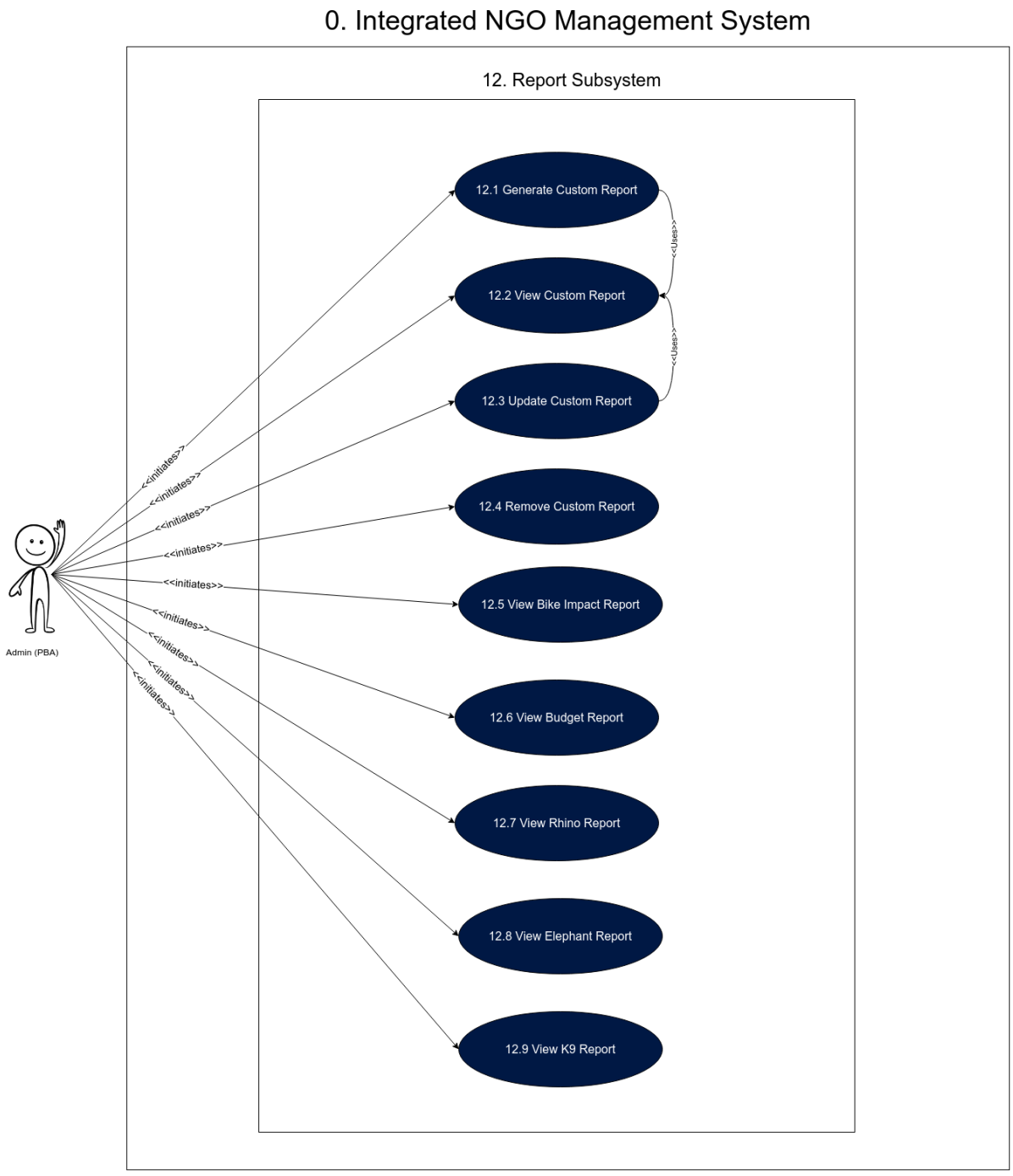


Figure 12: Report Subsystem - Use Case Diagram

2.5.13 Use Case Diagram – Machine Learning Subsystem

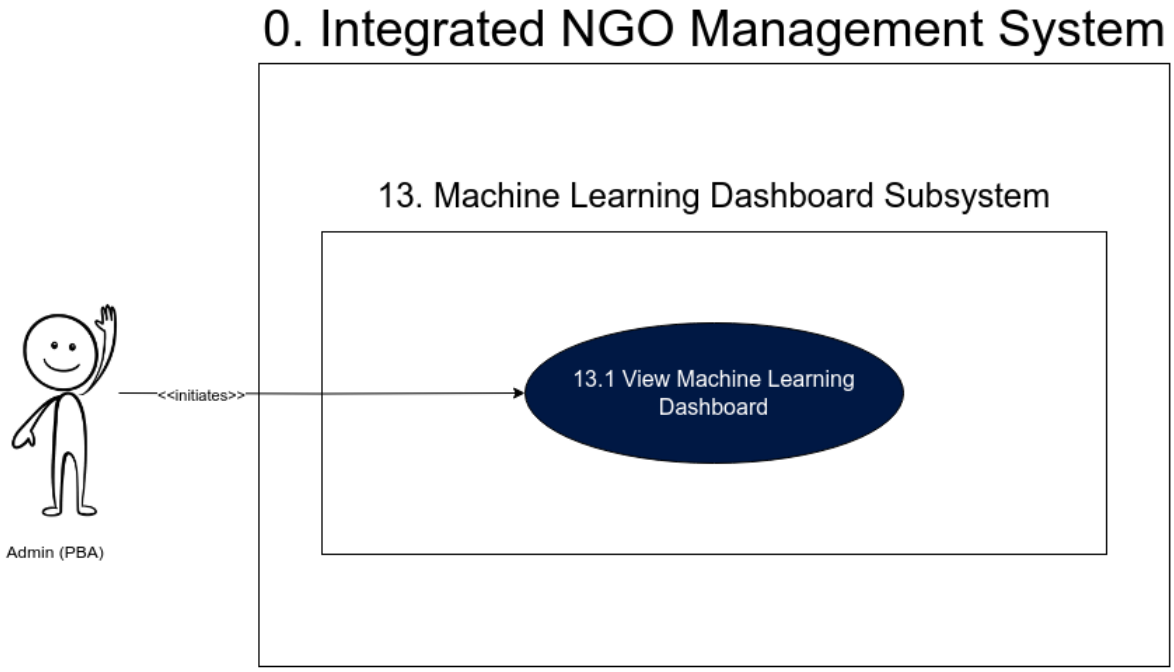


Figure 13: Machine Learning Subsystem - Use Case Diagram

2.5.14 Use Case Diagram – System Subsystem

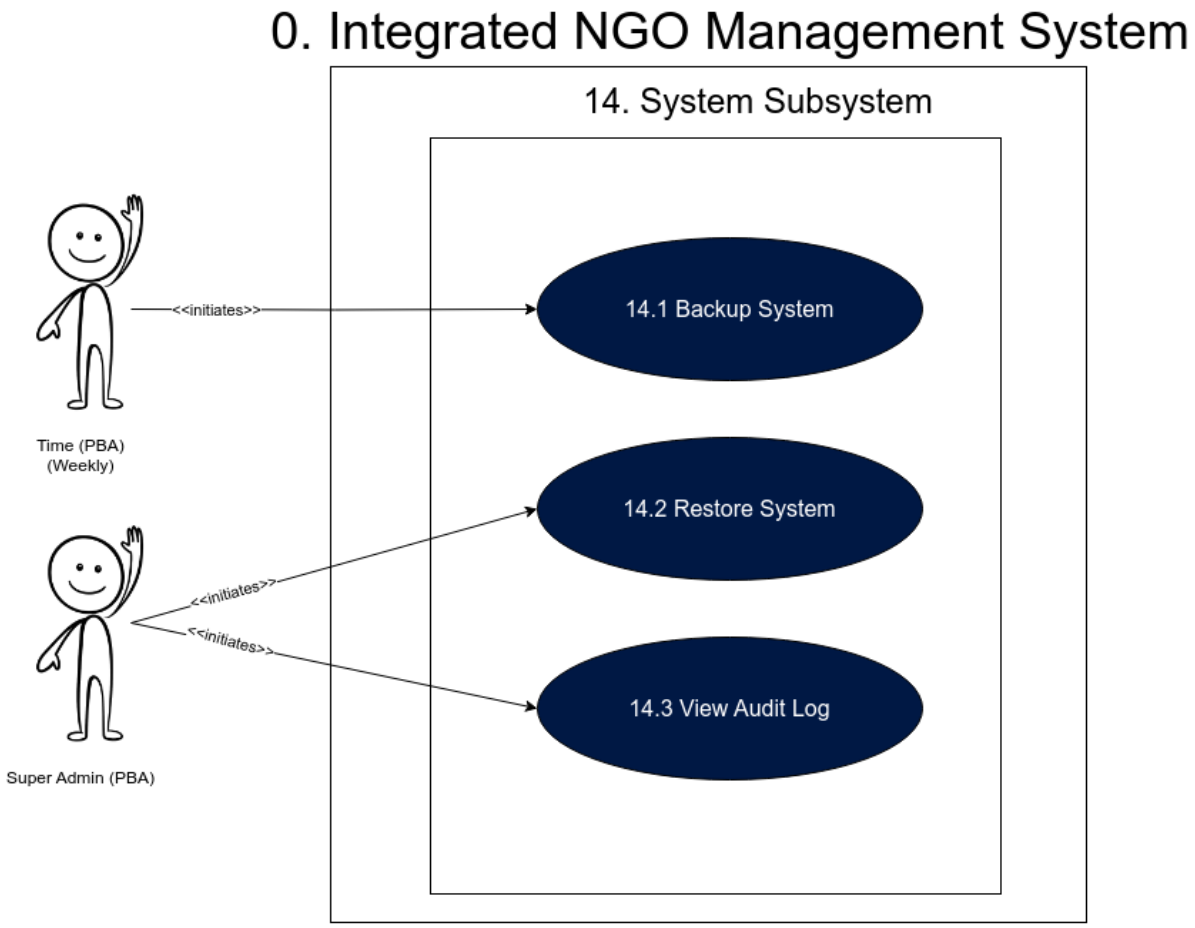


Figure 14: System Subsystem - Use Case Diagram

2.5.15 Use Case Diagram – Budget Subsystem

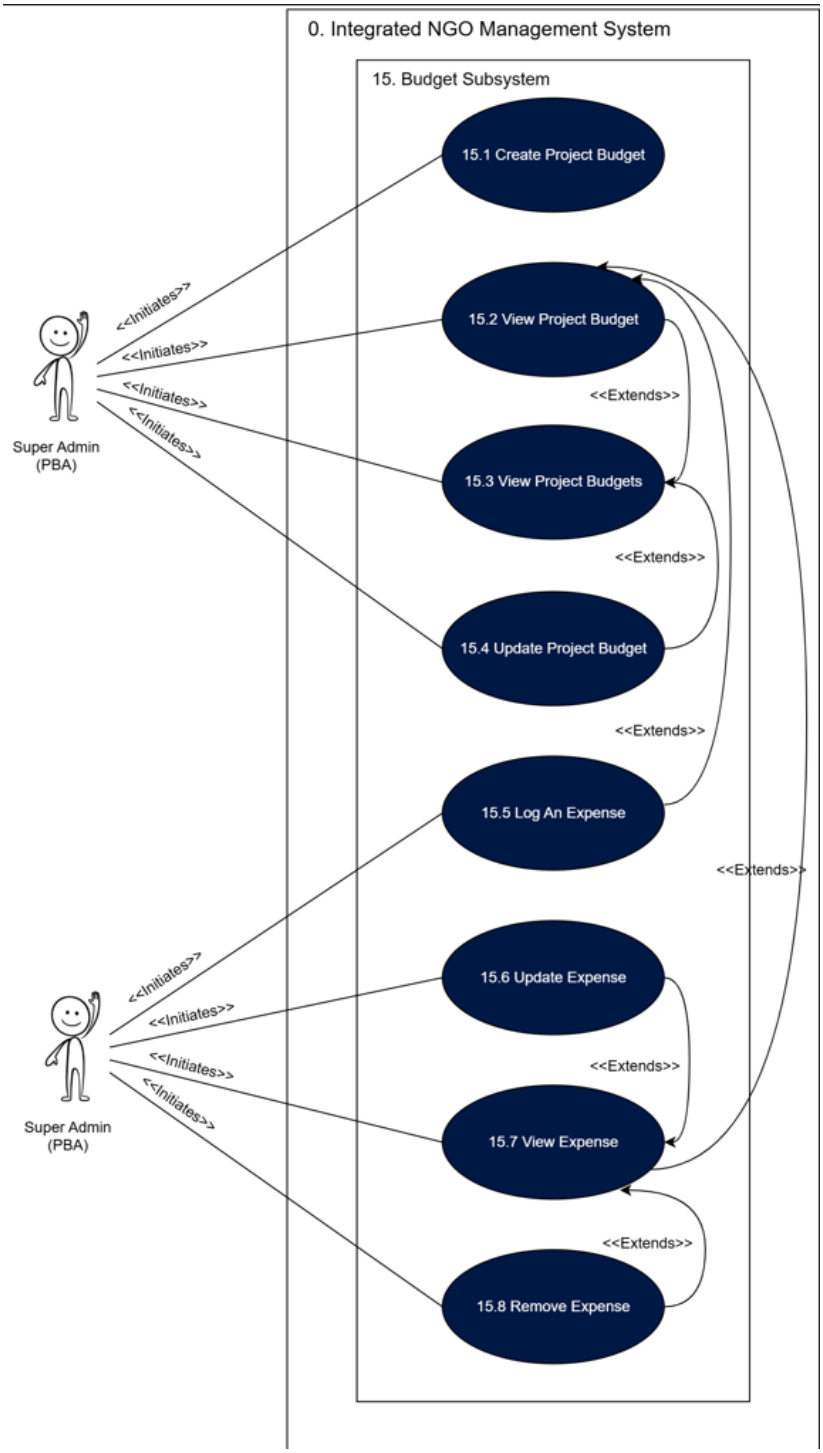


Figure 15: Budget Subsystem - Use Case Diagram

2.6 Logical Entity Relationship Diagram

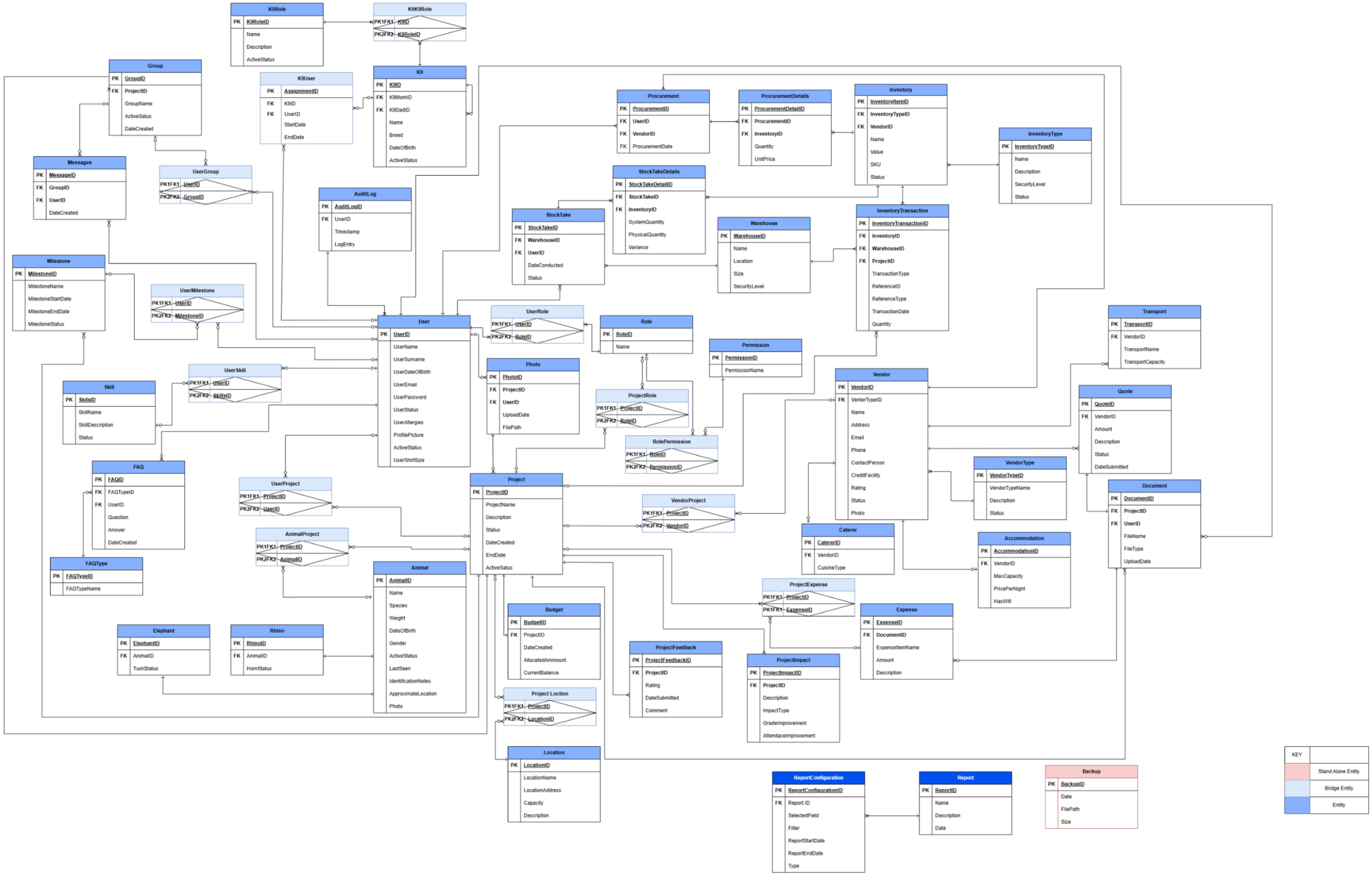


Figure 16: Entity Relationship Diagram

2.7 Decomposition Diagram

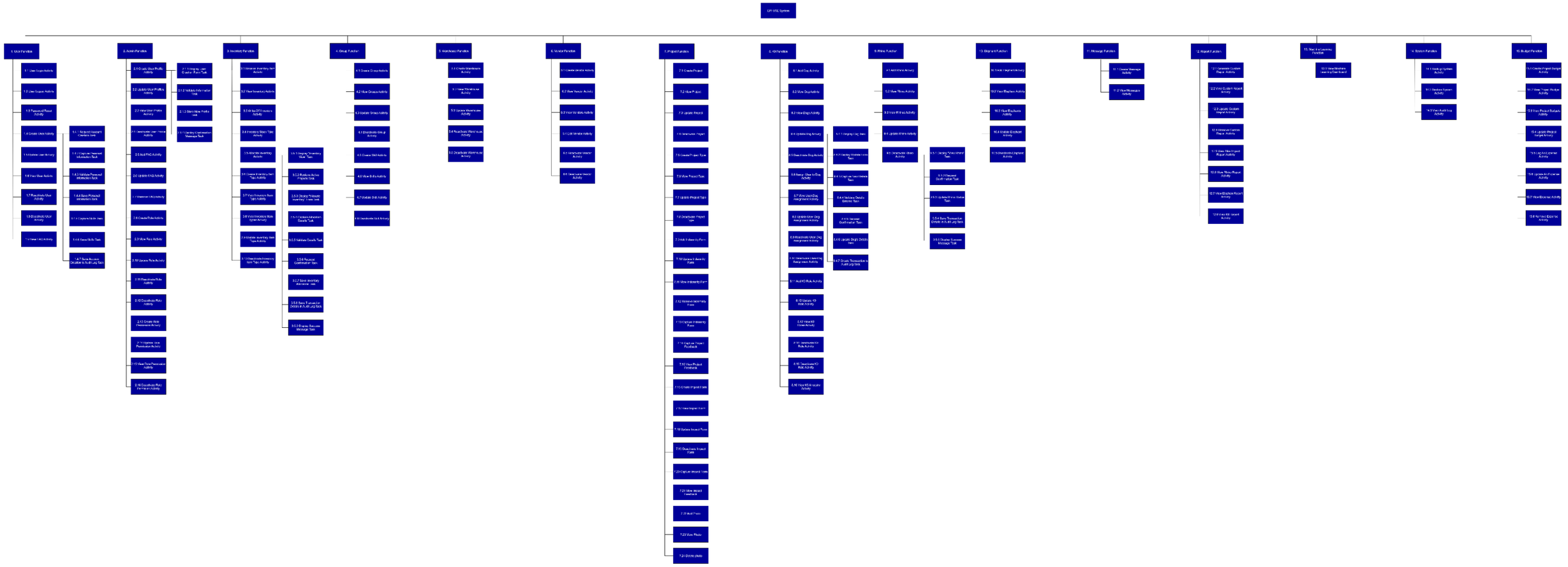
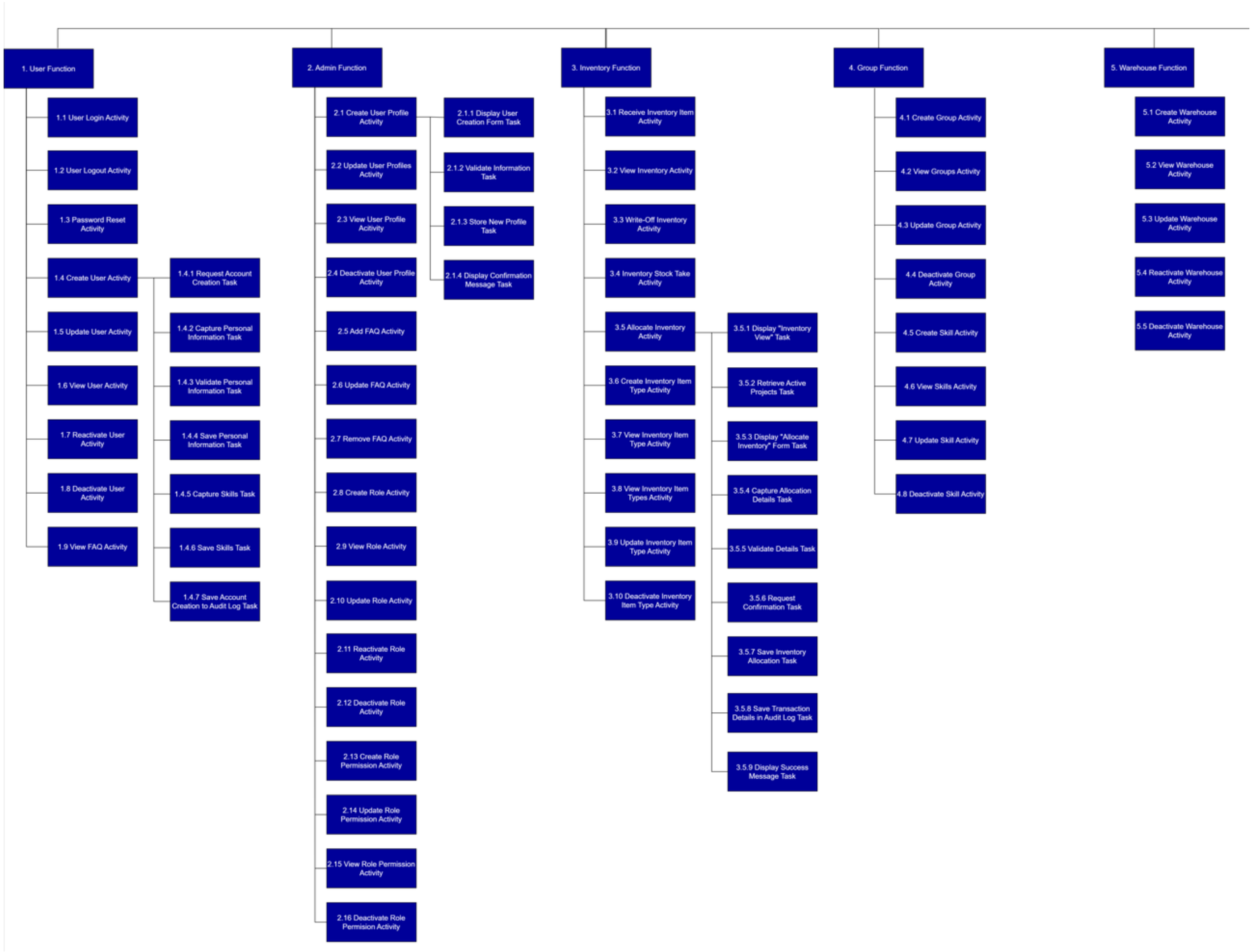
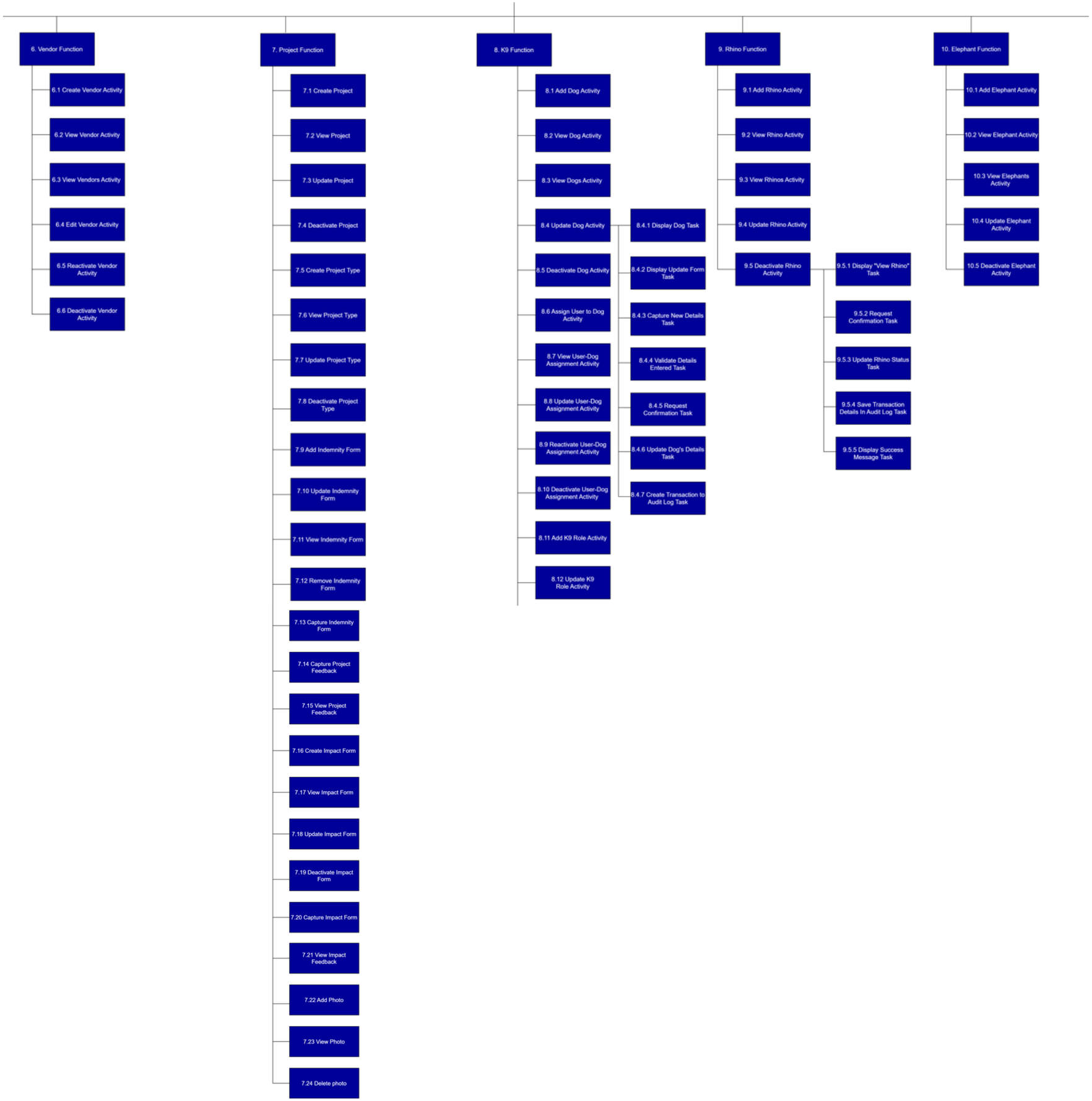


Figure 17 Decomposition Diagram


1. User Function
1.1 User Login Activity
1.2 User Logout Activity
1.3 Password Reset Activity
1.4 Create User Activity
1.4.1 Request Account Creation Task
1.4.2 Capture Personal Information Task
1.4.3 Validate Personal Information Task
1.4.4 Save Personal Information Task
1.4.5 Capture Skills Task
1.4.6 Save Skills Task
1.4.7 Save Account Creation to Audit Log Task
1.5 Update User Activity
1.6 View User Activity
1.7 Reactivate User Activity
1.8 Deactivate User Activity
1.9 View FAQ Activity
2. Admin Function
2.1 Create User Profile Activity
2.1.1 Display User Creation Form Task
2.1.2 Validate Information Task
2.1.3 Store New Profile Task
2.1.4 Display Confirmation Message Task
2.2 Update User Profiles Activity
2.3 View User Profile Activity
2.4 Deactivate User Profile Activity
2.5 Add FAQ Activity
2.6 Update FAQ Activity
2.7 Remove FAQ Activity
2.8 Create Role Activity
2.9 View Role Activity
2.10 Update Role Activity
2.11 Reactivate Role Activity
2.12 Deactivate Role Activity
2.13 Create Role Permission Activity
2.14 Update Role Permission Activity
2.15 View Role Permission Activity
2.16 Deactivate Role Permision Activity
3. Inventory Function
3.1 Receive Inventory Item Activity
3.2 View Inventory Activity
3.3 Write-Off Inventory Activity
3.4 Inventory Stock Take Activity
3.5 Allocate Inventory Activity
3.5.1 Display "Inventory View" Task
3.5.2 Retrieve Active Projects Task
3.5.3 Display "Allocate Inventory" Form Task
3.5.4 Capture Allocation Details Task
3.5.5 Validate Details Task
3.5.6 Request Confirmation Task
3.5.7 Save Inventory Allocation Task
3.5.8 Save Transaction Details in Audit Log Task
3.5.9 Display Success Message Task
3.6 Create Inventory Item Type Activity
3.7 View Inventory Item Type Activity
3.8 View Inventory Item Types Activity
3.9 Update Inventory Item Type Activity
3.10 Deactivate Inventory Item Type Activity
4. Group Function
4.1 Create Group Activity
4.2 View Groups Activity
4.3 Update Group Activity
4.4 Deactivate Group Activity
4.5 Create Skill Activity
4.6 View Skills Activity
4.7 Update Skill Activity
4.8 Deactivate Skill Activity
5. Warehouse Function
5.1 Create Warehouse Activity
5.2 View Warehouse Activity
5.3 Update Warehouse Activity
5.4 Reactivate Warehouse Activity
5.5 Deactivate Warehouse Activity


6. Vendor Function
6.1 Create Vendor Activity
6.2 View Vendor Activity
6.3 View Vendors Activity
6.4 Edit Vendor Activity
6.5 Reactivate Vendor Activity
6.6 Deactivate Vendor Activity
7. Project Function
7.1 Create Project
7.2 View Project
7.3 Update Project
7.4 Deactivate Project
7.5 Create Project Type
7.6 View Project Type
7.7 Update Project Type
7.8 Deactivate Project Type
7.9 Add Indemnity Form
7.10 Update Indemnity Form
7.11 View Indemnity Form
7.12 Remove Indemnity Form
7.13 Capture Indemnity Form
7.14 Capture Project Feedback
7.15 View Project Feedback
7.16 Create Impact Form
7.17 View Impact Form
7.18 Update Impact Form
7.19 Deactivate Impact Form
7.20 Capture Impact Form
7.21 View Impact Feedback
7.22 Add Photo
7.23 View Photo
7.24 Delete photo
8. K9 Function
8.1 Add Dog Activity
8.2 View Dog Activity
8.3 View Dogs Activity
8.4 Update Dog Activity
8.4.1 Display Dog Task
8.4.2 Display Update Form Task
8.4.3 Capture New Details Task
8.4.4 Validate Details Entered Task
8.4.5 Request Confirmation Task
8.4.6 Update Dog's Details Task
8.4.7 Create Transaction to Audit Log Task
8.5 Deactivate Dog Activity
8.6 Assign User to Dog Activity
8.7 View User-Dog Assignment Activity
8.8 Update User-Dog Assignment Activity
8.9 Reactivate User-Dog Assignment Activity
8.10 Deactivate User-Dog Assignment Activity
8.11 Add K9 Role Activity
8.12 Update K9 Role Activity
9. Rhino Function
9.1 Add Rhino Activity
9.2 View Rhino Activity
9.3 View Rhinos Activity
9.4 Update Rhino Activity
9.5 Deactivate Rhino Activity
9.5.1 Display "View Rhino" Task
9.5.2 Request Confirmation Task
9.5.3 Update Rhino Status Task
9.5.4 Save Transaction Details In Audit Log Task
9.5.5 Display Success Message Task
10. Elephant Function
10.1 Add Elephant Activity
10.2 View Elephant Activity
10.3 View Elephants Activity
10.4 Update Elephant Activity
10.5 Deactivate Elephant Activity

- 11. Message Function
  - 11.1 Create Message Activity
  - 11.2 View Messages Activity
- 12. Report Function
  - 12.1 Generate Custom Report Activity
  - 12.2 View Custom Report Activity
  - 12.3 Update Custom Report Activiry
  - 12.4 Remove Custom Report Activity
  - 12.5 View Bike Impact Report Activity
  - 12.6 View Rhino Report Activity
  - 12.7 View Elephan Report Activity
  - 12.8 View K9 Report Activity
- 13. Machine Learning Function
  - 13.1 View Machine Learning Dashboard
- 14. System Function
  - 14.1 Backup System Activity
  - 14.2 Restore System Activity
  - 14.3 View Audit Log Activity
- 15. Budget Function
  - 15.1 Create Project Budget Activity
  - 15.2 View Project Budget Activity
  - 15.3 View Project Budgets Activity
  - 15.4 Update Project Budget Activity
  - 15.5 Log An Expense Activity
  - 15.6 Update An Expense Activity
  - 15.7 View Expense Activity
  - 15.8 Remove Expense Activity

2.8 Logical Use Case Narratives

2.8.1 User Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.1	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	User Login	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A user logs into the system using their credentials (email & password) to access a role-specific dashboard and gain access to all parts of the system they have permission to access.	
Pre-condition:	The user must already have a pre-existing account on the system	
Trigger:	The super administrator requests to create a backup of the database.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user enters their email address and password into the respective fields on the login screen and selects the “Login” button.	**Step 2:** The system retrieves the record associated with the provided email address form the User table. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize [ALT]

System Name: Integrated NGO Management System		
		Step 3: The system hashes the password provided and compares it to the hashed password stored in the User table.
		Step 4: Upon finding a successful match, the system retrieves the user's role from the UserRole table. [ALT]
		Step 5: The system redirects she user to the appropriate dashboard.
		Step 6: Save login action to the Audit Log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 2: No record is found with a matching email address; the system suggests that the user sign's up (Use Case 1.4).	
	[ALT] Step 4: The hashed passwords do not match, and an error message is displayed. The user is redirected back to the login screen.	
Conclusion:	User is successfully authenticated and granted access.	
Post-condition:	An active session is created for the user.	
Business rules:	• Secure Access • Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 134: 1.1 User Login - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.2	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	User Logout	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A user logs out to terminate their session, preventing unauthorized physical access via their device.	
Pre-condition:	The user must be currently logged into the system.	
Trigger:	The user clicks the "Logout" button.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the Logout button.	**Step 2:** The system terminates the current user session.
		Step 3: The system redirects the user to the login screen.
Alternate courses:	None	
Conclusion:	Session is terminated successfully.	
Post-condition:	Access to the system is revoked until the user logs in.	
Business rules:	• Secure Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 135: 1.2 User Logout - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.3	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Reset Password	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None.	
Other participating actors:	• Email Service	
Other interested stakeholders:	Administrator	
Description:	A user resets their password by receiving a link via email and entering a new password in the case that they forgot the password or believe that an unauthorized actor has access to their credentials.	
Pre-condition:	User must have an existing account.	
Trigger:	User clicks "Forgot Password" on the login page.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: User enters the email address associated to their profile on the password reset screen and selects the “Reset Password” button.	**Step 2:** The system searches for the email address in the User table to ensure that an account is associated with it. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize [ALT]
		Step 3: Once the email is confirmed, the system informs the user to check their email address.

System Name: Integrated NGO Management System		
		Step 4: The system sends a reset password link to the confirmed email address.
	Step 5: The user opens the link they received and enters their desired new password in the provided field.	**Step 6:** The system hashes the new password and compares it to the Password field in the user table to ensure the old and new passwords do not match.
		Step 7: The system updates the Password field for the specific user in the User table. [ALT]
		Step 8: The system redirects the user to the Login screen.
		Step 9: Save password reset action to the Audit Log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 2: The email address provided does not exist in the User table, the system suggests that the user Creates a profile (Use Case 1.4).	
	[ALT] Step 7: The password matches the previous password; the system displays an error message and prompts the user to provide another new password.	
Conclusion:	Password successfully updated.	
Post-condition:	The old password hash is replaced by the new hash. .	
Business rules:	• Ability to recover Account • Secure Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 136: 1.3 Reset Password - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.4	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Create User	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Guest (Prospective User)	
Primary system actor (psa):	• Guest	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A guest creates a profile by providing login credentials and personal details (age, gender, location, skills).	
Pre-condition:	The email provided must be valid and not yet registered in the system.	
Trigger:	The guest navigates to the Sign-Up page.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user enters the credentials they wish to use and personal information into the provided fields.	**Step 2:** The system validates their provided email address and password to ensure it matches the prescribed format. [ALT]
		Step 3: The system displays a list of skills the user can select
	Step 4: The user selects all skills that applies to them.	
	Step 5: The user selects the "Sign Up" Button	**Step 6:** The system hashes the entered password and saves it alongside the other fields in the User Table. The skills get saved in the UserSkills table. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture

System Name: Integrated NGO Management System		
		• ActiveStatus • UserShirtSize **User Skills:** • UserID [PK1FK1] • SkillID [PK2FK2]
		Step 7: Save Account creation transaction to AuditLog table **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 2: The email address provided already exists on the system. The user is informed and redirected to the Login Screen	
	[ALT] Step 2: The password or email address do not match regular conventions. The user is prompted to change them.	
Conclusion:	A new user account is created.	
Post-condition:	A new record is added to the database User table. .	
Business rules:	• Valid email address required • Secure Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 137: 1.4 Create User - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.5	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Update User	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A user edits their personal profile details, such as location or skills, to keep their information current.	
Pre-condition:	The user must be logged into their account.	
Trigger:	The user selects the "Edit Profile" option in their dashboard.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user edits the desired fields and selects the "Save" button. [ALT]	**Step 2:** The system saves the new details to the User table. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize
	Step 3: The user edits their skills and selects the "Save" button.	**Step 4:** The system saves the new updated skills to the UserSkills table. **User Skills:** • UserID [PK1FK1] • SkillID [PK2FK2]
		Step 5: Save Update action to AuditLog **AuditLog:** • AuditLogID [PK] • UserID [FK]

System Name: Integrated NGO Management System		
		• Timestamp • LogEntry
Alternate courses:	[ALT] Step 1: A field is left empty or an unsupported character is used. The system highlights the error and displays an error message.	
Conclusion:	User profile is updated in the database.	
Post-condition:	The database reflects the most recent user information. .	
Business rules:	• Secure Access • Accurate and current profile details.	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use. • A user cannot edit their age as this is automatically managed.	
Assumptions:	• None.	
Open issues:	• None.	

Table 138: 1.5 Update User - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.6	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View User	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	User	
Description:	The system allows an authorized user to search for and view the full profile details of a specific user.	
Pre-condition:	The actor must be logged in with appropriate administrative permissions.	
Trigger:	The actor selects the "View Users" option from the User Management dashboard.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user enters their search criteria [ALT]	**Step 2:** The system validates the search input
	Step 3: The user selects the "Search" button.	**Step 4:** The system retrieves matching records from the User table. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize
	Step 5: The user selects the profile they wish to view from the list of retrieved profiles. [ALT]	**Step 6:** The system displays all information pertaining to the selected user by joining the User and UserSkills tables. **User Skills:** • UserID [PK1FK1] • SkillID [PK2FK2]

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 1: A user wishes to view their own profile. They select their profile on the navbar and goes directly to **Step 6.**
	[ALT] Step 5: No matching users are found; the system displays an error message.
Conclusion:	The administrator successfully reviews the user's detailed information.
Post-condition:	None.
Business rules:	• Secure Access • Accurate and current profile details.
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 139: 1.6 View User - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.7	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Reactivate User	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	The User being reactivated	
Description:	An Admin restores access to a previously deactivated user account, allowing them to log in and interact with the system again.	
Pre-condition:	The targeted user must currently have a status of "Deactivated."	
Trigger:	The Admin selects a deactivated user and clicks the "Reactivate" button.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The Admin navigates to the "Deactivated Users" list.	**Step 2:** The system displays all users from the User table where Active Satus is false. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize
	Step 3: The administrator selects the user they wish to reactivate	**Step 4:** The system prompts the user for confirmation.
	Step 5: The administrator confirms the action. [ALT]	**Step 6:** The system updates the ActiveStatus for the specified record in the User table.

System Name: Integrated NGO Management System		
		Step 7: The system records the change to the Audit Log Table. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 5: A The user does not provide confirmation, and the use case is terminated.	
Conclusion:	The user account is restored to an active state.	
Post-condition:	The user is once again able to log in using their credentials.	
Business rules:	• Secure Access • Accurate and current profile details.	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 140: 1.7 Reactivate User - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 28 March 2026 **Version: 1.0**
Use case name:	1.8	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Deactivate User	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A user deactivates their profile. The profile is hidden from active lists, but historical project data is preserved.	
Pre-condition:	The user must be logged in to their account.	
Trigger:	The user clicks on the "Deactivate Profile" button.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: User initiates deactivation.	**Step 2:** The system prompts the user for a final confirmation.
	Step 3: User confirms the deactivation. [ALT]	**Step 4:** The system updates the UserStatus to “Deactivated” in the User table. **User:** • UserID [PK] • UserName • UserSurname • UserDateOfBirth • UserEmail • UserPassword • UserStatus • UserAllergies • ProfilePicture • ActiveStatus • UserShirtSize
		Step 5: Invoke UC 1.2 (User Logout)
		Step 6: Save account deactivation transaction to the Audit Log: **AuditLog:** • AuditLogID [PK] • UserID [FK]

System Name: Integrated NGO Management System		
		• Timestamp • LogEntry
Alternate courses:	[ALT] Step 3: If the user clicks "Cancel" the system returns them to the profile page without making changes.	
Conclusion:	User account is set to an inactive state	
Post-condition:	User can no longer log in; account is hidden from the public "Users List." .	
Business rules:	• Secure Access • Accurate and current profile details. • Data Retention	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 141: 1.8 Deactivate User - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date: 28 March 2026** **Version: 2.0**
Use case name:	1.9	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View FAQ	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	Administrator	
Description:	A user accesses a context-aware help section to resolve questions about the system or specific subsystems.	
Pre-condition:	The user must be logged in.	
Trigger:	The user clicks on the "Help" button.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: User clicks the "Help" button while in a specific subsystem.	**Step 2**: The system identifies the current application context. [ALT]
	.	Step 3: The system retrieves the FAQ table for entries matching the specific FAQType FAQ: • FAQID • FAQTypeID • Question • Answer • DateCreated
		Step 4: The system displays the relevant FAQ questions and answers to the user.
Alternate courses:	[ALT] Step 2: If the user clicks "Help" on the homepage, the system displays the full FAQ list.	
Conclusion:	User receives relevant help information.	
Post-condition:	None.	

System Name: Integrated NGO Management System	
Business rules:	• Access to help function.
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 142: 1.9 View FAQ - Logical Use Case Narrative

2.8.2 Admin Subsystem Narratives

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Amahle Khoza</td><td>Date: 20 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Create User Profile</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>2.1</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• System</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• User</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case allows the Admin to create a new user profile such as name, surname, contact details, and role assignment.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">1. Admin must be logged into the system
2. Admin must have permission to create users</td></tr>
<tr><td>Trigger:</td><td colspan="2">Admin selects “Create User Profile”</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1:
Admin selects “Create User”.</td><td>Step 2:
The system displays user creation form.</td></tr>
<tr><td>Step 3:
Admin enters user details.</td><td>Step 4:
The system validates the entered information [ALT]</td></tr>
<tr><td>Step 5:
The Admin submits the completed form.</td><td>Step 6:
The system stored the new user profile in the database [ALT]</td></tr>
<tr><td></td><td>Step 7:
The system displays a confirmation message indicating successful creation.</td></tr>
<tr><td rowspan="2">Alternate courses:</td><td colspan="2">[ALT] Step 4: If the entered data fails validation, the System displays an error message and prompts the Admin to enter the correct information. The process returns to Step 3.</td></tr>
<tr><td colspan="2">[ALT] Step 6: If the system detects that email already exists, it prevents the creation and will notify the Admin. The process returns to Step 3.</td></tr>
</table>

Conclusion:	The user profile is successfully created and stored in the database.
Post-condition:	1. New users exist in the system 2. User can now log in
Business rules:	• Each user must have a unique email • A role must be assigned to every user
Implementation constraints and specifications:	• Password must be encrypted • Email validation must be enforced
Assumptions:	• Admin has the correct user details
Open issues:	• Should users verify after creation?

Table 143: 2.1 Create User Profile - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 20 March 2026** **Version: 1.0**
Use case name:	Update User Profile	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case describes the process to update user profile within the system in a controlled and secure manner.	
Pre-condition:	The Admin is authenticated and has appropriate permissions.	
Trigger:	The Admin selects the relevant system option.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin initiates the "Update User Profile" function.	Step 2: The system retrieves or prepares the necessary interface [ALT]
	Step 3: Admin performs the action to update user profile.	Step 4: The system validates the input [ALT]
	Step 5: The Admin confirms the action.	Step 6: The system processes and updates the database.
		Step 7: The system displays a confirmation message.
Alternate courses:	[ALT] Step 2: If data is not found, the System will notify the Admin and terminate the process.	
	[ALT] Step 4: If validation fails while updating the user profile, the system displays an error and returns to Step 3.	
Conclusion:	The operation is completed successfully with system integrity maintained.	
Post-condition:	The system database reflects the updated changes.	
Business rules:	• Only authorized users may perform this action • Data must remain consistent and accurate	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Input validation must be enforced • Role-based access control must be applied • All changes must be logged for auditing
Assumptions:	• Admin has correct authority
Open issues:	• None

Table 144 2.2 Update User Profile - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 20 March 2026** **Version: 1.0**
Use case name:	View User Profile	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case provides a detailed process for viewing "User Profile" while ensuring security, data integrity, and proper system control.	
Pre-condition:	The Admin is authenticated, authorised, and the system is operational.	
Trigger:	The Admin initiates the function from the system interface.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The Admin selects the relevant function from the interface.	**Step 2:** The system retrieves the necessary data [ALT]
	Step 3: The Admin performs the action to view user profile.	**Step 4:** The System validates the input or request to view the user's profile [ALT]
		Step 5: The System processes and updates the interface to reflect the user's profile.
Alternate courses:	[ALT] Step 2: If the user is not found, the System notifies the Admin and terminates the process.	
	[ALT] Step 4: If validation fails during view user profile, the System displays an error and returns to step 3.	
Conclusion:	The operation is completed successfully with system integrity maintained.	
Business rules:	• All records must have unique identifiers • Only authorised users may perform this action • Data must remain consistent and accurate	
Implementation constraints and specifications:	• Input validation must be enforced • Role-based access control must be applied	
Assumptions:	• Admin has correct authority	

System Name: Integrated NGO Management System	
Open issues:	• None identified

Table 145: 2.3 View User Profile - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 20 March 2026** **Version: 1.0**
Use case name:	Deactivate User Profile	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case describes the process to deactivate user profile within the system in a controlled and secure manner.	
Pre-condition:	The Admin is authenticated and has appropriate permissions.	
Trigger:	The Admin selects the relevant system option.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The Admin selects the relevant function from the interface to Deactivate User Profile.	**Step 2:** The System retrieves or prepares the necessary interface on the screen [ALT]
	Step 3: The Admin performs the action to deactivate user profile.	**Step 4:** The System validates the request [ALT]
	Step 5: The Admin confirms the action.	**Step 6:** The System processes and updates the database.
		Step 7: The System displays a confirmation message that the user profile has been deactivated.
Alternate courses:	[ALT] Step 2: If the user is not found, the System notifies the Admin and terminates the process.	
	[ALT] Step 4: If validation fails during deactivate user profile, the System displays an error.	
Conclusion:	The operation is completed successfully with system integrity maintained.	
Post-condition:	The system database reflects the completed operation with the user profile deactivated.	
Business rules:	• Only authorised users may perform this action • The data must remain consistent and accurate. • All records must have unique identifiers.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Input validation must be enforced • Role-based access control must be applied
Assumptions:	• Admin has correct authority
Open issues:	• None identified

Table 146: 2.4 Deactivate User Profile - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Amahle Khoza</td><td>Date: 21 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Add FAQ</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>2.5</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• System</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• User</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the process to add a frequently asked question within the system in a controlled and secure manner.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">• Admin is authenticated and authorised
• Required related data such as the FAQ types exist in the system</td></tr>
<tr><td>Trigger:</td><td colspan="2">The Admin select the relevant option being “Add FAQ” in the system.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The Admin selects the “Add FAQ” function.</td><td>Step 2: The system retrieves the relevant data from the database.</td></tr>
<tr><td>Step 3: The Admin inputs the FAQ.</td><td>Step 4: The System validates data against business rules and relationships [ALT]</td></tr>
<tr><td>Step 5: The Admin confirms the action to add the FAQ.</td><td>Step 6: The System updates the new FAQ record while maintaining referential integrity.</td></tr>
<tr><td></td><td>Step 7: The System confirms successful completion.</td></tr>
<tr><td>Alternate courses:</td><td colspan="2">[ALT] Step 4: If validation fails, the System displays an error and returns to step 3.</td></tr>
<tr><td>Conclusion:</td><td colspan="2">The operation to add the FAQ is completed successfully.</td></tr>
<tr><td>Post-condition:</td><td colspan="2">The system database reflects the completed operation</td></tr>
<tr><td>Business rules:</td><td colspan="2">• Only authorised users may perform this action.</td></tr>
<tr><td>Implementation constraints and specifications:</td><td colspan="2">• Input validation must be applied.
• Role-based access control must be enforced.</td></tr>
</table>

System Name: Integrated NGO Management System	
Assumptions:	• Admin has sufficient knowledge of the system • System is functioning correctly
Open issues:	• None identified

Table 147: 2.5 Add FAQ - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Update FAQ	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case allows editing of an existing FAQ while maintaining its link to a valid FAQ Type. The System ensures that the updated FAQ remains valid, uniquely identified, and correctly categorised.	
Pre-condition:	• Admin is authenticated • FAQ exists in the system • At least one FAQ Type exists	
Trigger:	Admin selects an FAQ and clicks “Update”	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the FAQ to update.	**Step 2:** System retrieves the FAQ details including its faqtype_id.
	Step 3: Admin edits the question, answer, or FAQ Type.	**Step 4:** The System validates that the new FAQ Type exists [ALT]
		Step 5: The System validates that the updated questions does not duplicate another FAQ.
	Step 6: The Admin confirms the update.	**Step 7:** System updates the FAQ record in the database.
		Step 8: System logs the update action.
		Step 9: System displays confirmation.
Alternate courses:	[ALT] Step 4: Duplicate question detected and the System rejects the update and returns to step 3.	
Conclusion:	The FAQ is successfully updates while maintaining valid categorisation.	
Post-condition:	Updated FAQ is stored.	
Business rules:	• Questions must be unique • Each FAQ must belong to a valid FAQ Type	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Maintain referential integrity
Assumptions:	• Admin reviews content before updating
Open issues:	• Version history for FAQs

Table 148: 2.6 Update FAQ - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Remove FAQ	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case describes an event that will allow the Admin to remove an FAQ from the system.	
Pre-condition:	• Admin is authenticated • The FAQ exists	
Trigger:	Admin selects the "Remove FAQ" function.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects FAQ.	**Step 2:** The System retrieves FAQ.
	Step 3: Admin selects the "Remove" button.	
	Step 4: Admin confirms deletion.	**Step 5:** System removes FAQ.
		Step 6: The System logs the action.
		Step 7: System confirms successful removal.
Alternate courses:	None	
Conclusion:	The FAQ is removed from active use.	
Post-condition:	FAQ is no longer visible to users.	
Business rules:	• FAQs should not be removed if they are required for audit.	
Implementation constraints and specifications:	• None	

System Name: Integrated NGO Management System	
Assumptions:	• Removal policy is defined.
Open issues:	• Retention period for deleted FAQs

Table 149: 2.7 Remove FAQ - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Create Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.8	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• User	
Other interested stakeholders:	None	
Description:	This use case describes an event where an Admin is required to create a role.	
Pre-condition:	• Admin is authenticated	
Trigger:	Admin selects “Create Role”.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects “Create Role”.	**Step 2:** The system displays the role creation form.
	Step 3: Admin fills in the form.	**Step 4:** The system displays the form with the details filled in by the Admin.
	Step 5: Admin confirms the details.	**Step 6:** The System validates the details.
		Step 7: The System successfully creates the role and displays confirmation.
Alternate courses:	None	
Conclusion:	Admin successfully created role.	
Post-condition:	• Role available for use.	
Business rules:	• Only authorised users may create roles.	
Implementation constraints and specifications:	• None	

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 150: 2.8 Create Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	View Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.9	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case enables the Admin to retrieve details of roles already defined in the system, including role names, descriptions, and associated permissions.	
Pre-condition:	• At least one role exists in the system • Admin is authenticated	
Trigger:	Admin selects "View Role"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin requests to view roles.	**Step 2:** System retrieves role records from the database.
		Step 3: System retrieves associated permissions for each role [ALT]
		Step 4: System compiles role data.
		Step 5: System displays roles in a structured format.
	Step 6: Admin reviews role details.	
Alternate courses:	[ALT] Step 3: Permission retrieval fails and the system displays role without permission and logs an error.	
Conclusion:	Admin successfully reviews role configurations.	
Post-condition:	• Role visibility is confirmed.	
Business rules:	• Only authorised Admins can view roles.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must support efficient querying of role-permission relationships.
Assumptions:	• Role permission relationships are correctly maintained.
Open issues:	• None

Table 151: 2.9 View Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Update Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.10	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to modify existing role details, including role name and description, to better reflect organisational responsibilities or changes in system access policies.	
Pre-condition:	• Admin is authenticated • Role must exist	
Trigger:	Admin selects "Update Role"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects a role.	**Step 2:** System retrieves the role details.
		Step 3: System displays all the editable fields.
	Step 4: The Admin updates role information.	**Step 5:** System should validate uniqueness of role name [ALT]
	Step 6: Admin confirms new update.	**Step 7:** System updates role record.
		Step 8: System logs modification.
		Step 9: System confirms success.
Alternate courses:	[ALT] Step 5: Duplicate role name therefore the system rejects and returns to step 4.	
Conclusion:	Role successfully updated.	
Post-condition:	• Updated role stored.	
Business rules:	• Role names mist be unique.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 152: 2.10 Update Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Reactivate Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.11	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to restore a previously deactivated role, making it available again for assignment to users.	
Pre-condition:	• Admin is authenticated • Role exists and is inactive	
Trigger:	Admin selects "Activate Role"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the inactive role.	Step 2: System retrieves role status [ALT]
	Step 3: Admin confirms reactivation.	Step 4: System updates the role status to active.
		Step 5: System logs actions.
		Step 6: System confirms success.
Alternate courses:	[ALT] Step 2:Role already exists so the system prevents the action.	
Conclusion:	Role successfully reactivated.	
Post-condition:	• Role is available for assignment.	
Business rules:	• Only inactive roles can be reactivated.	
Implementation constraints and specifications:	• None	
Assumptions:	• Role is still valid for current system use	

System Name: Integrated NGO Management System	
Open issues:	• None

Table 153: 2.11 Reactivate Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Deactivate Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.12	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to deactivate a role, preventing it from being assigned to users and restricting its use in access control without permanently deleting it.	
Pre-condition:	• Admin is authenticated • The role exists	
Trigger:	Admin selects “Deactivate Role”.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the role.	**Step 2:** System checks if the role is assigned to users.
		Step 3: System displays impact warming.
	Step 4: Admin confirms deactivation.	**Step 5:** System updates the tole status to inactive.
		Step 6: System logs action.
		Step 7: The System confirms success.
Alternate courses:	None	
Conclusion:	Role successfully deactivated.	
Post-condition:	• Role cannot be assigned to users.	
Business rules:	• Roles assigned to users may require reassignment before deactivation.	
Implementation constraints and specifications:	• None	

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 154: 2.12 Deactivate Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Create Role Permission	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.13	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to define and assign specific permissions to a role, determining what actions users associated with that role can perform within different subsystems (e.g., Inventory, Warehouse, Project Management). Permissions may include create, view, update, delete, or specialised actions such as "allocate inventory" or "capture feedback."	
Pre-condition:	• Role exists and is active • Admin is authenticated	
Trigger:	Admin selects "Create Role Permission".	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects a role to assign permissions to.	**Step 2:** System retrieves role details.
		Step 3: System retrieves available system modules and actions.
		Step 4: System displays permission configuration interface.
	Step 5: Admin selects the permissions	**Step 6:** The System validates that the permissions are not duplicated.
	Step 7: Admin confirms assignment.	**Step 8:** System creates role-permission records.
		System 9: System logs permission assignment and confirms success.
Alternate courses:	None	
Conclusion:	Permissions successfully assigned to role.	

System Name: Integrated NGO Management System	
Post-condition:	• Role has defined permissions.
Business rules:	• A role can have multiple permissions • A permission can be assigned to multiple roles
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 155: 2.13 Create Role Permission - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Update Role Permission	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.14	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to modify existing permissions assigned to a role, enabling dynamic adjustment of access rights as system requirements or organisational responsibilities change.	
Pre-condition:	• Admin is authenticated • Role and associated permissions exist	
Trigger:	Admin selects “Update Role Permission”.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects a role.	**Step 2:** System retrieves current permissions.
		Step 3: System displays editable permission matrix.
	Step 4: Admin modifies permissions.	**Step 5:** System validated permission.
	Step 6: Admin confirms the update.	**Step 7:** System updates and logs changes.
		System 8: System confirms success.
Alternate courses:	None	
Conclusion:	Role permissions successfully updated.	
Post-condition:	• Updated permissions enforced in the system	
Business rules:	• Changes must take effect immediately	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 156: 2.14 Update Role Permission - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	View Role Permission	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.15	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to view permissions assigned to a role.	
Pre-condition:	• Admin is authenticated • Role must exist in the system	
Trigger:	Admin selects "View Role Permission"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects role.	**Step 2:** System retrieves role details.
		Step 3: System retrieves associated permissions.
		Step 4: System retrieves associated permissions by subsystem.
	Step 5: Admin reviews permission.	
Alternate courses:	None	
Conclusion:	Admin successfully reviews role permissions.	
Post-condition:	• No changes made	
Business rules:	• Permissions must reflect current system configuration	
Implementation constraints and specifications:	• None	

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 157: 2.15 View Role Permission - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Deactivate Role Permission	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	2.16	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to deactivate specific permissions assigned to a role, restricting access to certain system functionalities without removing the role entirely.	
Pre-condition:	• Admin is authenticated • Role and permissions exist	
Trigger:	Admin selects “Deactivate Role Permission”.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects role.	**Step 2:** System retrieves assigned permissions.
		Step 3: System displays permission list.
	Step 4: The Admin selects the permission to deactivate.	**Step 5:** System validates that the permission is currently active.
	Step 6: Admin confirms action.	**Step 7:** System updates permission status to inactive.
		Step 8: System logs change.
		Step 9: The System confirms success.
Alternate courses:	None	
Conclusion:	Selected permissions are successfully deactivated.	
Post-condition:	• Role access is restricted accordingly	
Business rules:	• Admin selects “Deactivate Role Permission”	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• None
Assumptions:	• Admin selects the correct permissions.
Open issues:	• None

`Table 158: 2.16 Deactivate Role Permission - Logical Use Case Narrative

2.8.3 Inventory Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Receive Inventory Item	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Super Admin	
Description:	This use case describes the event in which inventory is received. The system will allow the admin to enter details regarding the inventory received. The system will validate the details entered before asking for confirmation. Once confirmed the system will add the inventory to the system. The use case ends once the received inventory is successfully captured on the system.	
Pre-condition:	Admin is logged in.	
Trigger:	Inventory is received.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Inventory is received.	
	Step 2: The admin will navigate to the inventory tab and select add.	**Step 3:** The system will display the "Receive Inventory" form allowing the admin to enter the received inventory details.
	Step 4: The admin enters the details and selects add. [ALT]	**Step 5:** The system validates that all required fields are completed. [ALT]
		Step 6: The system requests confirmation for the inventory to be added.
	Step 7: The admin confirms that the inventory can be created. [ALT]	**Step 8:** The system adds the inventory to the following tables: • InventoryTransaction

System Name: Integrated NGO Management System		
		• Procurement • ProcurementDetail [ALT]
		Step 9: The system saves the transaction to the AuditLog table.
		Step 10: The system displays a message informing the admin that the inventory has been added successfully.
Alternate courses:	[ALT] Step 4: The admin selects cancel. Use case terminates.	
	[ALT] Step 5: The validation is unsuccessful; the admin is prompted to enter the details into the empty fields. Go back to step 4.	
	[ALT] Step 7: The admin selects back. Go back to step 4.	
	[ALT] Step 8: The system fails to add the inventory into the relevant entities and displays an error message to the admin. Use case terminates.	
Conclusion:	The system displays a message informing the admin that the inventory has been added successfully.	
Post-condition:	The inventory is captured on the system.	
Business rules:	• Only a user with the required permissions can receive inventory.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 159: 3.1 Receive Inventory Item - Logical Use Case Narrative

<table>
<tr><th colspan="4">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Christopher Bailey</td><td colspan="2">Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>View Inventory</td><td colspan="2" rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>3.2</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="3">Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="3">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="3">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="3">• Super Admin</td></tr>
<tr><td>Description:</td><td colspan="3">This use case describes the event in which the admin views the inventory on the system. The system will retrieve and display all the inventory on the system as well as allowing the admin to filter. The use case ends once all the inventory on the system are displayed.</td></tr>
<tr><td>Pre-condition:</td><td colspan="3">Admin is logged in.</td></tr>
<tr><td>Trigger:</td><td colspan="3">Admin wants to view inventory items.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>Step 1: Admin wants to view inventory items.</td><td colspan="2"></td></tr>
<tr><td>Step 2: The super admin will navigate to the inventory tab and selects manage.</td><td colspan="2">Step 3: The system retrieves and displays a list of all the inventory on the system from the following tables:
• Inventory
• InventoryType
• InventoryTransaction</td></tr>
<tr><td>Step 4: The admin filters the inventory items.</td><td colspan="2">Step 5: The system filters and displays the inventory matching the criteria.</td></tr>
<tr><td>Step 6: The admin can view all inventory.</td><td colspan="2"></td></tr>
<tr><td>Alternate courses:</td><td colspan="3">None</td></tr>
<tr><td>Conclusion:</td><td colspan="3">The system filters and displays the inventory matching the criteria.</td></tr>
</table>

System Name: Integrated NGO Management System	
Post-condition:	The inventory is displayed.
Business rules:	• Only a user with the required permission can view inventory.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 160: 3.2 View Inventory - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Write-Off Inventory	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the admin writes-off inventory. The system will display all the inventory on the system and allow the admin to write-off inventory. The system will ask for confirmation before the inventory items are written off. The use case ends once the inventory items are written off on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Admin wants to write off inventory.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin wants to write off inventory.	**Step 2:** The system invokes use case 3.2 (View Inventory).
	Step 3: The admin selects write-off.	**Step 4:** The system displays the "Write-Off Inventory" form.
	Step 5: The admin selects and enters the amount of inventory to write off. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the inventory to be written off.
	Step 8: The admin confirms that the inventory can be written off. [ALT]	**Step 9:** The system writes off the inventory in the InventoryTransaction table. [ALT]

System Name: Integrated NGO Management System		
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the admin that the inventory has been written off successfully.
Alternate courses:	[ALT] Step 5: Admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful; the admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The admin does not confirm. Go back to step 5.	
	[ALT] Step 9: The system fails to write-off the inventory in the relevant entities and displays an error message to the admin. Use case terminates.	
Conclusion:	The system displays a message informing the admin that the inventory has been written off successfully.	
Post-condition:	The inventory is written-off.	
Business rules:	• Only a user with the required permissions can write off inventory.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 161: 3.3 Write-Off Inventory - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Inventory Stock Take	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin performs an inventory stock take. The system will allow the super admin to select a warehouse and view its inventory. The super admin will perform a stock take and enter the stock count. The system will determine the difference and capture the difference to adjust the inventory. The use case ends once inventory has been adjusted and the stock take is completed.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to perform an inventory stock take.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to perform an inventory stock take.	
	Step 2: The super admin will navigate to the inventory tab and select stock take.	**Step 3:** The system retrieves all the warehouses from the Warehouse table.
		Step 4: The system displays the "Stock Take" form.
	Step 5: The super admin selects a warehouse.	**Step 6:** The system retrieves all the inventory items in the selected warehouse using the InventoryTransaction table.

System Name: Integrated NGO Management System		
	Step 7: The super admin enters the physical stock count. [ALT]	**Step 8:** The system determines the difference and saves the results and adjusts the inventory in the following tables: • StockTake • StockTakeDetails • InventoryTransactions [ALT]
		Step 9: The system saves the transaction to the AuditLog table.
		Step 10: The system displays a message informing the admin that the inventory has been adjusted successfully.
Alternate courses:	[ALT] Step 7: The super admin selects cancel. Use case terminates	
	[ALT] Step 8: The system fails to add the stock take into the relevant entities and displays an error message to the admin. Use case terminates.	
Conclusion:	The system displays a message informing the admin that the inventory has been adjusted successfully.	
Post-condition:	The inventory is adjusted and the stock take is complete is removed.	
Business rules:	• Only a super admin can perform an inventory stock take	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 162: 3.4 Inventory Stock Take - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Allocate Inventory	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the admin allocates inventory to a project. The system will display all the inventory and allow the admin to allocate inventory to an active project. The system will validate inputted data and ask for confirmation before allocating inventory. The use case ends once inventory is allocated to a project.	
Pre-condition:	Admin is logged in.	
Trigger:	Admin wants to allocate inventory.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin wants to allocate inventory.	**Step 2:** The system invokes use case 3.2 (View Inventory)
	Step 3: The admin will select allocate.	**Step 4:** The system retrieves active projects from the Project table.
		Step 5: The system displays the "Allocate Inventory" form.
	Step 6: The admin selects a project and allocates inventory. [ALT]	**Step 7:** The system validates that all required fields are completed. [ALT]
		Step 8: The system requests confirmation for the inventory to be allocated.

System Name: Integrated NGO Management System		
	Step 9: The admin confirms that the inventory can be allocated. [ALT]	**Step 10:** The system allocates inventory to the project in the InventoryTransaction table. [ALT]
		Step 11: The system saves the transaction to the AuditLog table.
		Step 12: The system displays a message informing the admin that the inventory has been allocated successfully.
Alternate courses:	[ALT] Step 6: Admin selects cancel. Use case terminates	
	[ALT] Step 7: The validation is unsuccessful; the admin is prompted to enter the details into the empty fields. Go back to step 6.	
	[ALT] Step 9: The admin selects back. Go back to step 6.	
	[ALT] Step 10: The system fails to allocate the inventory in the relevant entities and displays an error message to the admin. Use case terminates.	
Conclusion:	The system displays a message informing the admin that the inventory has been allocated successfully.	
Post-condition:	The inventory is allocated to a project.	
Business rules:	• Only a user with the required permissions can allocate inventory.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 163: 3.5 Allocate Inventory - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Inventory Item Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin creates an inventory item type. The system will display all the current inventory item types and allow the super admin to create a new inventory item type before being validated and saved on the system. The use case ends once the new inventory item type is saved on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to create a new inventory item type.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to create a new inventory item type.	**Step 2:** The system invokes use case 3.8 (View Inventory Item Types)
	Step 3: The super admin will select create.	**Step 4:** The system will display the "Create Inventory Item Type" form to provide the following information: • Name • Description • SecurityLevel
	Step 5: The super admin enters the inventory item type details and select create. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]

System Name: Integrated NGO Management System		
		Step 7: The system requests confirmation for the inventory item type to be created.
	Step 8: The super admin confirms that the inventory item type can be created. [ALT]	**Step 9:** The system adds the inventory item type to the InventoryType table. [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the inventory item type has been created successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful; the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to add the inventory item type into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the inventory item type has been created successfully.	
Post-condition:	The inventory item type is created successfully.	
Business rules:	• Only a super admin can create an inventory item type.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 164: 3.6 Create Inventory Item Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Inventory Item Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to view an inventory item type on the system. The system will retrieve and display all the inventory item types on the system before the super admin selects an inventory item type to view. The system will retrieve the selected inventory item type's details and display them to the super admin. The use case ends once the system displays the selected inventory item type's details.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to view an inventory item type.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view an inventory item type.	**Step 2:** The system invokes use case 3.8 (View Inventory Item Types)
	Step 3: The super admin will select an inventory item type.	**Step 4:** The system retrieves the selected inventory item type's details from the InventoryType.
		Step 4: The system will display the "View Inventory Item Type" popup displaying the selected inventory item type's details including: • Name • Description • SecurityLevel

System Name: Integrated NGO Management System		
	Step 5: Super admin can view an inventory item type.	
Alternate courses:	None	
Conclusion:	Super admin can view an inventory item type.	
Post-condition:	The inventory item type's details is displayed.	
Business rules:	• Super admins can view an inventory item type. • Admins can view an inventory item type.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 165: 3.7 View Inventory Item Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Inventory Item Types	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.8	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to view all the inventory item types on the system. The system will retrieve all the inventory item types on the system and display them to the super admin. The use case ends once all the inventory item types are displayed.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to view all the inventory item types.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view all the inventory item types.	
	Step 2: The super admin will navigate to the inventory tab.	
	Step 3: The super admin will select type.	**Step 4:** The system retrieves all the inventory item types on the system from the InventoryType table.
		Step 5: The system displays the "Inventory Item Type View", displaying the following details for each inventory item type: • Name • Description • SecurityLevel

System Name: Integrated NGO Management System		
	Step 6: The super admin can view all the inventory item types.	
Alternate courses:	None	
Conclusion:	The super admin can view all the inventory item types.	
Post-condition:	All inventory item types are displayed.	
Business rules:	• Super admins can view all inventory item types. • Admins can view all inventory item types.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 166: 3.8 View Inventory Item Types - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Inventory Item Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.9	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to update an inventory item type on the system. The system allows the super admin to select an inventory item type and then update the inventory item type's details. The system will validate the details entered before asking the super admin to confirm the updated details. Once confirmed the system will inform the super admin that the inventory item type's details have been successfully updated. The use case ends once the inventory item type's updated details have been successfully captured on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to update an inventory item type.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to update an inventory type.	**Step 2:** The system invokes use case 3.7 (View Inventory Item type)
	Step 3: The super admin selects update.	**Step 4:** The system displays the "Update Inventory Item Type" form to update the following information: • Name • Description • SecurityLevel
	Step 5: The super admin enters the updated details. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]

System Name: Integrated NGO Management System		
		Step 7: The system requests confirmation for the inventory item type to be updated.
	Step 8: The super admin confirms that the inventory item type can be updated. [ALT]	**Step 9:** The system saves the updated details in the InventoryType table. [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the inventory item type was updated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful; the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to update the inventory item type in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the inventory item type was updated successfully.	
Post-condition:	The inventory item type's details are updated.	
Business rules:	• Only super admins can update an inventory item type.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 167: 3.9 Update Inventory Item Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Inventory Item Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	3.10	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to deactivate an inventory item type on the system. The system will display the inventory item type and allow the super admin to deactivate the inventory item type. The system will ask the super admin to confirm the deactivation before the inventory item type is deactivated. The use case ends once the inventory item type has been successfully deactivated on the system.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to deactivate an inventory item type.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super Admin wants to deactivate an inventory item type.	**Step 2:** The system invokes use case 3.7 (View Inventory Item type)
	Step 3: The super admin selects deactivate.	**Step 4:** The system requests confirmation for the inventory item type to be deactivated.
	Step 5: The super admin confirms that the inventory item type can be deactivated. [ALT]	**Step 6:** The system sets the status of the inventory item type to deactivated in the InventoryType table. [ALT]
		Step 7: The system saves the transaction to the AuditLog table.
		Step 8: The system displays a message informing the super admin that the inventory item type was deactivated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	

System Name: Integrated NGO Management System	
	[ALT] Step 6: The system fails to update the status of the inventory item type in the relevant entities and displays an error message to the super admin. Use case terminates.
Conclusion:	The system displays a message informing the super admin that the inventory item type was deactivated successfully.
Post-condition:	The inventory item type is deactivated on the system.
Business rules:	• Only super admins can deactivate an inventory item type.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 168: 3.10 Deactivate Inventory Item Type - Logical Use Case Narrative

2.8.4 Group Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Group	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Users	
Description:	This use case describes the event of when a super admin wants to add a new group to the system. The system prompts the super admin to enter the new group's details. The use case ends when the system saves the new group onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to add a new group to the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to add a new group to the system.	**Step 2:** The system invokes UC 4.2 (View Groups) to display a pop-up showing all the groups' details.
	Step 3: The super admin selects to add a new group to the project.	**Step 4:** The system displays an add group form to provide the following information: • Name • Project • Status • Members
	Step 5: The super admin enters the group's details and selects save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the new group to be added.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the new group onto the Group table

System Name: Integrated NGO Management System		
		and the assigned users onto the UserGroup table. [ALT]
		Step 10: The system automatically creates a messaging channel for the group members to communicate.
		Step 11: The system records the transaction to the AuditLog table.
		Step 12: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the new group. Terminate use case.	
Conclusion:	The system saves the new group onto the Group table and the assigned users onto the UserGroup table.	
Post-condition:	The new group's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can add a new group.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 169: 4.1 Create Group - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Groups	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a user wants to view the groups for a specific project on the system The user searches for a project on the system and then selects the project to view. The use case ends when the system displays the groups for a specific project and the details for each of the groups.	
Pre-condition:	The user must be logged into the system.	
Trigger:	The user wants to view the groups for a specific project to which they are assigned.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user wants to view the groups for a specific project to which they are assigned.	
	Step 2: The user selects the team tab and then selects all groups.	**Step 3:** The system retrieves the current projects and the number of groups associated with them on the system from the Project and Group table.
		Step 4: The system displays the projects and the number of groups they have based on user roles.
	Step 5: The user searches for a project by entering the name. [ALT]	**Step 6:** The system displays the project matching the criteria. [ALT]
	Step 7: The user selects the project.	**Step 8:** The system retrieves all the groups for that project, their details as well as the assigned users from the tables:

System Name: Integrated NGO Management System		
		• Group table • UserGroup table
		Step 9: The system displays a pop-up with all the details. These details include: • Groups' Names • Groups' Statuses • Groups and Members
	Step 10: The user can view the groups.	
Alternate courses:	[ALT] Step 4a: The user does not search for a project and just selects the project directly from the ones displayed. Go to Step 6.	
	[ALT] Step 4b: The user uses the filter to search for a project. Go to Step 6.	
	[ALT] Step 6: The system does not find any projects matching the specified criteria. Go back to step 4.	
Conclusion:	The system displays all the groups' details.	
Post-condition:	All the related groups' details are displayed.	
Business rules:	• Users can only view groups they are assigned to. • Admins and super admins can view all groups on the system.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 170: 4.2 View Groups - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Group	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to update a group's details on the system. The system will display the group's details and then allow the super admin to update the group. The use case ends when the system successfully saves the new details onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to update a group's details on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to update a group's details on the system.	**Step 2:** The system invokes UC 4.2 (View Groups) to display a pop-up showing all the groups' details.
	Step 3: The super admin selects update.	**Step 4:** The system displays an update group form with all the fields as editable. These include: • Groups' Names And a table with the users with the ability to add and remove members from each group in the project.
	Step 5: The super admin updates a group's details and selects save. [ALT]	**Step 6:** The system validates the updated data. [ALT]
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the changes to the Group and UserGroup tables. [ALT]

System Name: Integrated NGO Management System		
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the admin. Go back to step 4.	
	[ALT] Step 8: The super admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the changes. Terminate use case.	
Conclusion:	The system saves the changes to the Group and UserGroup tables.	
Post-condition:	The updated group's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can update a group's details.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 171: 4.3 Update Group - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Group	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to deactivate a group on the system. The system will display the group and then allow the super admin to deactivate the group. The use case ends when the group has been successfully deactivated on the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to deactivate a group on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to deactivate a group on the system.	**Step 2:** The system invokes UC 4.2 (View Groups) to display a pop-up showing all the groups' details.
	Step 3: The super admin selects deactivate on a specific group. [ALT]	**Step 4:** The system requests confirmation for the group deactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system sets the group's status to be inactive in the Group table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.	

System Name: Integrated NGO Management System	
	[ALT] Step 6: The system fails to deactivate the group. Terminate use case.
Conclusion:	The system sets the group's status to inactive in the Group table.
Post-condition:	The group has been deactivated on the system, and a confirmation message is shown.
Business rules:	• Only super admin can deactivate a group.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 172: 4.4 Deactivate Group - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Skill	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.5	
Priority:	Medium	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to add a new skill to the system. The system prompts the super admin to enter the new skill's details. The use case ends when the system saves the new skill onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to add a new skill to the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to add a new skill to the system.	**Step 2:** The system invokes UC 4.6 (View Skills) to display a table showing all the current skills.
	Step 3: The super admin selects add.	**Step 4:** The system displays an add skill form to provide the following information: • Skill • Description
	Step 5: The super admin enters the new skill's details and selects save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the skill to be added.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the new skill onto the Skills table. [ALT]
		Step 10: The system records the transaction to the AuditLog table.

System Name: Integrated NGO Management System		
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind about adding a new skill. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The super admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the new skill. Terminate use case.	
Conclusion:	The system saves the new skill onto the Skills table.	
Post-condition:	The new skills' details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can add a new skill.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 173: 4.5 Create Skill - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>View Skills</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>4.6</td></tr>
<tr><td>Priority:</td><td>Medium</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event of when an admin wants to view the skills on the system. The system will display a list of skills and their details in a table. The use case ends when the admin can see the skills and their details on the system.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The admin must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The admin wants to view the skills on the system.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The admin wants to view the skills on the system.</td><td></td></tr>
<tr><td>Step 2: The admin selects the skills widget on the home dashboard.</td><td>Step 3: The system retrieves all the current skills on the system from the Skills table.</td></tr>
<tr><td></td><td>Step 4: The system displays a table showing all the current skills.
The table includes:
• Skill
• Description
• Status
• Actions</td></tr>
<tr><td>Step 5: The admin can view the skills.</td><td></td></tr>
<tr><td>Alternate courses:</td><td colspan="2">None</td></tr>
<tr><td>Conclusion:</td><td colspan="2">The system displays a table showing all the current skills.</td></tr>
<tr><td>Post-condition:</td><td colspan="2">The skill’s details are displayed.</td></tr>
</table>

System Name: Integrated NGO Management System	
Business rules:	• Admins and super admins can view the skills.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 174: 4.6 View Skills - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Skill	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.7	
Priority:	Medium	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to update an existing skill on the system The system will display the selected skill's details and the super admin will be able to modify the skill's information. The use case ends when the system saves the updated skill's details onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to update a skill's details on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to update a skill's details on the system.	**Step 2:** The system invokes UC 4.6 (View Skills) to display a table showing all the current skills.
	Step 3: The super admin selects update in a specific row.	**Step 4:** The system displays the specific row's fields as editable. These include • Skill • Description
	Step 5: The super admin updates the skill's details and selects save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the changes to be made to the skill.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the updated skill's details onto the Skills table. [ALT]

System Name: Integrated NGO Management System		
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind about updating a skill. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 2.	
	[ALT] Step 8: The super admin does not confirm. Go back to step 2.	
	[ALT] Step 9: The system fails to save the changes. Terminate use case.	
Conclusion:	The system saves the updated skill's details onto the Skills table.	
Post-condition:	The updated skill's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can update a skill.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 175: 4.7 Update Skill - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Skill	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	4.8	
Priority:	Medium	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to deactivate a skill on the system. The system displays the active skills and allows the super admin to select a skill to deactivate. The use case ends when the skill has been successfully deactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to deactivate a skill on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to deactivate a skill on the system.	**Step 2:** The system invokes UC 4.6 (View Skills) to display a table showing all the current skills.
	Step 3: The super admin selects deactivate on a specific skill where the status is active. [ALT]	**Step 4:** The system requests confirmation for the skill deactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system updates the skill's status to be inactive in the Skills table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3a: The super admin changes their mind about deactivating a skill. Terminate use case.	

System Name: Integrated NGO Management System	
	[ALT] Step 3b: The super admin selects deactivate where the status is already inactive. The system displays a message to the super admin. Terminate use case.
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.
	[ALT] Step 6: The system fails to deactivate the skill. Terminate use case.
Conclusion:	The system updates the skill's status to inactive in the Skills table.
Post-condition:	The active skill has been deactivated, and a confirmation message is shown.
Business rules:	• Only super admin can deactivate a skill.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 176: 4.8 Deactivate Skill - Logical Use Case Narrative

2.8.5 Warehouse Management Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Create Warehouse	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	5.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to register a new warehouse within the system, capturing its identifying and operational details such as warehouse name, physical location, storage capacity, and status. The warehouse serves as a storage entity for inventory items and is critical for tracking stock distribution and allocation.	
Pre-condition:	• Admin is authenticated to manage warehouse • Warehouse does not already exist	
Trigger:	Admin selects “Create Warehouse”	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin initiates warehouse creation.	**Step 2:** System displays warehouse creation form.
	Step 3: Admin captures warehouse details (name, location, capacity, description) [ALT]	**Step 4:** System validates required fields and its uniqueness.
	Step 5: Admin confirms submission.	**Step 6:** System creates the warehouse and confirms successful creation.
Alternate courses:	[ALT] Step 3: System highlights errors and prevents submission due to missing required fields.	

System Name: Integrated NGO Management System	
Conclusion:	Warehouse successfully created and is available for use.
Post-condition:	• Warehouse stored in system
Business rules:	• Warehouse must be unique per location
Implementation constraints and specifications:	Must integrate with inventory subsystem
Assumptions:	• Admin provides accurate warehouse details.
Open issues:	• None

Table 177: 5.1 Create Warehouse - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	View Warehouse	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	5.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to retrieve and view details of a specific warehouse, including its location, operational status, and associated inventory (if integrated).	
Pre-condition:	• Admin is authenticated to manage warehouse • Warehouse should exist	
Trigger:	Admin selects "View Warehouse"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the warehouse.	**Step 2:** System retrieves warehouse details [ALT]
		Step 3: System compiles warehouse data.
		Step 4: System displays warehouse information.
Alternate courses:	[ALT] Step 2: System does not find the warehouse and displays error message.	
Conclusion:	Warehouse information successfully displayed.	
Post-condition:	• No data modified	
Business rules:	• Only authorised users may view warehouse data.	
Implementation constraints and specifications:	None	

System Name: Integrated NGO Management System	
Assumptions:	• Warehouse data is up to date.
Open issues:	• None

Table 178: 5.2 View Warehouse - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Update Warehouse	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	5.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to modify warehouse details such as name, location, or capacity to reflect operational changes.	
Pre-condition:	• Admin is authenticated to manage warehouse • Warehouse should exist	
Trigger:	Admin selects "Update Warehouse"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the warehouse.	**Step 2:** System retrieves warehouse details
		Step 3: System displays editable warehouse data.
	Step 4: Admin updates details.	**Step 5:** System validates updates.
	Step 6: Admin confirms details.	**Step 7:** System updates warehouse and confirms success.
Alternate courses:	None	
Conclusion:	Warehouse successfully updates	
Post-condition:	• Updates changes stored	
Business rules:	• Only authorised users may update warehouse data.	
Implementation constraints and specifications:	None	

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 179: 5.3 Update Warehouse - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Amahle Khoza</td><td>Date: 22 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Reactivate Warehouse</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>5.4</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• System</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">User</td></tr>
<tr><td>Description:</td><td colspan="2">This use case allows the Admin to restore a previously deactivated warehouse, making it available again for inventory storage and operations.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">• Admin is authenticated to manage warehouse
• Warehouse should exist and is inactive</td></tr>
<tr><td>Trigger:</td><td colspan="2">Admin selects “Activate Warehouse”</td></tr>
<tr><td rowspan="4">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: Admin selects the warehouse.</td><td>Step 2: System retrieves warehouse details status.</td></tr>
<tr><td>Step 3: Admin confirms the activation of the warehouse.</td><td>Step 4: System updates status to active.
.</td></tr>
<tr><td></td><td>Step 5: The System logs the actions and confirms success.</td></tr>
<tr><td>Alternate courses:</td><td colspan="2">None</td></tr>
<tr><td>Conclusion:</td><td colspan="2">Warehouse successfully reactivated.</td></tr>
<tr><td>Post-condition:</td><td colspan="2">• Warehouse is now available for operations</td></tr>
<tr><td>Business rules:</td><td colspan="2">• Only inactive warehouses can be reactivated</td></tr>
<tr><td>Implementation constraints and specifications:</td><td colspan="2">None</td></tr>
</table>

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 180: 5.4 Reactivate Warehouse - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Amahle Khoza	**Date: 22 March 2026** **Version: 1.0**
Use case name:	Deactivate Warehouse	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	5.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• System	
Other participating actors:	• None	
Other interested stakeholders:	User	
Description:	This use case allows the Admin to deactivate a warehouse, preventing it from being used for inventory storage while preserving historical data.	
Pre-condition:	• Admin is authenticated to manage warehouse • Warehouse should exist	
Trigger:	Admin selects "Deactivate Warehouse"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Admin selects the warehouse.	**Step 2:** System checks for existing inventory in the warehouse.
		Step 3: System displays warning if inventory exists.
	Step 4: Admin confirms deactivation.	**Step 5:** System updates warehouse status to inactive.
		Step 6: System logs action and confirms success.
Alternate courses:	None	
Conclusion:	Warehouse successfully deactivated	
Post-condition:	• Warehouse is no longer available for allocation	
Business rules:	• Only authorised users may deactivate warehouses.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	None
Assumptions:	• None
Open issues:	• None

Table 181: 5.5 Deactivate Warehouse - Logical Use Case Narrative

2.8.6 Vendor Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Vendor	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin creates a vendor on the system. The super admin will select to add a vendor and enter the vendor's information. The system will validate the details inputted before informing the super admin that the vendor has been successfully added. The use case ends once the vendor has successfully been added to the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to create a vendor.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to create a vendor.	**Step 2:** The system invokes use case 6.3 (View Vendors)
	Step 3: The super admin selects add.	**Step 4:** The system will display the "Add Vendor" form to provide the following information: • Name • Type • Email • Phone • Contact Person • Credit Facility • Address • Rating

System Name: Integrated NGO Management System		
		• Photo (Optional)
		Step 5: The system retrieves vendor types from the VendorType table.
	Step 6: The super admin will select a type of vendor.	
	Step 7: The super admin will enter the vendor's details and select add. [ALT]	**Step 8:** The system validates that all required fields are completed. [ALT]
		Step 9: The system requests confirmation for the rhino to be added.
	Step 10: The super admin confirms that the rhino can be added. [ALT]	**Step 11:** The system adds the vendor to the following tables based on the type selected: • Vendor • Accommodation • Caterer [ALT]
		Step 12: The system saves the transaction to the AuditLog table.
		Step 13: The system displays a message informing the super admin that the vendor has been added successfully.
Alternate courses:	[ALT] Step 7: The super admin selects cancel. Use case terminates.	
	[ALT] Step 8: The validation is unsuccessful; the super admin is prompted to enter the details into the empty fields. Go back to step 7.	
	[ALT] Step 10: The super admin selects back. Go back to step 7.	
	[ALT] Step 11: The system fails to add the vendor into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the vendor has been added successfully.	
Post-condition:	The new vendor is successfully captured on the system.	
Business rules:	• Only a super admin can create a vendor.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 182: 6.1 Create Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Vendor	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Admin	
Description:	This use case describes the event in which a super admin wants to view a vendor on the system. The system will retrieve and display all the vendors on the system before the super admin searches using the vendor's name and selects a vendor to view. The system will retrieve the selected vendor's details and display them to the super admin. The use case ends once the system displays the selected vendor's details.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to view a vendor.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view a vendor.	**Step 2:** The system will invoke use case 6.3 (View Vendors).
	Step 3: The super admin searches for a vendor using the vendor's name. [ALT]	**Step 4:** The system filters and displays the vendor matching the criteria searched. [ALT]
	Step 5: The super admin selects the vendor.	**Step 6:** The system retrieves the selected vendor's details from the following tables based on type: • Vendor • VendorType • Accommodation • Caterer
		Step 7: The system displays the "View Vendor" popup displaying the selected vendor.

System Name: Integrated NGO Management System		
	Step 8: Super admin can view a vendor.	
Alternate courses:	[ALT] Step 3a: The super admin selects a displayed vendor. [ALT] Step 3b: The super admin filters the displayed vendors using the filter option.	
	[ALT] Step 4: The system fails to find any vendors with a matching name. Go back to step 3.	
Conclusion:	Super admin can view a vendor.	
Post-condition:	The vendor's details are displayed.	
Business rules:	• Super admins can view a vendor. • Admins can view a vendor.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 183: 6.2 Create Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Vendors	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Admin	
Description:	This use case describes the event in which a super admin wants to view all the vendors on the system. The system will retrieve all the vendors on the system and display them in a list to the super admin. The use case ends once all the vendors are displayed.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to view all the vendors.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view all the vendors.	
	Step 2: The super admin will navigate to the Inventory tab.	
	Step 3: The super admin selects vendor.	**Step 4:** The system retrieves all the vendors on the system from the following tables: • Vendor • VendorType • Accommodation • Caterer
		Step 5: The system displays the "Vendor View", displaying the following details for each vendor: • Name • Address • Email • Phone • Type

System Name: Integrated NGO Management System		
	Step 6: The super admin can view all the vendors.	
Alternate courses:	None	
Conclusion:	The super admin can view all the vendors.	
Post-condition:	The vendor's details are displayed.	
Business rules:	• Super admins can view all vendors. • Admins can view all vendors.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 184: 6.3 Create Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Edit Vendor	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to update a vendor on the system. The system allows the super admin to select a vendor and then update the vendor's details. The system will validate the details entered before asking the super admin to confirm the updated details. Once confirmed the system will inform the super admin that the vendor's details have been successfully updated. The use case ends once the vendor's updated details have been successfully captured on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to edit a vendor.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to edit a vendor.	**Step 2:** The system will invoke use case 6.2 (View Vendor).
	Step 3: The super admin will select update.	**Step 4:** The system displays the "Update Vendor" form with the vendors current details in editable fields.
		Step 5: The system retrieves the vendor types from the VendorType table.
	Step 6: The super admin enters the updated details. [ALT]	**Step 7:** The system validates that all required fields are completed. [ALT]

System Name: Integrated NGO Management System		
		Step 8: The system requests confirmation for the vendor to be updated.
	Step 9: The super admin confirms that the vendor can be updated. [ALT]	**Step 10:** The system saves the updated details in the following tables based on the vendor type: • Vendor • Accommodation • Caterer [ALT]
		Step 11: The system saves the transaction to the AuditLog table.
		Step 12: The system displays a message informing the super admin that the vendor was updated successfully.
Alternate courses:	[ALT] Step 6: The super admin selects cancel. Use case terminates.	
	[ALT] Step 7: The validation is unsuccessful; the super admin is prompted to enter the details into the empty fields. Go back to step 6.	
	[ALT] Step 9: The super admin selects back. Go back to step 6.	
	[ALT] Step 10: The system fails to update the vendor in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the vendor was updated successfully.	
Post-condition:	The vendor's updated details are successfully captured on the system.	
Business rules:	• Only a super admin can edit a vendor.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 185: 6.4 Edit Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Reactivate Vendor	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin reactivates a vendor on the system. The system will display the vendor and allow the super admin to reactivate the vendor. The system will ask the super admin to confirm the reactivation before the vendor is reactivated. The use case ends once the vendor is reactivated on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to reactivate a vendor.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to reactivate a vendor.	**Step 2:** The system will invoke use case 6.2 (View Vendor).
	Step 3: The super admin selects reactivate.	**Step 4:** The system requests confirmation for the vendor to be reactivated.
	Step 5: The super admin confirms that the vendor can be reactivated. [ALT]	**Step 6:** The system updates the status of the vendor to active in the Vendor table. [ALT]
		Step 7: The system saves the transaction to the AuditLog table.
		Step 8: The system displays a message informing the super admin that the vendor was reactivated successfully.

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.
	[ALT] Step 6: The system fails to update the status of the vendor in the relevant entities and displays an error message to the super admin. Use case terminates.
Conclusion:	The system displays a message informing the super admin that the vendor was reactivated successfully.
Post-condition:	The vendor is successfully reactivated on the system.
Business rules:	• Only a super admin can reactivate a vendor.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 186: 6.5 Reactivate Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Vendor	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to deactivate a vendor on the system. The system will display the vendor and allow the super admin to deactivate the vendor. The system will ask the super admin to confirm the deactivation before the vendor is deactivated. The use case ends once the vendor has been successfully deactivated on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to deactivate a vendor.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to deactivate a vendor.	**Step 2:** The system invokes use case 6.2 (View Vendor).
	Step 3: The super admin selects deactivate.	**Step 4:** The system requests confirmation for the vendor to be deactivated.
	Step 5: The super admin confirms that the vendor can be deactivated. [ALT]	**Step 6:** The system sets the status of the vendor to deactivated in the Vendor table. [ALT]
		Step 7: The system saves the transaction to the AuditLog table.
		Step 8: The system displays a message informing the super admin that the vendor was deactivated successfully.

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.
	[ALT] Step 6: The system fails to update the status of the vendor in the relevant entities and displays an error message to the super admin. Use case terminates.
Conclusion:	The system displays a message informing the super admin that the vendor was deactivated successfully.
Post-condition:	The vendor is successfully deactivated on the system.
Business rules:	• Only a super admin can deactivate a vendor.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 187: 6.6 Deactivate Vendor - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Vendor Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to add a new vendor type to the system. The system prompts the super admin to enter the new vendor type's details. The use case ends when the system saves the new vendor type onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to add a new vendor type to the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to add a new vendor type to the system.	**Step 2:** The system invokes UC 6.9 (View Vendor Types) to display a pop-up with a table showing all the current types.
	Step 3: The super admin selects add.	**Step 4:** The system displays an add vendor type form to provide the following information: • Type • Description
	Step 5: The super admin enters the new vendor type's details and selects save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the new vendor type to be added.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the new type onto the VendorType table. [ALT]

System Name: Integrated NGO Management System		
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the new group. Terminate use case.	
Conclusion:	The system saves the new type onto the VendorType table.	
Post-condition:	The new vendor type's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can add a new vendor type.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 188: 6.7 Create Vendor Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Vendor Types	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.8	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to update an existing vendor type on the system The system will display the selected vendor type's details and the super admin will be able to modify the vendor type's information. The use case ends when the system saves the updated vendor type's details onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to update a vendor type's details on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to update a vendor type's details on the system.	**Step 2:** The system invokes UC 6.9 (View Vendor Types) to display a pop-up with a table showing all the current types.
	Step 3: The super admin selects update in a specific row.	**Step 4:** The system displays the specific row's fields as editable. These include • Type • Description
	Step 5: The super admin updates the vendor type's details and clicks save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the changes to be made to the vendor type.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the updated vendor type's details onto the VendorType table. [ALT]

System Name: Integrated NGO Management System		
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the admin. Go back to step 2.	
	[ALT] Step 8: The super admin does not confirm the changes. Go back to step 2.	
	[ALT] Step 9: The system fails to save the changes. Terminate use case.	
Conclusion:	The system saves the updated vendor type's details onto the VendorType table.	
Post-condition:	The updated vendor type's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can update a vendor type.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 189: 6.8 Update Vendor Types - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Vendor Types	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.9	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when an admin wants to view the vendor types on the system. The system will display a list of vendor types and their details in a table. The use case ends when the admin can see the vendor types and their details on the system.	
Pre-condition:	The admin must be logged into the system.	
Trigger:	The admin wants to view the vendor types on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The admin wants to view the vendor types on the system.	**Step 2:** The system invokes UC 6.3 (View Vendors) to display the list of vendors.
	Step 3: The admin selects vendor types.	**Step 4:** The system retrieves the vendor types' details from the VendorType table.
		Step 5: The system displays a pop-up with a table showing all the current vendor types. The table includes: • Type • Description • Status • Actions
	Step 6: The admin can view the vendor types.	
Alternate courses:	None	

System Name: Integrated NGO Management System	
Conclusion:	The system displays a pop-up with a table showing all the current vendor types.
Post-condition:	The vendor type's details are displayed.
Business rules:	• Admins and super admins can view all vendor types on the system.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 190: 6.9 View Vendor Types - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Reactivate Vendor Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.10	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to reactivate a vendor type on the system. The system displays the inactive vendor types and allows the super admin to select a vendor type to reactivate. The use case ends when the vendor type has been successfully reactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to reactivate a vendor type on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to reactivate a vendor type on the system.	**Step 2:** The system invokes UC 6.9 (View Vendor Types) to display a pop-up with a table showing all the current types.
	Step 3: The super admin selects reactivate on a specific vendor type where the status is inactive. [ALT]	**Step 4:** The system requests confirmation for the vendor type reactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system updates the vendor type's status to active in the VendorType table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3a: The super admin changes their mind and closes the pop-up. Terminate use case.	

System Name: Integrated NGO Management System	
	[ALT] Step 3b: The super admin selects reactivate where the status is already active. The system displays a message to the super admin. Terminate use case.
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.
	[ALT] Step 6: The system fails to deactivate the vendor type. Terminate use case.
Conclusion:	The system updates the vendor type's status to active in the Vendor table.
Post-condition:	The inactive vendor type has been reactivated, and a confirmation message is shown.
Business rules:	• Only super admin can reactivate a vendor type.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 191: 6.10 Reactivate Vendor Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Vendor Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	6.11	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to deactivate a vendor type on the system. The system displays the active vendor types and allows the super admin to select a vendor type to deactivate. The use case ends when the vendor type has been successfully deactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to deactivate a vendor type on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to deactivate a vendor type on the system.	**Step 2:** The system invokes UC 6.9 (View Vendor Types) to display a pop-up with a table showing all the current types.
	Step 3: The super admin selects deactivate on a specific vendor type where the status is active. [ALT]	**Step 4:** The system requests confirmation for the vendor type deactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system updates the vendor type's status to inactive in the VendorType table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3a: The super admin changes their mind and closes the pop-up. Terminate use case.	

System Name: Integrated NGO Management System	
	[ALT] Step 3b: The super admin selects deactivate where the status is already inactive. The system displays a message to the super admin. Terminate use case.
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.
	[ALT] Step 6: The system fails to deactivate the vendor type. Terminate use case.
Conclusion:	The system updates the vendor type's status to inactive in the VendorType table.
Post-condition:	The active vendor type has been deactivated, and a confirmation message is shown.
Business rules:	• Only super admin can deactivate a vendor type.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 192 :6.11 Deactivate Vendor Type - Logical Use Case Narrative

2.8.7 Project Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Create Project	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super-Admin	
Primary system actor (PSA):	• Super-Admin	
Other participating actors:	• None	
Other interested stakeholders:	Users who have been added to the new project	
Description:	A Super-Admin will create a new project that will represent a new NGO goal. The system will open a form where the Super-Admin will input the project: name, dates, staff, teams, milestones and deadlines to help plan and organise the project. The system will capture all the information and alert all the relevant staff and create the schedule.	
Pre-condition:	The staff have all been previously added to the system. The staff roles have all been previously added to the system. The project type has been added to the system previously.	
Trigger:	The super admin clicks the “Create new project button” in the Projects nav bar or from the home widgets. A trigger would also be that the NGO wants to create a new segment for a new type of community service. Or they want to distinguish between events in different years so they create a new project.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click the “create new project button”	**Step 2:** The system will load the form displaying all the project: name, dates, staff, teams, milestones and deadlines for the admin to complete.
	Step 3: The super admin will complete the form, adding all	

System Name: Integrated NGO Management System		
	locations, staff and other import details to the form. [ALT] [ALT] [ALT]	
		Step 4: The system will verify all the fields are entered correctly and then save the new project and its details into the system. [ALT]
		Step 5: The system will automatically alert all staff members involved and add the project deadlines and details to be easily viewed.
	Step 6: The super admin receives confirmation that the project has been captured.	
Alternate courses:	[ALT] Step 3: The staff member is not on the system yet: the admin will click "Add user" and the pop-up for the use case "Add user" will appear so the admin can quickly add them and then return back to the form. [ALT] Step 3: The project type is not on the system yet: the admin will click "Add project type" and the pop-up for the use case "Add project type" will appear so the admin can quickly add them and then return back to the form. [ALT] Step 3: The role is not on the system yet: the admin will click "Add role" and the pop-up for the use case "Add role" will appear so the admin can quickly add them and then return to the form.	
	[ALT] Step 4: The fields have not been filled in correctly: The system will show the admin which field can't be verified, along with the reason, so they can easily try again.	
Conclusion:	The project, along with its details, have been successfully added to the system.	
Post-condition:	The new project is added to the list, and all staff members involved with the project have been sent details regarding the project's details and deadlines.	
Business rules:	• Secure Access • Role Based Access	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• The super admin is the person organising the project
Open issues:	• None

Table 193: 7.1 Create Project - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Project	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• User	
Primary system actor (PSA):	• User	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	A user clicks on a specific project to see all its details, like the name, staff, dates, project type, milestones, and deadline. They see this in the project's nav bar option.	
Pre-condition:	The user must already have a pre-existing account on the system The user is logged in	
Trigger:	The user wants to see a specific projects full details	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user clicks on a specific project from a list of projects on the "Projects" page. [ALT]	**Step 2:** The system retrieves the details of the project they selected, and it will display all the details in sections, relative to the user's role. [ALT]
	Step 3: The user can find the specific detail they need or just view the project details as a whole.	
Alternate courses:	[ALT] Step 1: The project has not yet been created, the user will click the "Create New Project" button	
	[ALT] Step 2: The system cannot display the details of the event correctly, the system will display an error message asking the user to "Try again later"	
Conclusion:	The user can view all the project details for a specific project, broken down into categories.	
Post-condition:	The project is displayed along will all other projects.	

System Name: Integrated NGO Management System	
Business rules:	• Secure Access • Role Based Access • Non admin users cannot access budgeting information
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.

Table 194: 7.2 View Project - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Update Project	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.3	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super-Admin	
Primary system actor (PSA):	• Super-Admin	
Other participating actors:	• Users involved in the project being edited	
Other interested stakeholders:	Administrator	
Description:	A Super-Admin will edit an existing project already saved in the system. The system will open the project's form where the Super-Admin will be able to edit the project: name, dates, staff, teams, milestones and deadlines to help plan and organise the project. The system will capture all the information and alert all the relevant staff and update the schedule.	
Pre-condition:	The project has already been created The staff have all been previously added to the system. The staff roles have all been previously added to the system. The project type has been added to the system previously.	
Trigger:	The super admin wants to change or update details regarding the project.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click on the project they wish to update or edit [ALT]	**Step 2:** The system will load the form displaying all the project: name, dates, staff, teams, milestones and deadlines for the admin to complete. [ALT]
	Step 3: The super admin will click the "edit project" button.	**Step 4:** The system will load the form displaying all the project: name, dates, staff, teams, milestones and deadlines and all other information as editable fields so the user can update what they need to.
	Step 5: The super admin will select the field they would like	**Step 6:** The system will verify all the fields are entered

System Name: Integrated NGO Management System		
	to edit and correct the projects details where needed. This can be: adding staff, changing the project name or type , changing the date or location or a number of other factors. [ALT] [ALT] [ALT]	correctly and then save the project updates to the system. [ALT]
		Step 7: The system will automatically alert all staff members involved and add the project deadlines and details to be easily viewed.
Alternate courses:	[ALT] Step 1: The project has not yet been created, the super admin must click “Create new project”	
	[ALT] Step 2: The project details have errors displaying, the system will display a message saying: “error, please restore the system”	
	[ALT] Step 5: The staff member is not on the system yet: the admin will click “Add user” and the pop-up for the use case “Add user” will appear so the admin can quickly add them and then return back to the form. [ALT] Step 5: The project type is not on the system yet: the admin will click “Add project type” and the pop-up for the use case “Add project type” will appear so the admin can quickly add them and then return back to the form. [ALT] Step 5: The role is not on the system yet: the admin will click “Add role” and the pop-up for the use case “Add role” will appear so the admin can quickly add them and then return to the form. [ALT] Step 6: The fields have not been filled in correctly: The system will show the admin which field can’t be verified, along with the reason, so they can easily try again.	
Conclusion:	The project has been successfully updated to meet the new project objectives.	
Post-condition:	The schedules of all users involved has been updated.	
Business rules:	• Secure Access • Role Based Access	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 195: 7.3 Update Project - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Deactivate Project	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	•	
Other interested stakeholders:	Administrator	
Description:	A super admin can deactivate projects after they are completed or if the project no longer supports the NPO's organisational goals.	
Pre-condition:	The project is already created and saved on the system	
Trigger:	The super admin wants to remove a project so users can no longer see it for any reason.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click on the project they wish to deactivate [ALT]	**Step 2:** The system will load the form displaying all the project: name, dates, staff, teams, milestones and deadlines of the project they wish to deactivate. [ALT]
	Step 3: The user will click the "Deactivate project button"	**Step 4:** The system will ask the user: "Are you sure you wish to deactivate" and then it will display the project name for confirmation.
	Step 5: Upon The user will confirm the wish to deactivate the project. [ALT]	**Step 6:** U The system will deactivate the project and show the project name and "has been deactivated". [ALT]
Alternate courses:	[ALT] Step 1: The project has not yet been created, the super admin must click "Create new project"	

System Name: Integrated NGO Management System	
	[ALT] Step 2: The project details have errors displaying, the system will display a message saying: “error, please restore the system” [ALT] Step 5: The admin doesn’t want to deactivate this project and clicks “Cancel” [ALT] Step 6: The project has an error deactivating and the system shows an error message to the admin to inform them their deactivation has been unsuccessful
Conclusion:	The project has been deactivated from the system.
Post-condition:	The admin has been notified of the project’s status.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 196: 7.4 Deactivate Project - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Create Project Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin would like to create a new category for projects that will represent a type of project.	
Pre-condition:	The user must already have a pre-existing account on the system The user has super admin privileges	
Trigger:	The super admin would like to create a new category for projects.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: After use case 7.6 (View Project type) the super admin will click the “Create project type ” button	**Step 2:** The system will display a form with the fields to add a new project type name.
	Step 3: The super admin will type the name of the new project type and click save. [ALT]	**Step 4:** The system will verify that the information has been filled in correctly and will then save it to the system.
		Step 5: Upon The system will notify the super admin that the new project type has been successfully captured . [ALT]
Alternate courses:	[ALT] Step 3: The user will click cancel if they dont want to add a new type	
	[ALT] Step 5: The system will tell the admin that there is already a role with that name [ALT] Step 5: The system will tell the admin that their project type has not been successfully captured	

System Name: Integrated NGO Management System	
Conclusion:	The new project type has been added to the system.
Post-condition:	The new project type appears in the list of project types.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 197: 7.5 Create Project Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Project Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	•	
Other interested stakeholders:	Administrator	
Description:	The super admin can click "View Project types" to be able to see, edit, add or deactivate project types to better categorise the projects being completed.	
Pre-condition:	The user must already have a pre-existing account on the system Ther user must have super admin privileges	
Trigger:	The super admin wants to see all project types.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks on "View project types" button on the projects page.	**Step 2:** The system retrieves the recorded project types and displays them in a table for the super admin to view. They can then click "update" or "deactivate" on existing project types or "Create new project type" [ALT]
Alternate courses:	[ALT] Step 2: There is an error retrieving the project types and the system displays an error message to the admin	
Conclusion:	The project types are displayed in a table for the admin to easily access or view.	
Post-condition:	Each project type has 2 buttons: "Update", "Deactivate" and there is a "create" project type button.	
Business rules:	• Secure Access • Role Based Access	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 198: 7.6 View Project Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Update Project Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Users	
Description:	A super admin wants to change the name of a project type. This is so users can better understand what those types of projects actually represent.	
Pre-condition:	The project type has already been added and captured onto the system	
Trigger:	A super admin wants to change the name of a project type.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks the "Update" button on the table displays, next to the project type they with to update.	**Step 2:** The system retrieves the recorded project type and displays the name as a field so the admin can edit it. [ALT]
	Step 3: The admin types the name of the updated project type. [ALT]	**Step 4:** The system captured the update and notifies the admin of the successful update. [ALT]
Alternate courses:	[ALT] Step 2: There is an error retrieving the project types so the systems displays an error message and suggested restoring the system.	
	[ALT] Step 3: The admin clicks the "cancel" button. [ALT] Step 4: The update couldn't be capture so the systems displays an error message	
Conclusion:	The project type details are updated	
Post-condition:	The system displays the updated project type.	

System Name: Integrated NGO Management System	
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• The super admin has the correct updated name of the project type
Open issues:	• None

Table 199: 7.7 Update Project Type - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Deactivate Project Type	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.8	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin deactivates a project type, and it will remove it from the displayed table of project types, this would be because a project type is not longer being used	
Pre-condition:	The project type has already been captured on the system	
Trigger:	The super admin wants to delete a project type no longer in use.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click on the project type they wish to deactivate from the table of all project types [ALT]	**Step 2:** The system will load the form displaying all the project. Type and its name that they wish to deactivate. [ALT]
	Step 3: The user will click the "Deactivate project type button"	**Step 4:** The system will ask the user: "Are you sure you wish to deactivate" and then it will display the project type name for confirmation.
	Step 5: Upon The user will confirm the wish to deactivate that project type. [ALT]	**Step 6:** The system will deactivate the project type and show the project type and "has been deactivated". [ALT]
Alternate courses:	[ALT] Step 1: The project type has not yet been created, the super admin must click "Create new project type"	
	[ALT] Step 2: The project type details have errors displaying, the system will display a message saying: "error, please restore the system" [ALT] Step 5: The admin doesn't want to deactivate this project type and clicks "Cancel"	

System Name: Integrated NGO Management System	
	[ALT] Step 6: The project type has an error deactivating and the system shows an error message to the admin to inform them their deactivation has been unsuccessful
Conclusion:	The project type has been deactivated.
Post-condition:	The project type is no longer displayed on view project types.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 200: 7.8 Deactivate Project Type - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Alexander Steven</td><td>Date: 25 March 2025
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Add Indemnity Form</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>7.9</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (PBA):</td><td colspan="2">• Super Admin</td></tr>
<tr><td>Primary system actor (PSA):</td><td colspan="2">• Super Admin</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">Administrator</td></tr>
<tr><td>Description:</td><td colspan="2">The super admin can add a new Indemnity form that can be added to projects to make form completions and capturing easier.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The admin is signed in and the form hasn’t been added yet.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The project requires a new type of indemnity form.</td></tr>
<tr><td rowspan="4">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The super admin clicks on “Add new indemnity form”</td><td>Step 2: The system shows fields to place a name for the form and text and or field to upload pdf file.</td></tr>
<tr><td>Step 3: The admin pastes the text and or pdf file and fills in a name
[ALT]</td><td>Step 4: The system verifies the fields and the form are correct before saving. [ALT]</td></tr>
<tr><td></td><td>Step 5: The system notifies the admin that the new form has been successfully captured .
[ALT]</td></tr>
<tr><td>Alternate courses:</td><td colspan="2">[ALT] Step 3: The user clicks “Cancel” to no longer add the form</td></tr>
<tr><td></td><td colspan="2">[ALT] Step 4: The new indemnity form details have errors saving, the system will display a message saying: “error, please restore the system”
[ALT] Step 5: The new indemnity form has an error saving and the system shows an error message to the admin to inform them their capture has been unsuccessful</td></tr>
<tr><td>Conclusion:</td><td colspan="2">The new indemnity form have been captures on the system</td></tr>
</table>

System Name: Integrated NGO Management System	
Post-condition:	The new indemnity form is shown with all indemnity forms in a table.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 201: 7.9 Add Indemnity Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	7.10	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Update Indemnity Form	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin can update an existing Indemnity form that can be added to projects to make form completions and capturing easier.	
Pre-condition:	The admin is signed in and the form hasn't been added yet.	
Trigger:	The project requires an update to an indemnity forms context in order to be more accurate.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks on "update indemnity form"	**Step 2:** The system shows fields to place a name for the form and text and or field to upload pdf file where the admin can add or edit fields.
	Step 3: The admin pastes the new text and or new pdf file and or changes the name [ALT]	**Step 4:** The system verifies the fields and the form are correct before saving. [ALT]
		Step 5: The system notifies the admin that the indemnity form has been successfully updated . [ALT]
Alternate courses:	[ALT] Step 3: The user clicks "Cancel" to no longer add the form [ALT] Step 4: The indemnity form details have errors updating, the system will display a message saying: "error, please restore the system" [ALT] Step 5: The indemnity form has an error updating and the system shows an error message to the admin to inform them their capture has been unsuccessful	
Conclusion:	The new indemnity form have been captures on the system	

System Name: Integrated NGO Management System	
Post-condition:	The new indemnity form is shown with all indemnity forms in a table.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 202: 7.10 Update Indemnity Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Indemnity Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.11	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin can click “View Indemnity Form” to be able to see, edit, add or deactivate Indemnity Forms to better help the projects.	
Pre-condition:	The user must already have a pre-existing account on the system Ther user must have super admin privileges	
Trigger:	The super admin wants to see all Indemnity forms.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks on “View Indemnity Forms” button on the projects page.	**Step 2:** The system retrieves the recorded Indemnity forms and displays them in a table for the super admin to view. They can then click “update” or “deactivate” on Indemnity forms or “Create new Indemnity Form” [ALT]
Alternate courses:	[ALT] Step 2: There is an error retrieving the Indemnity forms and the system displays an error message to the admin	
Conclusion:	The Indemnity forms are displayed in a table for the admin to easily access or view.	
Post-condition:		
Business rules:	• Secure Access • Role Based Access	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 203: 7.11 View Indemnity Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Remove Indemnity Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.12	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin can remove an existing indemnity form that is no longer being used to reduce clutter.	
Pre-condition:	The admin is signed in and the form hasn't been added yet.	
Trigger:	The admin wants to remove an unused form.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click on the indemnity form they wish to deactivate from the table of all indemnity forms [ALT]	**Step 2:** The system will load the form displaying all the indemnity form content and its name that they wish to deactivate. [ALT]
	Step 3: The user will click the "Deactivate indemnity form button"	**Step 4:** The system will ask the user: "Are you sure you wish to deactivate" and then it will display the indemnity form name for confirmation.
	Step 5: Upon The user will confirm the wish to deactivate that indemnity form. [ALT]	**Step 6:** The system will deactivate the indemnity form and show the indemnity form and "has been deactivated". [ALT]
Alternate courses:	[ALT] Step 1: The indemnity form has not yet been created, the super admin must click "Create new indemnity form"	
	[ALT] Step 2: The indemnity forms details have errors displaying, the system will display a message saying: "error, please restore the system" [ALT] Step 5: The admin doesn't want to deactivate this indemnity form and clicks "Cancel"	

System Name: Integrated NGO Management System	
	[ALT] Step 6: The indemnity form has an error deactivating, and the system shows an error message to the admin to inform them their deactivation has been unsuccessful
Conclusion:	The indemnity form has been deactivated.
Post-condition:	The indemnity form is no longer displayed on view indemnity forms.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 204: 7.12 Remove Indemnity Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Capture Indemnity Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.13	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	•	
Other interested stakeholders:	Administrator	
Description:	A super admin can attach an indemnity form to a project which will make it so that anyone who has been added to that project must complete the compulsory indemnity form in order to view the project details	
Pre-condition:	The super admin must already have a pre-existing account on the system	
Trigger:	The project would benefit from including an indemnity form.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks a box with the text ”Include an indemnity form”	**Step 2:** The system will display a list of the current indemnity forms for the admin to select from, along with a “create new indemnity form” button
	Step 3: The admin will select the indemnity form they require from the list of options. [ALT]	**Step 4:** The system will capture their selection and make a signature for that indemnity form be required for all staff in that project.
		Step 5: Upon creating the indemnity form task, the system will notify the admin of the successful [ALT]
Alternate courses:	[ALT] Step 3: Admin will create a new form, if there choice isn’t on the list	
	[ALT] Step 5: The system will give a notification error if unsuccessful	
Conclusion:	The indemnity form must now be required to be captured before all staff can interact with that project.	

System Name: Integrated NGO Management System	
Post-condition:	The required event is now visible to all users involved in the project.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 205: 7.13 Remove Indemnity Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Capture Project Feedback	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.14	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	•	
Other interested stakeholders:	Administrator	
Description:	A super admin can attach a project feedback form to a project which will make it so that anyone who has been added to that project will be send a form 1 week after the project commences asking them for a rating and additional comments to help improve the events.	
Pre-condition:	The project assigned to the feedback form must have commenced	
Trigger:	The super admin clicks a radio button called “Include Project Feedback form”.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks a box with the text “Include a Project Feedback form”	**Step 2:** The system will capture their selection and make a project feedback form for all staff in that project after the project ends.
		Step 3: Upon creating the indemnity form task, the system will notify the admin of the successful [ALT]
	[ALT] Step 3: Admin will be notified if there is an error.	
Conclusion:	The project feedback form will be displayed to all staff within that project, after the project is over	
Post-condition:	The form will open for all users after the event has finished	

System Name: Integrated NGO Management System	
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 206: 7.14 Capture Project Feedback - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Project Feedback	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.15	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A super admin can view all of the past projects and see the feedback forms ratings and comments for each project	
Pre-condition:	The projects have been saved on the system and not been deactivated	
Trigger:	The admin clicks on a specific project on view feedback forms. Or if the search a completed project in the projects nav bar	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The admin clicks on a specific project that has been completed, or they click on a specific project on the feedback page. [ALT]	**Step 2:** The system retrieves the records for the feedback ratings and comments for that project and displays it neatly for the admin to read. [ALT]
Alternate courses:	[ALT] Step 1: The admin clicks cancel if they are on the wrong project.	
	[ALT] Step 2: The system cannot open the information from the system and displays an error message.	
Conclusion:	The admin can see the feedback results of their desired project.	
Post-condition:	Active projects are all displayed with their feedback data inside.	
Business rules:	• Secure Access • Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	

Table 207: 7.15 View Project Feedback - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Create Impact Feedback Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.16	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin can add a new impact form that can be added to projects to make form completions and capturing easier for community members.	
Pre-condition:	The admin creating the form has permissions to create impact forms already.	
Trigger:	The admins would like to capture data from their projects impact or just for general use.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks on “Add new impact form”	**Step 2:** The system shows fields to place a name for the impact form and a button that says “Add question” where the admin can choose a question type and write the question. The admin can add as many questions as they want
	Step 3: The admin pastes the text and all their questions for the community partners. [ALT]	**Step 4:** The system verifies the fields and the impact form is correct before saving. [ALT]
		Step 5: The system notifies the admin that the new impact form has been successfully captured . [ALT]
Alternate courses:	[ALT] Step 3: The user clicks “Cancel” to no longer add the form [ALT] Step 4: The new indemnity form details have errors saving, the system will display a message saying: “error, please restore the system”	

System Name: Integrated NGO Management System	
	[ALT] Step 5: The new indemnity form has an error saving, and the system shows an error message to the admin to inform them their capture has been unsuccessful
Conclusion:	The new impact form have been captures on the system
Post-condition:	The new impact form is shown with all impact forms in a table.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 208: 7.16 Create Impact Feedback Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Impact Forms	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.17	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A super admin can view all of the past projects and see the impact forms questions and answers, and see what project they are assigned to.	
Pre-condition:	The projects have been saved on the system and not been deactivated	
Trigger:	The admin clicks on a specific project on view impact forms of. Or if the search a completed project in the projects nav bar	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The admin clicks on a specific project that has been completed, or they click on a specific project on the impact page. [ALT]	**Step 2:** The system retrieves the records for the impact questions and answers for that project and displays it neatly for the admin to read. [ALT]
Alternate courses:	[ALT] Step 1: The admin clicks cancel if they are on the wrong project.	
	[ALT] Step 2: The system cannot open the information from the system and displays an error message.	
Conclusion:	The admin can see the impact results of their desired project.	
Post-condition:	Active projects are all displayed with their feedback data inside.	
Business rules:	• Secure Access • Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None	

System Name: Integrated NGO Management System	
Open issues:	• None

Table 209: 7.17 View Impact Forms - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Update Impact Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.18	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• User	
Primary system actor (PSA):	• User	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The super admin can update an existing impact form that can be added to projects to make form completions and capturing easier. The questions we ask community partners can be updated	
Pre-condition:	The admin is signed in and the form hasn't been added yet.	
Trigger:	The project requires an update to an impact forms context in order to capture more accurate.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin clicks on "update impact form"	**Step 2:** The system shows fields to place a name for the form and the questions that are currently on the form
	Step 3: The admin pastes the new questions and question types and or changes the name of the impact form [ALT]	**Step 4:** The system verifies the fields and the impact form is correct before saving. [ALT]
		Step 5: The system notifies the admin that the impact form has been successfully updated . [ALT]
Alternate courses:	[ALT] Step 3: The user clicks "Cancel" to no longer add the form	
	[ALT] Step 4: The impact form details have errors updating, the system will display a message saying: "error, please restore the system"	

System Name: Integrated NGO Management System	
	[ALT] Step 5: The impact form has an error updating and the system shows an error message to the admin to inform them their capture has been unsuccessful
Conclusion:	The new impact form have been captures on the system
Post-condition:	The new impact form is shown with all impact forms in a table.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 210: 7.18 Update Impact Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Deactivate Impact Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.19	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	The admin can deactivate an impact form from the list of all the forms being displays, this can remove unused forms and clear clutter from the system.	
Pre-condition:	The impact form has already been created and is displayed on the list	
Trigger:	The admin wishes to remove a form that is no longer needed or being used.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will click on the impact form they wish to deactivate from the table of all impact forms [ALT]	**Step 2:** The system will load the form displaying all the impact form content and its name that they wish to deactivate. [ALT]
	Step 3: The user will click the "Deactivate impact form button"	**Step 4:** The system will ask the user: "Are you sure you wish to deactivate" and then it will display the impact form name for confirmation.
	Step 5: Upon The user will confirm the wish to deactivate that impact form. [ALT]	**Step 6:** The system will deactivate the impact form and show the impact form and "has been deactivated". [ALT]
Alternate courses:	[ALT] Step 1: The impact form has not yet been created, the super admin must click "Create new impact form"	
	[ALT] Step 2: The impact forms details have errors displaying, the system will display a message saying: "error, please restore the system"	

System Name: Integrated NGO Management System	
	[ALT] Step 5: The admin doesn't want to deactivate this impact form and clicks "Cancel" [ALT] Step 6: The impact form has an error deactivating and the system shows an error message to the admin to inform them their deactivation has been unsuccessful
Conclusion:	The impact form has been deactivated.
Post-condition:	The impact form is no longer displayed on view forms.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 211: 7.19 Deactivate Impact Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Capture Impact Form	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.20	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• Community Partner	
Other interested stakeholders:	Administrator	
Description:	When creating or editing a project the super admin can check the "Include impact form" button and choose from the list of impact forms or create a new one and add an email address to send the form to. The system will then capture the results and send the impact form.	
Pre-condition:	The impact form has already been captured in the system	
Trigger:	The project admin wants to capture impact data with regard to their project.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin will select a box that says: "Include impact form"	**Step 2:** The system retrieves the recorded impact forms for the user to choose from and adds a field to enter the contact's email address or phone number. [ALT]
	Step 3: The super admin chooses the impact form they wish to use and enters the community partners contact details. [ALT]	**Step 4:** The system verifies the number, email and fields and captures the questions. [ALT]
		Step 5: The system sends an email/ message to the community partner so they can answer the questions and the system will capture their answer. [ALT]
		Step 6: Once the form has been completed the system will notify the admin.

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 2: No impact forms could be retrieved, and we send an error message to the admin
	[ALT] Step 3: The admin clicks "Cancel" and returns to the viewing page. [ALT] Step 4: The fields are not correct, and we notify the admin of their error. [ALT] Step 5: There is an error sending the impact form, and we notify the admin that the form was unable to send.
Conclusion:	The impact form has been sent to the community partner.
Post-condition:	The community partner can open and complete the form.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 212: 7.20 Capture Impact Form - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Impact Feedback	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.21	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• Super Admin	
Primary system actor (PSA):	• Super Admin	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A super admin can view all of the project's impact results and see the impact form questions and answers, and see what project they are assigned to and which community partner answered.	
Pre-condition:	The impact forms have been saved on the system and not been deactivated	
Trigger:	The admin clicks on a specific impact form on view impact forms of. Or if the search a completed project in the projects nav bar	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The admin clicks on a specific impact form that has been completed, or they click on a specific project on the impact page. [ALT]	**Step 2:** The system retrieves the records for the impact questions and answers for that project and displays it neatly for the admin to read. [ALT]
Alternate courses:	[ALT] Step 1: The admin clicks cancel if they are on the wrong impact form.	
	[ALT] Step 2: The system cannot open the information from the system and displays an error message.	
Conclusion:	The admin can see the impact results of their impact form.	
Post-condition:	Active projects are all displayed with their feedback data inside.	
Business rules:	• Secure Access • Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	

System Name: Integrated NGO Management System	
Assumptions:	• None
Open issues:	• None

Table 213: 7.21 View Impact Feedback - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Add Photo	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.22	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• User	
Primary system actor (PSA):	• User	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A user can click on a project to see all photos from that project, they can then click the add photos button to upload multiple photos to that album.	
Pre-condition:	The project's photo group has already been added and the user has access to that project	
Trigger:	The user wishes to add more images to the project's image folder.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the project group they wish to add photos to in the photos page and clicks the "Add photos button"	**Step 2:** The system will load a page where the user can upload multiple images from their device. [ALT]
	Step 3: The user drags and drops or uploads their images to the photo album	**Step 4:** The system captures all the new images onto the existing photo album.
		Step 5: Upon the success of the image uploading, the system will notify the user of the success. [ALT]
Alternate courses:	[ALT] Step 2: The system has an error displaying the correct photo album and sends an error message to the user	
	[ALT] Step 5: The images are not uploaded successfully, and the system notifies the user of the error.	
Conclusion:	User's photos have been successfully added to the photo album.	

System Name: Integrated NGO Management System	
Post-condition:	The newly added images appear in the project's photo album.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None
Open issues:	• None

Table 214: 7.22 Add Photo - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	View Photo	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.23	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• User	
Primary system actor (PSA):	• User	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A user can click on a project to see all photos from that project, they can then click the project to see all photos in that album. All projects are displayed in a table with their names and the project photo.	
Pre-condition:	All the project's photo group has already been added and the user has access to those projects	
Trigger:	The user wishes to view the images in a project's image folder.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the project group they wish to view the photos of from a list of all the photo albums	**Step 2:** The system will load a page where the user can view multiple images from that project. [ALT]
Alternate courses:	[ALT] Step 2: The system has an error displaying the correct photo album and sends an error message to the user	
Conclusion:	The project's photos have been successfully displayed to the user.	
Post-condition:	The images appear in the project's photo album.	
Business rules:	• Secure Access • Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None	

System Name: Integrated NGO Management System	
Open issues:	• None

Table 215: 7.23 View Photo - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Alexander Steven	**Date:** 25 March 2025 **Version: 1.0**
Use case name:	Delete Photo	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	7.24	
Priority:	High	
Source:	EPI-USE	
Primary business actor (PBA):	• User	
Primary system actor (PSA):	• User	
Other participating actors:	• None	
Other interested stakeholders:	Administrator	
Description:	A user can click on a project to see all photos from that project, they can then click the delete photos button to make all photos in that album that they have uploaded have a remove button the user can press to remove them.	
Pre-condition:	The project's photo group has already been added and the user has access to that project They user uploaded the photos they wish to delete or they are a super admin	
Trigger:	The user wishes to remove images from the project's image folder.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the project group they wish to remove photos from in the photos page and clicks the "Remove photos button"	**Step 2:** The system will load a page where the user can see multiple images from the folder, with an "X" button they can press to remove that photo. [ALT]
	Step 3: The user clicks on all the photos they wish to delete and then click "done" [ALT]	**Step 4:** The system captures all the images to remove and removes them from the existing photo album.
		Step 5: Upon the success of the image removal, the system will notify the user of the success. [ALT]
Alternate courses:	[ALT] Step 2: The system has an error displaying the correct photo album and sends an error message to the user	
	[ALT] Step 3: The user clicks "Cancel" if they are on the wrong photo album.	

System Name: Integrated NGO Management System	
	[ALT] Step 5: The images are not uploaded successfully, and the system notifies the user of the error.
Conclusion:	User's photos have been successfully removed from the photo album.
Post-condition:	The removed images do not appear in the project's photo album.
Business rules:	• Secure Access • Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• The user is deleting photo's they uploaded or are a super admin
Open issues:	• None

Table 216: 7.24 Delete Photo - Logical Use Case Narrative

2.8.8 K9 Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Add Dog	**USE CASE TYPE** Business Requirements: ☐ System Analysis: ✓ System Design: ☐
Use case id:	8.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to add a new dog (K9) to the system. The system prompts the super admin to enter the new dog's details. The use case ends when the system saves the new dog onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to add a new dog to the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to add a new dog to the system.	**Step 2:** The system invokes UC 8.3 (View Dogs) to display the list of dogs.
	Step 3: The super admin selects to add a new dog.	**Step 4:** The system displays an add dog form to provide the following information: • Name • Breed • Age • Gender • Role • Status With optional fields which include: • Parents • Image
	Step 5: The super admin enters the dog's details and selects save. [ALT]	**Step 6:** The system validates the entered details. [ALT]

System Name: Integrated NGO Management System		
		Step 7: The system requests confirmation for the new dog to be added.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the new dog to the K9 table. [ALT]
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the form. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the new dog. Terminate use case.	
Conclusion:	The system saves the new dog to the K9 table.	
Post-condition:	The new dog's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can add a new dog.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 217: 8.1 Add Dog - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>View Dog</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>8.2</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Handler</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event when a handler wants to view a dog (K9) on the system. The handler searches for a dog on the system and then selects the dog to view. The use case ends when the system displays the selected dog’s details.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The handler must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The handler wants to view a dog on the system.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The handler wants to view a dog on the system.</td><td>Step 2: The system invokes UC 8.3 (View Dogs) to display the list of dogs.</td></tr>
<tr><td>Step 3: The handler searches for a dog by entering its name. [ALT]</td><td>Step 4: The system filters and displays the dog matching the search criteria. [ALT]</td></tr>
<tr><td>Step 5: The handler selects the dog.</td><td>Step 6: The system retrieves the selected dog’s details from the K9 table.</td></tr>
<tr><td></td><td>Step 7: The system displays a pop-up with all the dog’s details. These include:
• Name
• Image
• Breed
• Age
• Gender
• Role
• Status
• Handler
• Parents</td></tr>
</table>

System Name: Integrated NGO Management System		
	Step 8: The handler can view the dog's details.	
Alternate courses:	[ALT] Step 3a: The handler does not search for a dog and just selects the dog directly from the ones displayed. Go to Step 6.	
	[ALT] Step 3b: The handler uses the filter to search for a dog. Go to Step 4.	
	[ALT] Step 4: The system does not find any dogs matching the specified criteria. Go back to step 2.	
Conclusion:	The system displays all the dog's details.	
Post-condition:	All the related dog's details are displayed.	
Business rules:	• Handlers can only view the details of dogs they are assigned to. • Admins and super admins can view the details of all dogs on the system.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 218: 8.2 View Dog - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>View Dogs</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>8.3</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Handler</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event when a handler wants to view all dogs (K9) on the system. The handler selects the animal tab and then selects K9. The use case ends when the system displays the all dog on the system.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The handler must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The handler wants to view all dogs on the system.</td></tr>
<tr><td rowspan="6">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The handler wants to view all dogs on the system.</td><td></td></tr>
<tr><td>Step 2: The handler selects the animal tab.</td><td></td></tr>
<tr><td>Step 3: The handler then selects K9.</td><td>Step 4: The system retrieves the current dogs on the system from the K9 table.</td></tr>
<tr><td></td><td>Step 5: The system displays the current dogs based on user roles with the following details:
• Image
• Name
• Age
• Gender
• Breed
• Role</td></tr>
<tr><td>Step 6: The handler can view the dogs.</td><td></td></tr>
<tr><td>Alternate courses:</td><td colspan="2">None</td></tr>
</table>

System Name: Integrated NGO Management System	
Conclusion:	The system displays the current dogs on the system from the K9 table based on user roles.
Post-condition:	All the dogs on the system are displayed based on the user's permissions.
Business rules:	• Handlers can only view dogs assigned to them. • Admins and super admins can view all dogs on the system.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 219: 8.3 View Dogs - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Dog	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to update a dog's (K9) details on the system. The system will display the dog's details and then allow the super admin to update the dog. The use case ends when the system successfully saves the new details onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to update a dog's details on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to update a dog's details on the system.	**Step 2:** The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog's details.
	Step 3: The super admin selects to update the dog's details.	**Step 4:** The system displays an update dog form with all the following fields as editable: • Name • Breed • Age • Gender • Role • Parents • Image
	Step 5: The super admin updates the dog's details and selects save. [ALT]	**Step 6:** The system validates the updated data. [ALT]
		Step 7: The system requests confirmation for the changes to be made to the dog's details.

System Name: Integrated NGO Management System		
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the changes to the K9 table. [ALT]
		Step 10: The system records the transaction to the AuditLog table.
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The admin does not confirm the changes. Go back to step 4.	
	[ALT] Step 9: The system fails to save the changes. Terminate use case.	
Conclusion:	The system saves the changes to the K9 table.	
Post-condition:	The updated dog's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can update a dog's details.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 220: 8.4 Update Dog - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Dog	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to deactivate a dog (K9) on the system. The system will display the dog and then allow the super admin to deactivate the dog. The use case ends when the dog has been successfully deactivated on the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to deactivate a dog on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to deactivate a dog on the system.	**Step 2:** The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog's details.
	Step 3: The super admin selects deactivate. [ALT]	**Step 4:** The system requests confirmation for the dog to be deactivated.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system sets the dog's status to inactive in the K9 table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.	
	[ALT] Step 6: The system fails to change the status. Terminate use case.	

System Name: Integrated NGO Management System	
Conclusion:	The system sets the dog's status to deactivated in the K9 table.
Post-condition:	The dog has been deactivated on the system, and a confirmation message is shown.
Business rules:	• Only super admin can deactivate a dog.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 221: 8.5 Deactivate Dog - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Assign User to Dog	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• Handler (ERA)	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to assign a user (handler) to a dog (K9) on the system. The system will display the dog and then allow the super admin to assign a handler to them. The use case ends when the dog has been assigned a handler successfully on the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to assign a handler to a dog on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to assign a handler to a dog on the system.	**Step 2:** The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog's details.
	Step 3: The super admin selects to assign a handler.	**Step 4:** The system displays an assign handler form to allow a handler to be assigned to a dog.
	Step 5: The super admin selects a handler from a list and selects save. [ALT]	**Step 6:** The system requests confirmation for the handler to be assigned.
	Step 7: The super admin confirms. [ALT]	**Step 8:** The system creates a user-dog assignment in the K9User table, linking the selected handler to the dog. [ALT]
		Step 9: The system notifies the assigned handler of the new assignment.
		Step 10: The system records the transaction to the AuditLog table.

System Name: Integrated NGO Management System		
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 7: The super admin does not confirm. Go back to step 4.	
	[ALT] Step 8: The system fails to create the user-dog assignment. Terminate use case.	
Conclusion:	The system creates a user-dog assignment in the K9User table, linking the selected handler to the dog.	
Post-condition:	The dog has been assigned to a handler on the system, and a confirmation message is shown.	
Business rules:	• Only super admin can assign a handler to a dog. • A dog can only have one handler.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 222: 8.6 Assign User to Dog - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View User-Dog Assignment	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event when an admin wants to view the user (handler) to dog (K9) assignments currently on the system. The admin selects to view assignments, and the system displays a list of all dogs along with their currently assigned handlers. The use case ends when the admin can see the current assignments for all dogs.	
Pre-condition:	The admin must be logged into the system.	
Trigger:	The admin wants to view all the dogs and their current handlers on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The admin wants to view all the dogs and their current handlers on the system.	**Step 2:** The system invokes UC 8.3 (View Dogs) to display the list of dogs.
	Step 3: The admin selects to view the current assignments.	**Step 4:** The system retrieves the current assignments on the system from the K9User table.
		Step 5: The system displays a pop-up with a table showing all the current dog and handler assignments. The table includes: • Dog's Image • Dog name • Handler • Start Date [ALT]
	Step 6: The admin can view the current assignments.	

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 5: There are currently no active assignments, and the system displays a message to the admin. Terminate use case
Conclusion:	The system displays a pop-up with a table showing all the current dog and handler assignments from the K9User table
Post-condition:	All the current dog and handler assignments are displayed.
Business rules:	• Admins and super admins can view user-dog assignments.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 223: 8.7 View User-Dog Assignment - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Update User-Dog Assignment</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>8.8</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• Handler (ERA)</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event when a super admin wants to update a user (handler) to dog (K9) assignment on the system. The system will display the dog and then allow the super admin to change the handler from one to another. The interface uses a dynamic button that displays “Assign” when no handler exists and “Update” when a handler is already assigned. The use case ends when the dog’s handler has been changed successfully on the system.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The super admin must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The super admin wants to change a dog’s handler on the system.</td></tr>
<tr><td rowspan="6">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The super admin wants to change a dog’s handler on the system.</td><td>Step 2: The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog’s details.</td></tr>
<tr><td>Step 3: The super admin selects to change the handler.</td><td>Step 4: The system displays an assign handler form to allow the dog’s handler to be changed.</td></tr>
<tr><td>Step 5: The super admin selects a new handler from a list and selects save. [ALT]</td><td>Step 6: The system requests confirmation for the handler to be changed.</td></tr>
<tr><td>Step 7: The super admin confirms. [ALT]</td><td>Step 8: The system creates a new user-dog assignment in the K9User table, linking the selected handler to the dog. [ALT]</td></tr>
<tr><td></td><td>Step 9: The system notifies the old handler that they are no longer assigned to this dog.</td></tr>
</table>

System Name: Integrated NGO Management System		
		Step 10: System notifies the new handler that they have been assigned to this dog.
		Step 11: The system records the transaction to the AuditLog table.
		Step 12: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 7: The super admin does not confirm. Go back to step 4.	
	[ALT] Step 8: The system fails to update the user-dog assignment. Terminate use case.	
Conclusion:	The system creates a new user-dog assignment in the K9User table, linking the handler to the dog.	
Post-condition:	The dog has been assigned to a handler on the system, and a confirmation message is shown.	
Business rules:	• Only super admin can change a dog's handler. • A dog can only have one handler.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 224: 8.8 Update User-Dog Assignment - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Reactivate User-Dog Assignment	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.9	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• Handler (ERA)	
Other interested stakeholders:	• None	
Description:	This use case describes the event when a super admin wants to reactivate a user (handler) to dog (K9) assignment on the system. The system will display the dog and then allow the super admin to reactivate the dog's previous user-dog assignment. The use case ends when the dog's handler assignment has been successfully reactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to reactivate a user-dog assignment on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to reactivate a user-dog assignment on the system.	**Step 2:** The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog's details.
	Step 3: The super admin selects to reactivate the previous handler assignment. [ALT]	**Step 4:** The system searches the K9User table for the previous handler assignment. [ALT]
		Step 5: The system requests confirmation for the user-dog assignment reactivation.
	Step 6: The super admin confirms. [ALT]	**Step 7:** The system reactivates the previous assignment by creating a new record in the K9User table with the previous handler. [ALT]
		Step 8: System notifies the handler that they have been assigned to this dog.

System Name: Integrated NGO Management System		
		Step 9: The system records the transaction to the AuditLog table.
		Step 10: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 4a: The dog has never had a handler before. The system displays a message to the super admin. Terminate use case.	
	[ALT] Step 4b: The dog currently has a handler. The system searches the K9User table for the previous handler assignment, then deactivates the current handler assignment. Go to step 5.	
	[ALT] Step 6: The super admin does not confirm. Go back to step 2.	
	[ALT] Step 7: The system fails to reactivate the user-dog assignment. Terminate use case.	
Conclusion:	The system reactivates the previous assignment.	
Post-condition:	The dog's previous assignment has been reactivated, and a confirmation message is shown.	
Business rules:	• Only super admin can reactivate a user-dog assignment.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 225: 8.9 Reactivate User-Dog Assignment - Logical Use Case Narrative

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Deactivate User-Dog Assignment</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>8.10</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• Handler (ERA)</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event when a super admin wants to deactivate a user (handler) to dog (K9) assignment on the system. The system will display the dog and then allow the super admin to deactivate the dog's user-dog assignment. The use case ends when the dog's handler assignment has been successfully deactivated.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">The super admin must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">The super admin wants to deactivate a user-dog assignment on the system.</td></tr>
<tr><td rowspan="7">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: The super admin wants to deactivate a user-dog assignment on the system.</td><td>Step 2: The system invokes UC 8.2 (View Dog) to display a pop-up showing all the dog's details.</td></tr>
<tr><td>Step 3: The super admin selects to deactivate the current handler assignment. [ALT]</td><td>Step 4: The system identifies the current active assignment for the dog in the K9User table. [ALT]</td></tr>
<tr><td></td><td>Step 5: The system requests confirmation for the user-dog assignment deactivation.</td></tr>
<tr><td>Step 6: The super admin confirms. [ALT]</td><td>Step 7: The system deactivates the current assignment and displays the dog has no handler. [ALT]</td></tr>
<tr><td></td><td>Step 8: The system notifies the handler that they are no longer assigned to this dog.</td></tr>
</table>

System Name: Integrated NGO Management System		
		Step 9: The system records the transaction to the AuditLog table.
		Step 10: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 4: The dog does not have a current handler assignment. The system displays a message to the super admin. Terminate use case.	
	[ALT] Step 6: The super admin does not confirm. Go back to step 2.	
	[ALT] Step 7a: The system fails to deactivate the user-dog assignment. Terminate use case.	
Conclusion:	The system deactivates the current assignment and displays the dog has no handler.	
Post-condition:	The dog's current assignment has been deactivated, and a confirmation message is shown.	
Business rules:	• Only super admin can deactivate a user-dog assignment.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 226: 8.10 Deactivate User-Dog Assignment - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Add K9 Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.11	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to add a new dog (K9) role to the system. The system prompts the super admin to enter the new role's details. The use case ends when the system saves the new role onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to add a new K9 role to the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to add a new K9 role to the system.	**Step 2:** The system invokes UC 8.13 (View K9 Roles) to display a pop-up with a table showing all the current roles
	Step 3: The super admin selects add.	**Step 4:** The system displays an add K9 role form to provide the following information: • Role • Description
	Step 5: The super admin enters the new role's details. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the new role to be added.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the new role onto the K9Role table. [ALT]
		Step 10: The system records the transaction to the AuditLog table.

System Name: Integrated NGO Management System		
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 4.	
	[ALT] Step 8: The super admin does not confirm. Go back to step 4.	
	[ALT] Step 9: The system fails to save the new K9 role. Terminate use case.	
Conclusion:	The system saves the new role onto the K9Role table.	
Post-condition:	The new K9 role's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can add a new K9 role.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 227: 8.11 Add K9 Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update K9 Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.12	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to update an existing dog (K9) role on the system The system will display the selected role's details and the super admin will be able to modify the role's information. The use case ends when the system saves the updated role's details onto the system.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to update a K9 role's details on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to update a K9 role's details on the system.	**Step 2:** The system invokes UC 8.13 (View K9 Roles) to display a pop-up with a table showing all the current roles.
	Step 3: The super admin selects update in a specific row.	**Step 4:** The system displays the specific row's fields as editable: • Role • Description
	Step 5: The super admin updates the role's details and selects save. [ALT]	**Step 6:** The system validates the details entered. [ALT]
		Step 7: The system requests confirmation for the changes to be made to the role's details.
	Step 8: The super admin confirms. [ALT]	**Step 9:** The system saves the updated role's details onto the K9Role table. [ALT]
		Step 10: The system records the transaction to the AuditLog table.

System Name: Integrated NGO Management System		
		Step 11: The system displays a confirmation message.
Alternate courses:	[ALT] Step 5: The super admin changes their mind and closes the pop-up. Terminate use case.	
	[ALT] Step 6: The system finds an error and notifies the super admin. Go back to step 2.	
	[ALT] Step 8: The super admin does not confirm. Go back to step 2.	
	[ALT] Step 9: The system fails to save the changes. Terminate use case.	
Conclusion:	The system saves the updated role's details onto the K9Role table.	
Post-condition:	The updated K9 role's details have been saved onto the system, and a confirmation message is shown.	
Business rules:	• Only super admin can update a K9 role.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	• None	

Table 228: 8.12 Update K9 Role - Logical Use Case Narrative

<table>
<tr><th colspan="4">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td colspan="2">Nicholas Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td colspan="2">View K9 Roles</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td colspan="2">8.13</td></tr>
<tr><td>Priority:</td><td colspan="2">High</td></tr>
<tr><td>Source:</td><td colspan="2">EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="3">Handler</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="3">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="3">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="3">• None</td></tr>
<tr><td>Description:</td><td colspan="3">This use case describes the event of when a handler wants to view the dog (K9) roles on the system. The system will display a list of roles and their details in a table. The use case ends when the handler can see the roles and their details on the system.</td></tr>
<tr><td>Pre-condition:</td><td colspan="3">The handler must be logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="3">The handler wants to view the K9 roles on the system.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>Step 1: The handler wants to view the K9 roles on the system.</td><td colspan="2">Step 2: The system invokes UC 8.3 (View Dogs) to display the list of dogs.</td></tr>
<tr><td>Step 3: The handler selects roles.</td><td colspan="2">Step 4: The system retrieves the K9 roles’ details from the K9Role table.</td></tr>
<tr><td></td><td colspan="2">Step 5: The system displays a pop-up with a table showing all the current roles.
The table includes:
• Role
• Description
• Status
• Actions</td></tr>
<tr><td>Step 6: The handler can view the K9 roles.</td><td colspan="2"></td></tr>
<tr><td>Alternate courses:</td><td colspan="3">None</td></tr>
<tr><td>Conclusion:</td><td colspan="3">The system displays a pop-up with a table showing all the current roles.</td></tr>
</table>

System Name: Integrated NGO Management System	
Post-condition:	The K9 role's details are displayed.
Business rules:	• Handlers and admins can view all K9 roles on the system. • Super admins can view all K9 roles on the system.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 229: 8.13 View K9 Roles - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Reactivate K9 Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.14	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to reactivate a dog (K9) role on the system. The system displays the inactive roles and allows the super admin to select a role to reactivate. The use case ends when the K9 role has been successfully reactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to reactivate a K9 role on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to reactivate a K9 role on the system.	**Step 2:** The system invokes UC 8.13 (View K9 Roles) to display a pop-up with a table showing all the current roles.
	Step 3: The super admin selects reactivate on a specific role where the status is inactive. [ALT]	**Step 4:** The system requests confirmation for the role reactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system changes the role's status to active in the K9Role table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3a: The super admin changes their mind and closes the pop-up. Terminate use case.	

System Name: Integrated NGO Management System	
	[ALT] Step 3b: The super admin selects reactivate where the status is already active. The system displays a message to the super admin. Terminate use case.
	[ALT] Step 4: The super admin does not confirm. Go back to step 2.
	[ALT] Step 5: The system fails to change the role's status. Terminate use case.
Conclusion:	The system updates the role's status to active in the K9Role table.
Post-condition:	The inactive K9 role has been reactivated, and a confirmation message is shown.
Business rules:	• Only super admin can reactivate a K9 role.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 230: 8.14 Reactivate K9 Role - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate K9 Role	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	8.15	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event of when a super admin wants to deactivate a dog (K9) role on the system. The system displays the active roles and allows the super admin to select a role to deactivate. The use case ends when the K9 role has been successfully deactivated.	
Pre-condition:	The super admin must be logged into the system.	
Trigger:	The super admin wants to deactivate a K9 role on the system.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super admin wants to deactivate a K9 role on the system.	**Step 2:** The system invokes UC 8.13 (View K9 Roles) to display a pop-up with a table showing all the current roles.
	Step 3: The super admin selects deactivate on a specific role where the status is active. [ALT]	**Step 4:** The system requests confirmation for the role deactivation.
	Step 5: The super admin confirms. [ALT]	**Step 6:** The system updates the role's status to inactive in the K9Role table. [ALT]
		Step 7: The system records the transaction to the AuditLog table.
		Step 8: The system displays a confirmation message.
Alternate courses:	[ALT] Step 3a: The super admin changes their mind and closes the pop-up. Terminate use case.	

System Name: Integrated NGO Management System	
	[ALT] Step 3b: The super admin selects deactivate where the status is already inactive. The system displays a message to the super admin. Terminate use case.
	[ALT] Step 5: The super admin does not confirm. Go back to step 2.
	[ALT] Step 6: The system fails to change the role's status. Terminate use case.
Conclusion:	The system updates the role's status to inactive in the K9Role table.
Post-condition:	The active K9 role has been deactivated, and a confirmation message is shown.
Business rules:	• Only super admin can deactivate a K9 role.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	• None

Table 231: 8.15 Deactivate K9 Role - Logical Use Case Narrative

2.8.9 Rhino Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Add Rhino	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	9.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin adds a rhino to the system. The super admin will select to add a rhino and enter the rhino's information. The system will validate the details inputted before informing the super admin that the rhino has been successfully added. The use case ends once the rhino has successfully been added to the system.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to add a rhino.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to add a rhino.	**Step 2:** The system invokes use case 9.3 (View Rhinos)
	Step 3: The super admin will select add.	**Step 4:** The system displays the "Add Rhino" form to provide the following information: • Name • Species • Weight • DateOfBirth • Gender • Horn Status • Status • Last Seen • Identification notes • Approximate Location • Photo (Optional)

System Name: Integrated NGO Management System		
	Step 5: The super admin enters the details and selects add. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the rhino to be added.
	Step 8: The super admin confirms that the rhino can be added. [ALT]	**Step 9:** The system adds the rhino to the following tables: • Animal • Rhino [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the rhino has been added successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to add the rhino into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the animal has been added successfully.	
Post-condition:	The rhino has been added successfully to the system.	
Business rules:	• Only super admins can add rhinos to the system.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 232: 9.1 Add Rhino - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Rhino	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	9.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Admin	
Description:	This use case describes the event in which a super admin wants to view a rhino on the system. The system will retrieve and display all the rhinos on the system before the super admin searches using the rhino's name and selects a rhino to view. The system will retrieve the selected rhino's details and display them to the super admin. The use case ends once the system displays the selected rhino's details.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to view a rhino.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view a rhino.	**Step 2:** The system invokes use case 9.3 (View Rhinos).
	Step 3: The super admin searches for a rhino using the rhino's name. [ALT]	**Step 4:** The system filters and displays the rhino matching the criteria searched. [ALT]
	Step 5: The super admin selects the rhino.	**Step 6:** The system retrieves the selected rhino's details from the following tables: • Animal • Rhino
		Step 7: The system displays the "View Rhino" popup displaying the selected rhino's details including: • Name • Species • Weight • Age • Gender

System Name: Integrated NGO Management System		
		• Horn Status • Status • Last Seen • Identification notes • Approximate Location • Photo (Optional)
	Step 8: Super admin can view a rhino.	
Alternate courses:	[ALT] Step 3a: The super admin selects a displayed rhino. [ALT] Step 3b: The super admin filters the displayed rhinos using the filter option.	
	[ALT] Step 4: The system fails to find any rhinos with a matching name. Go back to step 3.	
Conclusion:	Super admin can view a rhino.	
Post-condition:	The rhino's details are displayed.	
Business rules:	• Super admins can view a rhino. • Admins can view a rhino on projects they are in.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 233: 9.2 View Rhino - Logical Use Case Narrative

<table>
<tr><th colspan="4">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td colspan="2">Christopher Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td colspan="2">View Rhinos</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td colspan="2">9.3</td></tr>
<tr><td>Priority:</td><td colspan="2">High</td></tr>
<tr><td>Source:</td><td colspan="2">EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="3">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="3">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="3">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="3">• None</td></tr>
<tr><td>Description:</td><td colspan="3">This use case describes the event in which a super admin wants to view all the rhinos on the system. The system will retrieve all the rhinos on the system and display them to the super admin. The use case ends once all the rhinos are displayed.</td></tr>
<tr><td>Pre-condition:</td><td colspan="3">Super admin is logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="3">Super Admin wants to view all the rhinos.</td></tr>
<tr><td rowspan="6">Typical Course of Events:</td><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>Step 1: Super Admin wants to view all the rhinos.</td><td colspan="2"></td></tr>
<tr><td>Step 2: The super admin will select the animal tab.</td><td colspan="2"></td></tr>
<tr><td>Step 3: The super admin selects rhino.</td><td colspan="2">Step 4: The system retrieves all the rhinos on the system from the following tables:
• Animal table
• Rhino table</td></tr>
<tr><td></td><td colspan="2">Step 5: The system displays the “Rhino View”, displaying the following details for each rhino:
• Name
• Age
• Species
• Approximate Location</td></tr>
<tr><td>Step 6: The super admin can view all the rhinos.</td><td colspan="2"></td></tr>
<tr><td>Alternate courses:</td><td colspan="3">None</td></tr>
<tr><td>Conclusion:</td><td colspan="3">The super admin can view all the rhinos.</td></tr>
</table>

System Name: Integrated NGO Management System	
Post-condition:	All rhinos on the system are displayed.
Business rules:	• Only super admins can view all rhinos.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 234: 9.3 View Rhino - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Rhino	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	9.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to update a rhino on the system. The system allows the super admin to select a rhino and then update the rhino's details. The system will validate the details entered before asking the super admin to confirm the updated details. Once confirmed the system will inform the super admin that the rhino's details have been successfully updated. The use case ends once the rhino's updated details have been successfully captured on the system.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to update a rhino.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super Admin wants to update a rhino.	**Step 2:** The system invokes use case 9.2 (View Rhino)
	Step 3: The super admin selects update.	**Step 4:** The system displays the "Update Rhino" form to update the following information: • Name • Species • Weight • DateOfBirth • Gender • Horn Status • Status • Last Seen • Identification notes • Approximate Location • Photo (Optional)

System Name: Integrated NGO Management System		
	Step 5: The super admin enters the updated details. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the rhino to be updated.
	Step 8: The super admin confirms that the rhino can be updated. [ALT]	**Step 9:** The system saves the updated details in the following tables: • Animal • Rhino [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the rhino was updated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to update the rhino in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the rhino was updated successfully.	
Post-condition:	The rhino's details are updated.	
Business rules:	• Only super admins can update a rhino.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 235: 9.4 Update Rhino - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Deactivate Rhino	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	9.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to deactivate a rhino on the system. The system will display the rhino and allow the super admin to deactivate the rhino. The system will ask the super admin to confirm the deactivation before the rhino is deactivated. The use case ends once the rhino has been successfully deactivated on the system.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to deactivate a rhino.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super Admin wants to update a rhino.	**Step 2:** The system invokes use case 9.2 (View Rhino)
	Step 3: The super admin selects deactivate.	**Step 4:** The system requests confirmation for the rhino to be deactivated.
	Step 5: The super admin confirms that the rhino can be deactivated. [ALT]	**Step 6:** The system sets the status of the rhino to deactivated in the Animal table. [ALT]
		Step 7: The system saves the transaction to the AuditLog table.
		Step 8: The system displays a message informing the super admin that the rhino was deactivated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6:	

System Name: Integrated NGO Management System	
	The system fails to update the status of the rhino in the relevant entities and displays an error message to the super admin. Use case terminates.
Conclusion:	The system displays a message informing the super admin that the rhino was deactivated successfully.
Post-condition:	The rhino is deactivated on the system.
Business rules:	• Only super admins can deactivate a rhino.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 236: 9.5 Deactivate Rhino - Logical Use Case Narrative

2.8.10 Elephant Subsystem Narratives

<table>
<tr><th colspan="3">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td>Christopher Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td>Add Elephant</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td>10.1</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Source:</td><td>EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="2">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="2">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="2">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="2">• None</td></tr>
<tr><td>Description:</td><td colspan="2">This use case describes the event in which a super admin adds an elephant to the system. The super admin will select to add an elephant and enter the elephant’s information. The system will validate the details inputted before informing the super admin that the elephant has been successfully added. The use case ends once the elephant has successfully been added to the system.</td></tr>
<tr><td>Pre-condition:</td><td colspan="2">Super admin is logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="2">Super Admin wants to add an elephant.</td></tr>
<tr><td rowspan="3">Typical Course of Events:</td><td>Actor Action</td><td>System Response</td></tr>
<tr><td>Step 1: Super Admin wants to add an elephant.</td><td>Step 2: The system invokes use case 10.3 (View Elephants)</td></tr>
<tr><td>Step 3: The super admin will select add.</td><td>Step 4: The system displays the “Add Elephant” form to provide the following information:
• Name
• Species
• Weight
• DateOfBirth
• Gender
• Tusk Status
• Status
• Last Seen
• Identification notes
• Approximate Location</td></tr>
</table>

System Name: Integrated NGO Management System		
		• Photo (Optional)
	Step 5: The super admin enters the details and selects add. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the elephant to be added.
	Step 8: The super admin confirms that the elephant can be added. [ALT]	**Step 9:** The system adds the elephant to the following tables: • Animal • Elephant [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the elephant has been added successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancellation. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful; the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 6.	
	[ALT] Step 9: The system fails to add the elephant into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the elephant has been added successfully.	
Post-condition:	The elephant has been added successfully to the system.	
Business rules:	• Only super admins can add elephants to the system.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 237: 10.1 Add Elephant - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Elephant	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	10.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Admin	
Description:	This use case describes the event in which a super admin wants to view an elephant on the system. The system will retrieve and display all the elephants on the system before the super admin searches using the elephant's name and selects an elephant to view. The system will retrieve the selected elephant's details and display them to the super admin. The use case ends once the system displays the selected elephant's details.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to view an elephant.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super Admin wants to view an elephant.	**Step 2:** The system invokes use case 10.3 (View Elephants).
	Step 3: The super admin searches for an elephant using the elephant's name. [ALT]	**Step 4:** The system filters and displays the elephant matching the criteria searched. [ALT]
	Step 5: The super admin selects the elephant.	**Step 6:** The system retrieves the selected elephant's details from the following tables: • Animal • Elephant
		Step 7: The system displays the "View Elephant" popup displaying the selected elephant's details including: • Name

System Name: Integrated NGO Management System		
		• Species • Weight • Age • Gender • Tusk Status • Status • Last Seen • Identification notes • Approximate Location • Photo (Optional)
	Step 8: Super admin can view an elephant.	
Alternate courses:	[ALT] Step 3a: The super admin selects a displayed elephant. [ALT] Step 3b: The super admin filters the displayed elephants using the filter option.	
	[ALT] Step 4: The system fails to find any elephants with a matching name. Go back to step 3.	
Conclusion:	Super admin can view an elephant.	
Post-condition:	The elephant's details are displayed.	
Business rules:	• Super admins can view an elephant. • Admins can view an elephant on projects they are in.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 238: 10.2 View Elephant - Logical Use Case Narrative

<table>
<tr><th colspan="4">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td colspan="2">Christopher Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td colspan="2">View Elephants</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td colspan="2">10.3</td></tr>
<tr><td>Priority:</td><td colspan="2">High</td></tr>
<tr><td>Source:</td><td colspan="2">EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="3">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="3">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="3">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="3">• None</td></tr>
<tr><td>Description:</td><td colspan="3">This use case describes the event in which a super admin wants to view all the elephants on the system. The system will retrieve all the elephants on the system and display them to the super admin. The use case ends once all the elephants are displayed.</td></tr>
<tr><td>Pre-condition:</td><td colspan="3">Super admin is logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="3">Super Admin wants to view all the elephants.</td></tr>
<tr><td rowspan="5">Typical Course of Events:</td><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>Step 1: Super Admin wants to view all the elephants.</td><td colspan="2"></td></tr>
<tr><td>Step 2: The super admin will select the animal tab.</td><td colspan="2"></td></tr>
<tr><td>Step 3: The super admin selects elephant.</td><td colspan="2">Step 4: The system retrieves all the elephants on the system from the following tables:
• Animal table
• Elephant table</td></tr>
<tr><td></td><td colspan="2">Step 5: The system displays the “Elephant View”, displaying the following details for each elephant:
• Name
• Age
• Species
• Approximate Location</td></tr>
<tr><td>Alternate courses:</td><td colspan="3">None</td></tr>
</table>

System Name: Integrated NGO Management System	
Conclusion:	The system displays the “Elephant View” with all the elephants on the system.
Post-condition:	All elephants on the system are displayed.
Business rules:	• Only super admins can view all elephants.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 239: 10.3 View Elephants - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Elephant	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	10.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin wants to update an elephant on the system. The system allows the super admin to select an elephant and then update the elephant's details. The system will validate the details entered before asking the admin to confirm the updated details. Once confirmed the system will inform the super admin that the elephant's details have been successfully updated. The use case ends once the elephant's updated details have been successfully captured on the system.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to update an elephant.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super Admin wants to update an elephant.	**Step 2:** The system invokes use case 10.2 (View Elephant)
	Step 3: The super admin selects update.	**Step 4:** The system displays the "Update Elephant" form to update the following information: • Name • Species • Weight • DateOfBirth • Gender • Tusk Status • Status

System Name: Integrated NGO Management System		
		• Last Seen • Identification notes • Approximate Location • Photo (Optional)
	Step 5: The super admin enters the updated details. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the elephant to be updated.
	Step 8: The super admin confirms that the elephant can be updated. [ALT]	**Step 9:** The system saves the updated details in the following tables: • Animal • Elephant [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the elephant was updated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to update the elephant in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the elephant was updated successfully.	
Post-condition:	The elephant's details are updated.	
Business rules:	• Only super admins can update an elephant.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	

System Name: Integrated NGO Management System	
Open issues:	• None

Table 240: 10.4 Update Elephant - Logical Use Case Narrative

System Name: Integrated NGO Management System			
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**	
Use case name:	Deactivate Elephant	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □	
Use case id:	10.5		
Priority:	High		
Source:	EPI-USE		
Primary business actor (pba):	Super Admin		
Primary system actor (psa):	• None		
Other participating actors:	• None		
Other interested stakeholders:	• None		
Description:	This use case describes the event in which a super admin wants to deactivate an elephant on the system. The system will display the elephant and allow the super admin to deactivate the elephant. The system will ask the super admin to confirm the deactivation before the elephant is deactivated. The use case ends once the elephant has been successfully deactivated on the system.		
Pre-condition:	Super admin is logged into the system.		
Trigger:	Super Admin wants to deactivate an elephant.		
Typical Course of Events:	**Actor Action**	**System Response**	
	Step 1: Super Admin wants to deactivate an elephant.	**Step 2:** The system invokes use case 10.2 (View Elephant)	
	Step 3: The super admin selects deactivate.	**Step 4:** The system requests confirmation for the elephant to be deactivated.	
	Step 5: The super admin confirms that the elephant can be deactivated. [ALT]	**Step 6:** The system sets the status of the elephant to deactivated in the Animal table. [ALT]	
		Step 7: The system saves the transaction to the AuditLog table.	
		Step 8: The system displays a message informing the super admin that the elephant was deactivated successfully.	
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.		

System Name: Integrated NGO Management System	
	[ALT] Step 6: The system fails to update the status of the elephant in the relevant entities and displays an error message to the super admin. Use case terminates.
Conclusion:	The system displays a message informing the super admin that the elephant was deactivated successfully.
Post-condition:	The elephant is deactivated on the system.
Business rules:	• Only super admins can deactivate an elephant.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 241: 10.5 Deactivate Elephant - Logical Use Case Narrative

2.8.11 Messages Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Message	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	11.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	User	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Group members (ERA)	
Description:	This use case describes the event when a user wants to send a message on a group to which they are assigned. The user will type out the message they wish to send. The use case ends when the user has sent the messages to a group.	
Pre-condition:	The user must be logged into the system.	
Trigger:	The user wants to send a message on a group to which they are assigned.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user wants to send a message to a group to which they are assigned.	**Step 2:** The system invokes UC 16.2 (View Messages) to display the group's messages.
	Step 3: The user types a message and then selects send. [ALT]	**Step 4:** The system validates the message. [ALT]
		Step 5: The system saves the new message to the Messages table, linking it to the user who sent it and the group it belongs to. [ALT]
		Step 6: The system updates the message view in real time for all members of the group.
		Step 7: The system notifies the all the group members of new message.

System Name: Integrated NGO Management System		
		Step 8: The system records the transaction to the AuditLog table.
Alternate courses:	[ALT] Step 3: The user changes their mind and closes the chat. Terminate use case.	
	[ALT] Step 4: The system finds an error and notifies the user. Go back to step 2.	
	[ALT] Step 5: The system fails to save the message. Terminate use case.	
Conclusion:	The system updates the message view in real time for all members of the group.	
Post-condition:	The system updates the messages on the group.	
Business rules:	• Users can only send messages on groups they are assigned to. • Admins and super admins can send messages on all groups.	
Implementation constraints and specifications:	• None	
Assumptions:	• None	
Open issues:	None	

Table 242: 11.1 Create Message - Logical Use Case Narrative

System Name: Integrated NGO Management System			
Author:	Nicholas Bailey	**Date: 14 March 2026** **Version: 1.0**	
Use case name:	View Messages	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □	
Use case id:	11.2		
Priority:	High		
Source:	EPI-USE		
Primary business actor (pba):	User		
Primary system actor (psa):	• None		
Other participating actors:	• None		
Other interested stakeholders:	• None		
Description:	This use case describes the event when a user wants to view a message on a group to which they are assigned. The user navigates to the messages tab and then selects a group. The use case ends when the user can see the messages on a group.		
Pre-condition:	The user must be logged into the system.		
Trigger:	The user wants to view messages on a group to which they are assigned.		
Typical Course of Events:	**Actor Action**	**System Response**	
	Step 1: The user wants to view messages on a group to which they are assigned.		
	Step 2: The user selects the messages tab and then selects a group.	**Step 3:** The system retrieves all the messages for the selected group from the Messages table. [ALT]	
		Step 4: The system displays the messages in chronological order to the user.	
	Step 5: The user views the messages.		
Alternate courses:	[ALT] Step 3: The group does not have any messages yet. Terminate use case.		
Conclusion:	The system displays messages to the user to view.		
Post-condition:	The system displays all the messages on the group.		

System Name: Integrated NGO Management System	
Business rules:	• Users can only view messages on groups they are assigned to. • Admins and super admins can view all messages on all groups.
Implementation constraints and specifications:	• None
Assumptions:	• None
Open issues:	None

Table 243: 11.2 View Messages - Logical Use Case Narrative

2.8.12 Report Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.1	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Generate Custom Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	A user with appropriate permissions creates a new report by selecting specific data fields and filters to retrieve tailored information from the system.	
Pre-condition:	The user must be logged in with "Generate Report" permissions.	
Trigger:	The user selects the "Create New Custom Report" option in the Reporting dashboard.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user enters a name and description for the report in the provided fields.	**Step 2:** The system provides a list of available data categories that can be used to make custom reports.
	Step 3: The user selects a category and selects specific fields.	**Step 4:** The system allows users to define filter criteria.
	Step 5: The user selects the “Generate” button	**Step 6:** The system validates the query. [ALT]
		Step 7: The system retrieves all required data from the database.
		Step 8: The system displays the report and stores a copy of the criteria used in the Report and ReportConfiguration tables: **Report table:** • ReportID [PK] • Name • Description • Date **ReportConfiguration Table:** • ReportConfigurationID [PK]

System Name: Integrated NGO Management System		
		• ReportID [FK • SelectedFields • Filter • ReportStartDate • ReportEndDate • Type
	Step 9: The User downloads the report in PDF format. [ALT]	**Step 10:** Custom report creation transaction is saved to the Audit Log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 6: The user selects conflicting filters that return no data, the system warns the user and suggests adjusting the criteria.	
	[ALT] Step 9: The user chooses not to download the report.	
Conclusion:	A custom report is created.	
Post-condition:	A new entry is saved in the Report Table.	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 244: 12.1 Generate Custom Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.2	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Custom Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	A user selects an existing custom report configuration to view the most up-to-date data presented in that specific format.	
Pre-condition:	At least one custom report must already exist in the system.	
Trigger:	The user selects an existing report from the "Saved Reports" list.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects a custom report from the list of previously defined reports.	**Step 2:** The system retrieves the saved configuration from the Report and ReportConfiguration tables. **Report table:** • ReportID [PK] • Name • Description • Date **ReportConfiguration Table:** • ReportConfigurationID [PK] • ReportID [FK • SelectedFields • Filter • ReportStartDate • ReportEndDate • Type
		Step 3: The system executes the query against the relevant data tables. [ALT]
		Step 4: The system displays the report

System Name: Integrated NGO Management System		
	Step 5: The User Downloads the report in PDF format. [ALT]	
Alternate courses:	[ALT] Step 3: No valid data is available, user is requested to update the report. (Use Case 12.3)	
	[ALT] Step 5: The user chooses not to download the report.	
Conclusion:	The user views the current data based on the saved report parameters.	
Post-condition:	The database reflects the updated report settings. .	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 245: 12.2 View Custom Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.3	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Update Custom Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	A user modifies the parameters of an existing custom report, such as adding a new column or changing the date range.	
Pre-condition:	The report must already exist on the system.	
Trigger:	The user selects "Edit Report" on a saved report configuration.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the custom report they wish to update.	**Step 2:** The system retrieves the saved configuration from the Report and ReportConfiguration tables. **Report table:** • ReportID [PK] • Name • Description • Date **ReportConfiguration Table:** • ReportConfigurationID [PK] • ReportID [FK • SelectedFields • Filter • ReportStartDate • ReportEndDate • Type
		Step 3: The system displays the configuration to the user, and allows the user to edit these fields.
	Step 4: The user modifies the provided fields.	**Step 5:** The system validates the new parameters. [ALT]

System Name: Integrated NGO Management System		
	Step 6: The user selects the "Save" Button.	**Step 7:** The system saves the changes to the custom report in the Report and ReportConfiguration tables and displays a confirmation message to the user. **Report table:** • ReportID [PK] • Name • Description • Date **ReportConfiguration Table:** • ReportConfigurationID [PK] • ReportID [FK • SelectedFields • Filter • ReportStartDate • ReportEndDate • Type
		Step 8: The system stores the transaction to the Audit Log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 5: The system could not successfully validate the provided parameters, and prompts the user to re-enter the those fields.	
Conclusion:	The report configuration is successfully modified.	
Post-condition:	None. .	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 246: 12.3 Update Custom Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.4	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Remove Custom Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	An administrator removes the custom report and its associated configuration from the database.	
Pre-condition:	The report must already exist on the system.	
Trigger:	The user selects "Remove Report" on a saved report configuration.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The administrator selects the “Remove” button when viewing a custom report.	**Step 2:** The system requests the user to confirm their intentions.
	Step 3: The user confirms the deactivation. [ALT]	**Step 4:** The removes the report and its configuration from the Report and Report Configuration tables. **Report table:** • ReportID [PK] • Name • Description • Date **ReportConfiguration Table:** • ReportConfigurationID [PK] • ReportID [FK • SelectedFields • Filter • ReportStartDate • ReportEndDate • Type

System Name: Integrated NGO Management System		
		Step 5: The system saves the Deactivation Transaction to the Audit Log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 3: The user does not confirm the removal; they are redirected to viewing the report.	
Conclusion:	Users can no longer see the custom report in the list of custom reports.	
Post-condition:	Custom report record is removed from the Report table.	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 247: 12.4 Remove Custom Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.5	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Bike Impact Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	Communities, Donors	
Description:	The user accesses a report that visualizes the impact of bike drops on communities by analysing historical attendance and learner performance data.	
Pre-condition:	User must be logged in. Bike location and community data must exist.	
Trigger:	The user selects "Bike Impact Report" from the reporting dashboard.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the desired period and specific community.	**Step 2:** The system retrieves data from the Inventory Transaction table for InventoryType bikes for projects of type bike mathing the provided criteria and information from the Project Impact table. **InventoryTransaction:** • InvetoryTransactionID [PK] • InventoryID [FK] • WarehouseID [FK] • ProjectID [FK] • TransactionType • ReferenceID • ReferenceType • TransactionDate • Quantity • DocumentID [FK] **ProjectImpact:** • ProjectImpactID • Description • ImpactType • GradeImprovement • AttendanceImprovement [ALT]

System Name: Integrated NGO Management System		
	Step 3: The user selects "View Report".	**Step 4:** The system displays the impact data visually on the screen
	Step 5: The user downloads the Report in PDF form. [ALT]	
Alternate courses:	[ALT] Step 2: No data could be found for the provided period or community; the system displays an error message.	
	[ALT] Step 5: The user decides not to download the report.	
Conclusion:	The user views the social impact metrics of the bicycle donation program.	
Post-condition:	None.	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 248: 12.5 View Bike Impact Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.6	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Budget Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	Financial Officers	
Description:	The user views a financial summary to monitor spending, compare actual costs against budgets, and support financial decision-making.	
Pre-condition:	User must be logged in. Budget data must exist.	
Trigger:	The user selects "Budget Report" and chooses a project.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the desired project and/or financial year.	**Step 2:** The system retrieves data from the Budget and Expense tables. **Budget:** • BudgetID [PK] • ProjectID [FK] • DateCreated • AllocatedAmount • CurrentBalance **Expense:** • ExpenseID [PK] • DocumentID [FK] • ExpenseItemName • Amount • Description [ALT]
	Step 3: The user selects "View Report".	**Step 4:** The system displays the budget data visually on the screen
	Step 5: The user downloads the Report in PDF form. [ALT]	
Alternate courses:	[ALT] Step 2: No data could be found for the provided period or project; the system displays an error message.	
	[ALT] Step 5: The user decides not to download the report.	

System Name: Integrated NGO Management System	
Conclusion:	The user assesses the financial health of the project.
Post-condition:	None.
Business rules:	• Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 249: 12.6 View Budget Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.7	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Rhino Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	Conservation Team	
Description:	The user views detailed rhino information to monitor conservation efforts. This includes location, population, de-horning and general animal information.	
Pre-condition:	User must be logged in. Rhino data must exist.	
Trigger:	User selects "View Rhino Report" from the menu.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the desired location for the report.	**Step 2:** The system retrieves data from the Rhino and Animal tables. **Animal:** • AnimalID [PK] • Name • Species • Weight • DateOfBirth • Gender • ActiveStatus • LastSeen • IdentificationNotes • ApproximateLocation • Photo **Rhino:** • RhinoID [PK] • AnimalID [FK] • HornStatus [ALT]
	Step 3: The user selects “View Report”.	**Step 4:** The system displays the rhino data visually on the screen
	Step 5: The user downloads the Report in PDF form. [ALT]	

System Name: Integrated NGO Management System	
Alternate courses:	[ALT] Step 2: No data could be found for the provided location; the system displays an error message.
	[ALT] Step 5: The user decides not to download the report.
Conclusion:	The user assesses overall Rhino information to gain a better understanding of conservation efforts.
Post-condition:	None.
Business rules:	• Role Based Access
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 250: 12.7 View Rhino Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.8	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Elephant Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	Conservation Team	
Description:	The user views detailed elephant information to monitor conservation efforts. This includes location, population, tusk-status and general animal information.	
Pre-condition:	User must be logged in. Elephant data must exist.	
Trigger:	User selects "View Elephant Report" from the menu.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects the desired location for the report.	**Step 2:** The system retrieves data from the Elephant and Animal tables. **Animal:** • AnimalID [PK] • Name • Species • Weight • DateOfBirth • Gender • ActiveStatus • LastSeen • IdentificationNotes • ApproximateLocation • Photo **Elephant** • ElephantID [PK] • AnimalID [FK] • TuskStatus [ALT]
	Step 3: The user selects “View Report”.	**Step 4:** The system displays the elephant data visually on the screen

System Name: Integrated NGO Management System		
	Step 5: The user downloads the Report in PDF form. [ALT]	
Alternate courses:	[ALT] Step 2: No data could be found for the provided location; the system displays an error message.	
	[ALT] Step 5: The user decides not to download the report.	
Conclusion:	The user assesses overall elephant information to gain a better understanding of conservation efforts.	
Post-condition:	None.	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 251: 12.8 View Elephant Report - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 26 March 2026 **Version:** 2.0
Use case name:	12.9	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View K9 Report	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	None.	
Other participating actors:	• None.	
Other interested stakeholders:	K9 Unit Managers	
Description:	The user views a report detailing the roles, health, and handler assignment of the K9 conservation units. If applicable genealogy will also be displayed.	
Pre-condition:	User must be logged in. Elephant data must exist.	
Trigger:	User selects "View K9 Report" from the menu.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user selects between a general report and a dog specific report.	**Step 2:** The system retrieves data from the K9K9Role, K9 and K9User tables. **K9:** • K9ID [PK] • K9MomID [FK] • K9DadID [FK] • Name • Breed • DateOfBirth • Status **K9K9Role:** • K9ID[PK1FK1] • K9RoleID [PK1FK1] **K9User:** • AssignmentID [PK] • K9ID [FK] • UserID [FK] • StartDate • EndDate [ALT]
	Step 3: The user selects “View Report”.	**Step 4:** The system displays the relevant K9 data visually on the screen

System Name: Integrated NGO Management System		
	Step 5: The user downloads the Report in PDF form. [ALT]	
Alternate courses:	[ALT] Step 2: No data could be found for the provided dog or overall report; the system displays an error message.	
	[ALT] Step 5: The user decides not to download the report.	
Conclusion:	The user views information regarding the K9 initiative to get a better understanding of the current K9 units.	
Post-condition:	None.	
Business rules:	• Role Based Access	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 252: 12.9 View K9 Report - Logical Use Case Narrative

2.8.13 Machine Learning Subsystem

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 25 March 2026 **Version: 1.0**
Use case name:	13.1	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	View Machine Learning Dashboard	
Priority:	Low	
Source:	EPI-USE	
Primary business actor (pba):	Administrator	
Primary system actor (psa):	• None.	
Other participating actors:	• User	
Other interested stakeholders:	None.	
Description:	This use case describes the event where a user wants to view all current Machine Learning Predictions in one place. Once the User requests to view this information, the Machine Learning Model runs with the most current data and displays it for the user.	
Pre-condition:	There should be enough data to accurately run the model,	
Trigger:	User selects the Machine Learning Dashboard.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: A user such as the administrator requests to view the Machine Learning dashboard.	**Step 2:** The system runs the most recent data for Risk Predictions, Team Predictions and budget Predictions through the Sci-kit learn model. [ALT]
		Step 3: The system displays the predictions by category and project.
Alternate courses:	[ALT] Step 1: The model fails to process all predictions; an error message is shown.	
Conclusion:	The dashboard is successfully displayed to the user.	
Post-condition:	The dashboard is successfully displayed to the user.	
Business rules:	• System must have Machine Learning Predictions.	

System Name: Integrated NGO Management System	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 253: 13.1 View Machine Learning Dashboard - Logical Use Case Narrative

2.8.14 System Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 25 March 2026 **Version: 1.0**
Use case name:	14.1	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Backup System	
Priority:	Low	
Source:	EPI-USE	
Primary business actor (pba):	Time	
Primary system actor (psa):	• None.	
Other participating actors:	• Super Administrator	
Other interested stakeholders:	• None.	
Description:	This use case describes the event where a super administrator makes a backup of the database in its current form. The action is also initiated automatically on a weekly basis, always keeping 4 automatic backups. The super administrator will select to create a backup, which prompts the system to make a backup of the database directory. The use case concludes when the database has been backed up and a success tis displayed to the super administrator.	
Pre-condition:	The super administrator must be logged in.	
Trigger:	Time initiates an automatic backup.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The system automatically initiates a backup at 03:00 every Sunday morning. **[ALT]**	**Step 2:** The system displays a message to all users logged in that they will be logged out until the system backup is finalised.
		Step 3: The system creates a copy of the database directory and adds details pertaining to the backup to the Backup table with the following attributes: **Backup:** • BackupID [PK] • Date • FilePath • Size
		Step 4: The system deletes the oldest backup to keep a maximum of 5 backups.

System Name: Integrated NGO Management System		
		[ALT]
		Step 5: System stores backup transaction to Audit Log **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 1: The super administrator manually initiates a backup at any time.	
	[ALT] Step 4: There are less than 5 backups, no backup is removed.	
Conclusion:	User is successfully authenticated and granted access.	
Post-condition:	An active session is created for the user.	
Business rules:	• Role Based Access • Ability to Backup Database.	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 254: 14.1 Backup System - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 25 March 2026 **Version: 1.0**
Use case name:	14.2	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	Restore System	
Priority:	Low	
Source:	EPI-USE	
Primary business actor (pba):	Super Administrator	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	This use case describes the event where the super administrator wants to restore the system's database from a previous backup. The system will display all available backups stored for the admin user to select from. The admin user will select the version they wish to restore to and select to submit. The system will display a confirmation message prompting the admin user to confirm the restoration of the system to that backup version. Upon receiving confirmation from the super administrator, the system will retrieve the backup version and restore the database to that previous state.	
Pre-condition:	The super administrator must be logged in.	
Trigger:	The super administrator requests to restore the system to a previous state.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super administrator requests to restore the system to a previous state.	**Step 2:** The system displays a list of stored backups referenced in the Backup table. **Backup:** • BackupID [PK] • Date • FilePath • Size
	Step 3: The super administrator selects the backup version they would like to restore to.	**Step 4**: The system displays a confirmation message to the super administrator.
	Step 5: The super administrator confirms their intention to restore the system to a previous backup. **[ALT]**	**Step 6:** The system Logs out all users before the Restore takes place.

System Name: Integrated NGO Management System		
		Step 7: The system replaces the current database directory with the version selected by the super administrator. [ALT]
		Step 8: The system records the transaction to the audit log. **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry
Alternate courses:	[ALT] Step 5: The super administrator does not provide confirmation to restore the system. Process is cancelled.	
	[ALT] Step 6: The system fails to restore to a previous state.	
Conclusion:	User is successfully authenticated and granted access. The system displays an error message, and the use case terminates.	
Post-condition:	An active session is created for the user.	
Business rules:	• Role Based Access • Ability to Backup Database.	
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.	
Assumptions:	• None.	
Open issues:	• None.	

Table 255: 14.2 Restore System - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Stephan Bredenhann	**Date:** 25 March 2026 **Version: 1.0**
Use case name:	View Audit Log	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	14.3	
Priority:	Low	
Source:	EPI-USE	
Primary business actor (pba):	Super Administrator	
Primary system actor (psa):	• None.	
Other participating actors:	• None.	
Other interested stakeholders:	None.	
Description:	This use case describes the event in which the super administrator would like to view all the audit logs within the system. The use case terminates once all the audit log is displayed.	
Pre-condition:	The super administrator must be logged in.	
Trigger:	The super administrator requests to view the full audit log.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The super administrator requests to view all audit logs.	**Step 2:** The system retrieves all fields from the AuditLog table and displays them in a table view. Each entry is a record in the AuditLog table including the following attributes: **AuditLog:** • AuditLogID [PK] • UserID [FK] • Timestamp • LogEntry **[ALT]**
Alternate courses:	[ALT] Step 2: The system fails to load the audit log data and displays an error message.	
Conclusion:	The system has successfully retrieved a list of all the audit logs within the system.	
Post-condition:	The system has successfully retrieved a list of all the audit logs within the system.	

System Name: Integrated NGO Management System	
Business rules:	• Role Based Access • Ability to Backup Database.
Implementation constraints and specifications:	• Must be completed before project day. • Must be user-friendly and easy to use.
Assumptions:	• None.
Open issues:	• None.

Table 256: 14.3 View Audit Log - Logical Use Case Narrative

2.8.15 Budget Subsystem Narratives

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Create Project Budget	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.1	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin creates a project budget. The system will allow the super admin to select a project and add a budget. The super admin will enter the projects budget and the system will validate the inputted data. The system will prompt the super admin for confirmation before the budget is saved. The use case ends once the project's budget has been created and captured.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to create a project budget.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to create a project budget.	
	Step 2: The super admin navigates to the budget tab.	
	Step 3: The super admin selects create budget.	**Step 4:** The system displays the "Create Project Budget" form to provide the following information: • Project • Amount
		Step 5: The system retrieves active projects from the Project table.

System Name: Integrated NGO Management System		
	Step 6: The super admin selects an active project and enters the budget. [ALT]	**Step 7:** The system validates that all required fields are completed. [ALT]
		Step 8: The system requests confirmation for the budget to be created.
	Step 9: The super admin confirms that the budget can be created. [ALT]	**Step 10:** The system adds the budget to the Budget table. [ALT]
		Step 11: The system saves the transaction to the AuditLog table.
		Step 12: The system displays a message informing the super admin that the project budget has been created successfully.
Alternate courses:	[ALT] Step 6: The super admin selects cancel. Use case terminates.	
	[ALT] Step 7: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 6.	
	[ALT] Step 9: The super admin selects back. Go back to step 6.	
	[ALT] Step 10: The system fails to add the budget into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the project budget has been created successfully.	
Post-condition:	The projects budget is successfully created and captured.	
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 257: 15.1 Create Project Budget - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Project Budget	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.2	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• Admin	
Description:	This use case describes the event in which a super admin wants to view a project budget on the system. The system will retrieve and display all the budgets of the active projects on the system before the super admin selects a budget to view. The system will retrieve the selected project's budget as well as expenses and display them to the super admin. The use case ends once the system displays the selected project's budget.	
Pre-condition:	Super admin is logged into the system.	
Trigger:	Super Admin wants to view a project budget.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view a project budget.	**Step 2:** The system invokes use case 15.3 (View Project Budgets).
	Step 3: The super admin selects the project budget.	**Step 4:** The system retrieves the selected project's budget and expenses from the following tables: • Project • Budget • Expense • ProjectExpense
		Step 7: The system displays the "View Project Budget", displaying the selected project budget as well as expenses.
	Step 8: Super admin can view a project budget.	
Alternate courses:	None	

System Name: Integrated NGO Management System	
Conclusion:	Super admin can view a project budget.
Post-condition:	The project budget is displayed.
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 258: 15.2 View Project Budget - Logical Use Case Narrative

<table>
<tr><th colspan="4">System Name: Integrated NGO Management System</th></tr>
<tr><td>Author:</td><td colspan="2">Christopher Bailey</td><td>Date: 14 March 2026
Version: 1.0</td></tr>
<tr><td>Use case name:</td><td colspan="2">View Project Budgets</td><td rowspan="4">USE CASE TYPE
Business Requirements: □
System Analysis: ✓
System Design: □</td></tr>
<tr><td>Use case id:</td><td colspan="2">15.3</td></tr>
<tr><td>Priority:</td><td colspan="2">High</td></tr>
<tr><td>Source:</td><td colspan="2">EPI-USE</td></tr>
<tr><td>Primary business actor (pba):</td><td colspan="3">Super Admin</td></tr>
<tr><td>Primary system actor (psa):</td><td colspan="3">• None</td></tr>
<tr><td>Other participating actors:</td><td colspan="3">• None</td></tr>
<tr><td>Other interested stakeholders:</td><td colspan="3">• None</td></tr>
<tr><td>Description:</td><td colspan="3">This use case describes the event in which a super admin wants to view all the project budgets on the system. The system will retrieve all the project budgets on the system and display them to the super admin. The use case ends once all the project budgets are displayed.</td></tr>
<tr><td>Pre-condition:</td><td colspan="3">Super admin is logged into the system.</td></tr>
<tr><td>Trigger:</td><td colspan="3">Super Admin wants to view all the project budgets.</td></tr>
<tr><td rowspan="6">Typical Course of Events:</td><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>Step 1: Super Admin wants to view all the project budgets.</td><td colspan="2"></td></tr>
<tr><td>Step 2: The super admin will select the budget tab.</td><td colspan="2"></td></tr>
<tr><td>Step 3: The super admin selects view budget.</td><td colspan="2">Step 4: The system retrieves all the active project budgets on the system from the following tables:
• Project
• Budget</td></tr>
<tr><td></td><td colspan="2">Step 5: The system displays the “Project Budget View”, displaying the following details for each project:
• Name
• Budget</td></tr>
<tr><td>Step 6: The super admin can view all the project budgets.</td><td colspan="2"></td></tr>
<tr><td>Alternate courses:</td><td colspan="3">None</td></tr>
</table>

System Name: Integrated NGO Management System	
Conclusion:	The super admin can view all the project budgets.
Post-condition:	All project budgets on the system are displayed.
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 259: 15.3 View Project Budgets - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update Project Budget	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.4	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin wants to update a project's budget. The system will display projects and their budget and allow the super admin to select and update a project's budget. The system will validate the details entered before asking the super admin to confirm the updated details. Once confirmed the system will inform the super admin that the project's budget has been successfully updated. The use case ends once the project's updated budget is captured on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to update a project budget.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to update a project budget.	**Step 2:** The system invokes use case 15.3 (View Project Budgets)
	Step 3: The super admin selects update.	**Step 4:** The system displays the "Update Budget" form to update the budget amount.
	Step 5: The super admin enters the updated budget. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the budget to be updated.

System Name: Integrated NGO Management System		
	Step 8: The super admin confirms that the budget can be updated. [ALT]	**Step 9:** The system saves the updated budget in the Budget table. [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the budget was updated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin does not confirm. Go back to step 5.	
	[ALT] Step 9: The system fails to update the budget in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the budget was updated successfully.	
Post-condition:	The project's budget is successfully updated.	
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 260: 15.4 Update Project Budget - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Log An Expense	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.5	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which an expense is logged. The system will display the project's budget and allow the super admin to add an expense by entering the expenses details and the system will validate the inputted data before asking for confirmation. Once confirmed, the system will capture the expense. The use case ends once the expense is captured on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to log an expense.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to log an expense.	**Step 2:** The system invokes use case 15.2 (View Project Budget)
	Step 3: The super admin will select to add an expanse.	**Step 4:** The system displays the "Log Expense" form to provide the following information: • Name • Description • Amount
	Step 5: The super admin enters the expense's details and selects add. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]
		Step 7: The system requests confirmation for the expense to be logged.

System Name: Integrated NGO Management System		
	Step 8: The super admin confirms that the expense can be logged. [ALT]	**Step 9:** The system adds the expense to the following tables: • Expense • ProjectExpense [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the expense has been logged successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to add the expense into the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the expense has been logged successfully.	
Post-condition:	The expense is added to the system.	
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature. • There should be sufficient budget to log an expense, if this is not the case, the user will be informed, and the expense will not be logged.	
Assumptions:	• None	
Open issues:	• None	

Table 261: 15.5 Log An Expense UCD

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Update An Expense	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.6	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin updates an expense. The system will display the expense's details and allow the super admin to update its details. The super admin will enter the updated details and the system will validate the updated details before asking the super admin to confirm the updated details. Once confirmed the system will inform the super admin that the updated details have been successfully captured. The use case ends once the expenses updated details are captured on the system.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to update an expense.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to update an expense.	**Step 2:** The system will invoke use case 15.7 (View Expense).
	Step 3: The super admin will select update.	**Step 4:** The system displays the "Update Expense" form to update the following: • Name • Description • Amount
	Step 5: The super admin enters the updated details. [ALT]	**Step 6:** The system validates that all required fields are completed. [ALT]

System Name: Integrated NGO Management System		
		Step 7: The system requests confirmation for the expense to be updated.
	Step 8: The super admin confirms that the expense can be updated. [ALT]	**Step 9:** The system saves the updated details in the Expense table. [ALT]
		Step 10: The system saves the transaction to the AuditLog table.
		Step 11: The system displays a message informing the super admin that the expense was updated successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The validation is unsuccessful, the super admin is prompted to enter the details into the empty fields. Go back to step 5.	
	[ALT] Step 8: The super admin selects back. Go back to step 5.	
	[ALT] Step 9: The system fails to update the expense in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the expense was updated successfully.	
Post-condition:	The expense's details are updated.	
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature. • There should be sufficient budget to log an expense if the value is updated, if this is not the case, the user will be informed, and the update will not be logged.	
Assumptions:	• None	
Open issues:	• None	

Table 262: 15.6 Update An Expense - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	View Expense	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.7	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which a super admin want to view an expanse. The system will display the project budget and allow the super admin to select an expense to view. The use case ends once the expense is displayed for the super admin.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to view an expense.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to view the expense.	**Step 2:** The system invokes use case 15.2 (View Project Budget)
	Step 3: The super admin selects an expense	**Step 4:** The system retrieves the selected expense's details form the Expense table.
		Step 5: The system displays the "View Expense" popup displaying the selected expense's details including: • Name • Description • Amount • Date Logged
	Step 6: The super admin can view an expense.	
Alternate courses:	None	

System Name: Integrated NGO Management System	
Conclusion:	The super admin can view an expense.
Post-condition:	The expense's details are displayed.
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.
Assumptions:	• None
Open issues:	• None

Table 263: 15.7 View Expense - Logical Use Case Narrative

System Name: Integrated NGO Management System		
Author:	Christopher Bailey	**Date: 14 March 2026** **Version: 1.0**
Use case name:	Remove Expense	**USE CASE TYPE** Business Requirements: □ System Analysis: ✓ System Design: □
Use case id:	15.8	
Priority:	High	
Source:	EPI-USE	
Primary business actor (pba):	Super Admin	
Primary system actor (psa):	• None	
Other participating actors:	• None	
Other interested stakeholders:	• None	
Description:	This use case describes the event in which the super admin removes an expense. The system will display the projects budget and expenses and allow the super admin to remove an expense. The system will ask for confirmation before the expense is removed. Once confirmed the system will remove the expense. The use case ends once an expenses is removed.	
Pre-condition:	Super admin is logged in.	
Trigger:	Super admin wants to remove an expense.	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: Super admin wants to remove an expense.	**Step 2:** The system will invoke use case 15.7 (View Expense).
	Step 3: The super admin will select remove.	**Step 4:** The system requests confirmation for the expense to be removed.
	Step 5: The super admin confirms that the expense can be removed. [ALT]	**Step 6:** The system removes the expense from the following tables: • Expense • Project Expense [ALT]
		Step 7: The system saves the transaction to the AuditLog table.

System Name: Integrated NGO Management System		
		Step 8: The system displays a message informing the super admin that the expense was removed successfully.
Alternate courses:	[ALT] Step 5: The super admin selects cancel. Use case terminates.	
	[ALT] Step 6: The system fails to remove expense in the relevant entities and displays an error message to the super admin. Use case terminates.	
Conclusion:	The system displays a message informing the super admin that the expense was removed successfully.	
Post-condition:	The expense is removed.	
Business rules:	• Financial data such as this is privileged, and thus only persons with appropriate roles may view it.	
Implementation constraints and specifications:	• The user requires an Internet connection to access this feature.	
Assumptions:	• None	
Open issues:	• None	

Table 264: 15.8 Remove Expense UCD

2.9 Primitive Data Flow Diagrams

2.9.1 User Subsystems Primitives

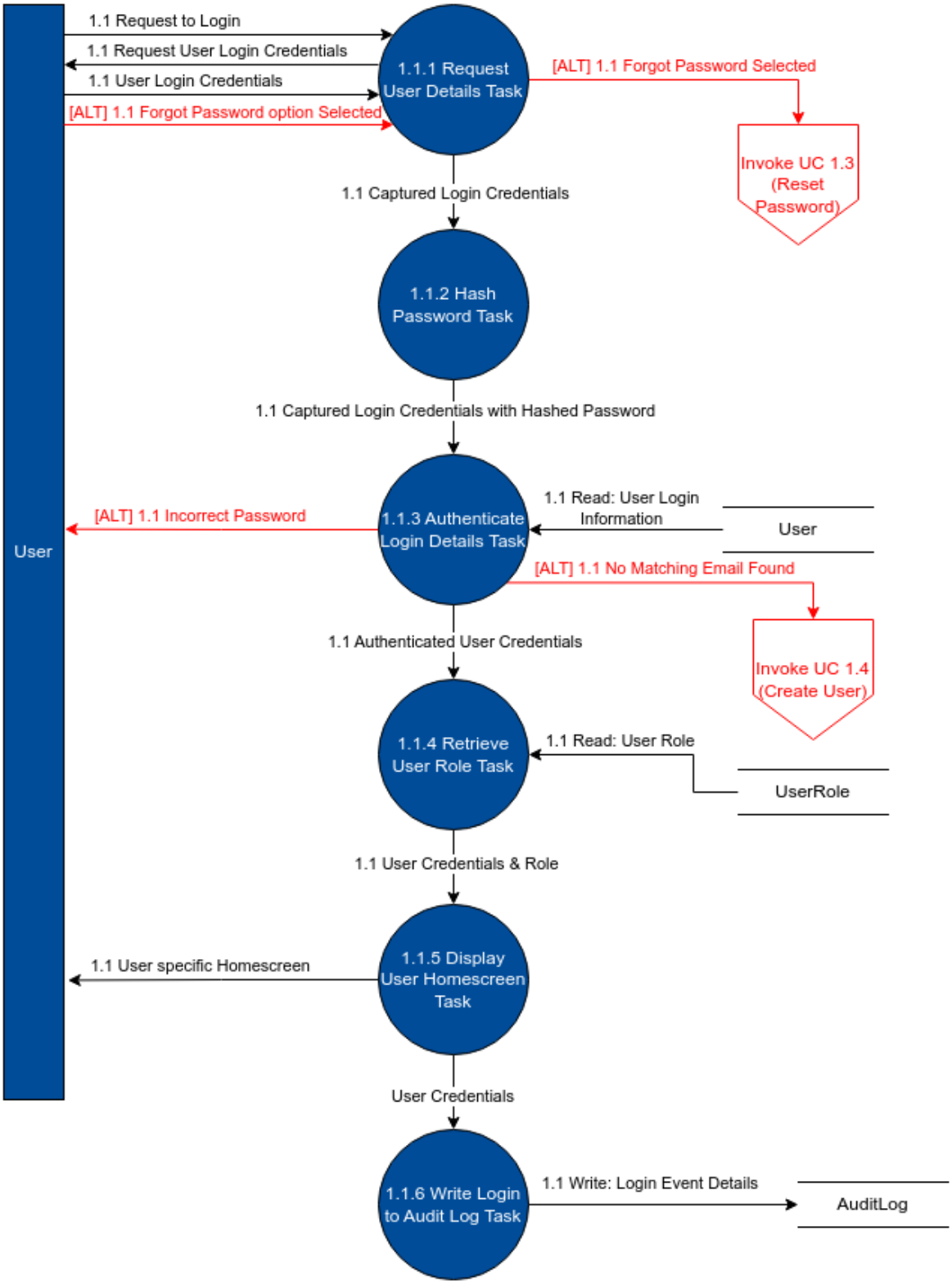


Figure 18: 1.1 User Login - Primitive Data Flow Diagram

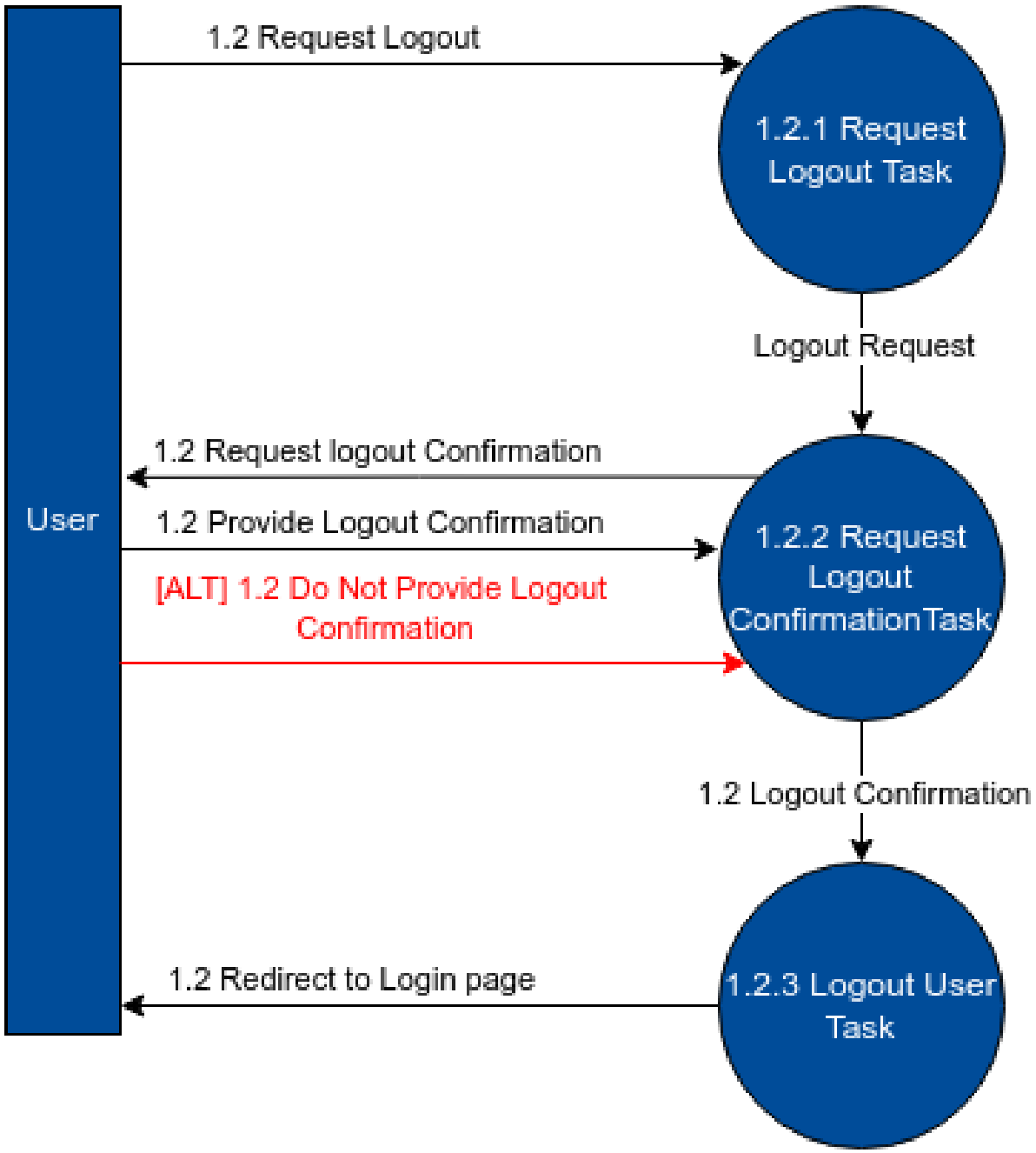


Figure 19: 1.2 User Logout - Primitive Data Flow Diagram

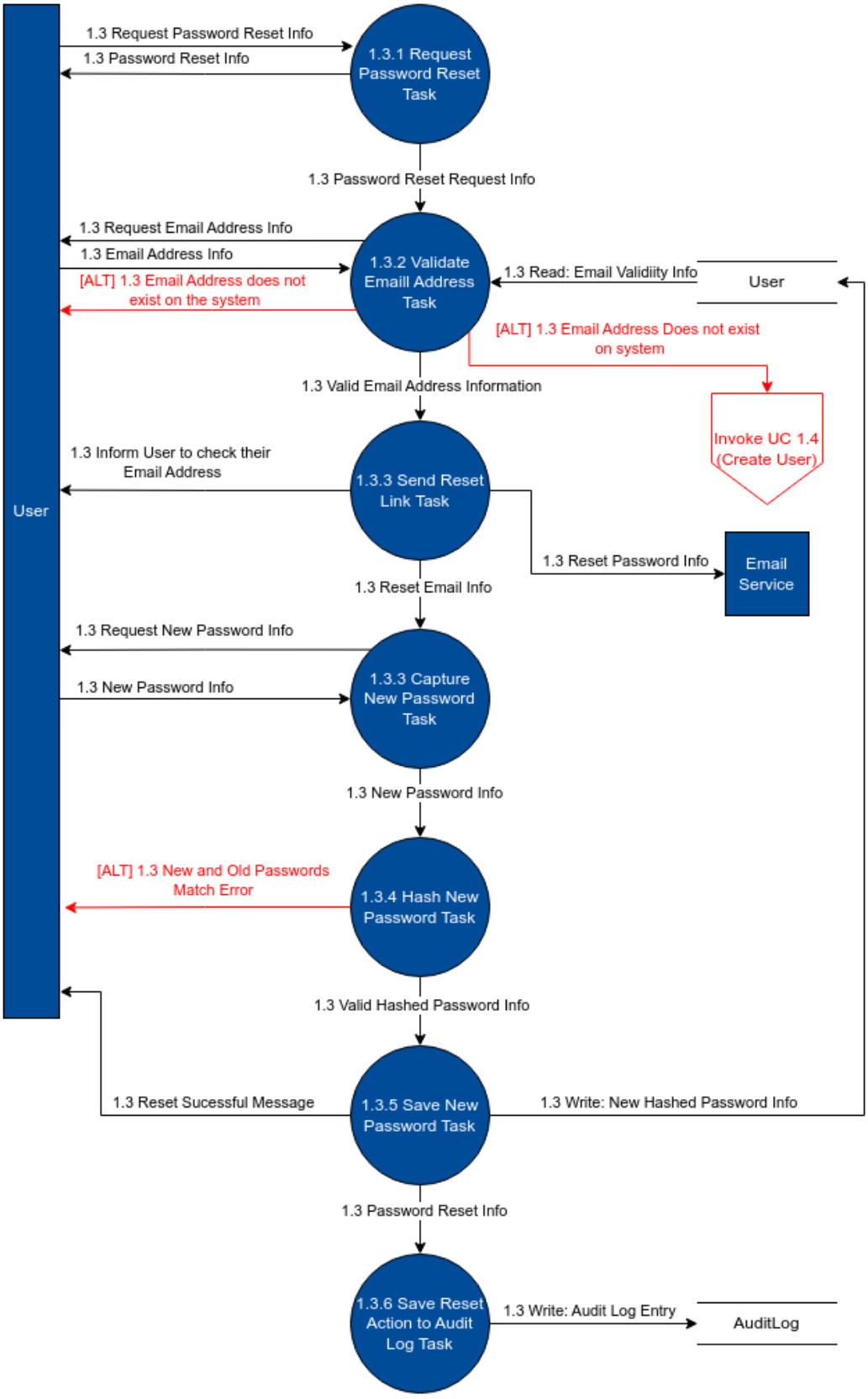


Figure 20: 1.3 Reset Password - Primitive Data Flow Diagram

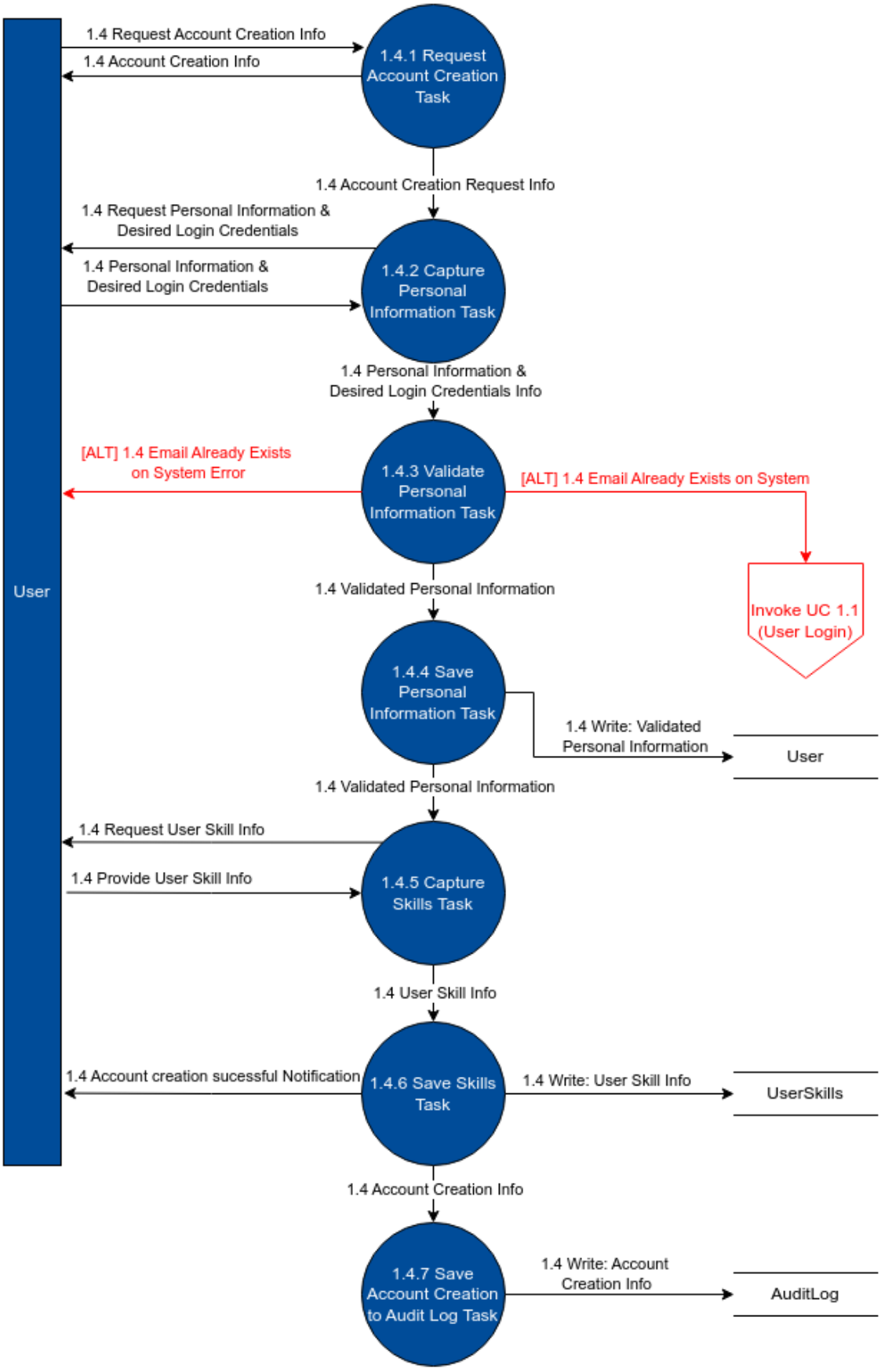


Figure 21: 1.4 Create User - Primitive Data Flow Diagram

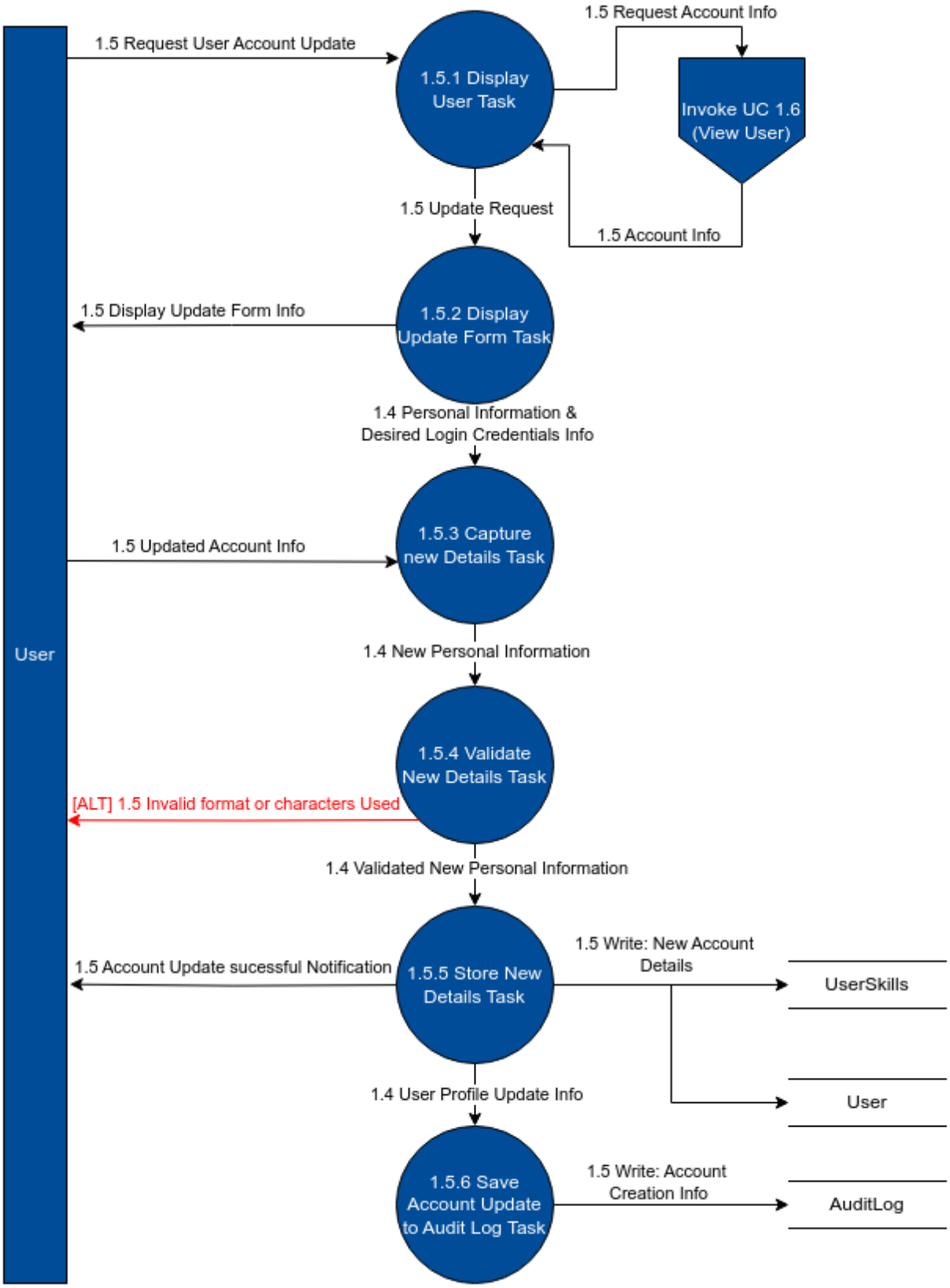


Figure 22: 1.5 Update User - Primitive Data Flow Diagram

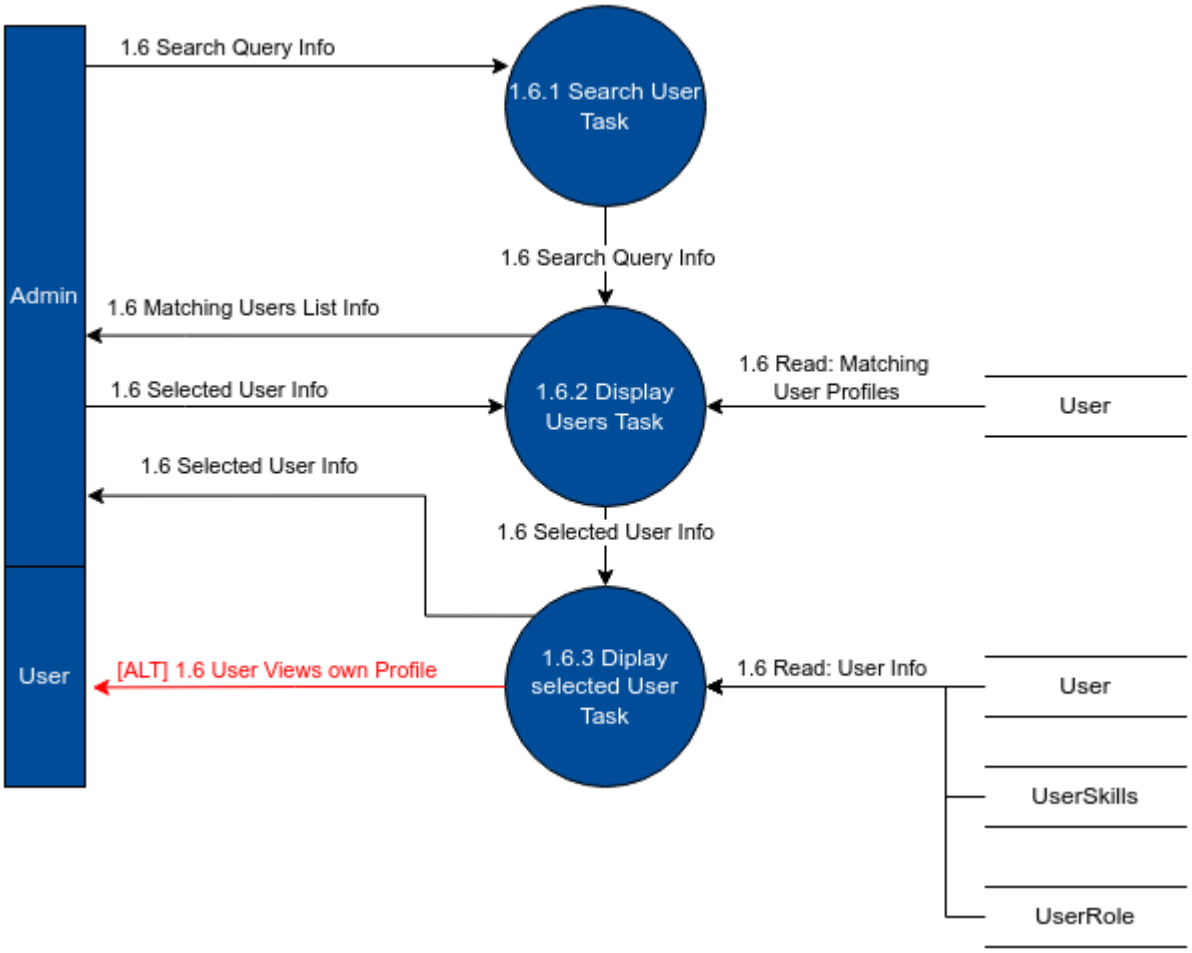


Figure 23: 1.6 View User - Primitive Data Flow Diagram

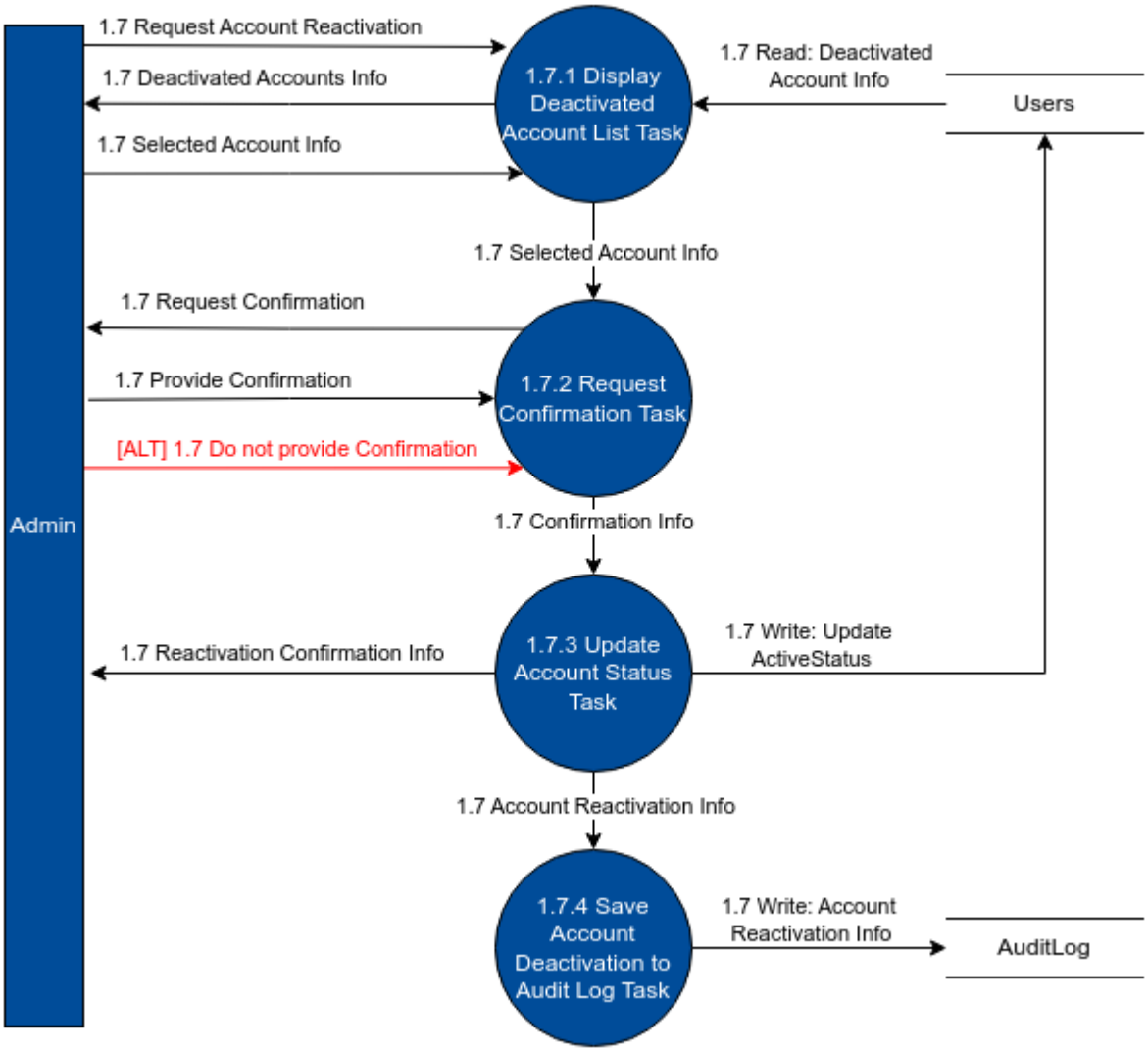


Figure 24: 1.7 Reactivate User - Primitive Data Flow Diagram

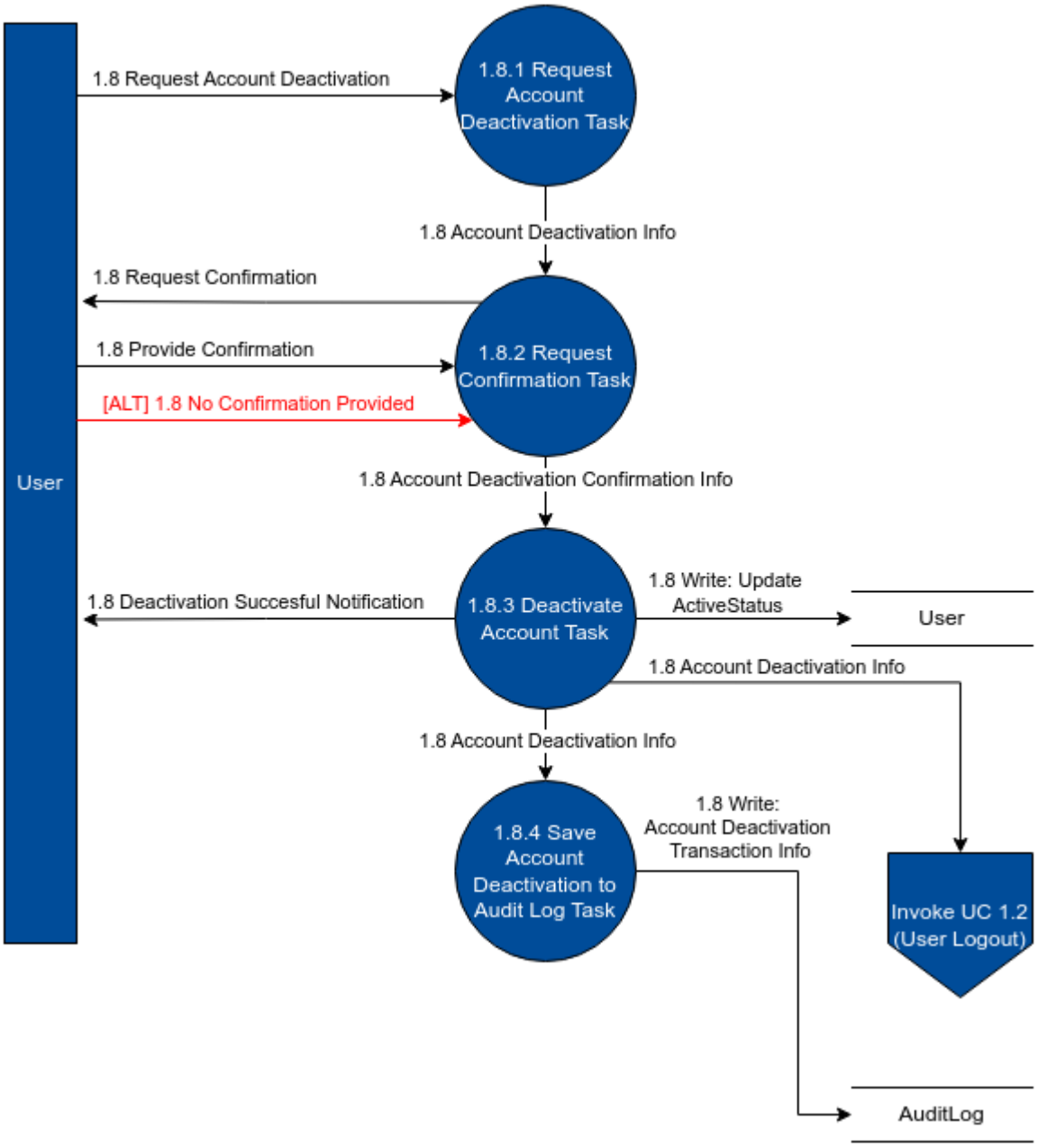


Figure 25: 1.8 Deactivate User - Primitive Data Flow Diagram

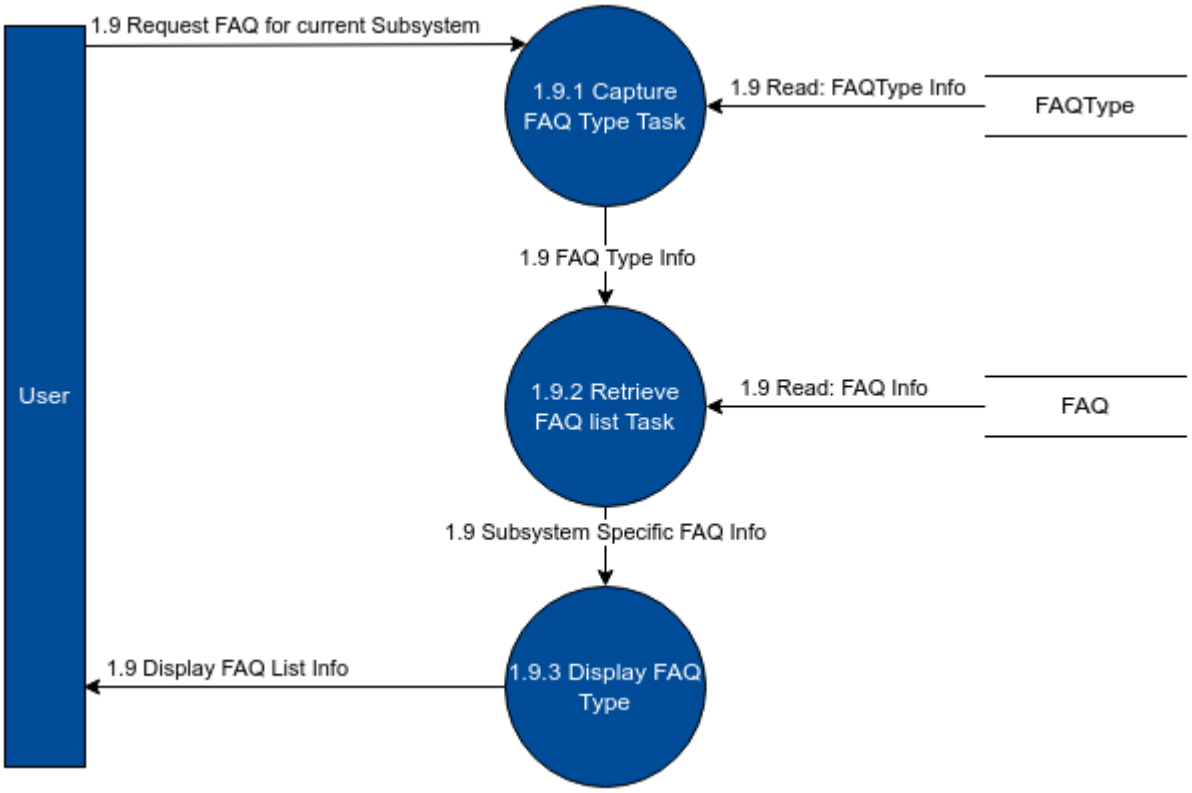


Figure 26: 1.9 View FAQ - Primitive Data Flow Diagram

2.9.2 Admin Subsystem Primitives

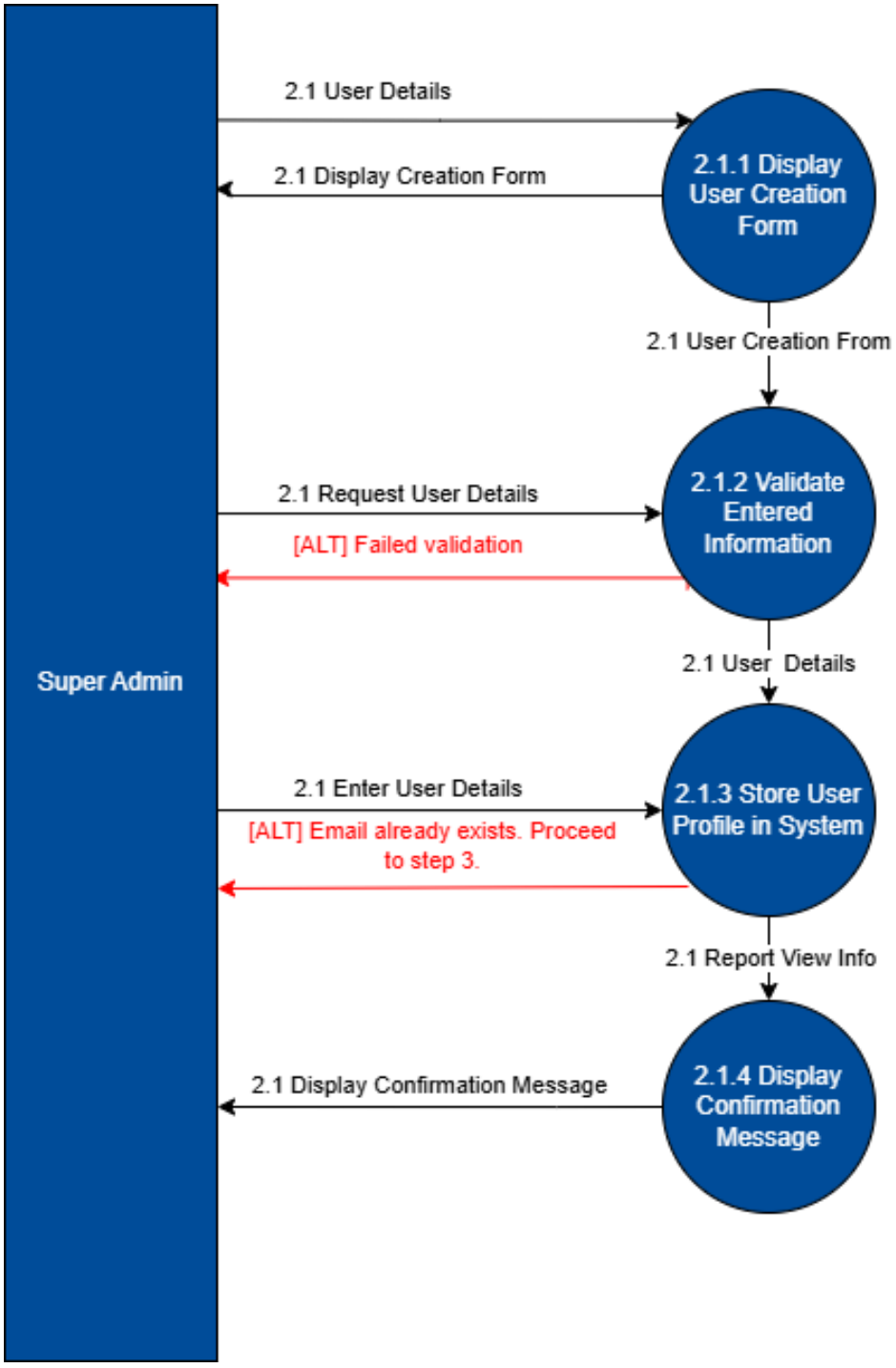


Figure 27: 2.1 Create User Profile - Primitive Data Flow Diagram

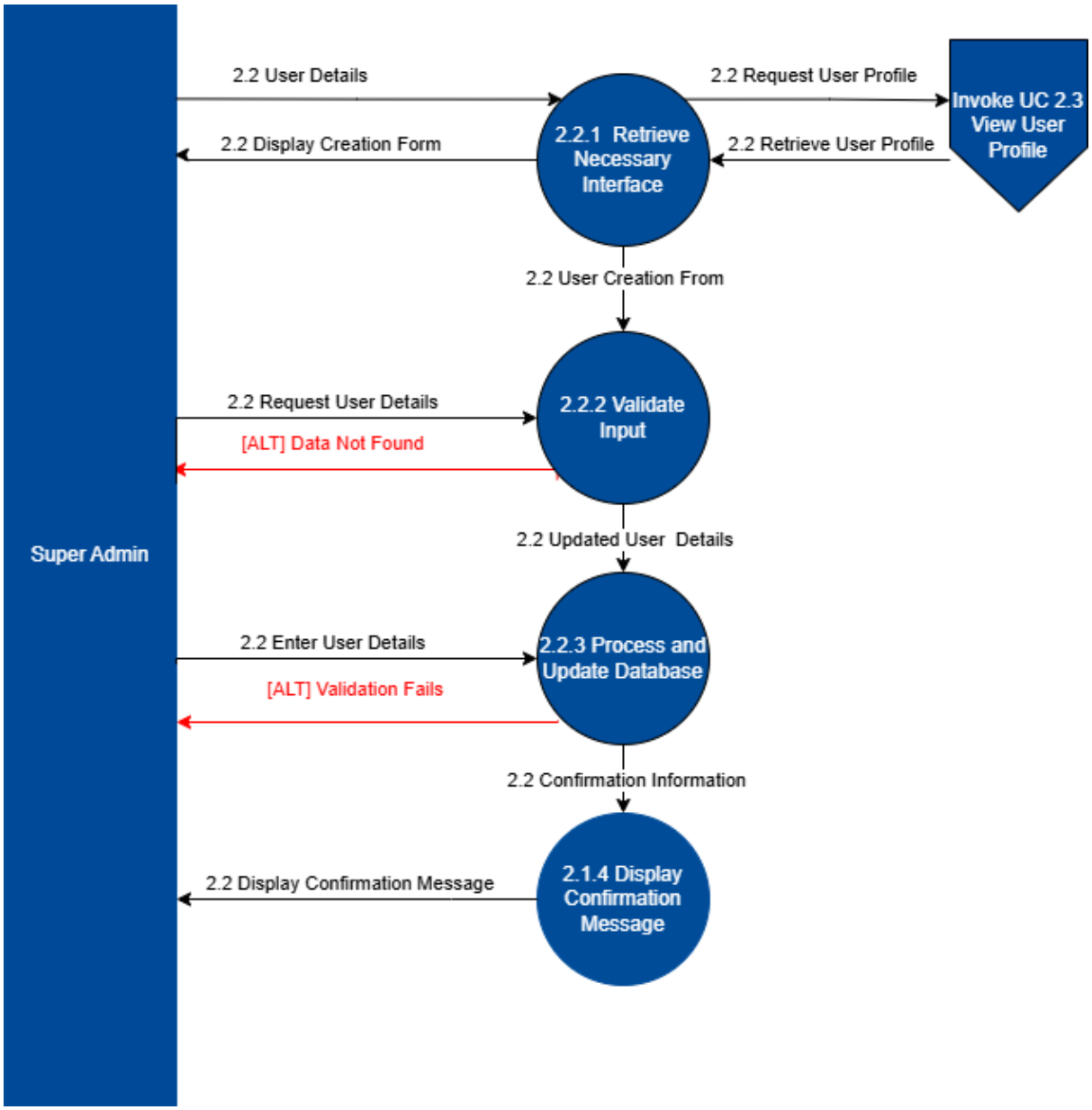


Figure 28: 2.2 Update User Primitive Data Flow Diagram

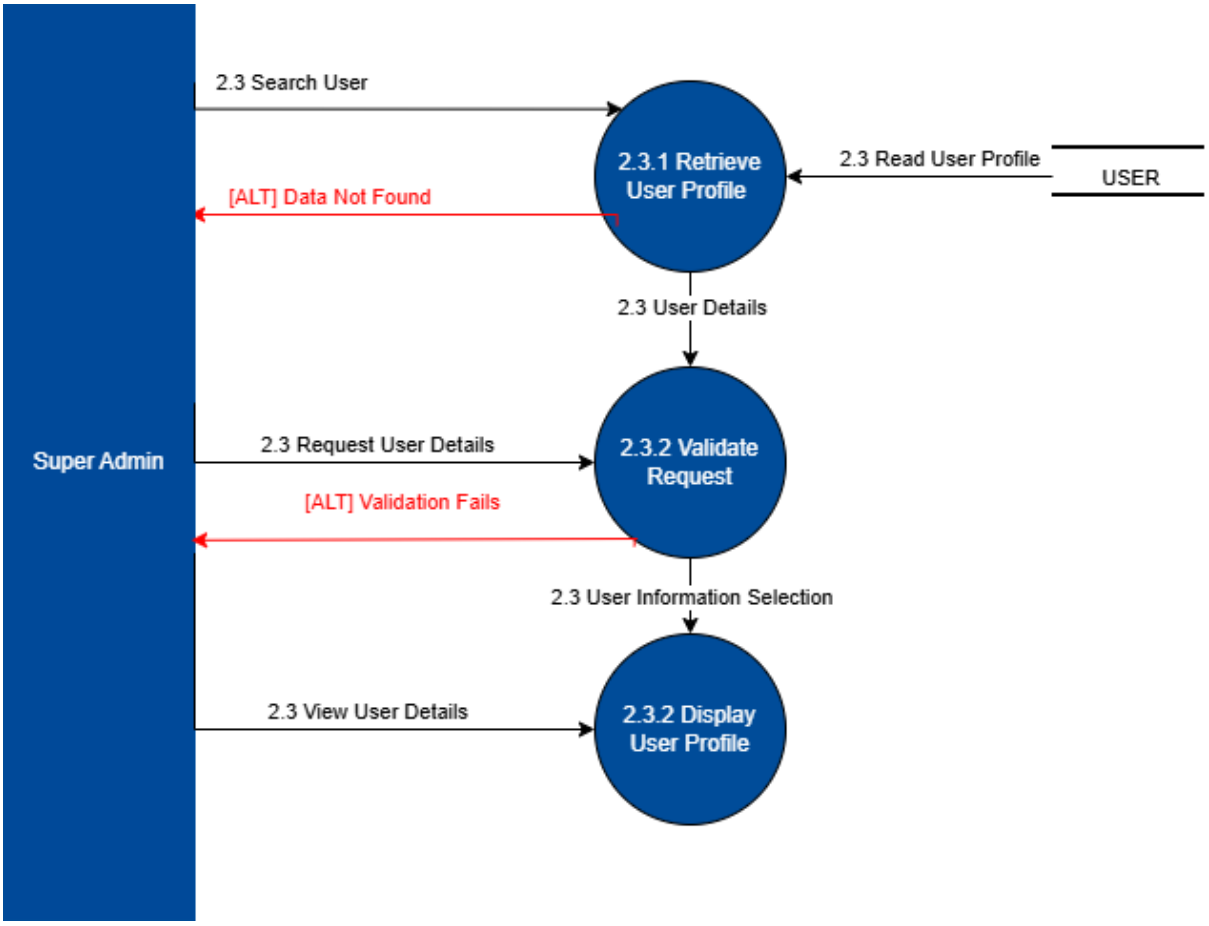


Figure 29 2.3 View User Profile - Primitive Data Flow Diagram

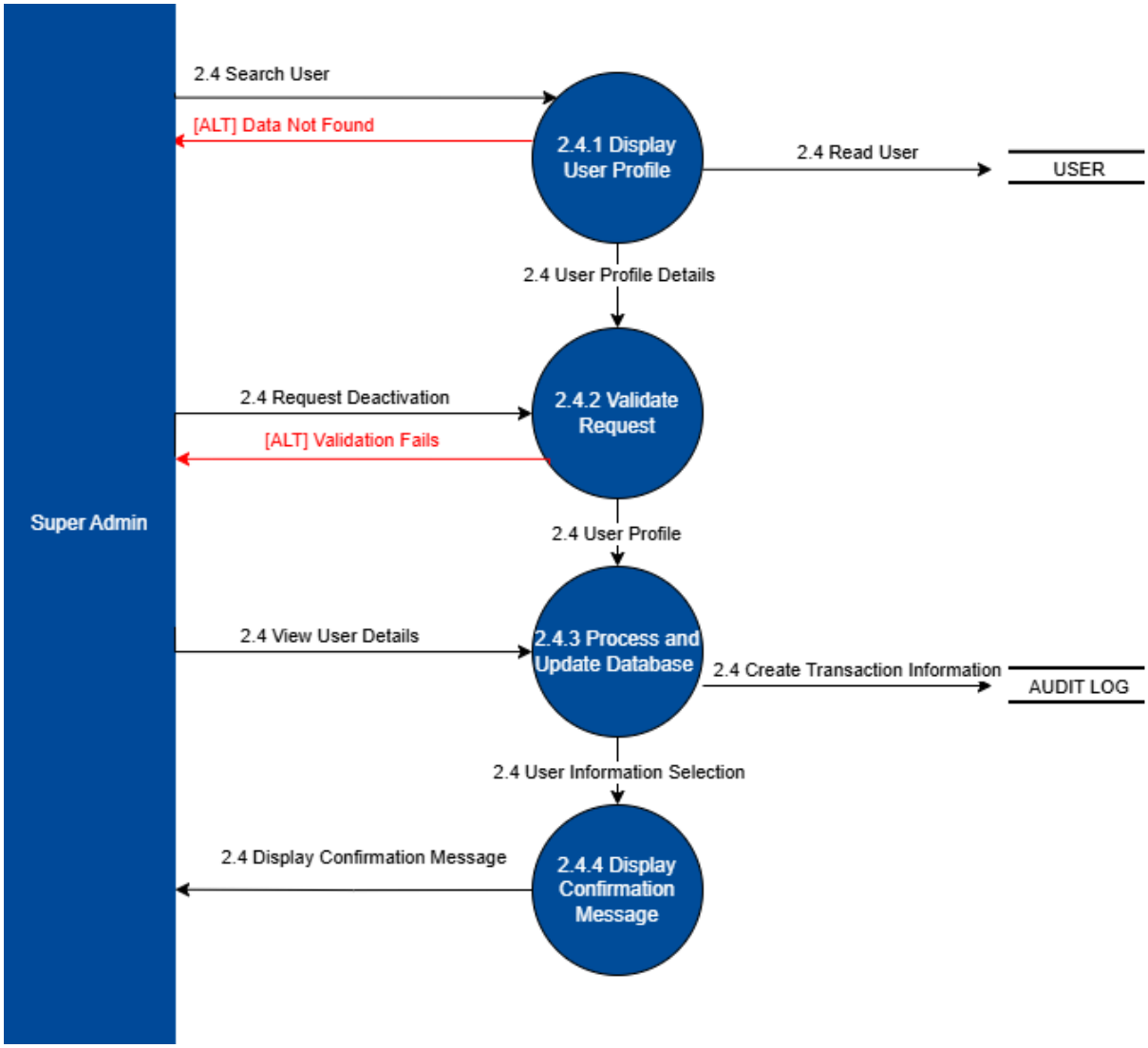


Figure 30 2.4 Search User Profile - Primitive Data Flow Diagram

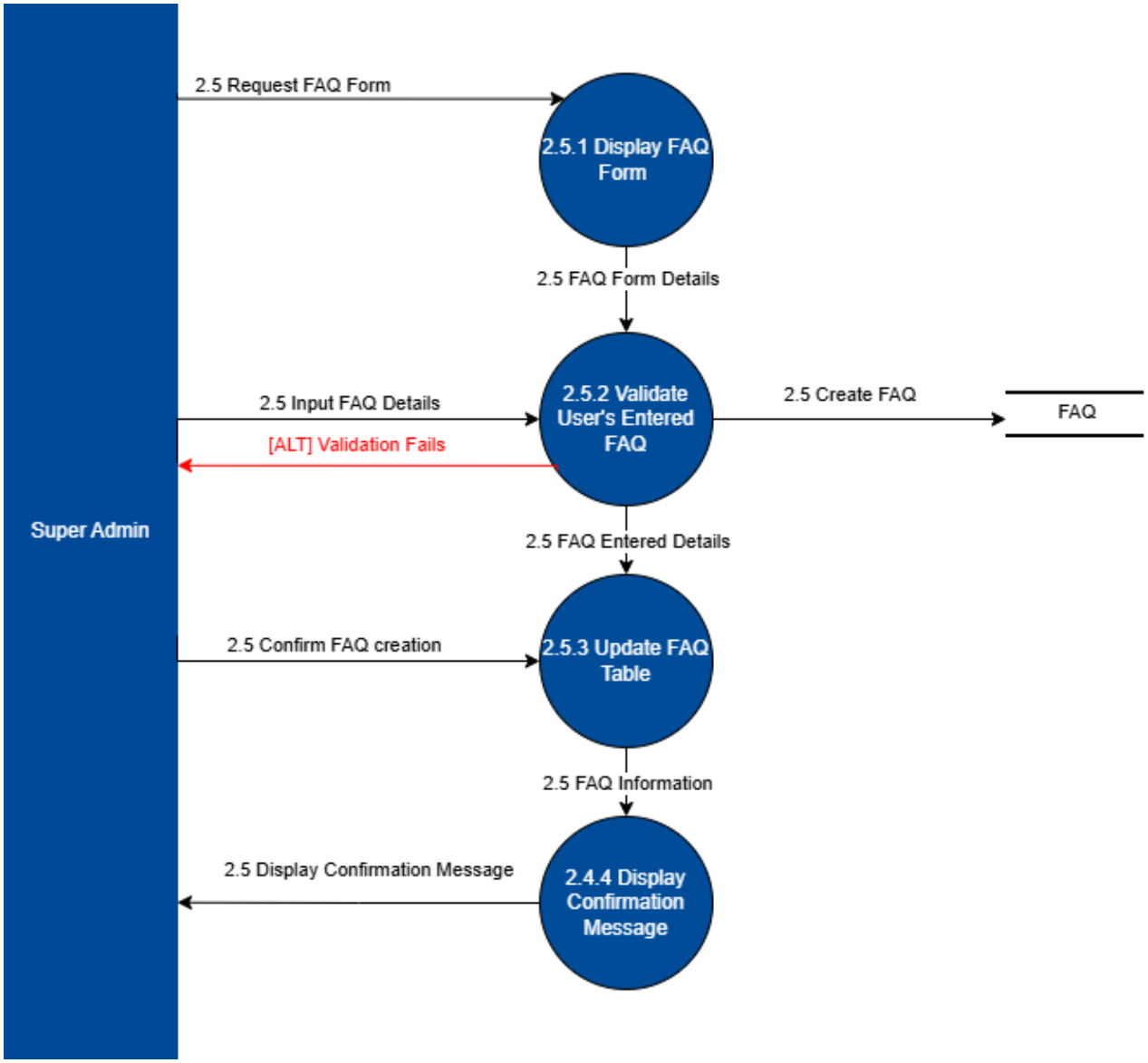


Figure 31 2.5 Add FAQ - Primitive Data Flow Diagram

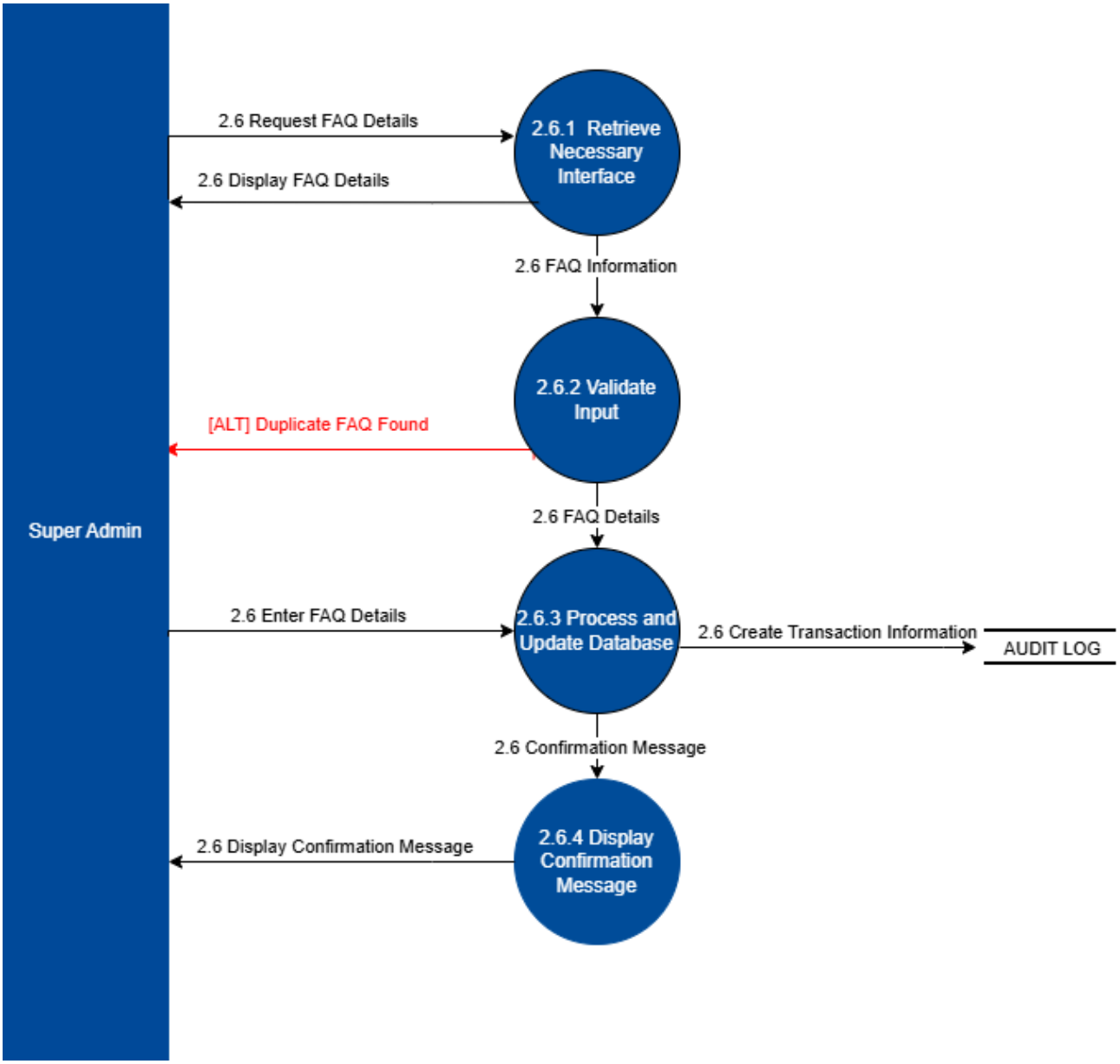


Figure 32 2.6 Update FAQ - Primitive Data Flow Diagram

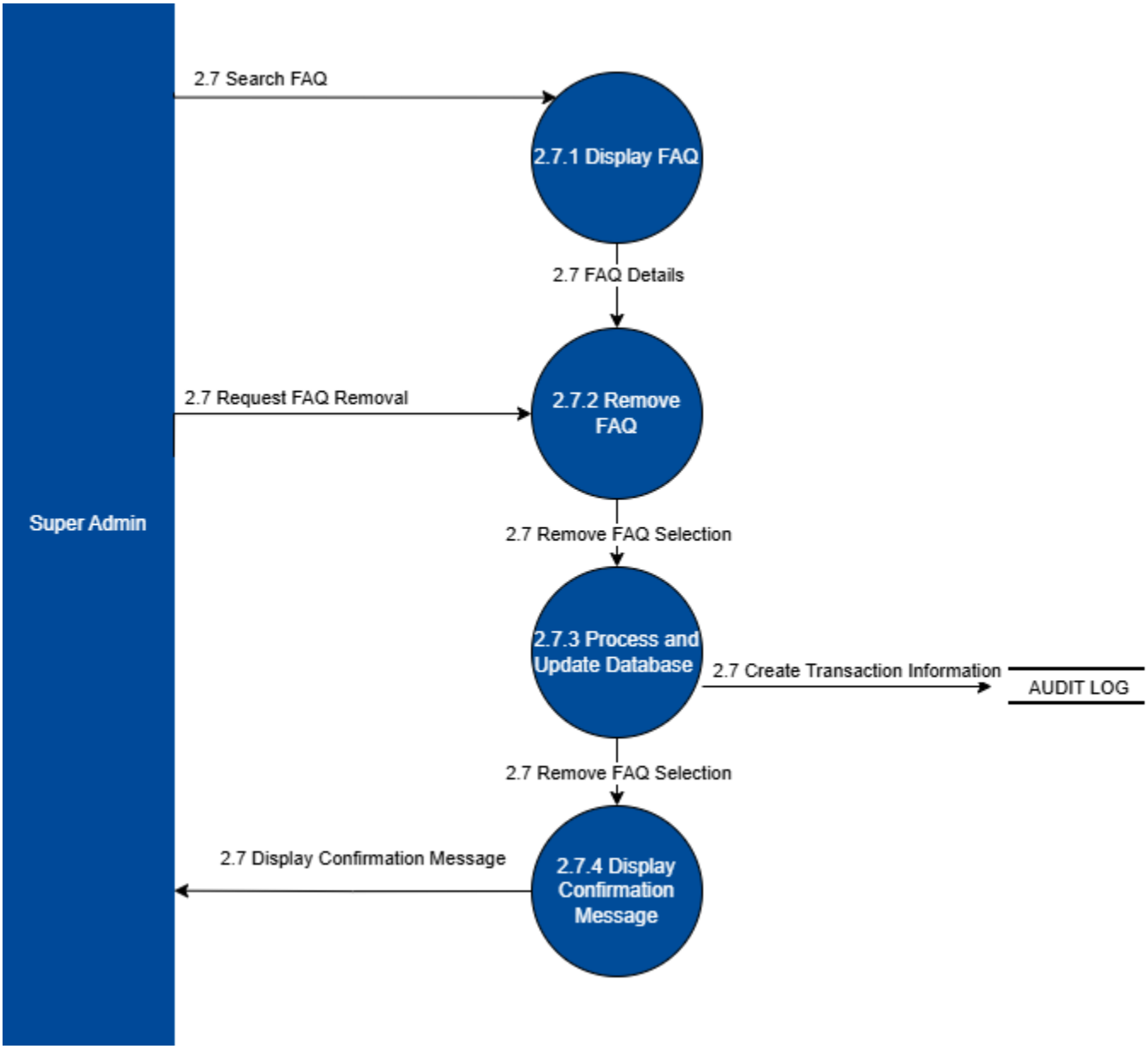


Figure 33 2.7 Remove FAQ - Primitive Data Flow Diagram

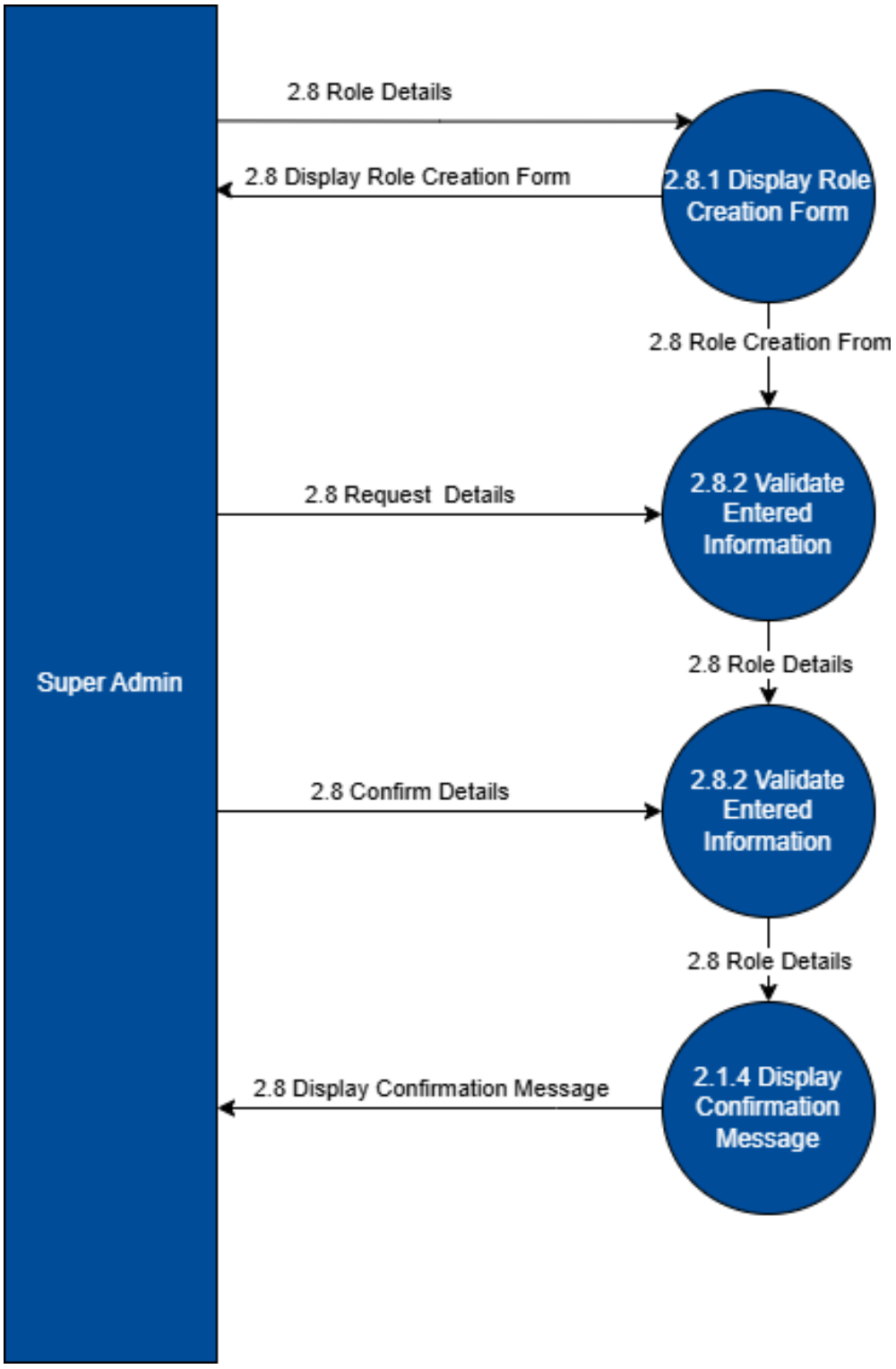


Figure 34 2.8 Create Role - Primitive Data Flow Diagram

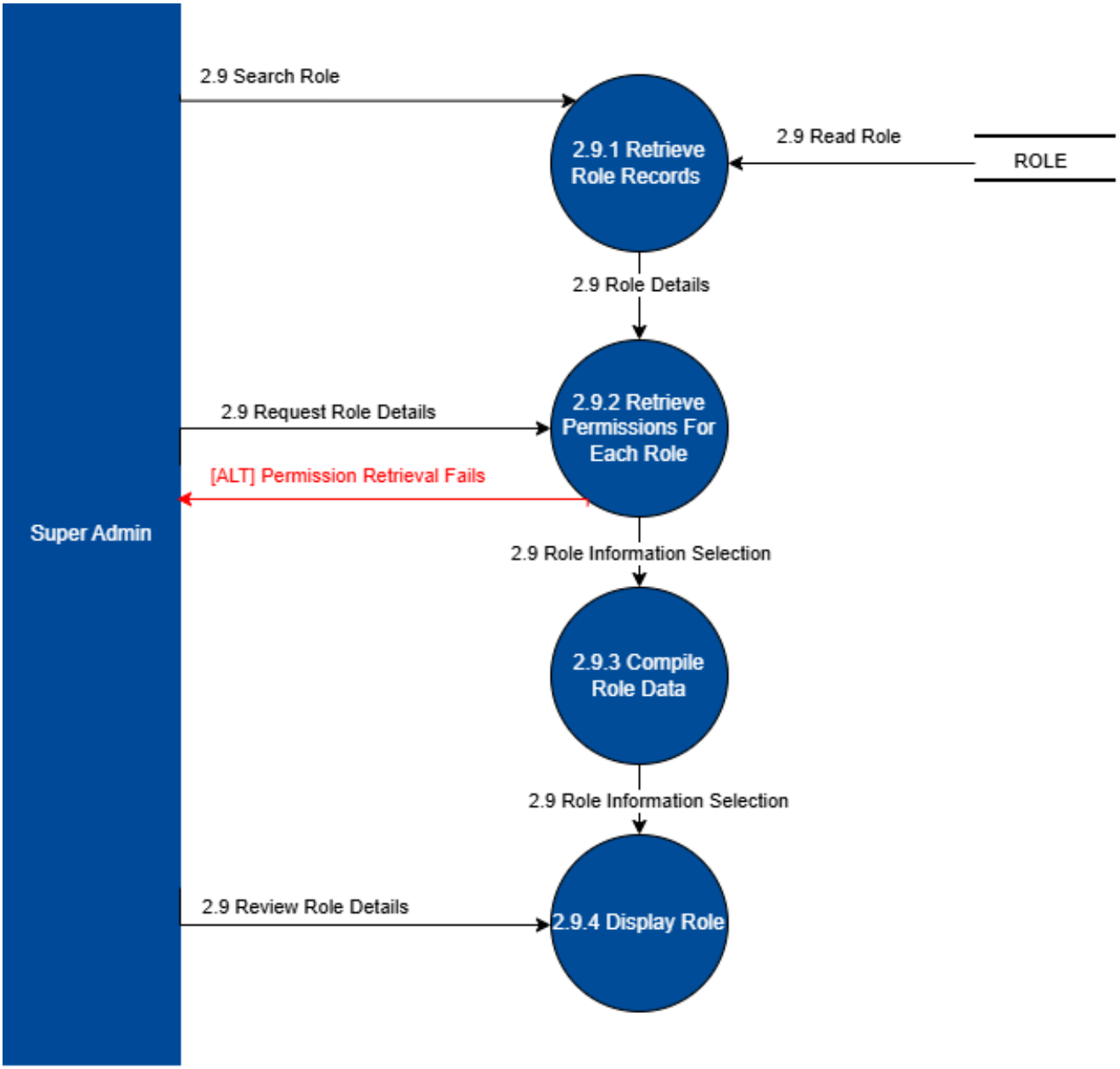


Figure 35 2.9 View Role - Primitive Data Flow Diagram

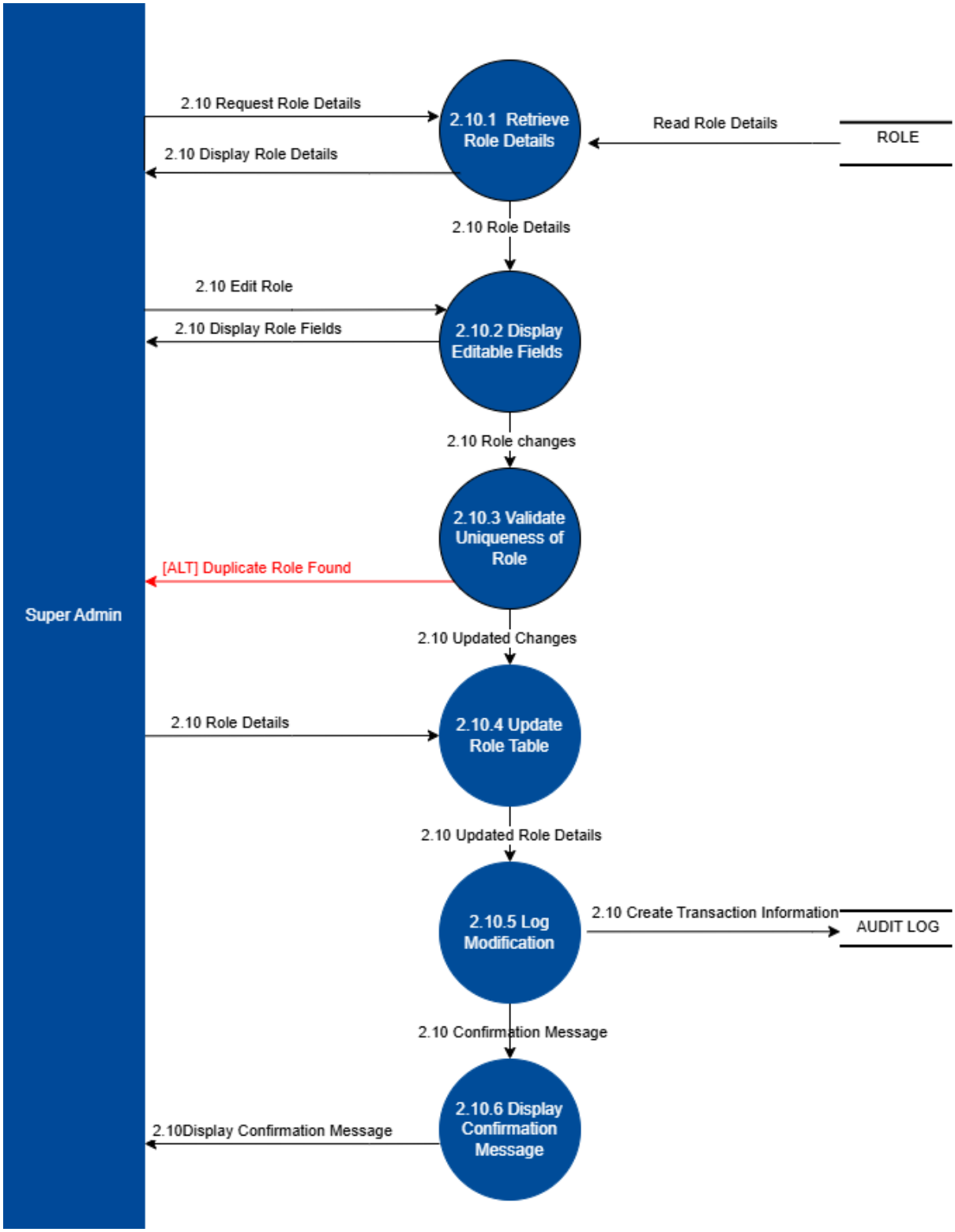


Figure 36 2.10 Update Role - Primitive Data Flow Diagram

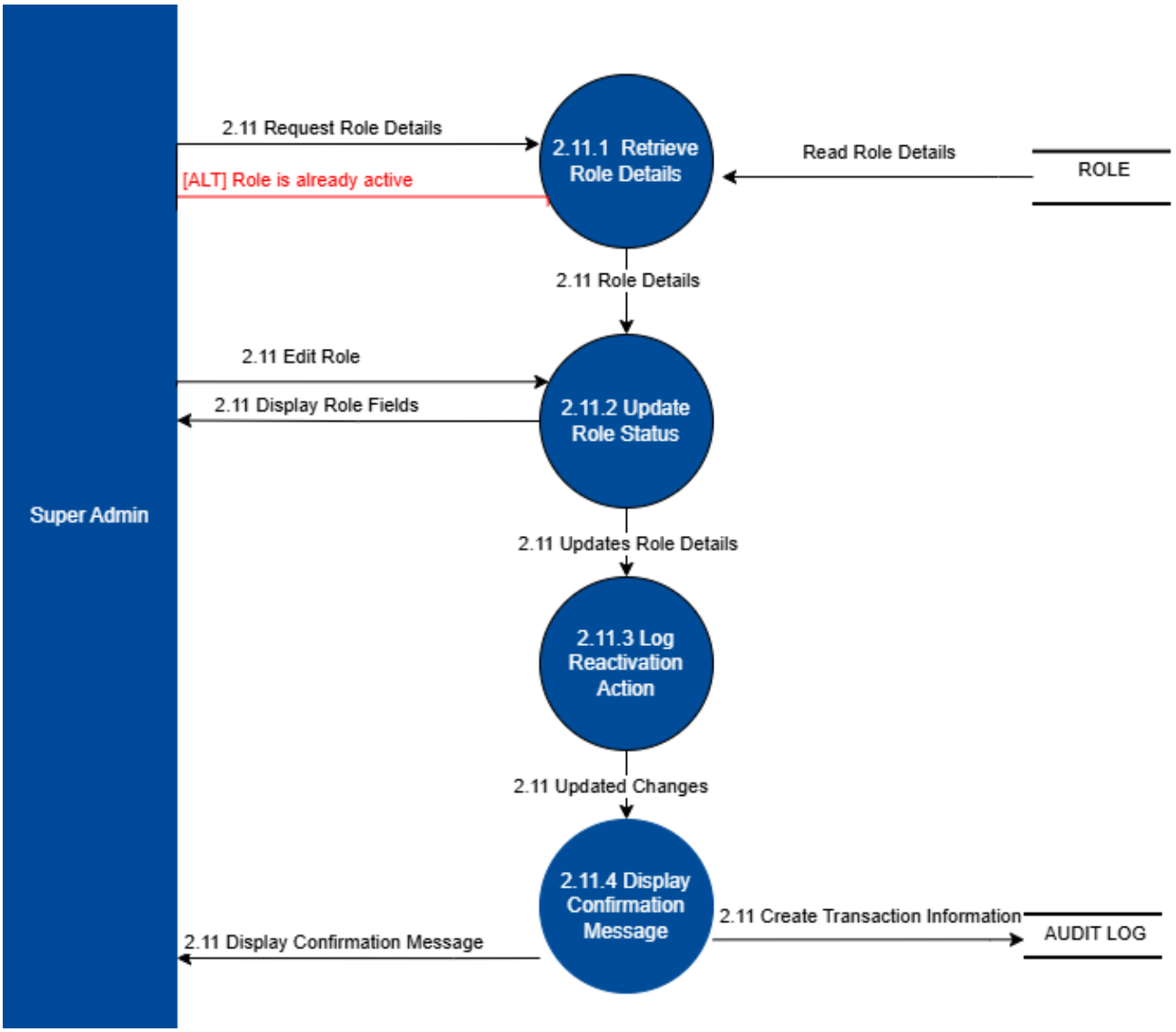


Figure 37 2.11 Reactivate Role - Primitive Data Flow Diagram

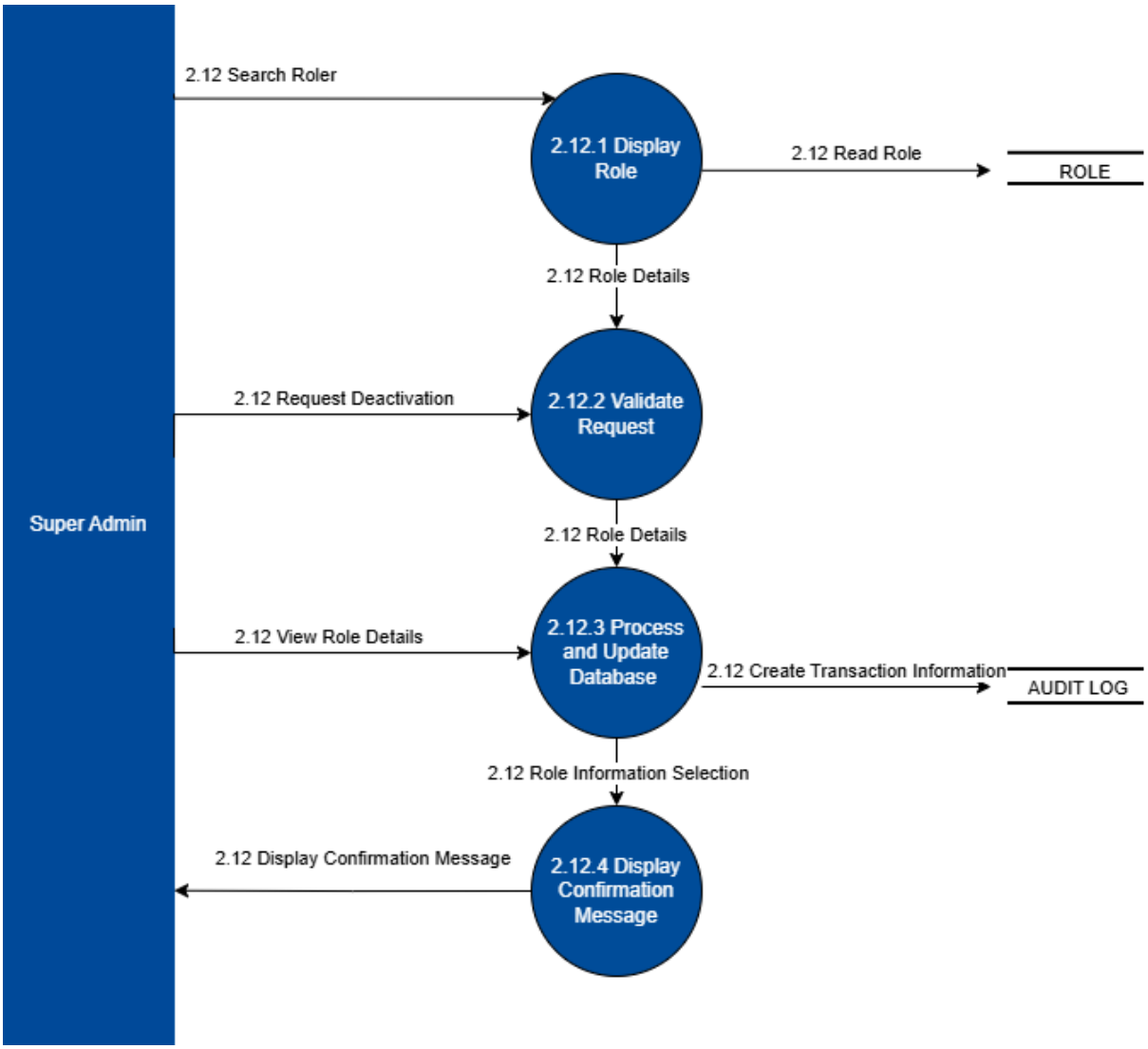


Figure 38 2.12 Deactivate Role - Primitive Data Flow Diagram

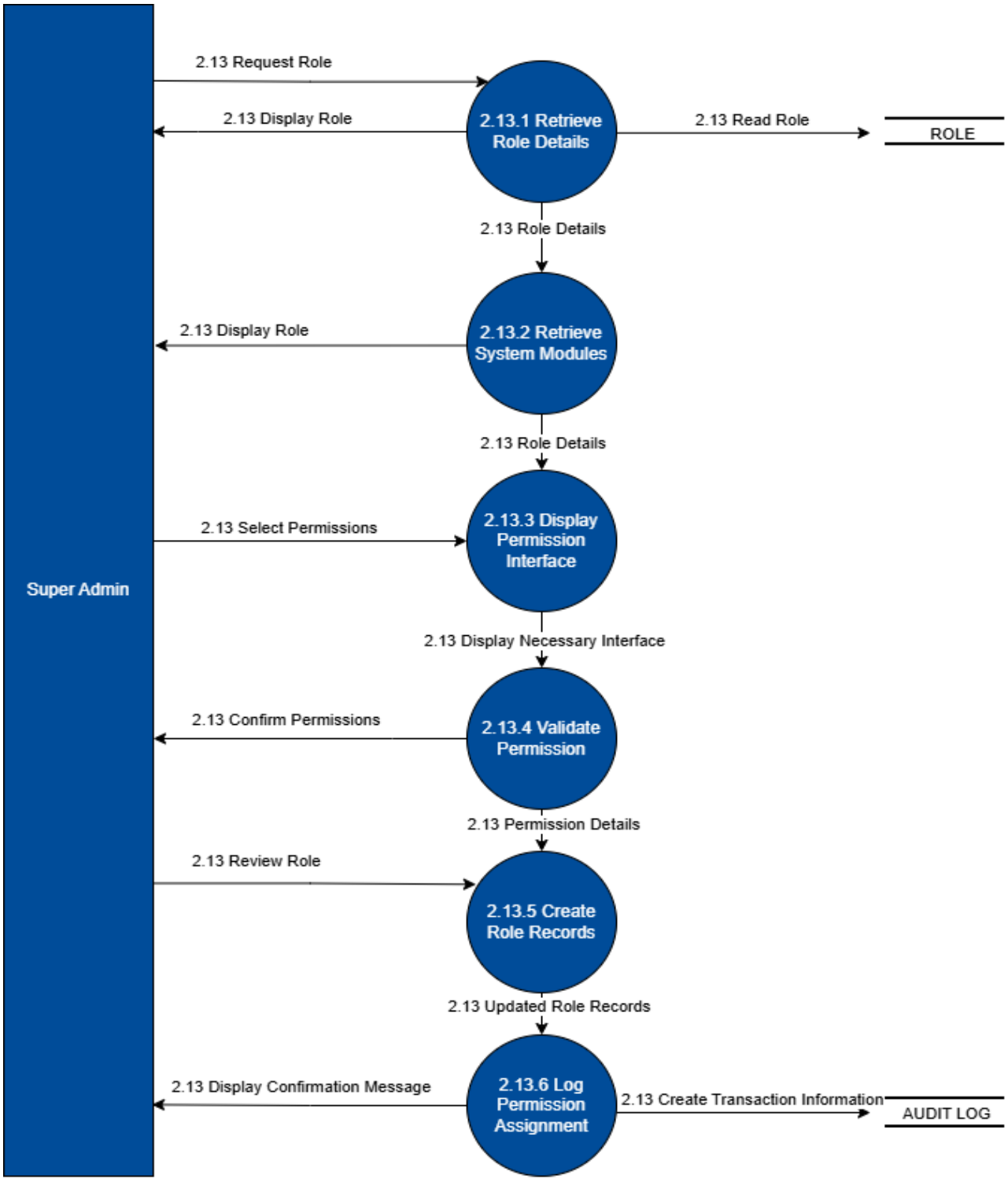


Figure 39 2.13 Create Role Permission - Primitive Data Flow Diagram

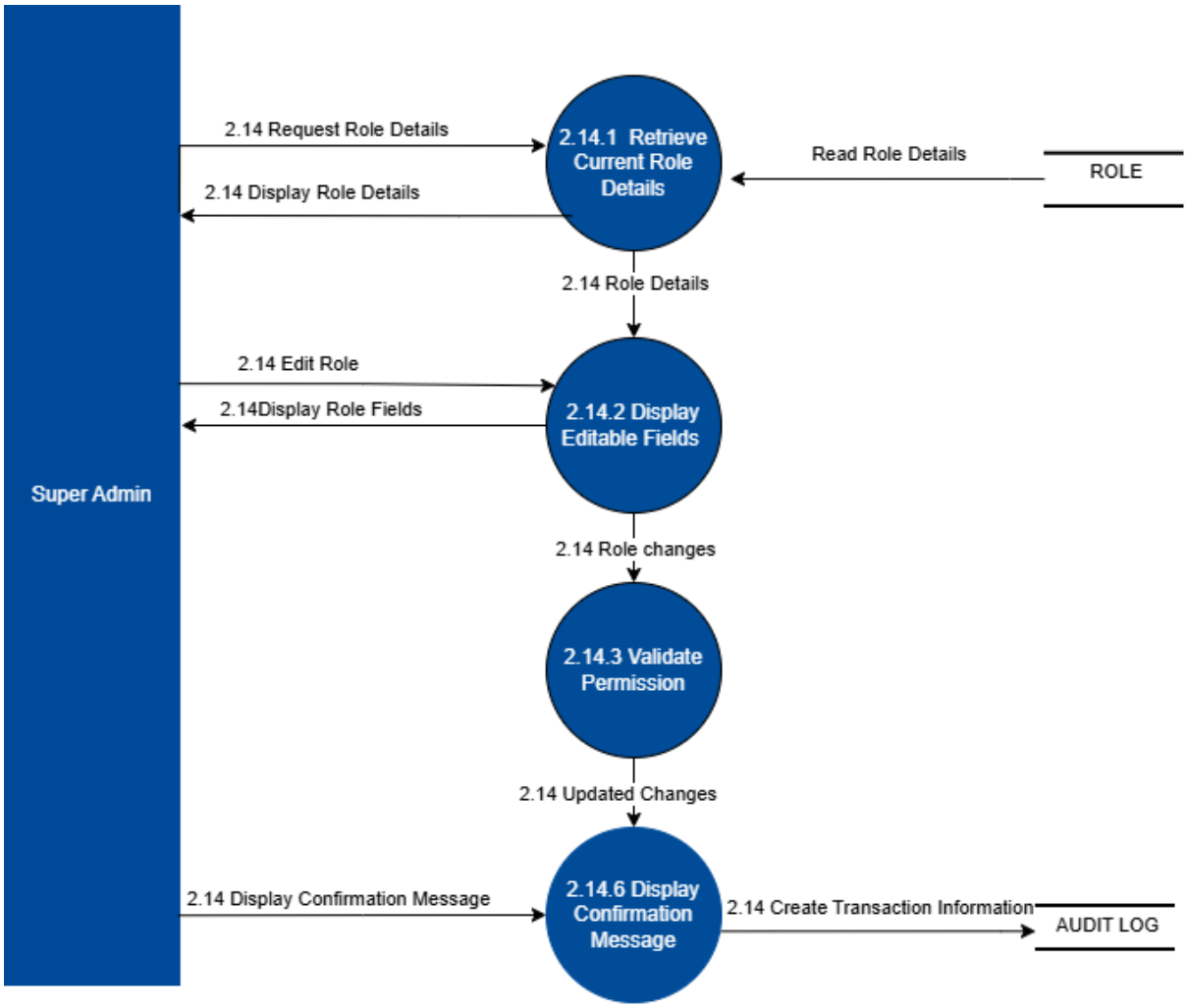


Figure 40 2.14 Update Role Permission - Primitive Data Flow Diagram

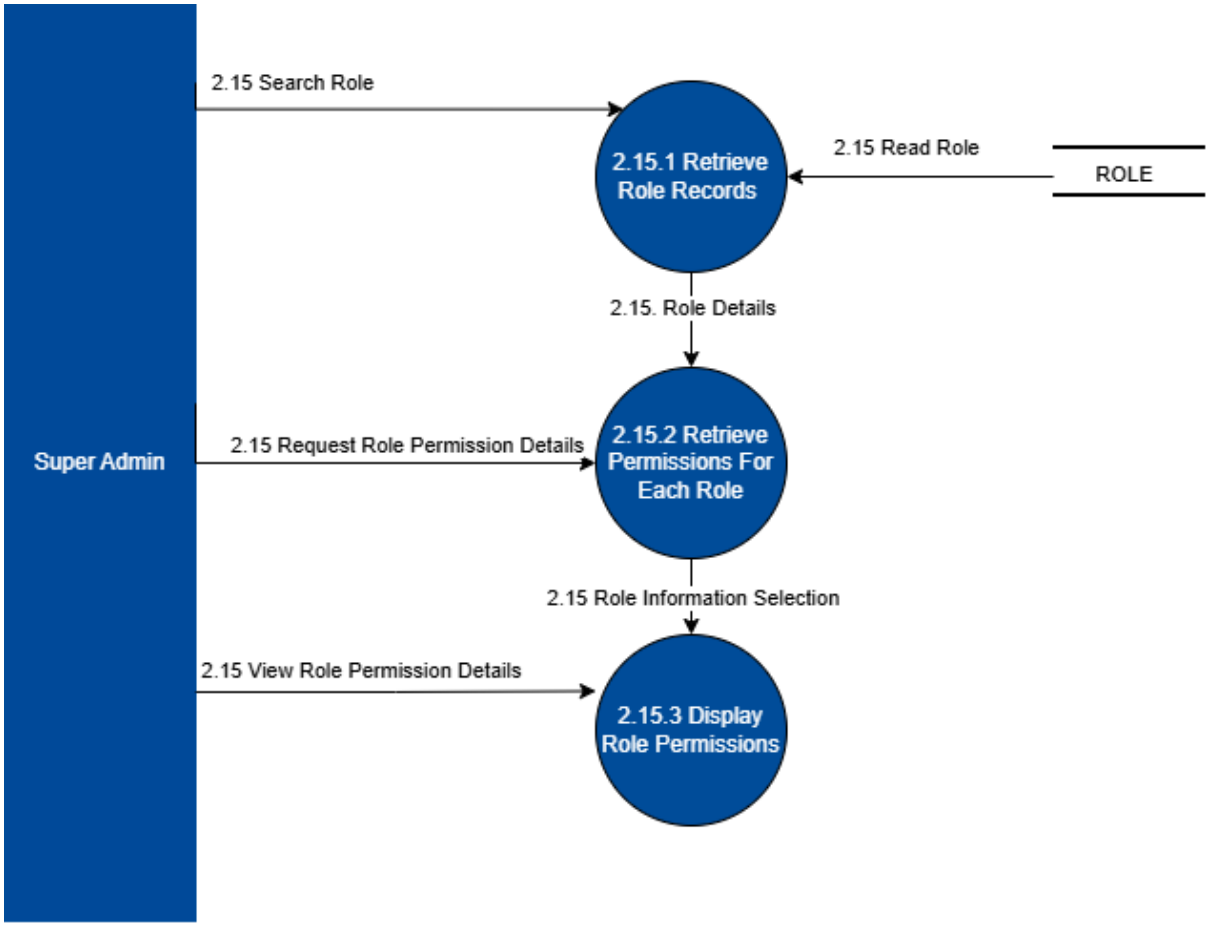


Figure 41 2.15 View Role Permission - Primitive Data Flow Diagram

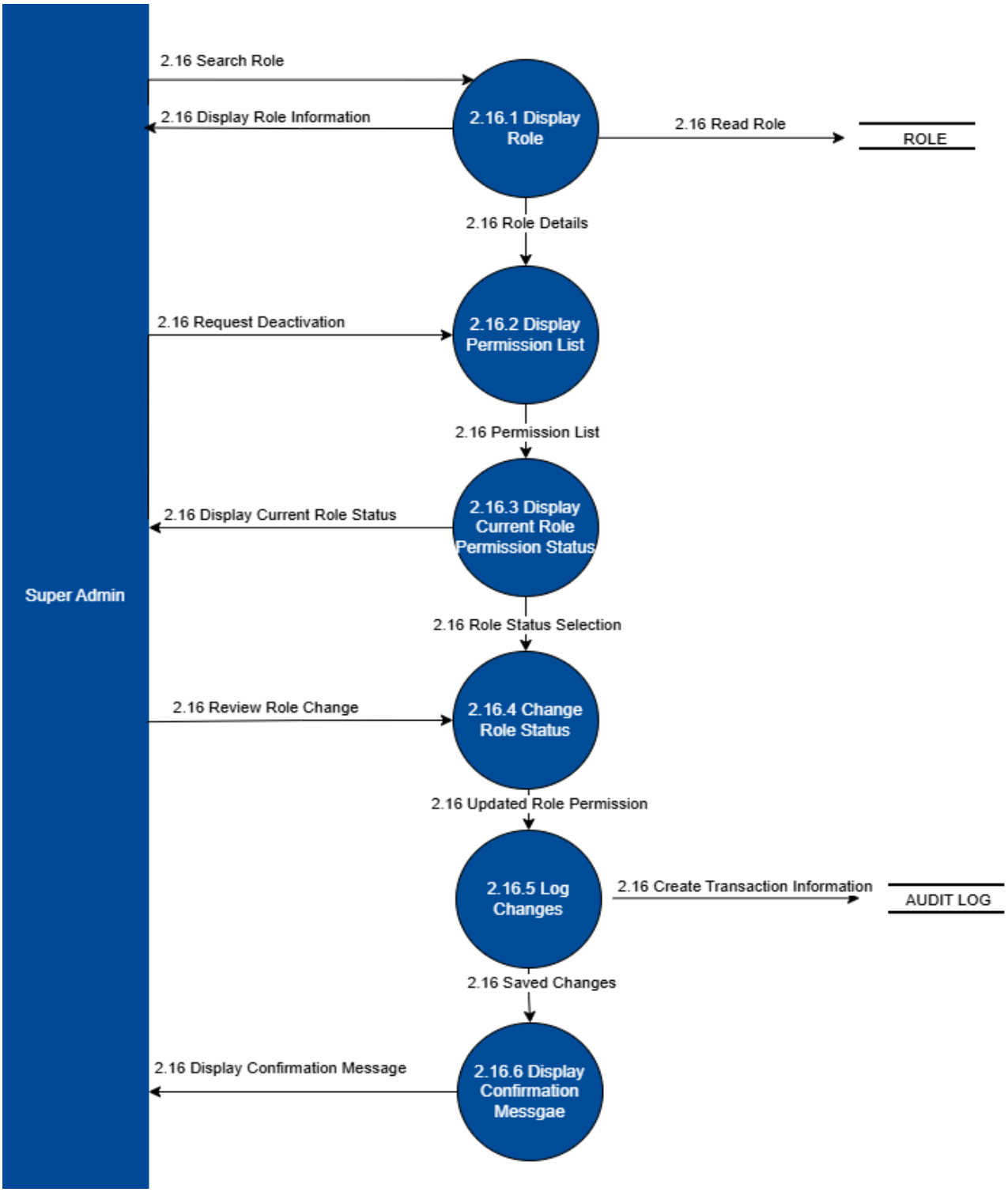


Figure 42 2.16 Deactivate Role Permission - Primitive Data Flow Diagram

2.9.3 Inventory Subsystem Primitives

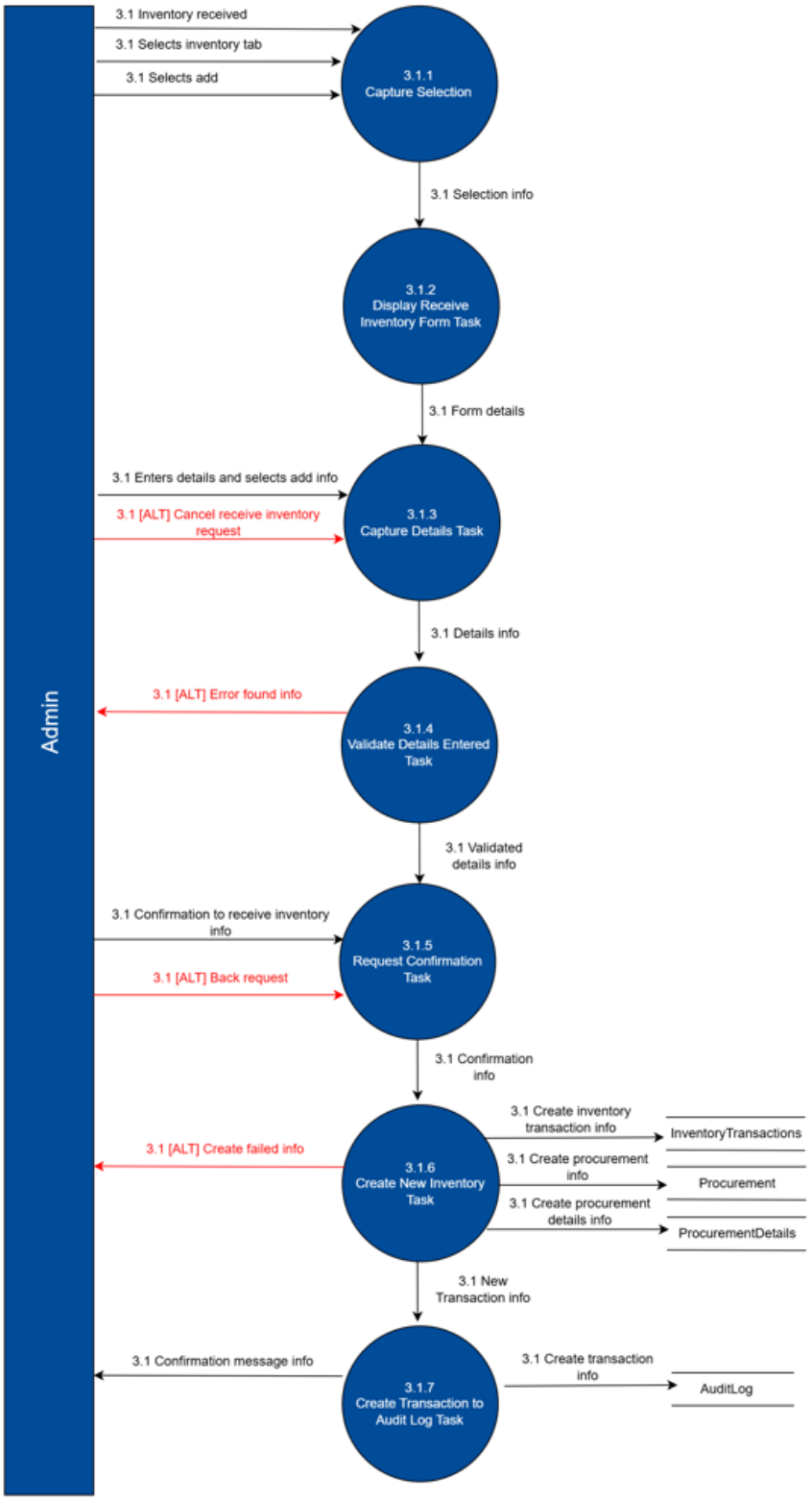


Figure 43 3.1 Receive Inventory Item - Primitive Data Flow Diagram

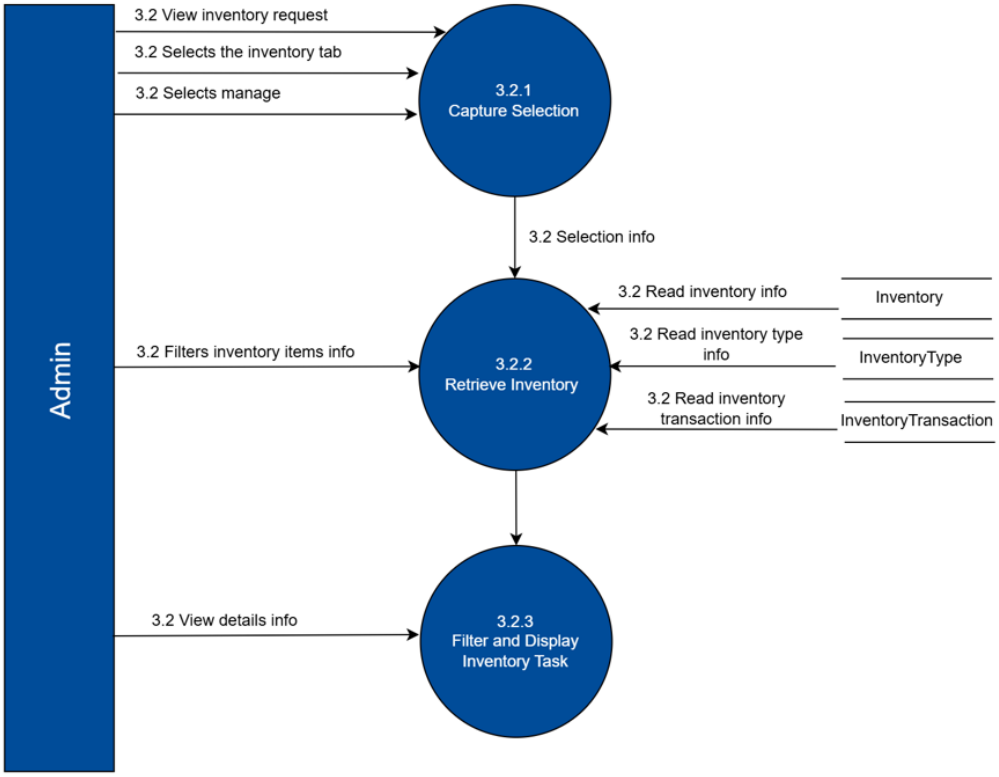


Figure 44 3.2 View Inventory - Primitive Data Flow Diagram

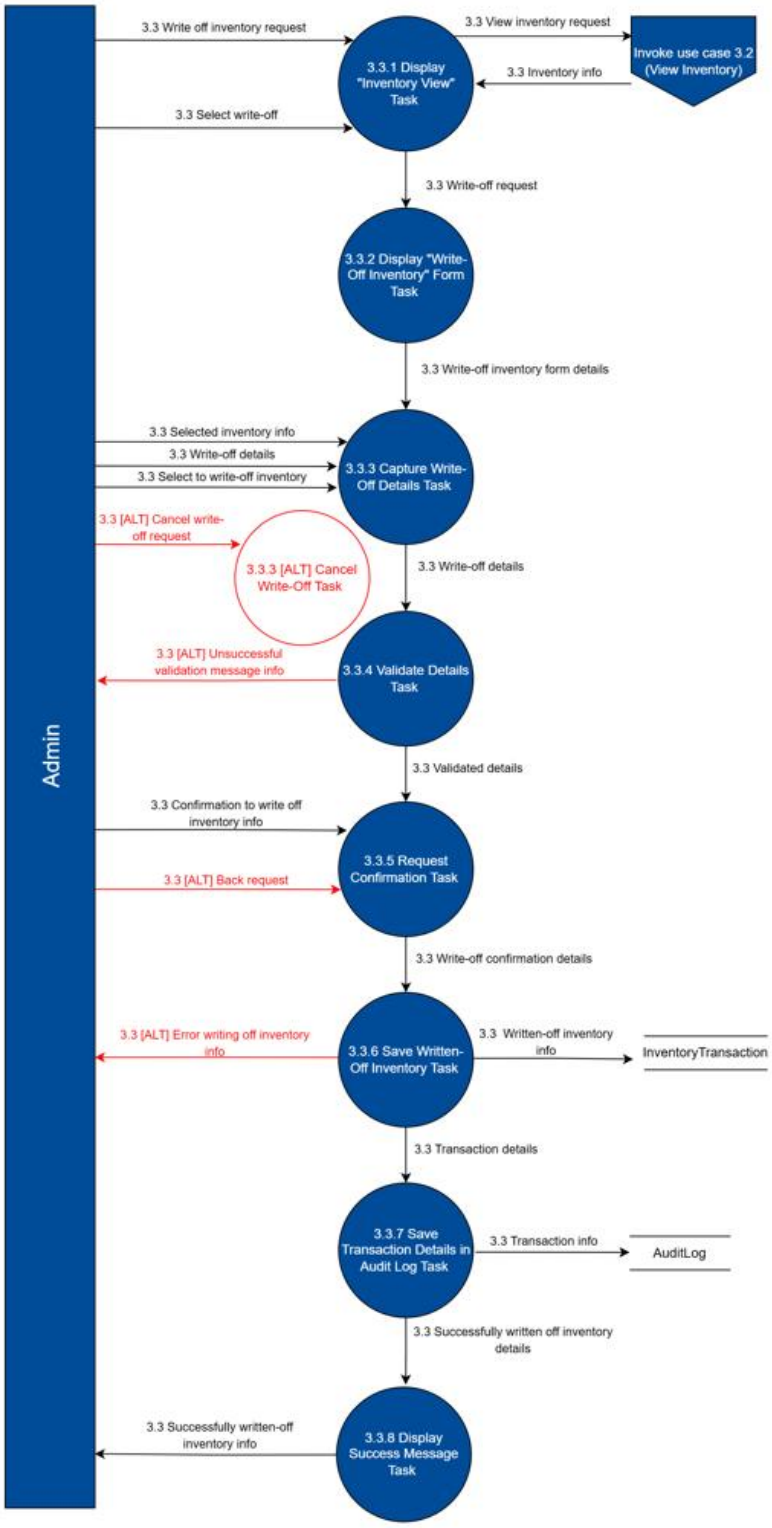


Figure 45 3.3 Write-Off Inventory - Primitive Data Flow Diagram

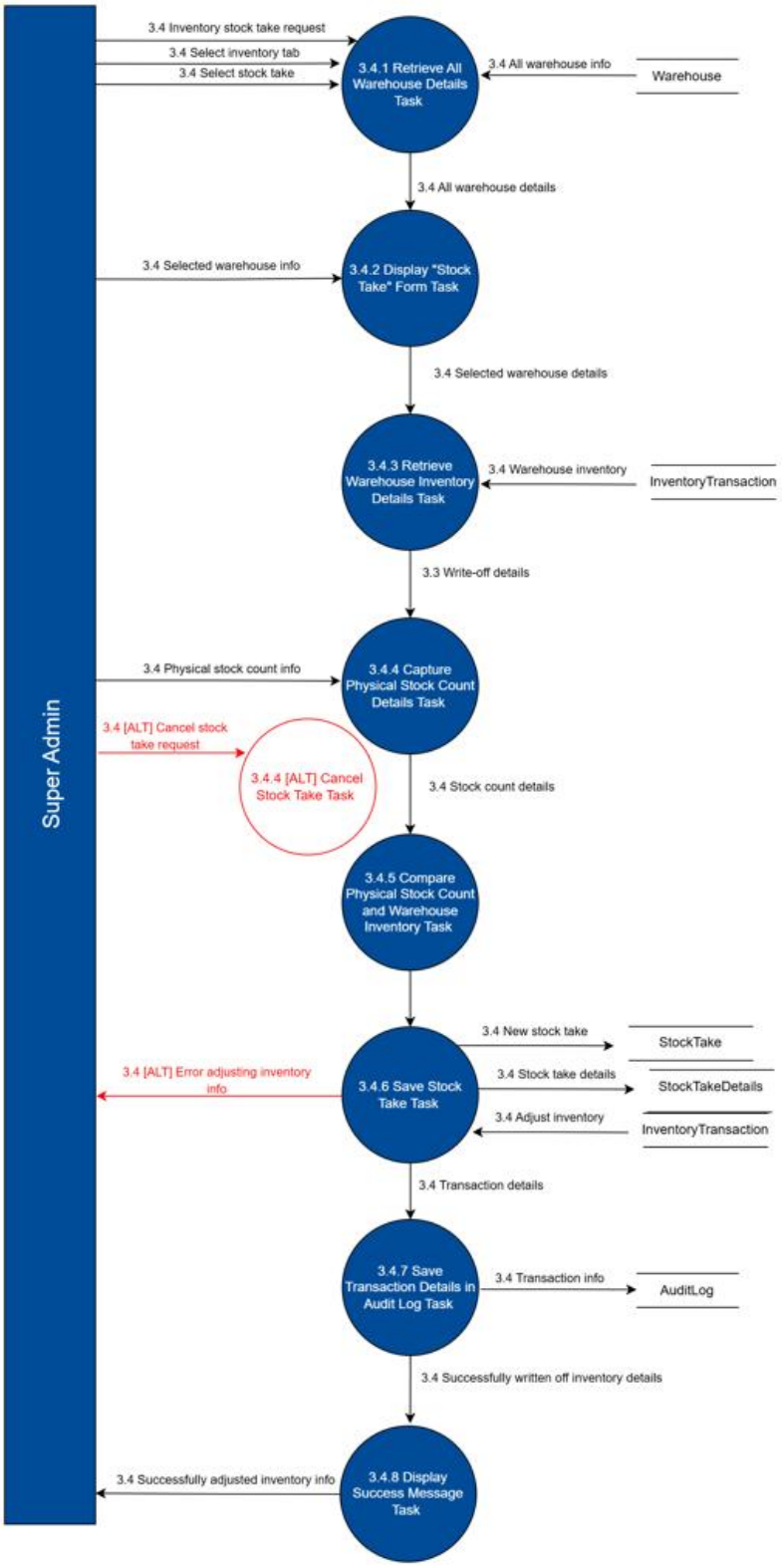


Figure 46 3.4 Inventory Stock Take - Primitive Data Flow Diagram

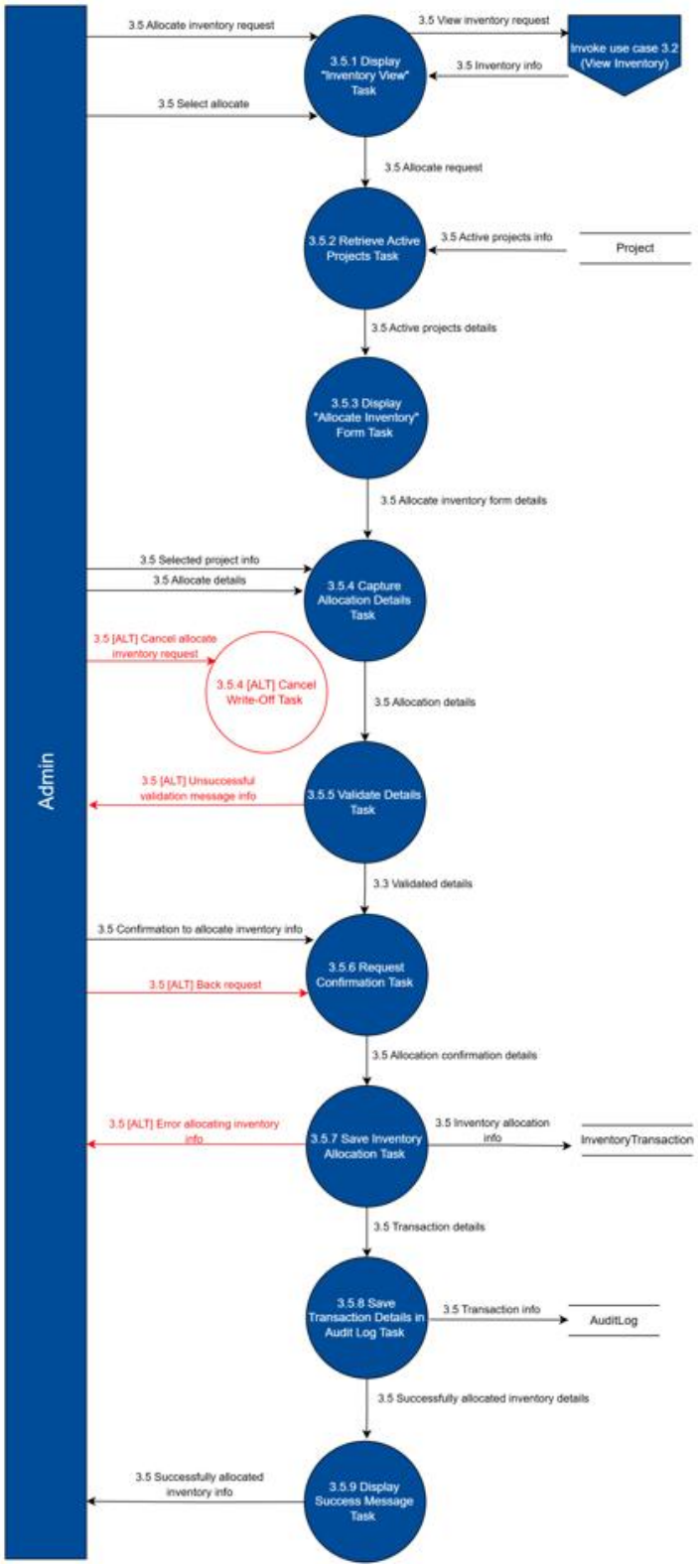


Figure 47 3.5 Allocate Inventory - Primitive Data Flow Diagram

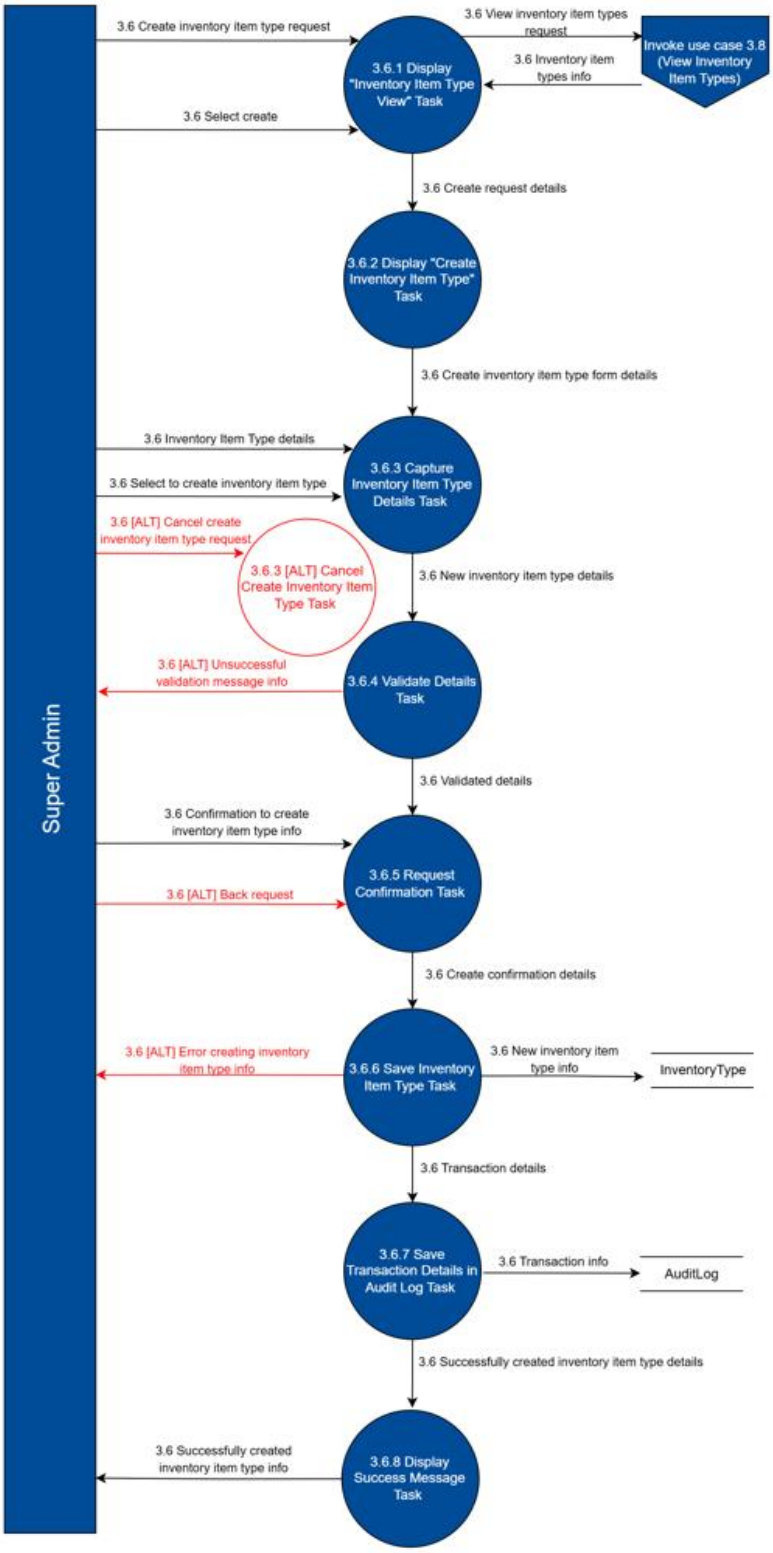


Figure 48 3.6 Create Inventory Item Type - Primitive Data Flow Diagram

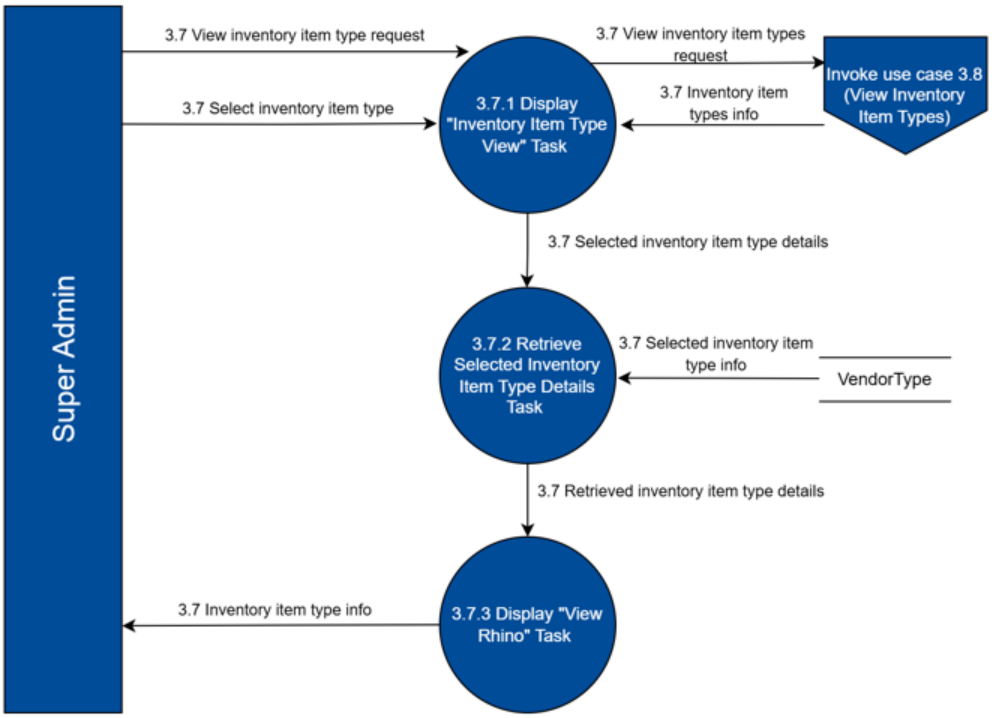


Figure 49 3.7 View Inventory Item Type - Primitive Data Flow Diagram

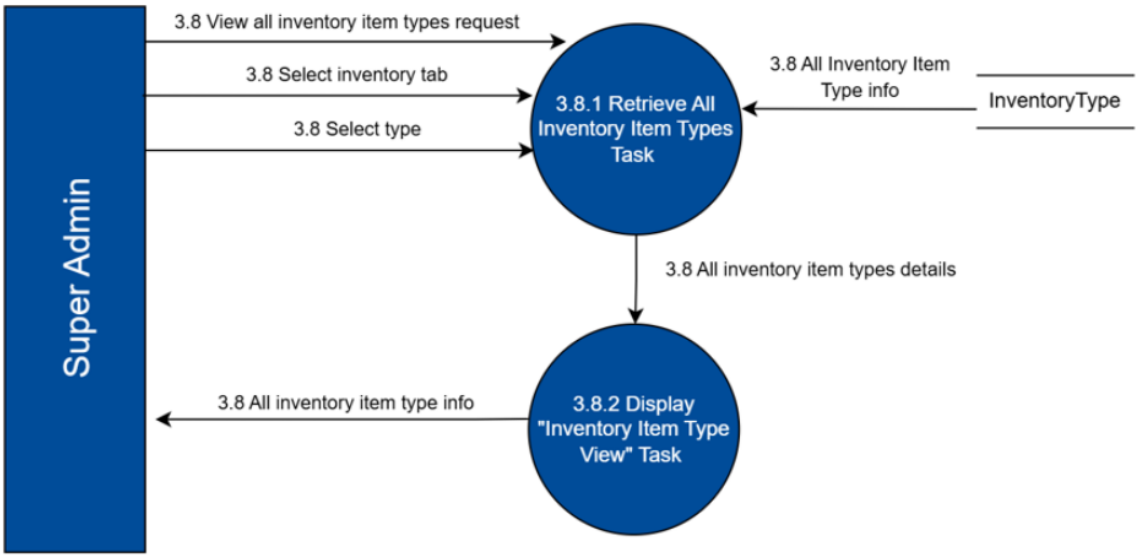


Figure 50 3.8 View Inventory Item Types - Primitive Data Flow Diagram

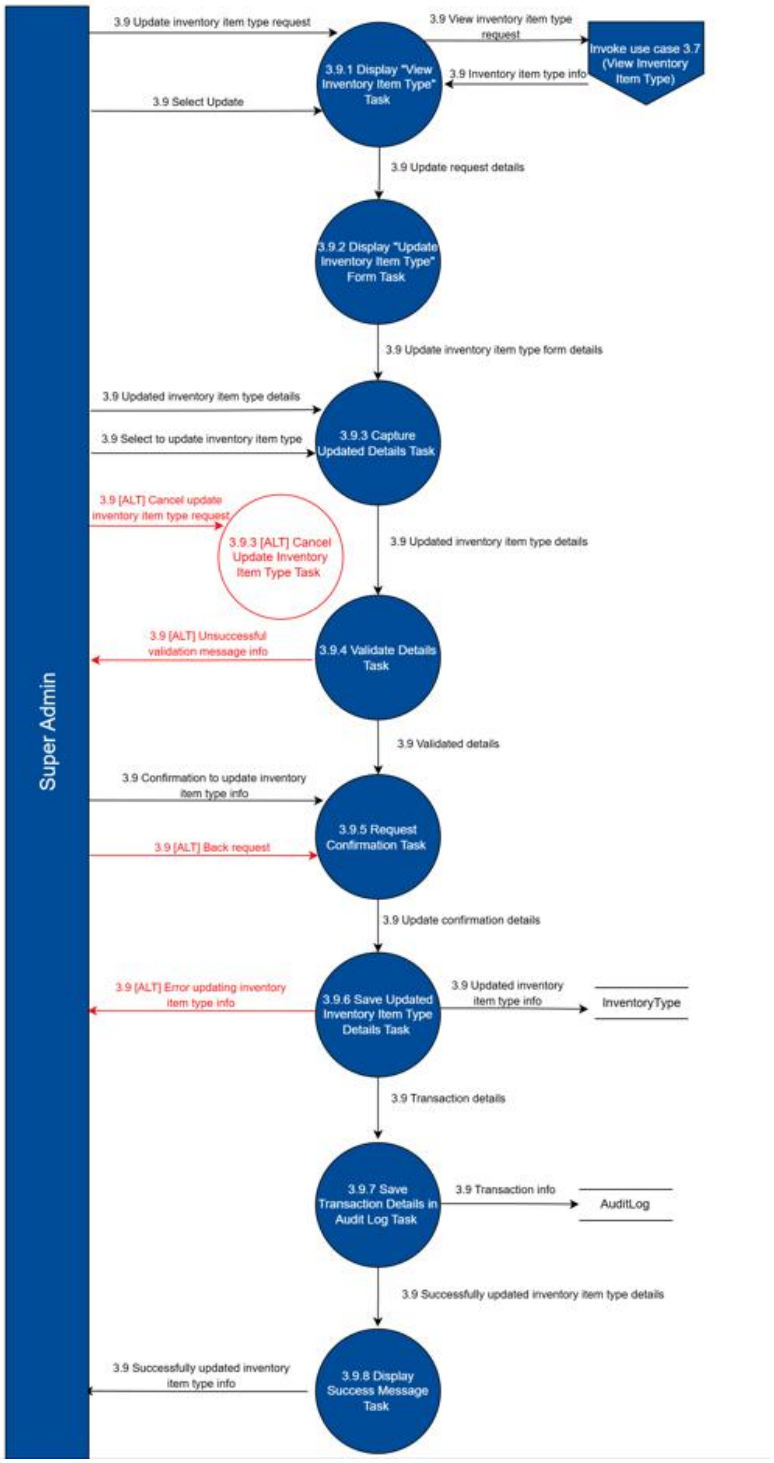


Figure 51 3.9 Update Inventory Item Type - Primitive Data Flow Diagram

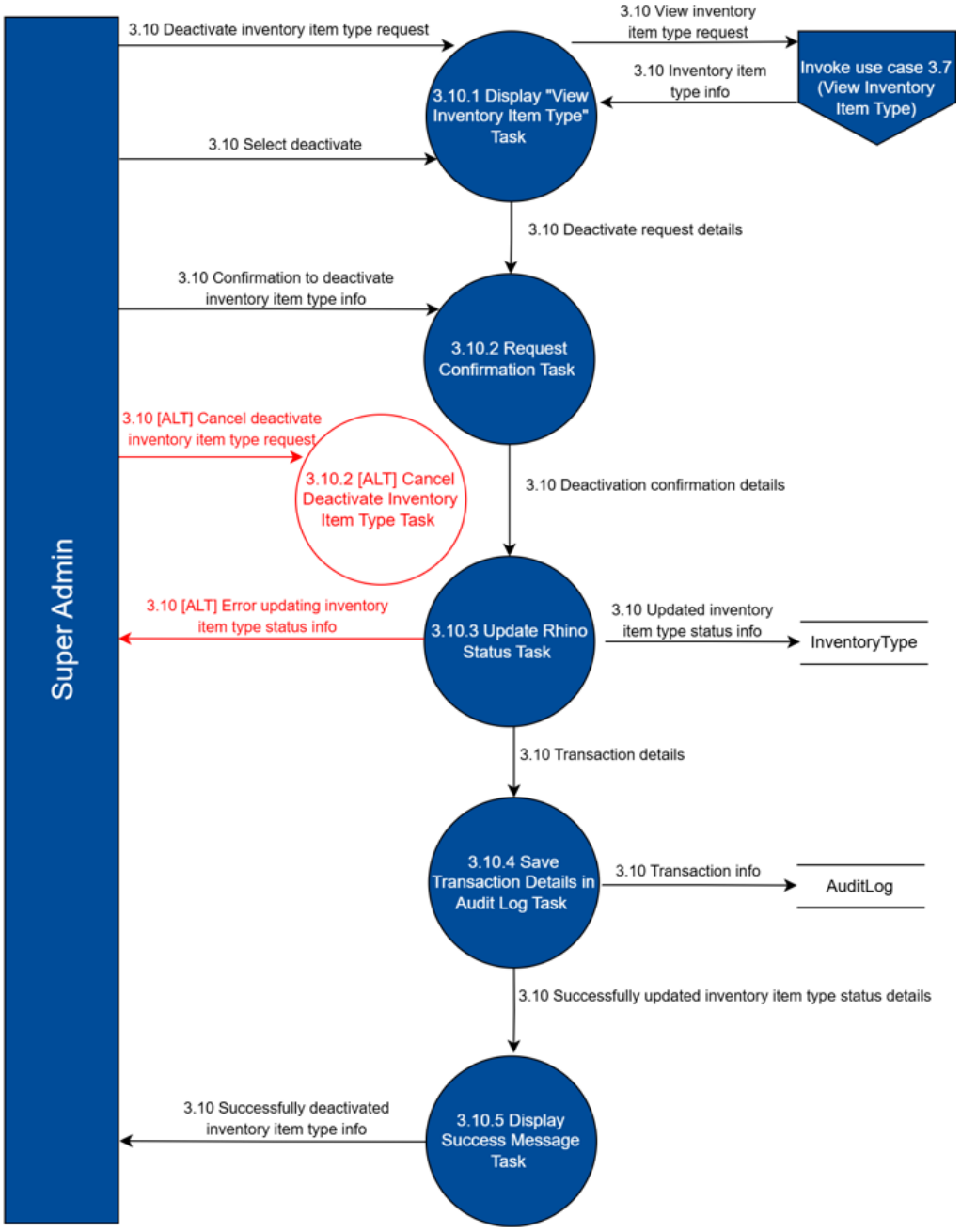


Figure 52 3.10 Deactivate Inventory Item Type - Primitive Data Flow Diagram

2.9.4 Group Subsystem Primitives

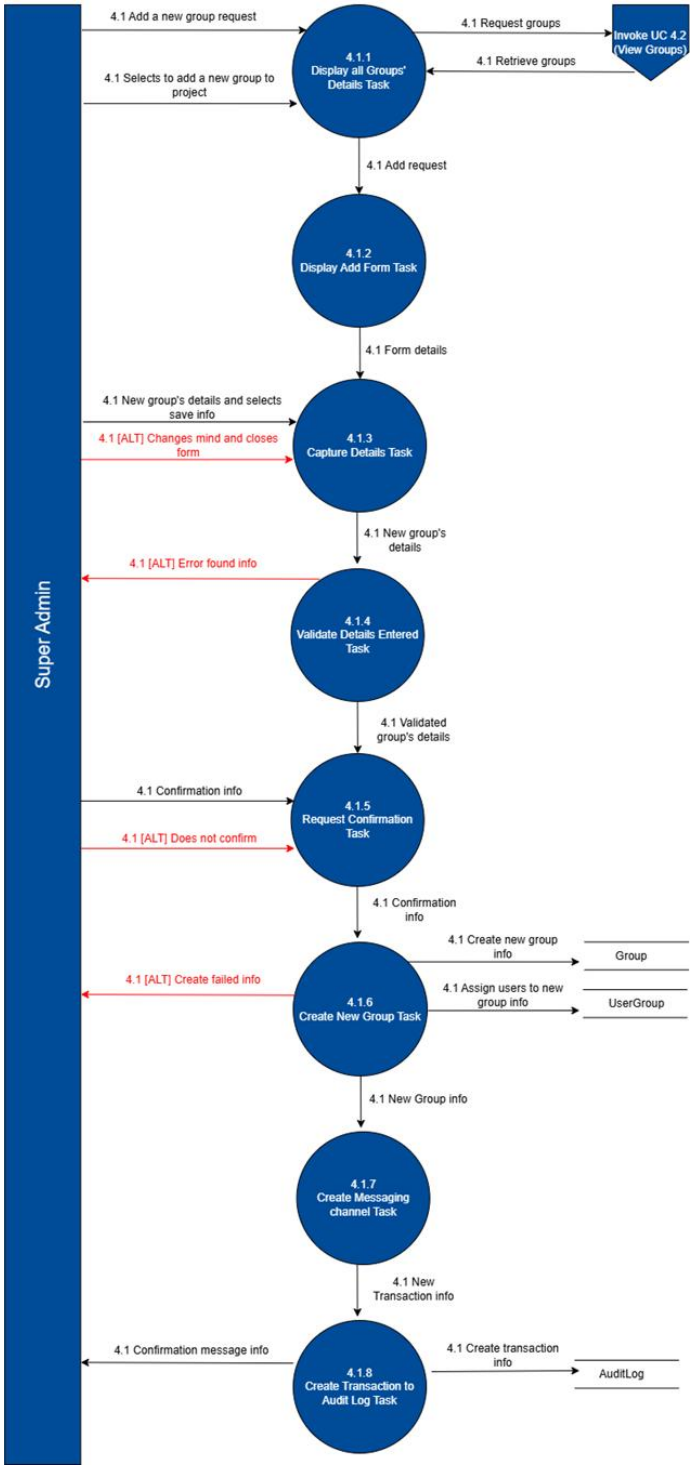


Figure 53 4.1 Create Group - Primitive Data Flow Diagram

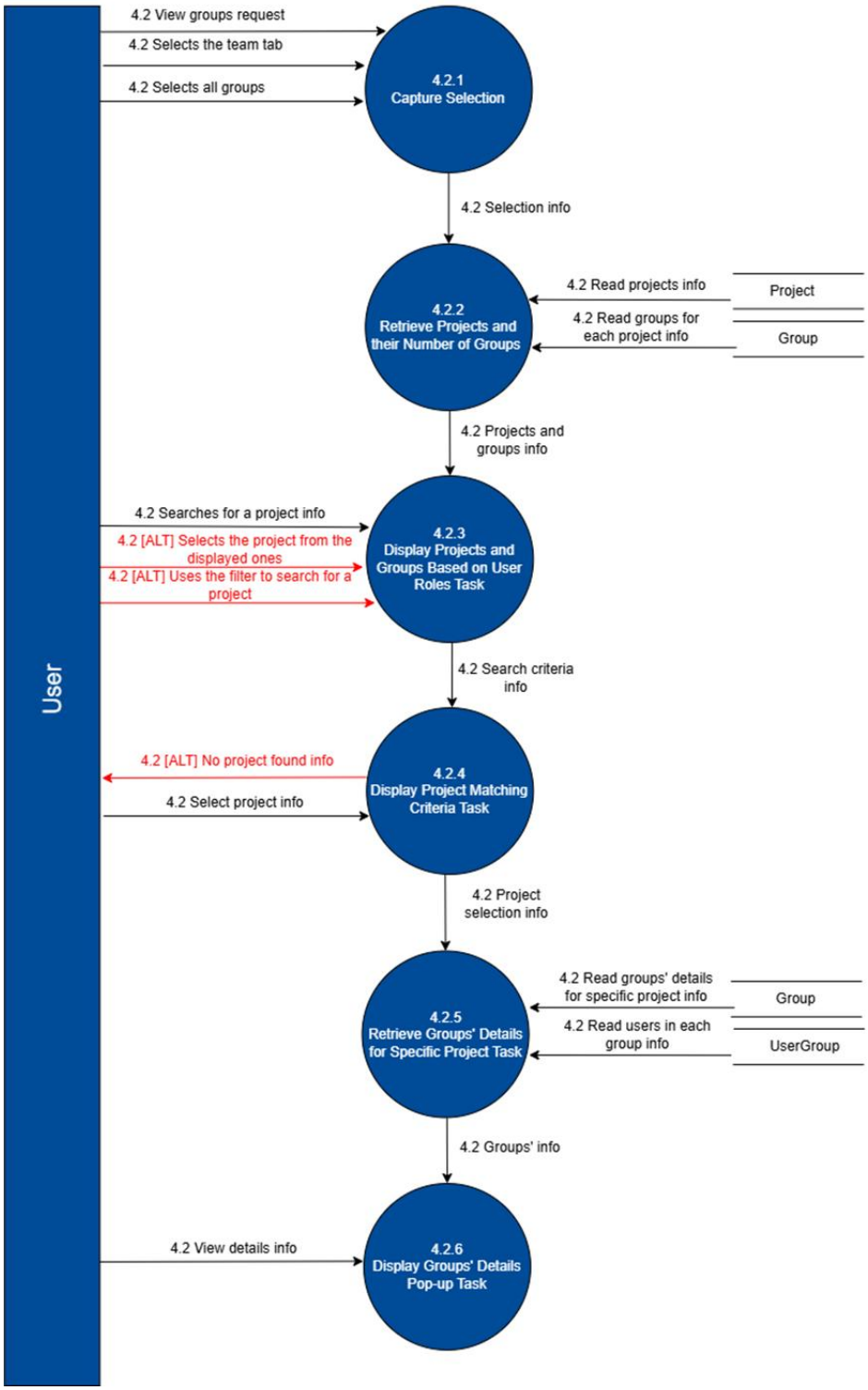


Figure 54 4.2 View Groups - Primitive Data Flow Diagram

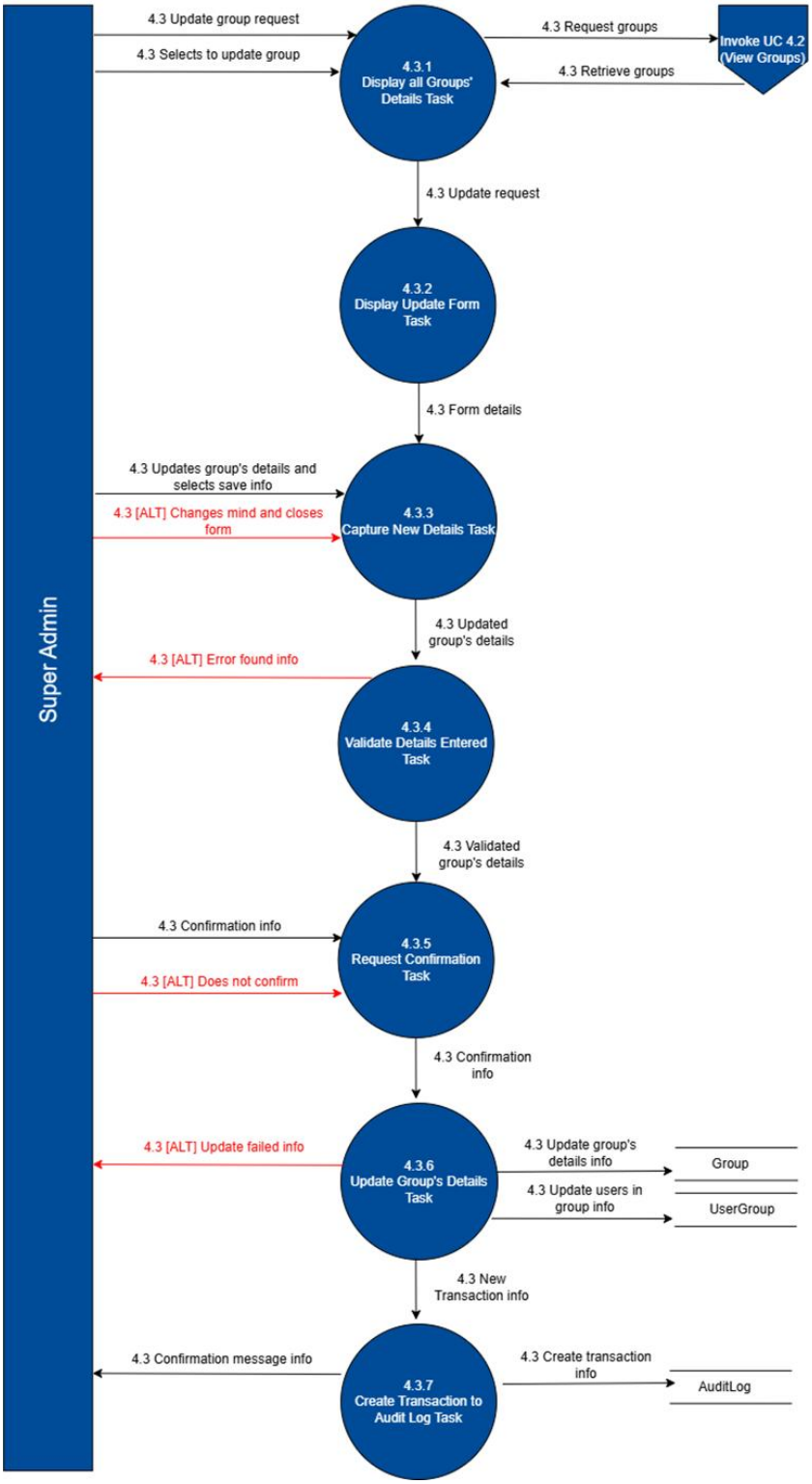


Figure 55 4.3 Update Group - Primitive Data Flow Diagram

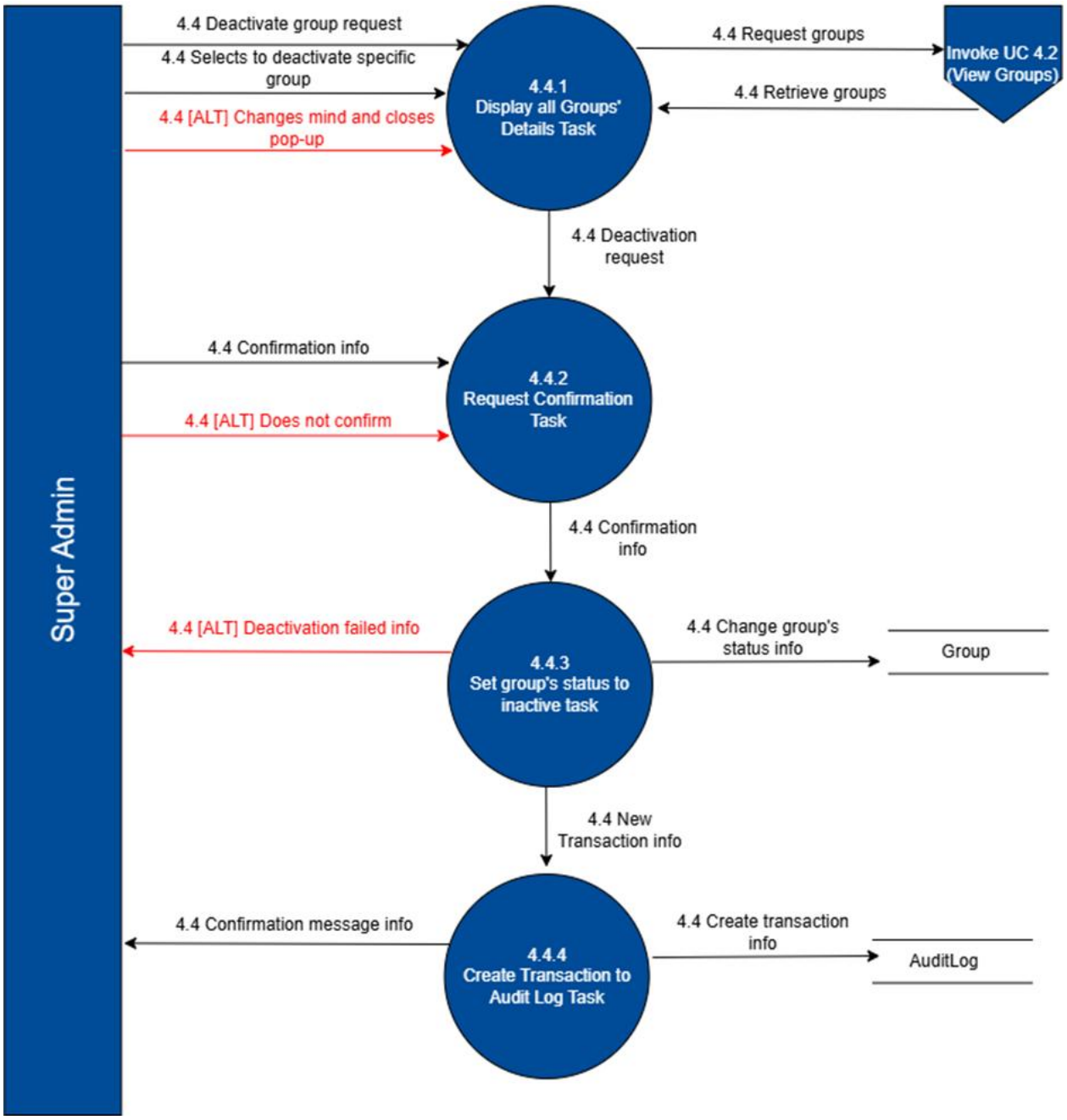


Figure 56 4.4 Deactivate Group - Primitive Data Flow Diagram

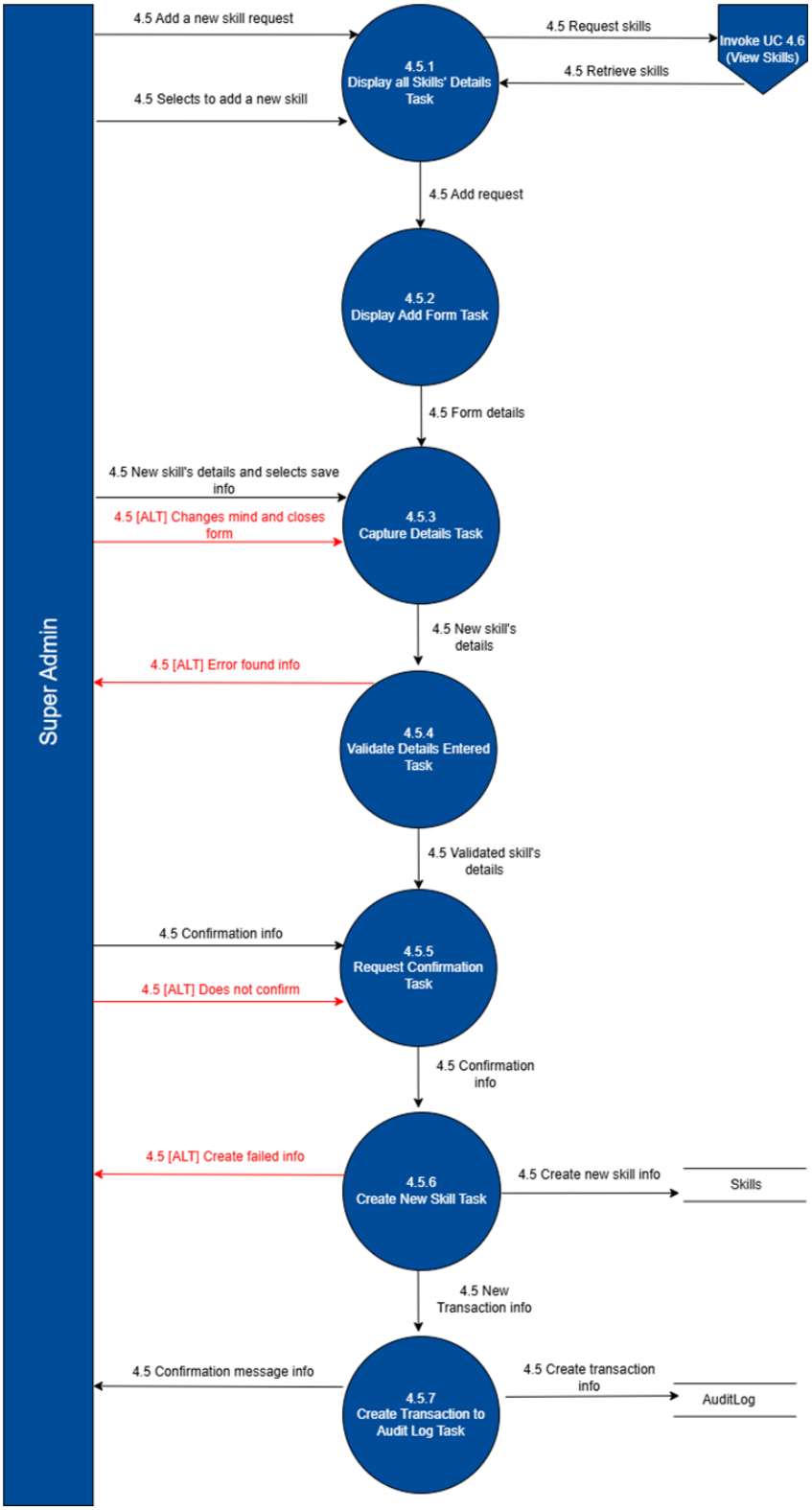


Figure 57 4.5 Create Skill - Primitive Data Flow Diagram

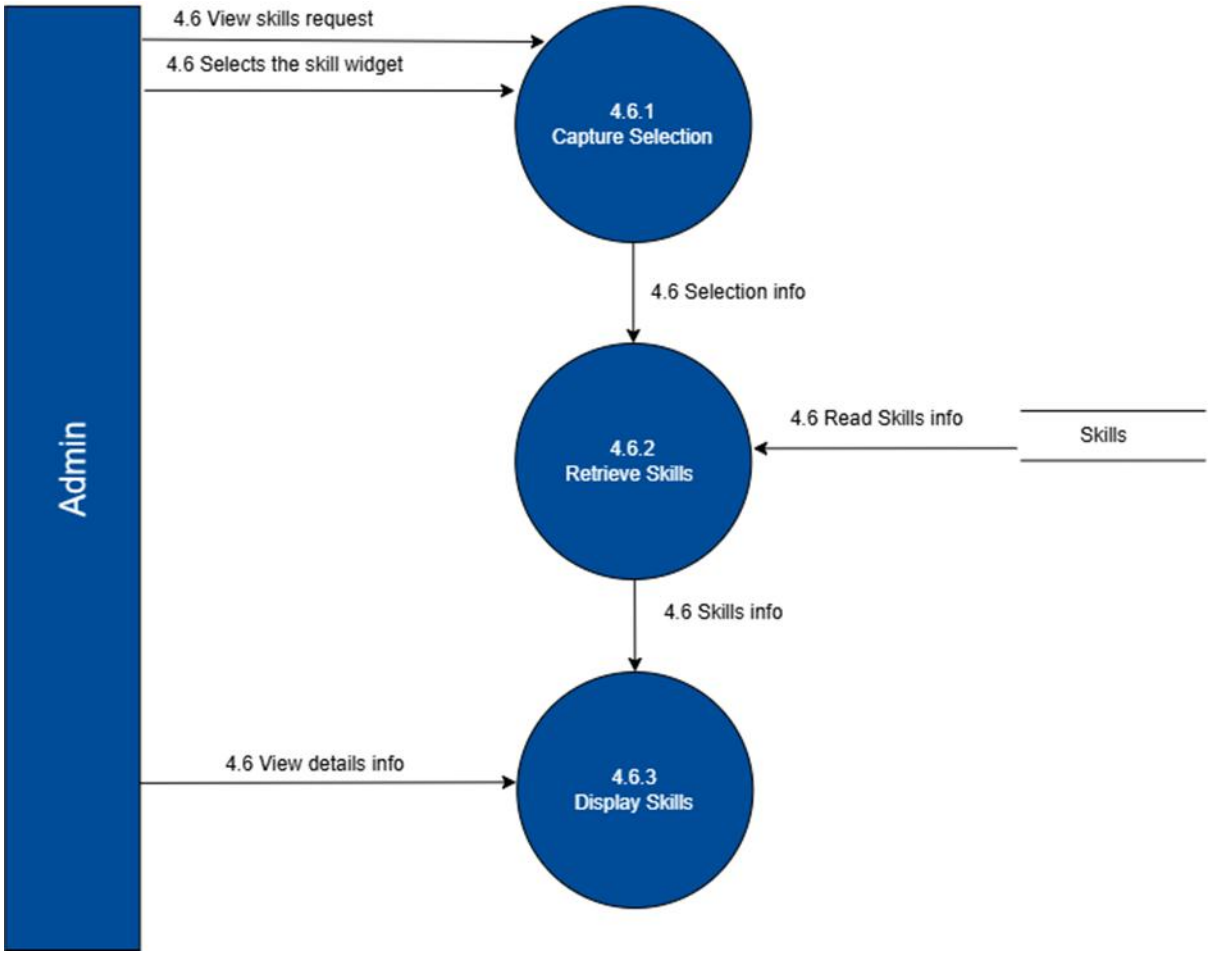


Figure 58 4.6 View Skills - Primitive Data Flow Diagram

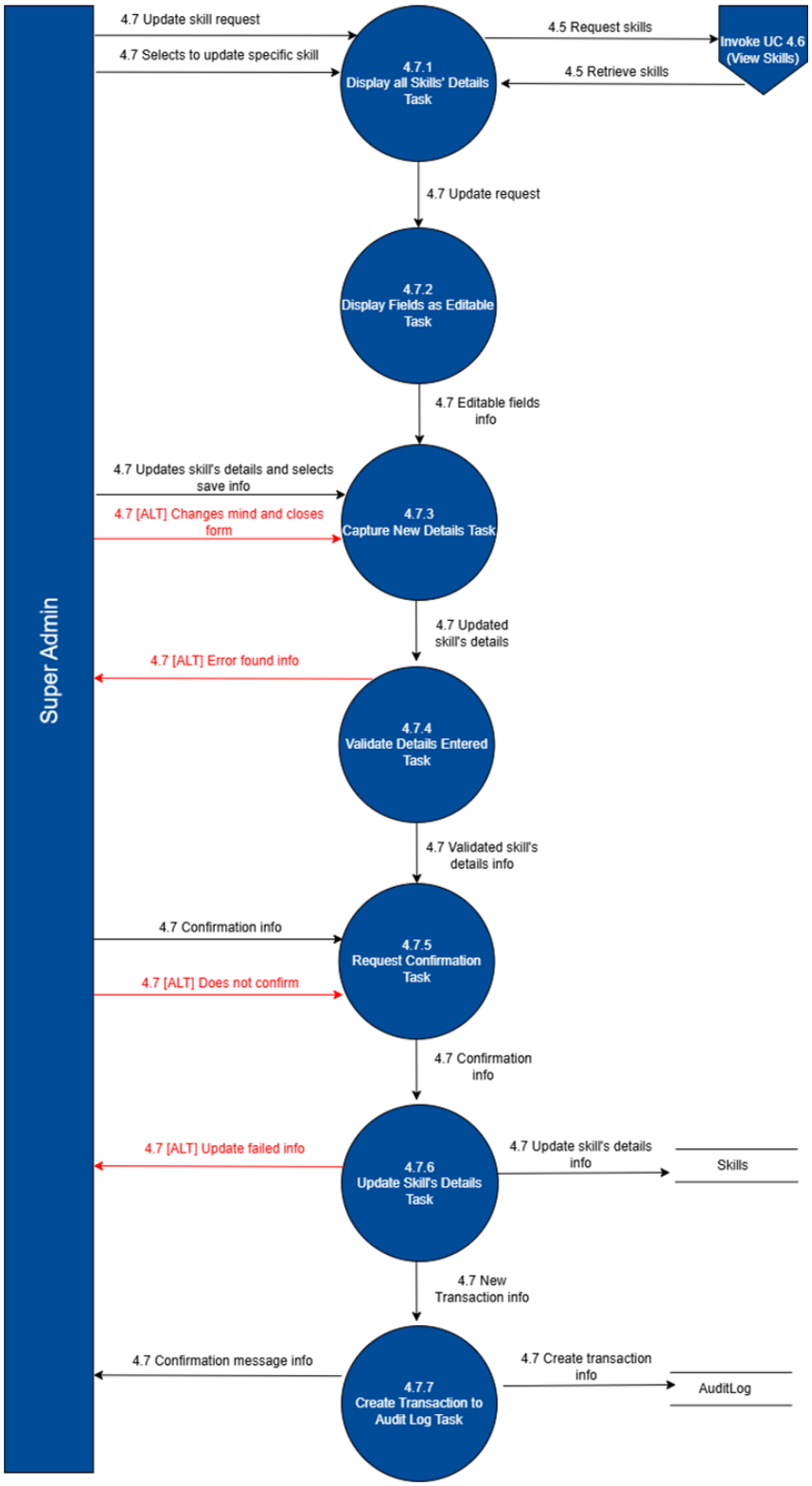


Figure 59 4.7 Update Skill - Primitive Data Flow Diagram

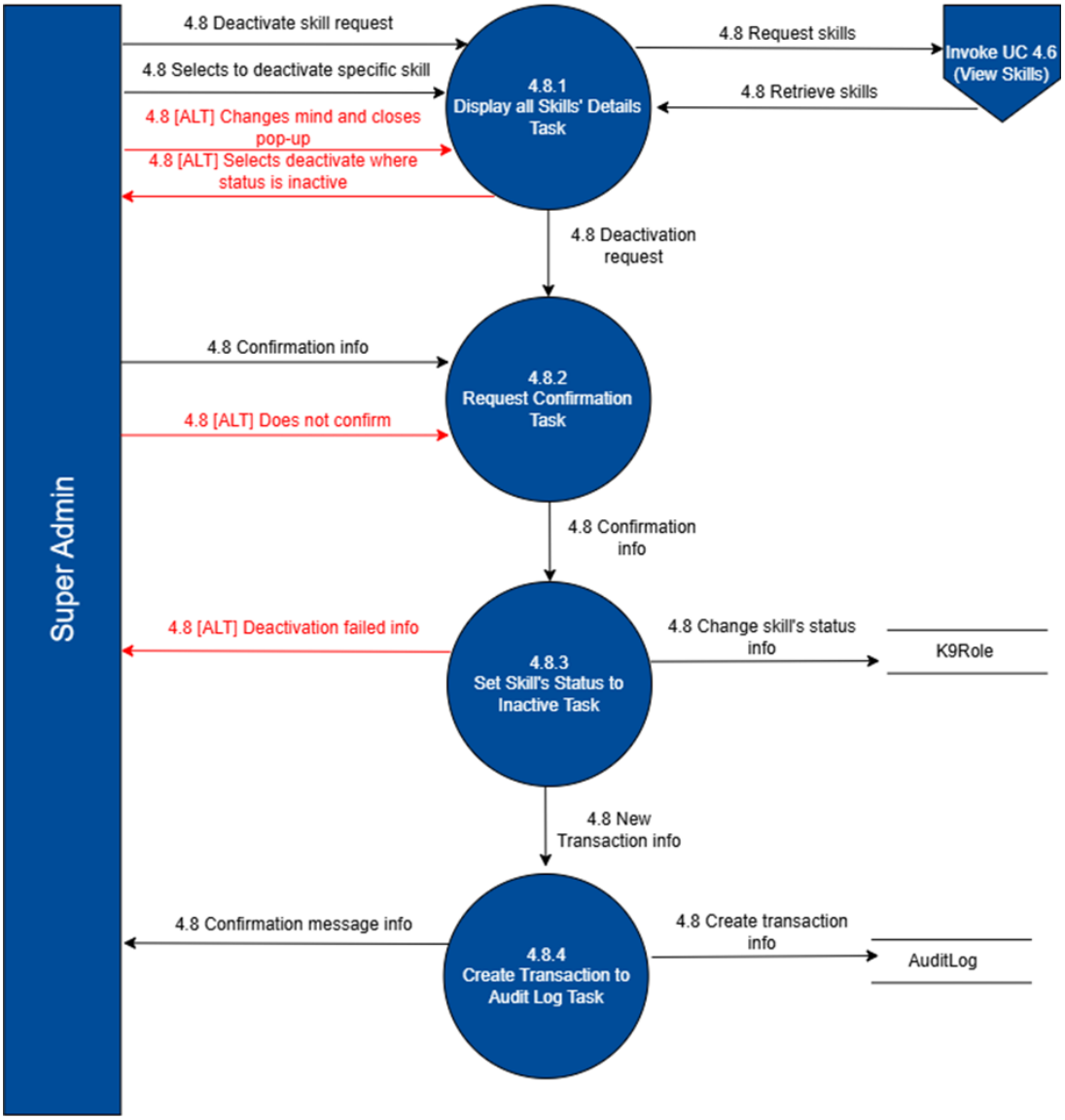


Figure 60 4.8 Deactivate Skill - Primitive Data Flow Diagram

2.9.5 Warehouse Subsystem Primitives

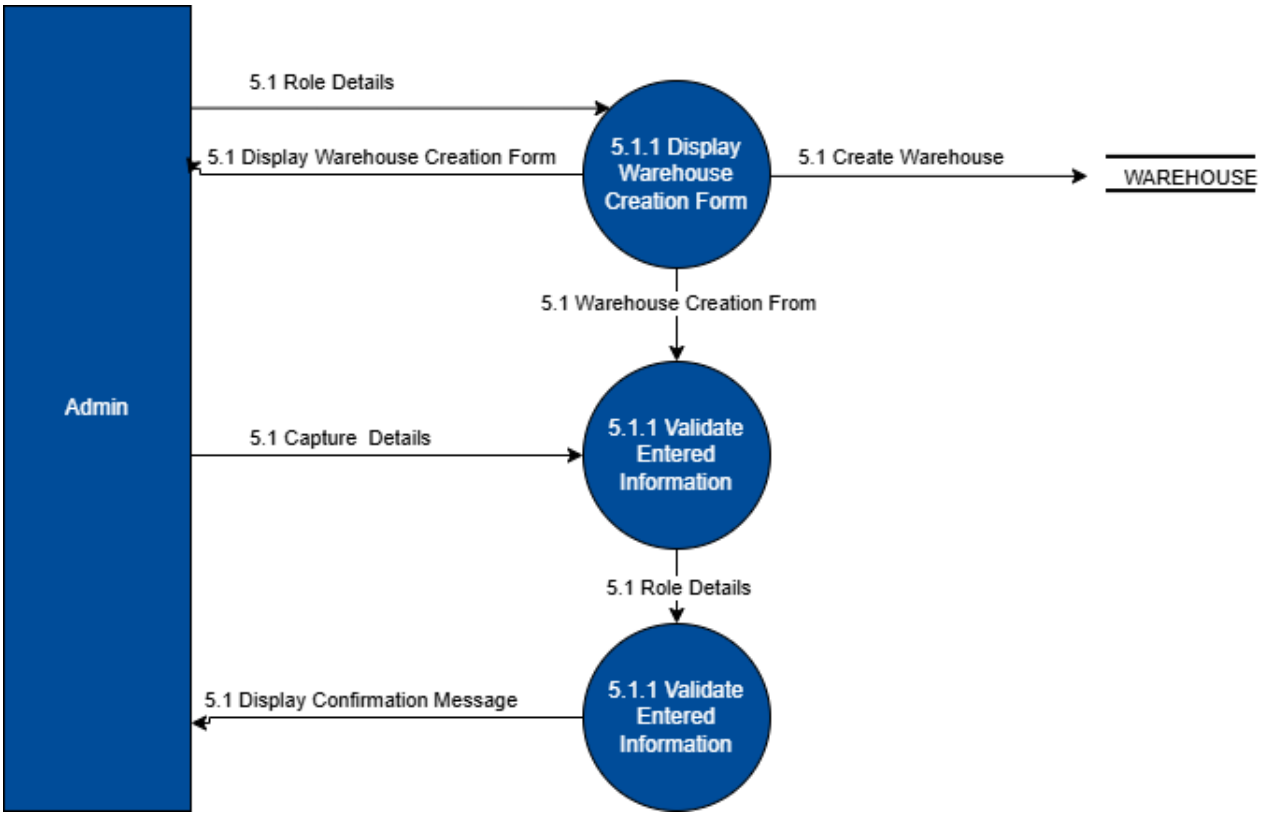


Figure 61 5.1 Create Warehouse - Primitive Data Flow Diagram

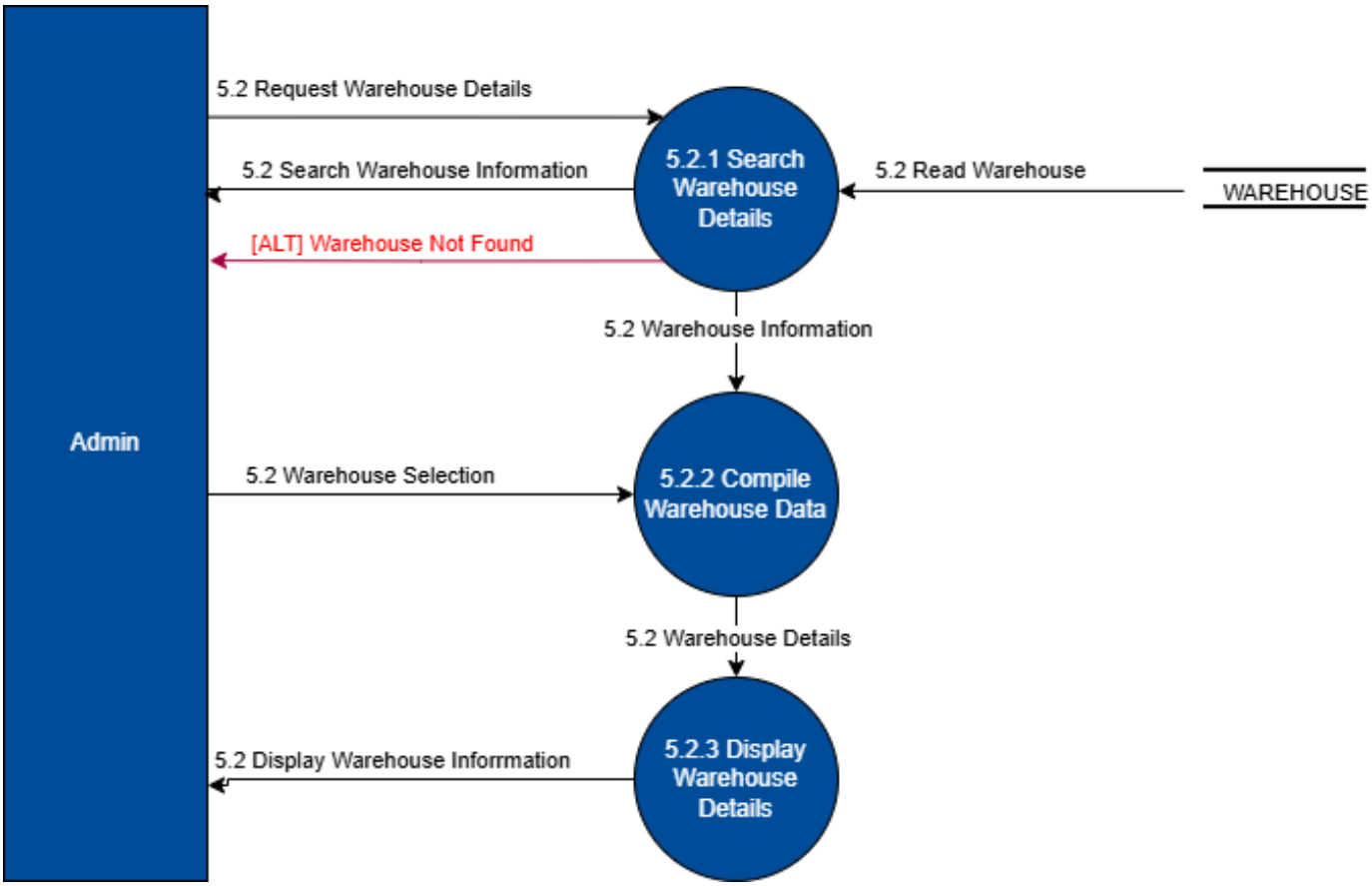


Figure 62 5.2 View Warehouse - Primitive Data Flow Diagram

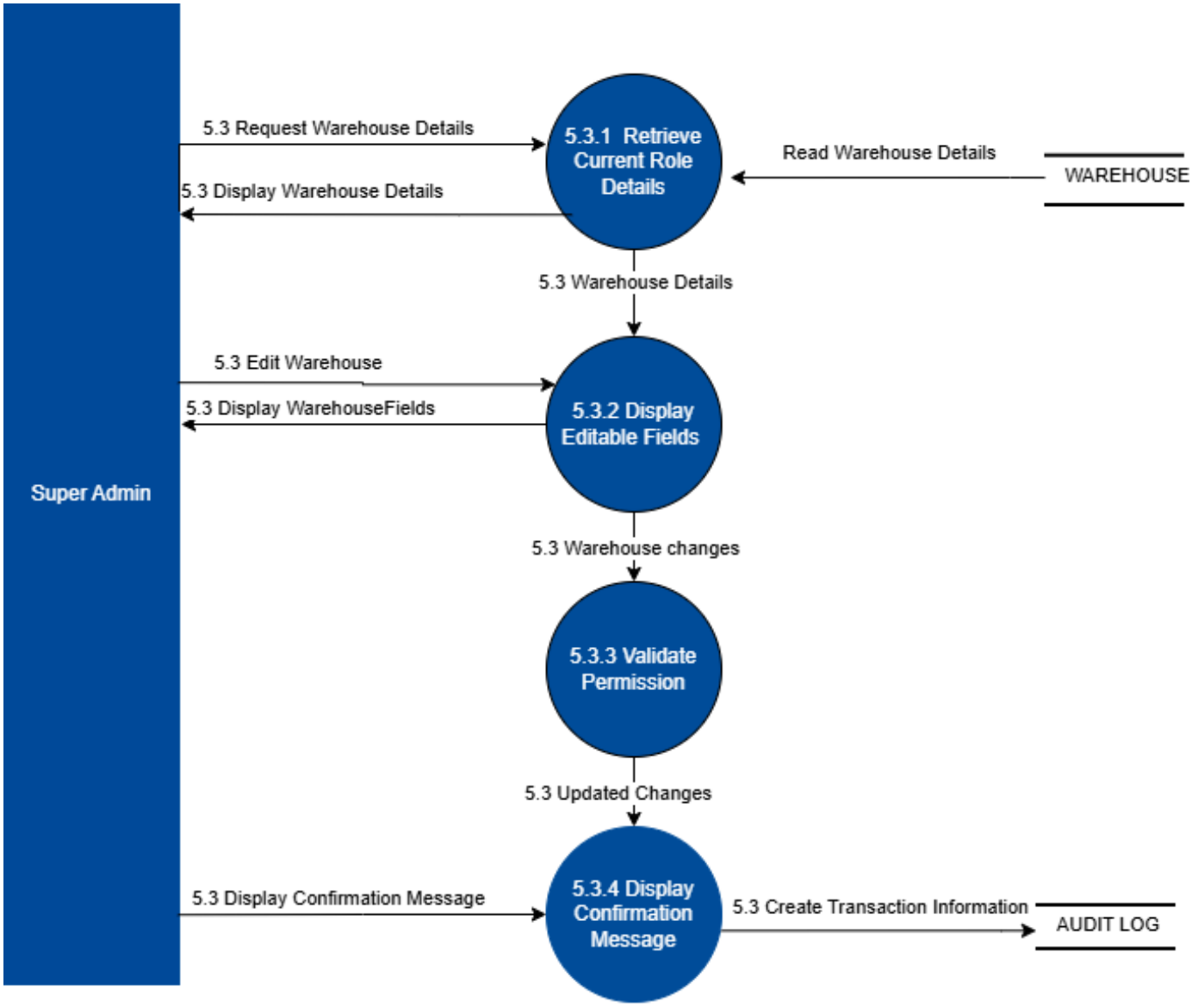


Figure 63 5.3 Update Warehouse - Primitive Data Flow Diagram

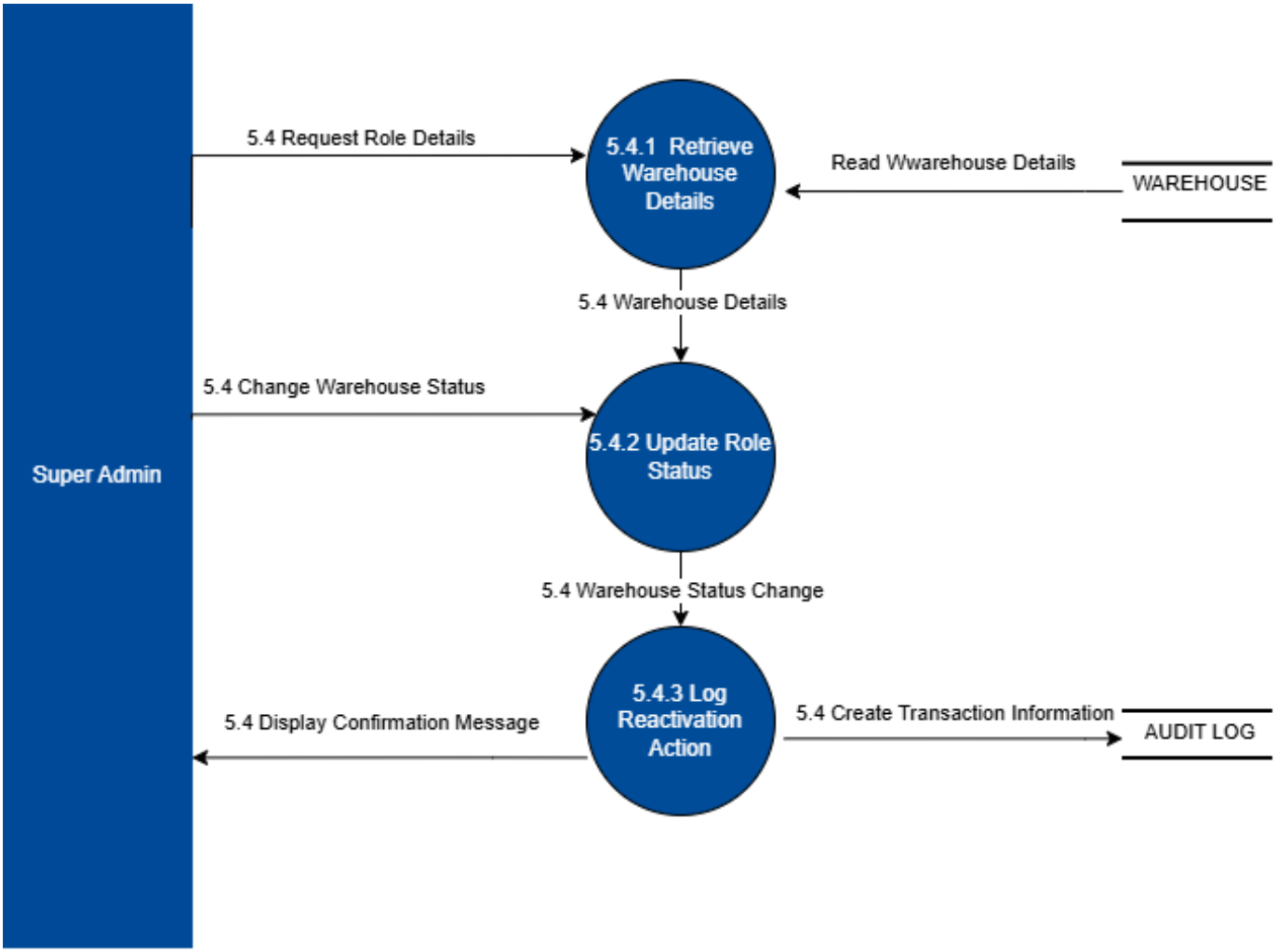


Figure 64 5.4 Reactivate Warehouse - Primitive Data Flow Diagram

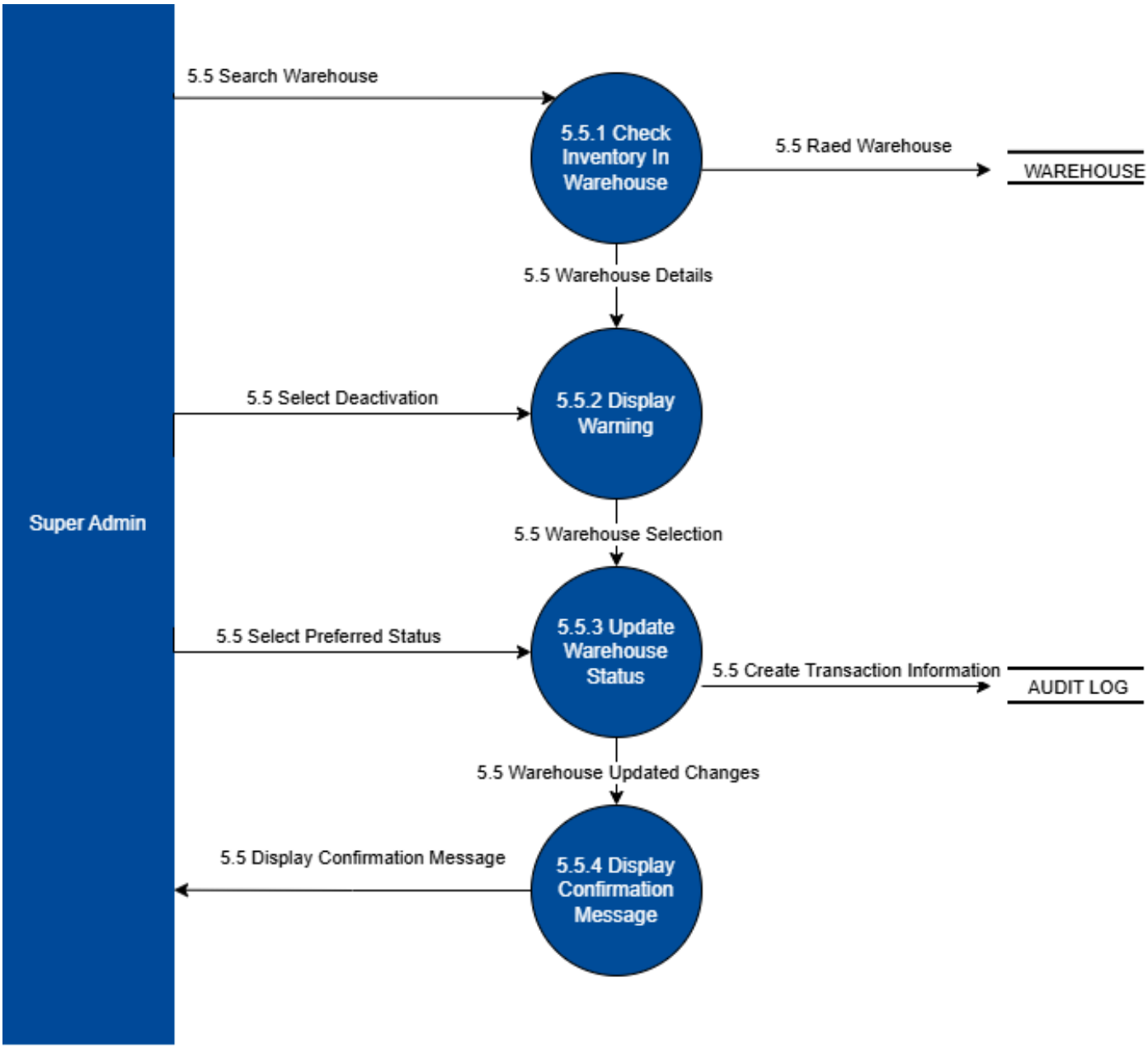


Figure 65 5.5 Deactivate Warehouse - Primitive Data Flow Diagram

2.9.6 Vendor Subsystem Primitives

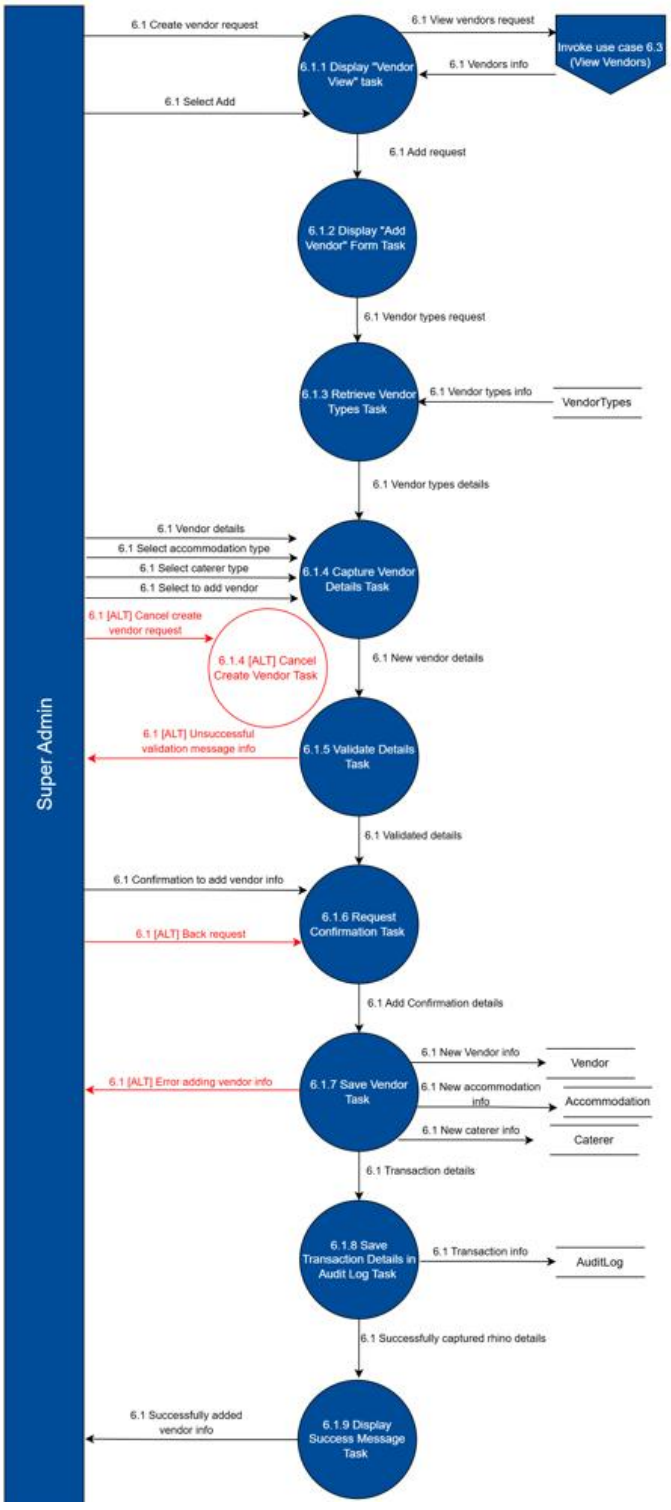


Figure 66 6.1 Create Vendor - Primitive Data Flow Diagram

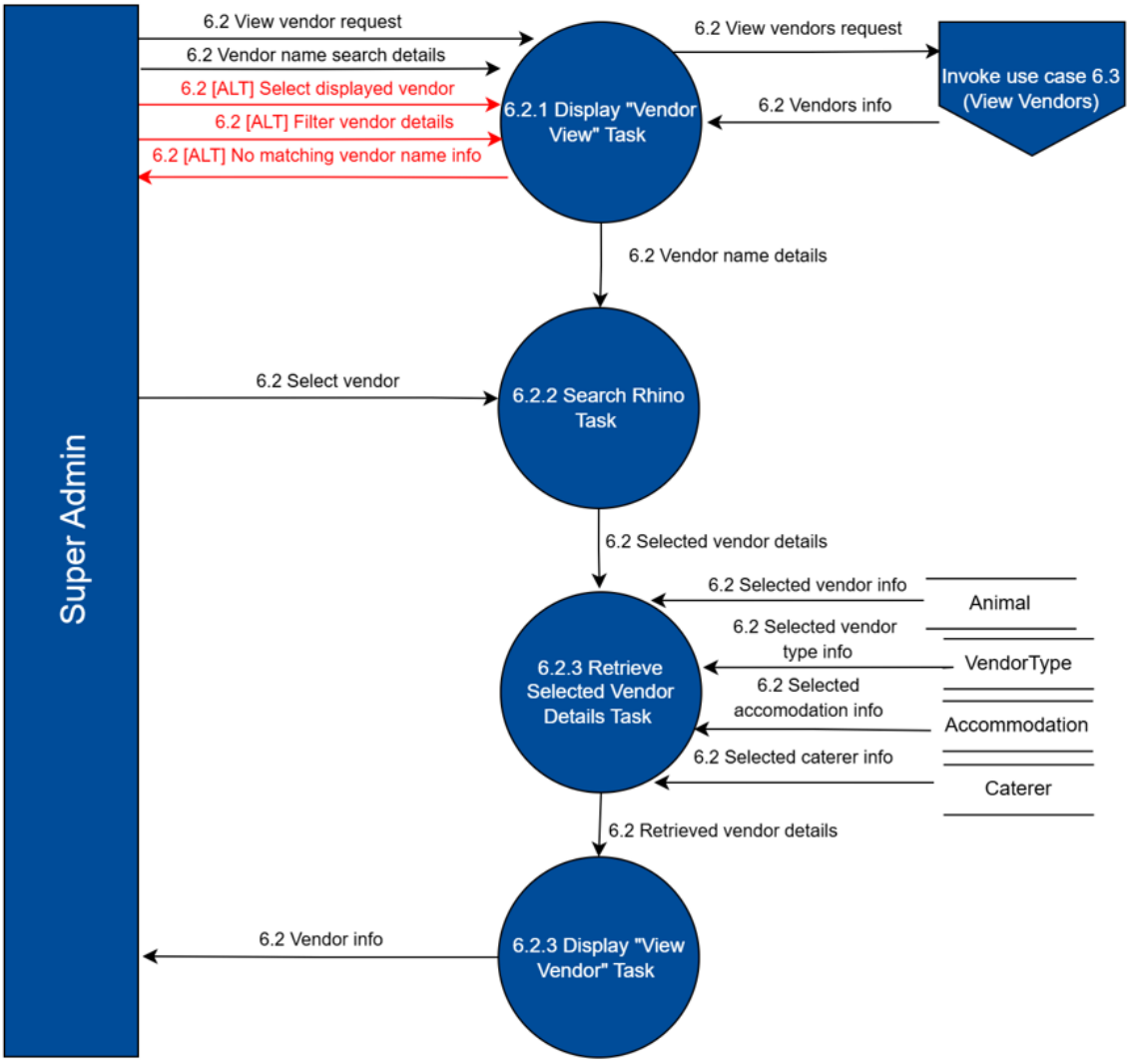


Figure 67 6.2 View Vendor - Primitive Data Flow Diagram

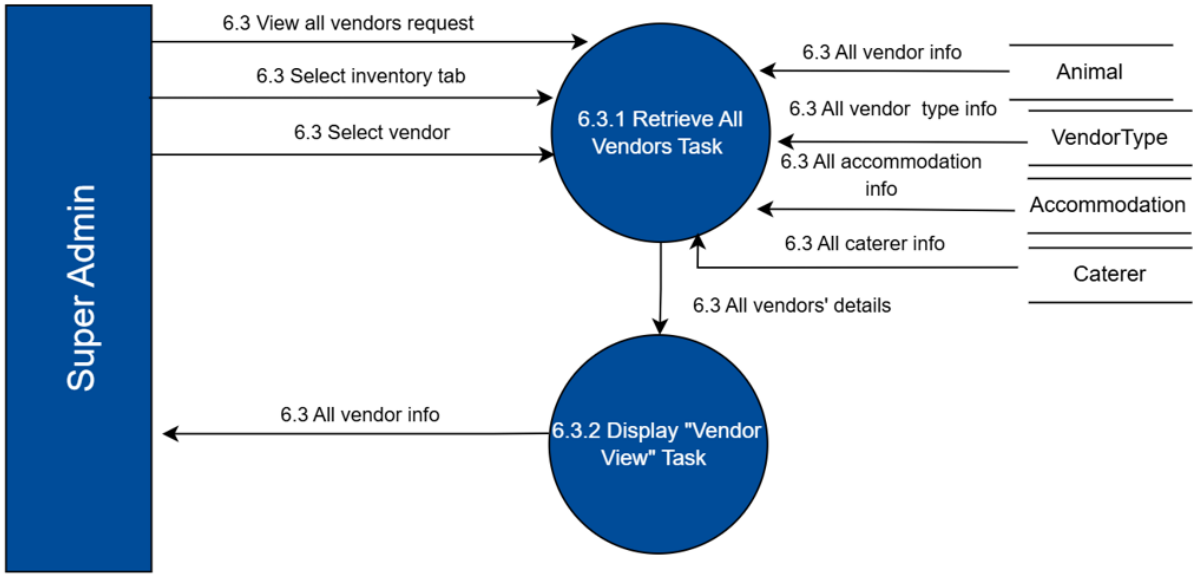


Figure 68 6.3 View Vendors - Primitive Data Flow Diagram

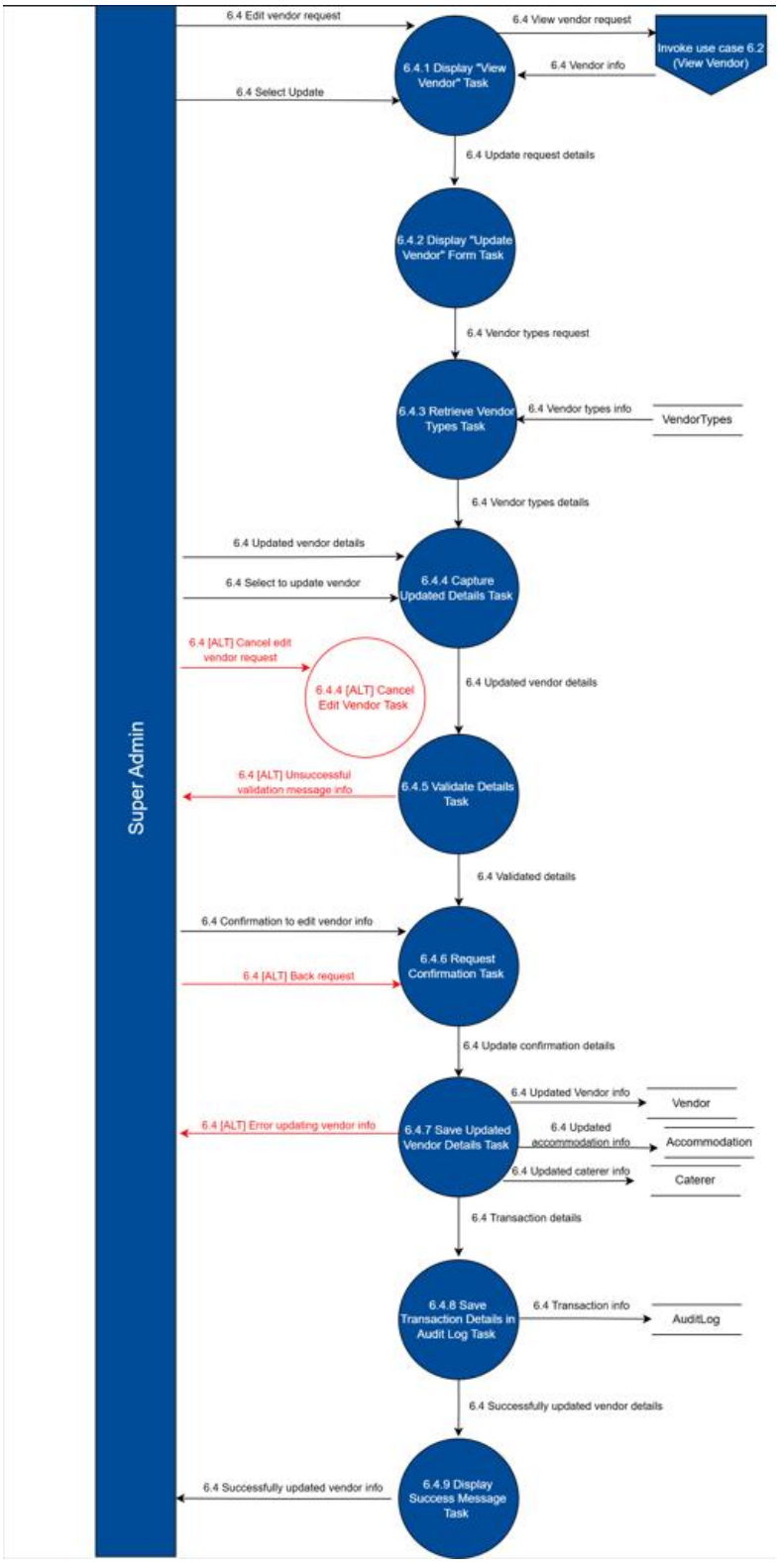


Figure 69 6.4 Edit Vendor - Primitive Data Flow Diagram

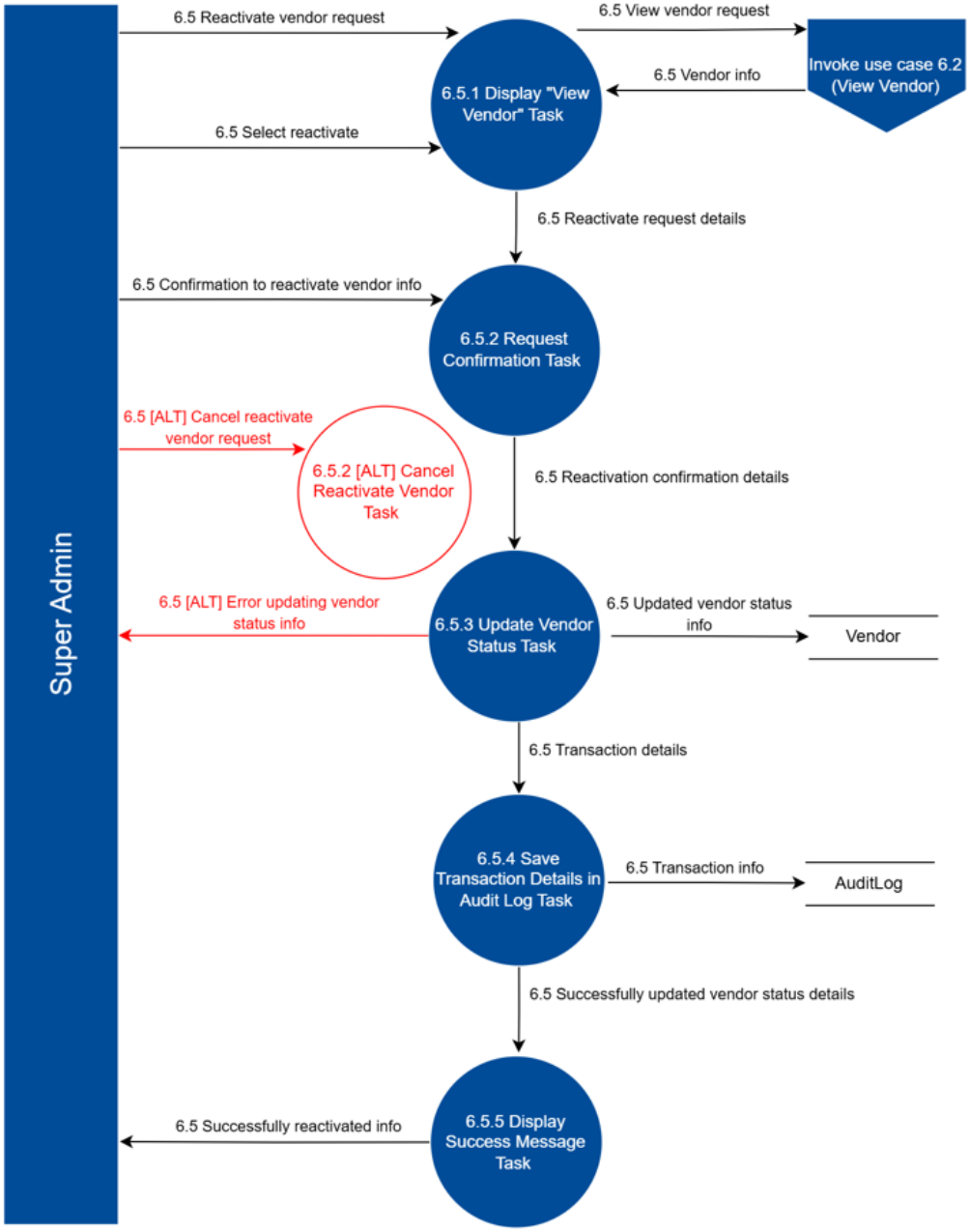


Figure 70 6.5 Reactivate Vendor - Primitive Data Flow Diagram

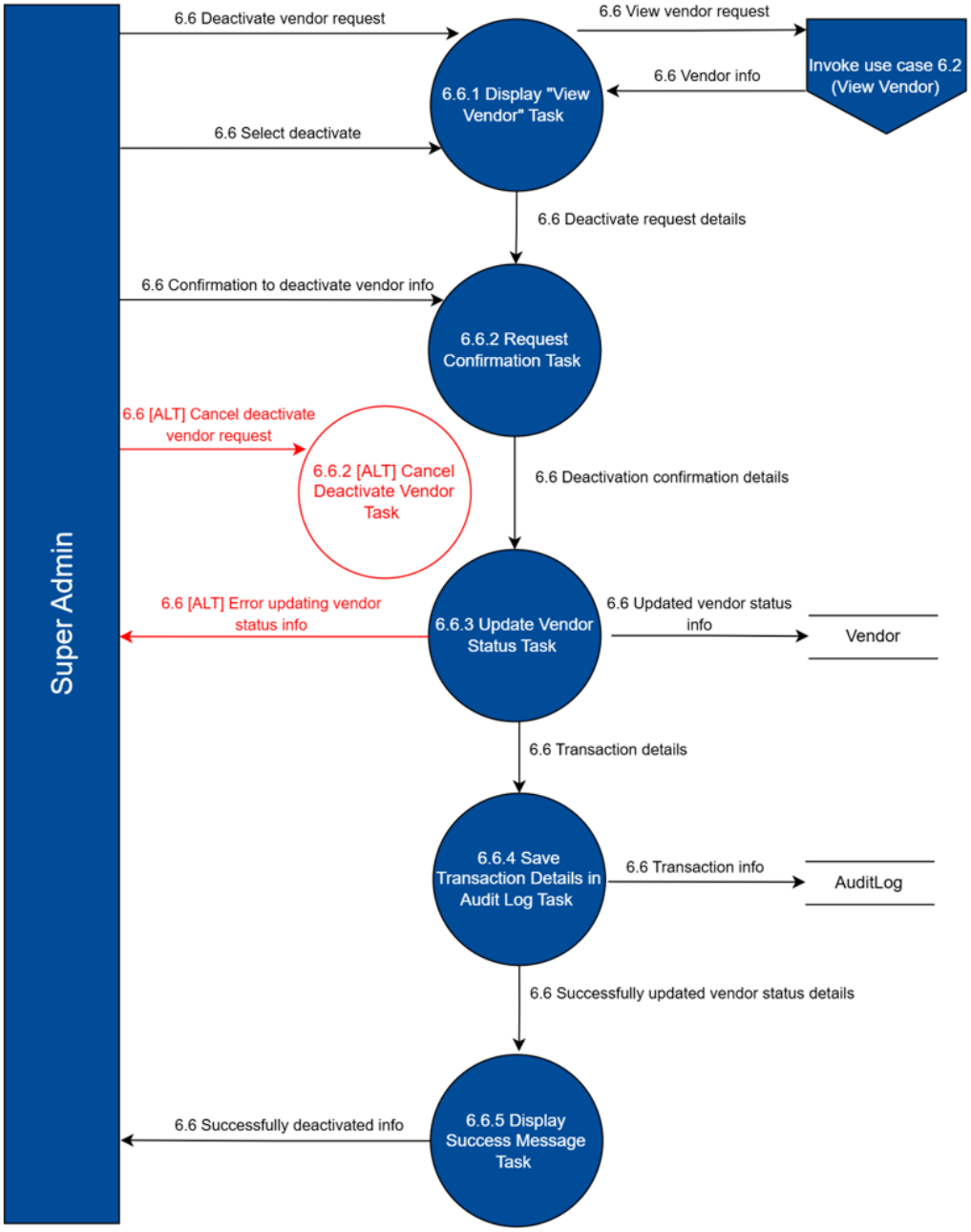


Figure 71 6.6 Deactivate Vendor - Primitive Data Flow Diagram

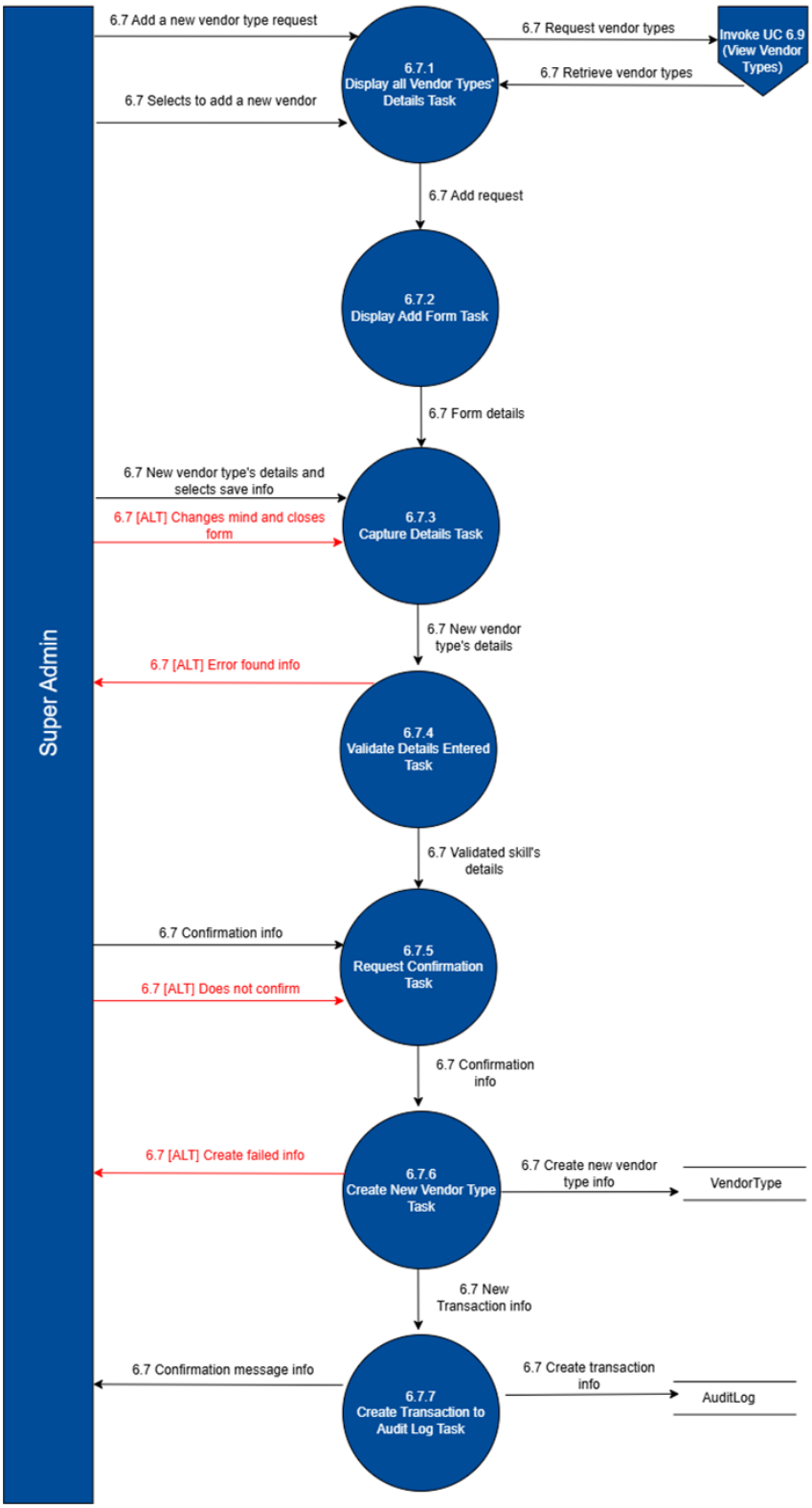


Figure 72 6.7 Create Vendor Type - Primitive Data Flow Diagram

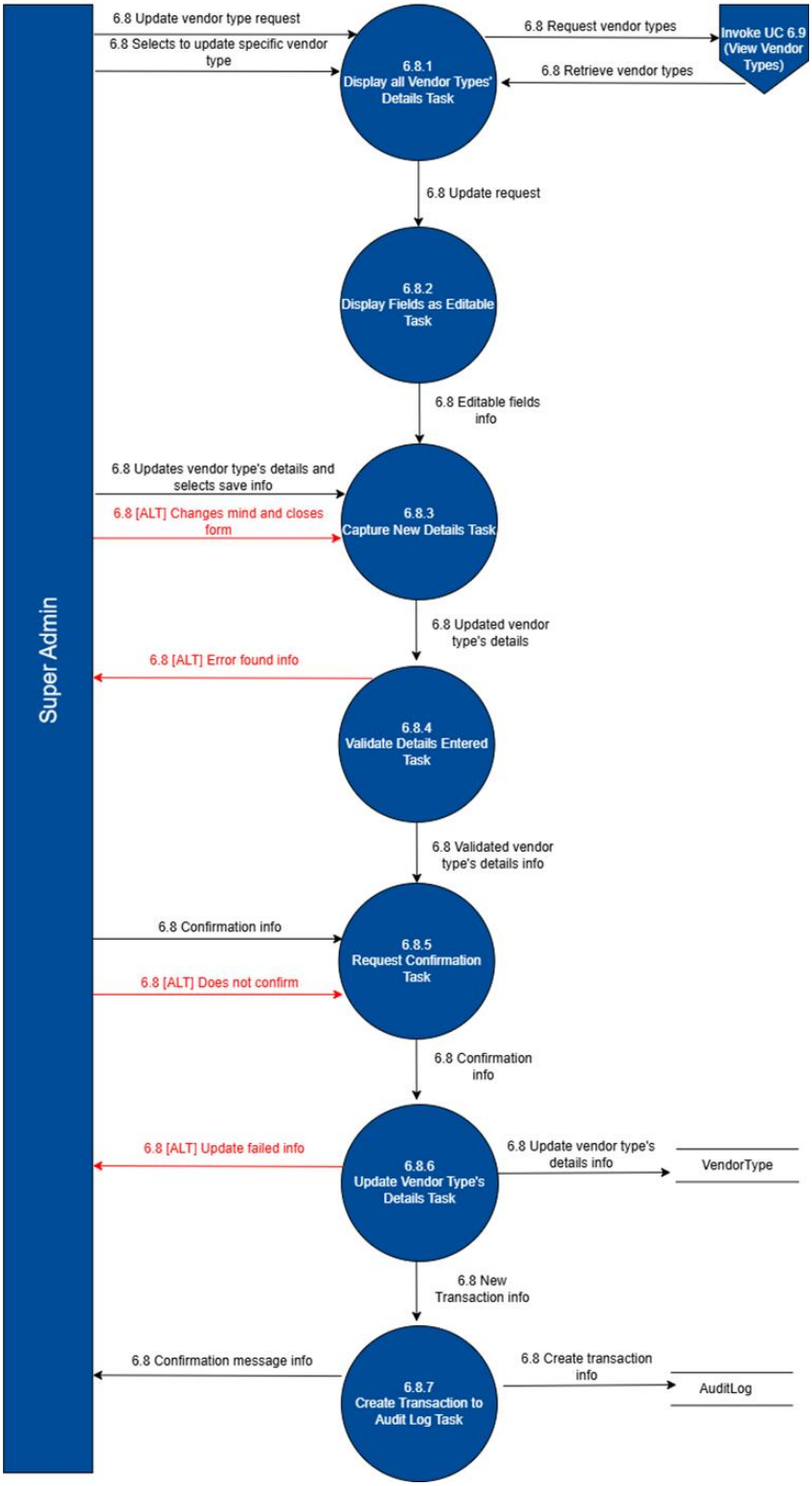


Figure 73 6.8 Update Vendor Type - Primitive Data Flow Diagram

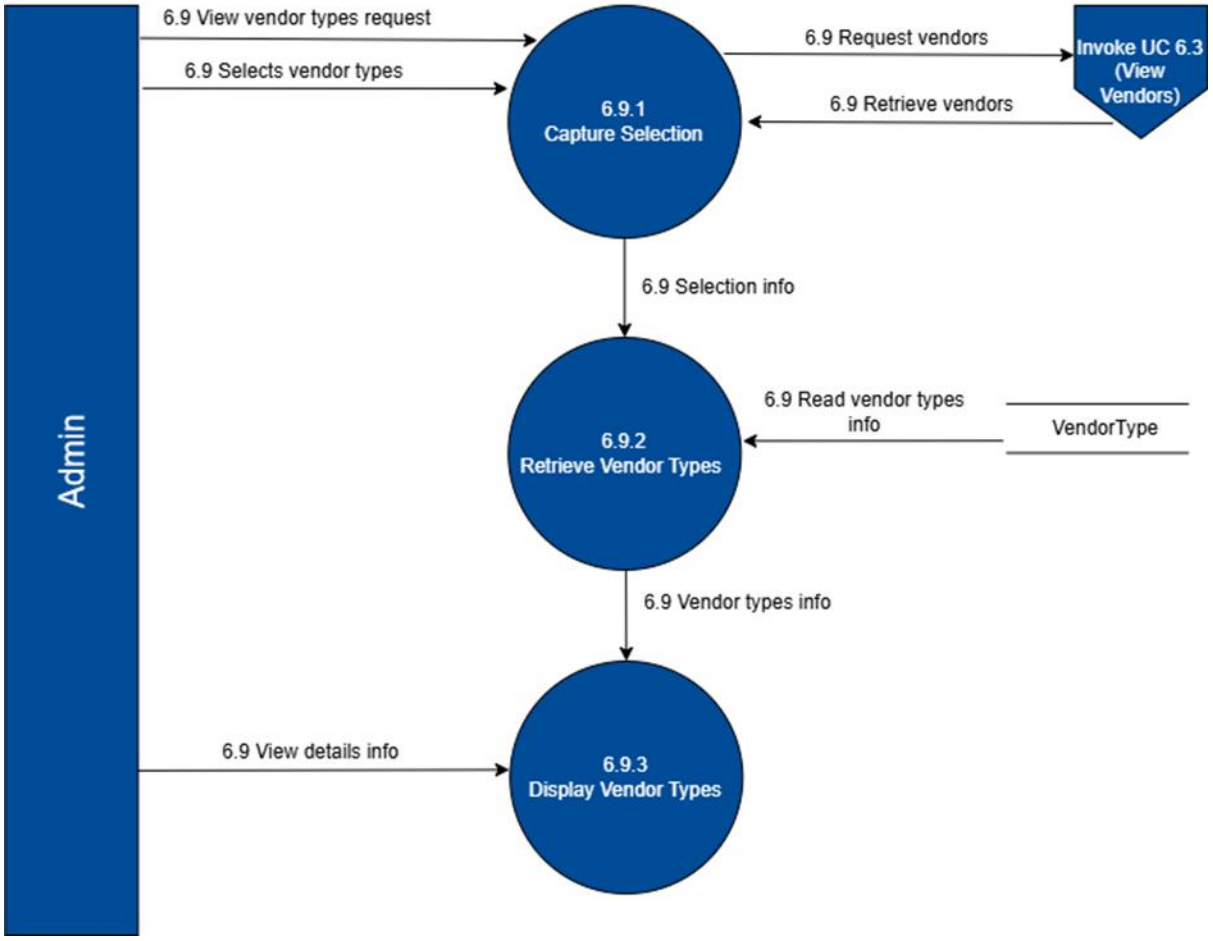


Figure 74 6.9 View Vendor Types - Primitive Data Flow Diagram

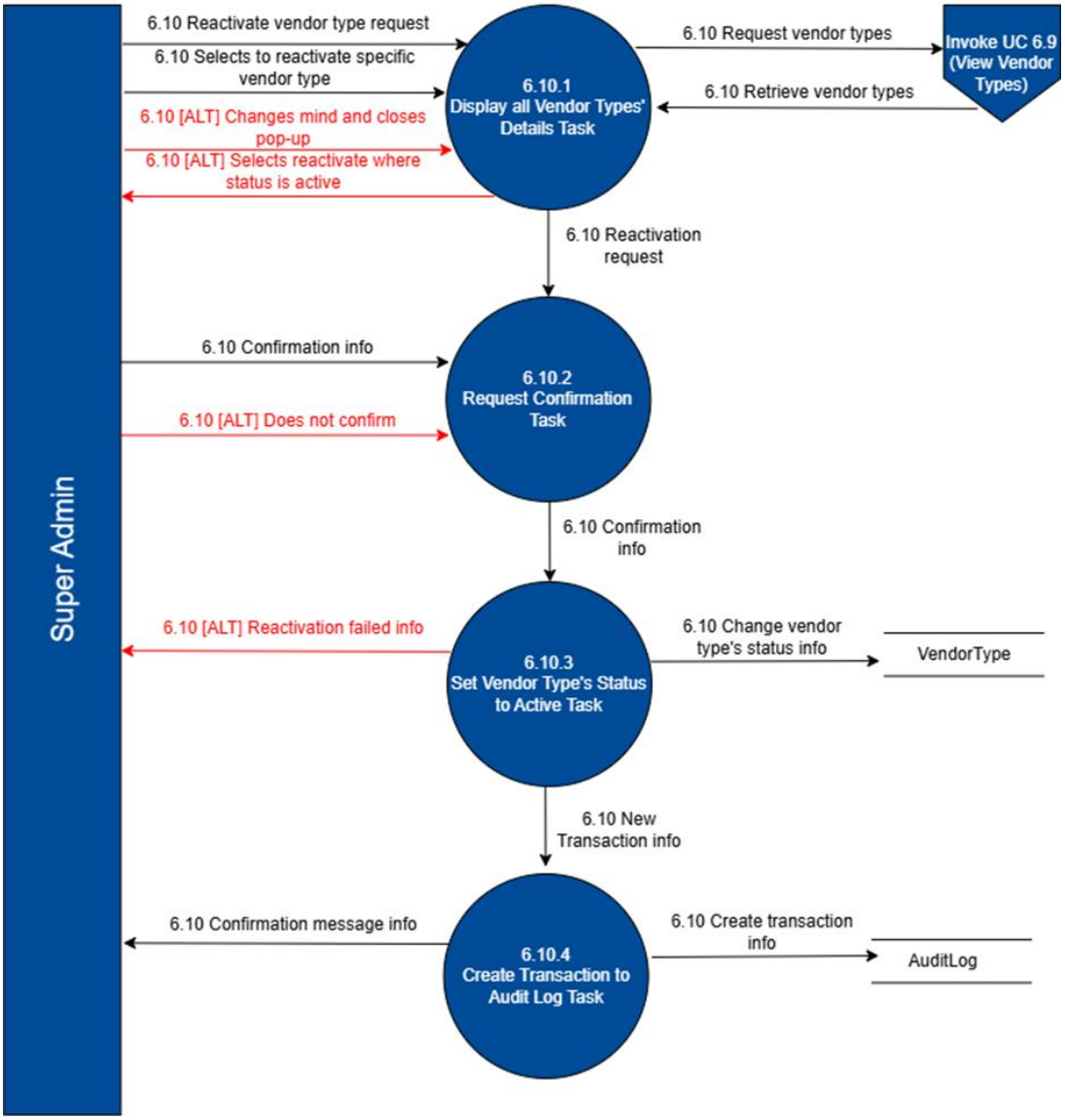


Figure 75 6.10 Reactivate Vendor Type - Primitive Data Flow Diagram

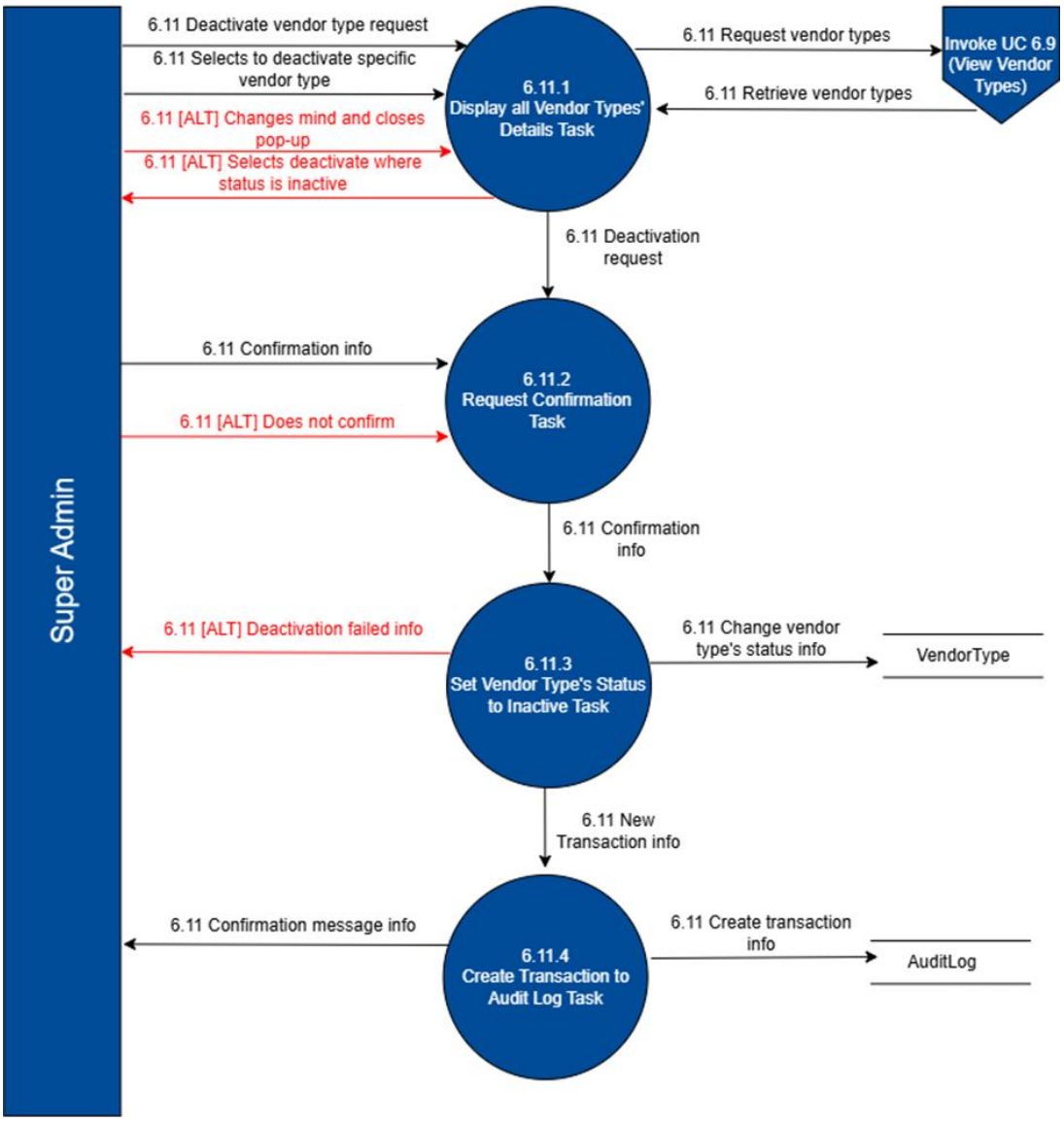


Figure 76 6.11 Deactivate Vendor Type - Primitive Data Flow Diagram

2.9.7 Project Subsystem Primitives

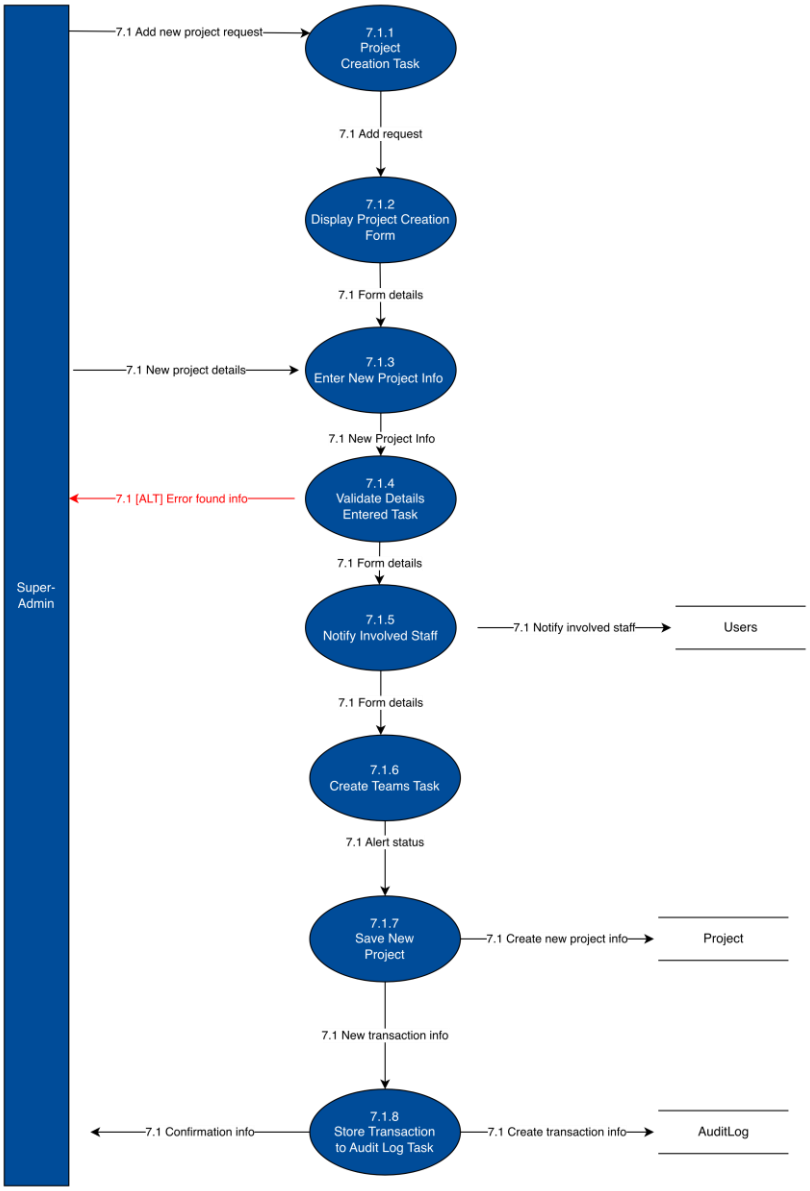


Figure 77 7.1 Add Project - Primitive Data Flow Diagram

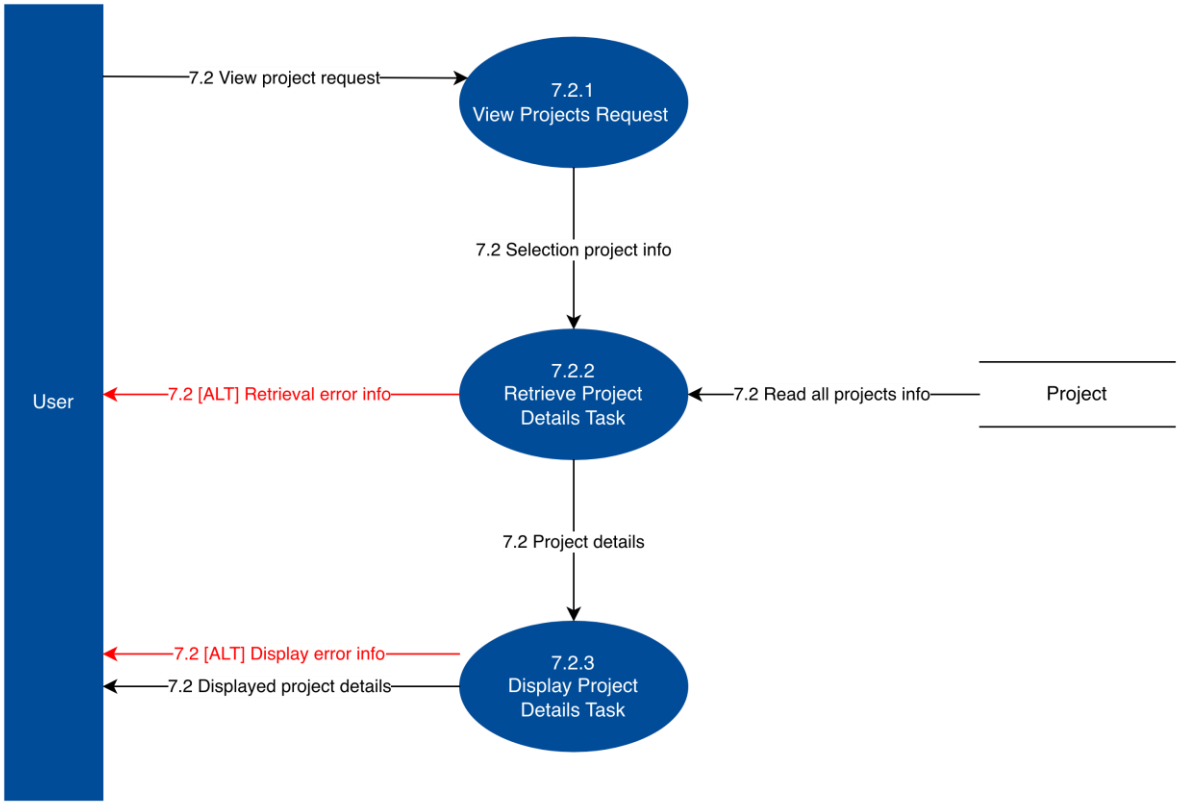


Figure 789 7.2 View Project - Primitive Data Flow Diagram

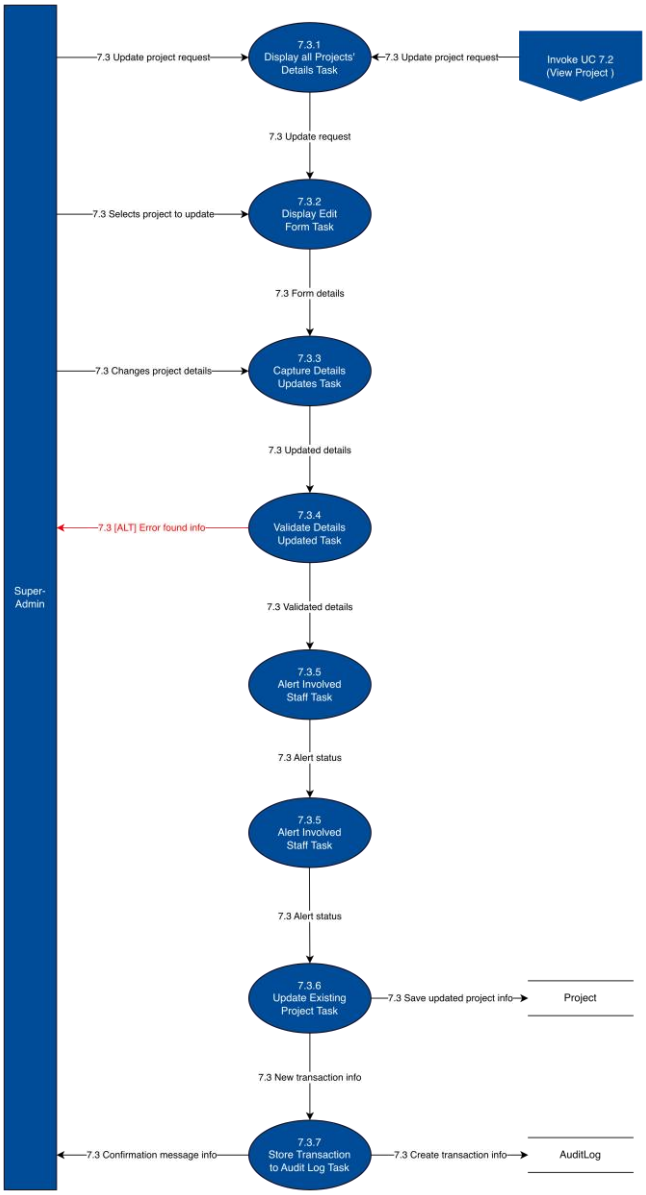


Figure 80 7.3 Update Project - Primitive Data Flow Diagram

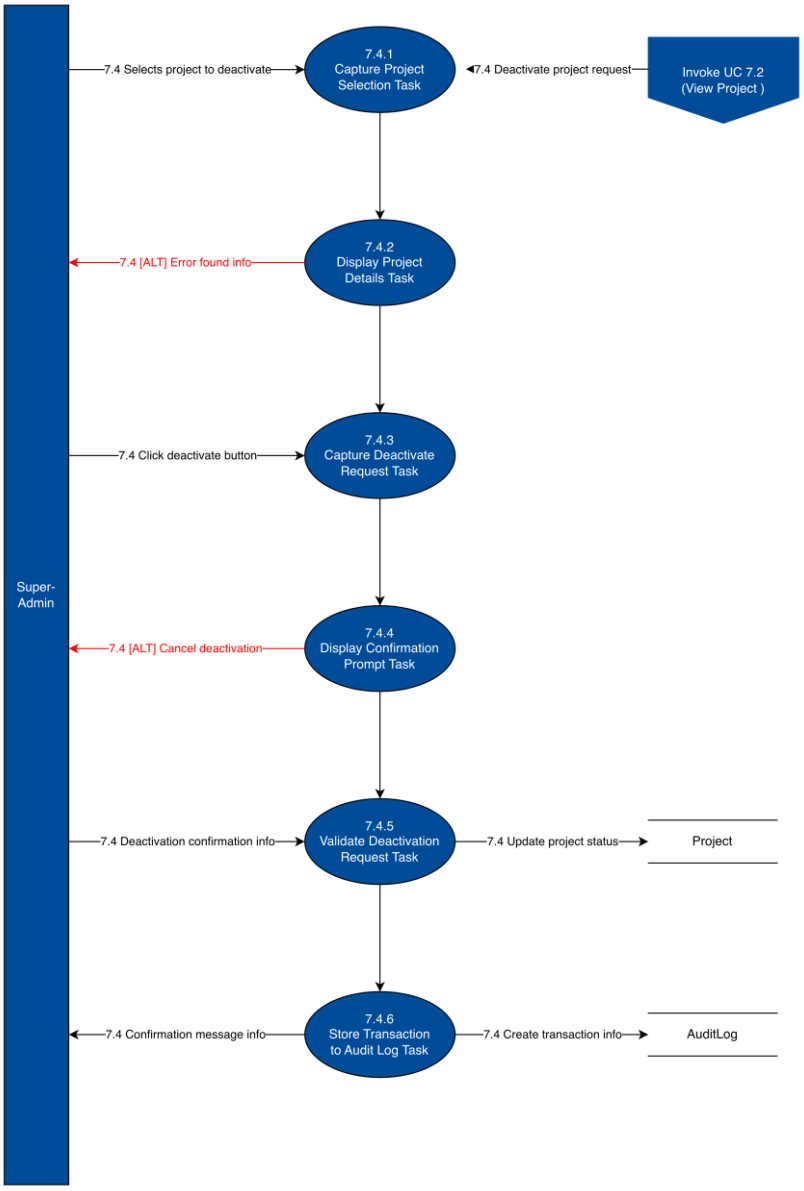


Figure 81 7.4 Deactivate Project - Primitive Data Flow Diagram

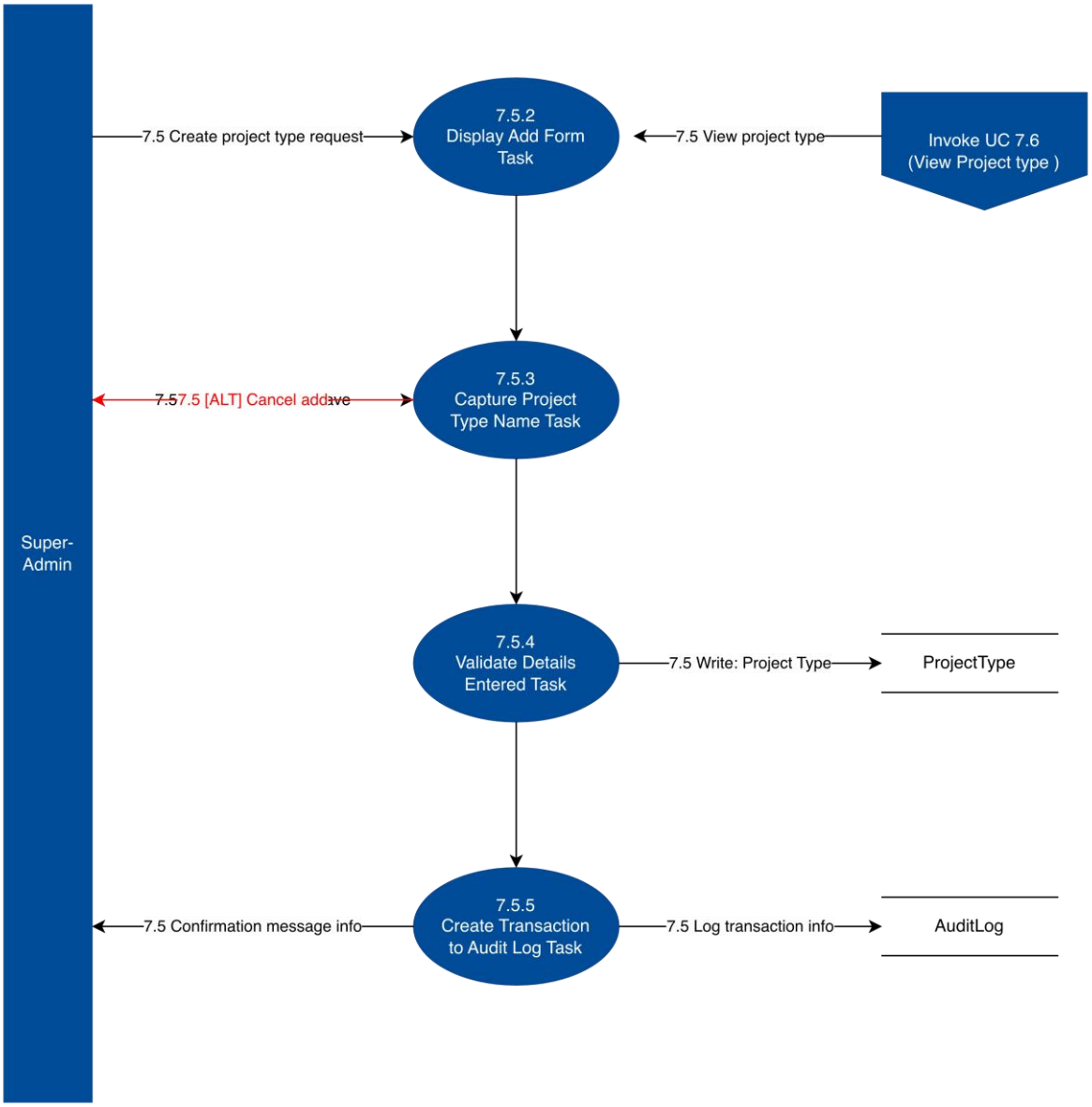


Figure 82 7.5 Create Project Type - Primitive Data Flow Diagram

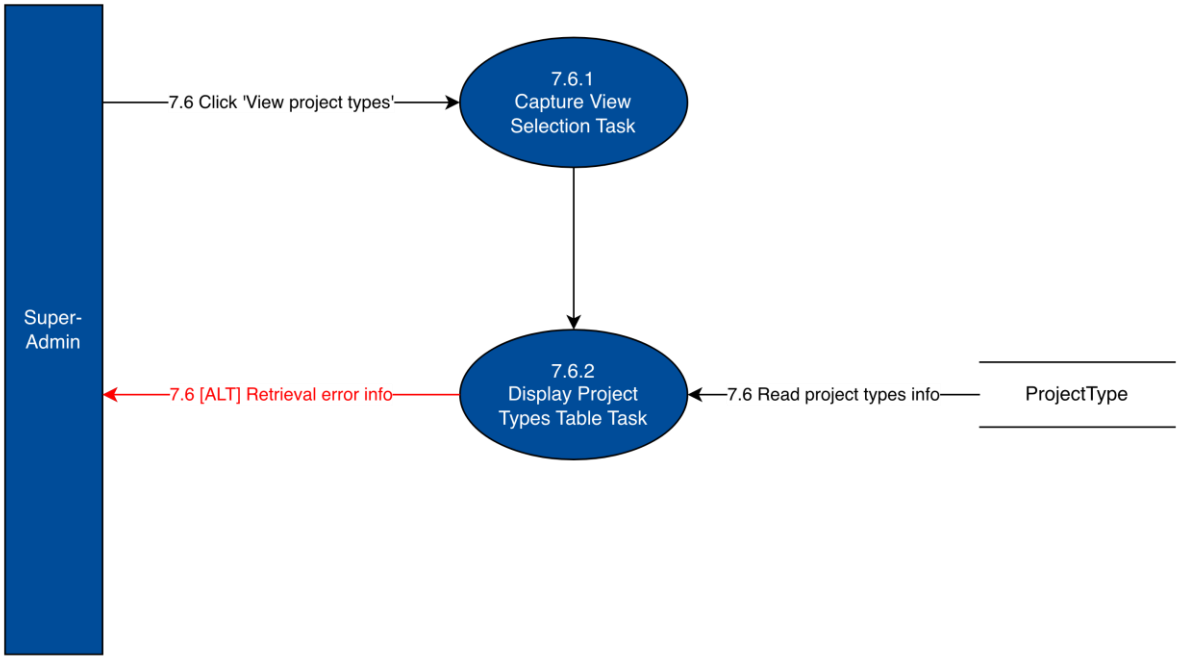


Figure 83 7.6 View Project Type - Primitive Data Flow Diagram

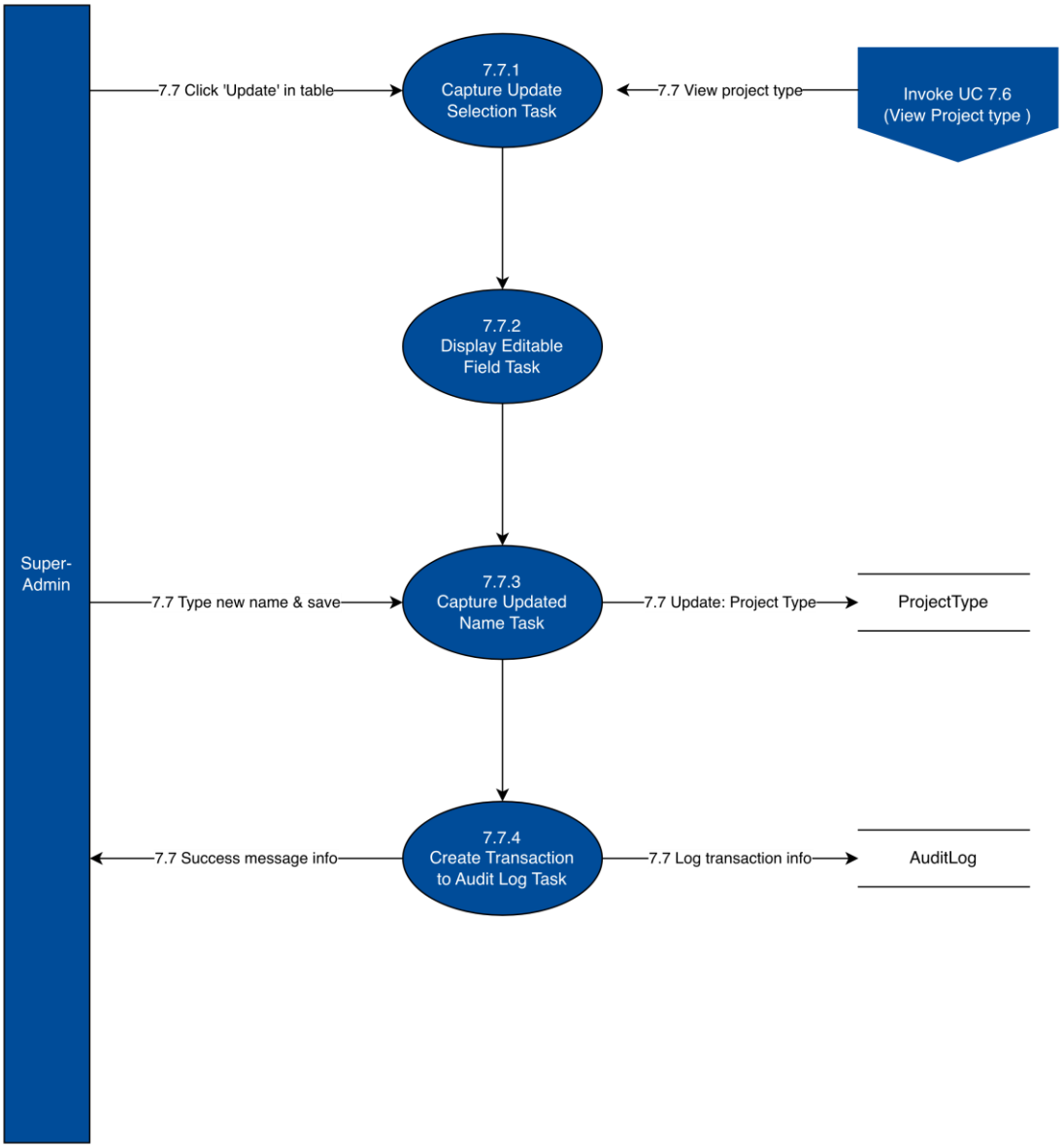


Figure 84 7.7 Update Project Type- Primitive Data Flow Diagram

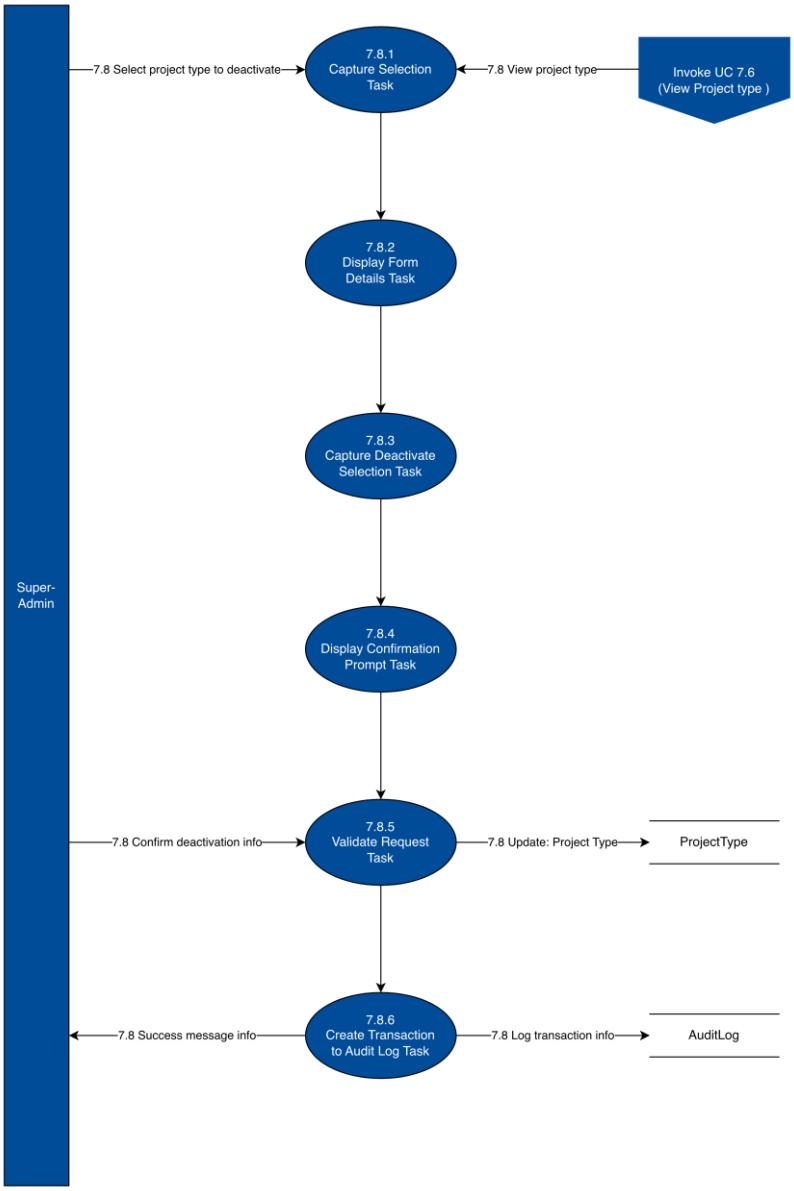


Figure 85 7.8 Deactivate Project Type - Primitive Data Flow Diagram

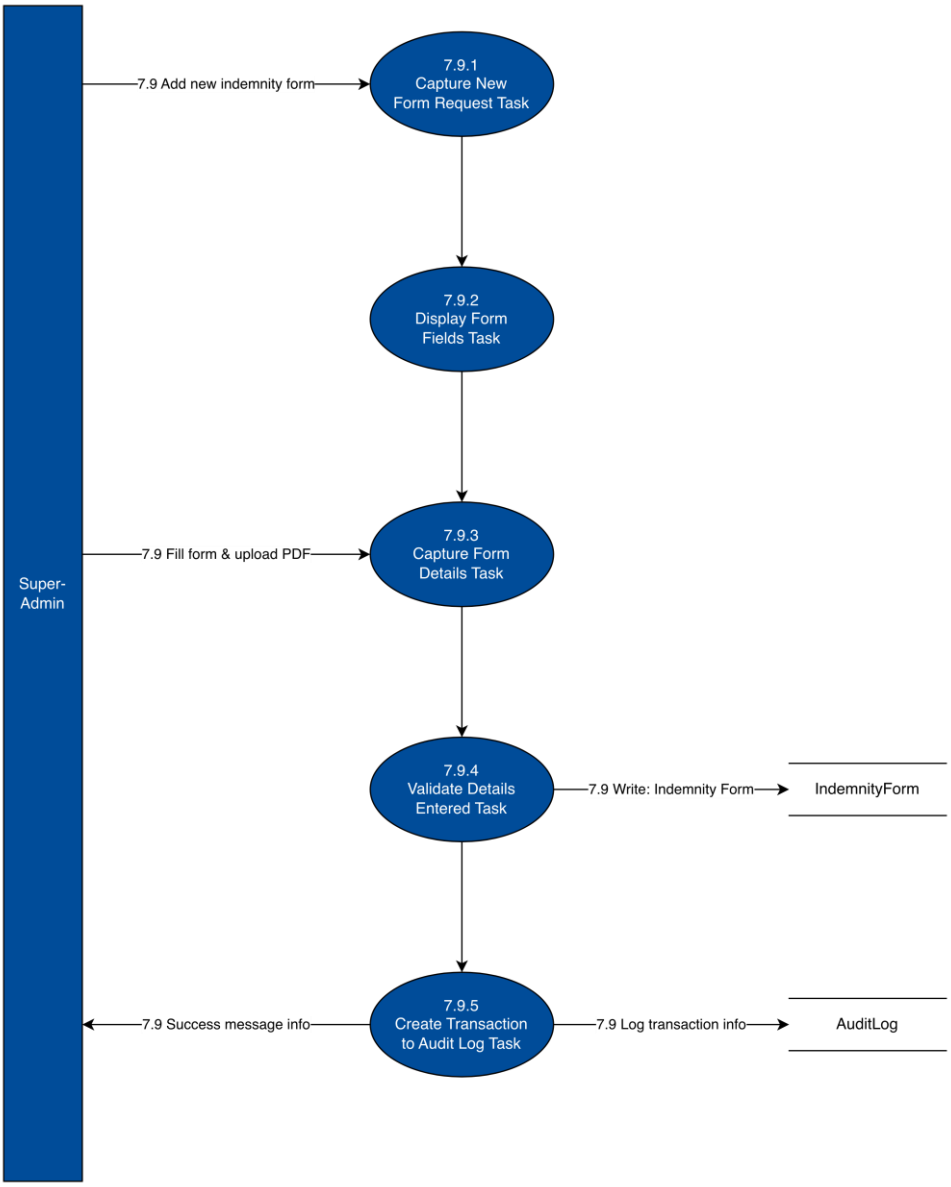


Figure 86 7.9 Add Indemnity Form - Primitive Data Flow Diagram

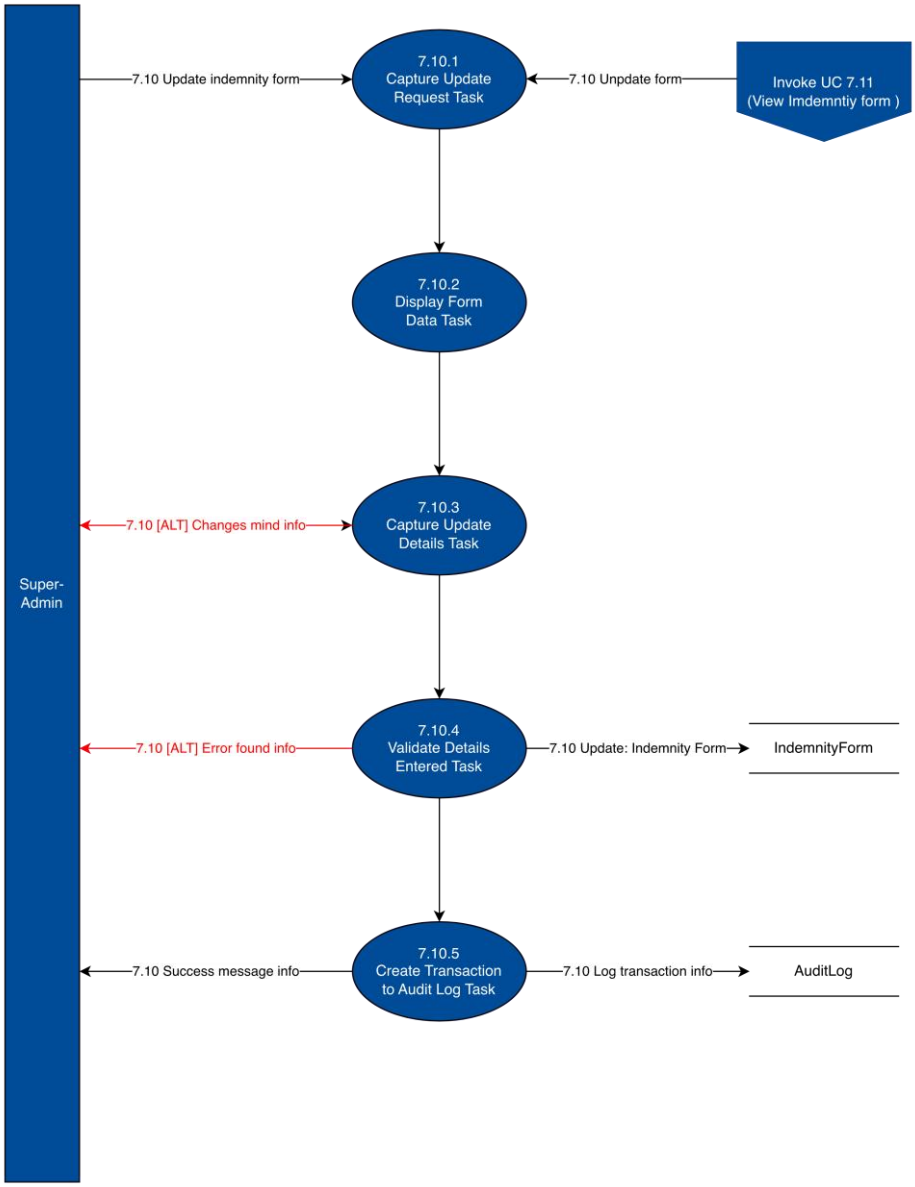


Figure 87 7.10 Update Indemnity Form - Primitive Data Flow Diagram

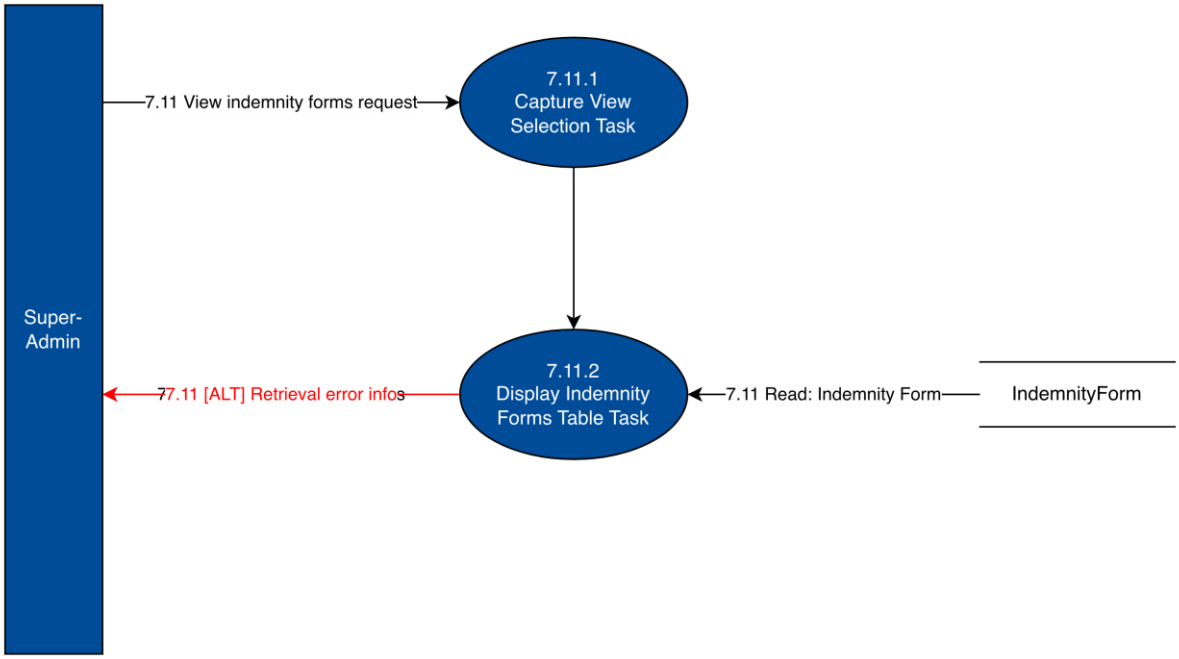


Figure 798 7.11 View Indemnity Form - Primitive Data Flow Diagram

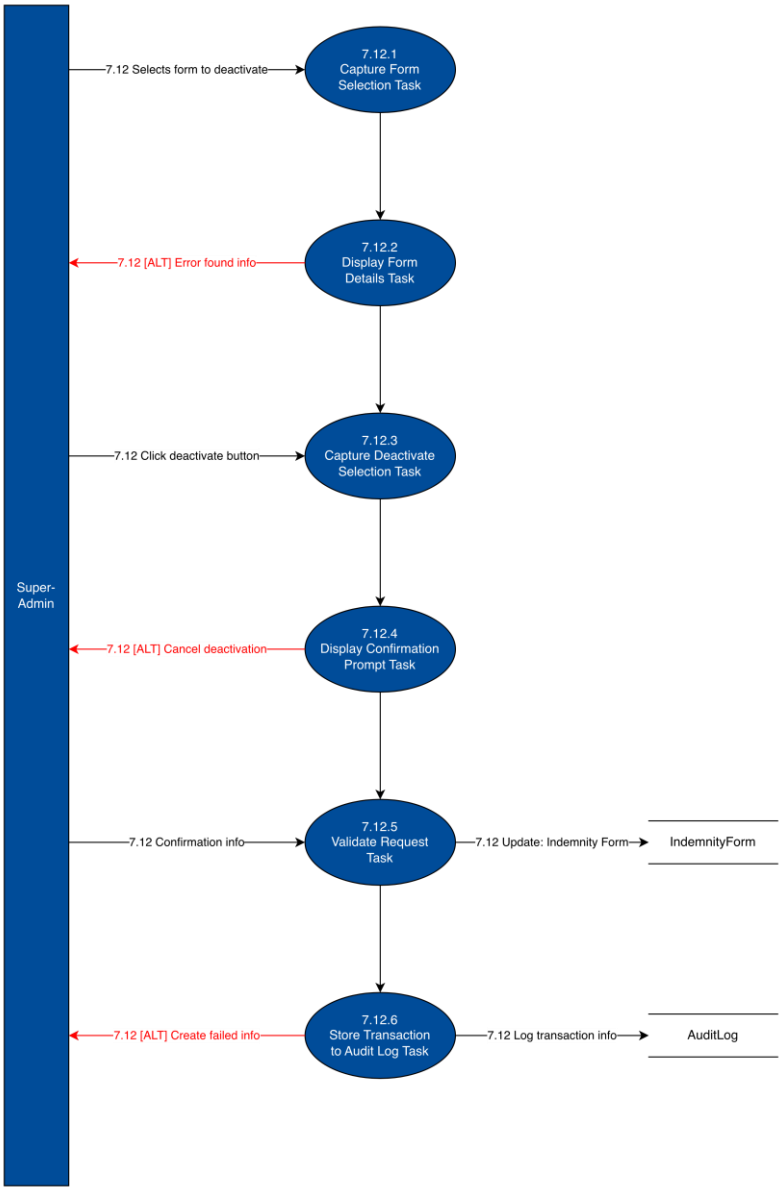


Figure 89 7.12 Remove Indemnitty Form - Primitive Data Flow Diagram

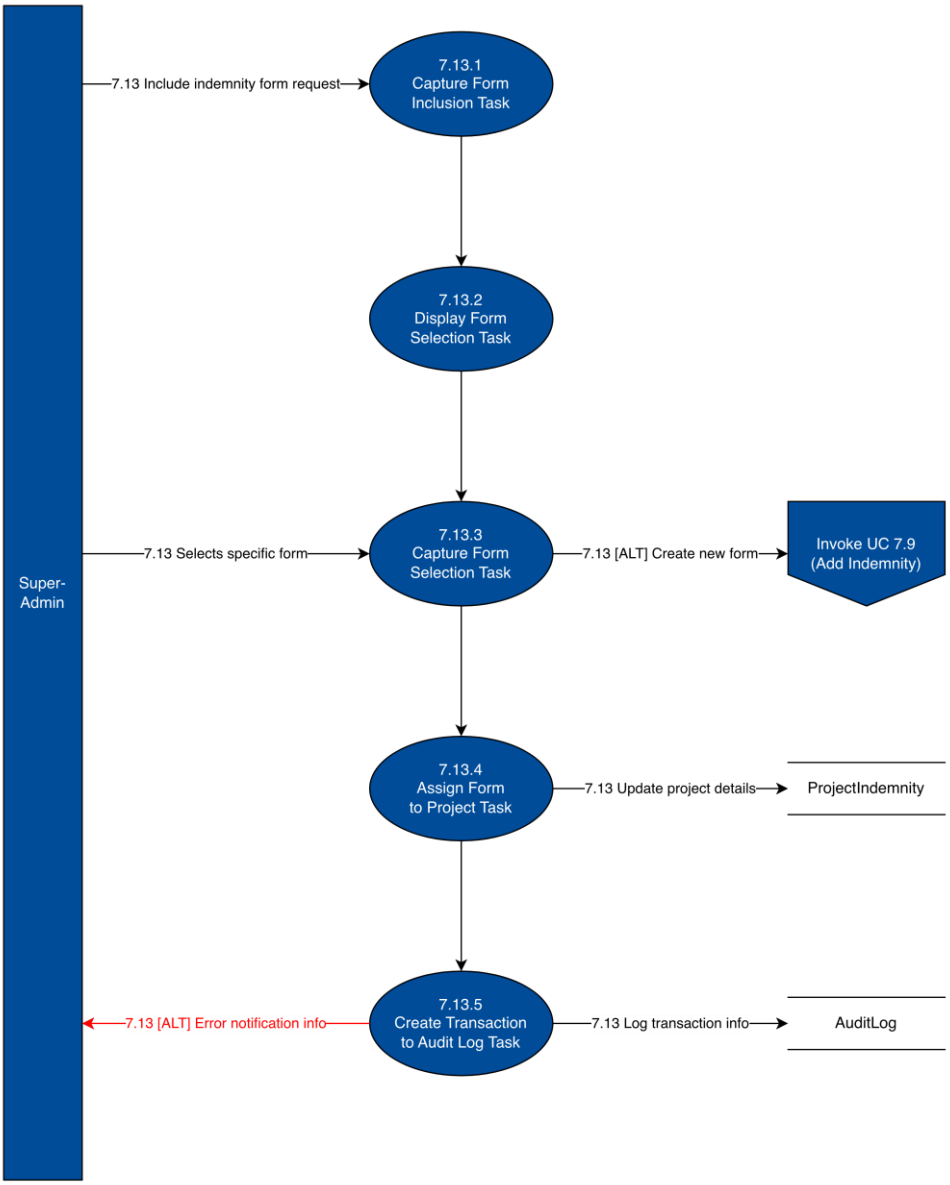


Figure 90 7.13 Capture Indemnity Form - Primitive Data Flow Diagram

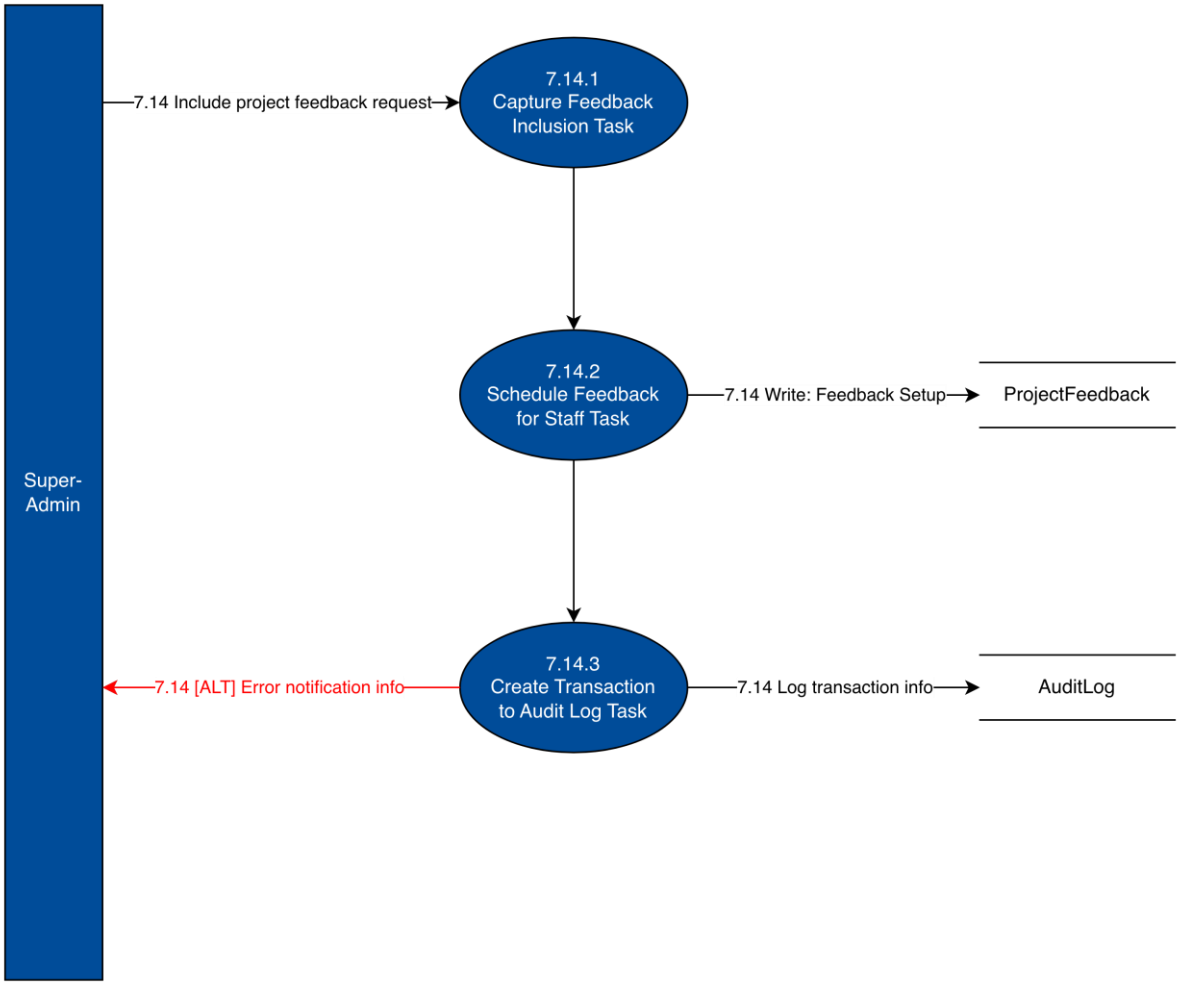


Figure 91 7.14 Capture Project Feedback - Primitive Data Flow Diagram

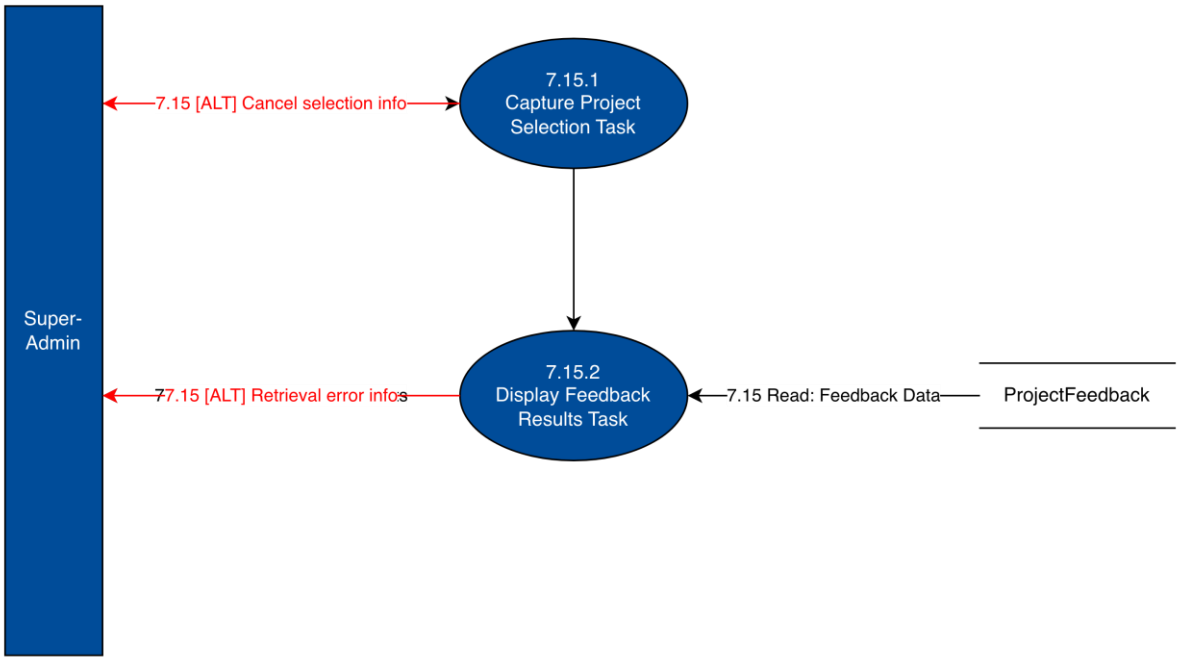


Figure 92 7.15 View Project Feedback - Primitive Data Flow Diagram

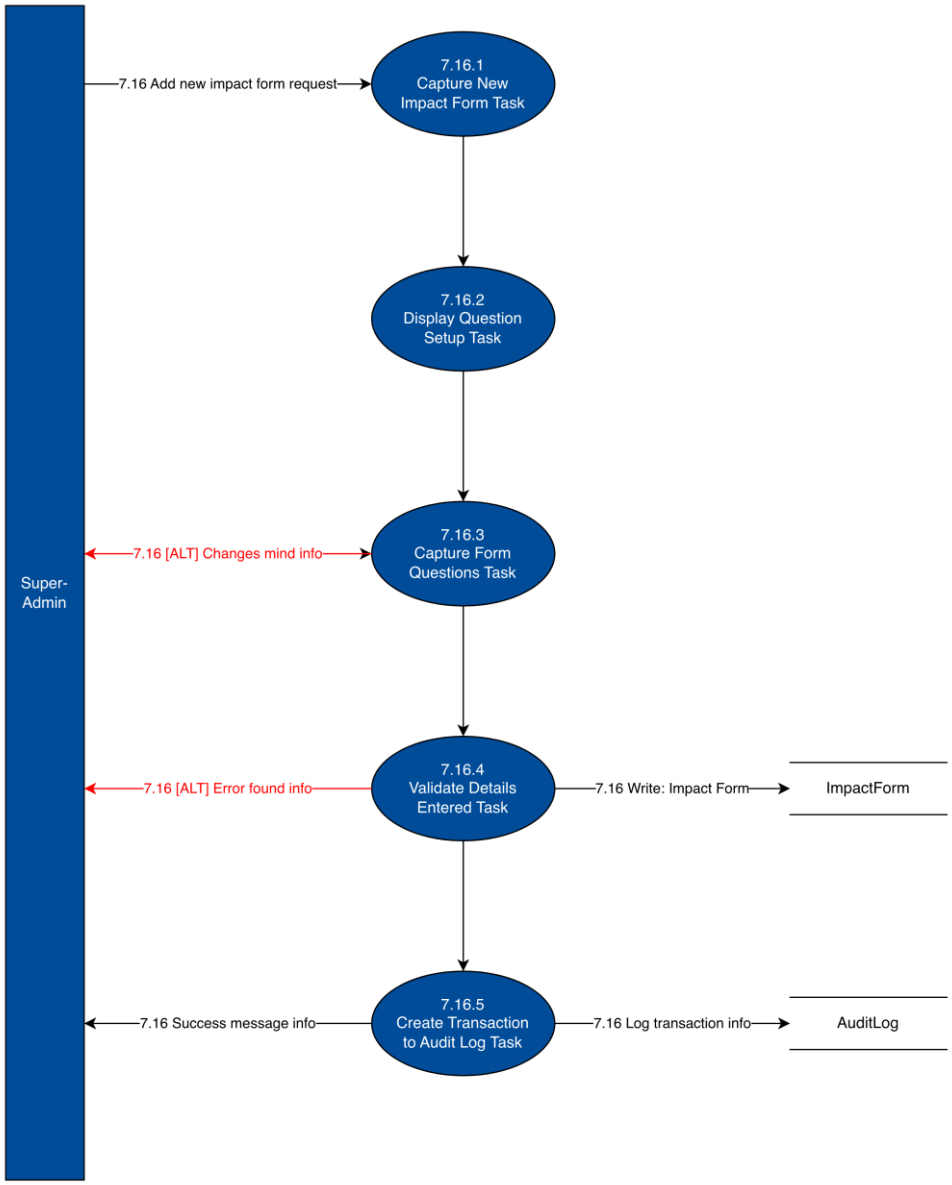


Figure 93 7.16 Create Impact Form - Primitive Data Flow Diagram

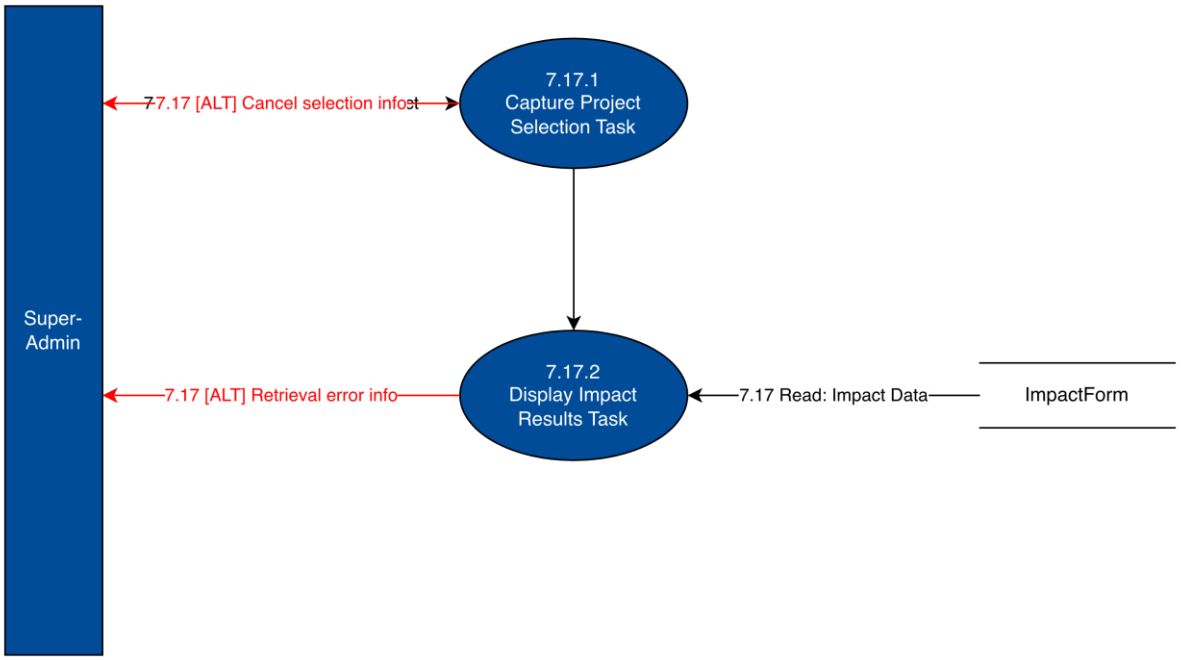


Figure 94 7.17 View Impact Form - Primitive Data Flow Diagram

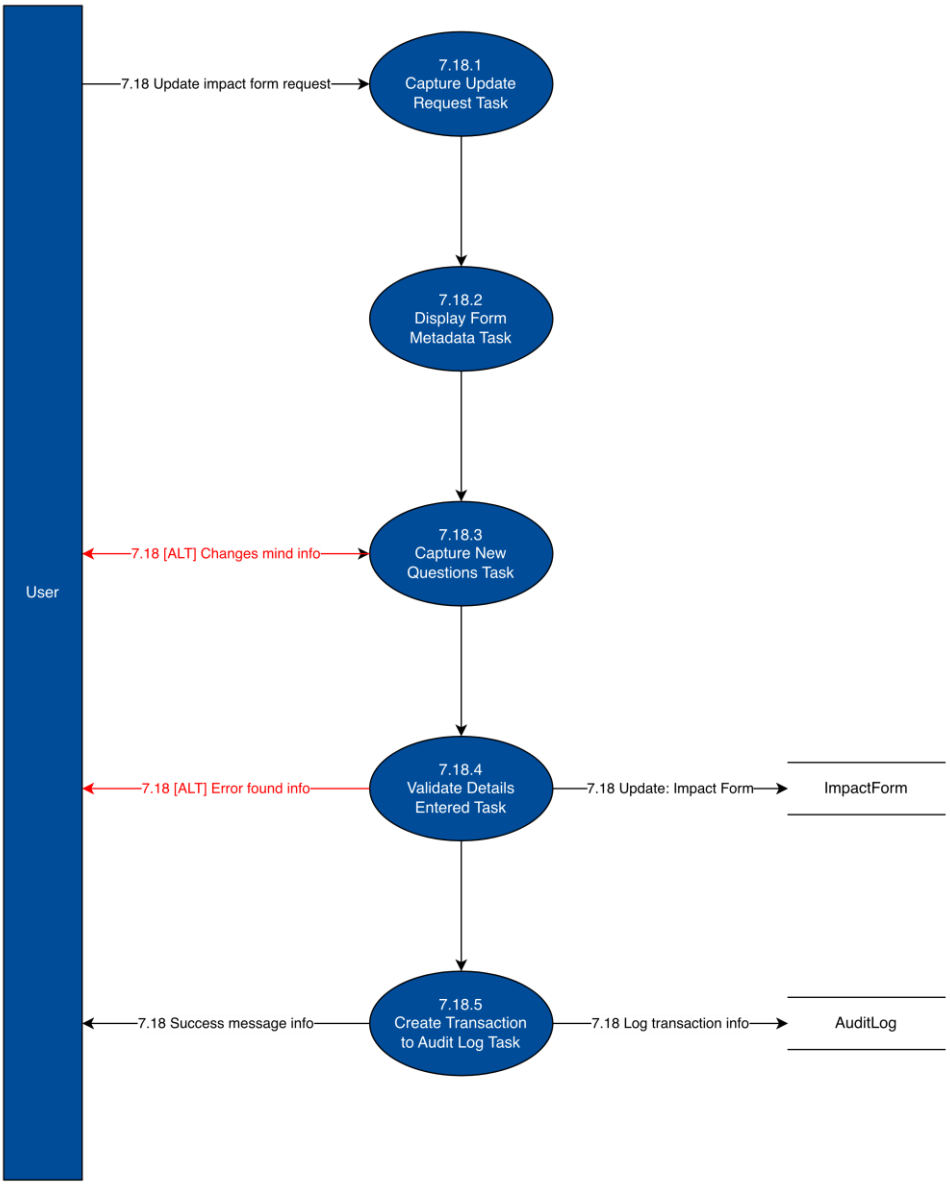


Figure 95 7.18 Update Impact Form - Primitive Data Flow Diagram

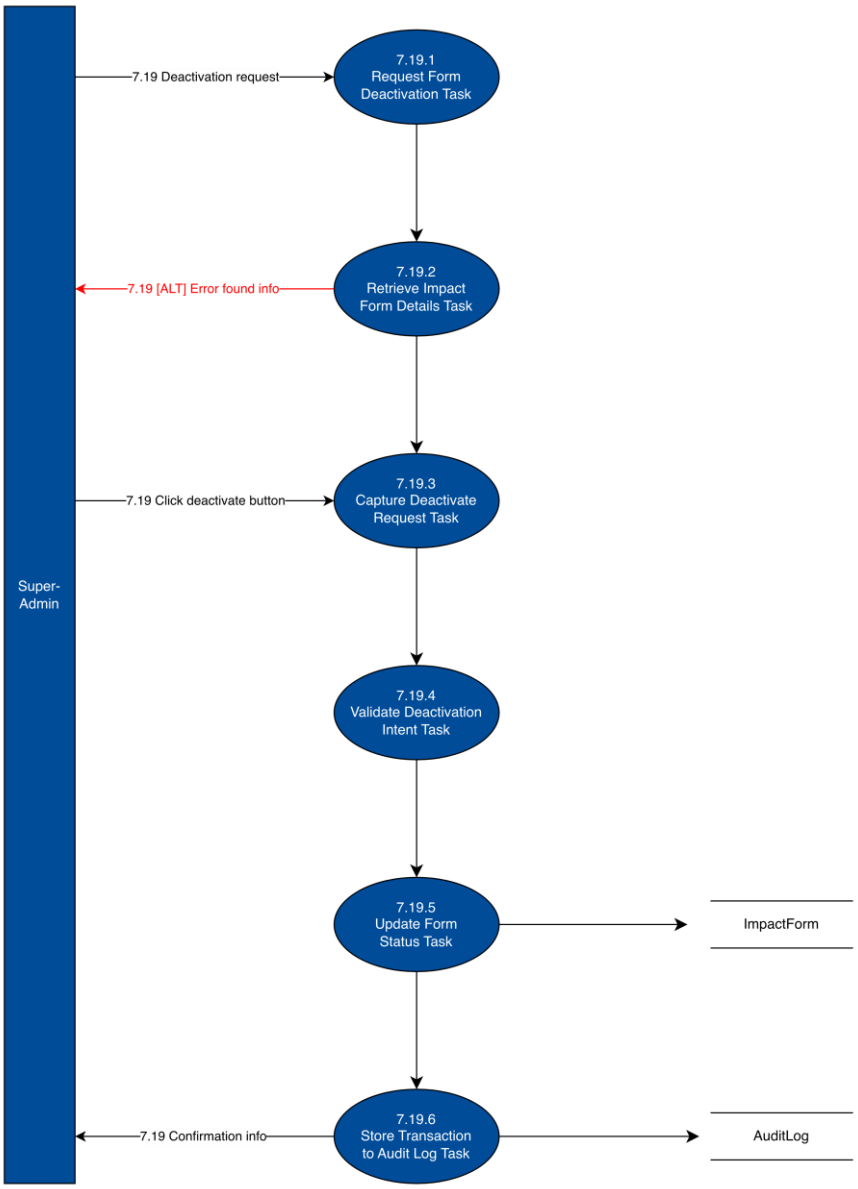


Figure 96 7.19 Deactivate Impact Form - Primitive Data Flow Diagram

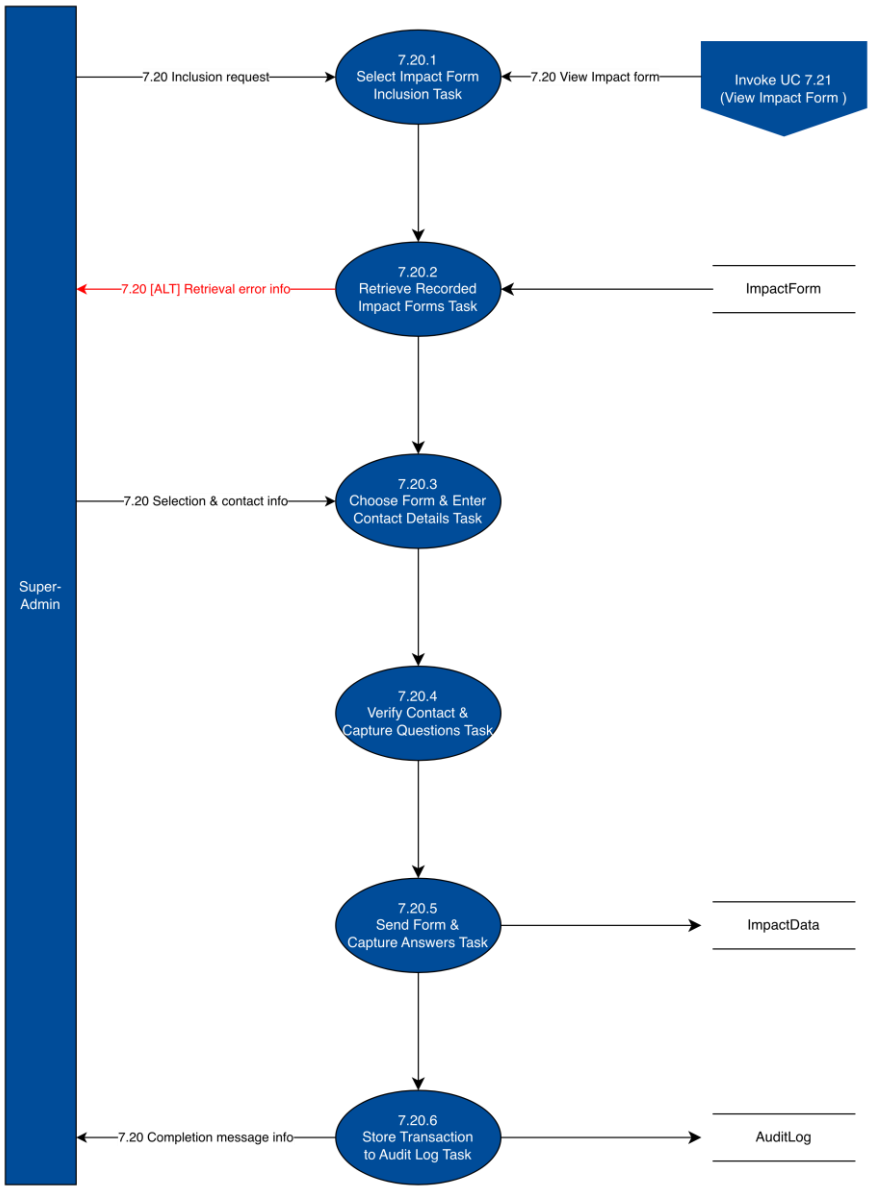


Figure 97 7.20 Capture Impact Form - Primitive Data Flow Diagram

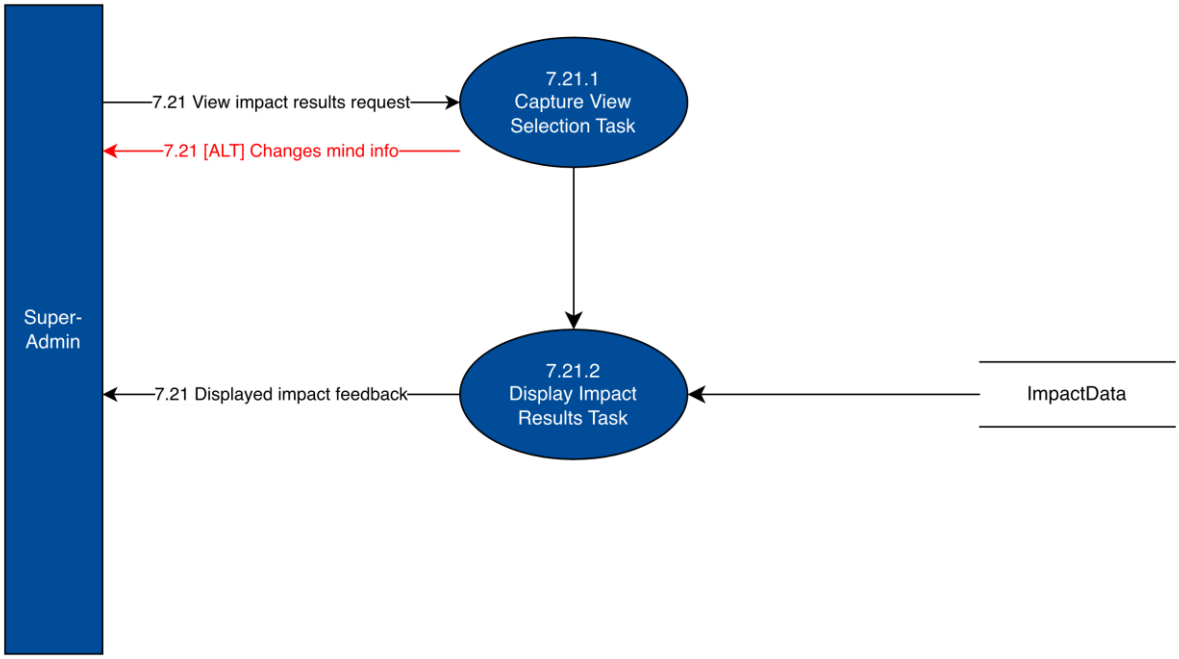


Figure 98 7.21 View Impact Results - Primitive Data Flow Diagram

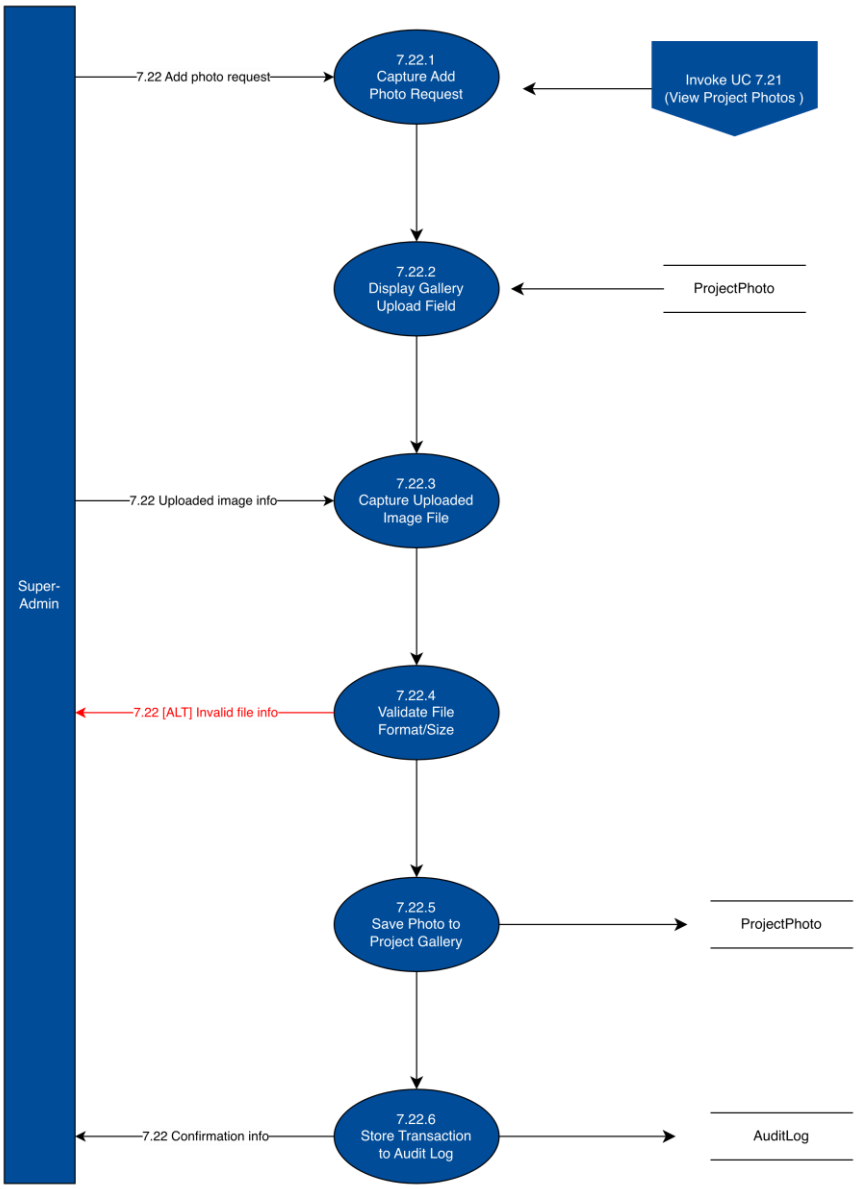


Figure 99 7.22 Add Project Photo - Primitive Data Flow Diagram

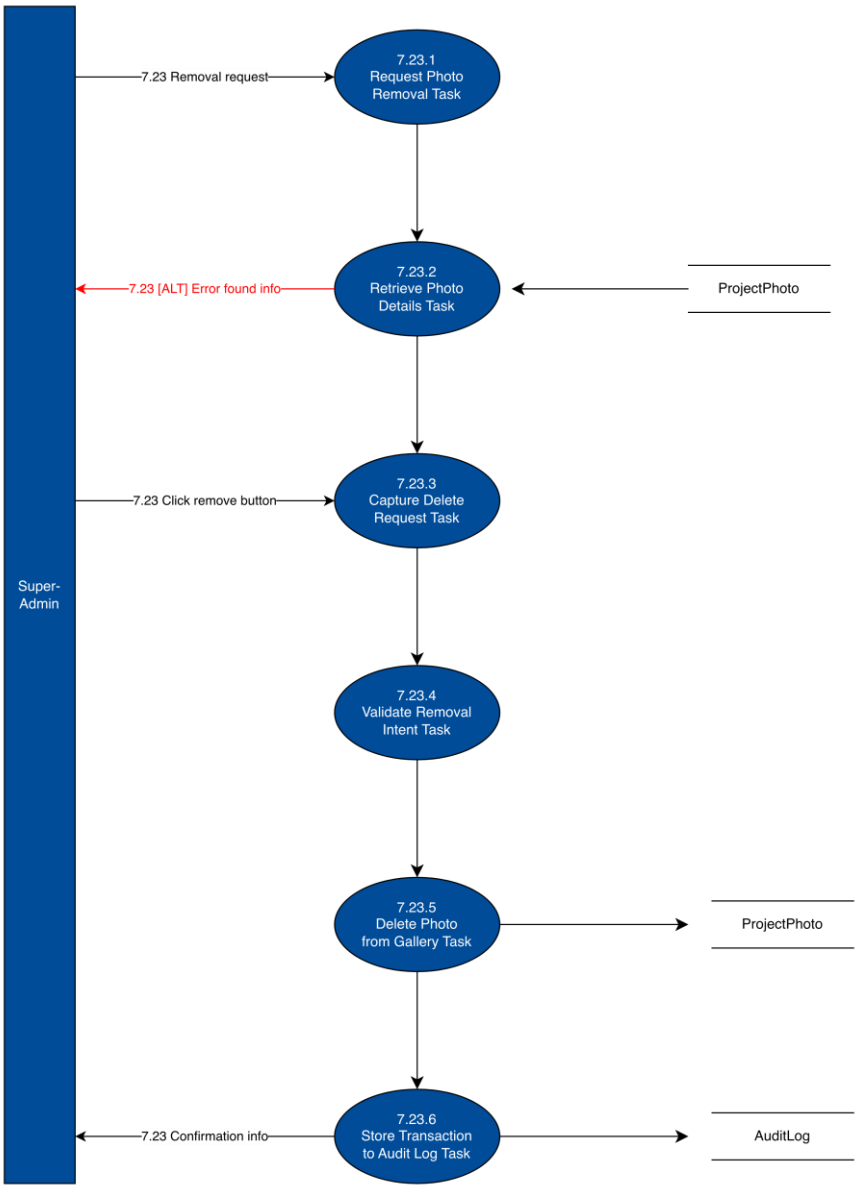


Figure 100 7.23 Remove Project Photo - Primitive Data Flow Diagram

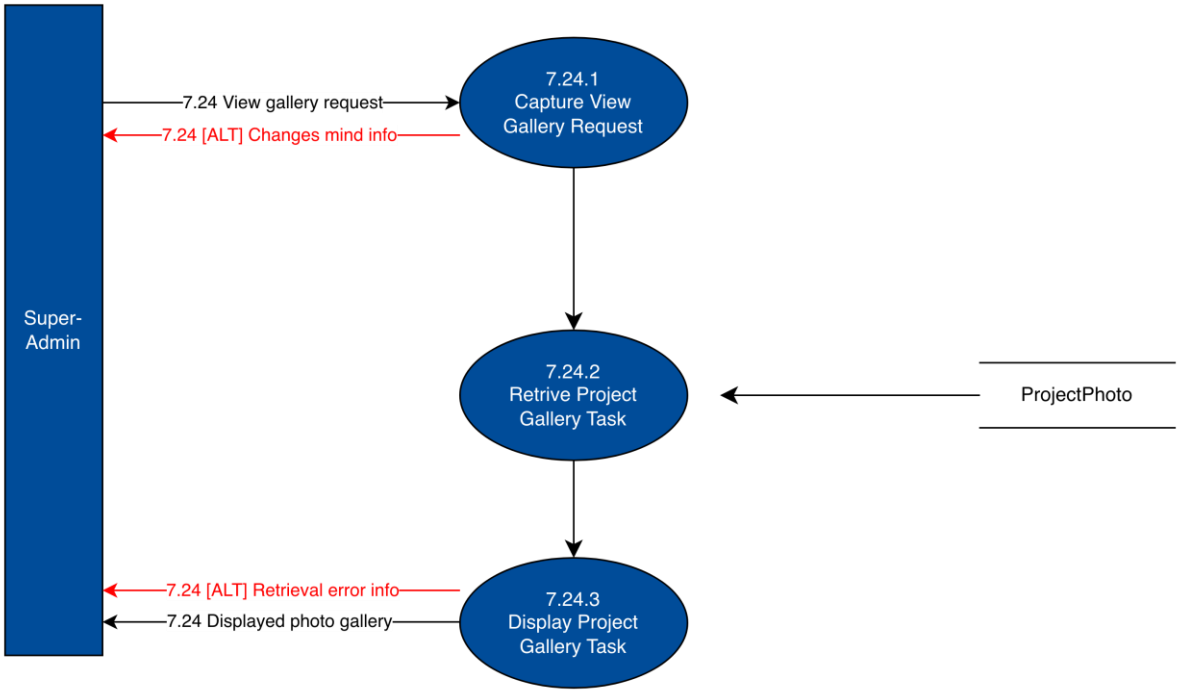


Figure 101 7.24 View Photo - Primitive Data Flow Diagram

2.9.8 K9 Subsystem Primitives

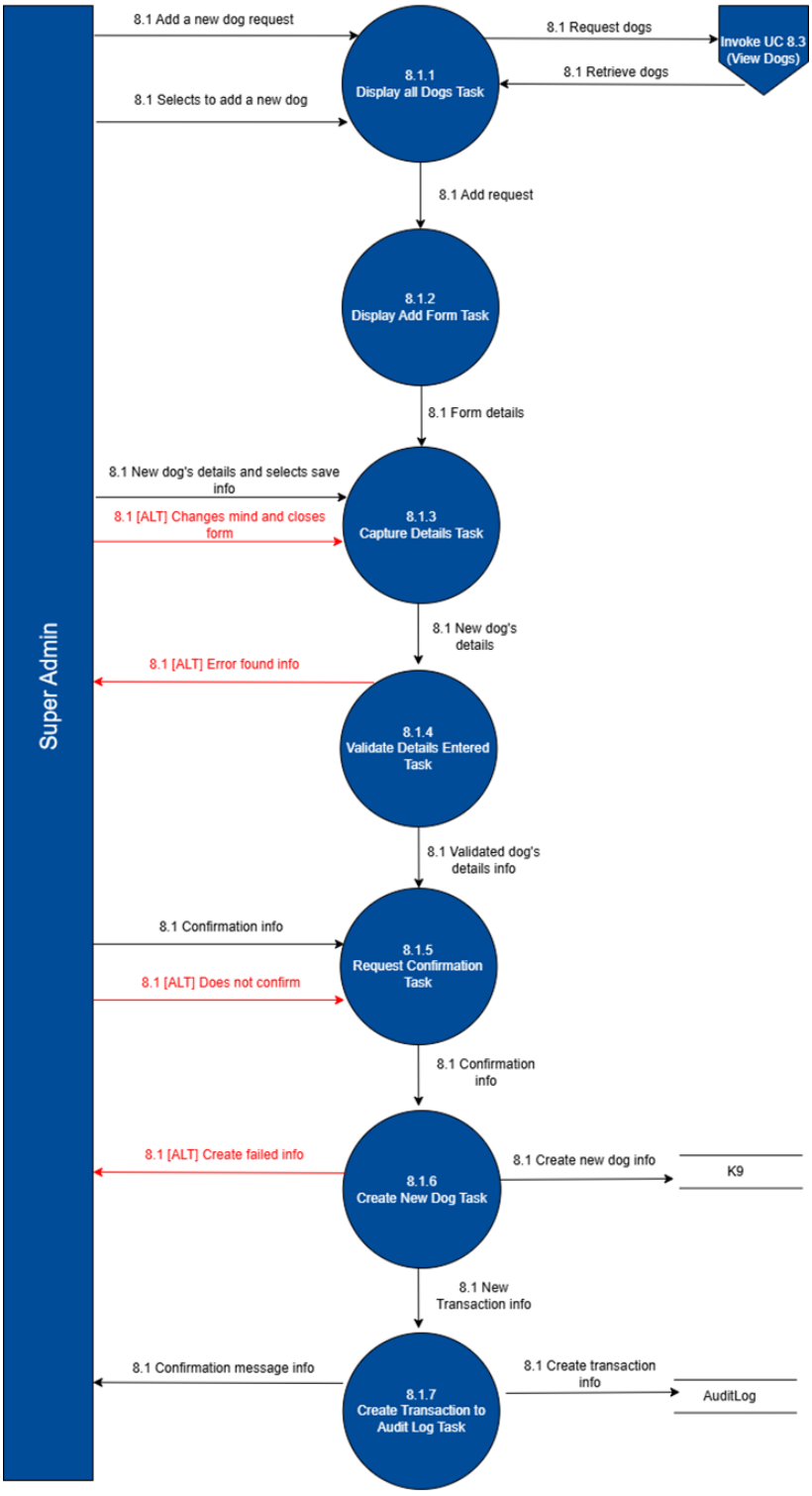


Figure 80 8.1 Add Dog - Primitive Data Flow Diagram

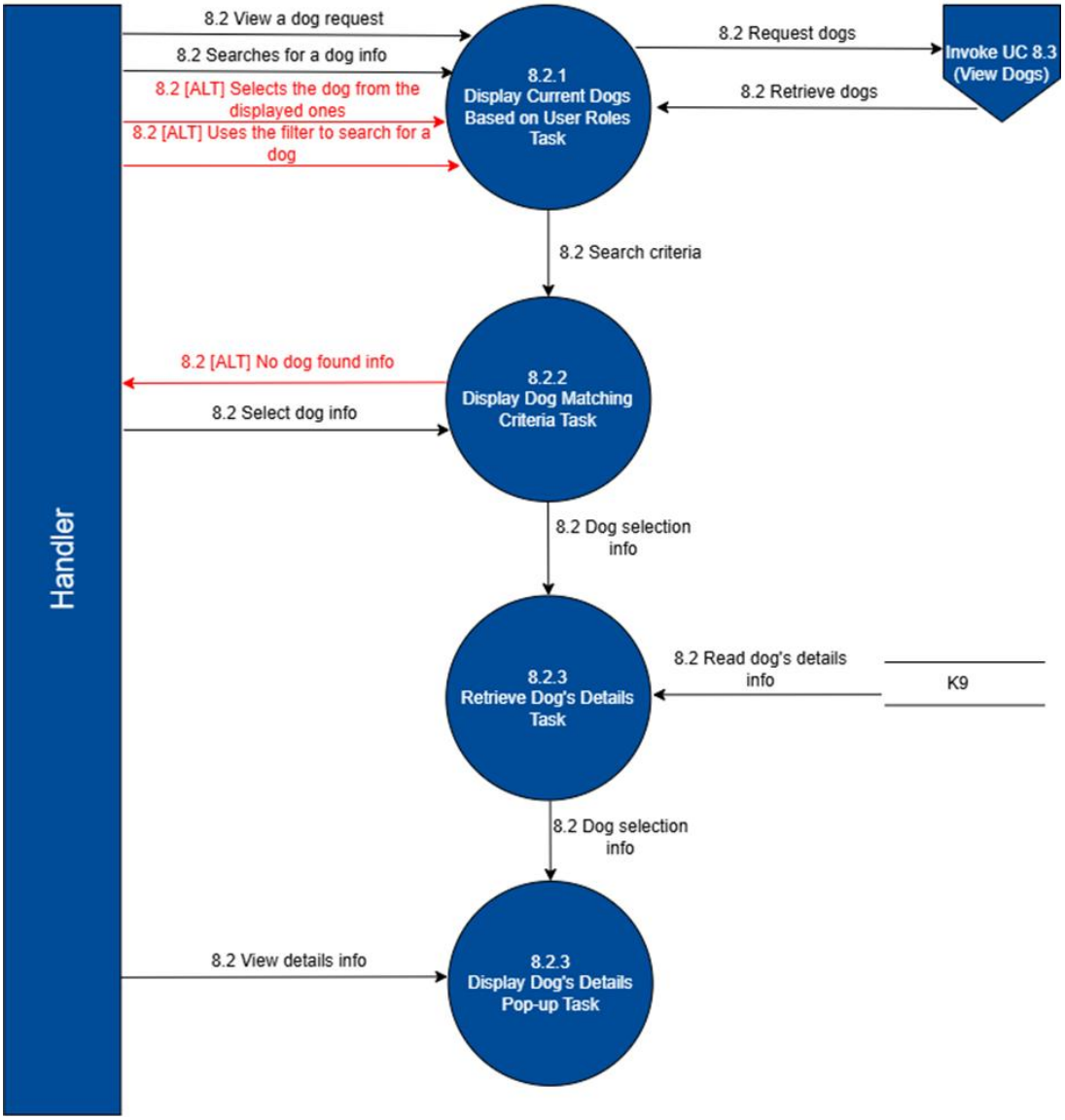


Figure 81 8.2 View Dog - Primitive Data Flow Diagram

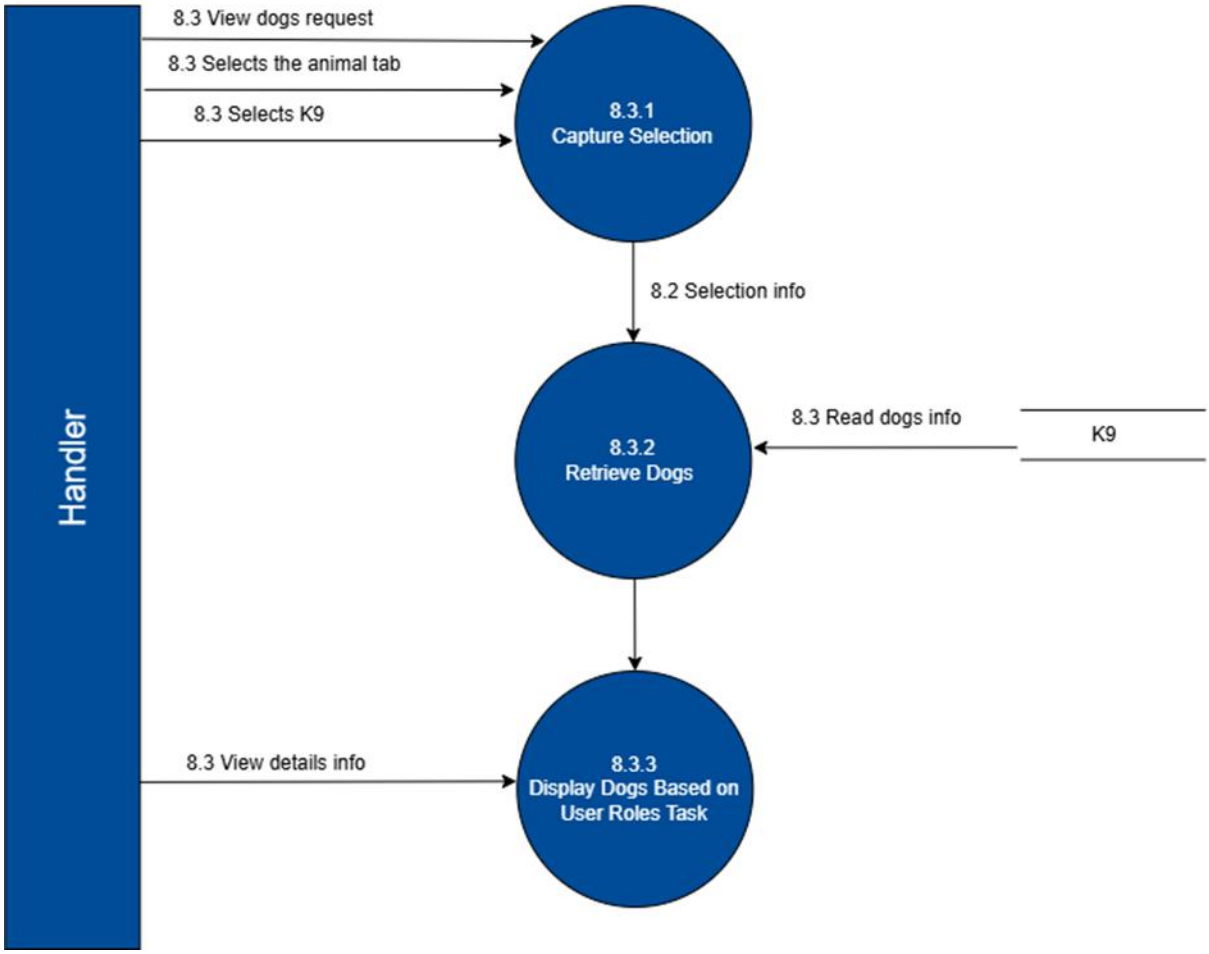


Figure 82 8.3 View Dogs - Primitive Data Flow Diagram

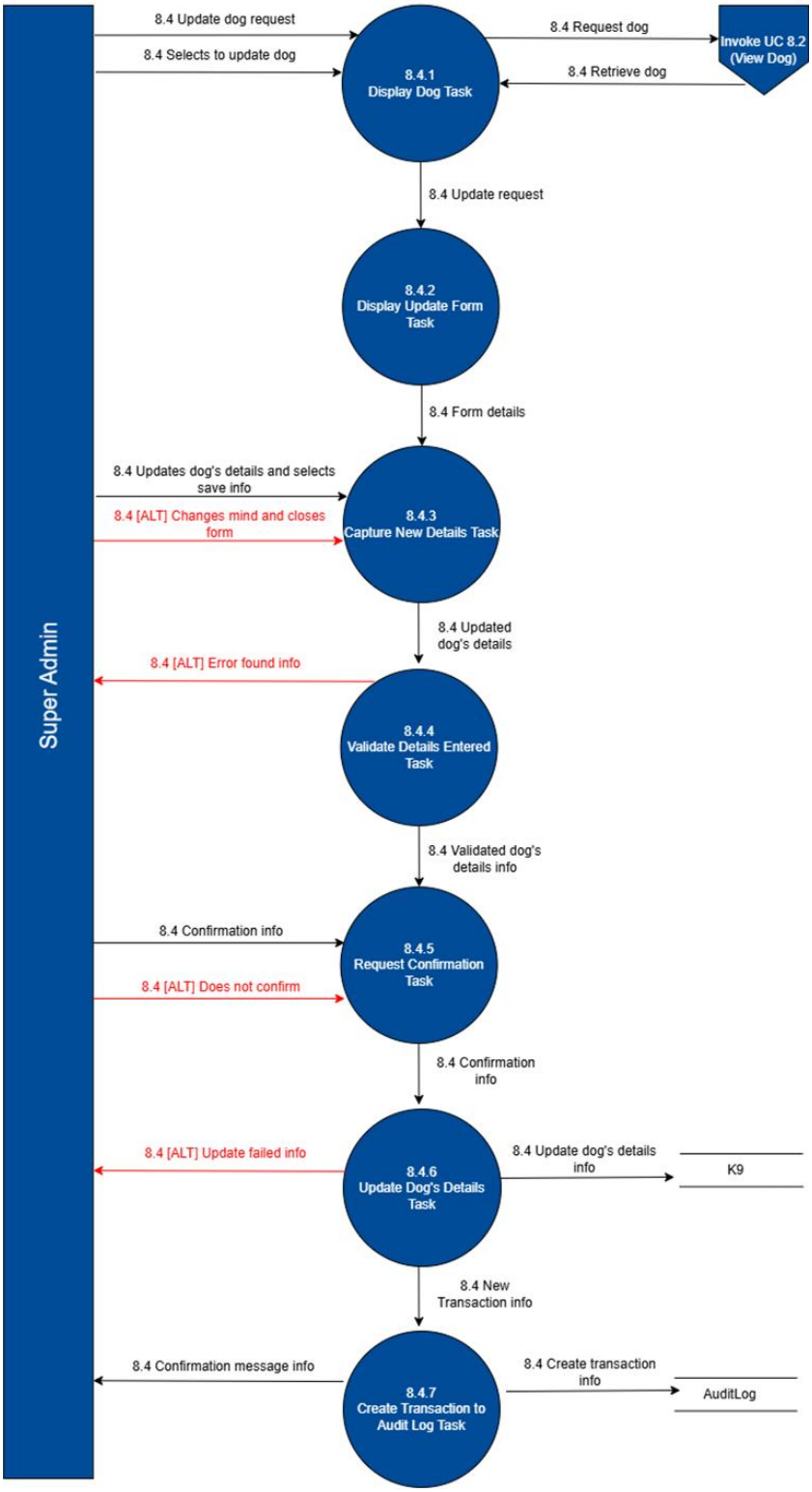


Figure 83 8.4 Update Dog - Primitive Data Flow Diagram

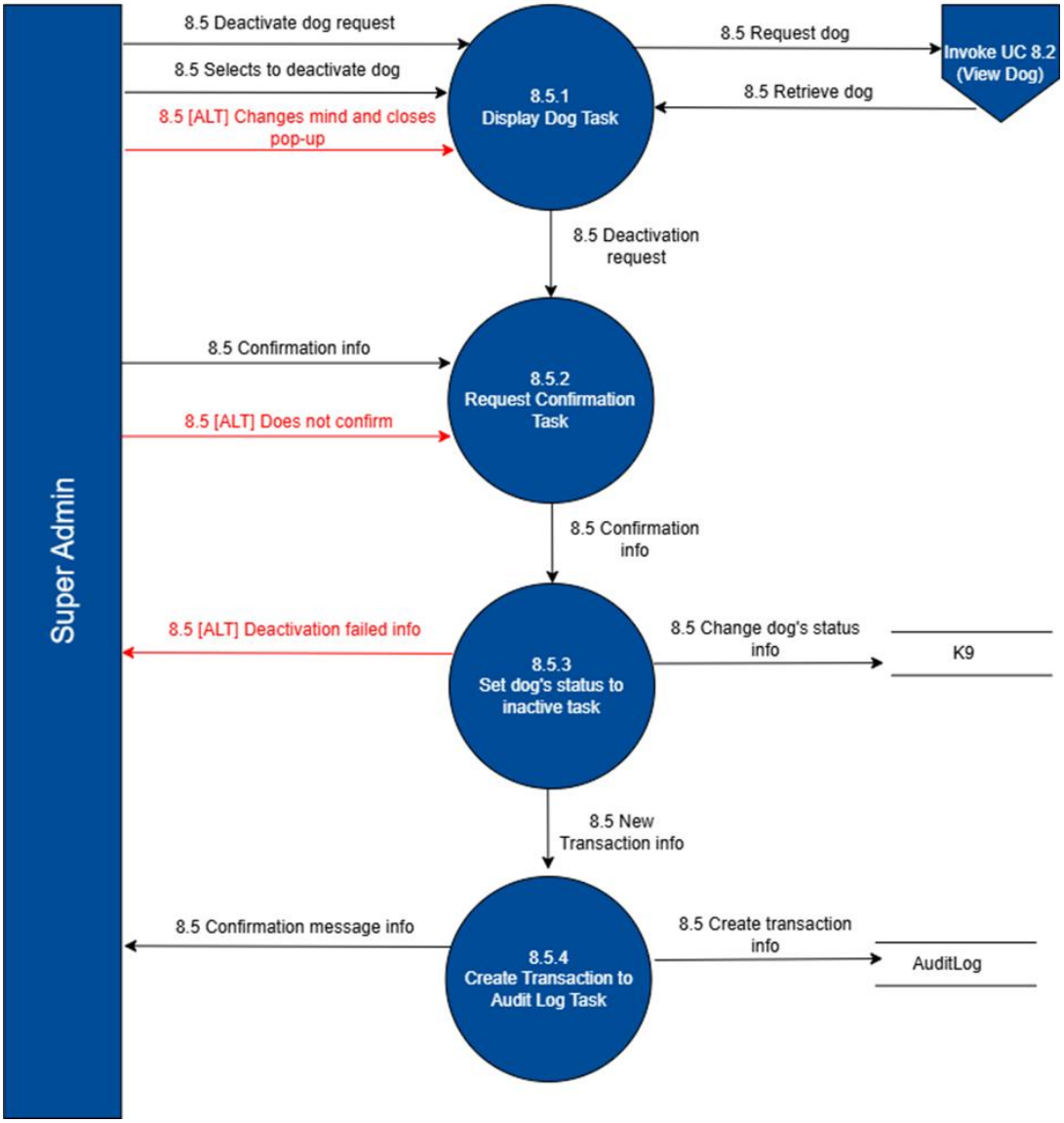


Figure 84 8.5 Deactivate Dog - Primitive Data Flow Diagram

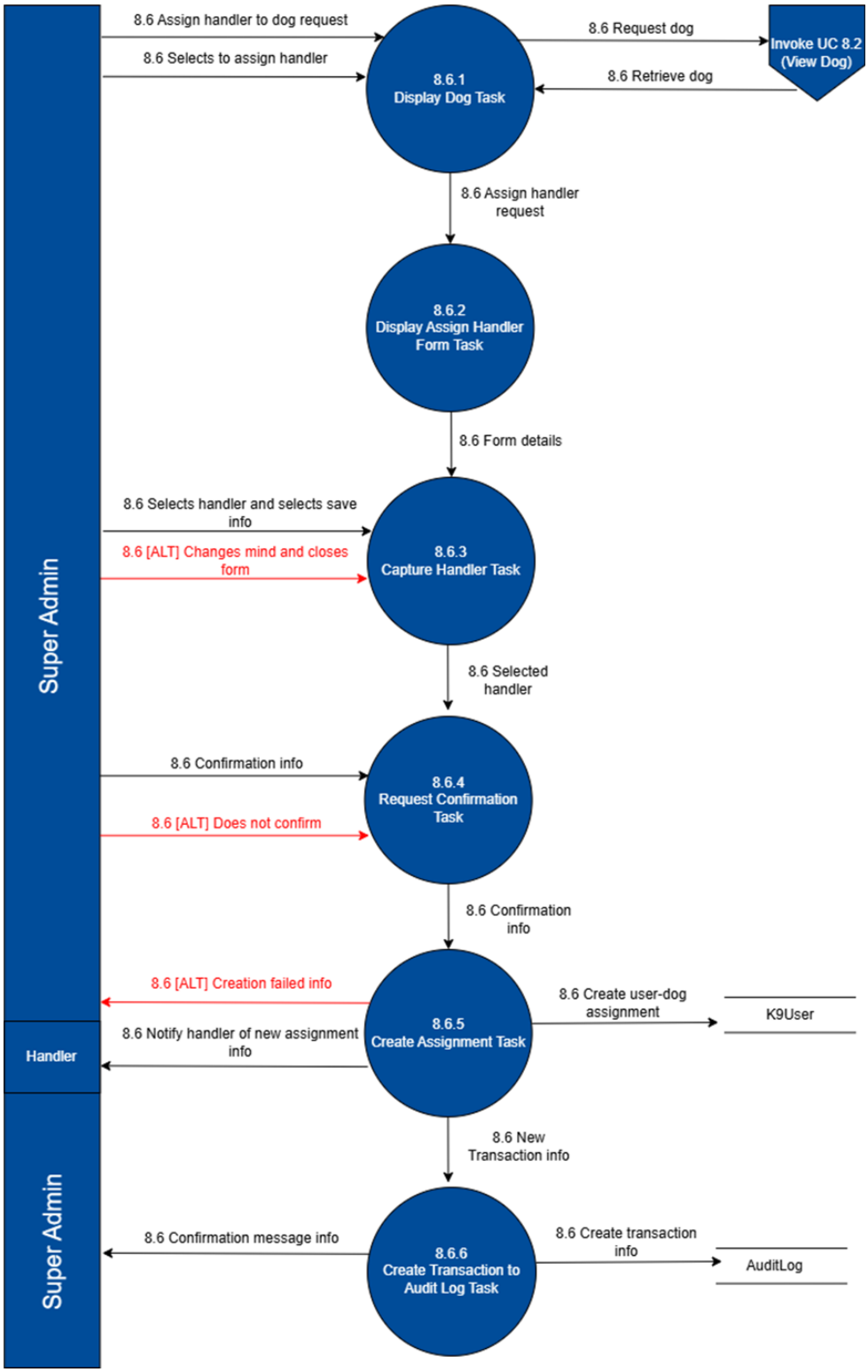


Figure 85 8.6 Assign User to Dog - Primitive Data Flow Diagram

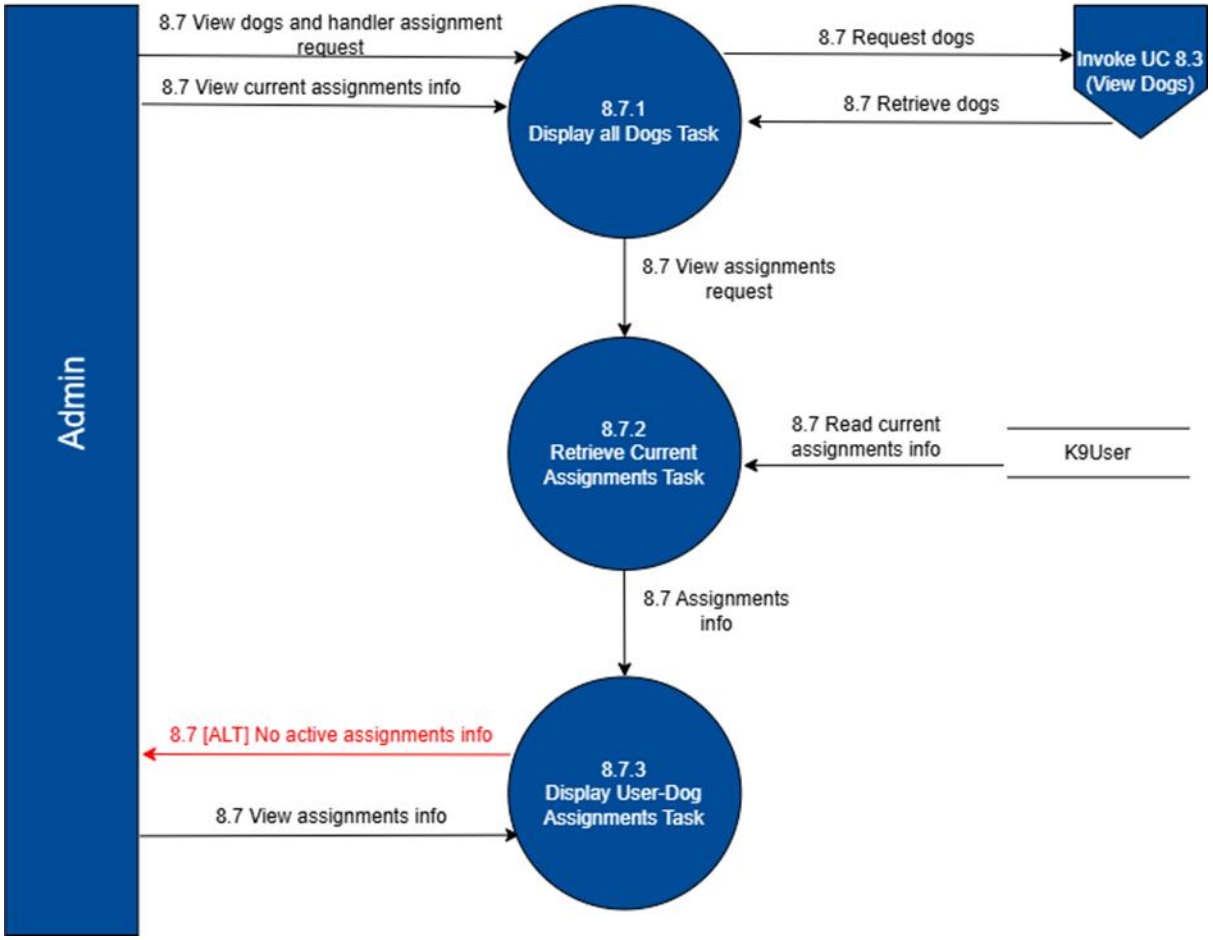


Figure 86 8.7 View User-Dog Assignment - Primitive Data Flow Diagram

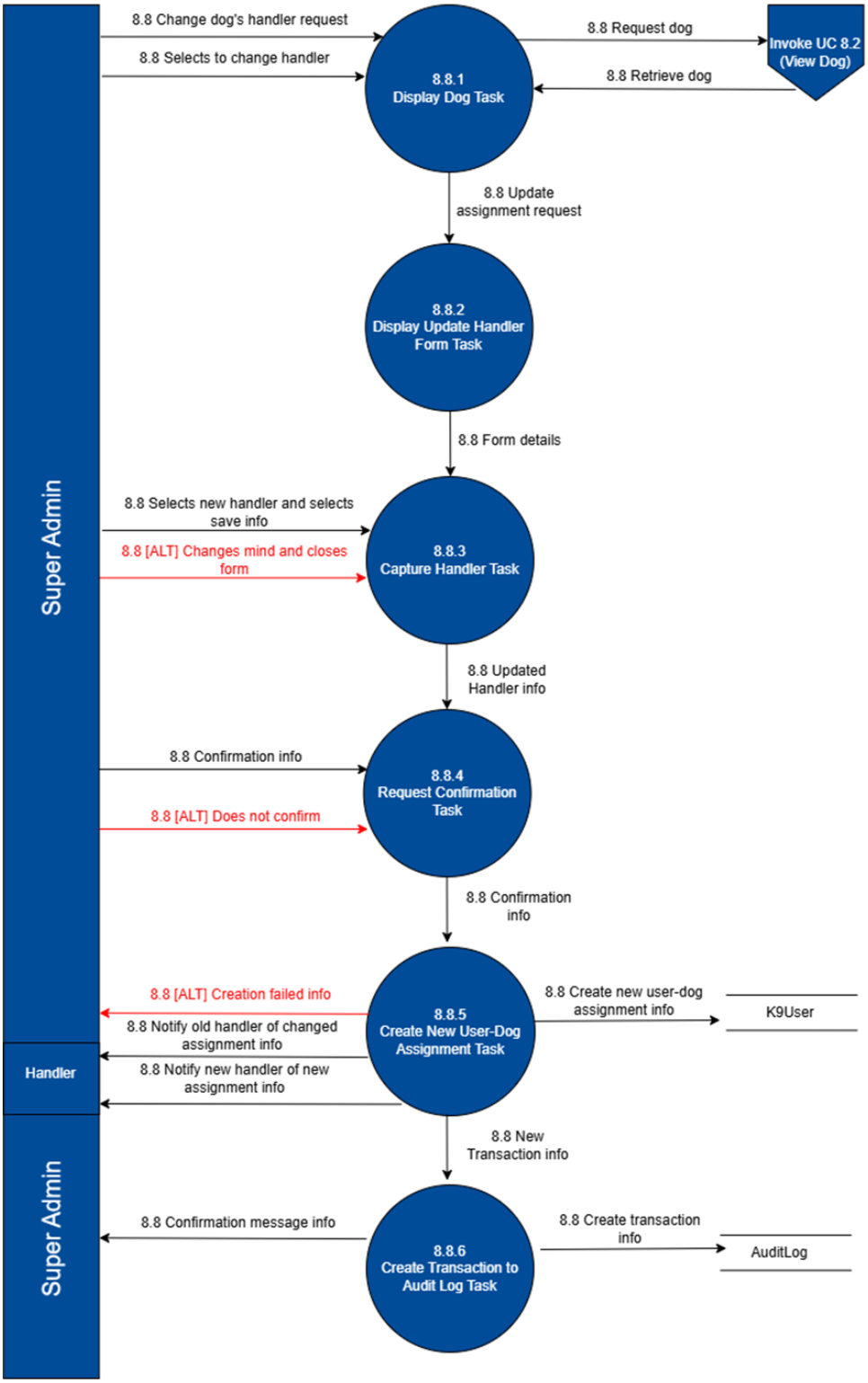


Figure 87 8.8 Update User-Dog Assignment - Primitive Data Flow Diagram

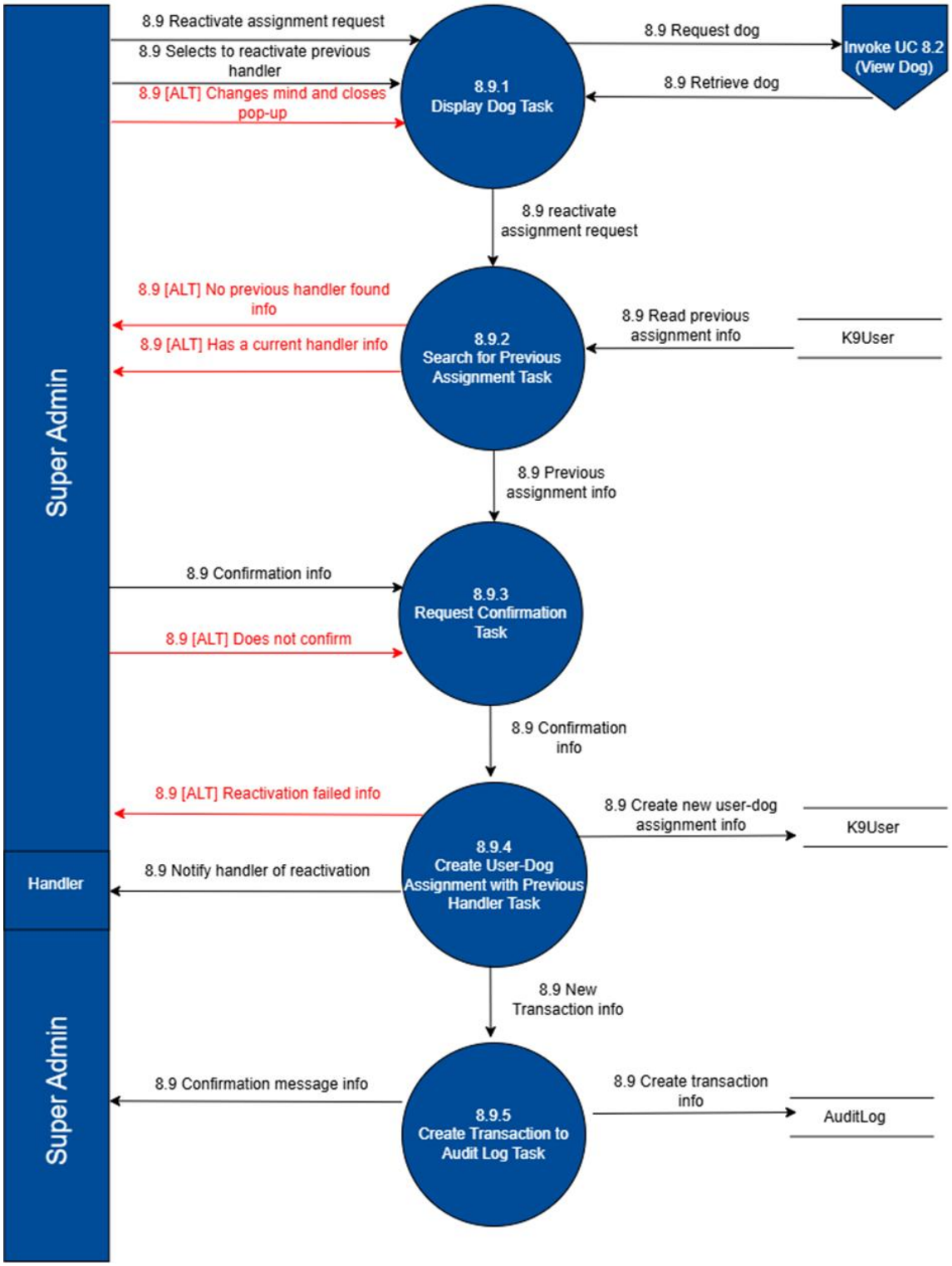


Figure 88 8.9 Reactivate User-Dog Assignment - Primitive Data Flow Diagram

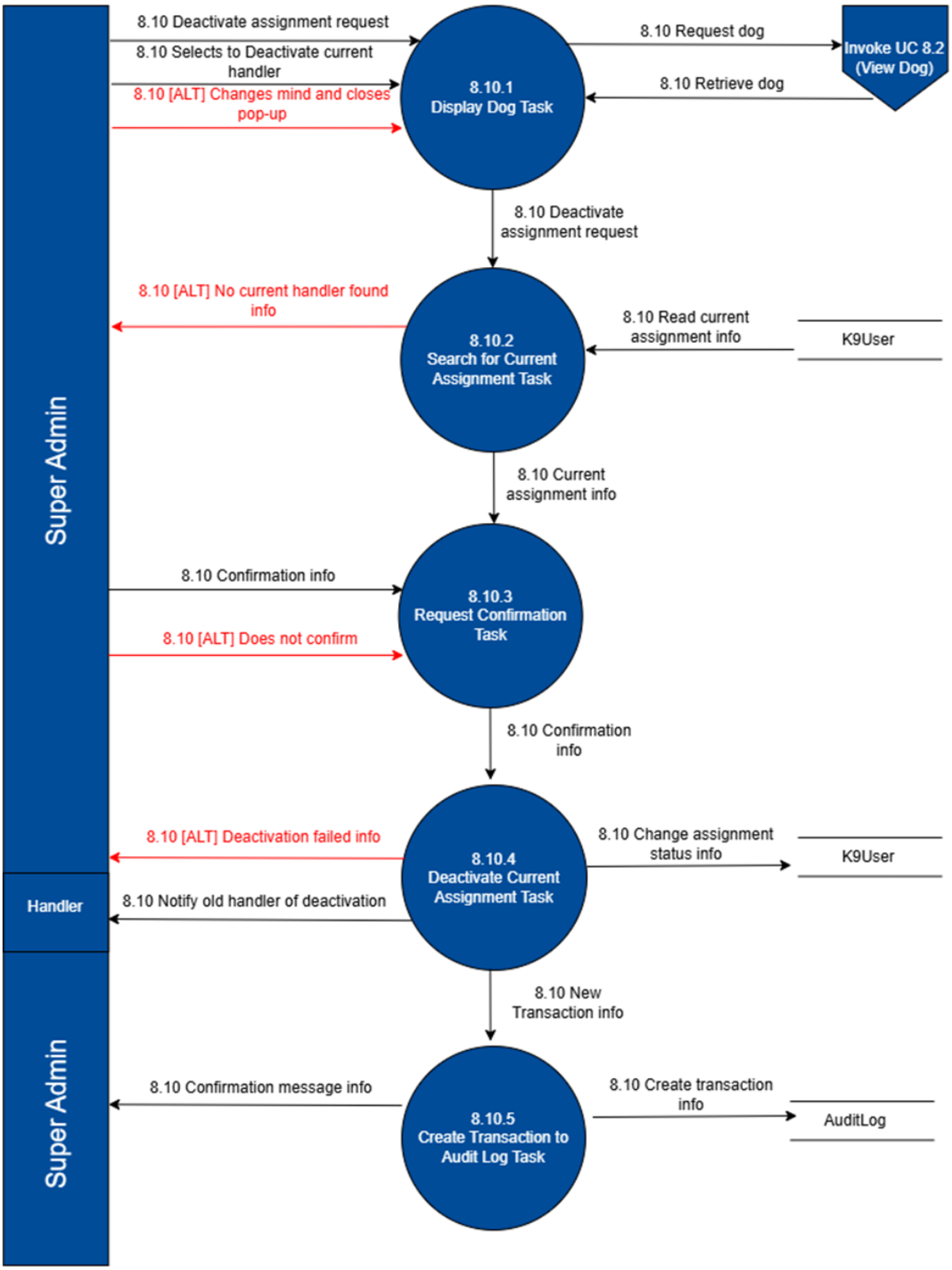


Figure 89 8.10 Deactivate User-Dog Assignment - Primitive Data Flow Diagram

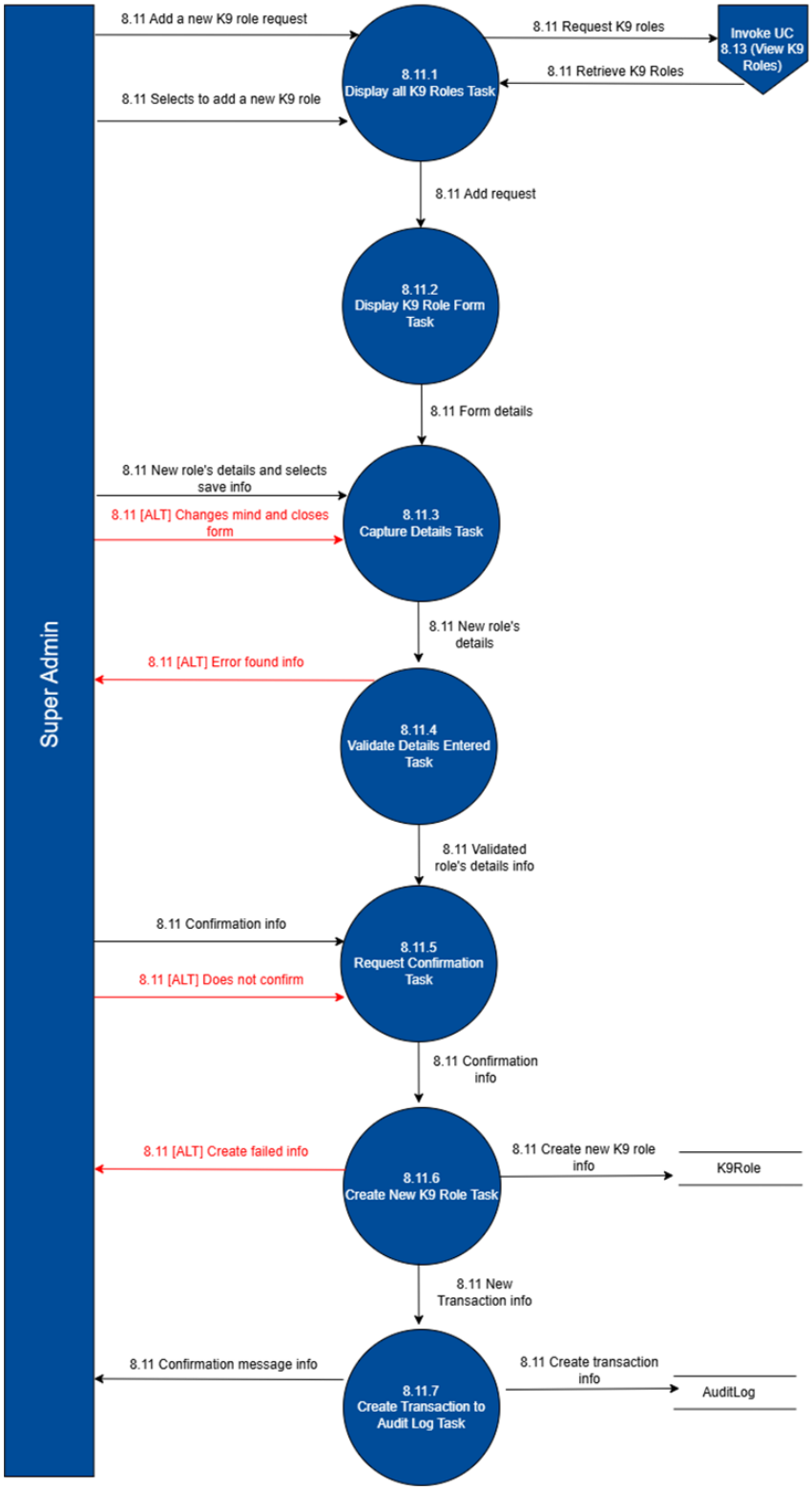


Figure 90 8.11 Add K9 Role - Primitive Data Flow Diagram

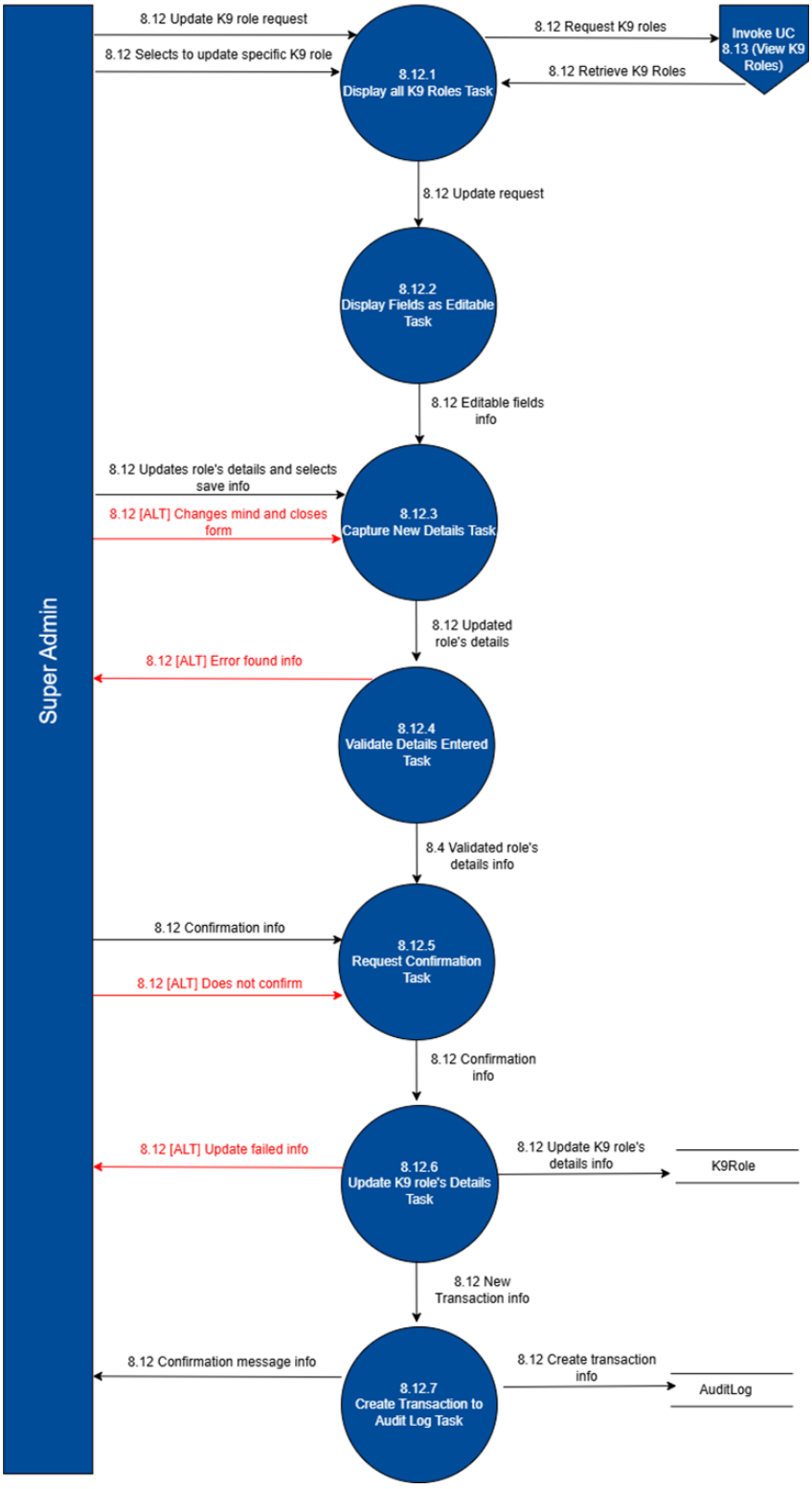


Figure 91 8.12 Update K9 Role - Primitive Data Flow Diagram

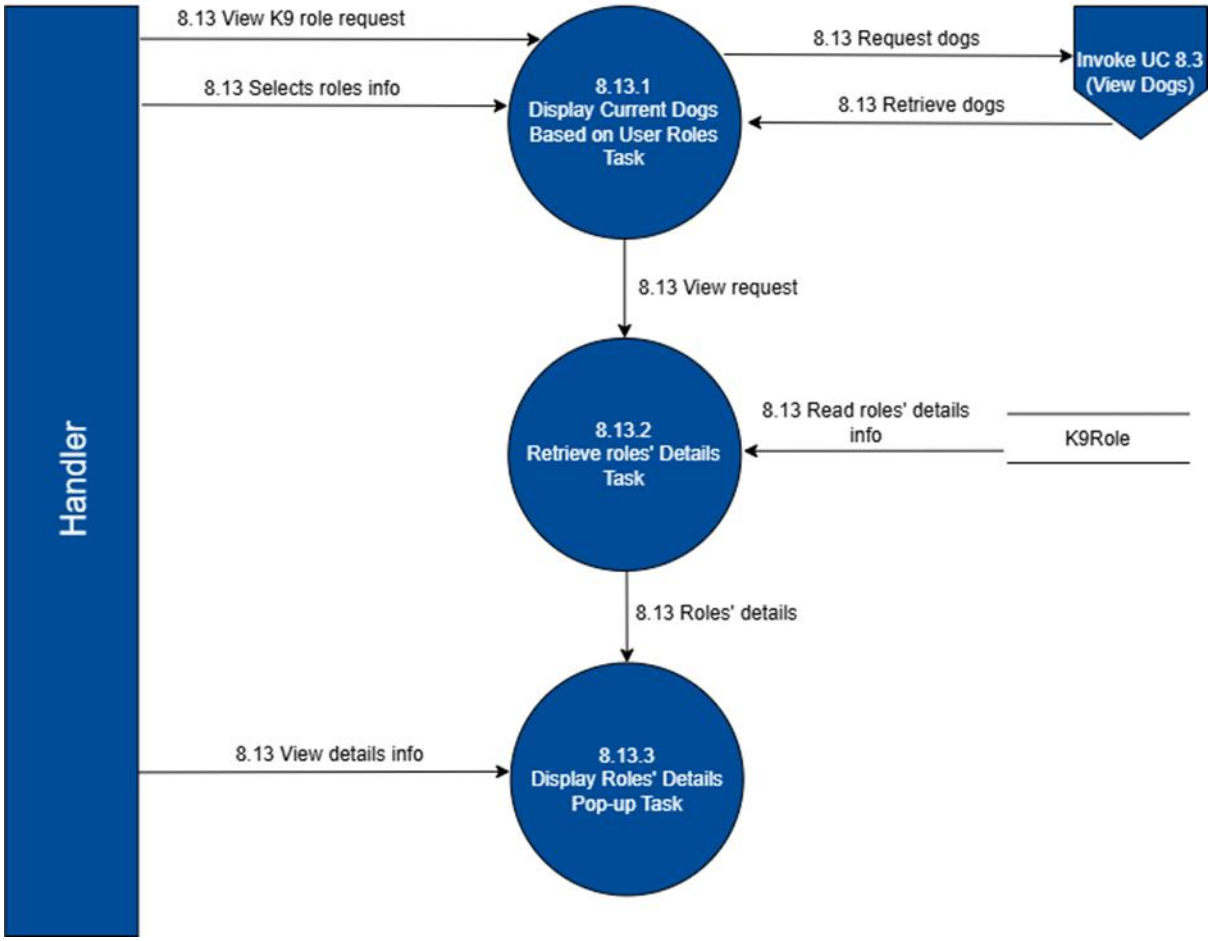


Figure 92 8.13 View K9 Role - Primitive Data Flow Diagram

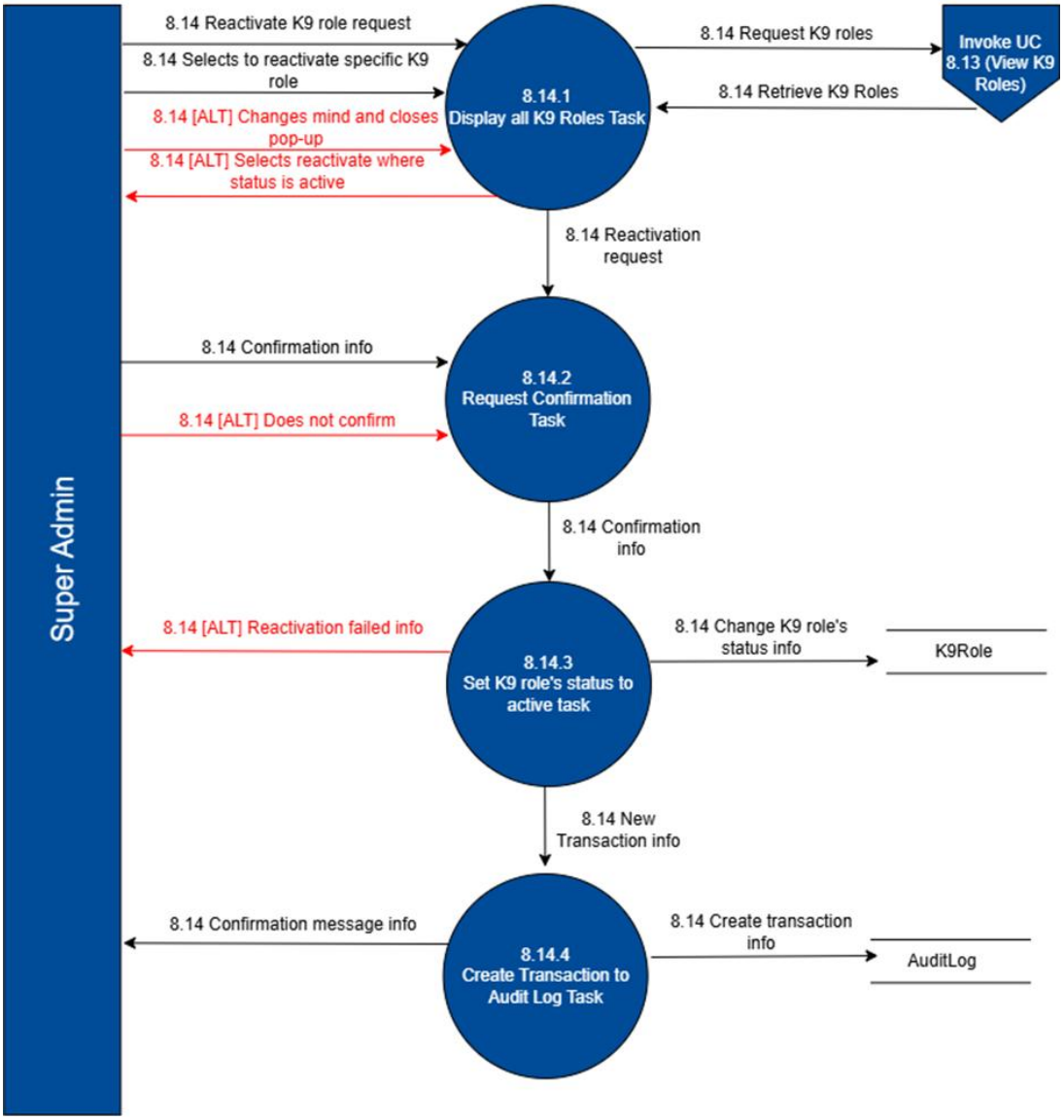


Figure 93 8.14 Reactivate K9 Role - Primitive Data Flow Diagram

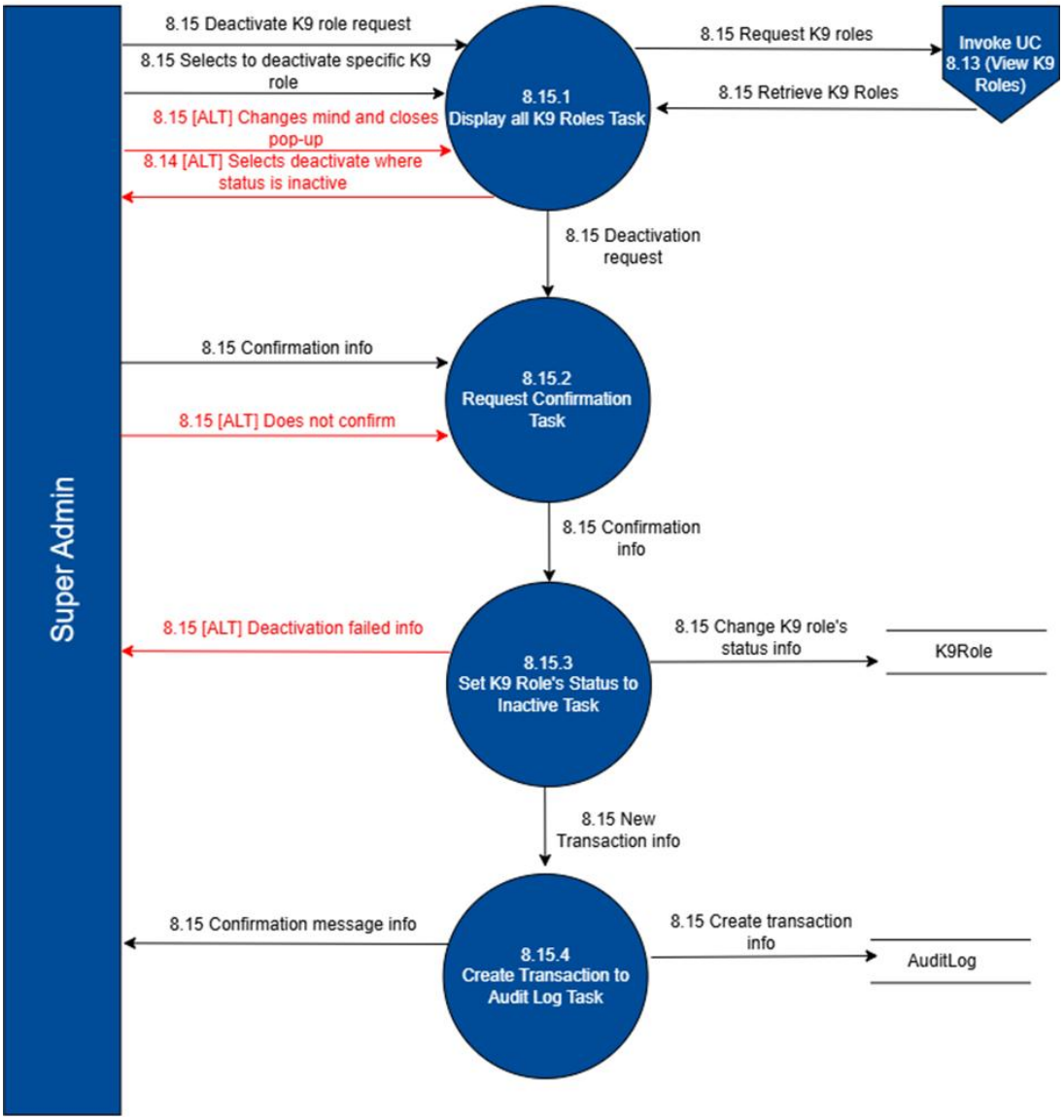


Figure 94 8.15 Deactivate Role - Primitive Data Flow Diagram

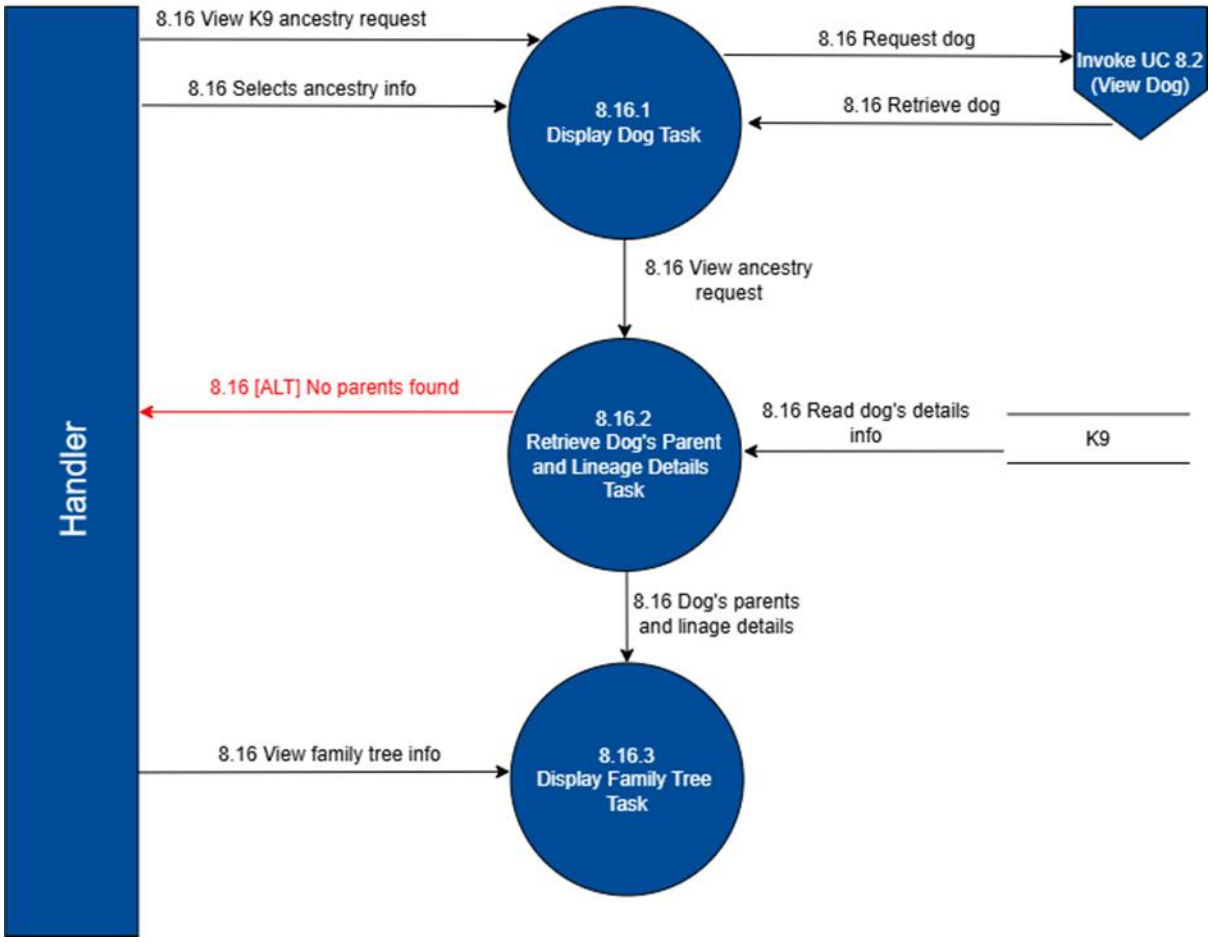


Figure 95 8.16 View K9 Ancestry - Primitive Data Flow Diagram

2.9.9 Rhino Subsystem Primitives

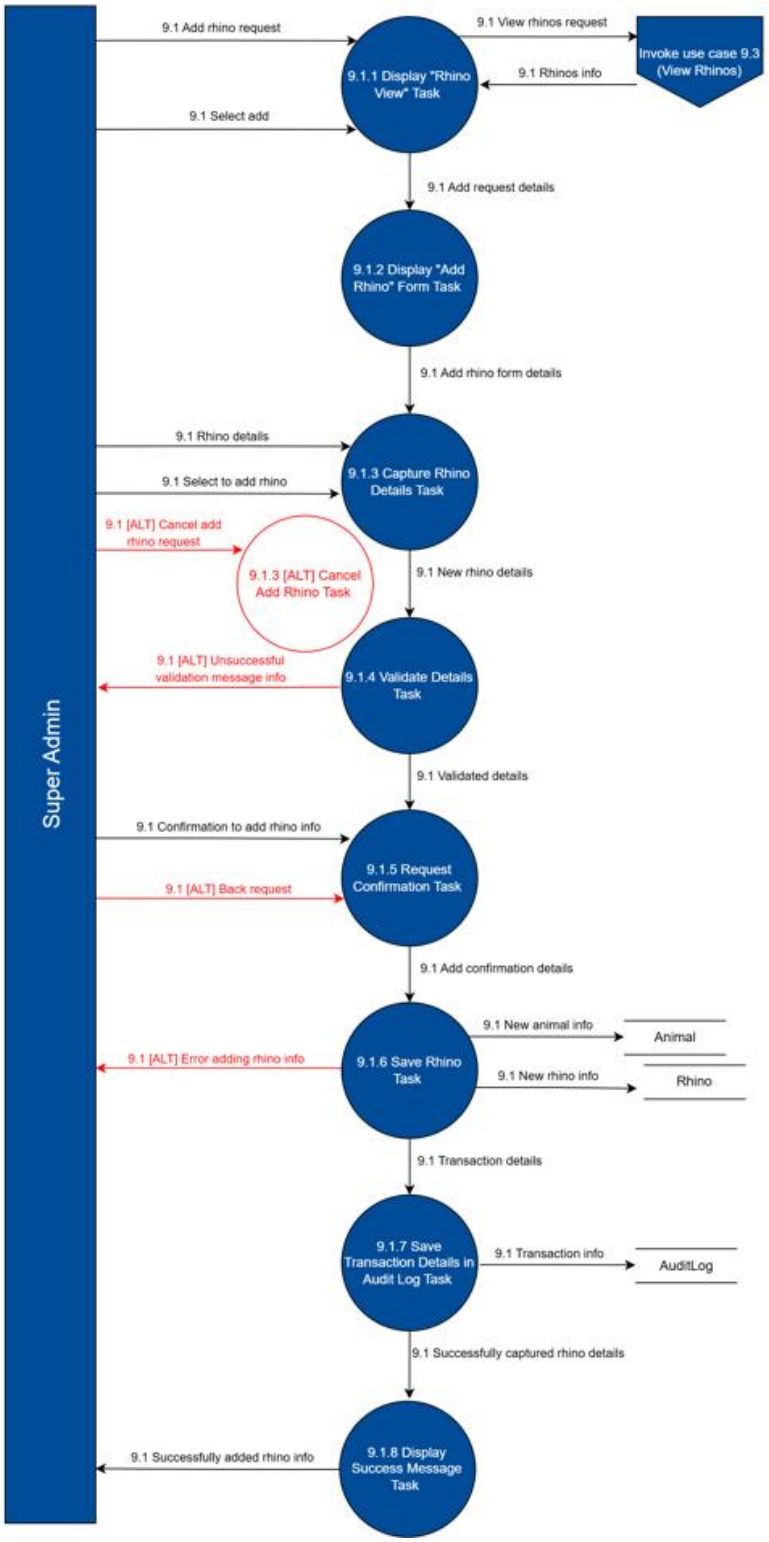


Figure 96 9.1 Add Rhino - Primitive Data Flow Diagram

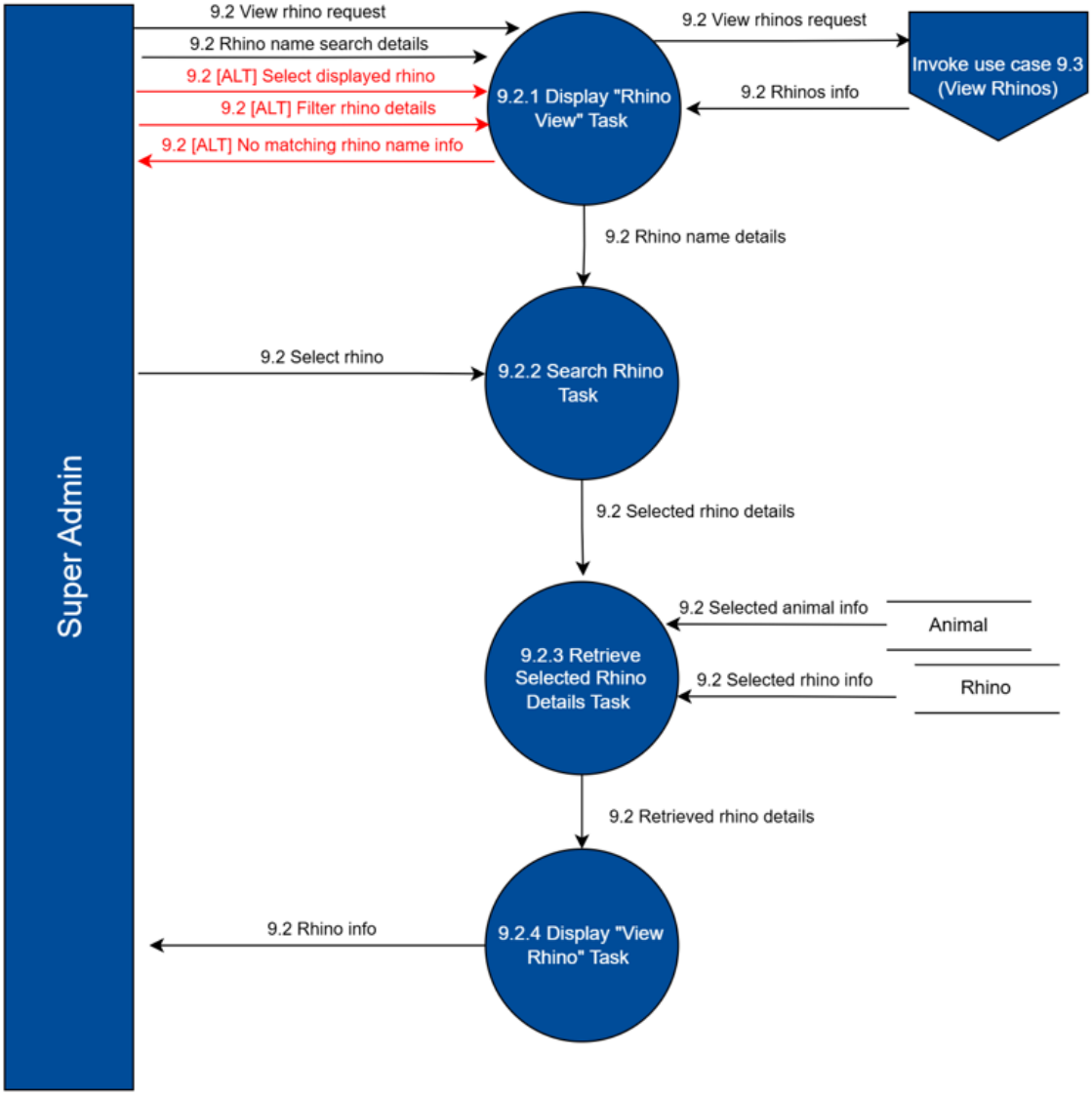


Figure 97 9.2 View Rhino - Primitive Data Flow Diagram

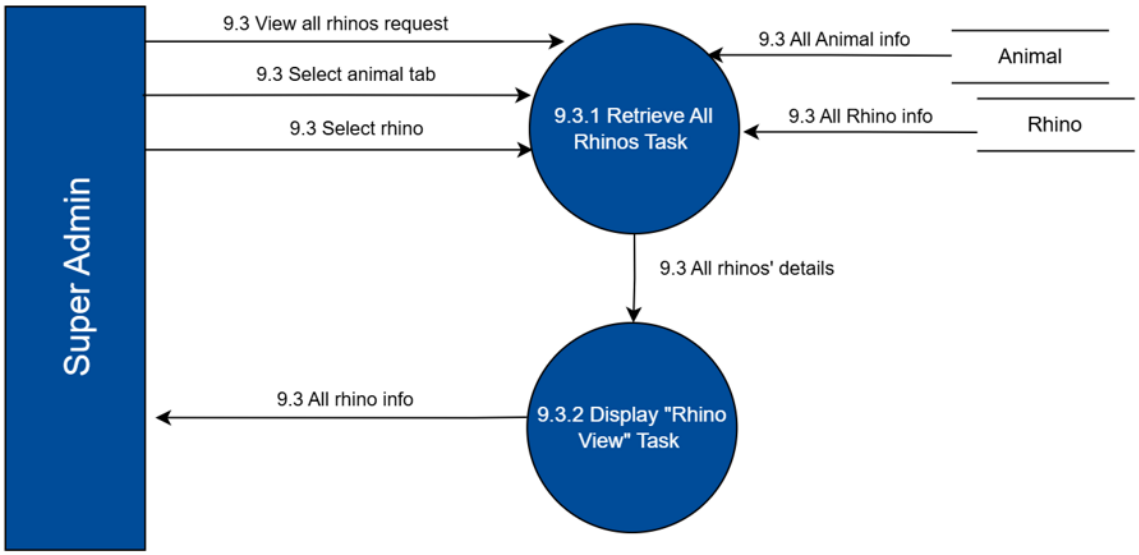


Figure 98 9.3 View Rhinos - Primitive Data Flow Diagram

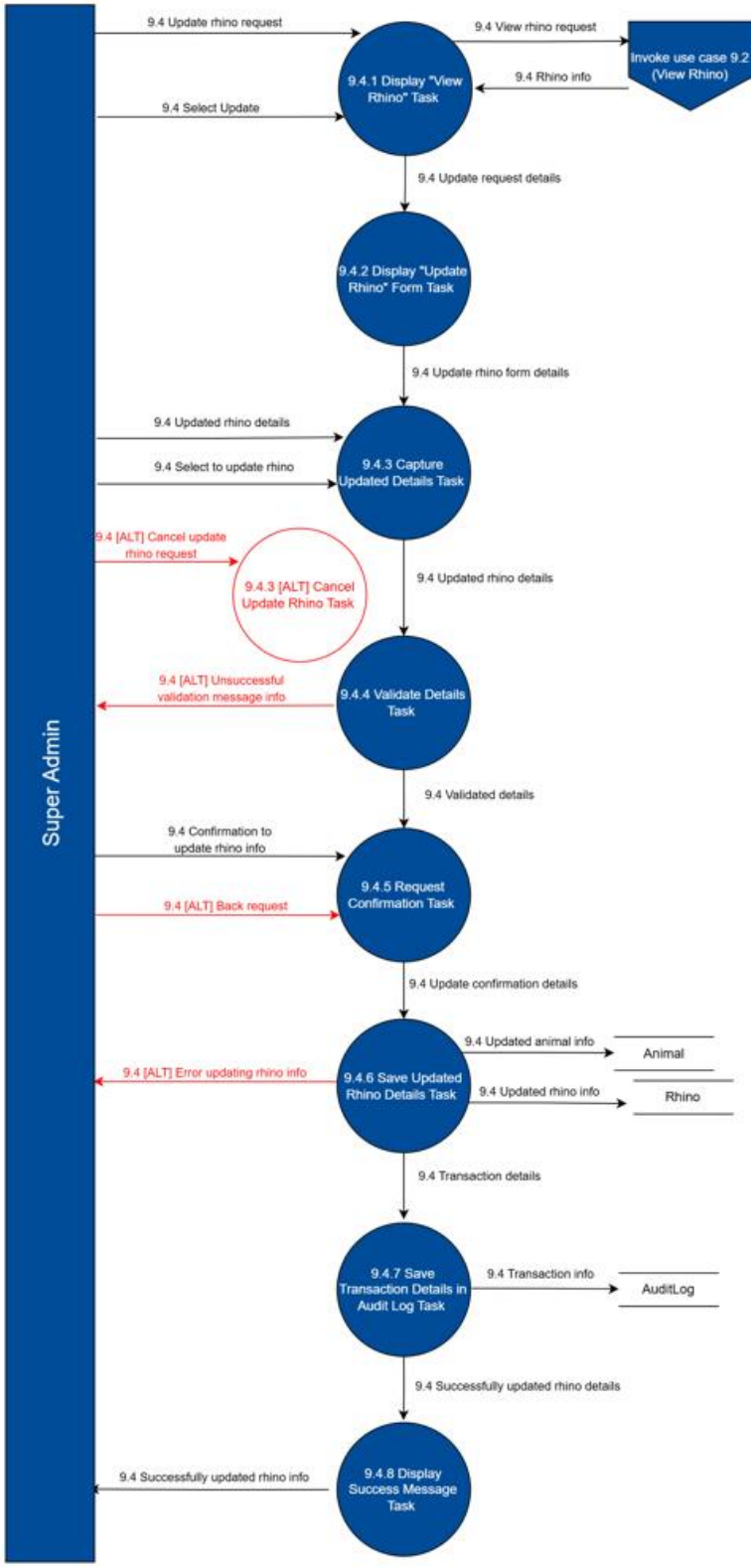


Figure 99 9.4 Update Rhino - Primitive Data Flow Diagram

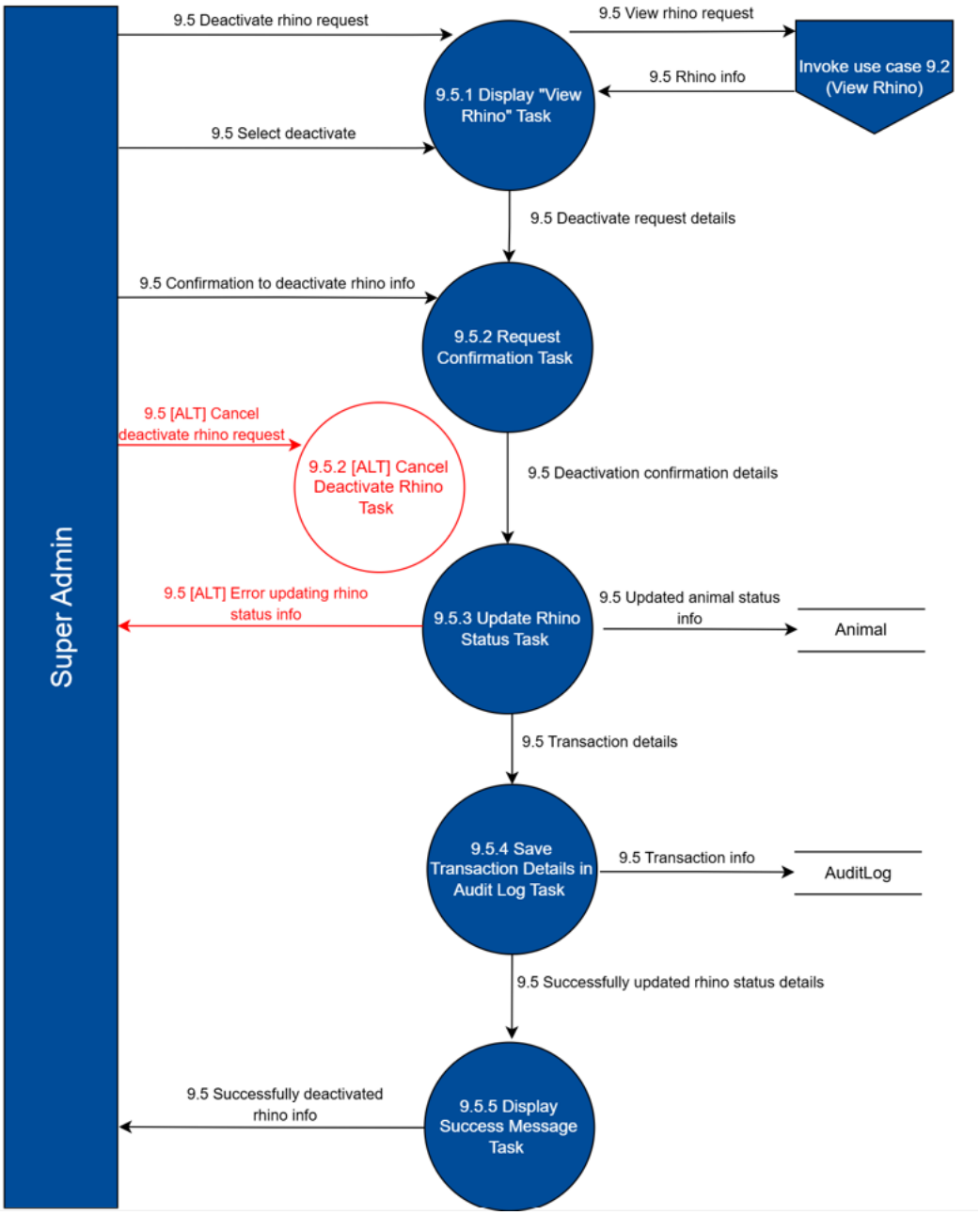


Figure 100 9.5 Deactivate Rhino - Primitive Data Flow Diagram

2.9.10 Elephant Subsystem Primitives

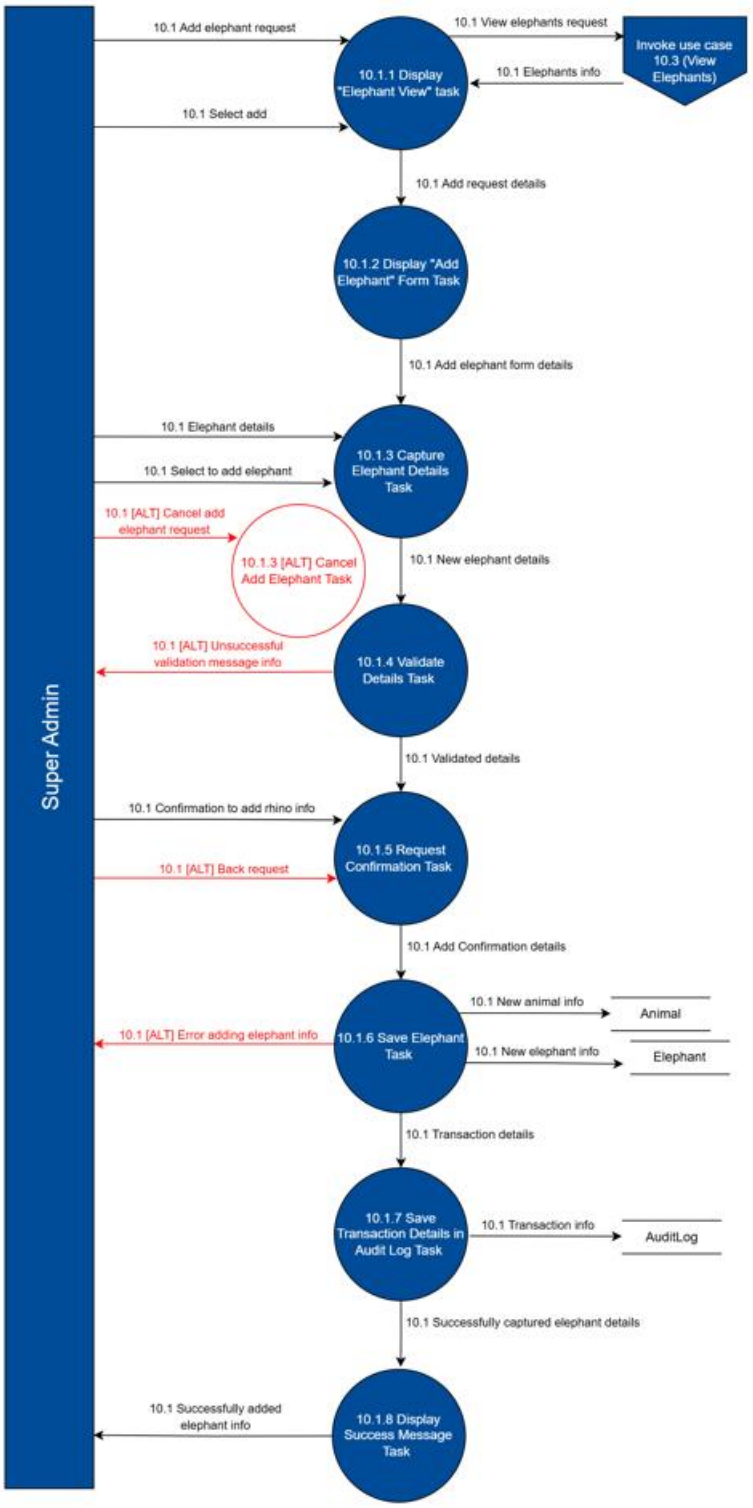


Figure 101 10.1 Add Elephant - Primitive Data Flow Diagram

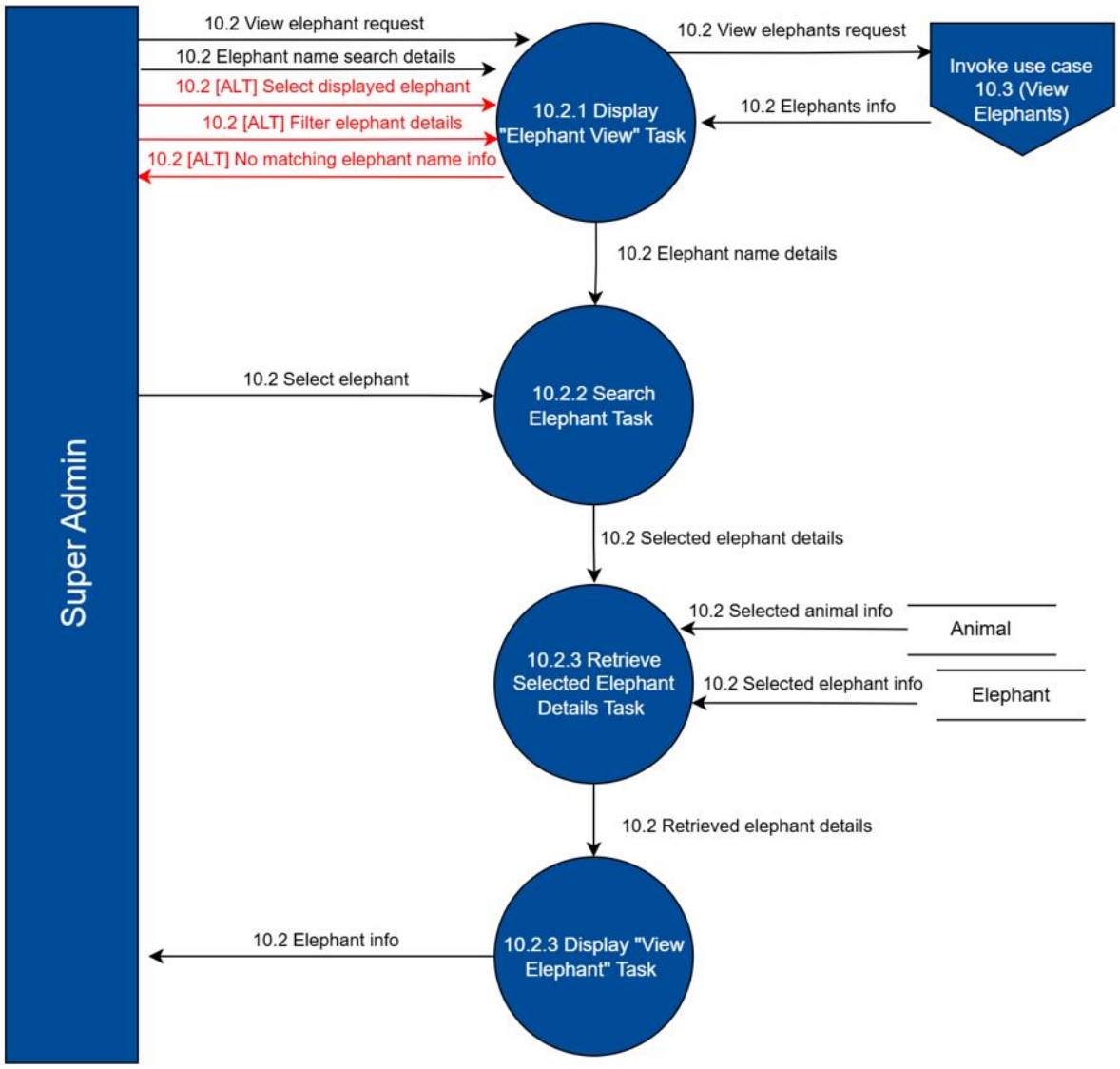


Figure 102 10.2 View Elephant - Primitive Data Flow Diagram

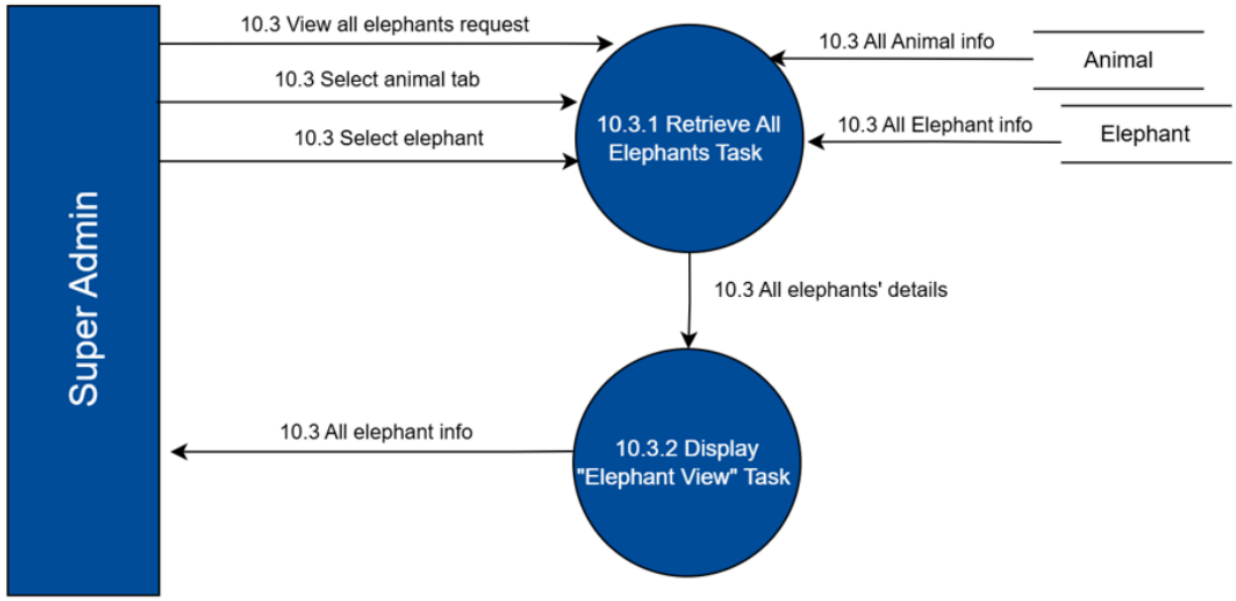


Figure 103 10.3 View Elephants - Primitive Data Flow Diagram

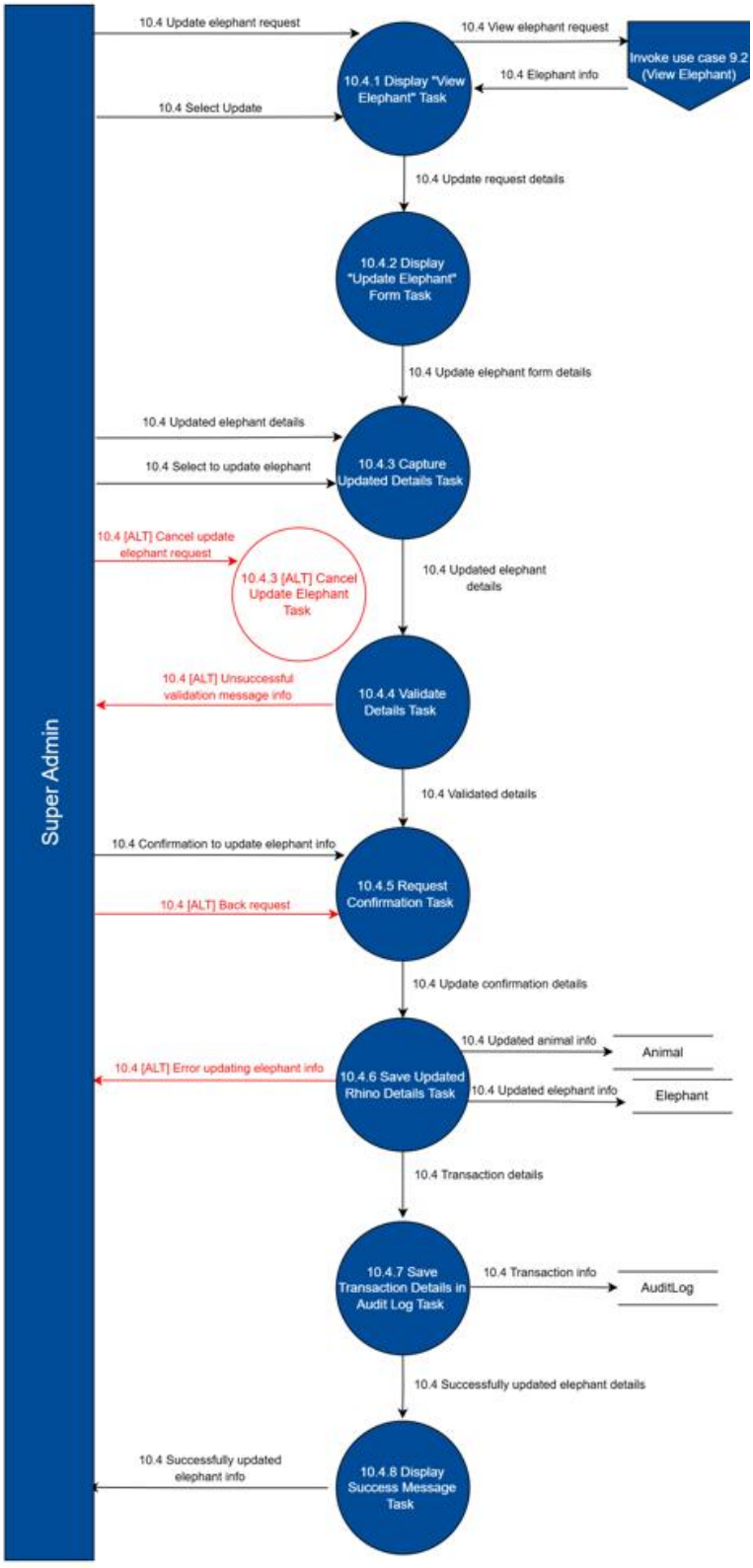


Figure 104 10.4 Update Elephant - Primitive Data Flow Diagram

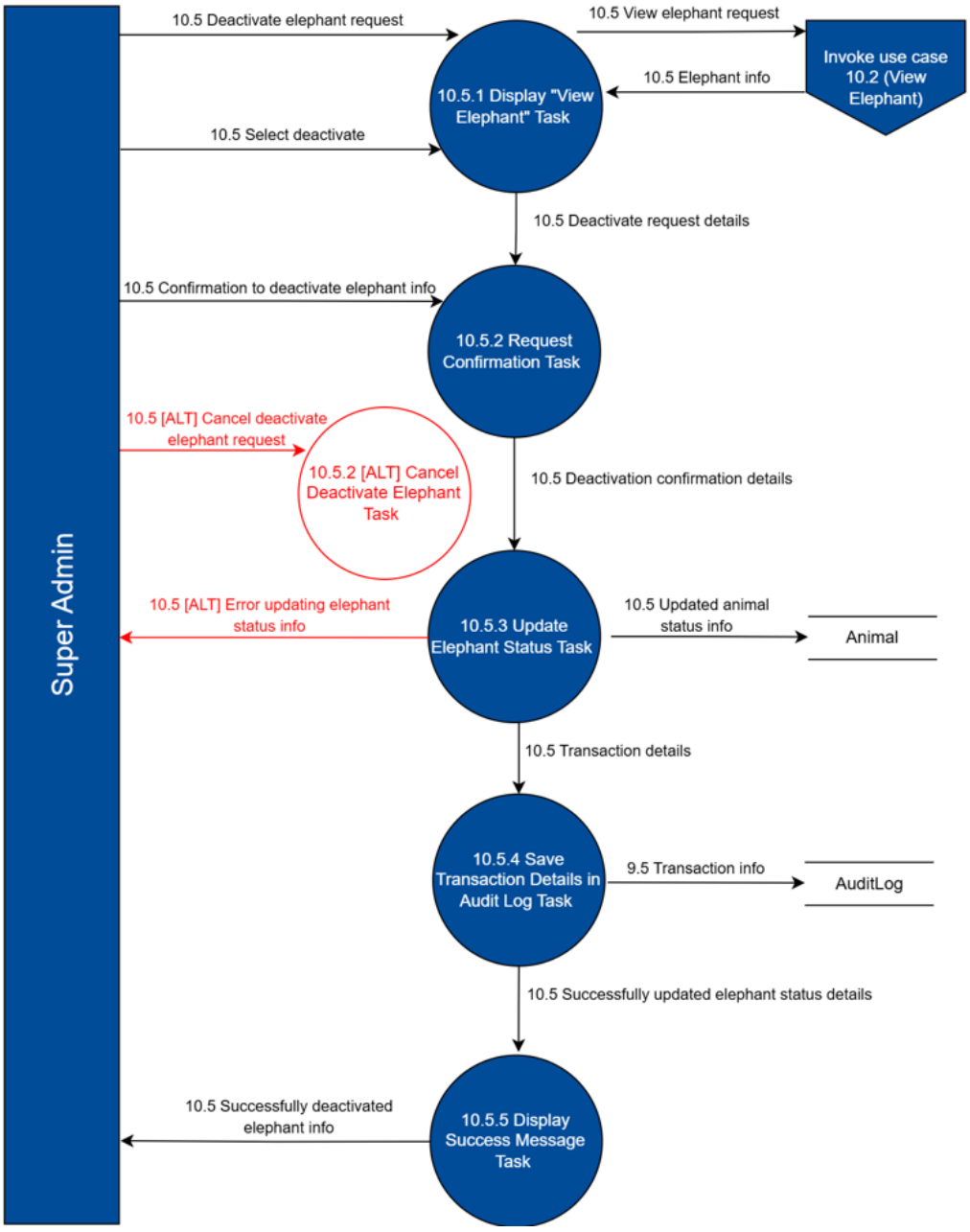
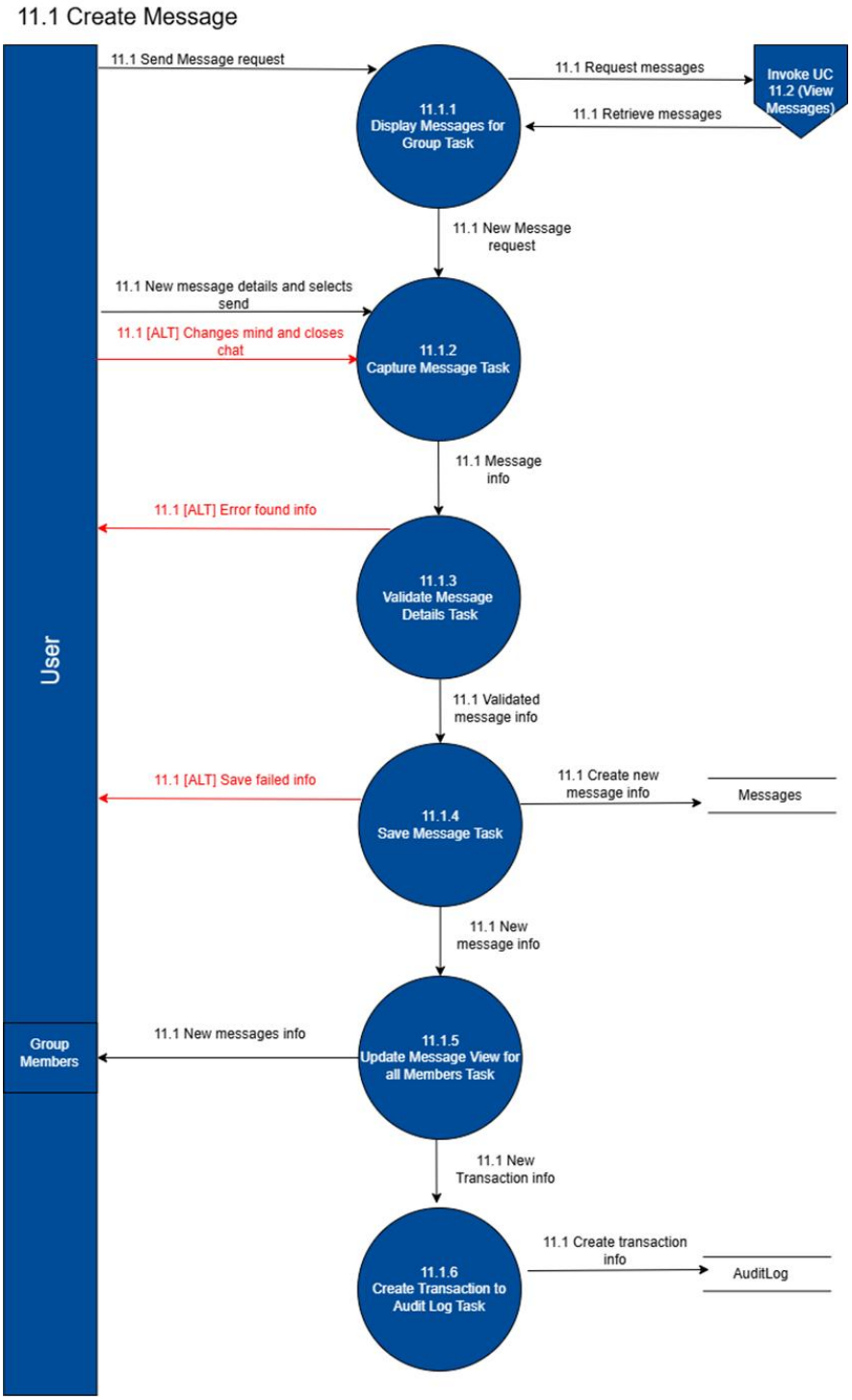
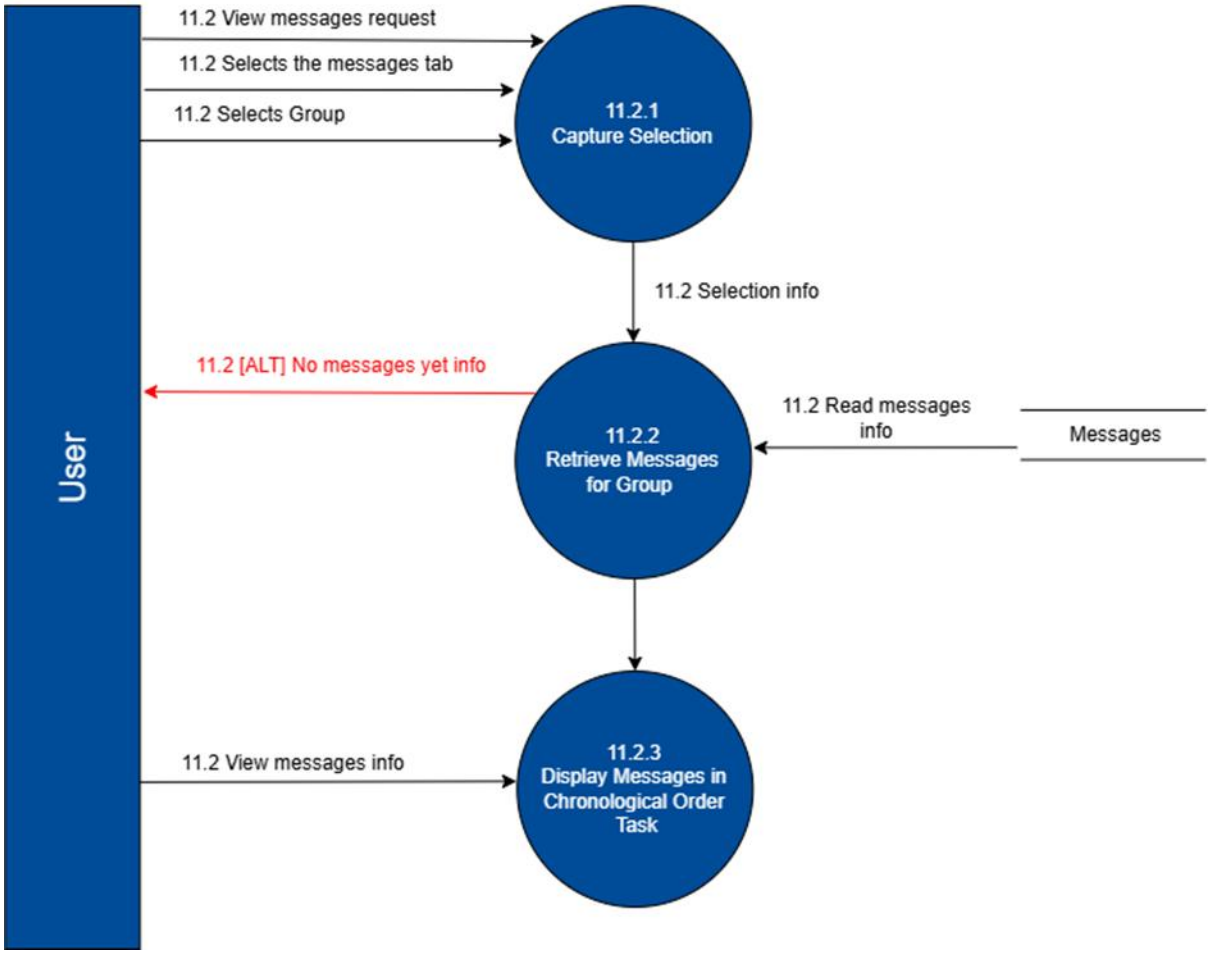


Figure 105 10.5 Deactivate Elephant - Primitive Data Flow Diagram

2.9.11 Message Subsystem Primitives




User
11.2 View messages request
11.2 Selects the messages tab
11.2 Selects Group
11.2.1
Capture Selection
11.2 Selection info
11.2 [ALT] No messages yet info
11.2.2
Retrieve Messages
for Group
11.2 Read messages
info
Messages
11.2 View messages info
11.2.3
Display Messages in
Chronological Order
Task

2.9.12 Report Subsystem Primitives

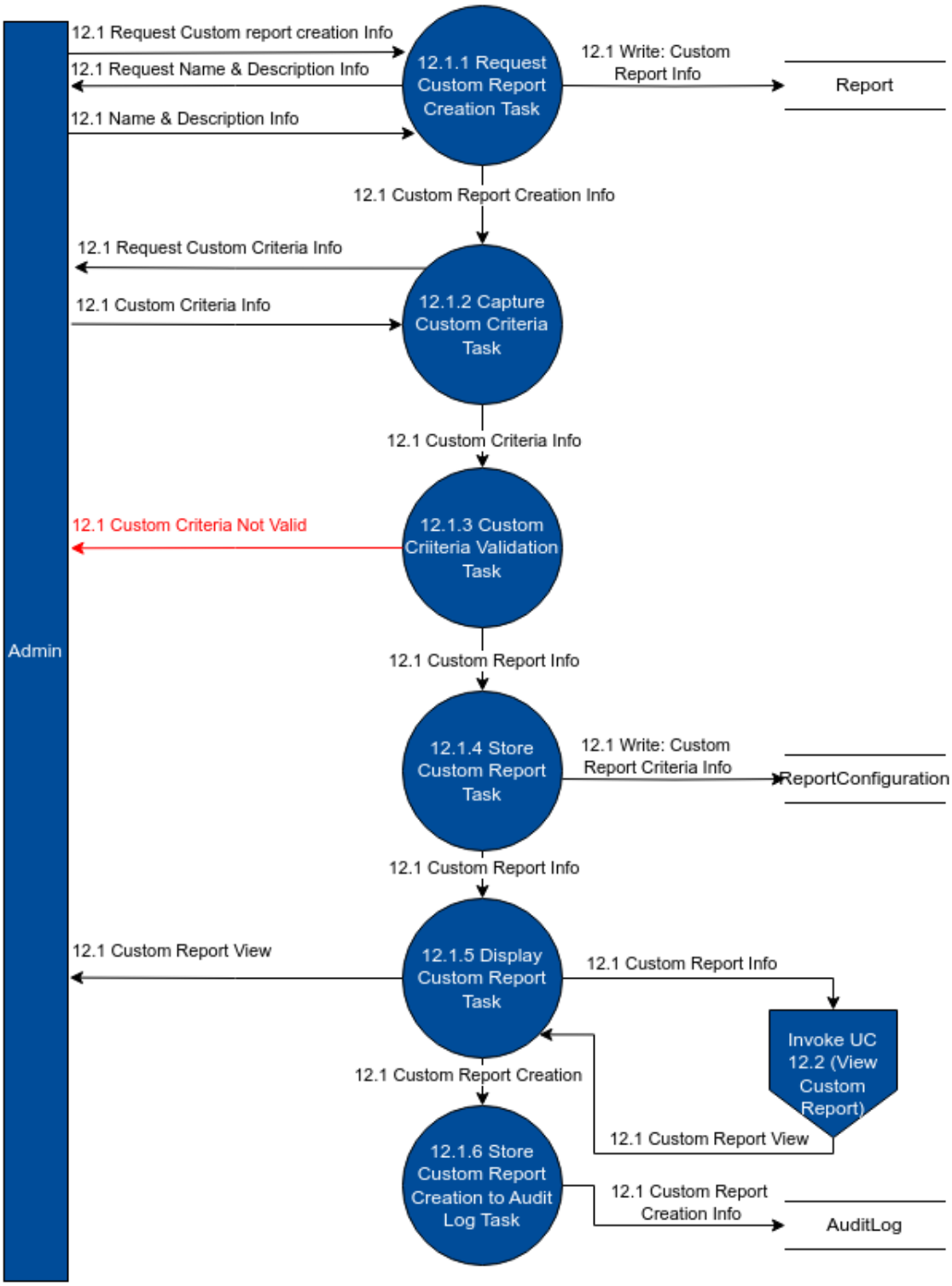


Figure 106: 12.1 Generate Custom Report - Primitive Data Flow Diagram

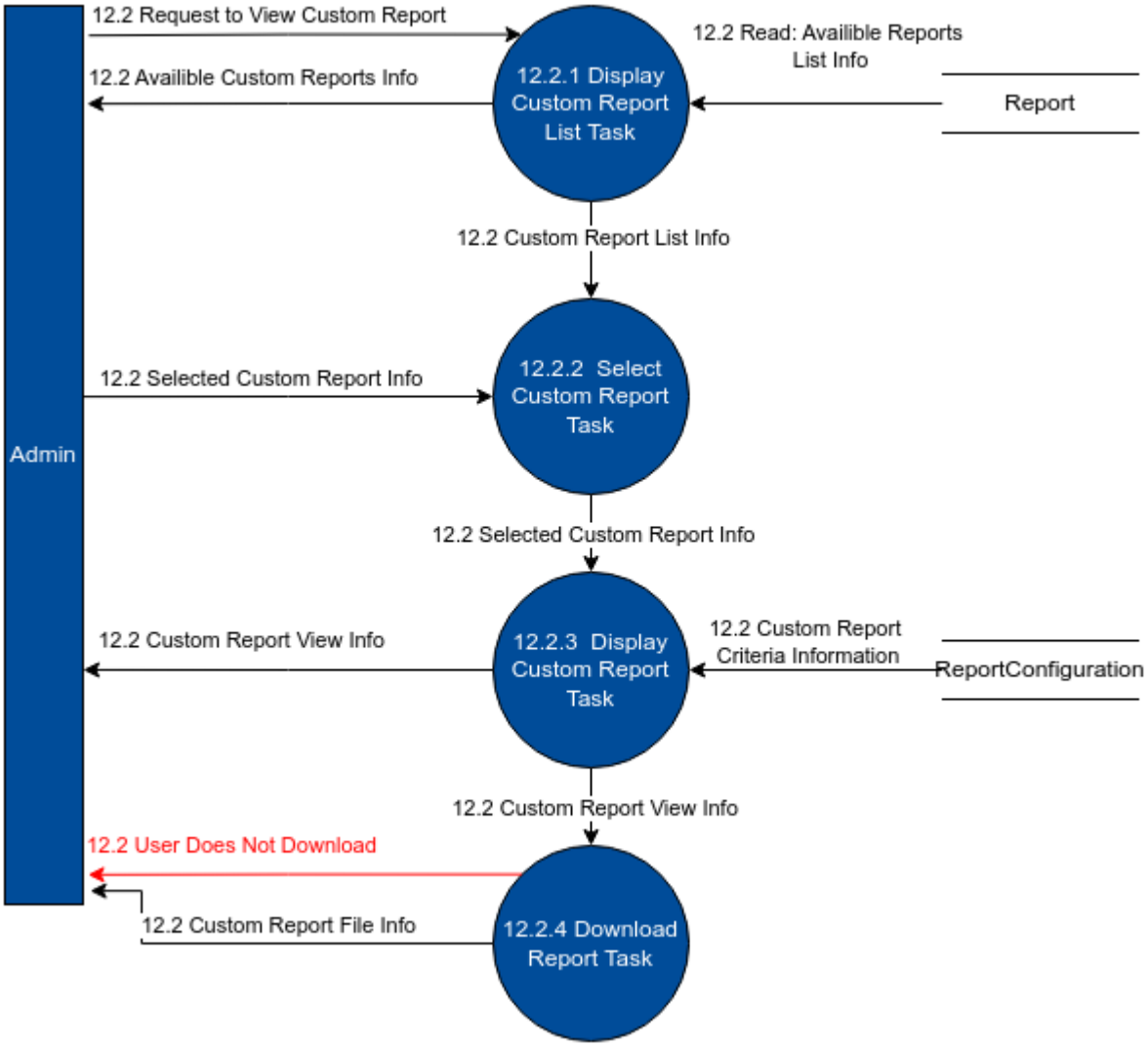


Figure 107: 12.2 View Custom Report - Primitive Data Flow Diagram

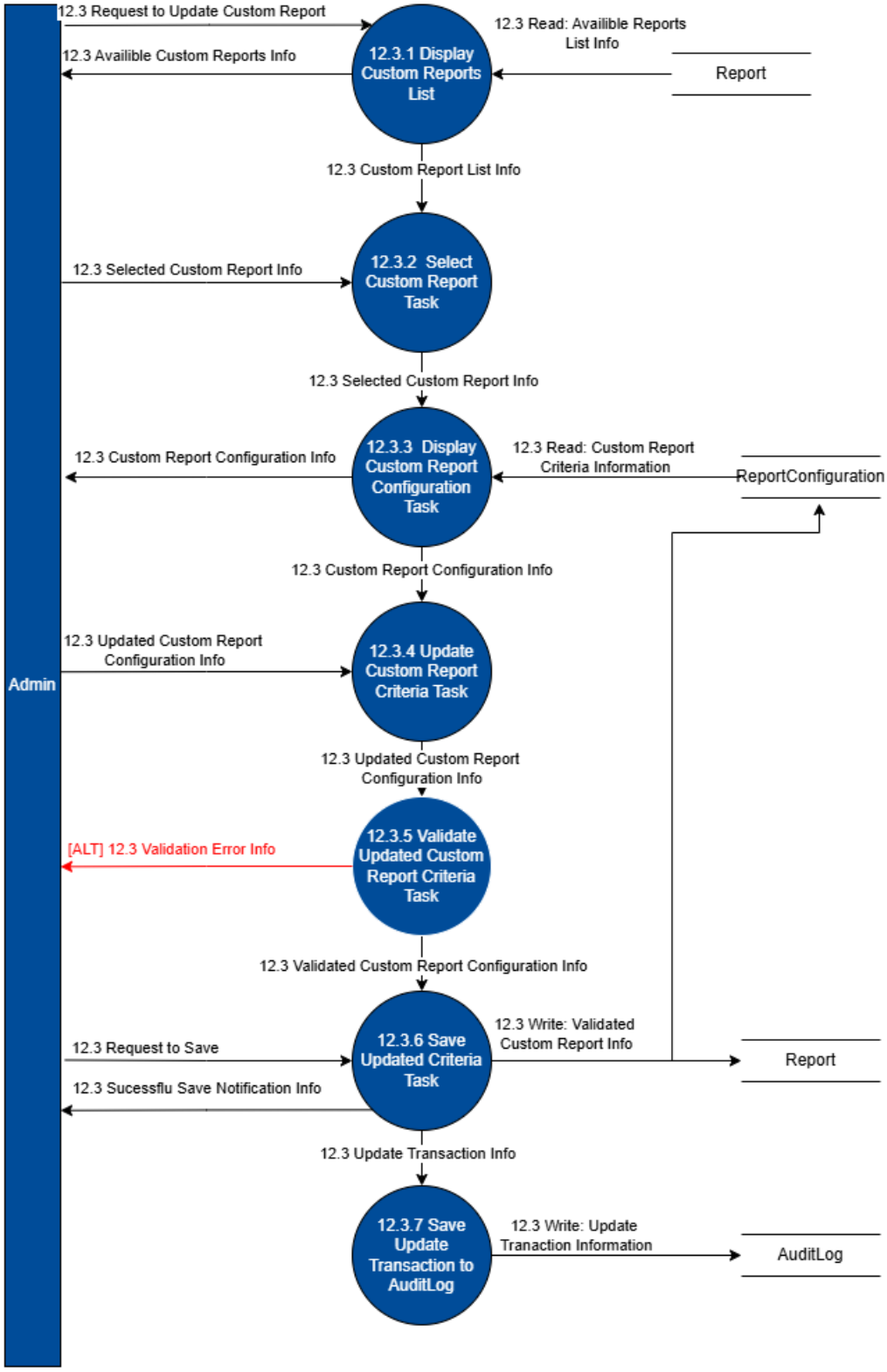


Figure 108: 12.3 Update Custom Report - Primitive Data Flow Diagram

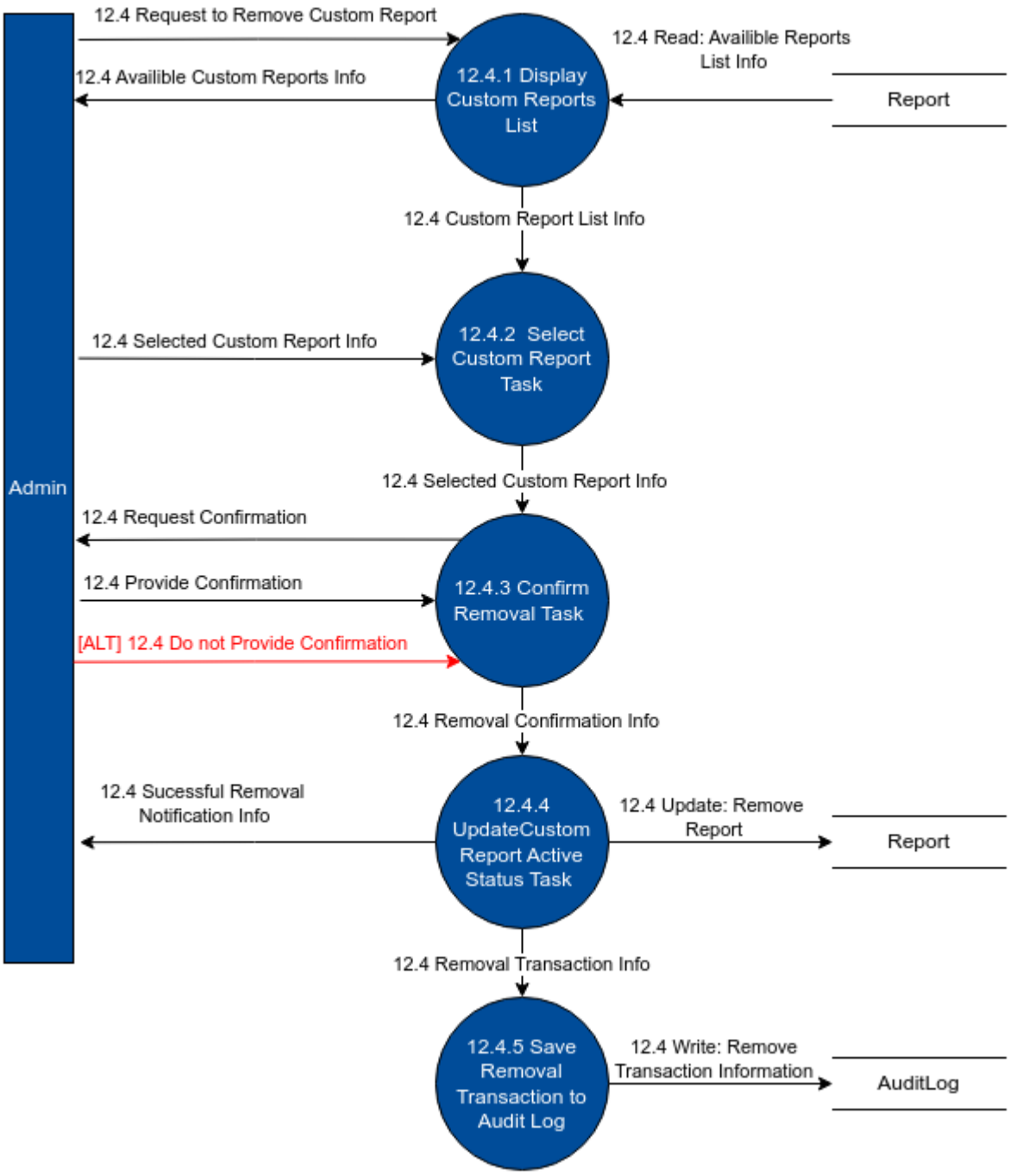


Figure 109: 12.4 Remove Custom Report - Primitive Data Flow Diagram

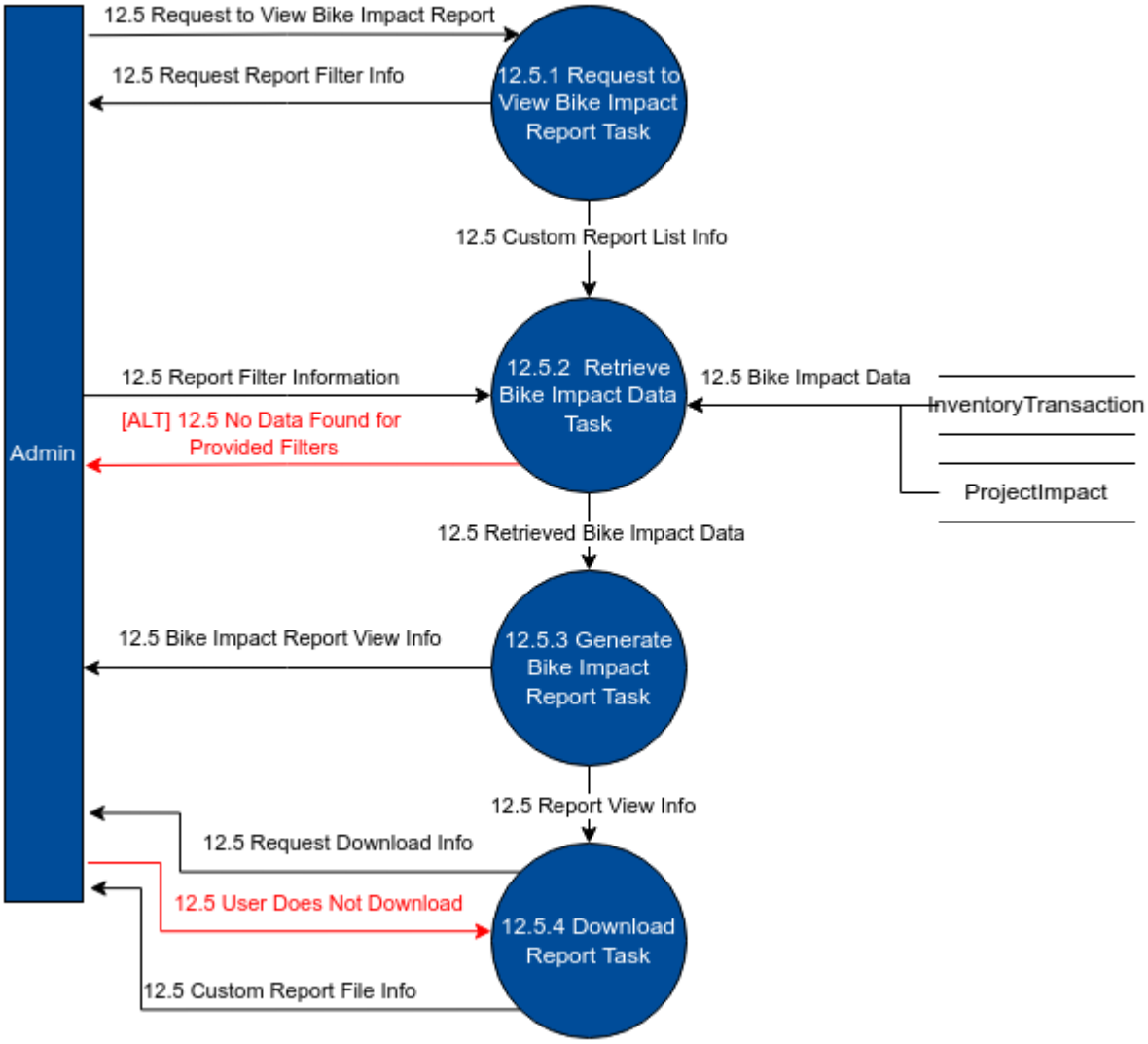


Figure 110: 12.5 View Bike Impact Report - Primitive Data Flow Diagram

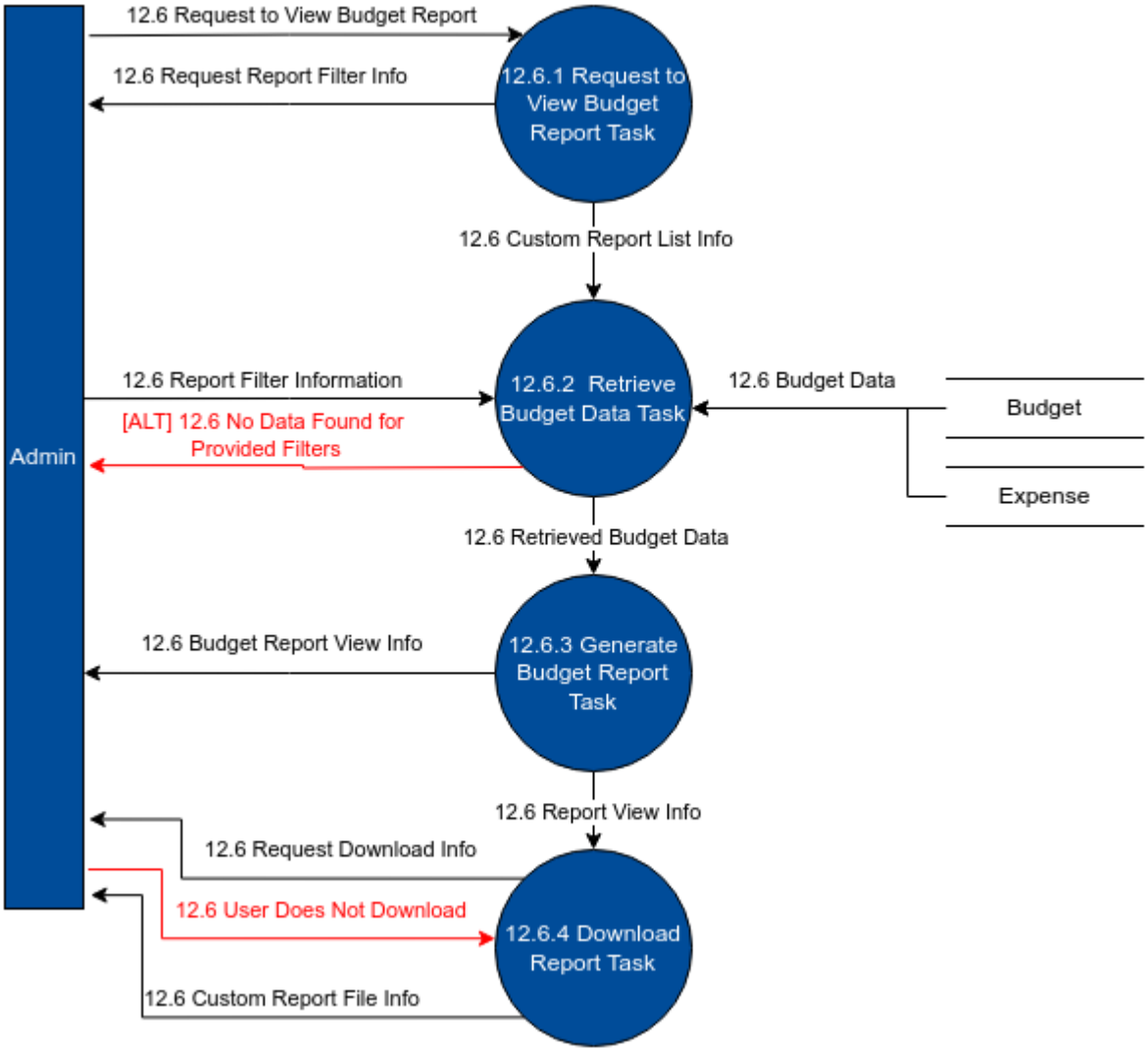


Figure 111: 12.6 View Budget Report - Primitive Data Flow Diagram

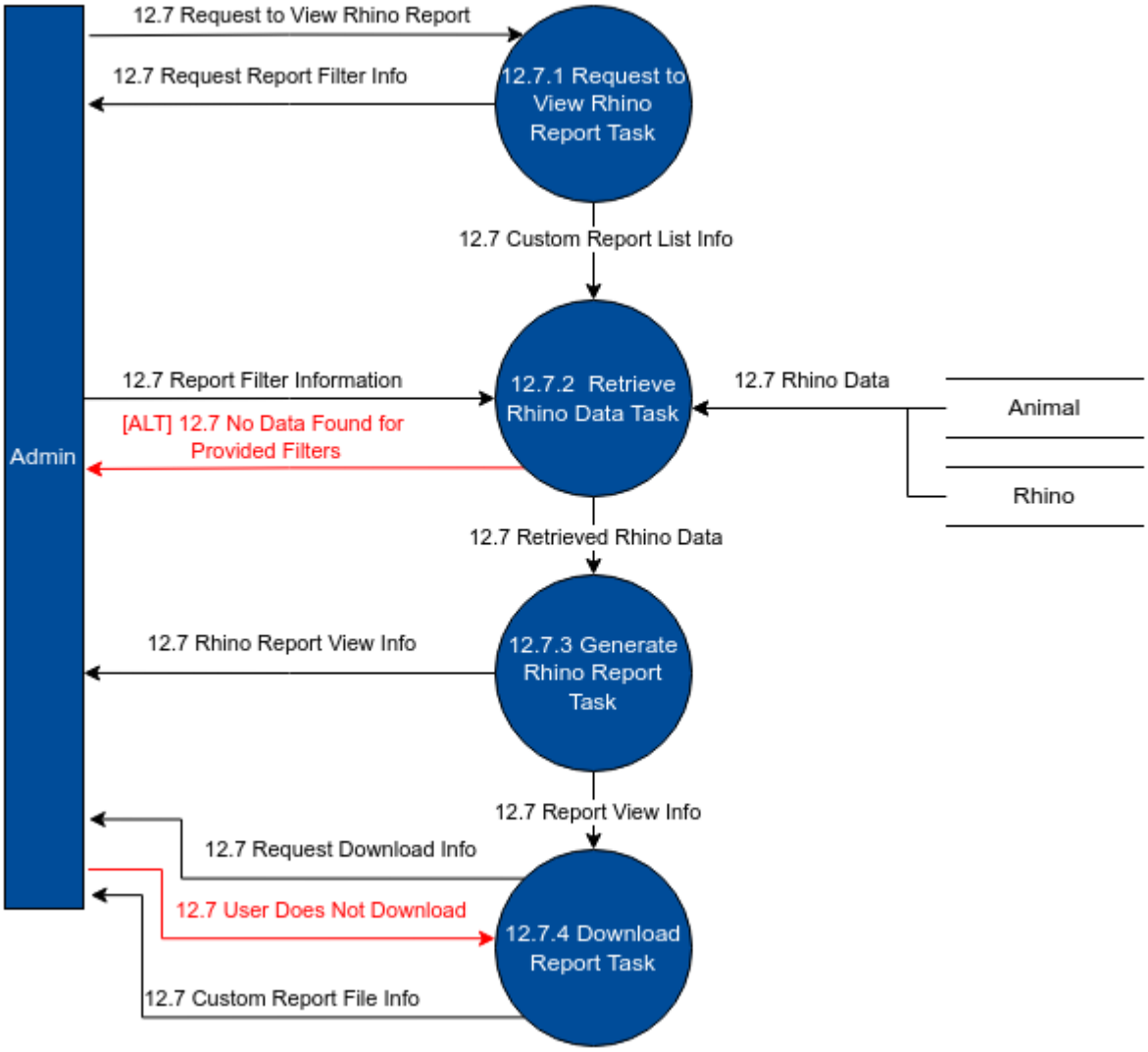


Figure 112: 12.7 View Rhino Report - Primitive Data Flow Diagram

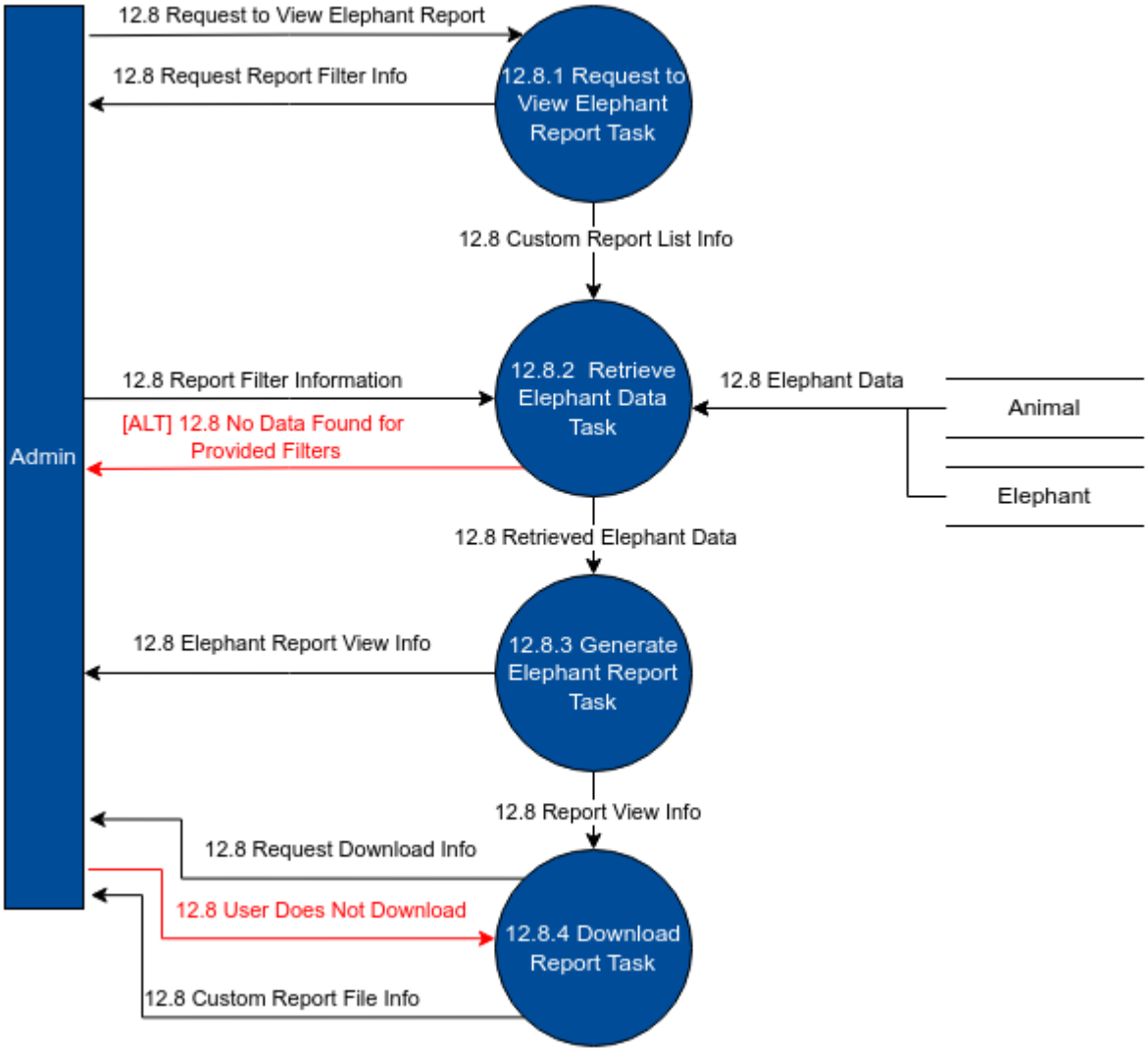


Figure 113: 12.8 View Elephant Report - Primitive Data Flow Diagram

2.9.13 Machine Learning Subsystem Primitives

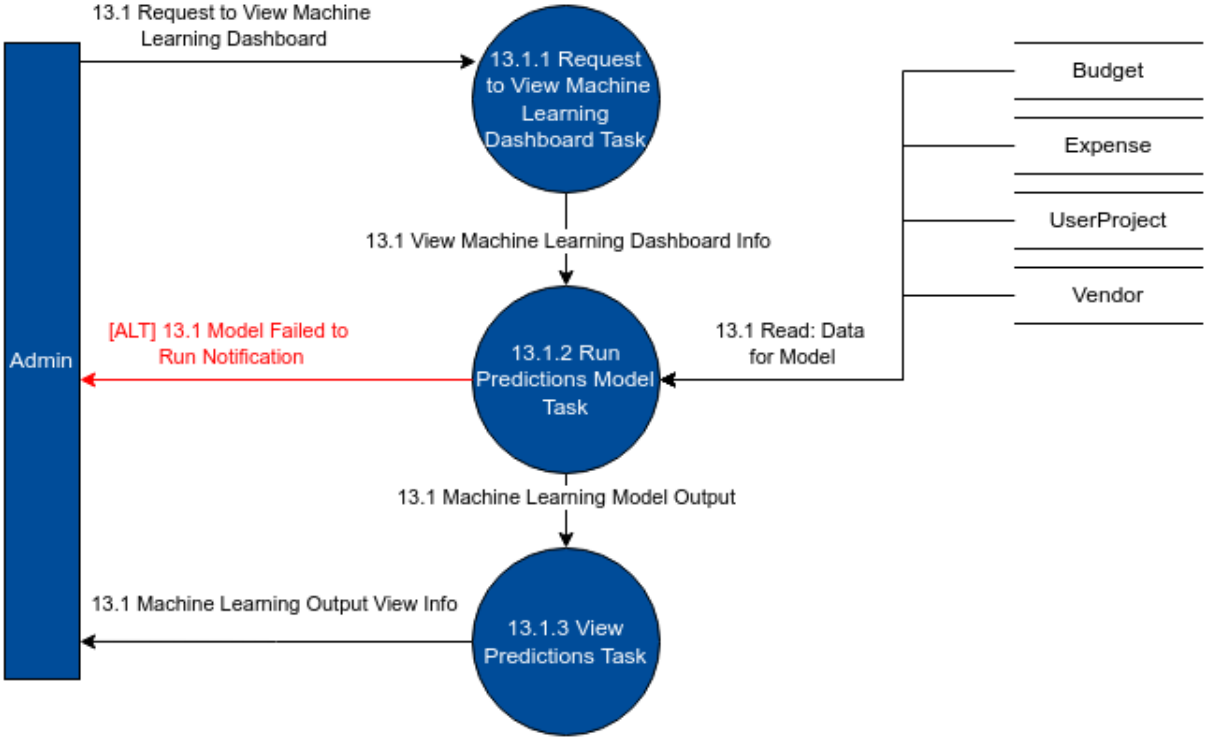


Figure 114: 13.1 View Audit Log - Primitive Data Flow Diagram

1.9.14 System Subsystem Primitives

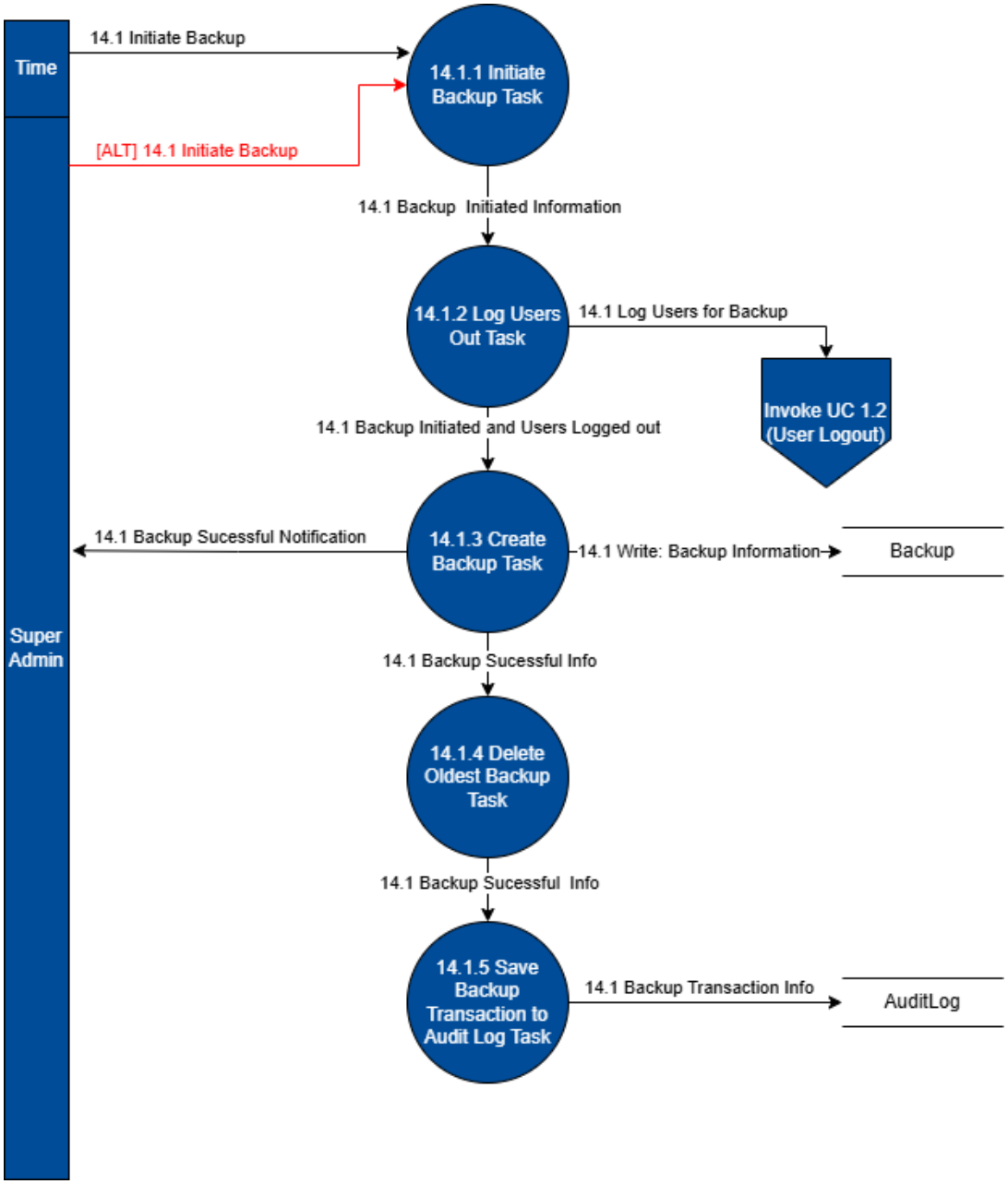


Figure 115: 14.1 Backup System - Primitive Data Flow Diagram

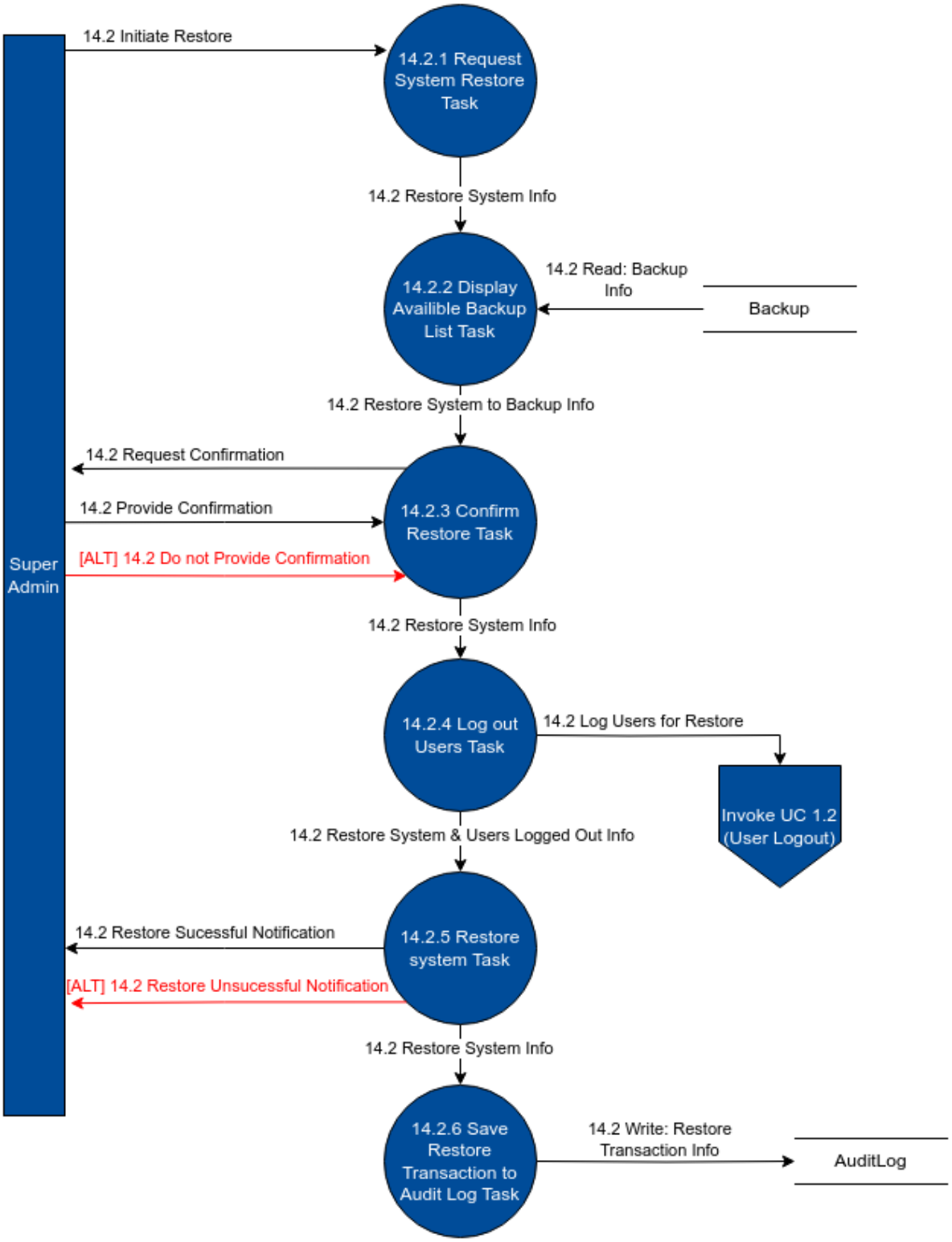


Figure 116: 14.2 Restore System - Primitive Data Flow Diagram

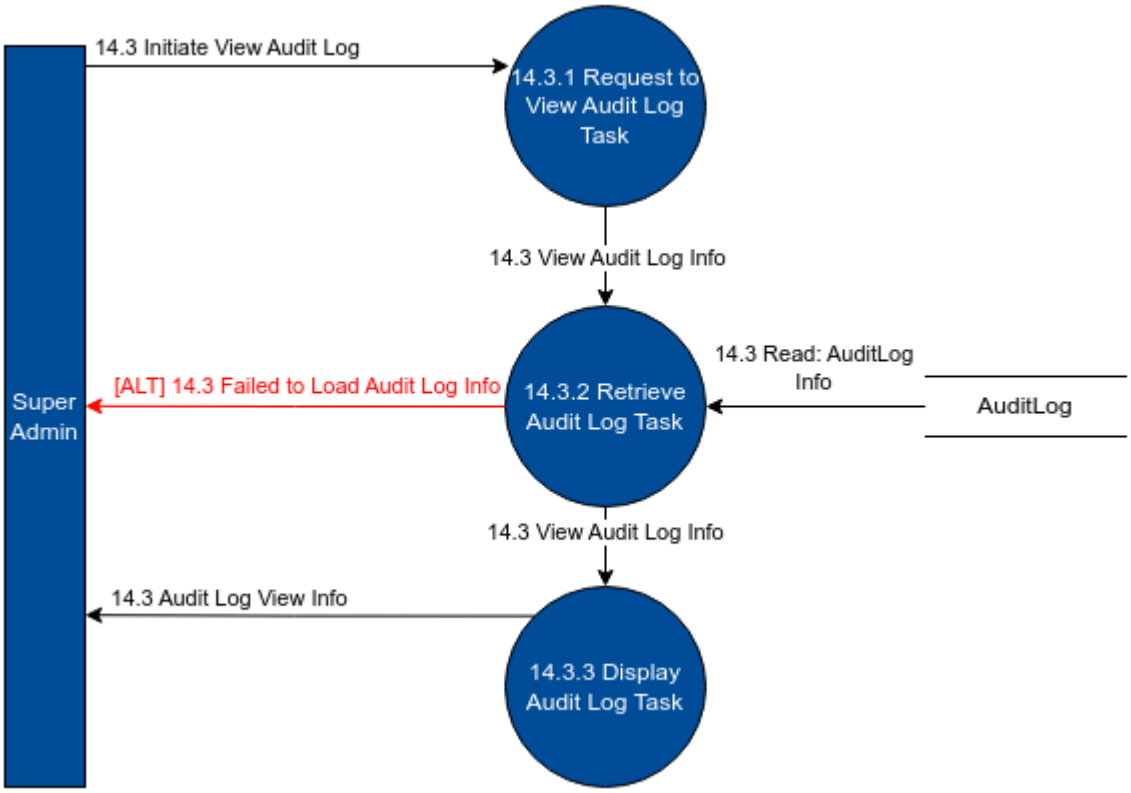


Figure 117: 14.3 View Audit Log - Primitive Data Flow Diagram

2.9.15 Budget Subsystem Primitives

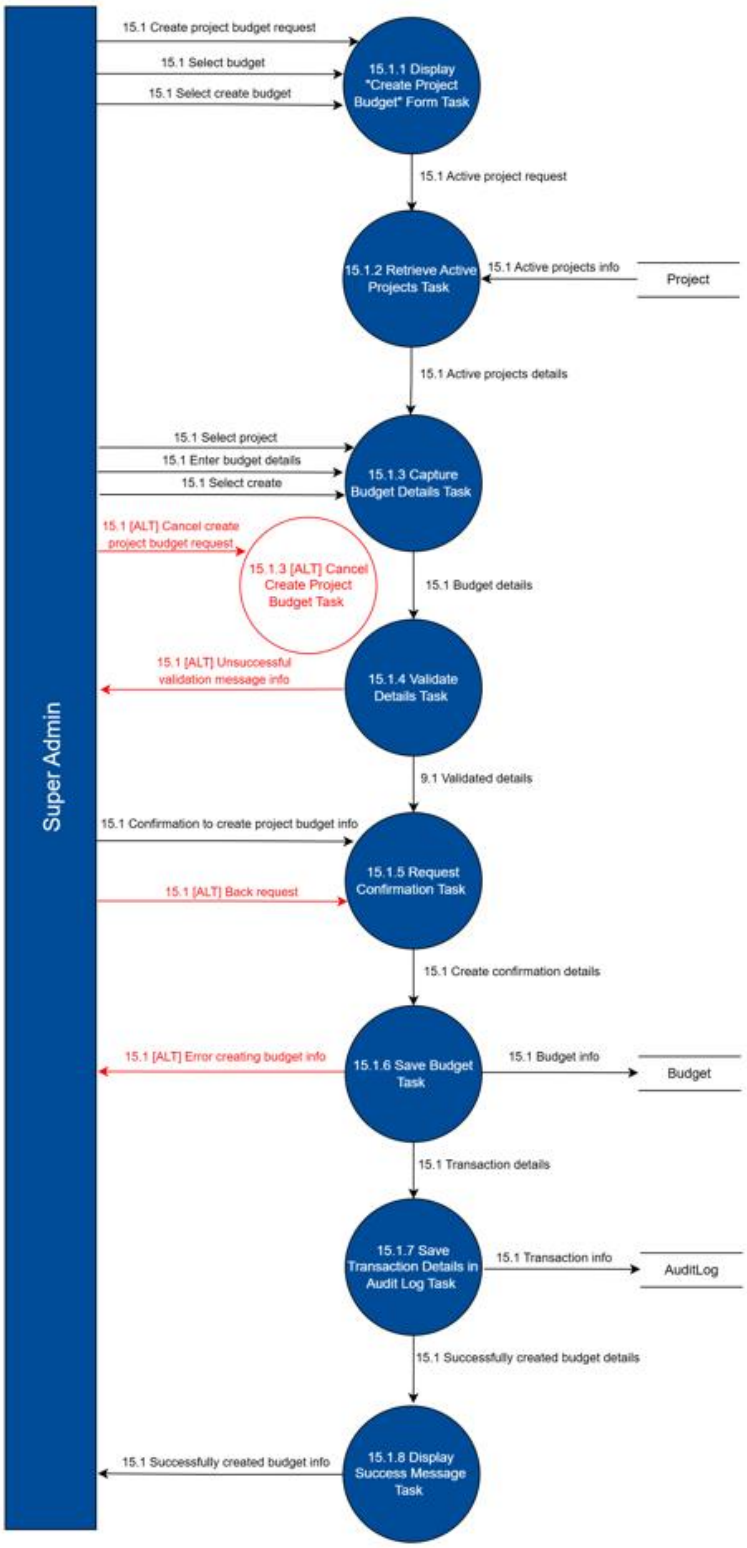


Figure 118 15.1 Create Project Budget - Primitive Data Flow Diagram

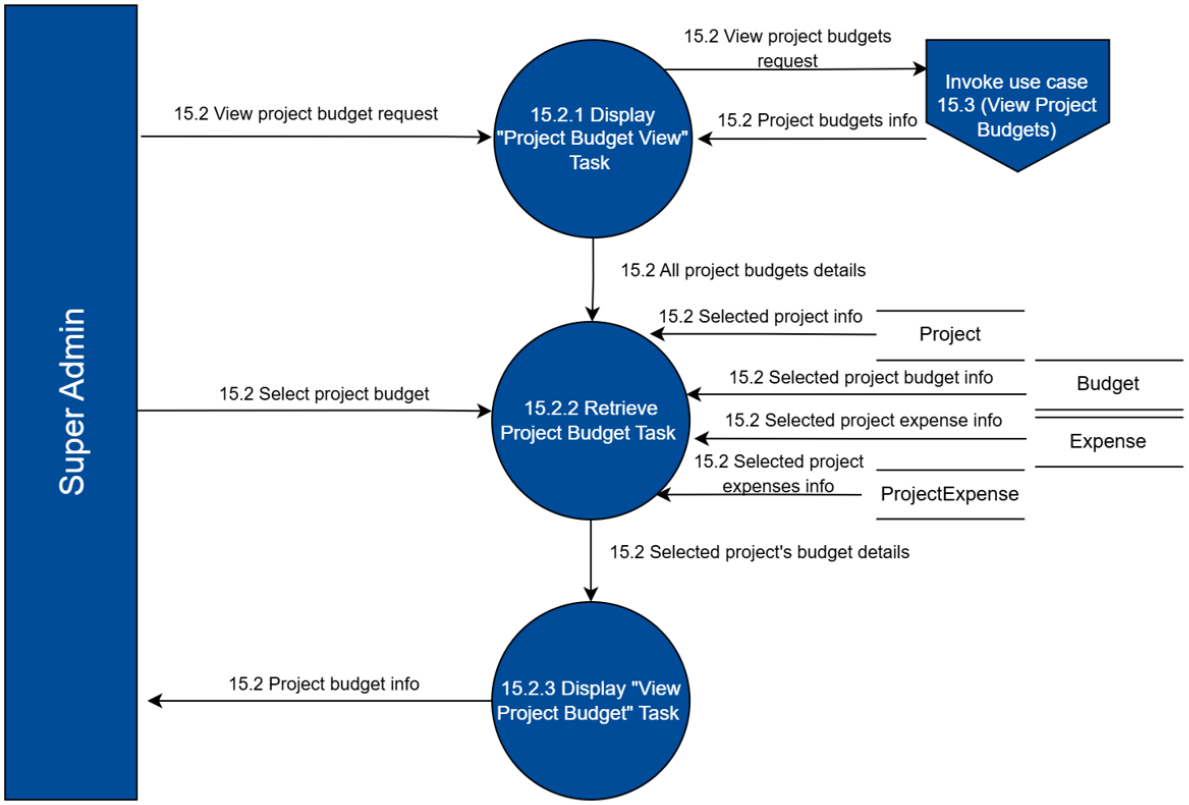


Figure 119 15.2 View Project Budget - Primitive Data Flow Diagram

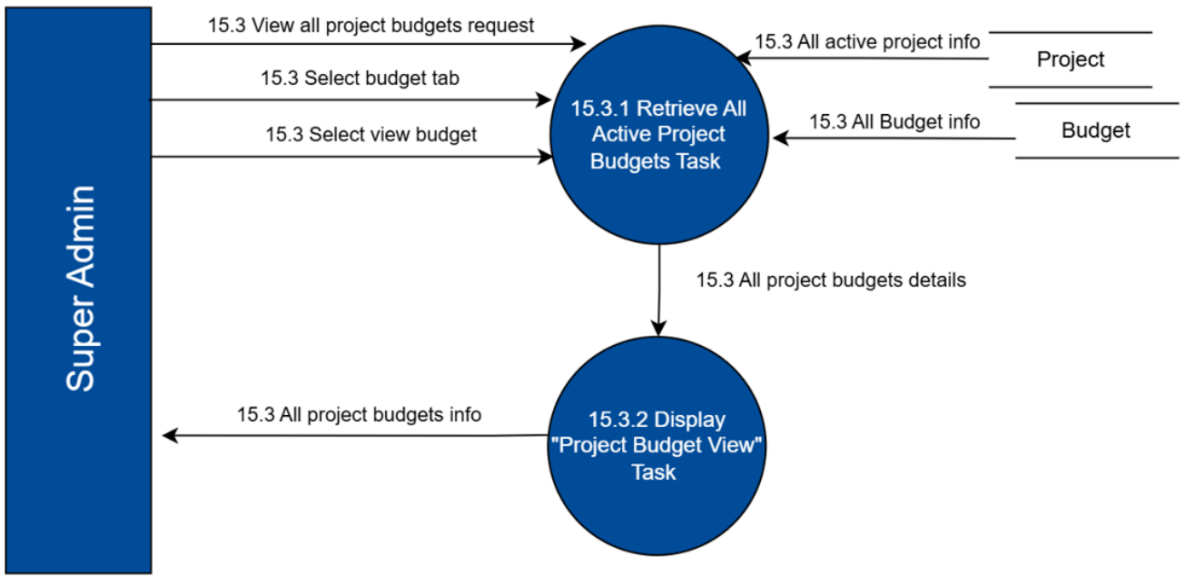


Figure 120 15.3 View Project Budgets - Primitive Data Flow Diagram

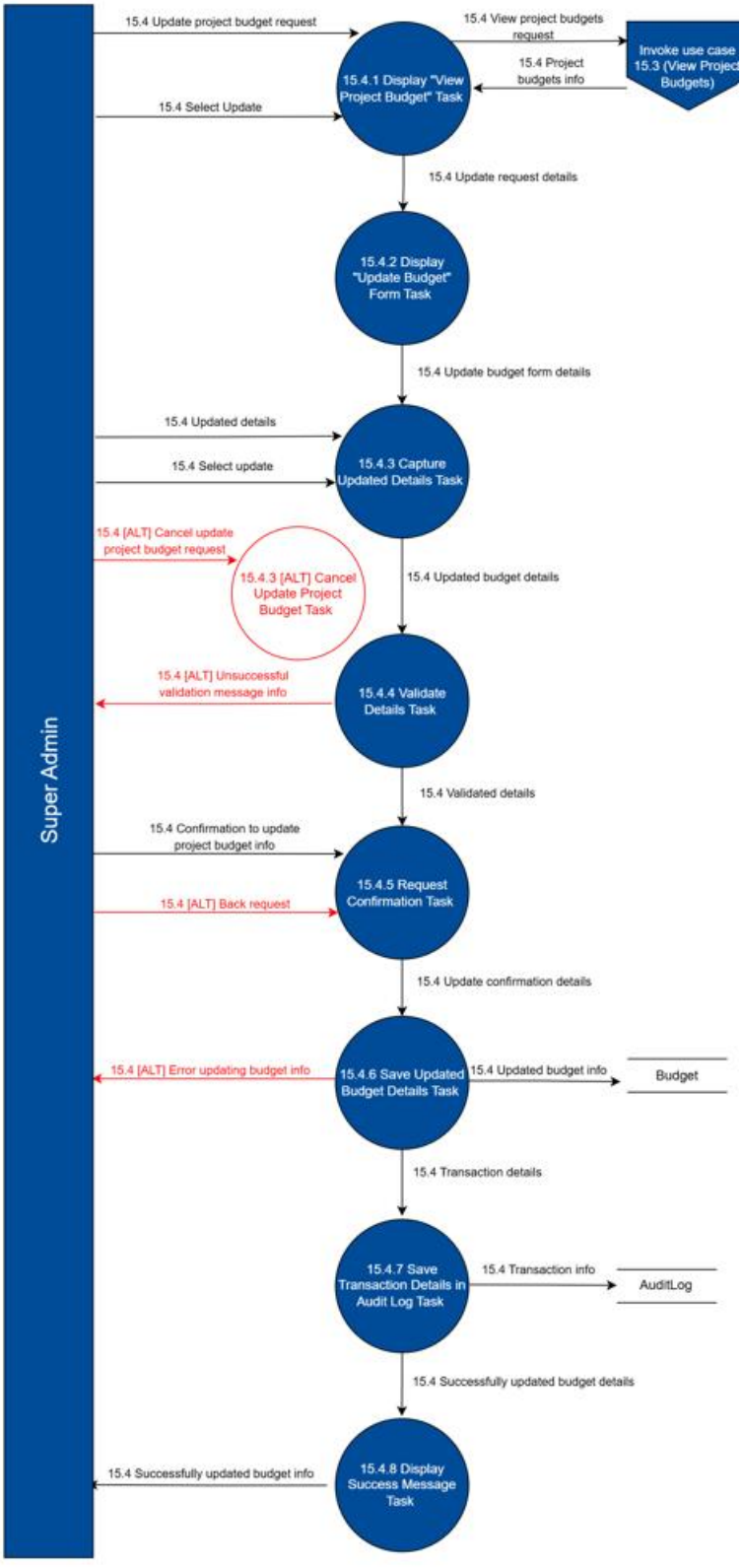


Figure 121 15.4 Update Project Budget - Primitive Data Flow Diagram

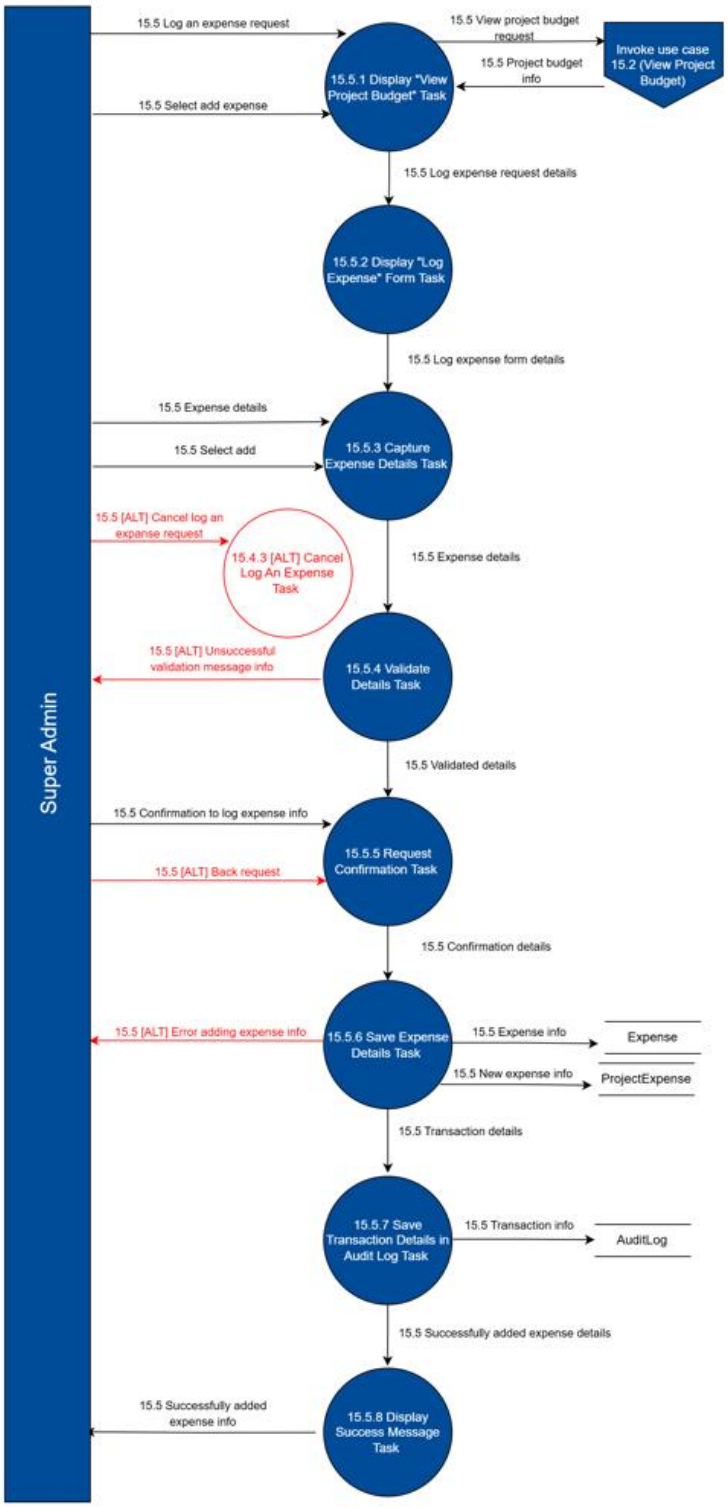


Figure 122 15.5 Log An Expense - Primitive Data Flow Diagram

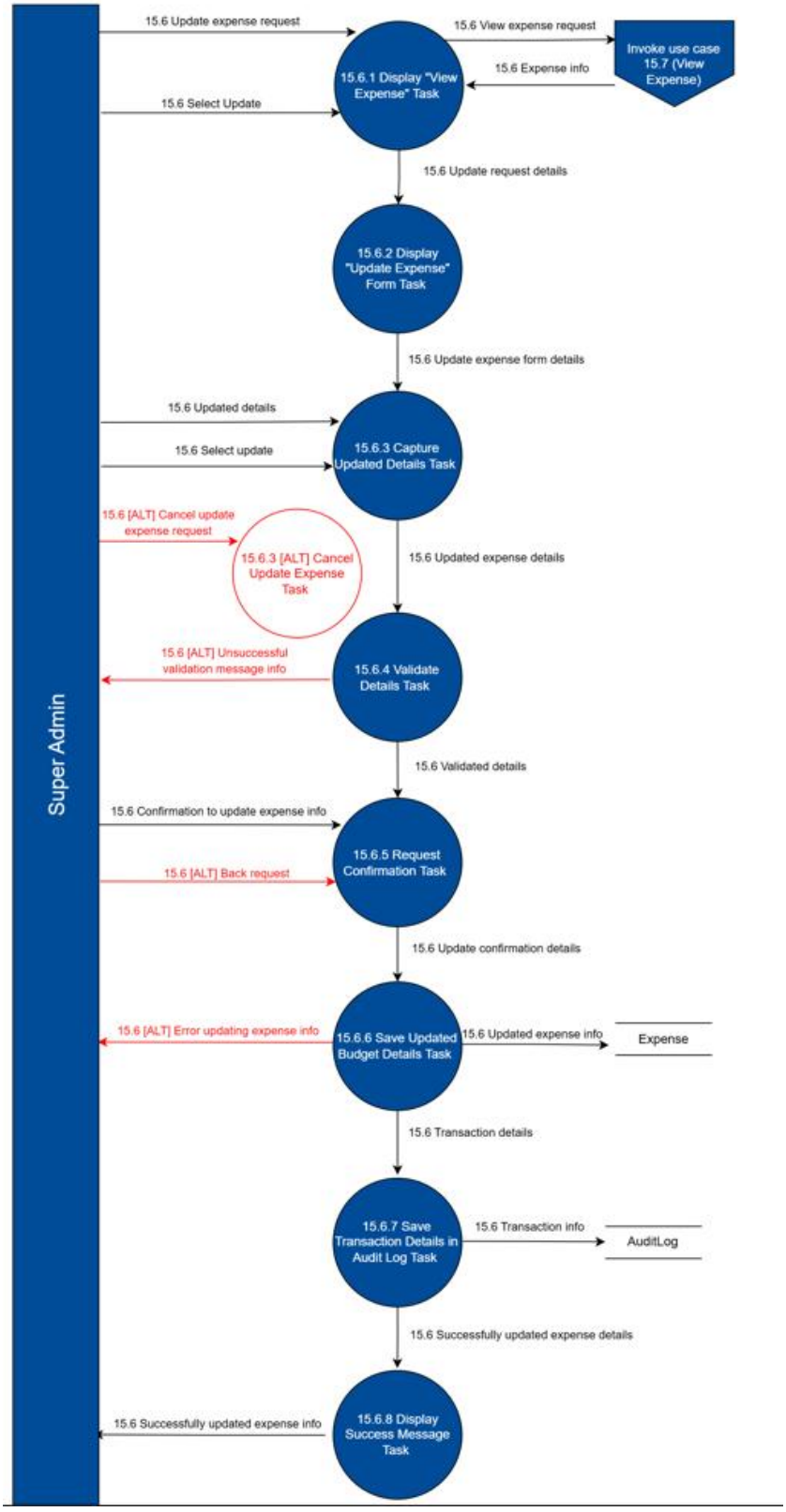


Figure 123 15.6 Update Expense - Primitive Data Flow Diagram

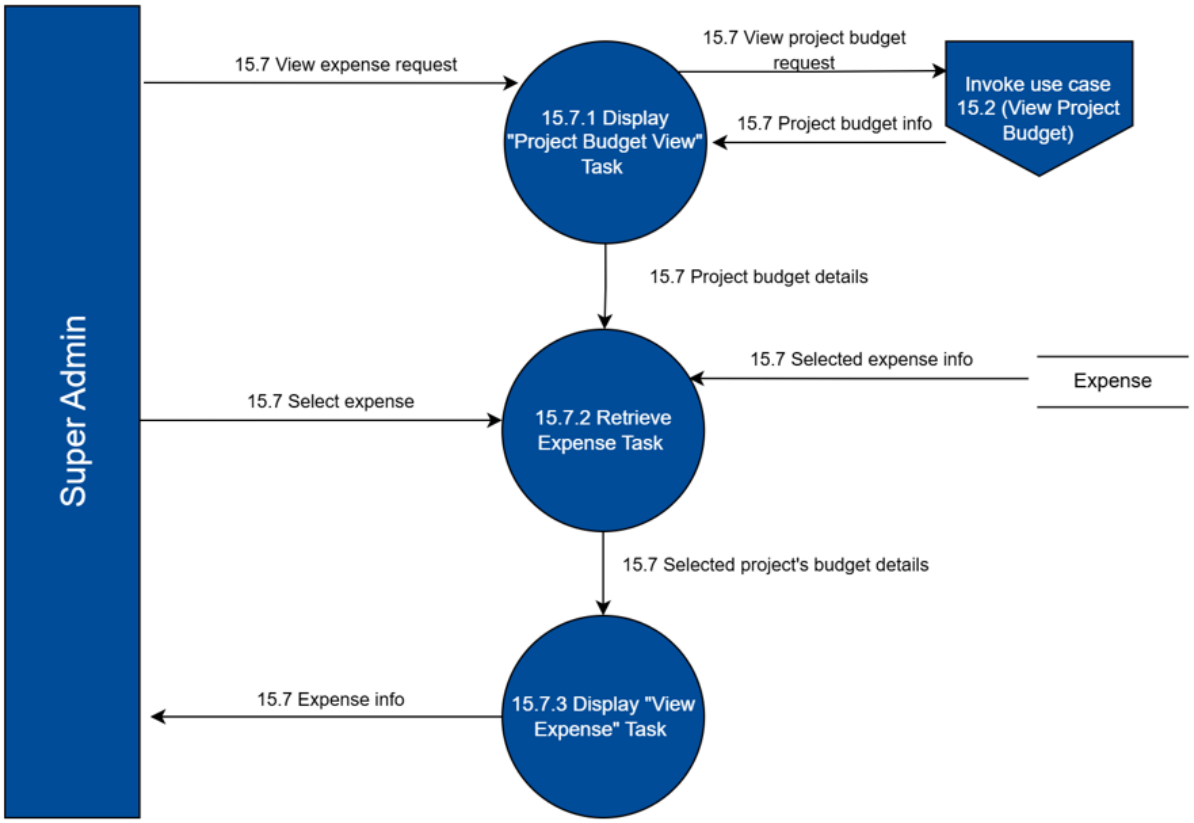


Figure 124 15.7 View Expense - Primitive Data Flow Diagram

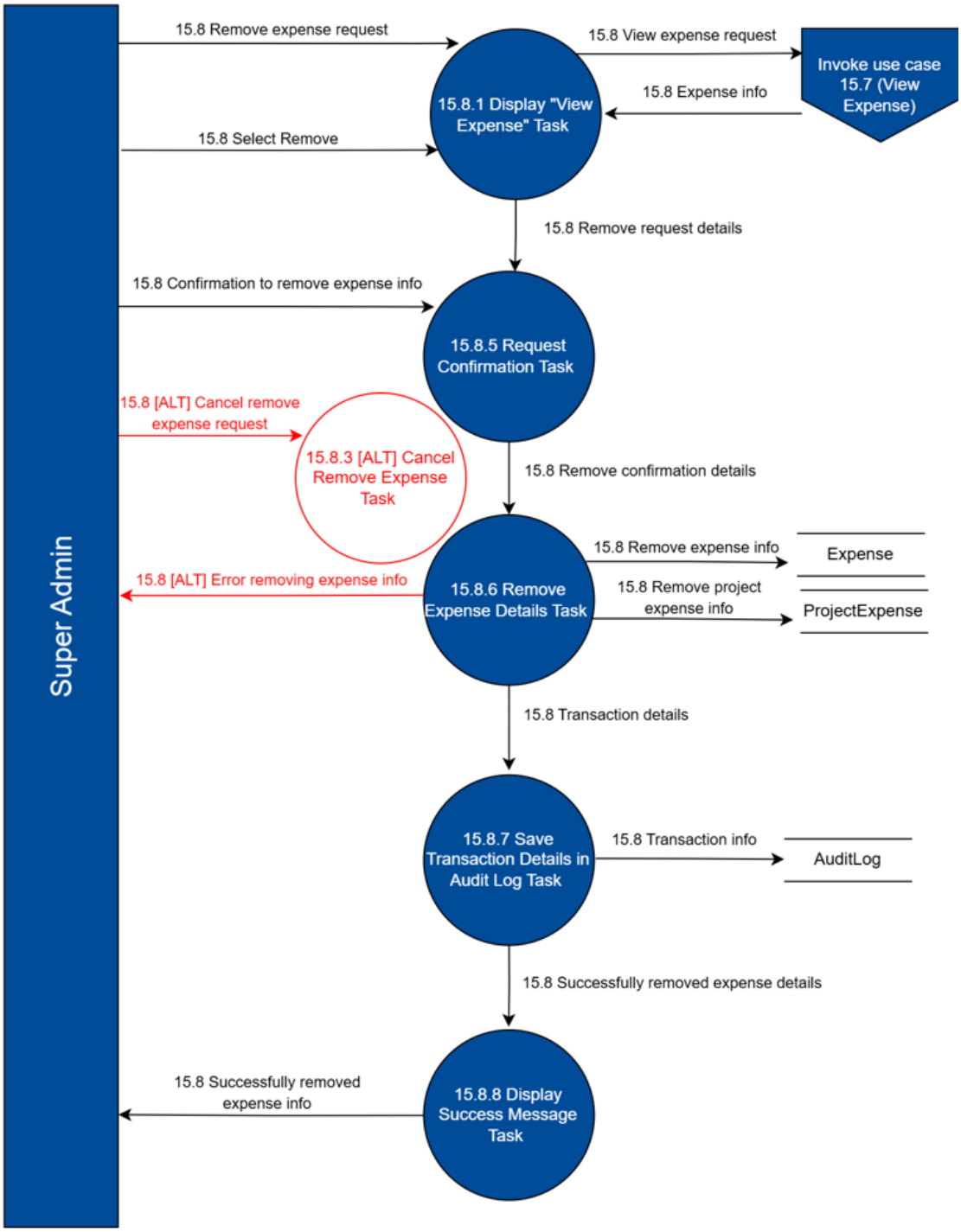


Figure 125 15.8 Remove Expense - Primitive Data Flow Diagram

2.10 Activity Diagrams

2.10.1 User Subsystem Activity Diagrams

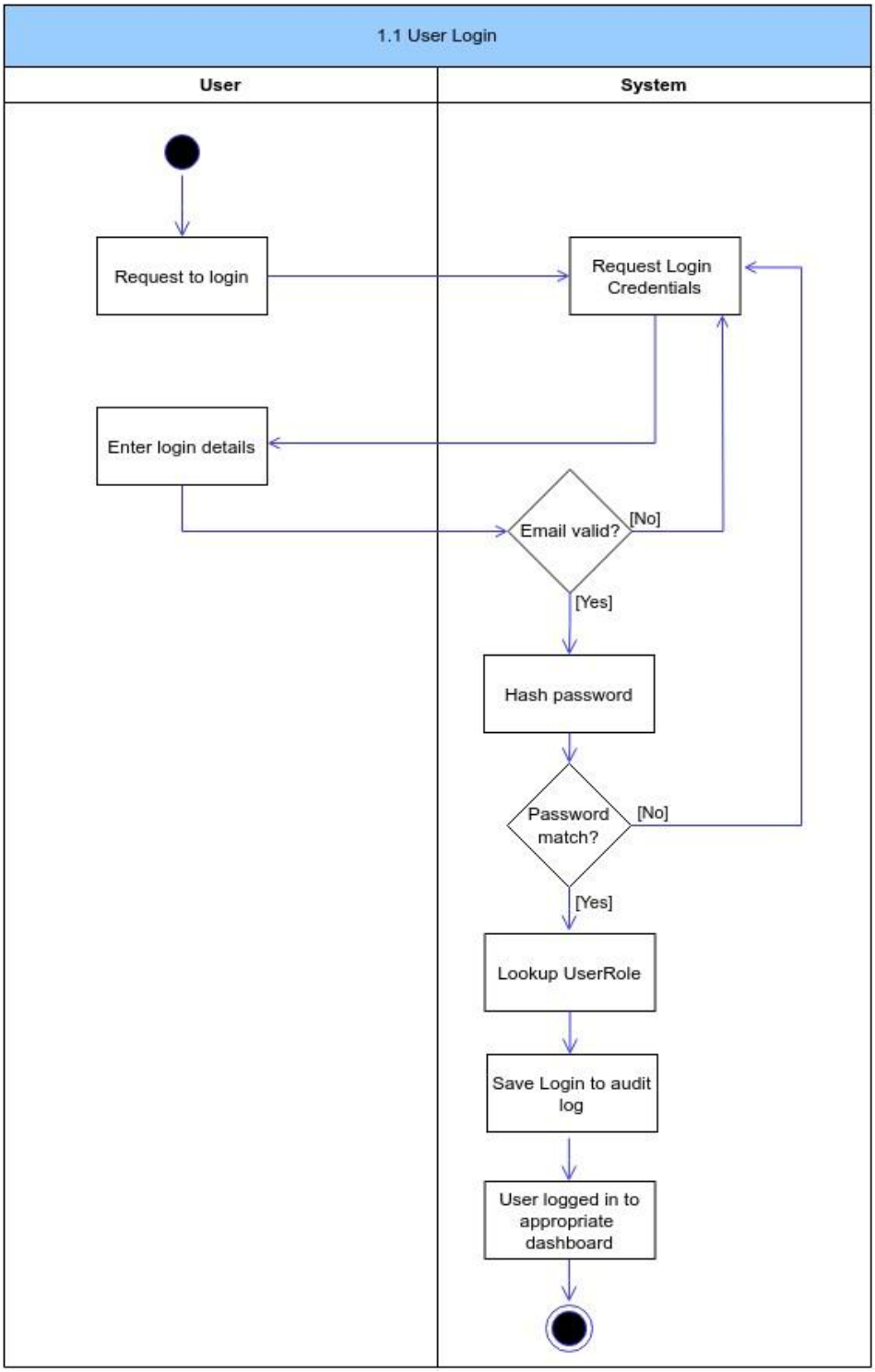


Figure 126 1.1 User Login - Activity Diagram

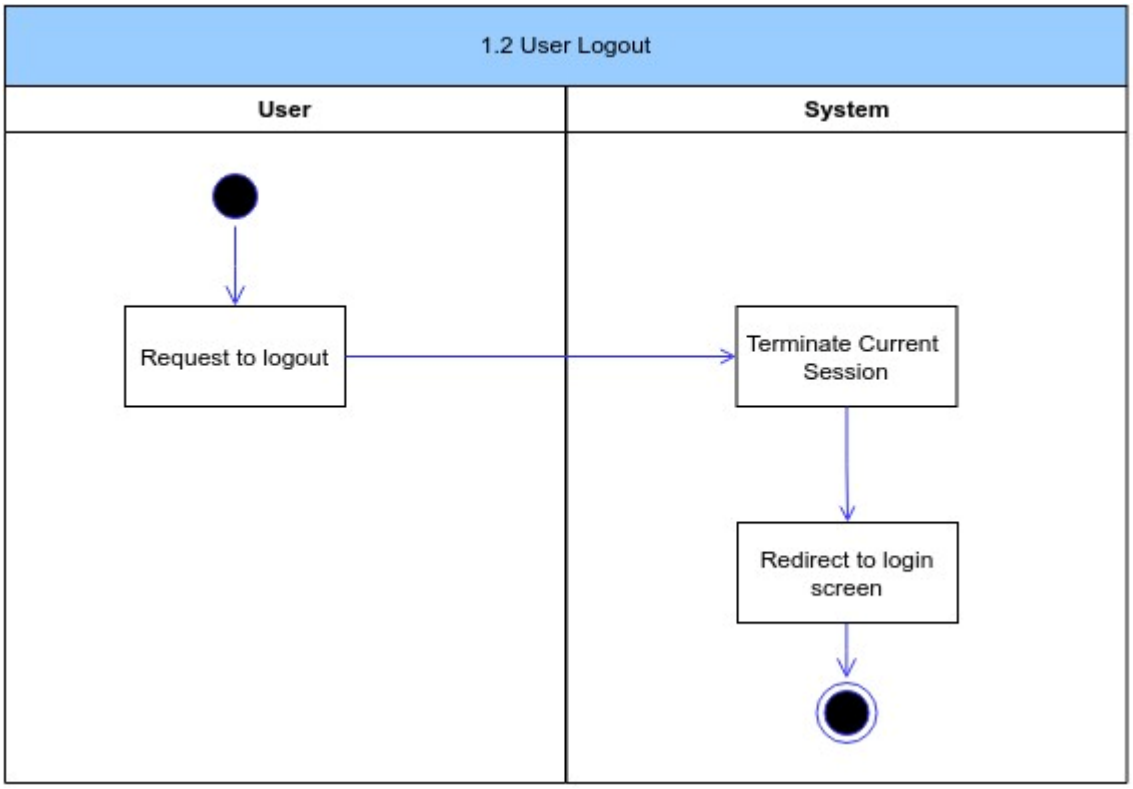


Figure 127 1.2 User Logout - Activity Diagram

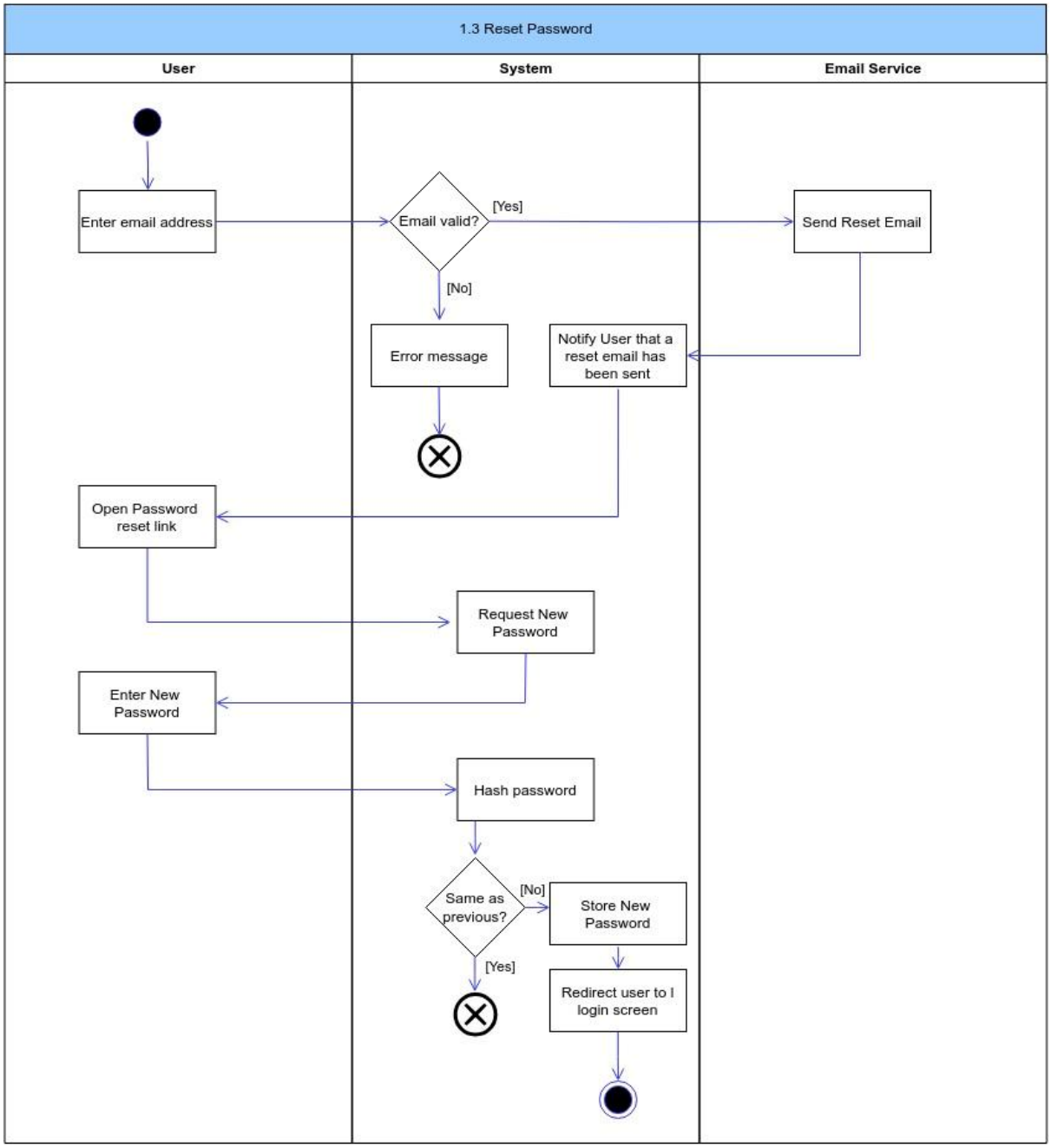


Figure 128 1.3 Reset Password - Activity Diagram

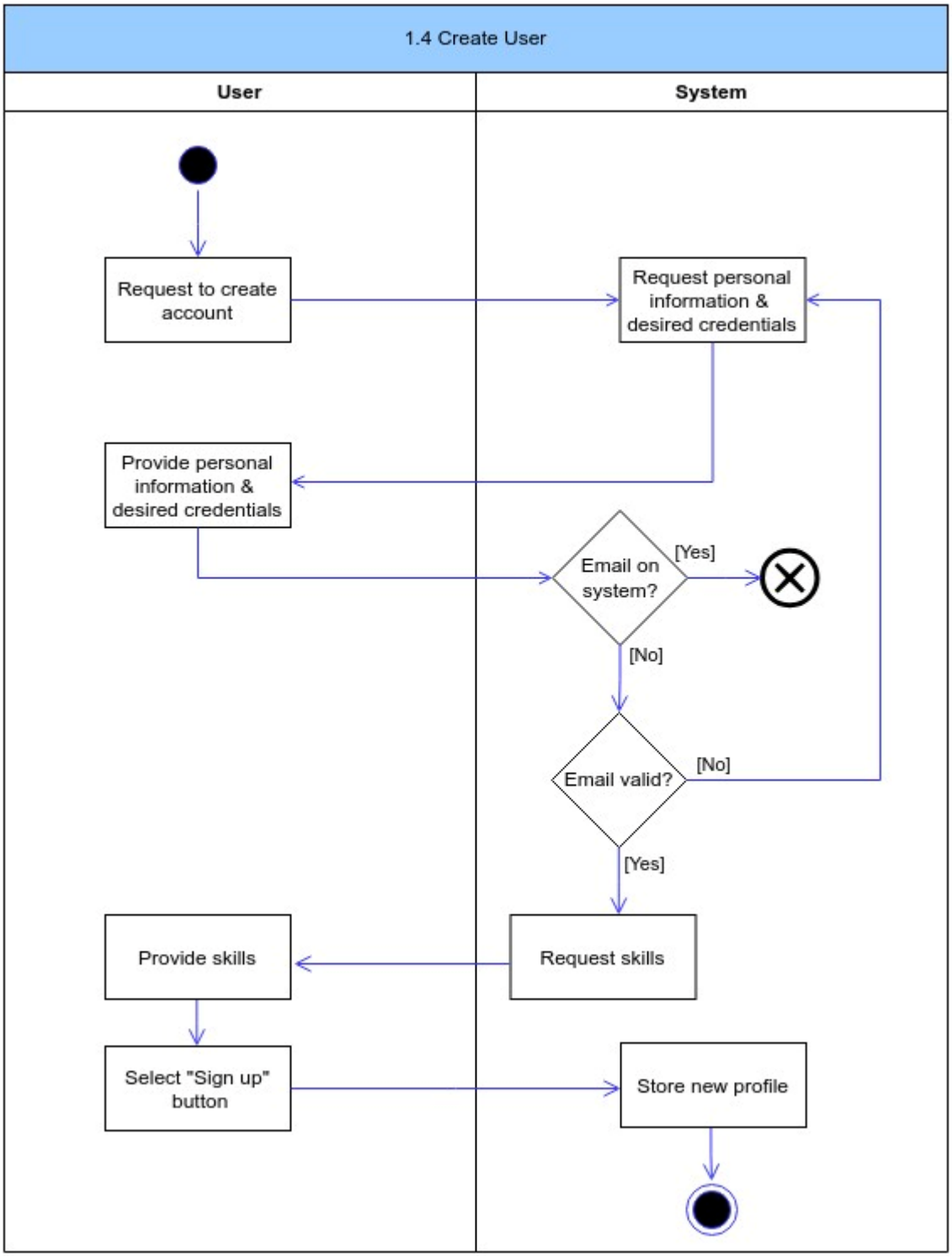


Figure 129 1.4 Create User - Activity Diagram

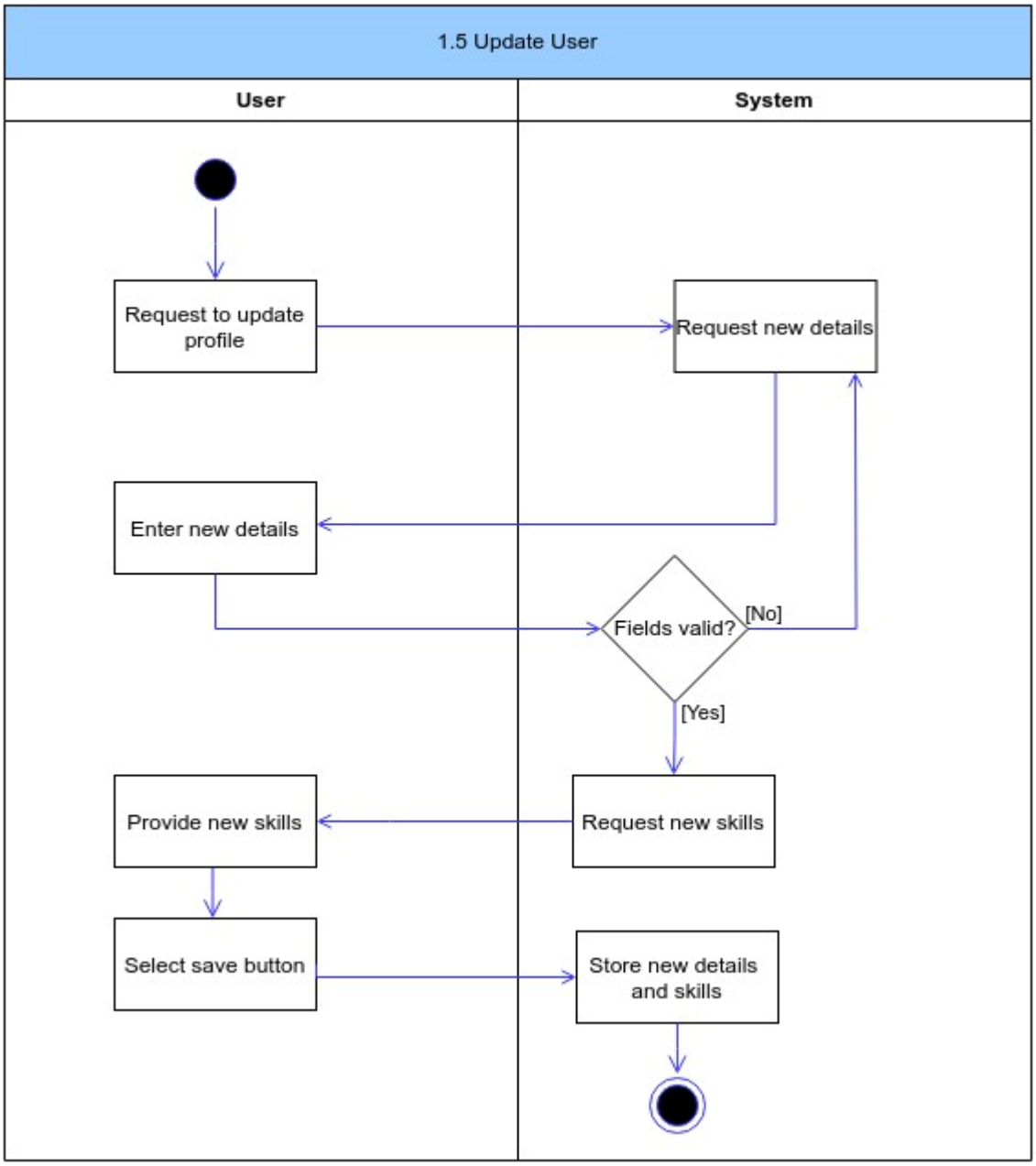


Figure 130 1.5 Update User - Activity Diagram

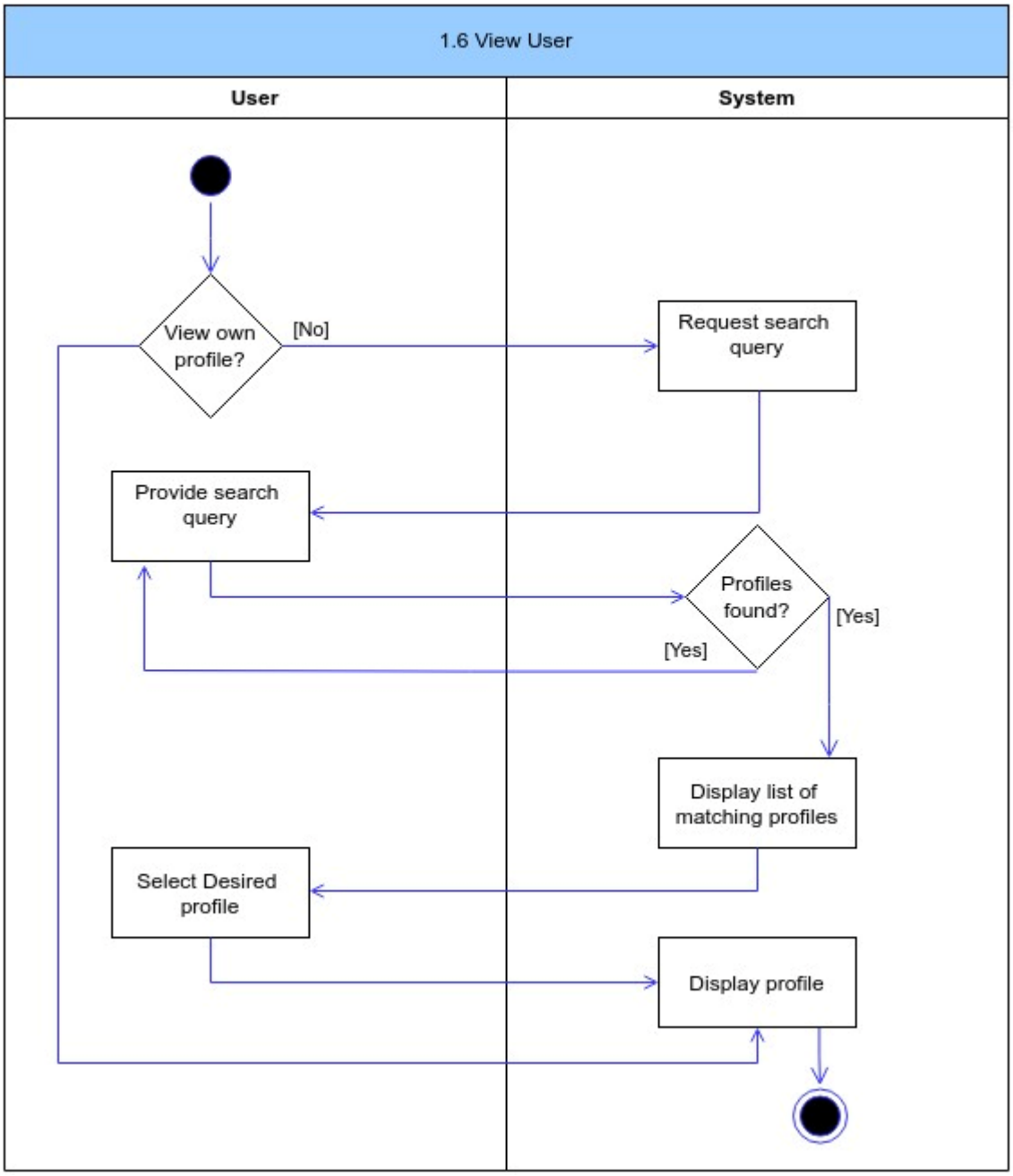


Figure 131 1.6 View User - Activity Diagram

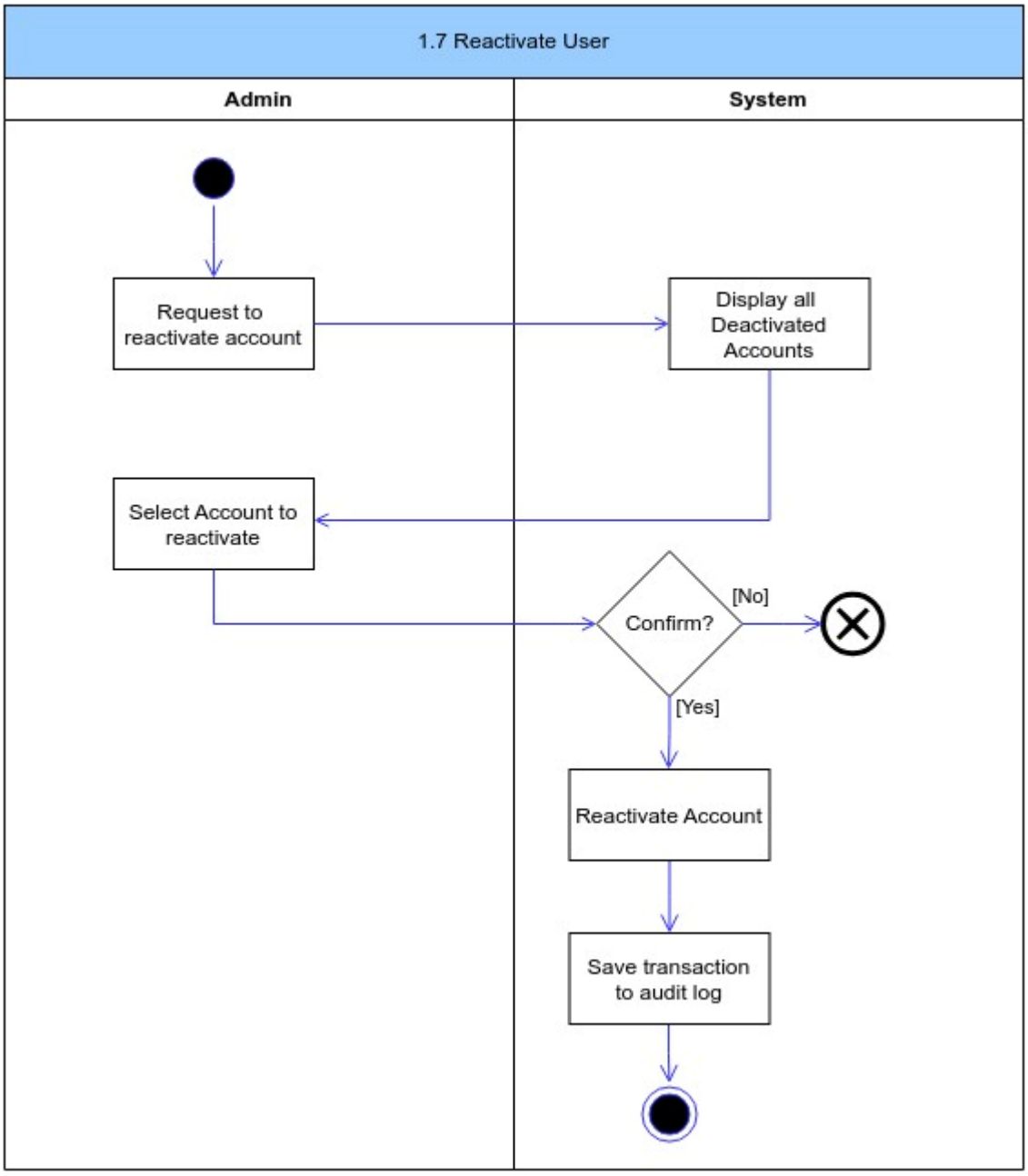


Figure 132 1.7 Reactivate User - Activity Diagram

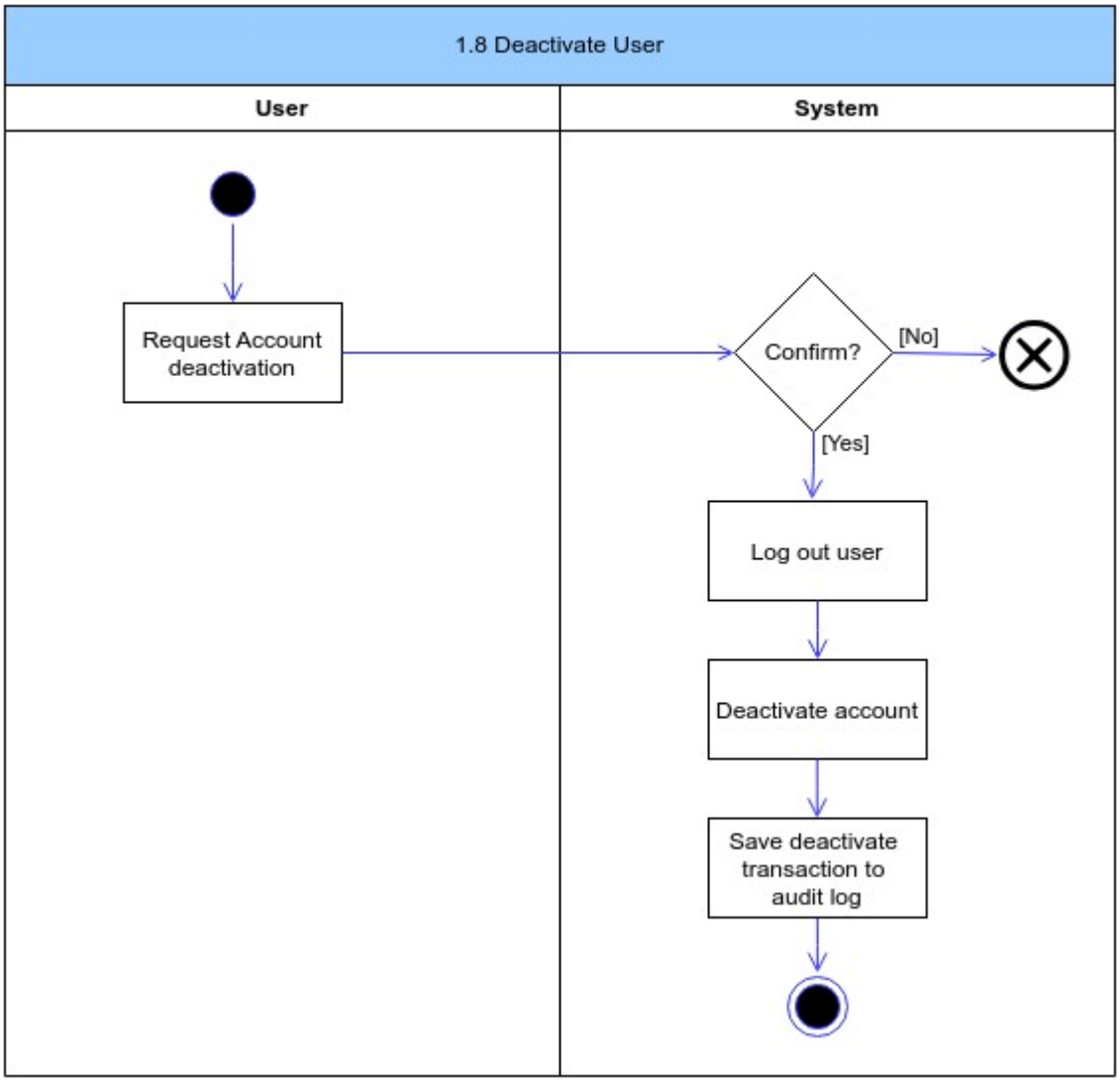


Figure 133 1.8 Deactivate User - Activity Diagram

2.10.2 Admin Subsystem Activity Diagrams

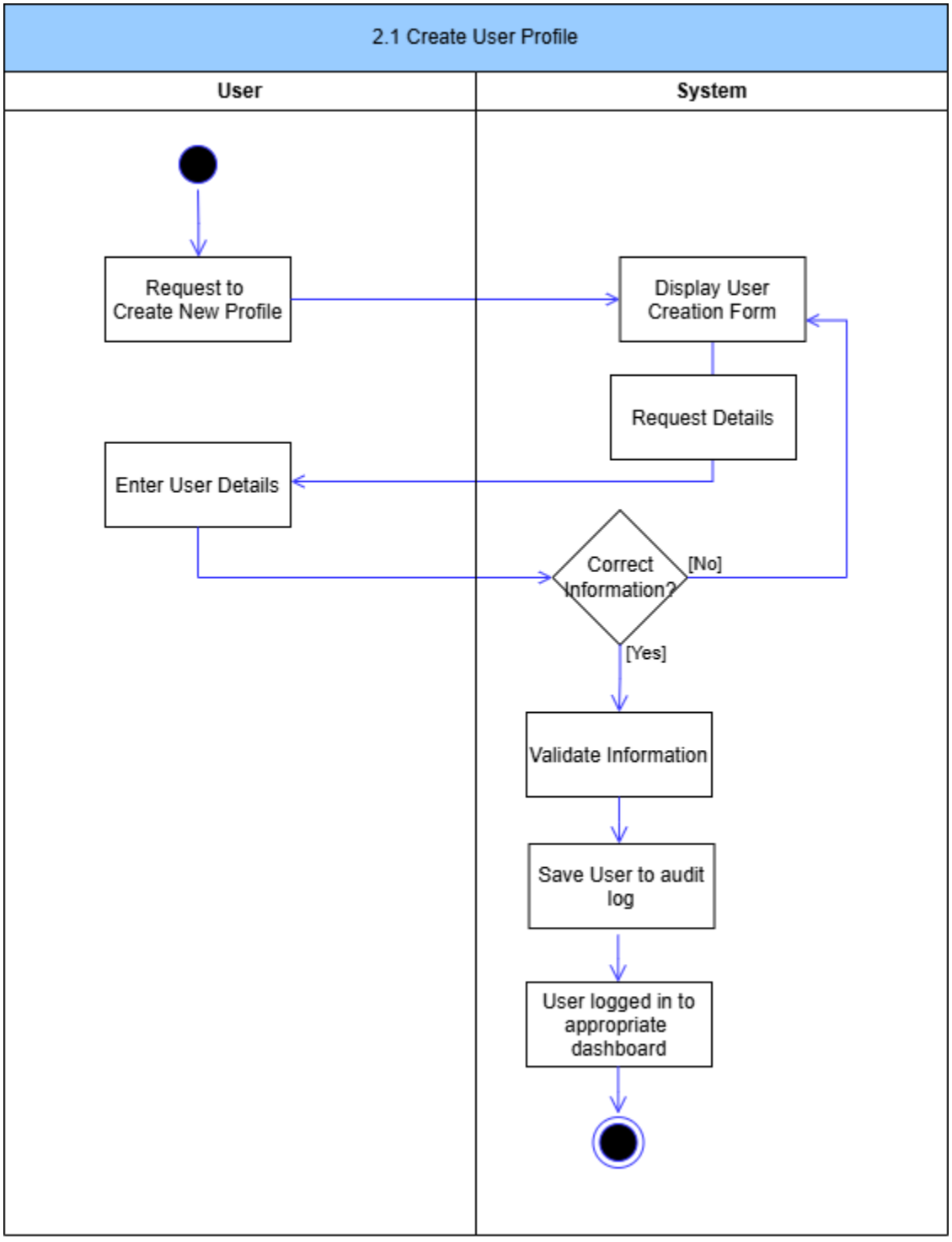


Figure 134 2.1 Create User Profile - Activity Diagram

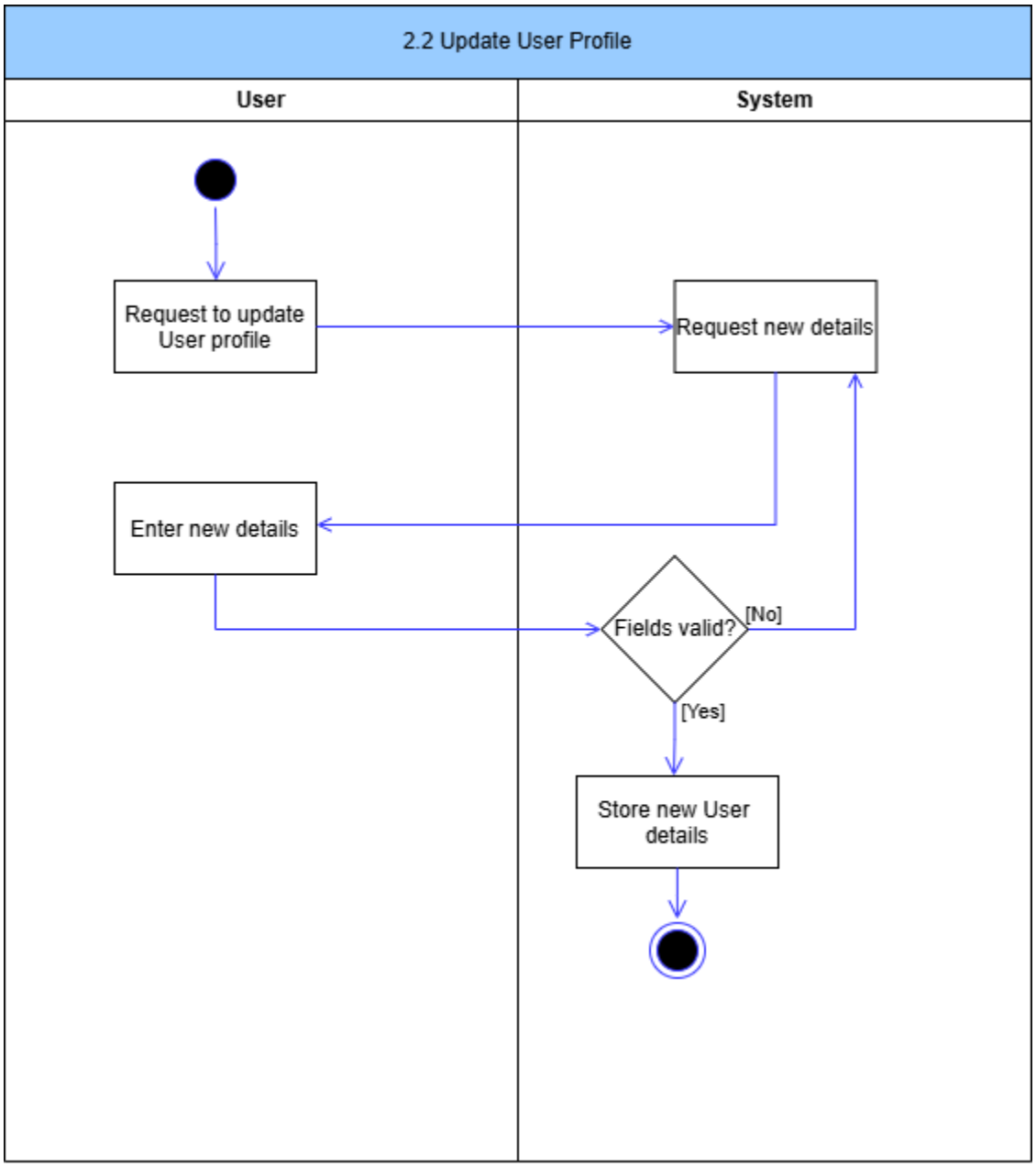


Figure 135 2.2 Update User Profile - Activity Diagram

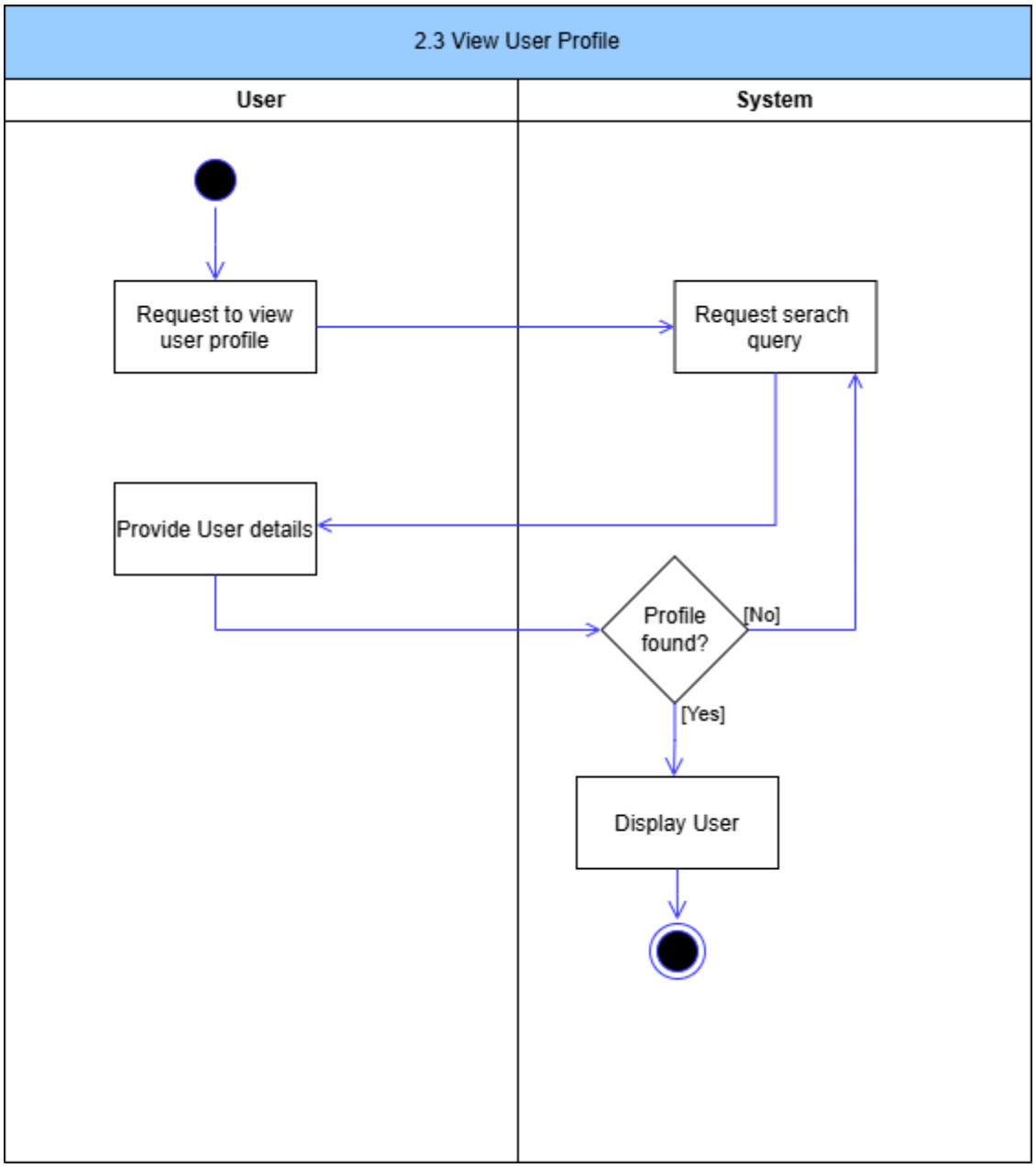


Figure 136 2.3 View User Profile - Activity Diagram

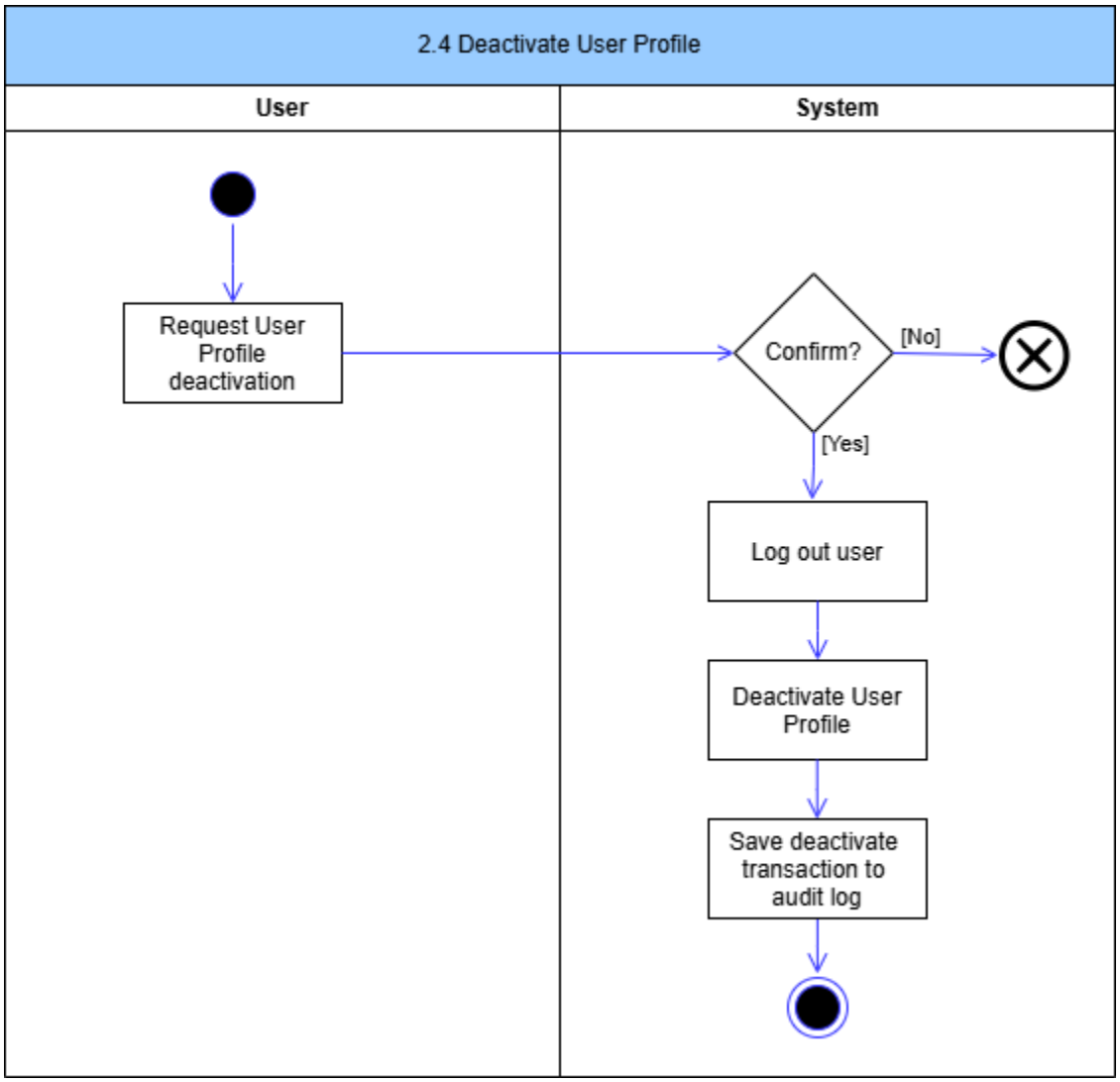


Figure 137 2.4 Deactivate User Profile - Activity Diagram

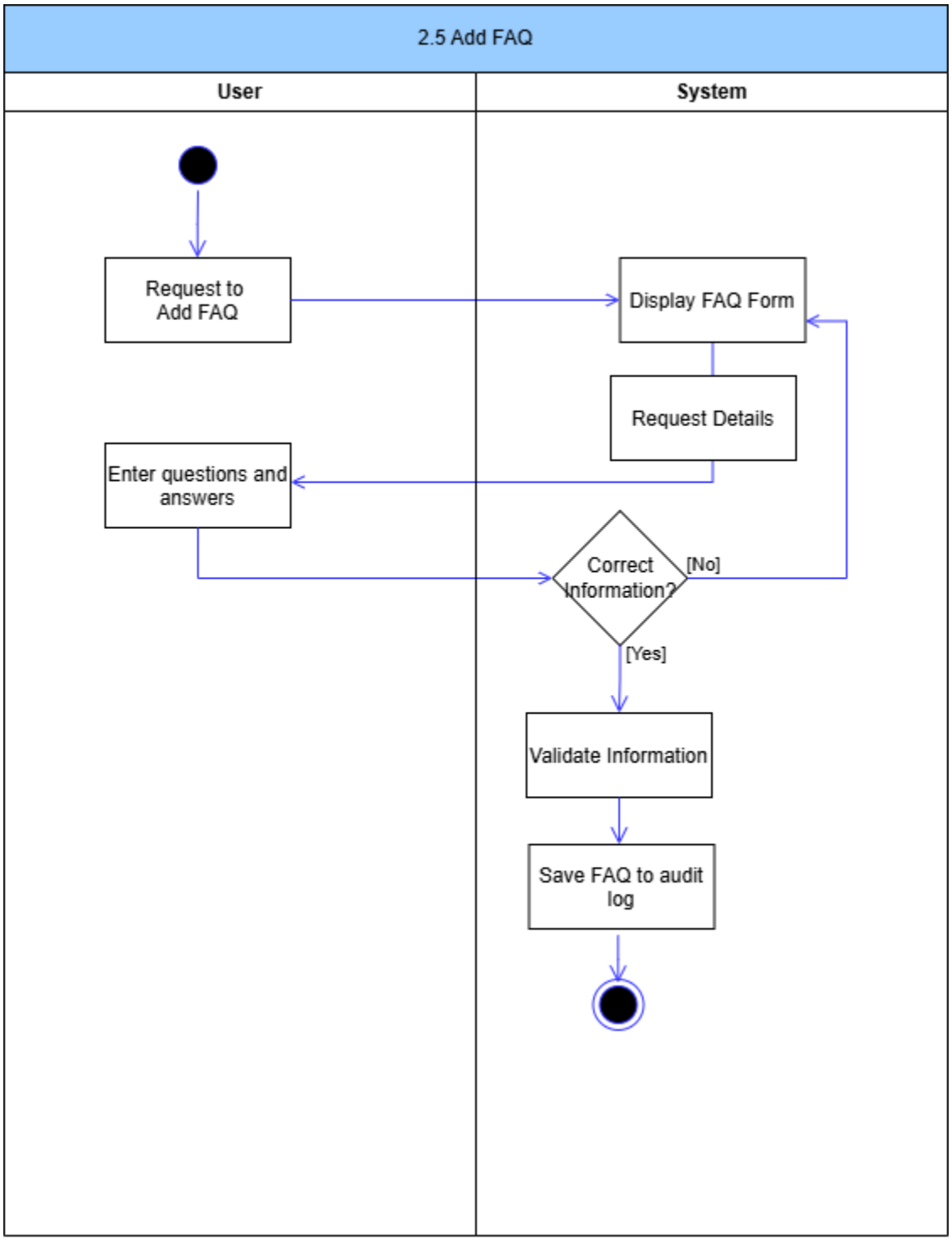


Figure 138 2.5 Add FAQ - Activity Diagram

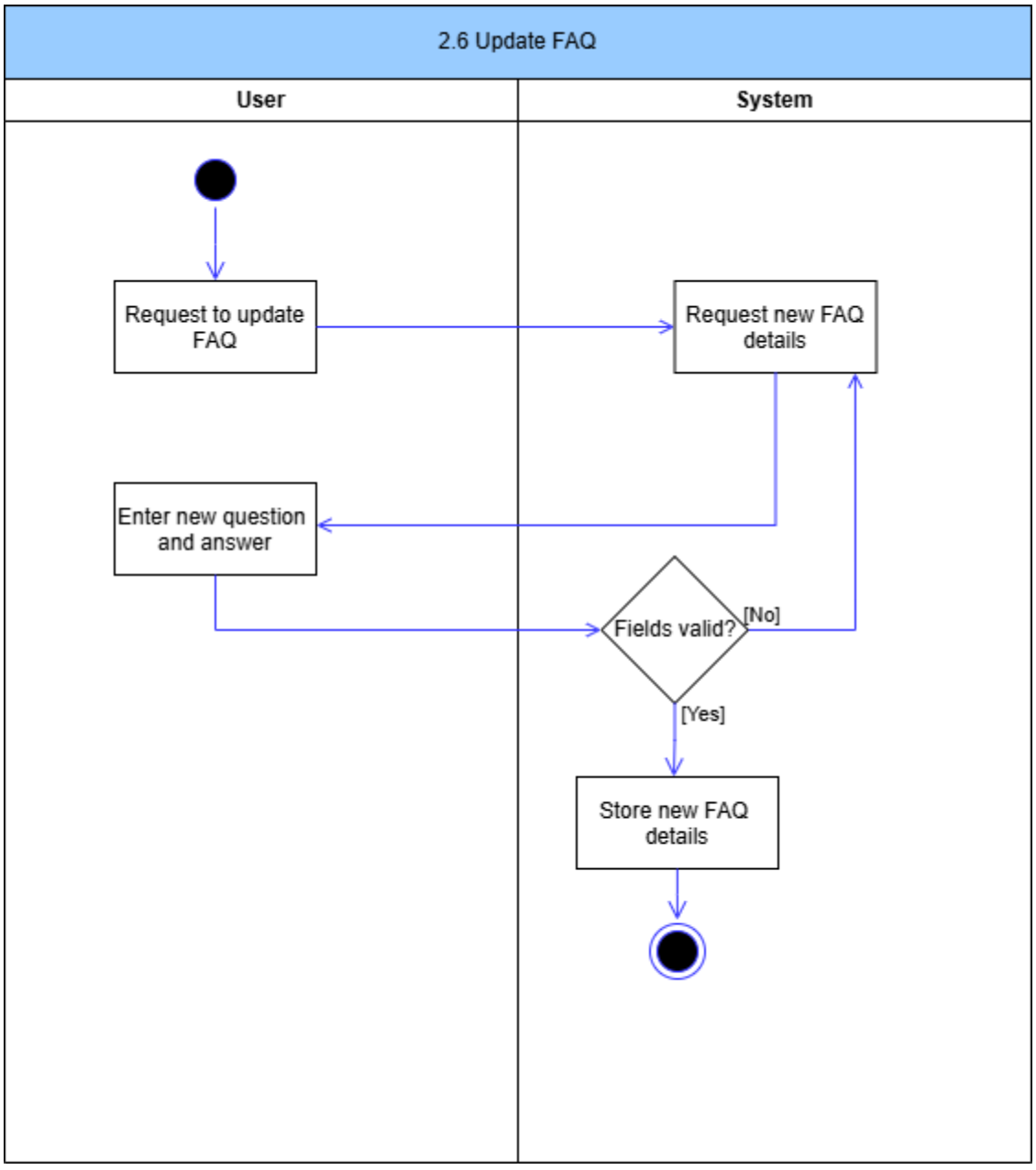


Figure 139 2.6 Update FAQ - Activity Diagram

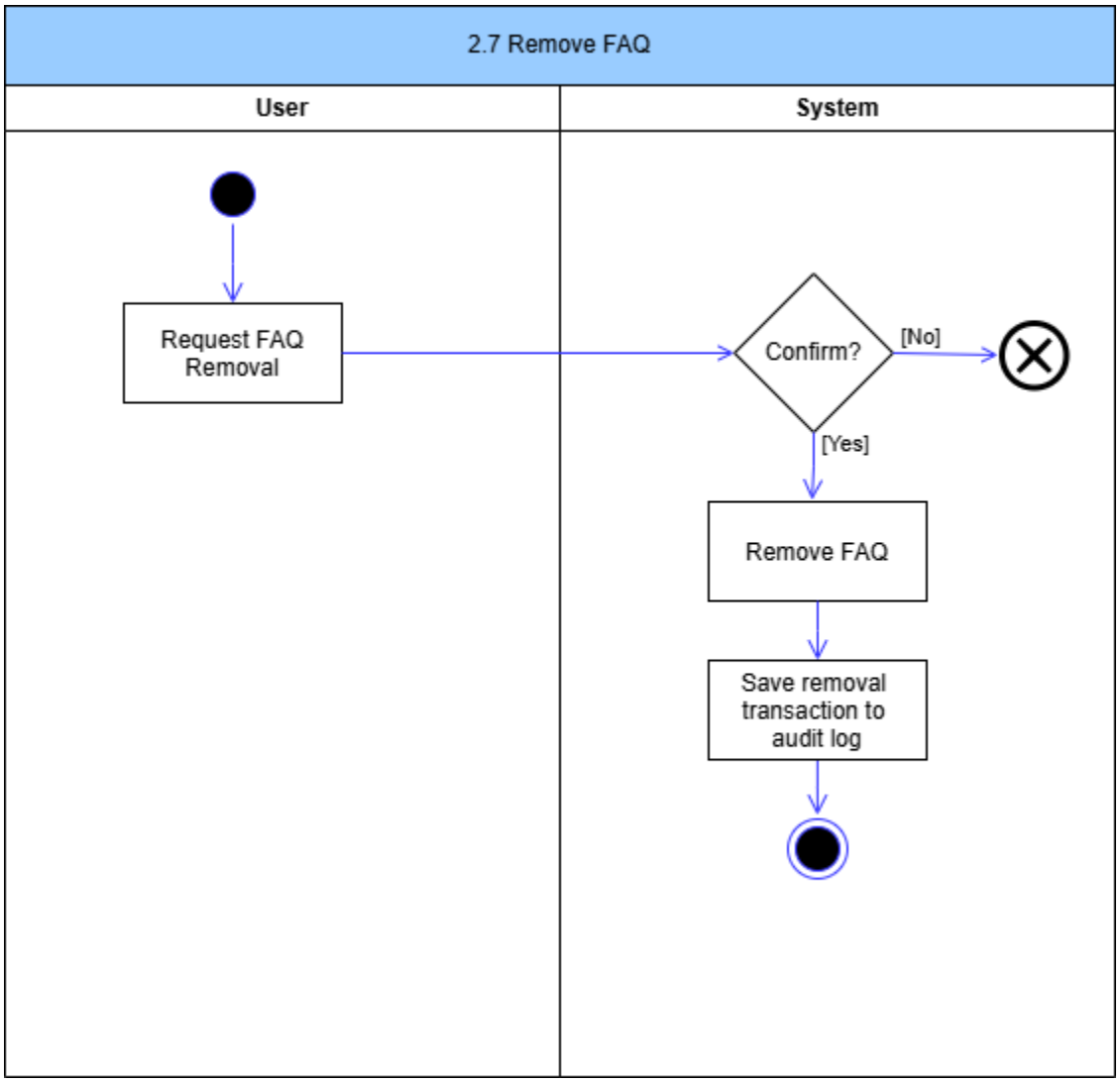


Figure 140 2.7 Remove FAQ - Activity Diagram

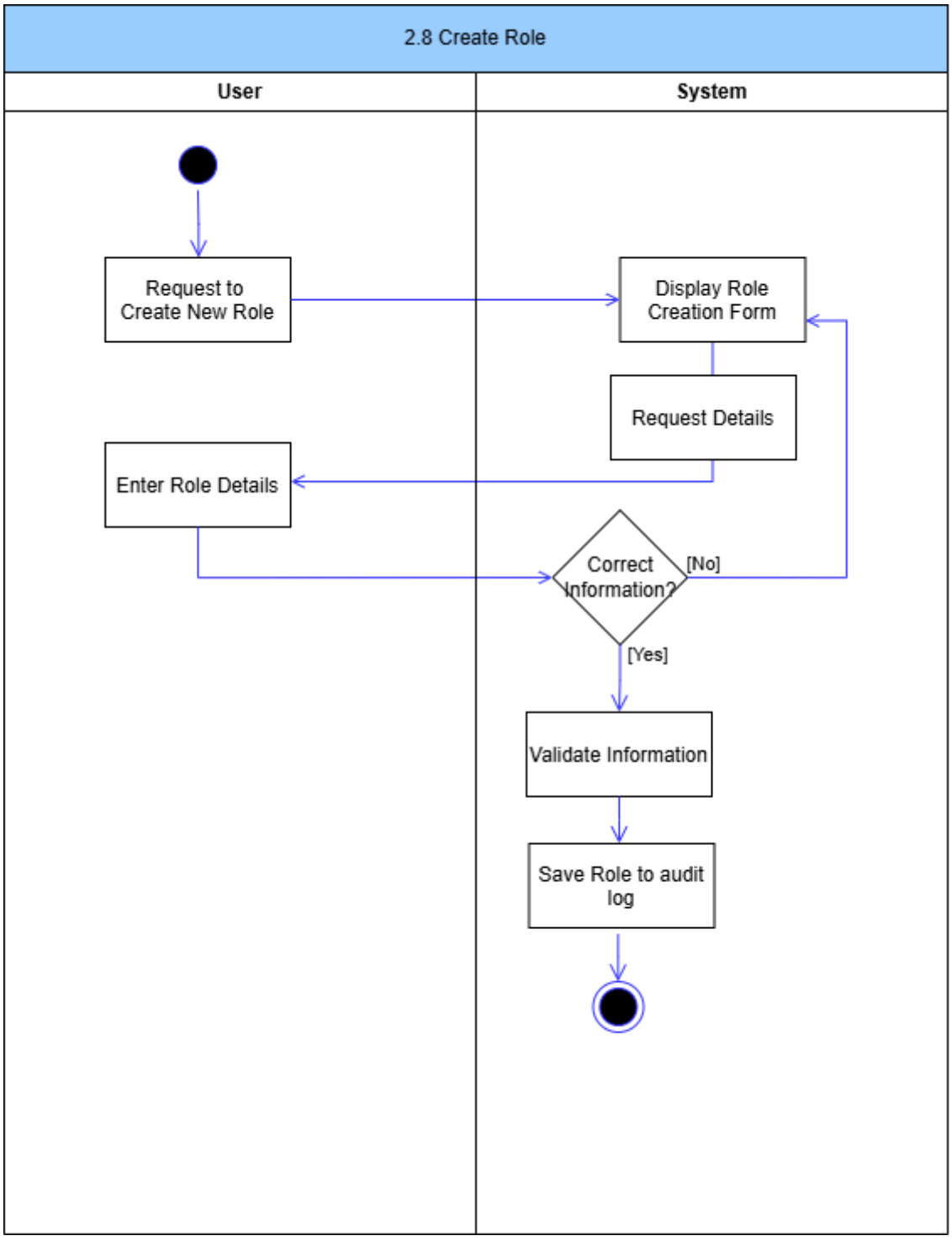


Figure 141 2.8 Create Role - Activity Diagram

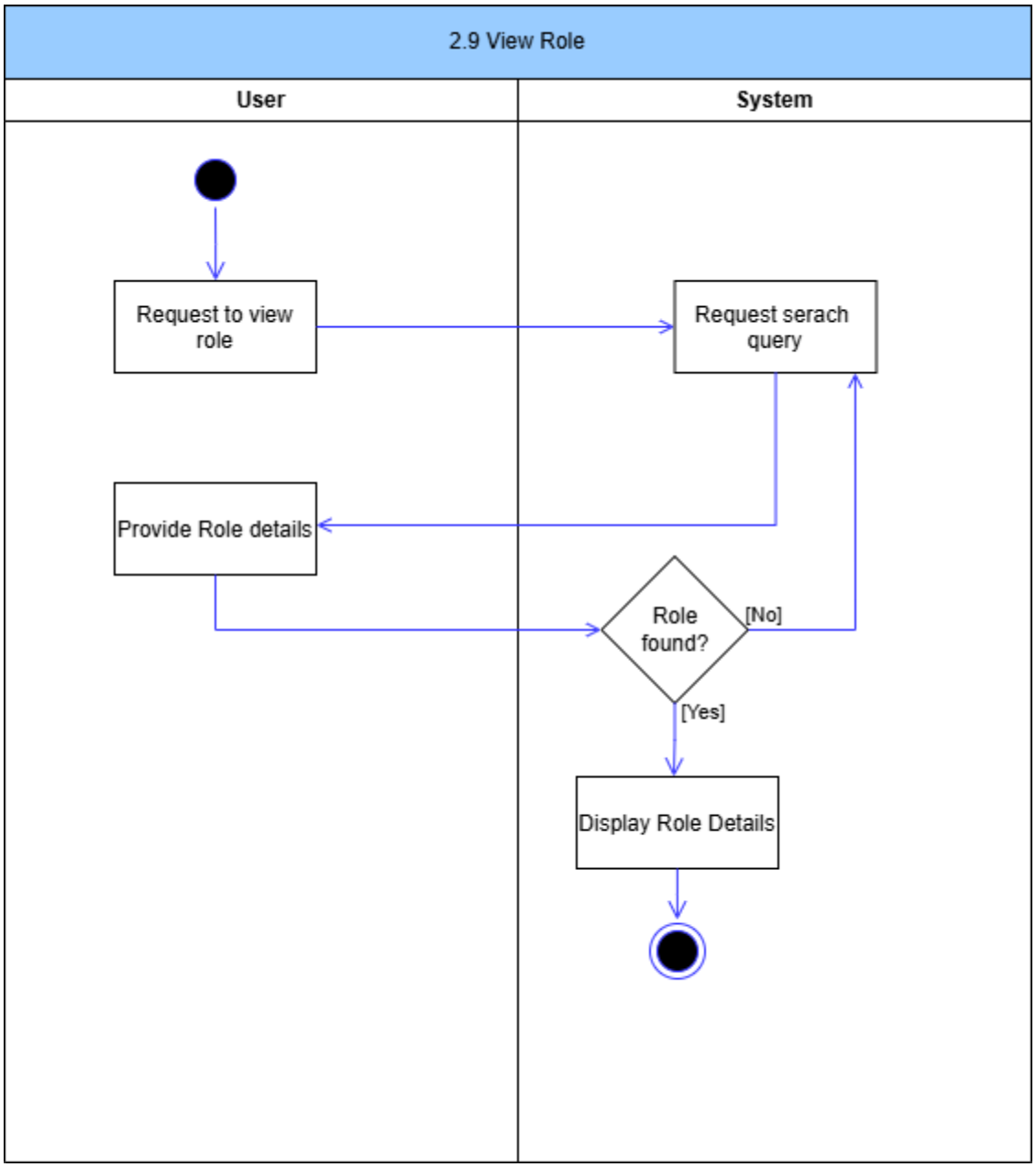


Figure 142 2.9 View Role - Activity Diagram

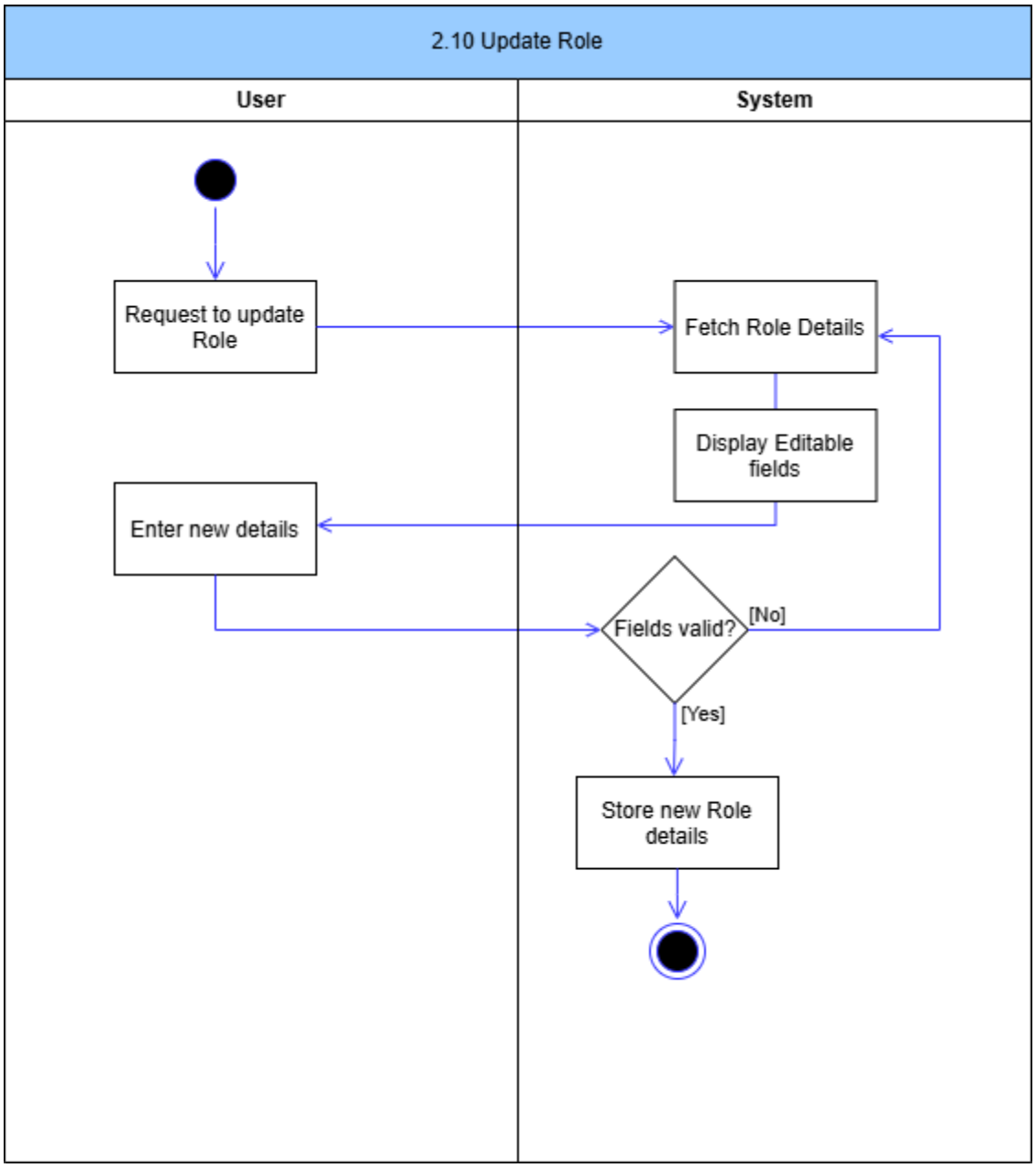


Figure 143 2.10 Update Role - Activity Diagram

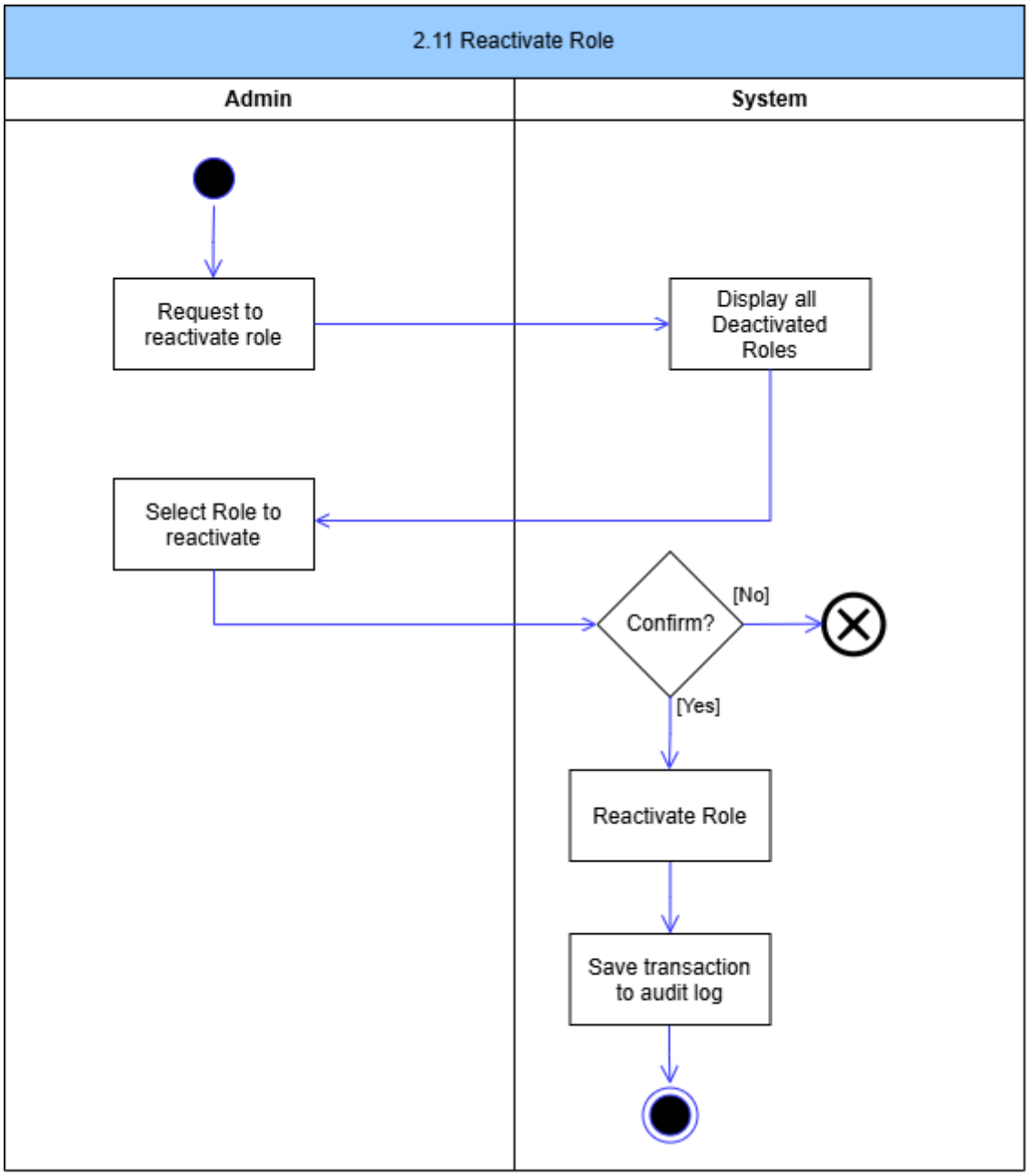


Figure 144 2.11 Reactivate Role - Activity Diagram

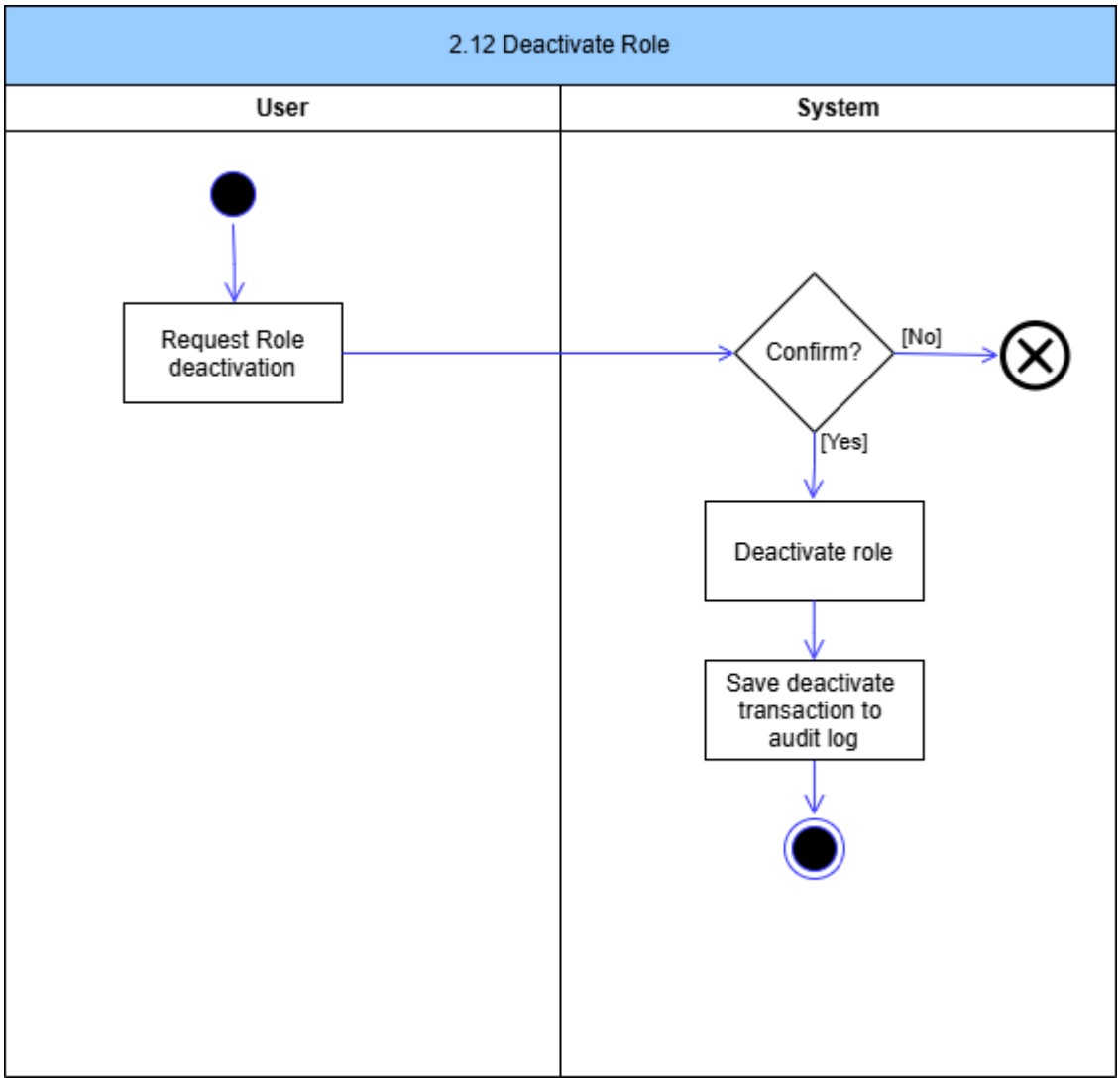


Figure 145 2.12 Deactivate Role - Activity Diagram

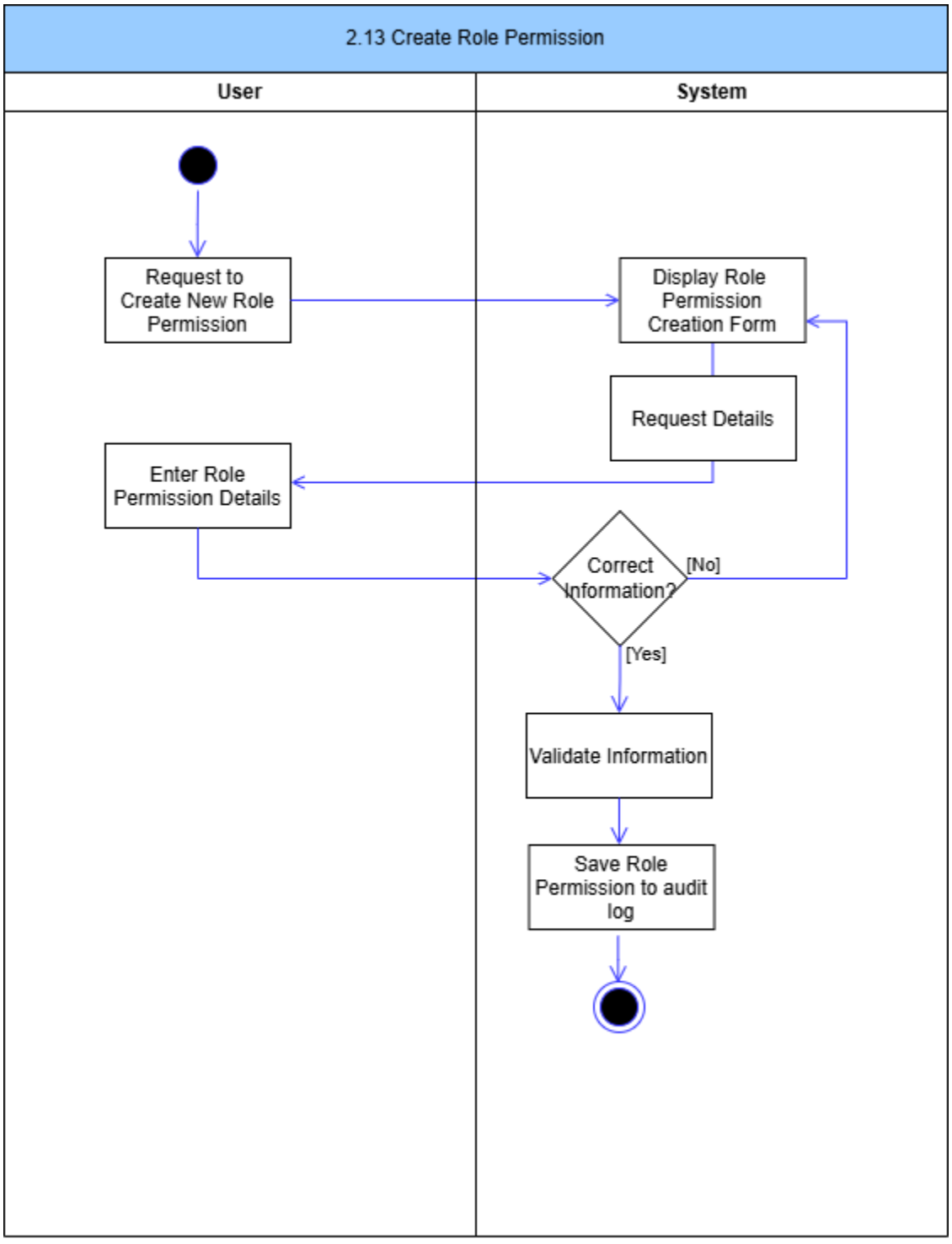


Figure 146 2.13 Create Role Permission - Activity Diagram

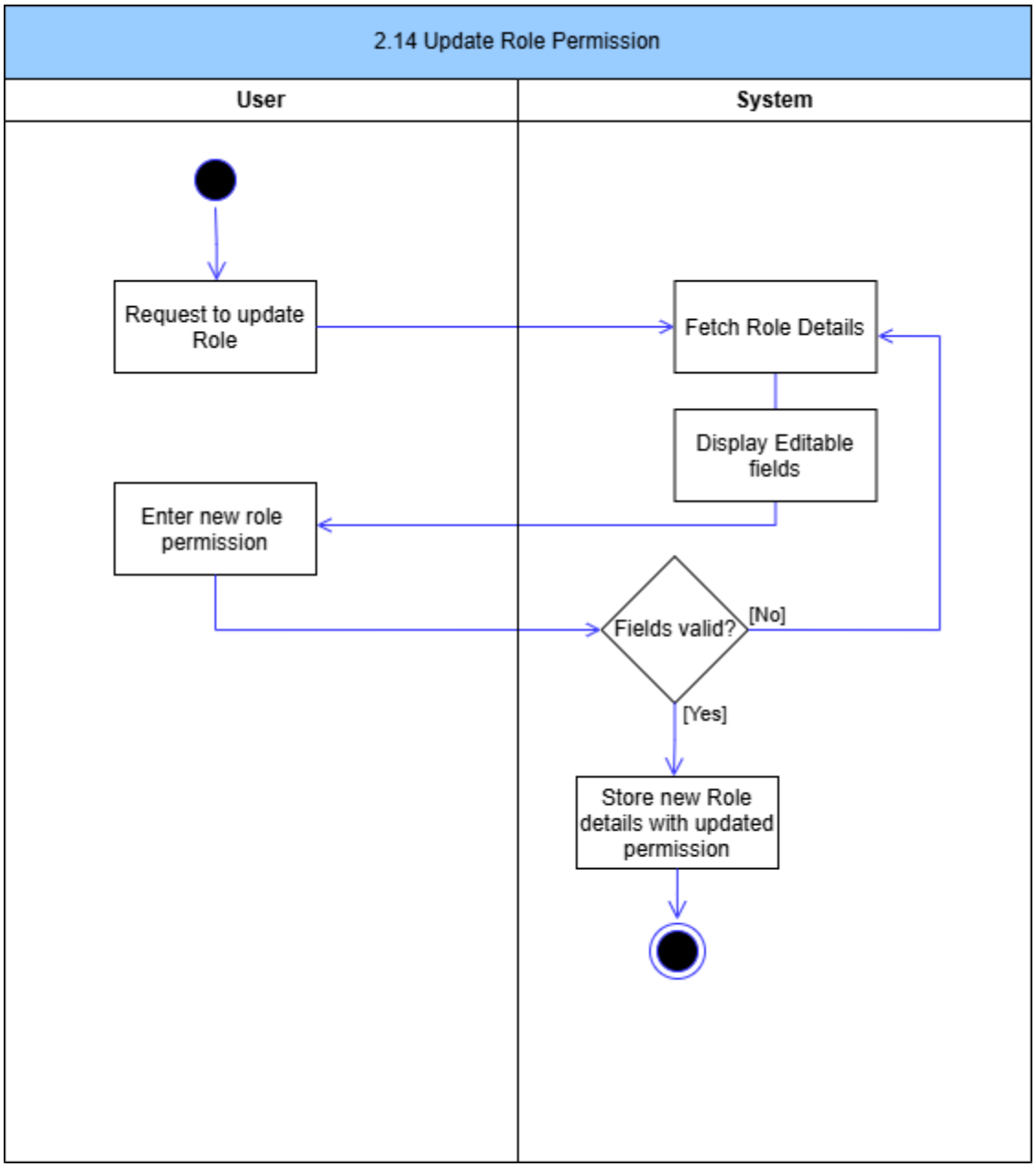


Figure 147 2.14 Update Role Permission - Activity Diagram

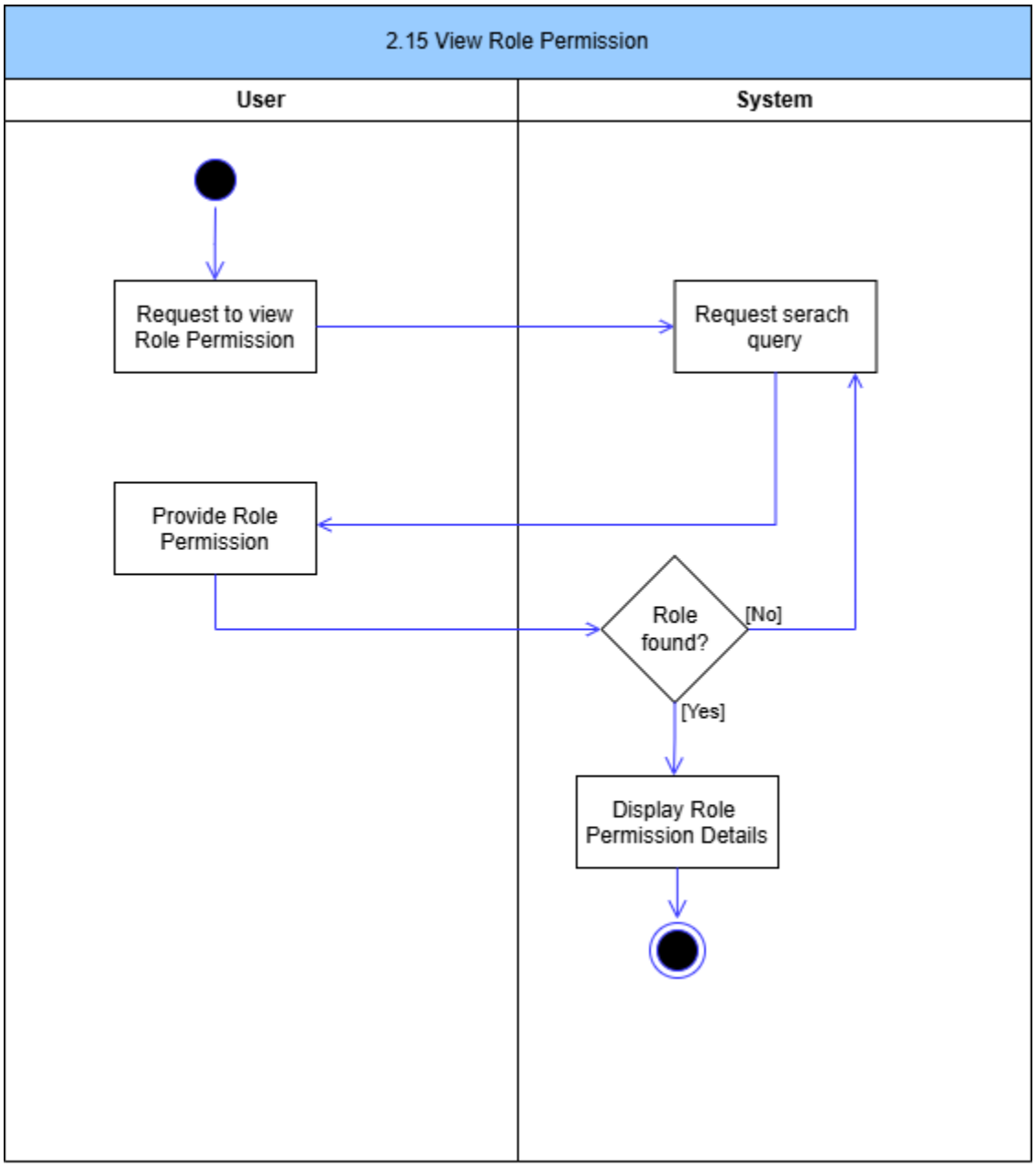


Figure 148 2.15 View Role Permission - Activity Diagram

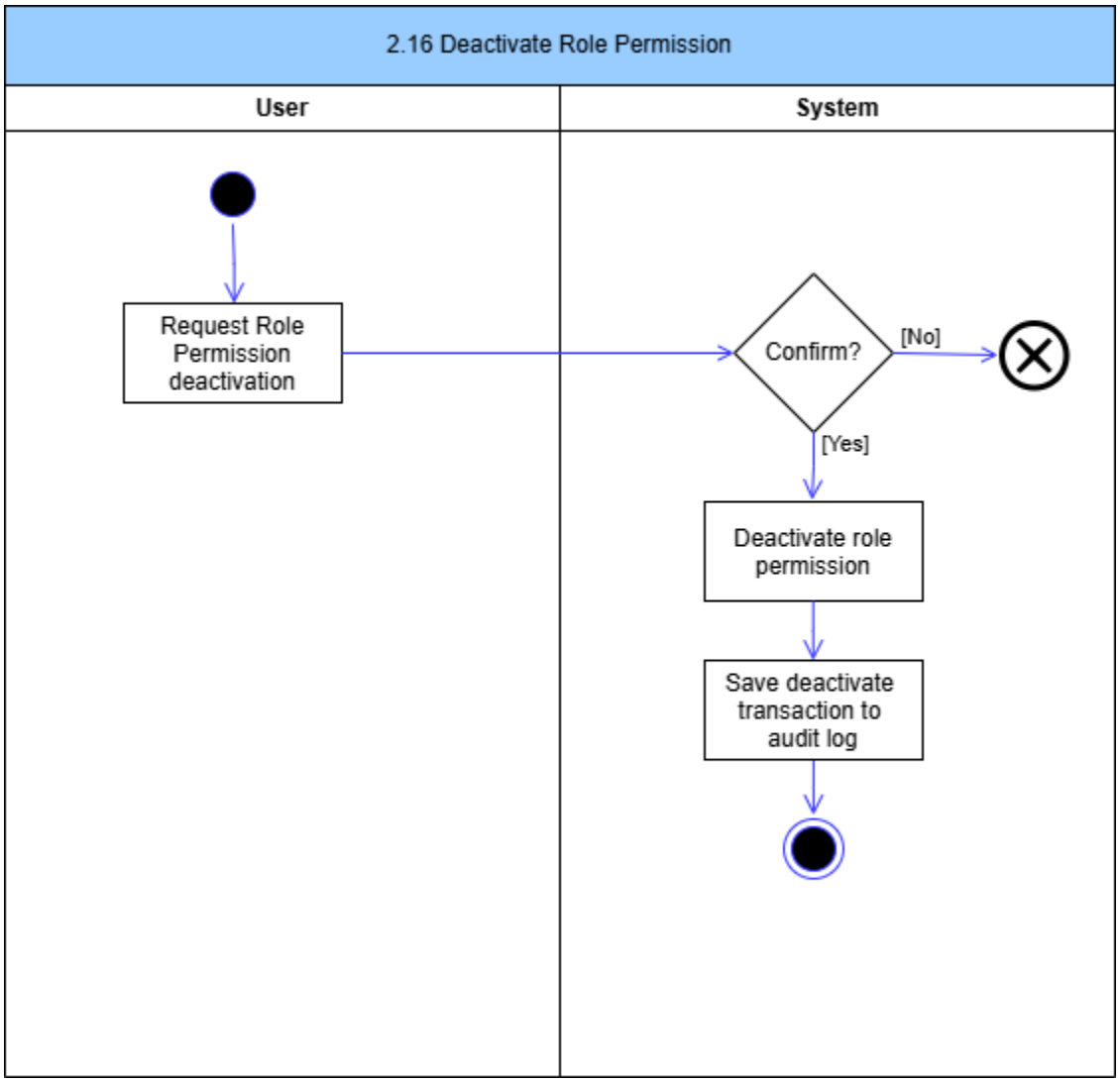


Figure 149 2.16 Deactivate Role Permission - Activity Diagram

2.10.3 Inventory Subsystem Activity Diagrams

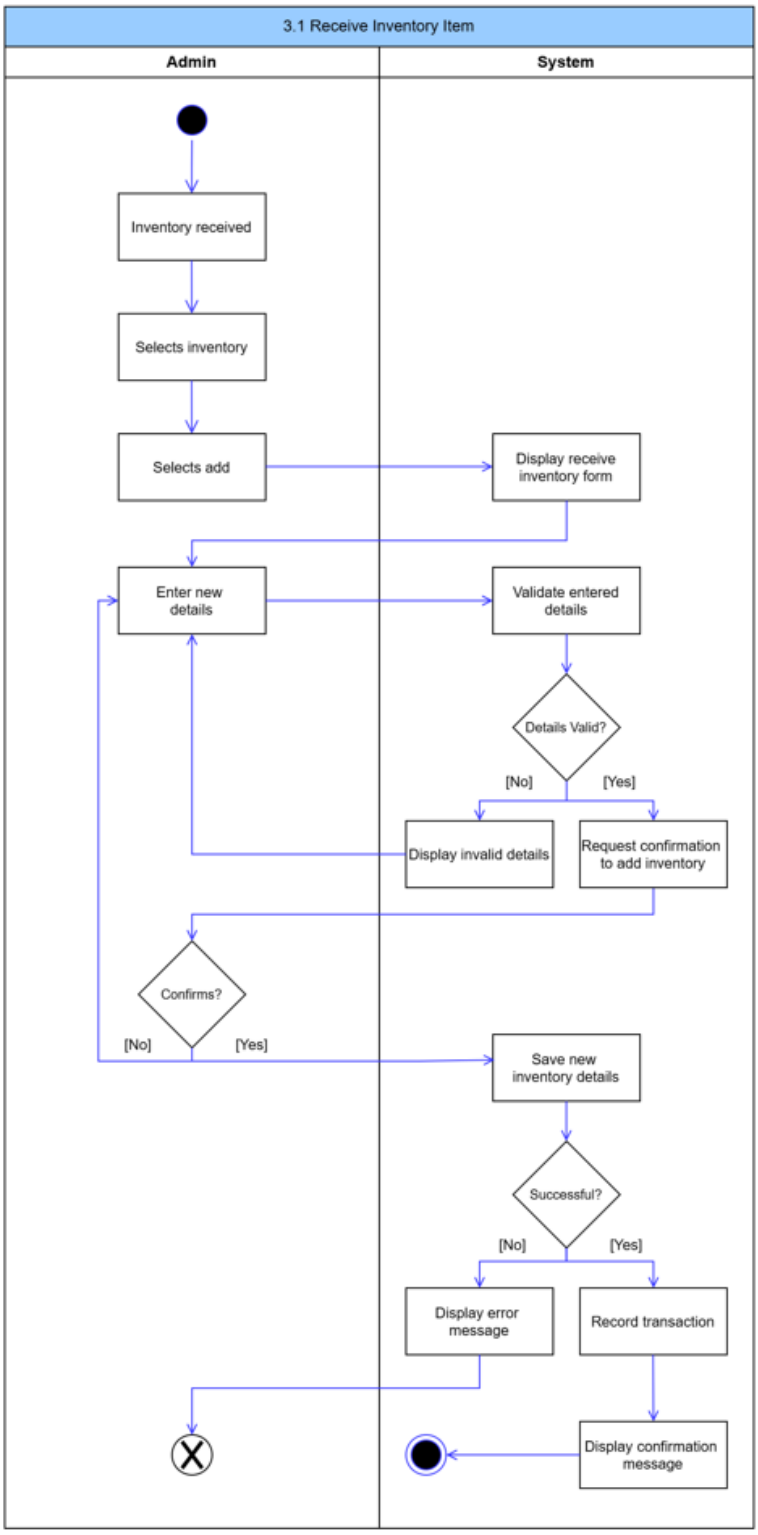


Figure 150 3.1 Receive Inventory Item - Activity Diagram

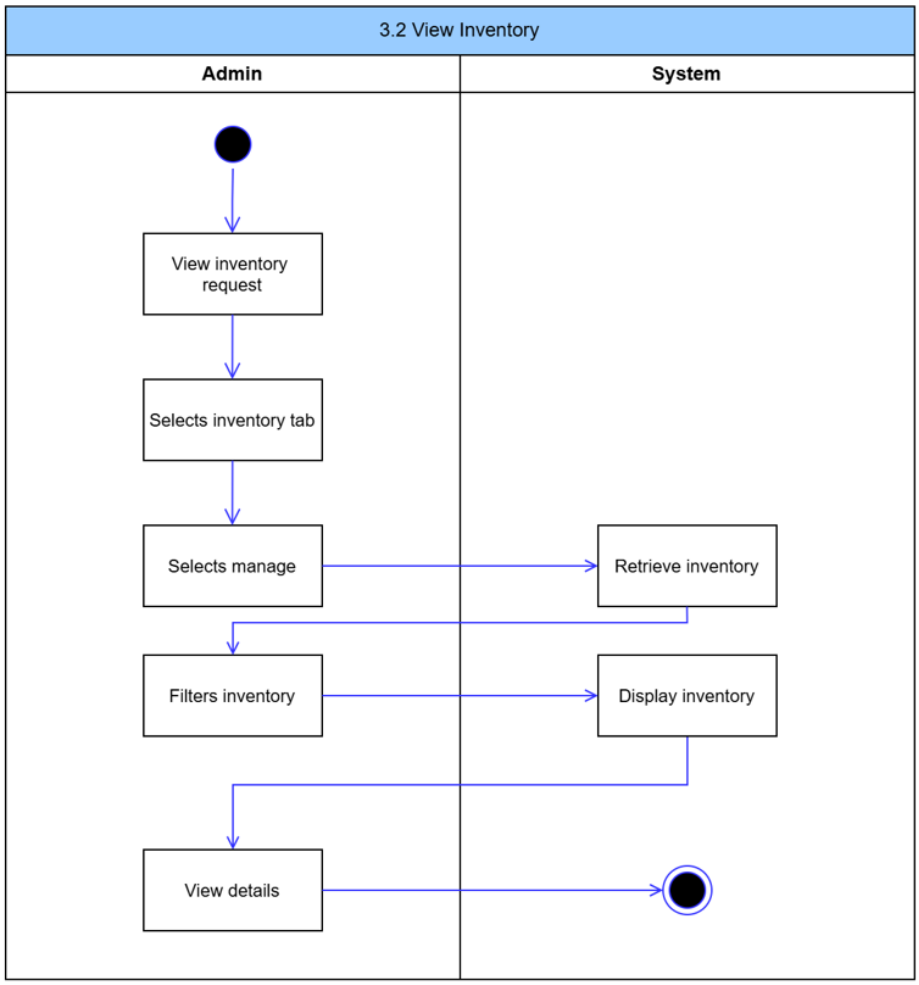


Figure 151 3.2 View Inventory - Activity Diagram

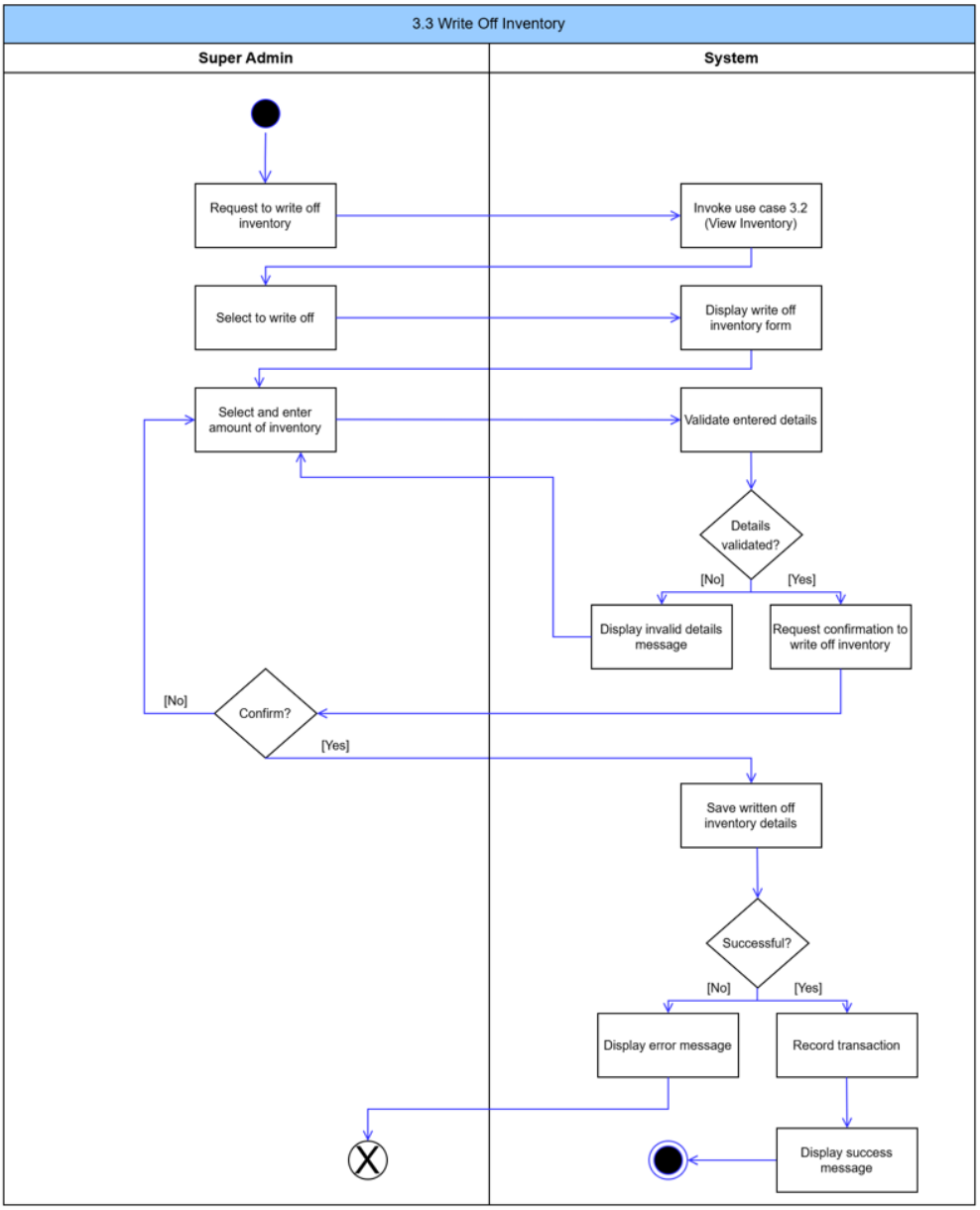


Figure 152 3.3 Write Off Inventory - Activity Diagram

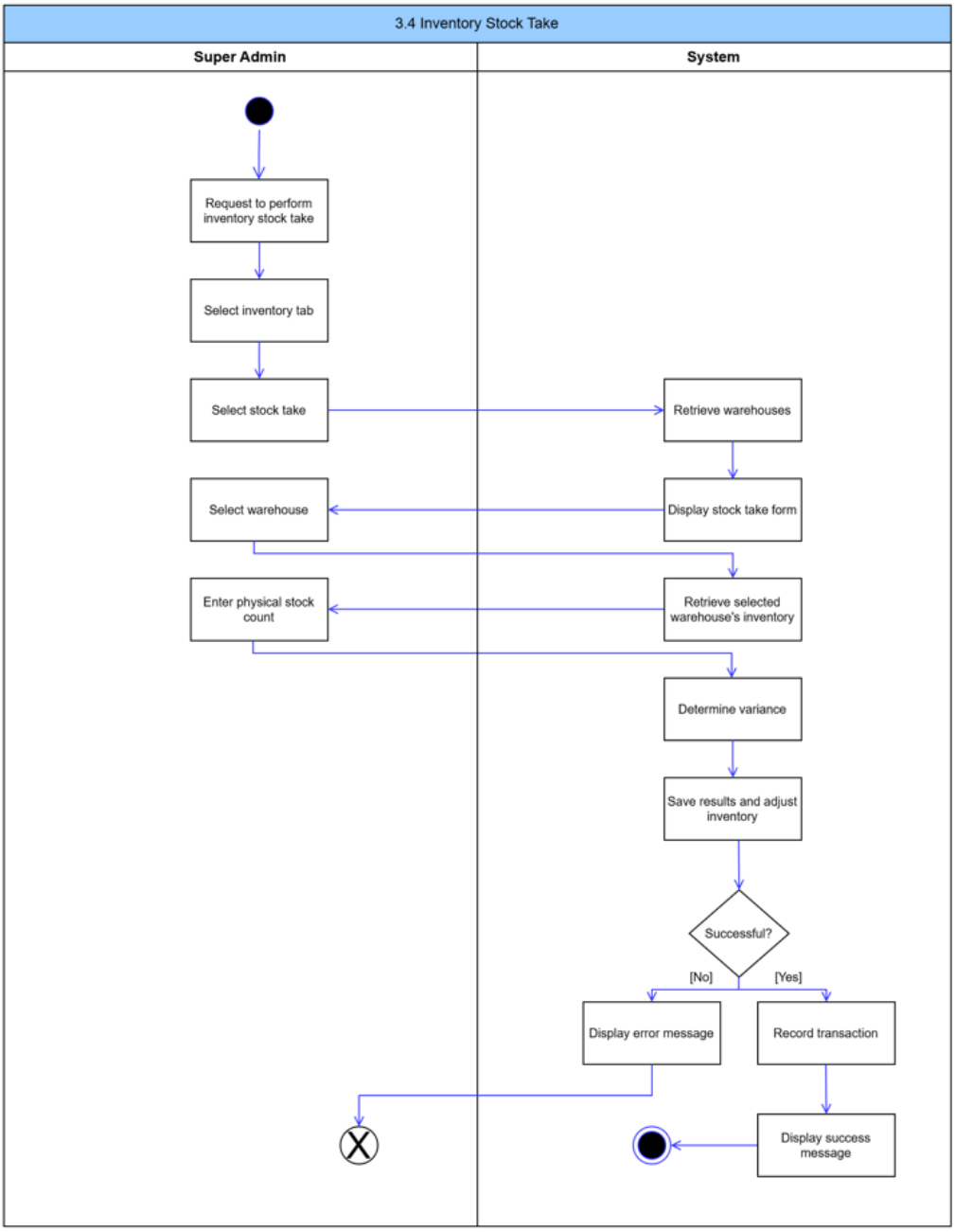


Figure 153 3.4 Inventory Stock Take - Activity Diagram

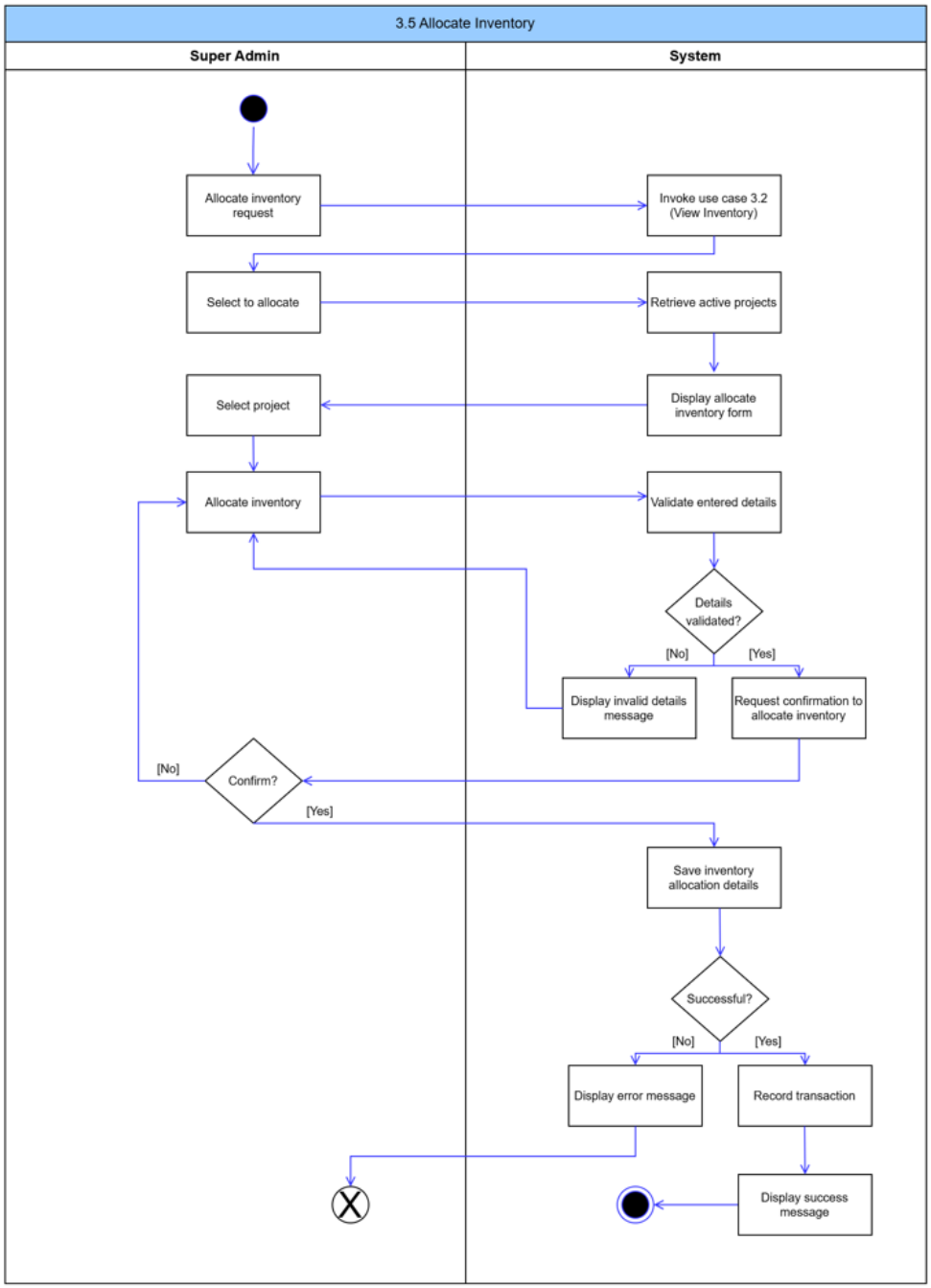


Figure 154 3.5 Allocate Inventory - Activity Diagram

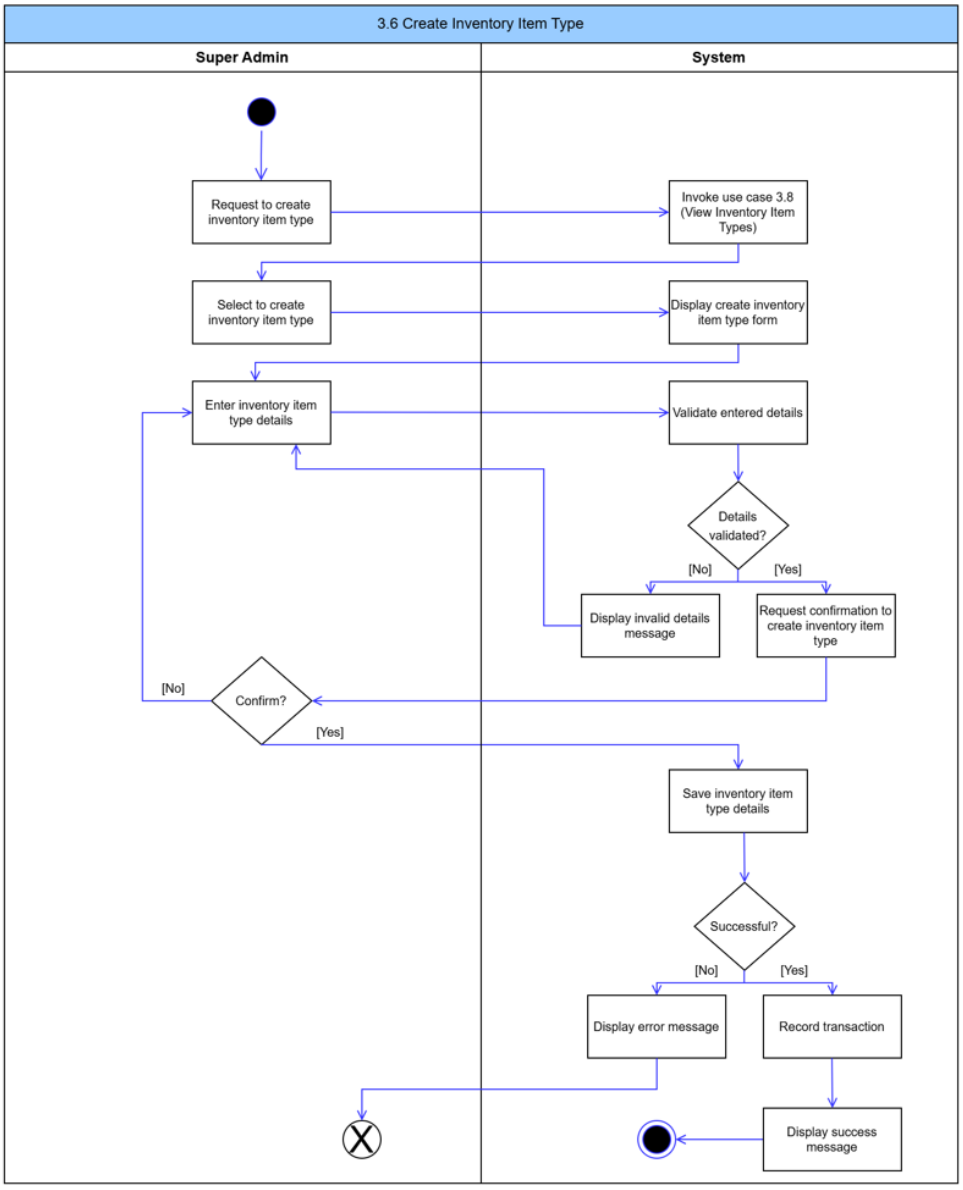


Figure 155 3.6 Create Inventory Item Type - Activity Diagram

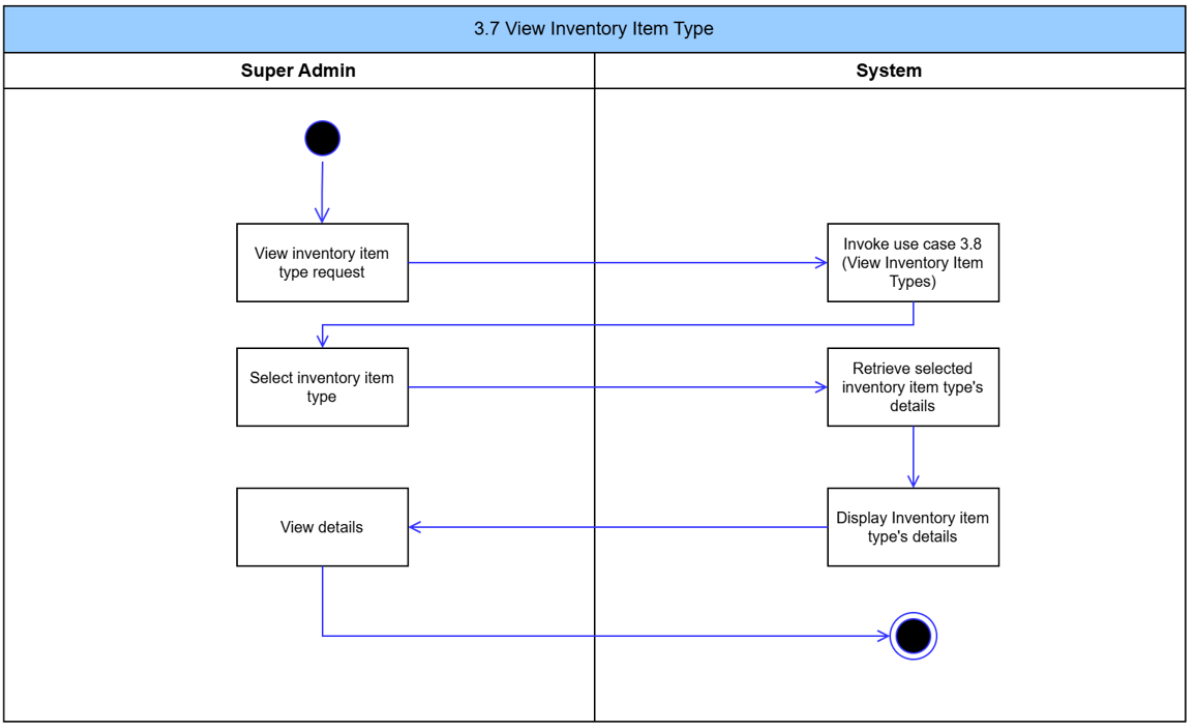


Figure 156 3.7 View Inventory Item Type - Activity Diagram

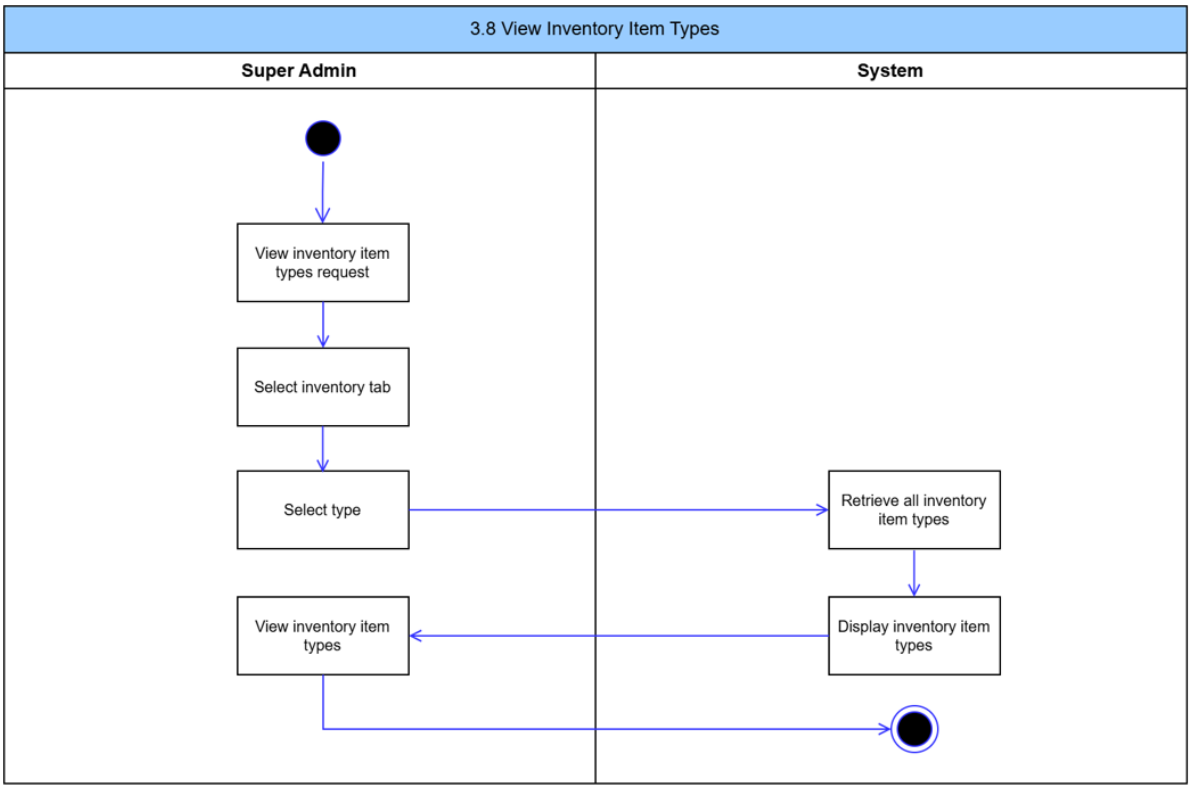


Figure 157 3.8 View Inventory Item Types - Activity Diagram

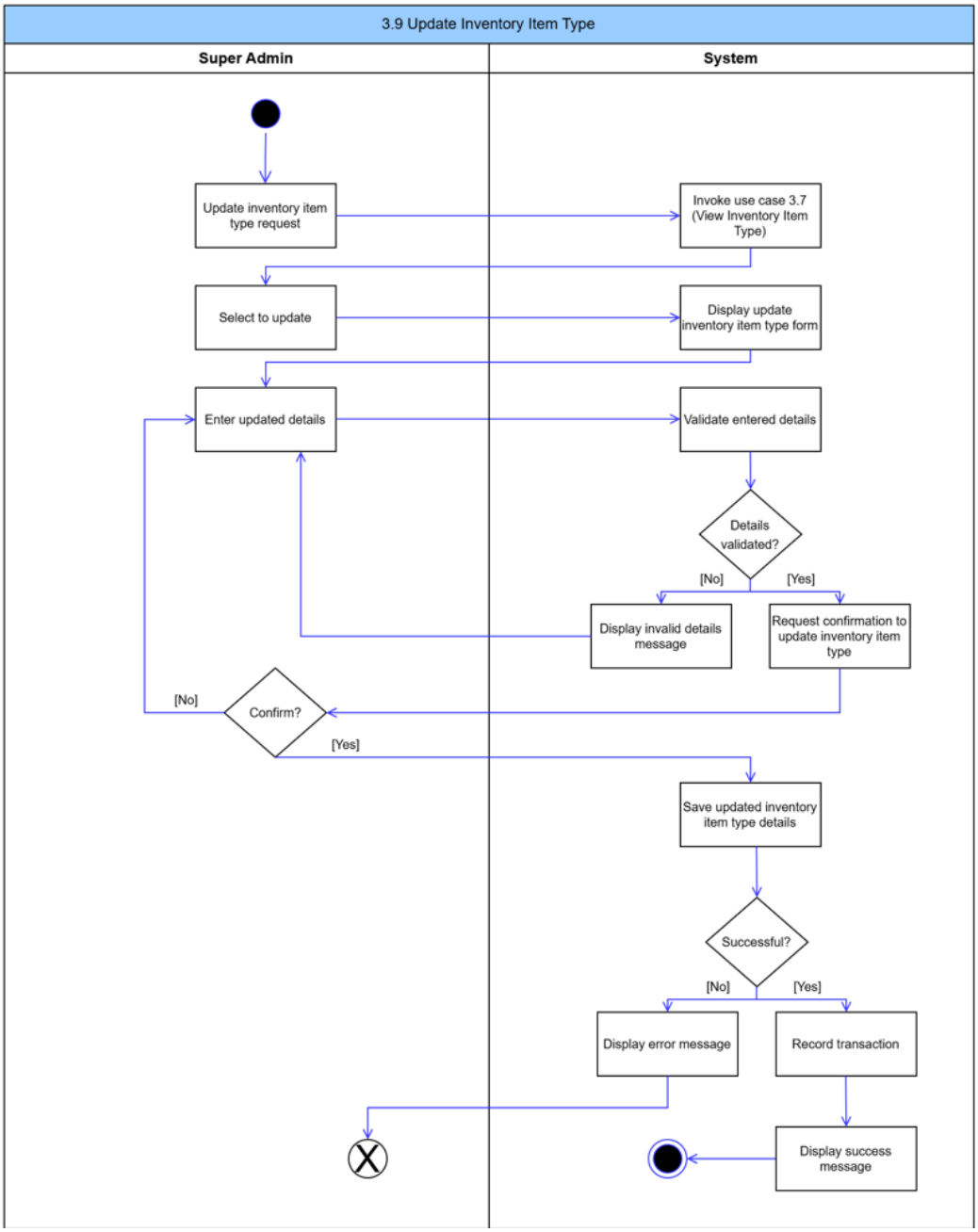


Figure 158 3.9 Update Inventory Item Type - Activity Diagram

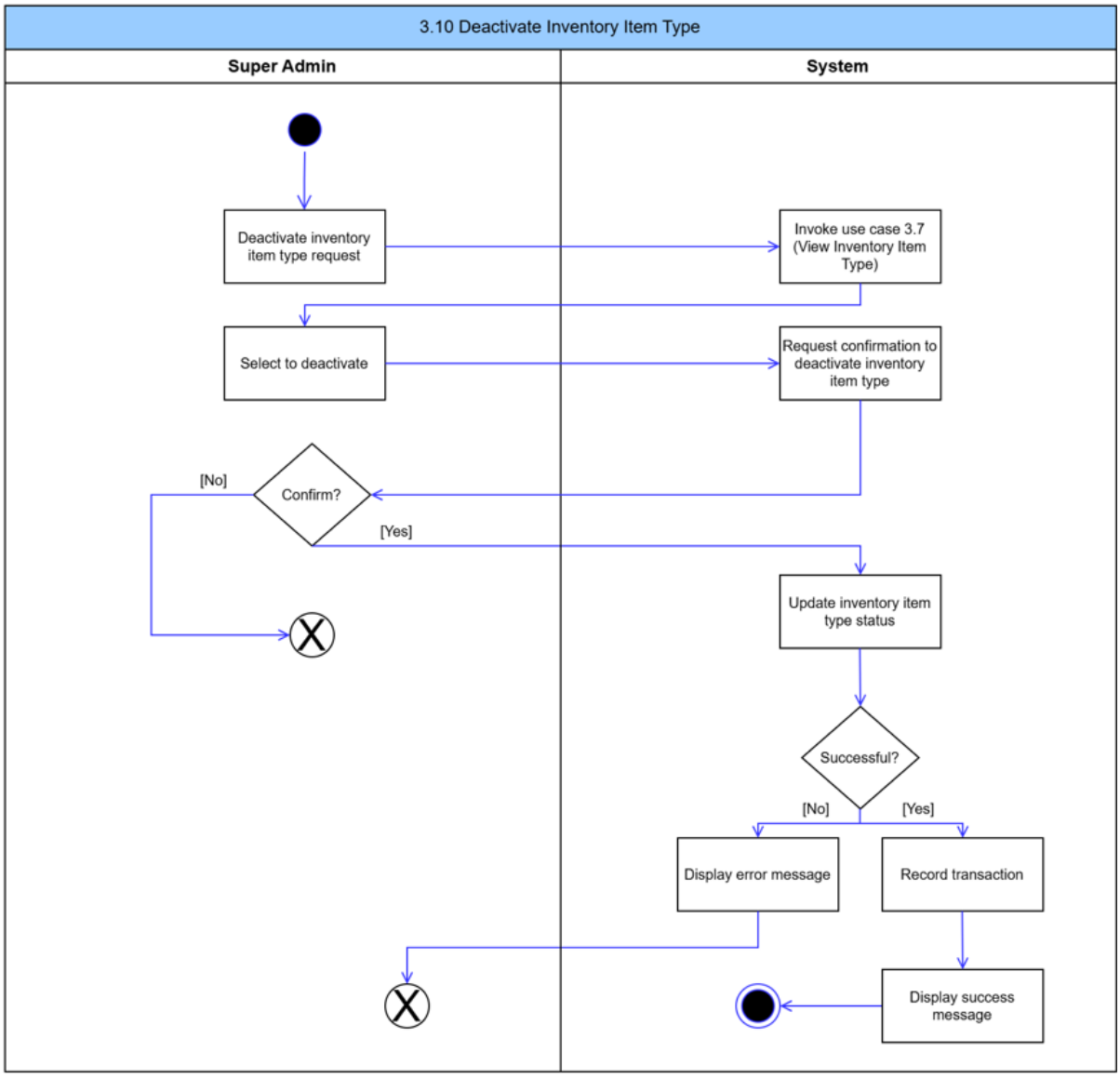


Figure 159 3.10 Deactivate Inventory Item Type - Activity Diagram

2.10.4 Group Subsystem Activity Diagrams

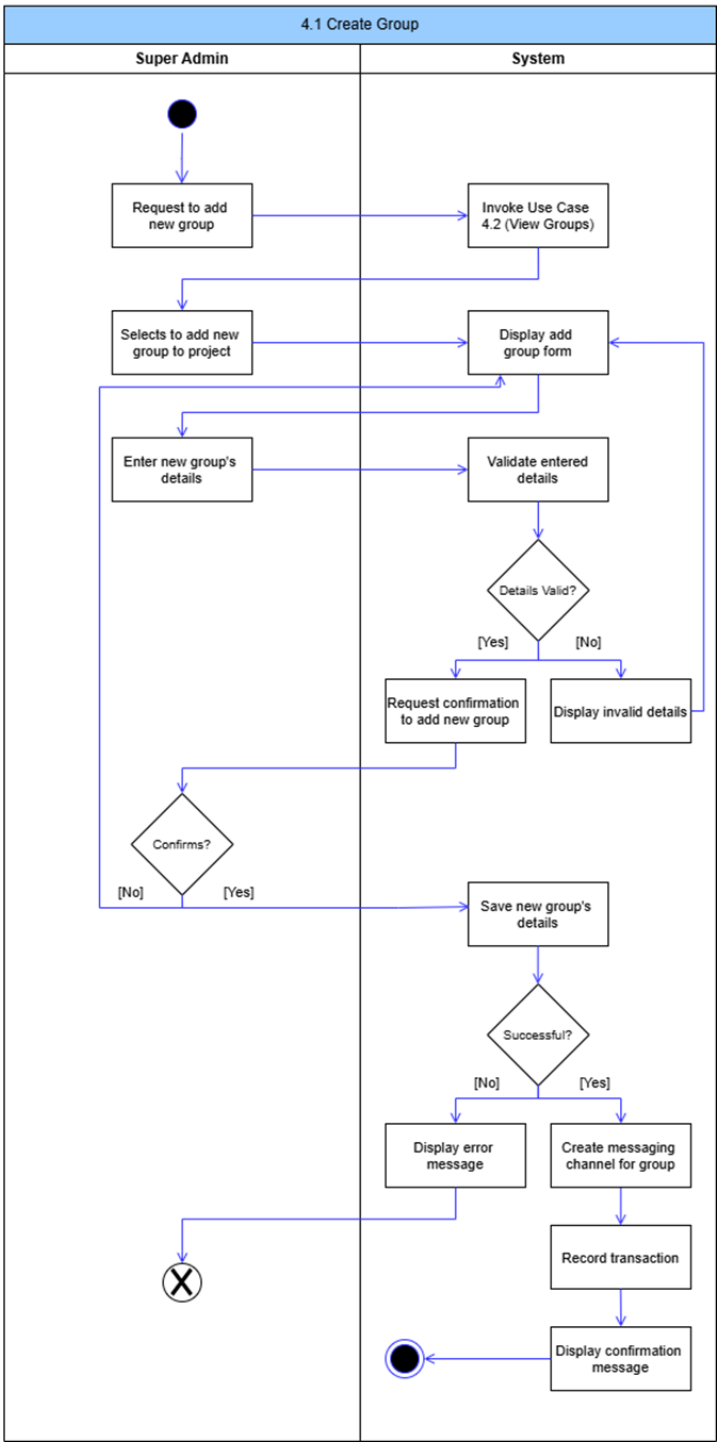


Figure 160 4.1 Create Group - Activity Diagram

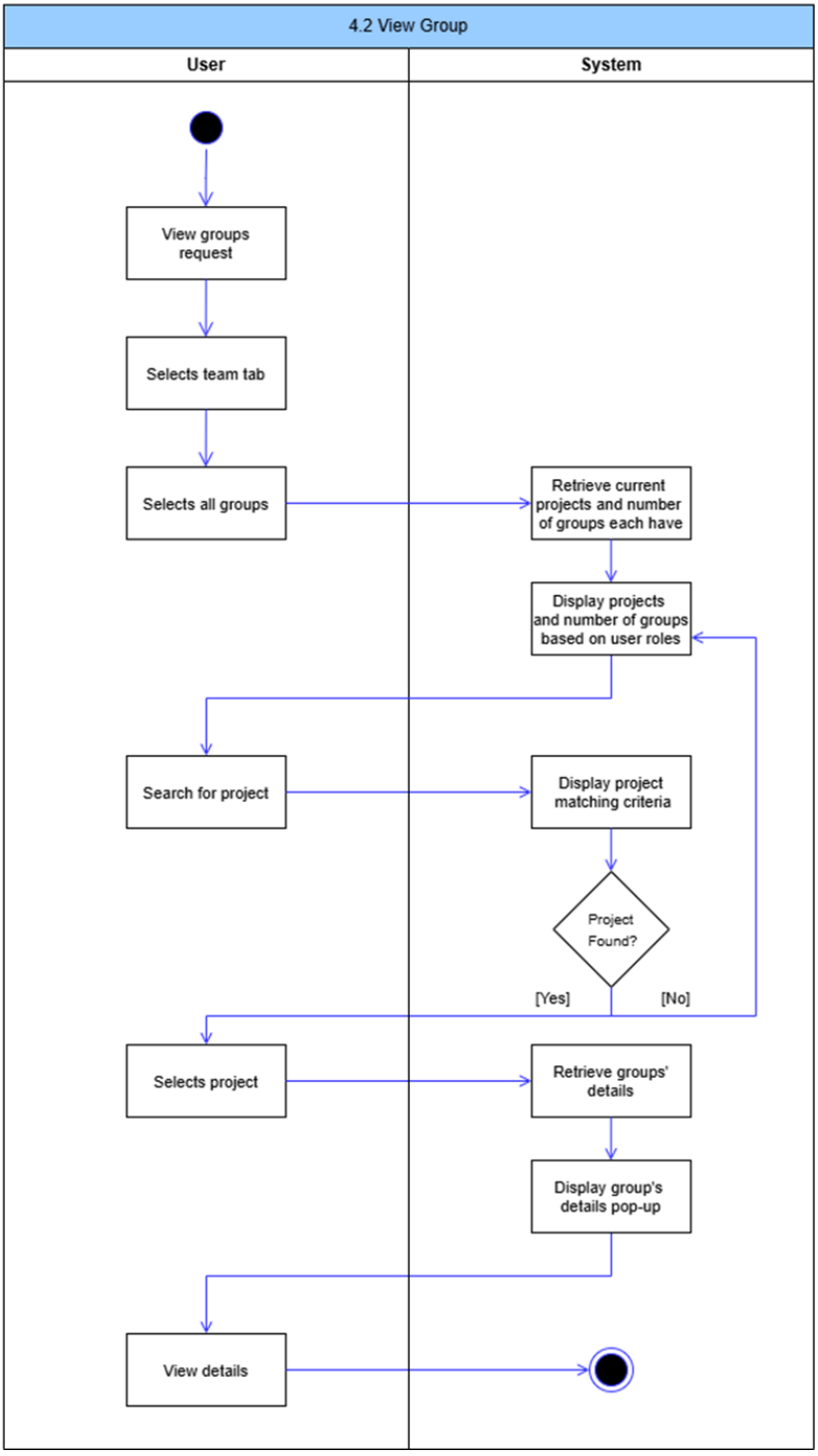


Figure 161 4.2 View Group - Activity Diagram

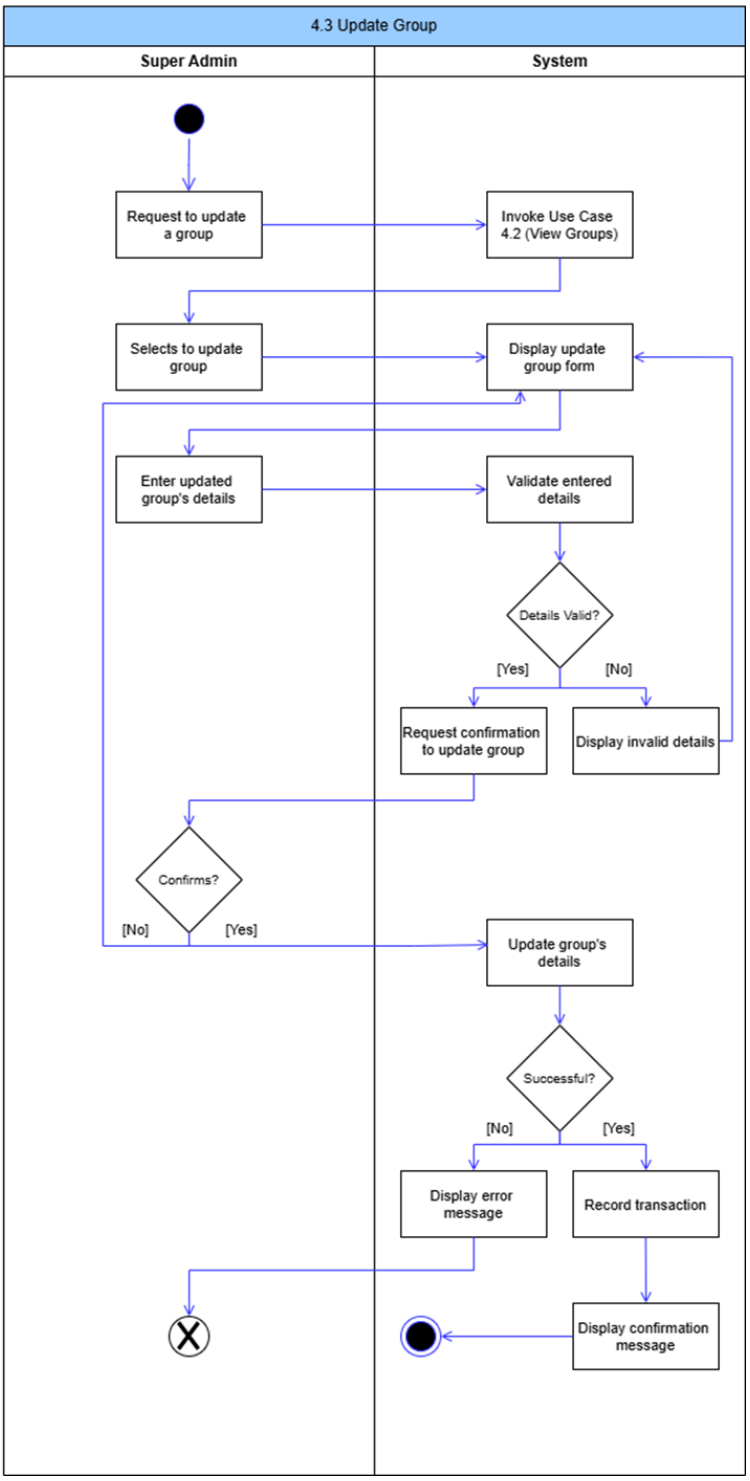


Figure 162 4.3 Update Group - Activity Diagram

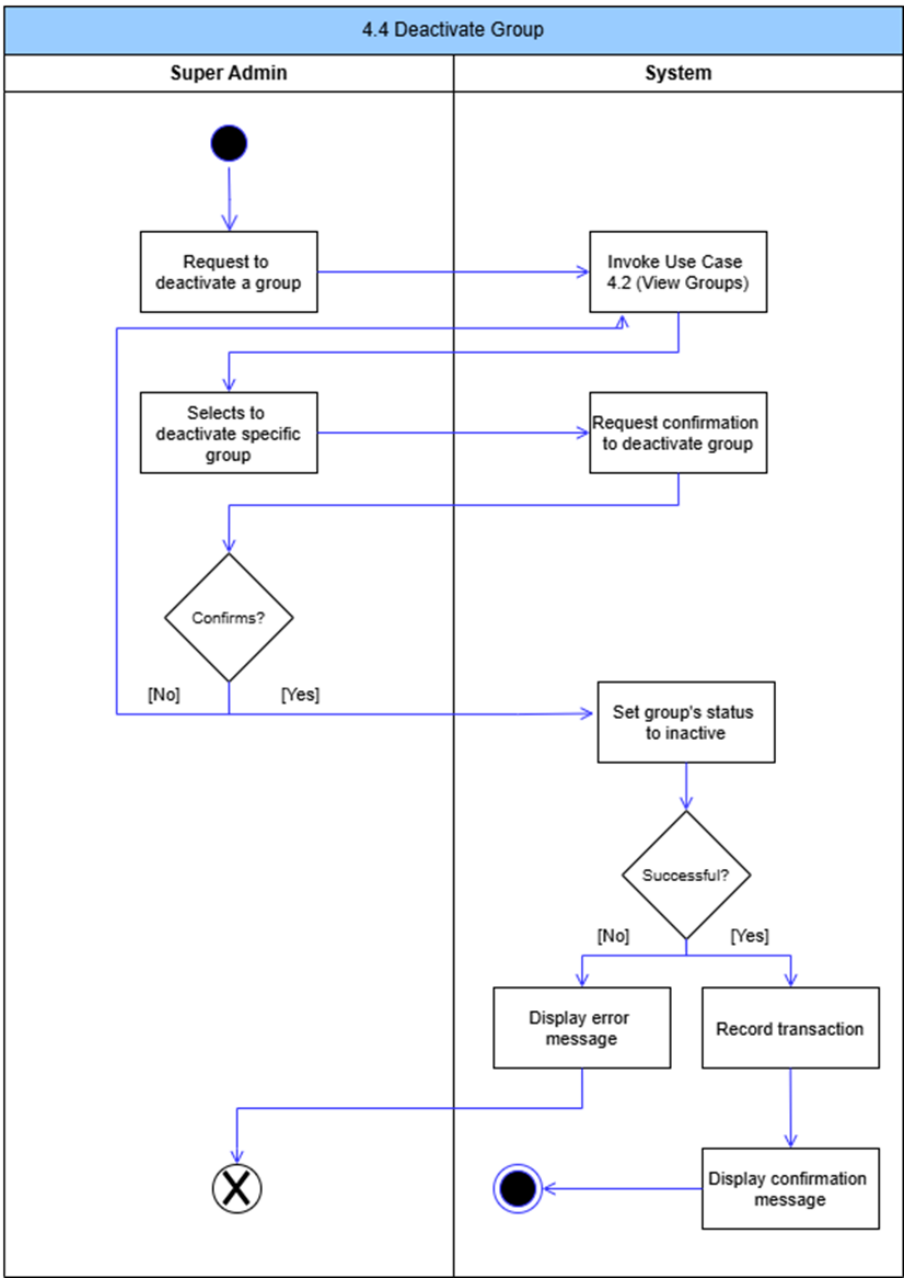


Figure 163 4.4 Deactivate Group - Activity Diagram

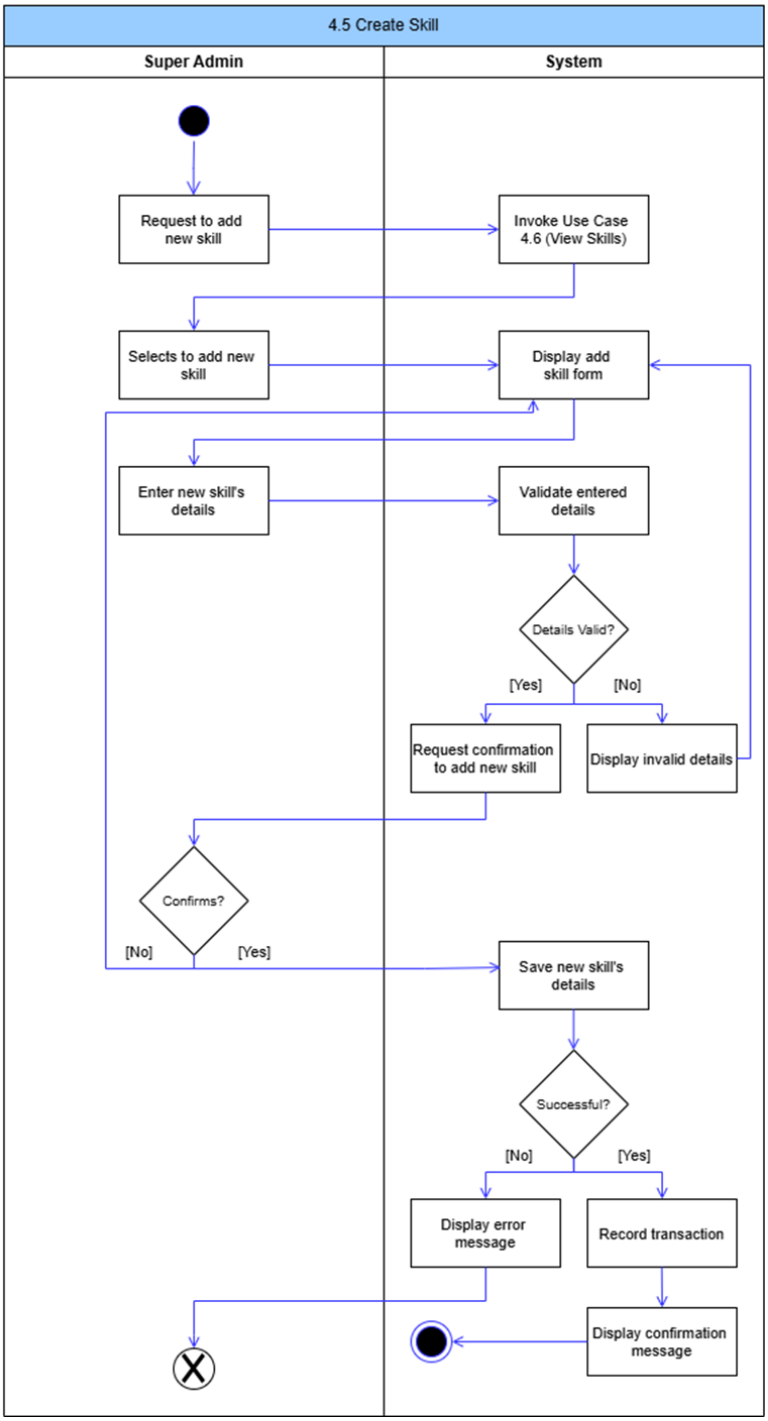


Figure 164 4.5 Create Skill - Activity Diagram

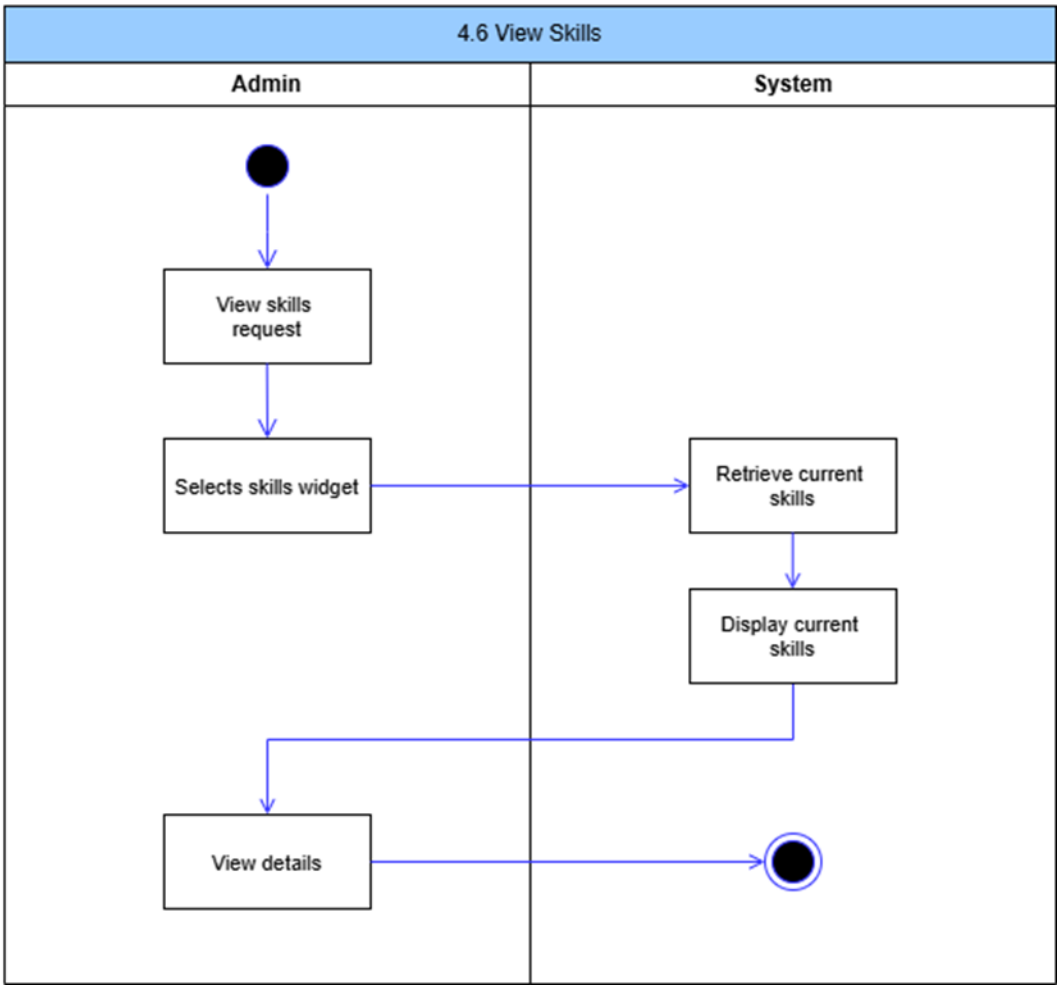


Figure 165 4.6 View Skills - Activity Diagram

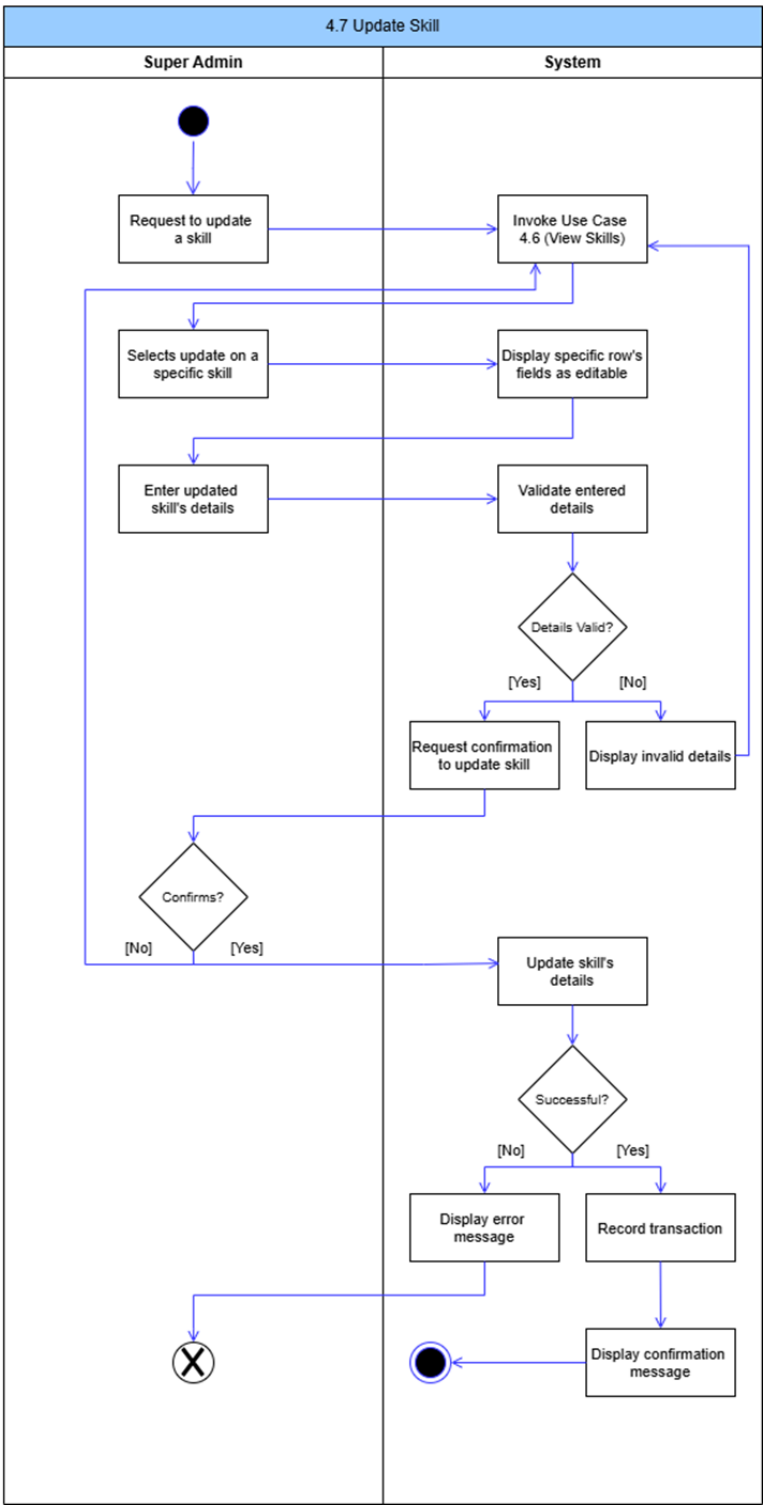


Figure 166 4.7 Update Skill - Activity Diagram

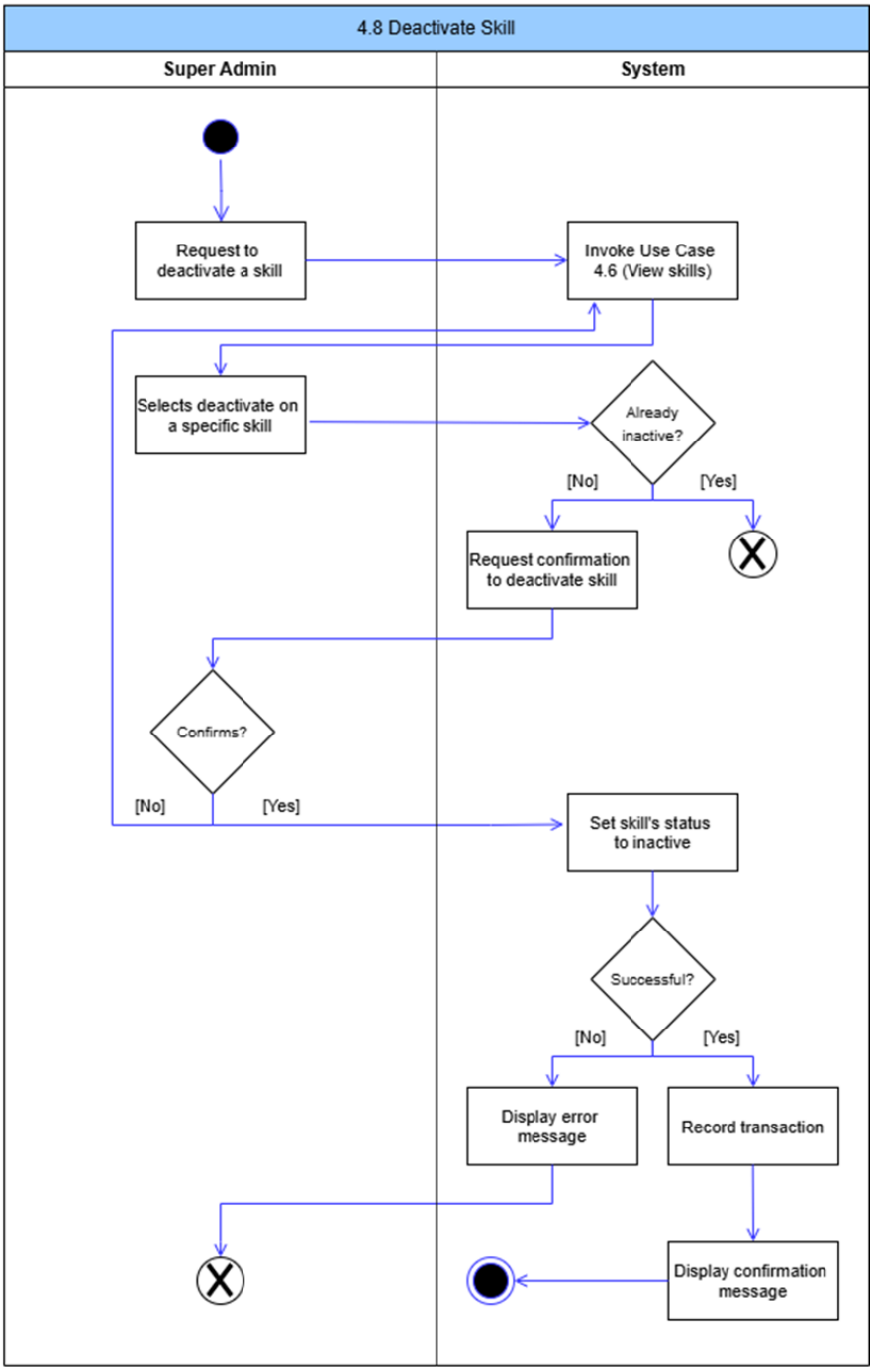


Figure 167 4.8 Deactivate Skill - Activity Diagram

2.10.5 Warehouse Subsystem Activity Diagrams

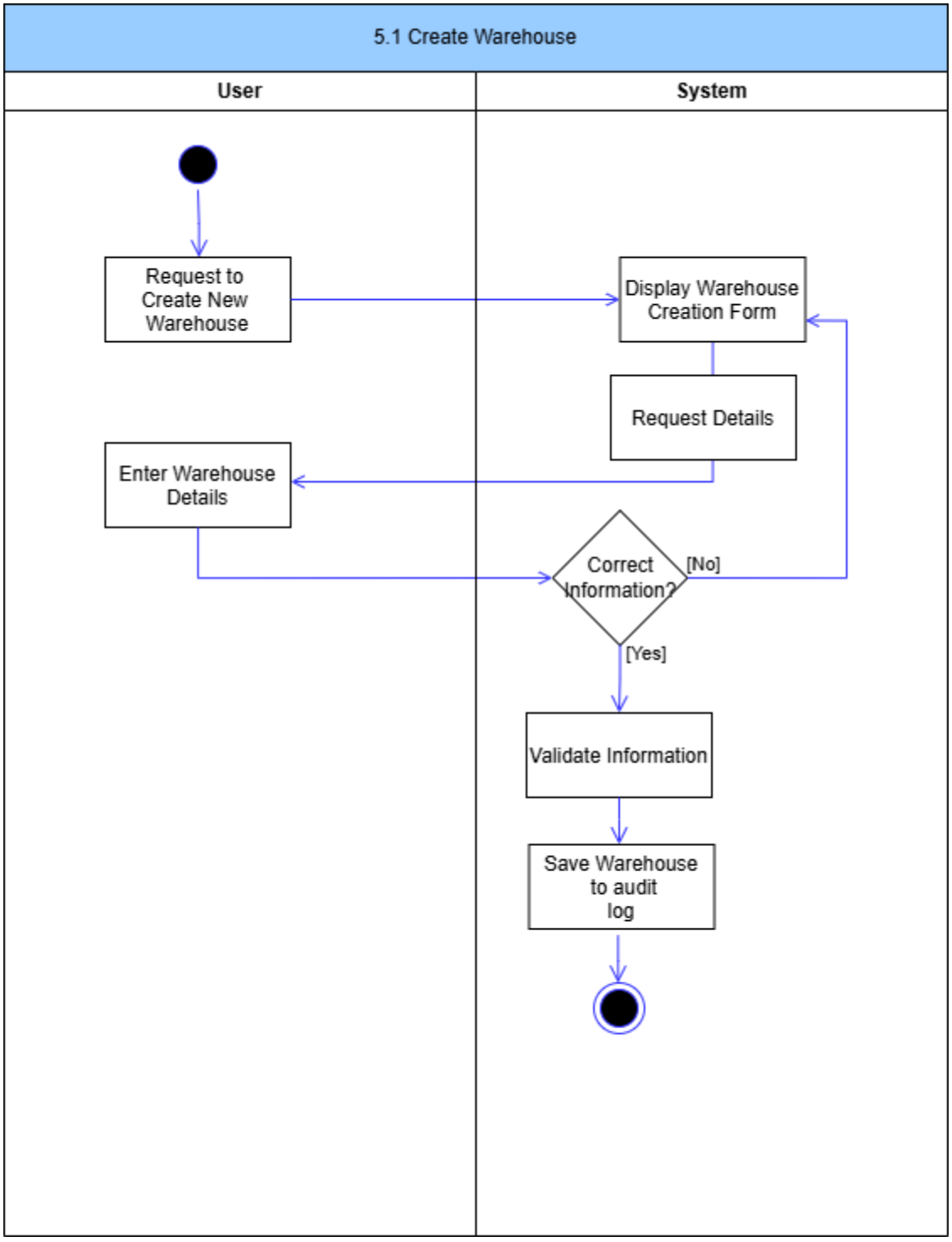


Figure 168 5.1 Create Warehouse - Activity Diagram

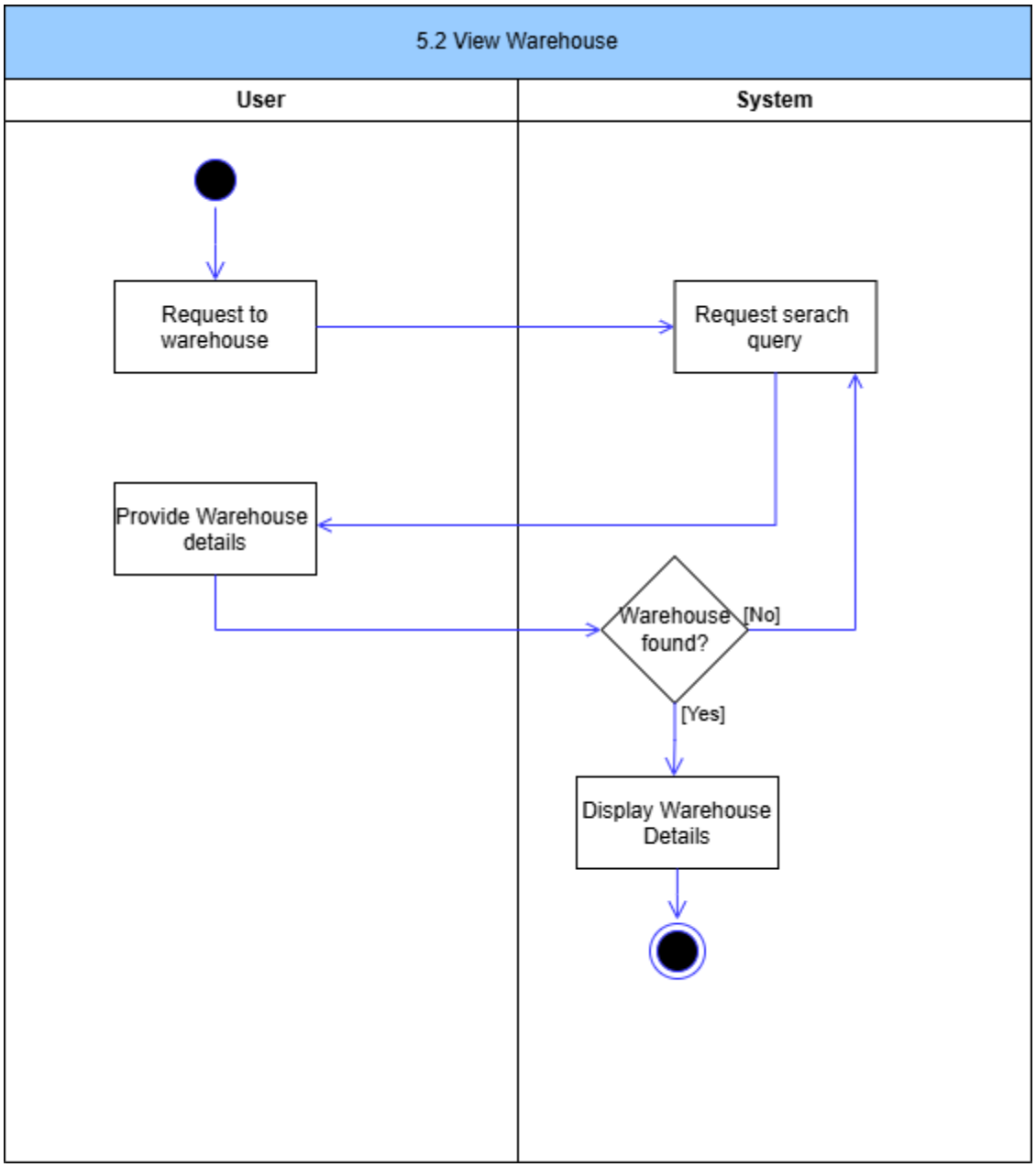


Figure 169 5.2 View Warehouse - Activity Diagram1

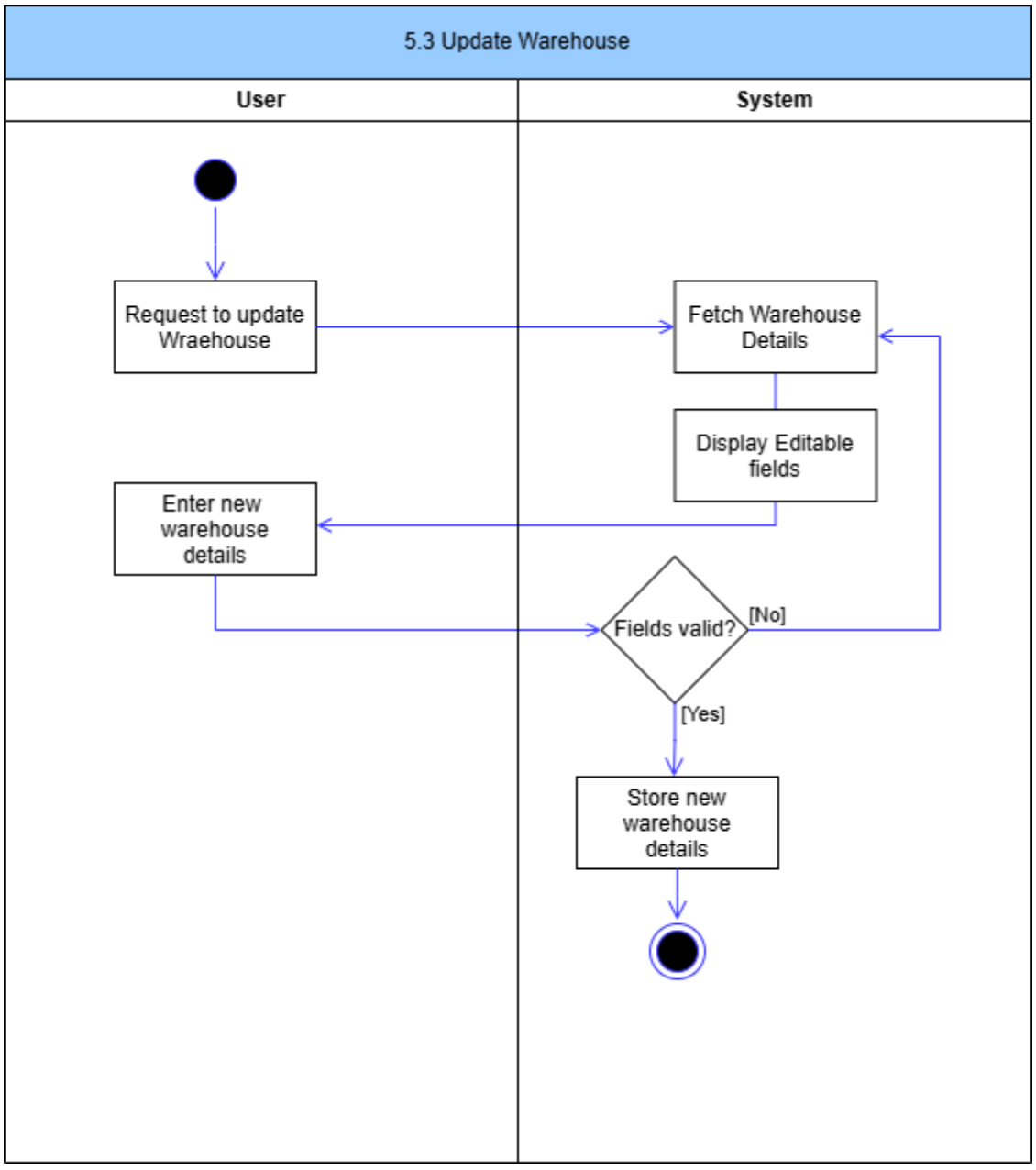


Figure 170 5.3 Update Warehouse - Activity Diagram

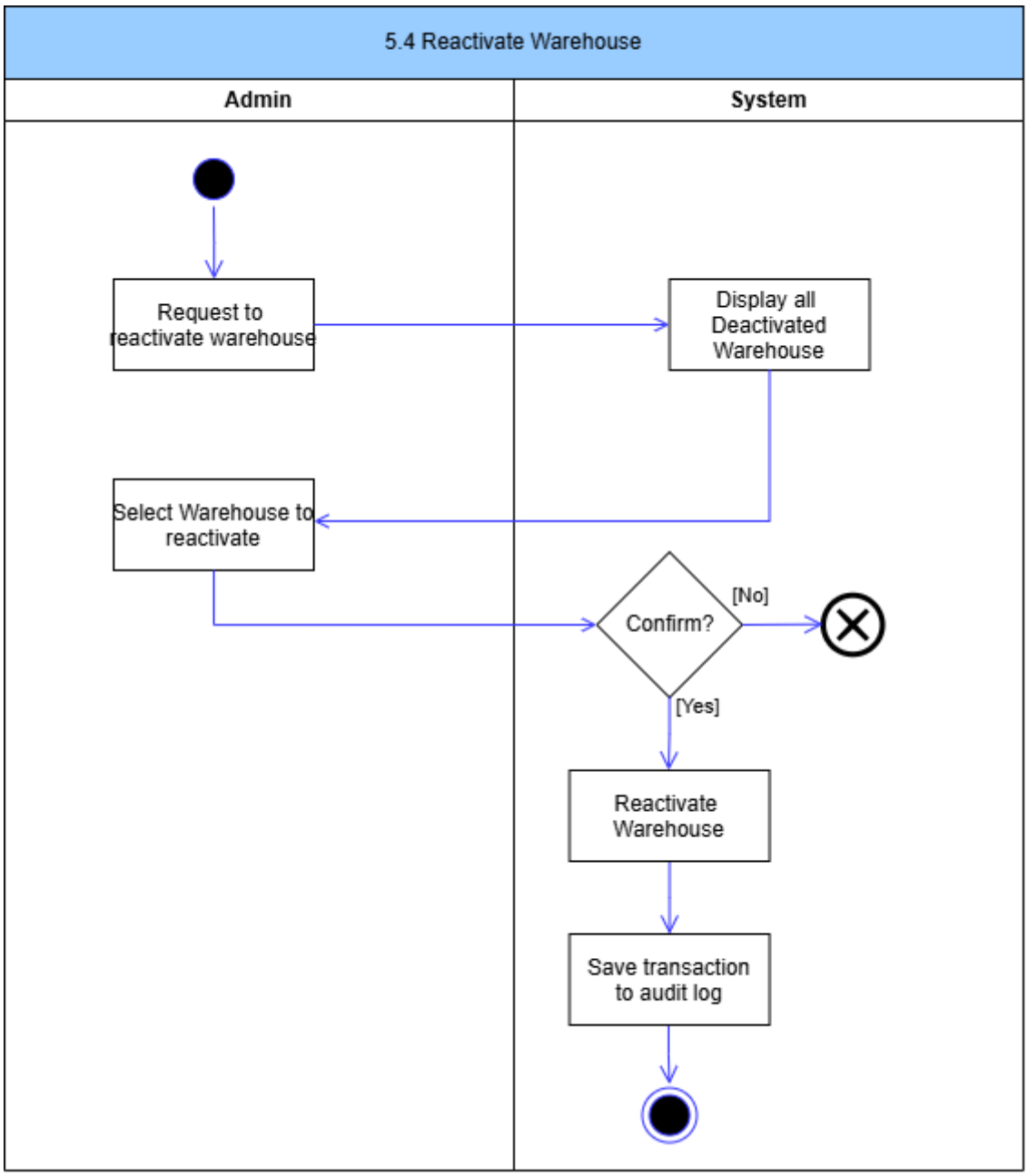


Figure 171 5.4 Reactivate Warehouse - Activity Diagram

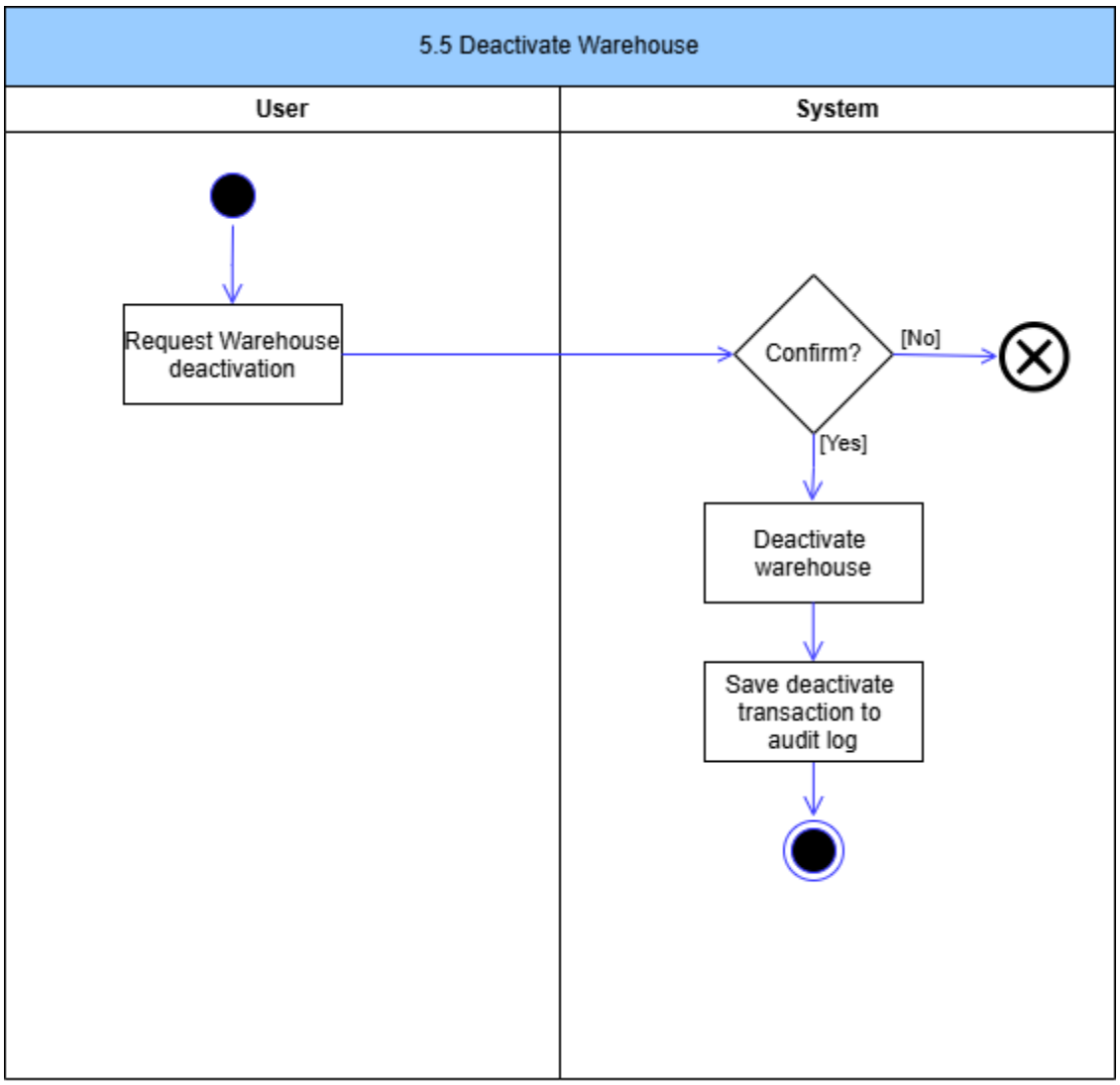


Figure 172 5.5 Deactivate Warehouse - Activity Diagram

2.10.6 Vendor Subsystem Activity Diagrams

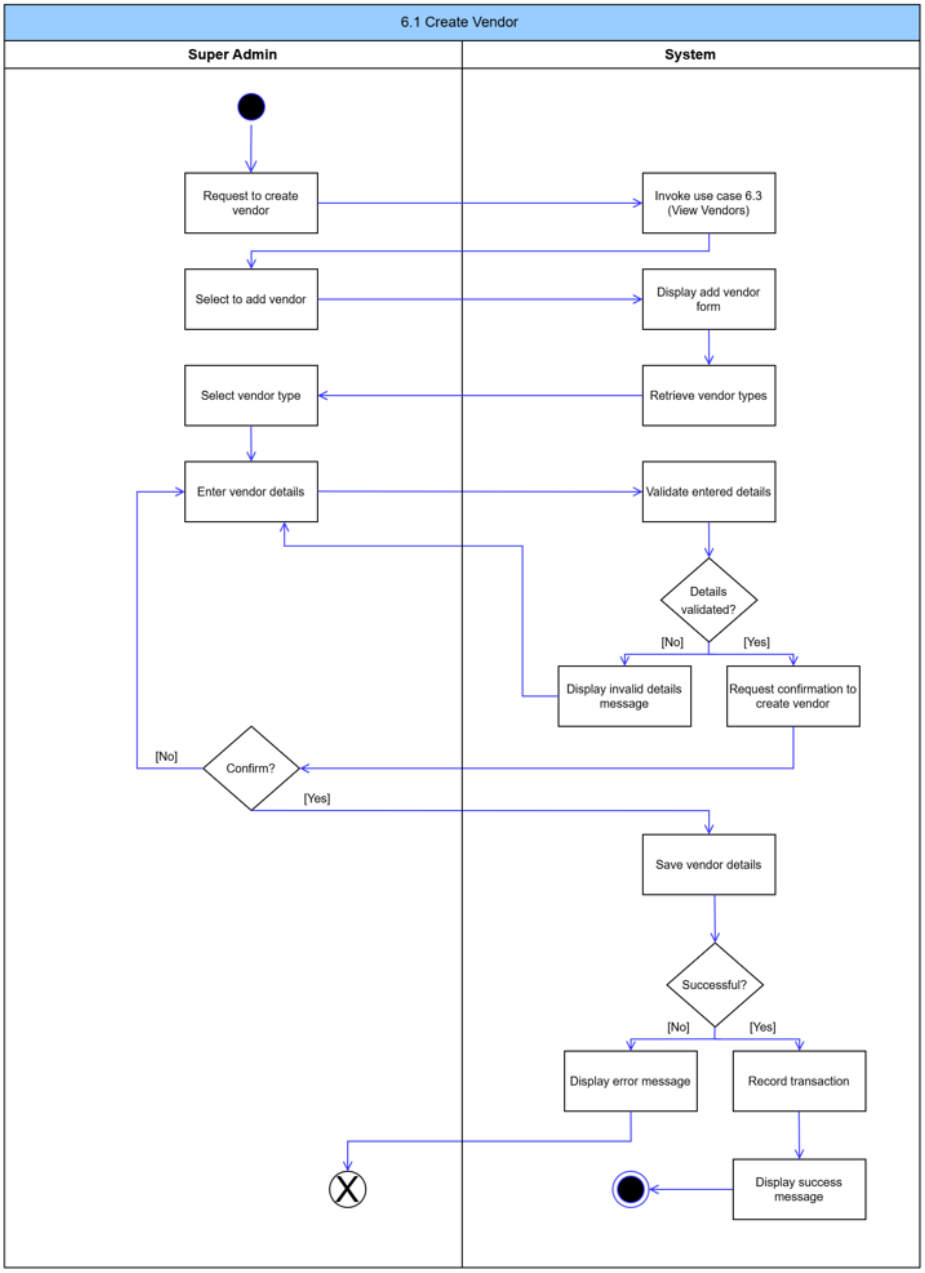


Figure 173 6.1 Create Vendor - Activity Diagram

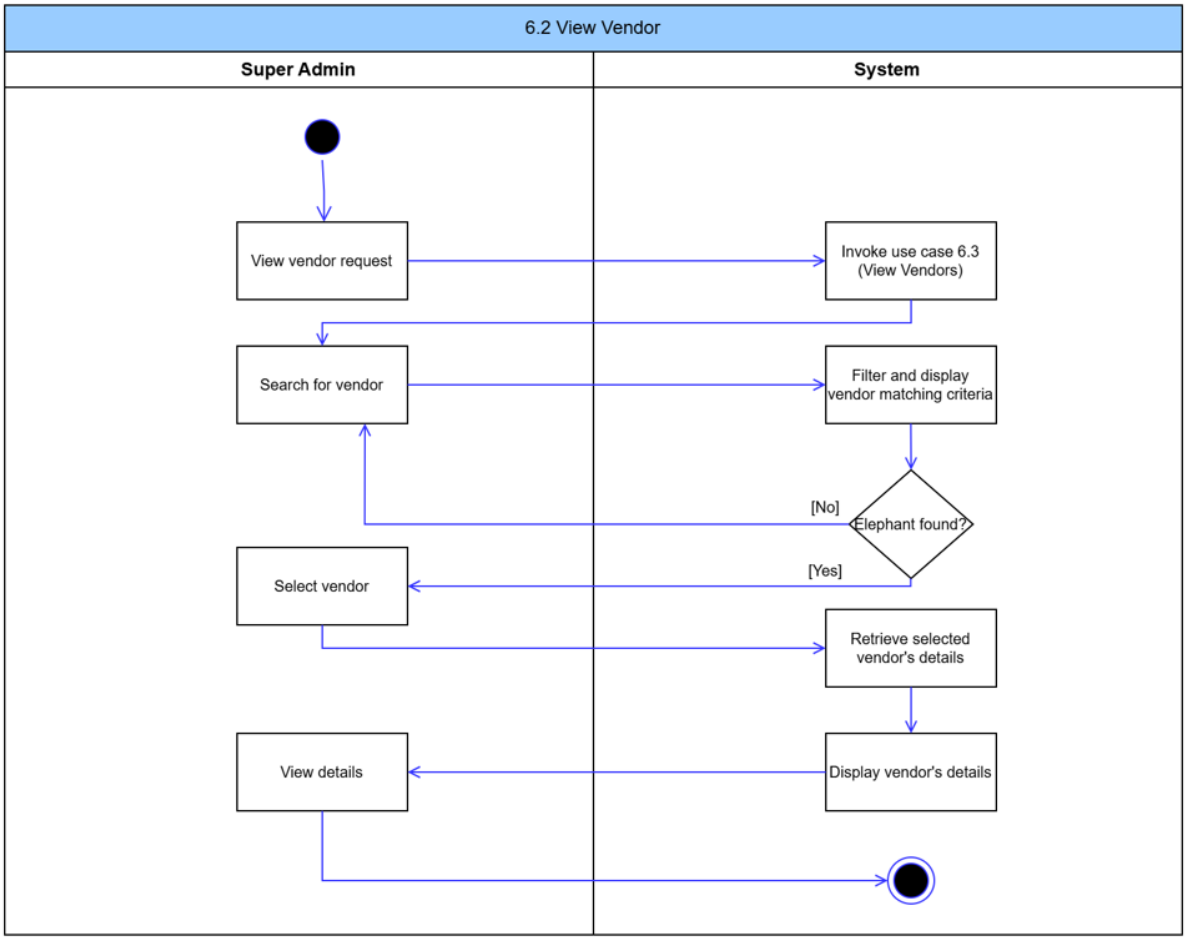


Figure 174 6.2 View Vendor - Activity Diagram

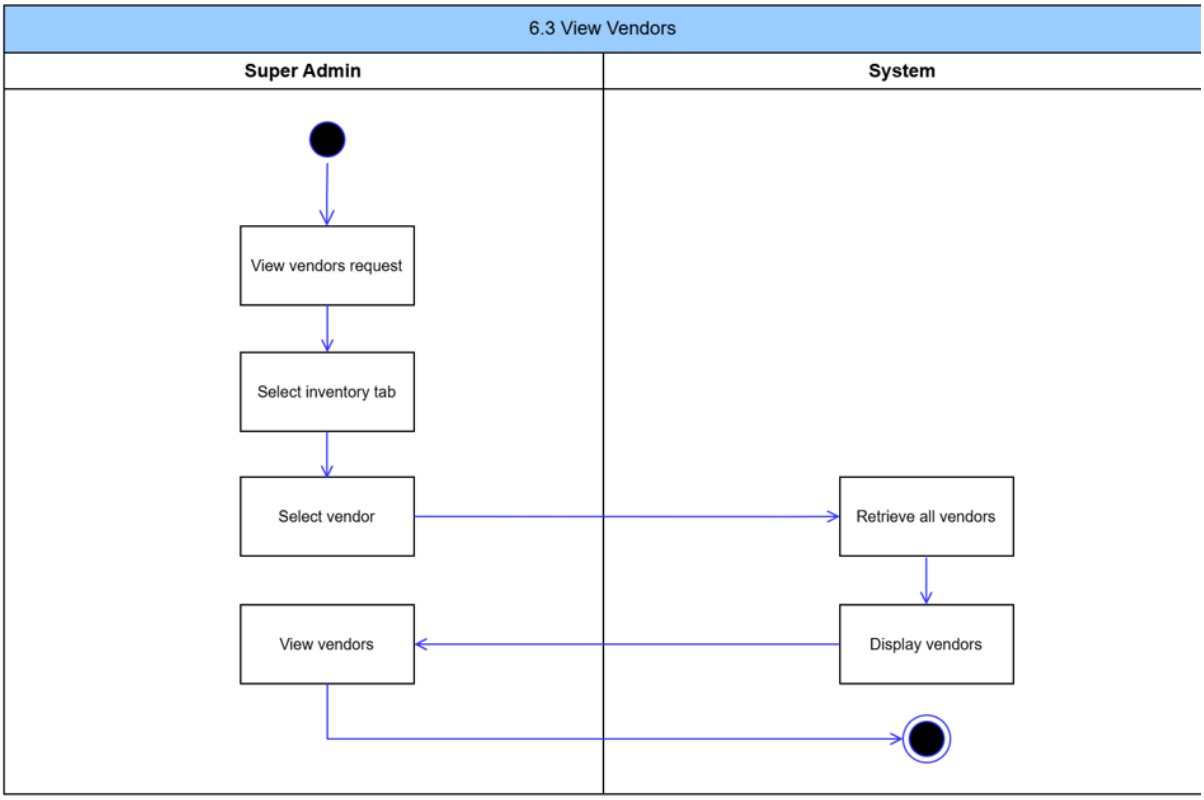


Figure 175 6.3 View Vendors - Activity Diagram

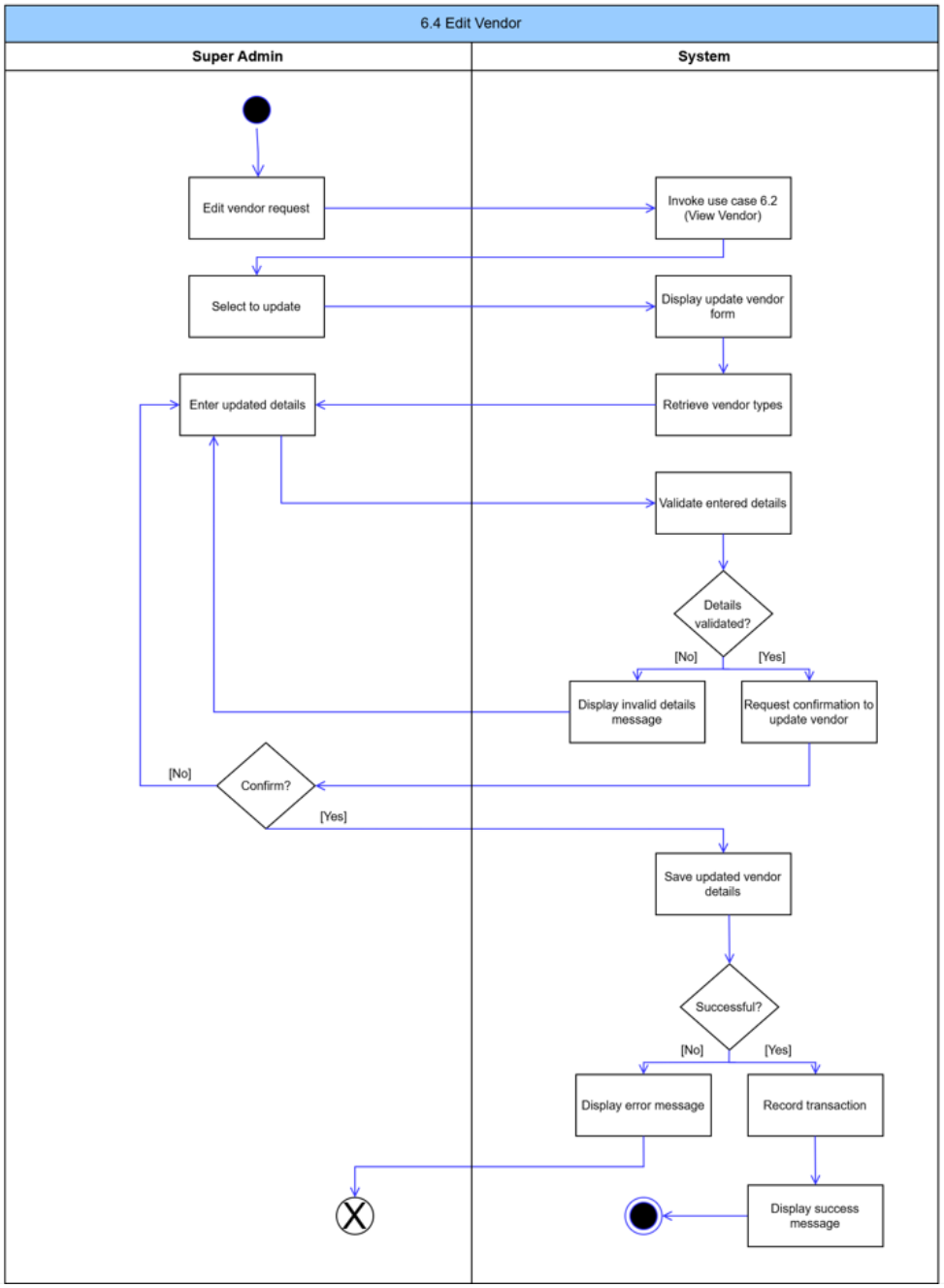


Figure 176 6.4 Edit Vendor - Activity Diagram

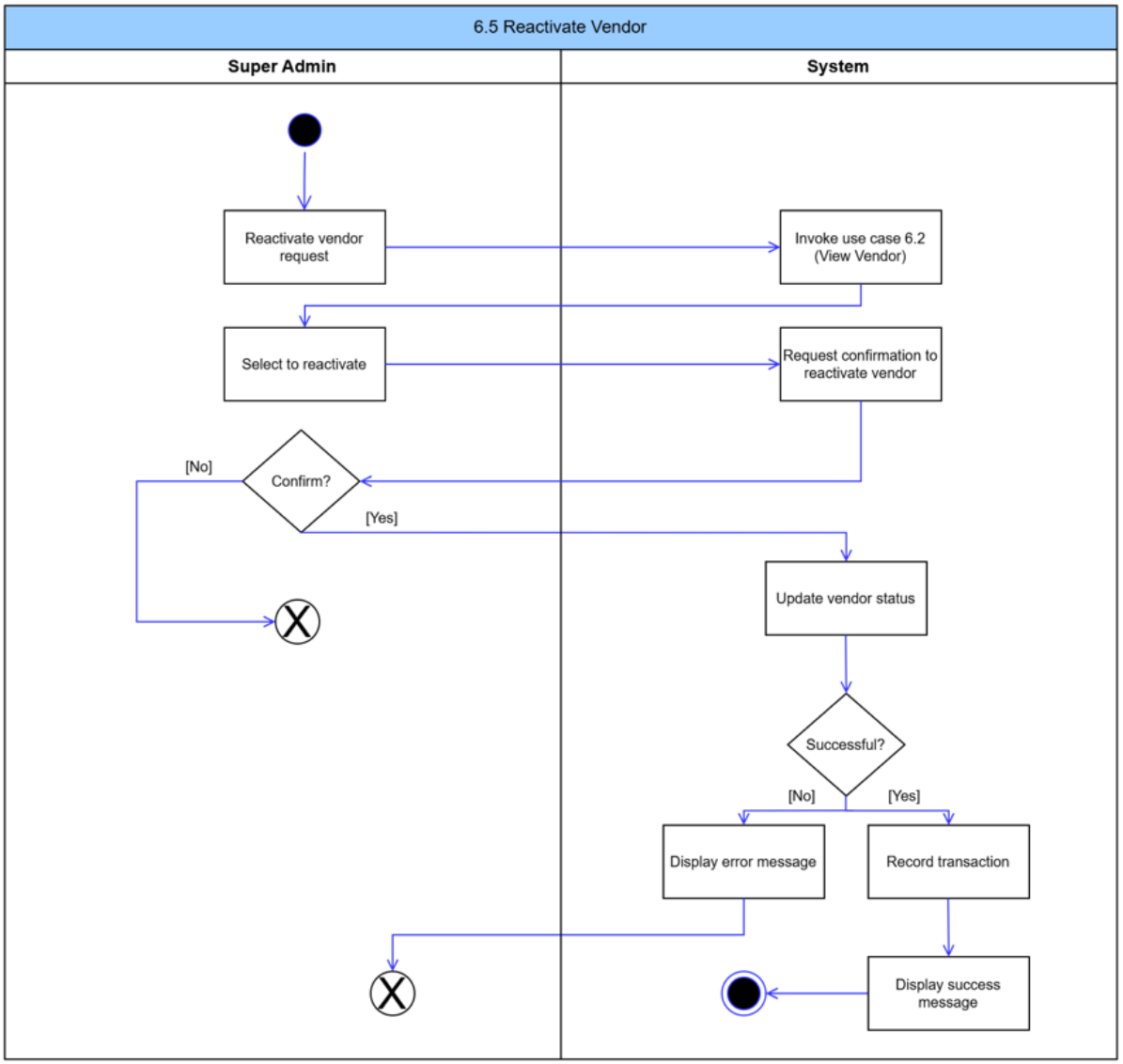


Figure 177 6.5 Reactivate Vendor - Activity Diagram

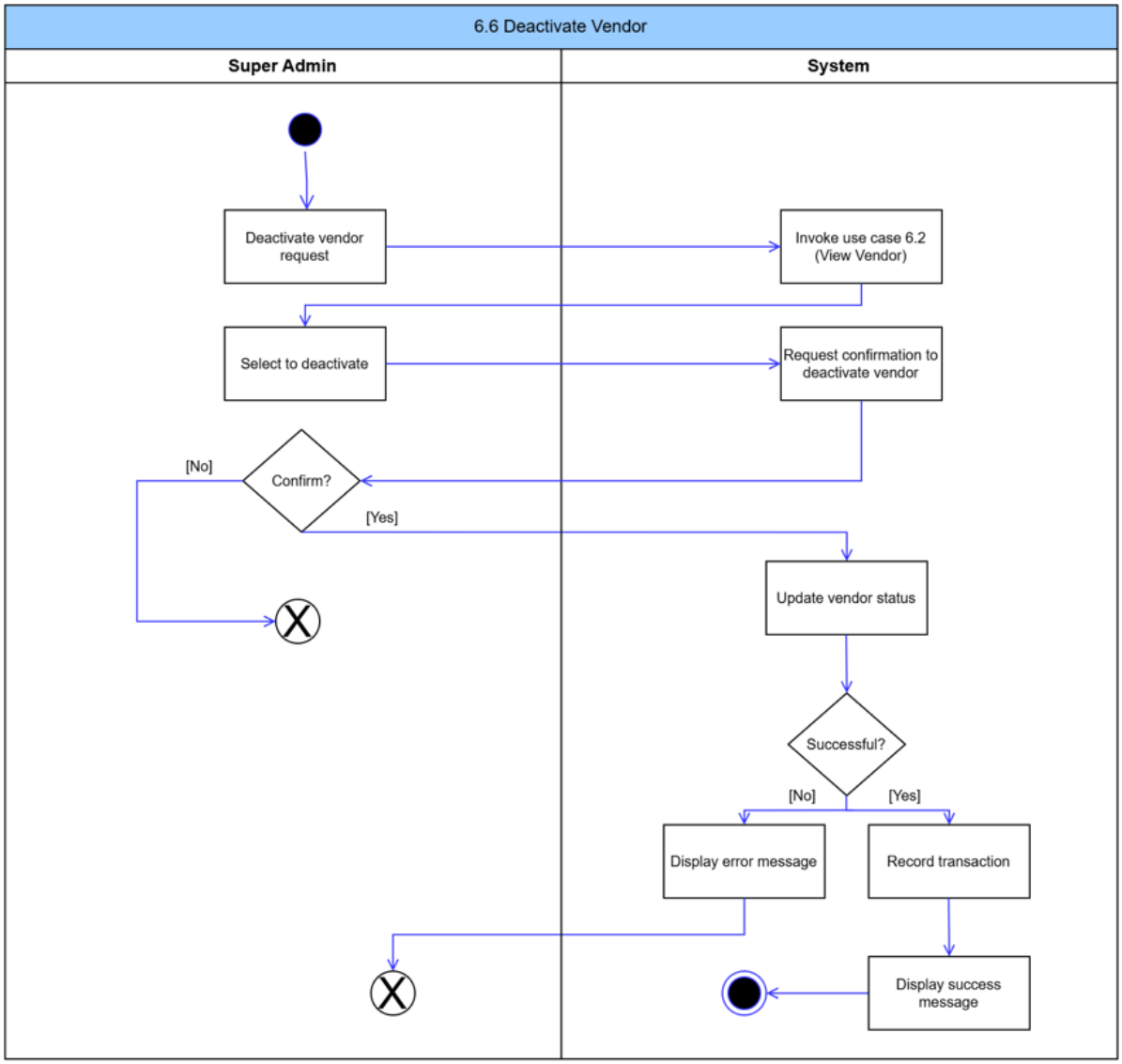


Figure 178 6.6 Deactivate Vendor - Activity Diagram

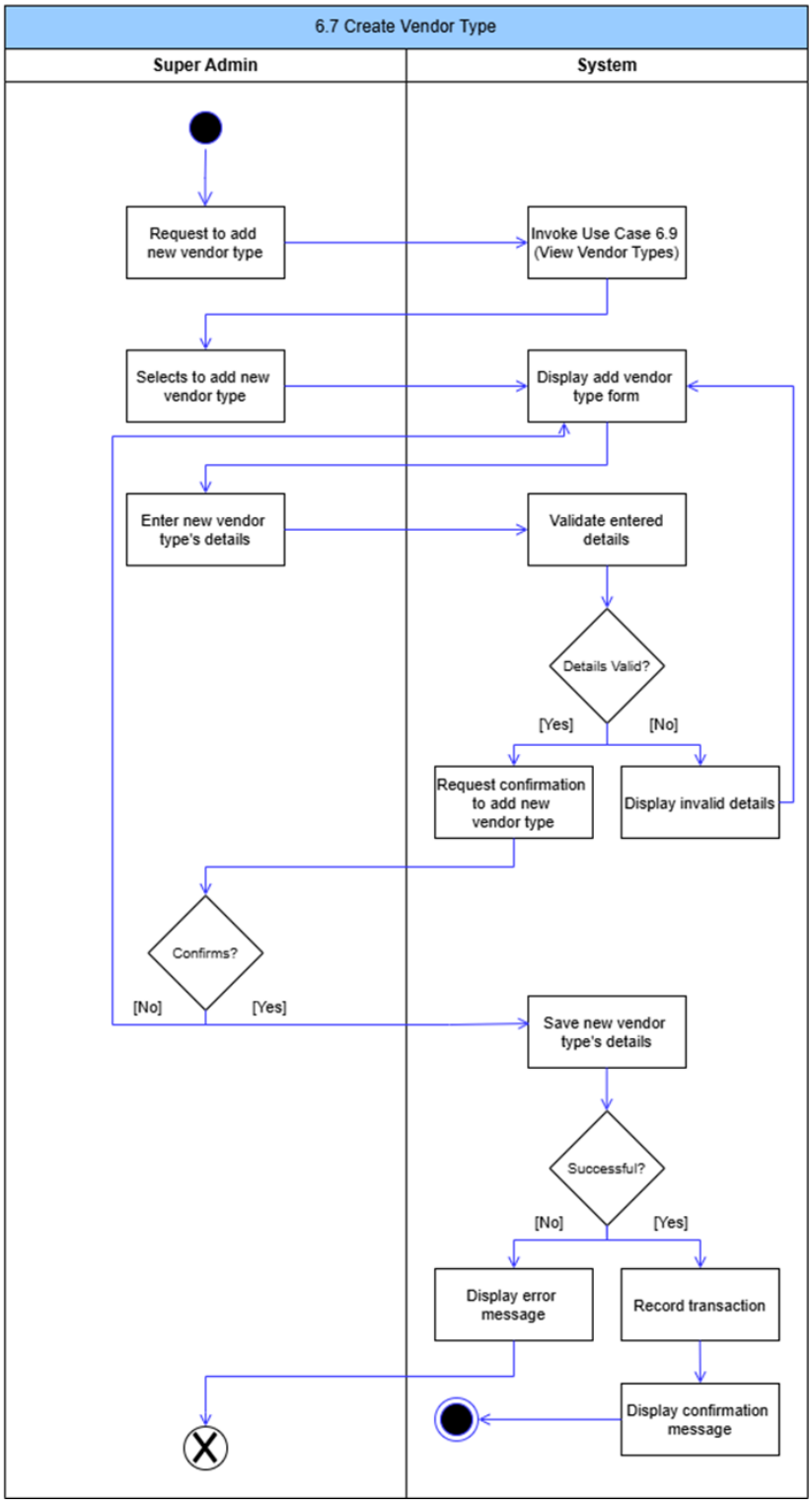


Figure 179 6.7 Create Vendor Type - Activity Diagram

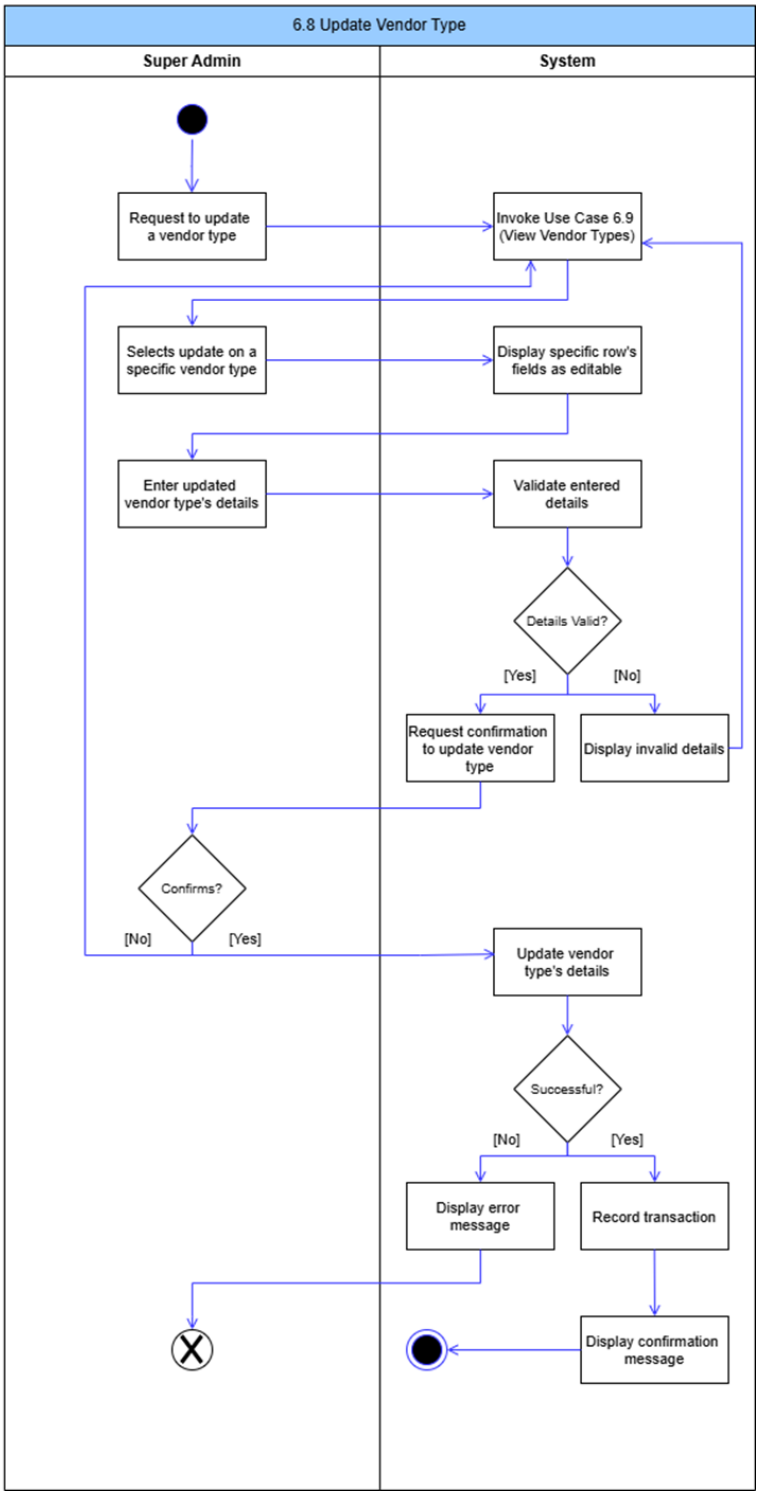


Figure 180 6.8 Update Vendor Type - Activity Diagram

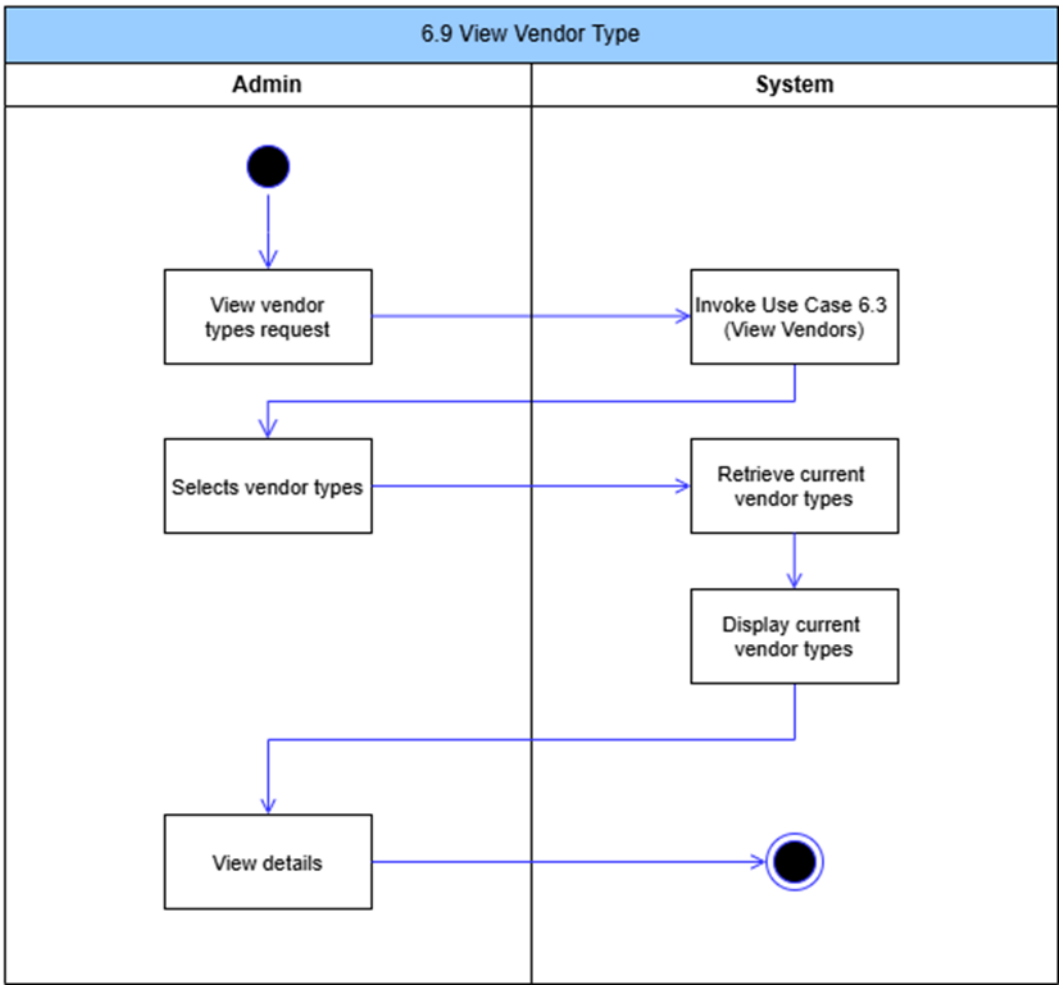


Figure 181 6.9 View Vendor Type - Activity Diagram

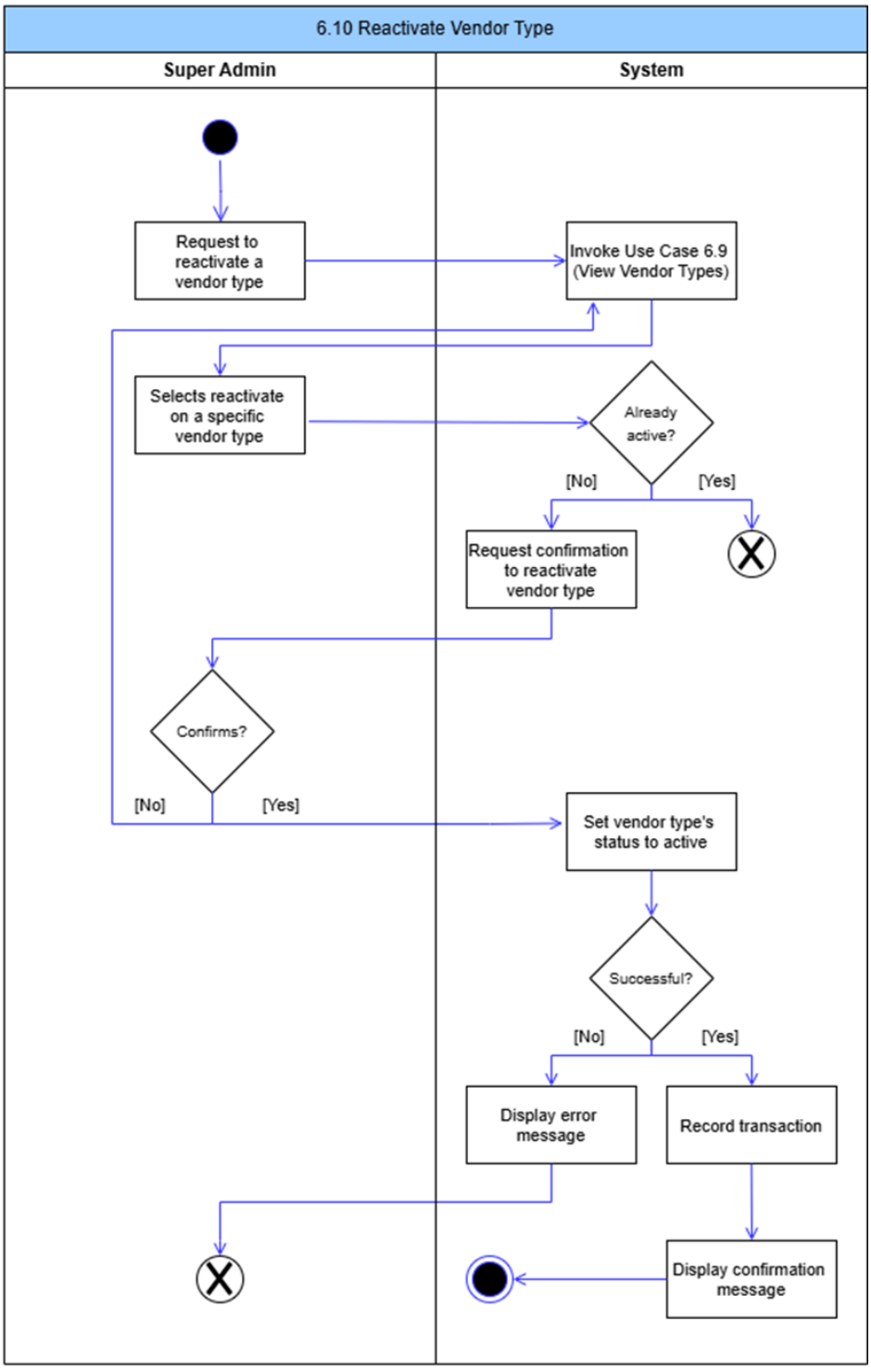


Figure 182 6.10 Reactivate Vendor Type - Activity Diagram

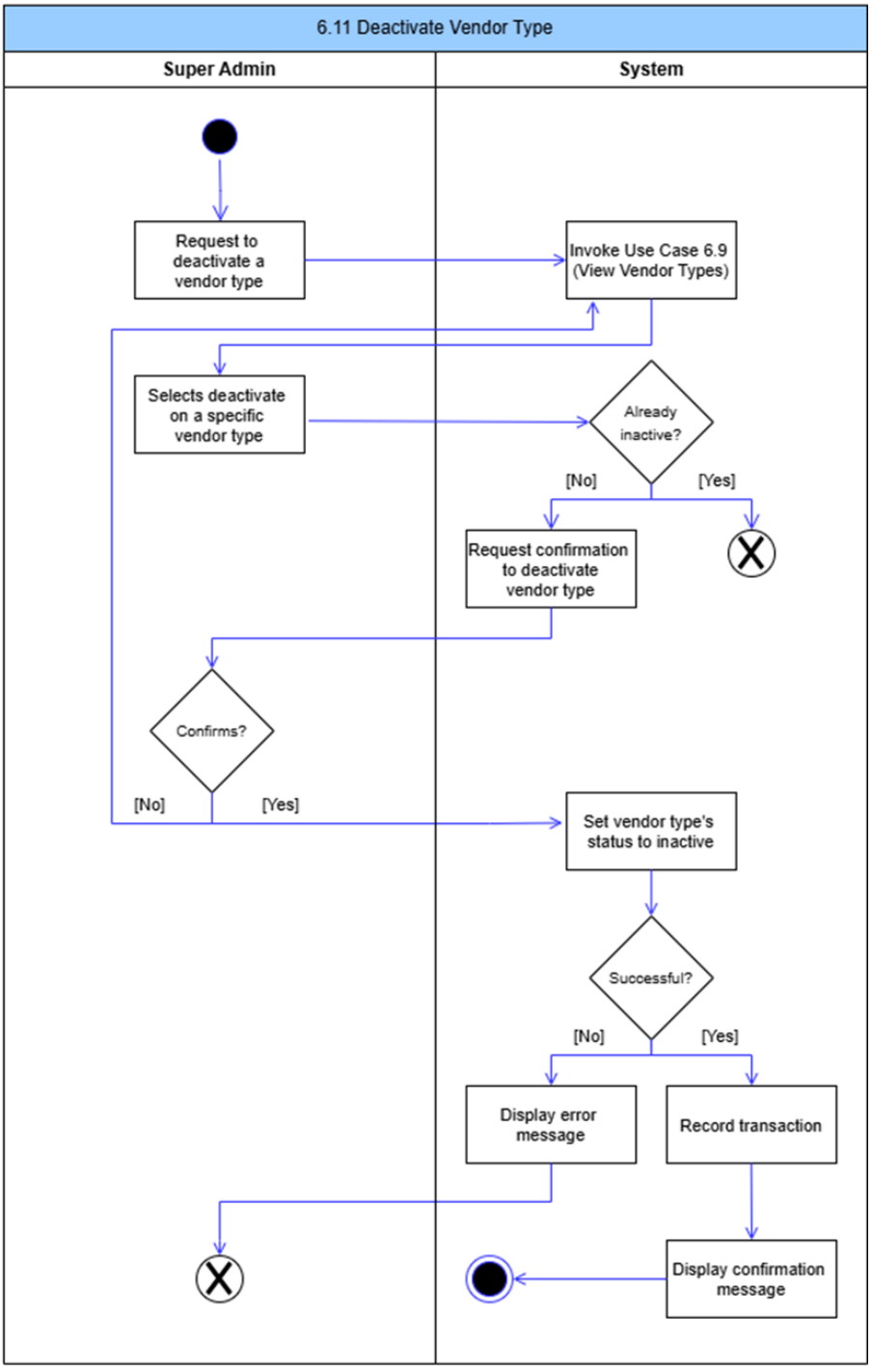


Figure 183 6.11 Deactivate Vendor Type - Activity Diagram

2.10.7 Project Management Subsystem Activity Diagrams

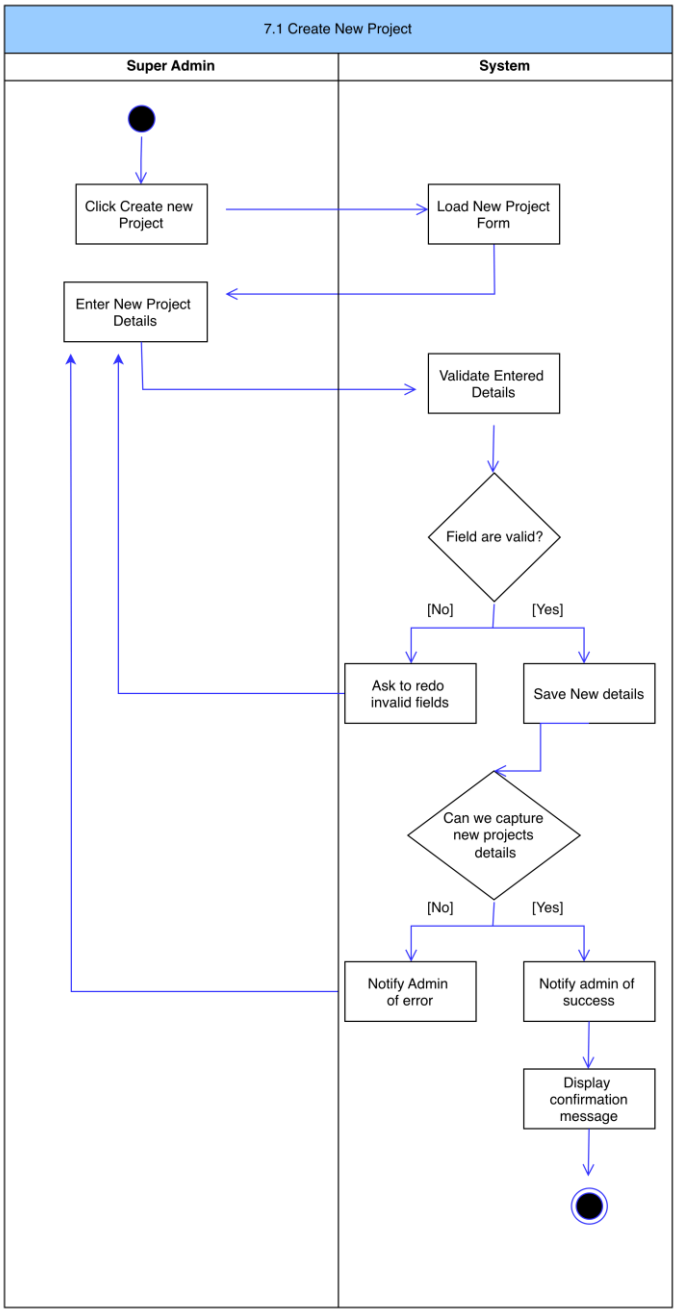


Figure 182 7.1 Create New Project - Activity Diagram

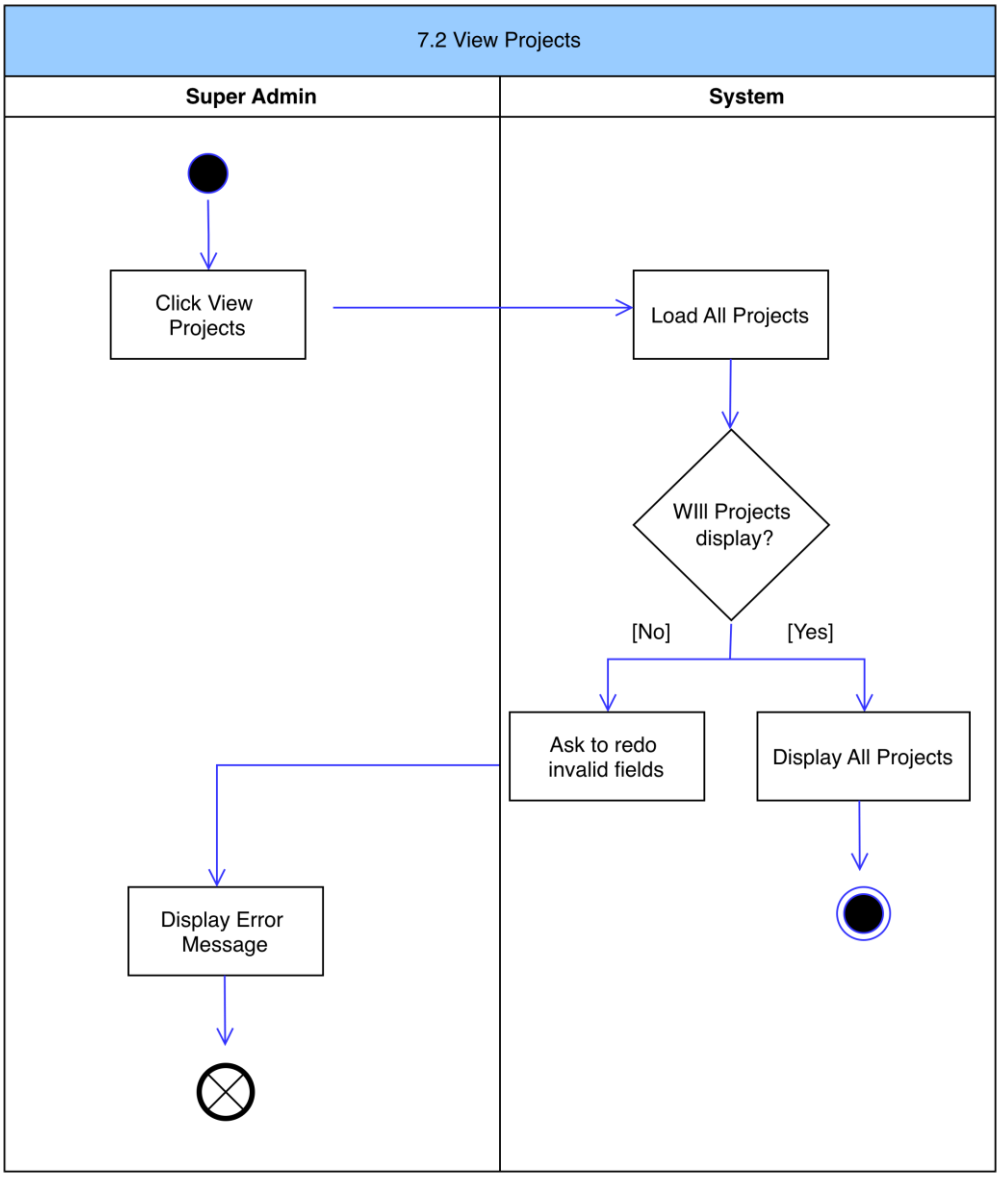


Figure 183 7.2 View Projects - Activity Diagram

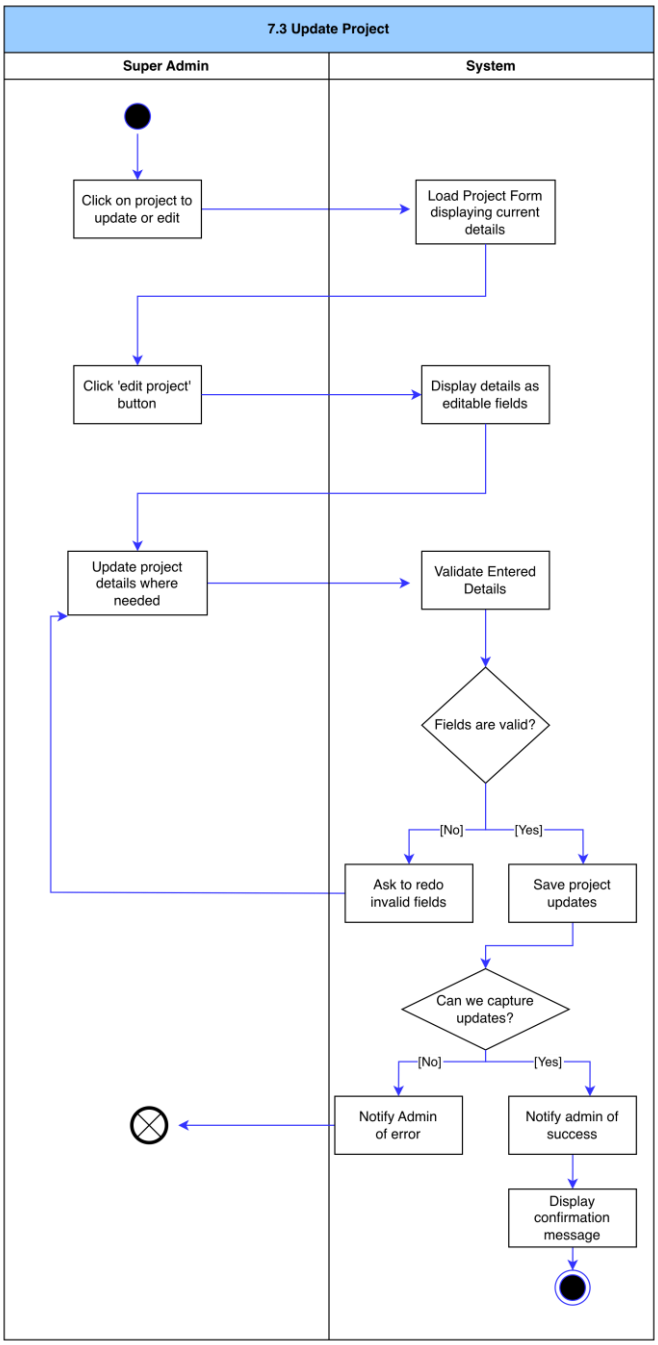


Figure 184 7.3 Update Projects - Activity Diagram

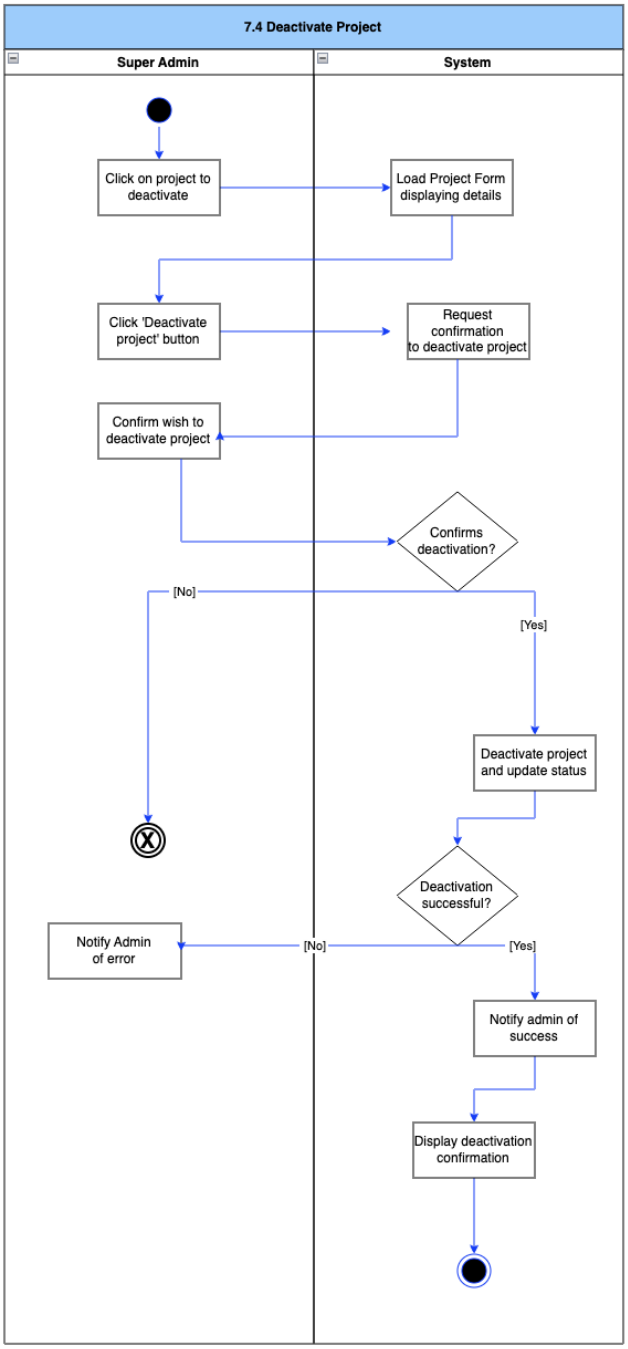


Figure 185 7.4 Deactivate Projects - Activity Diagram

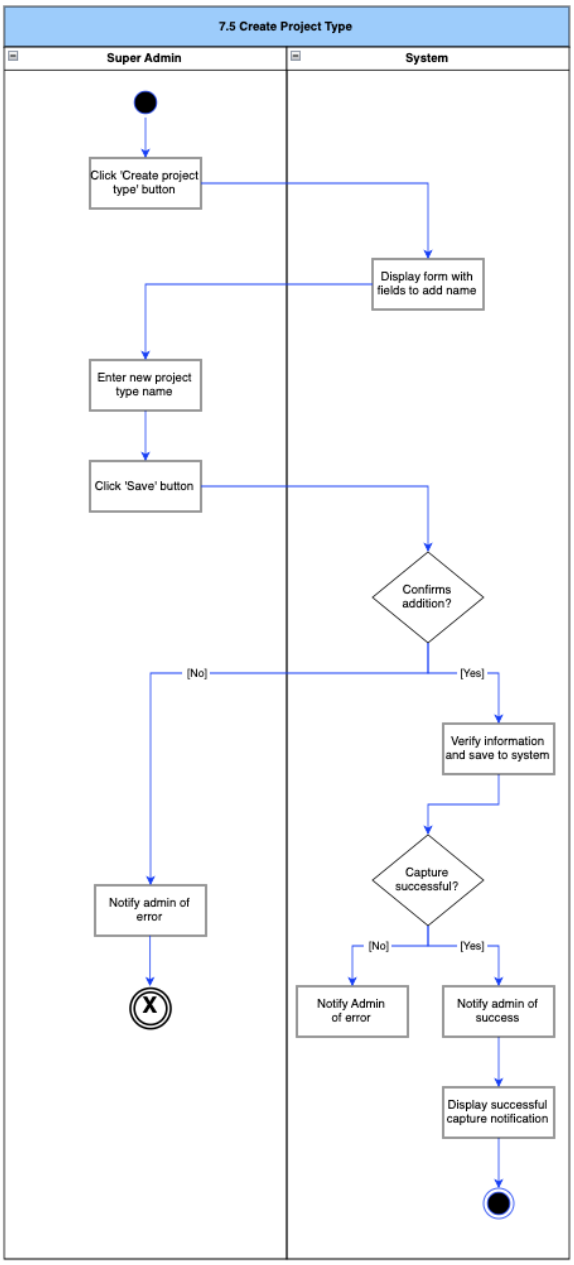


Figure 186 7.5 Create Project Type - Activity Diagram

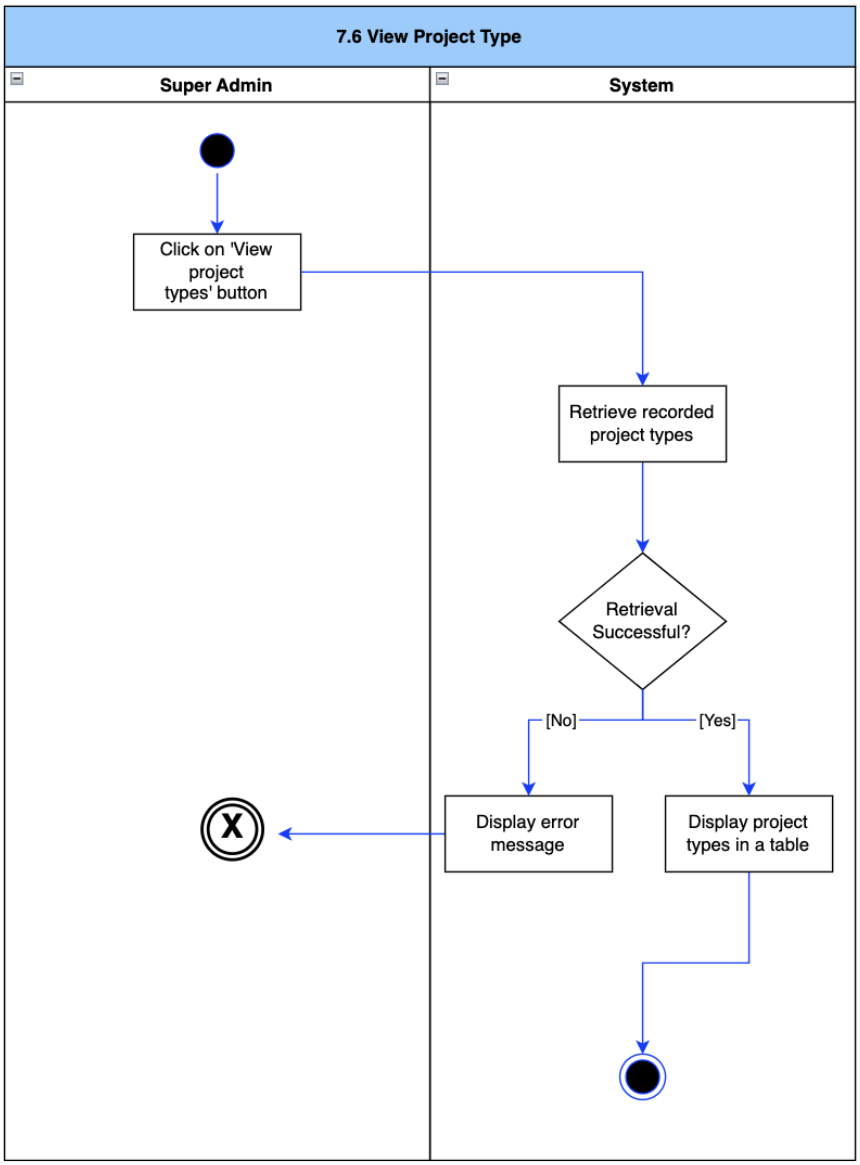


Figure 187 7.6 View Project Type - Activity Diagram

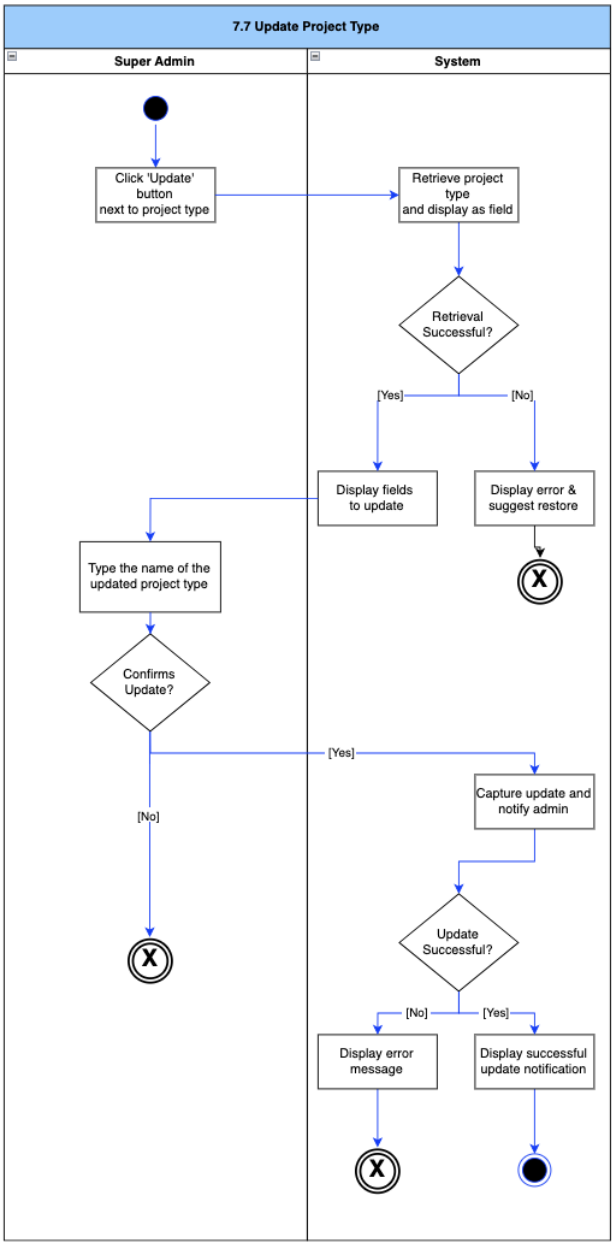


Figure 188 7.7 Update Project Type - Activity Diagram

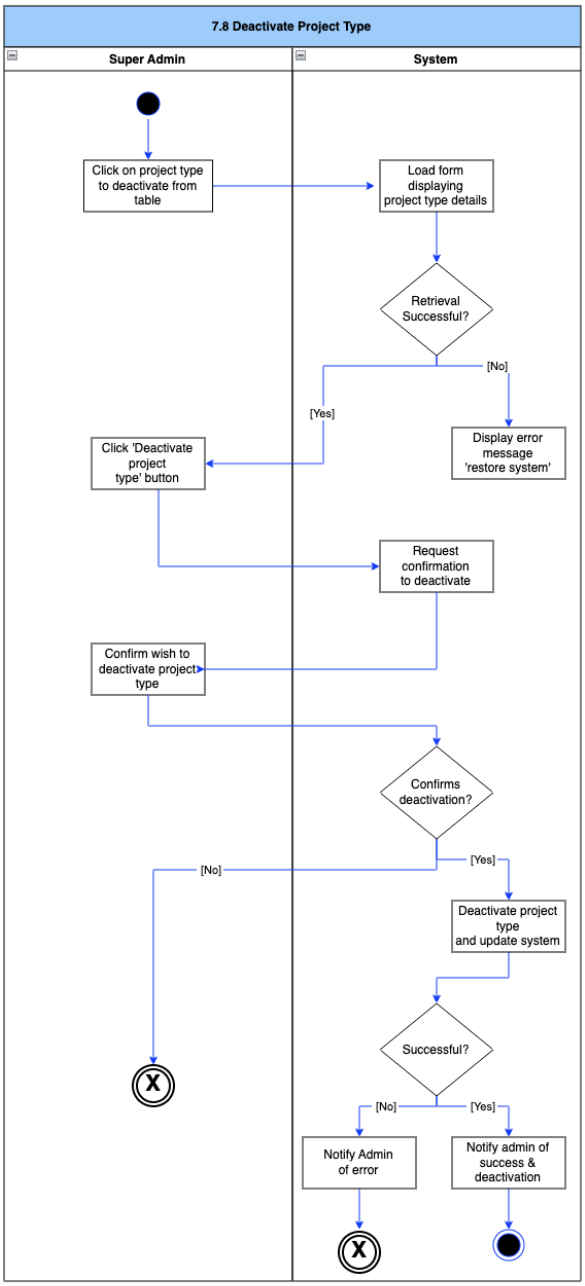


Figure 189 7.8 Deactivate Project Type - Activity Diagram

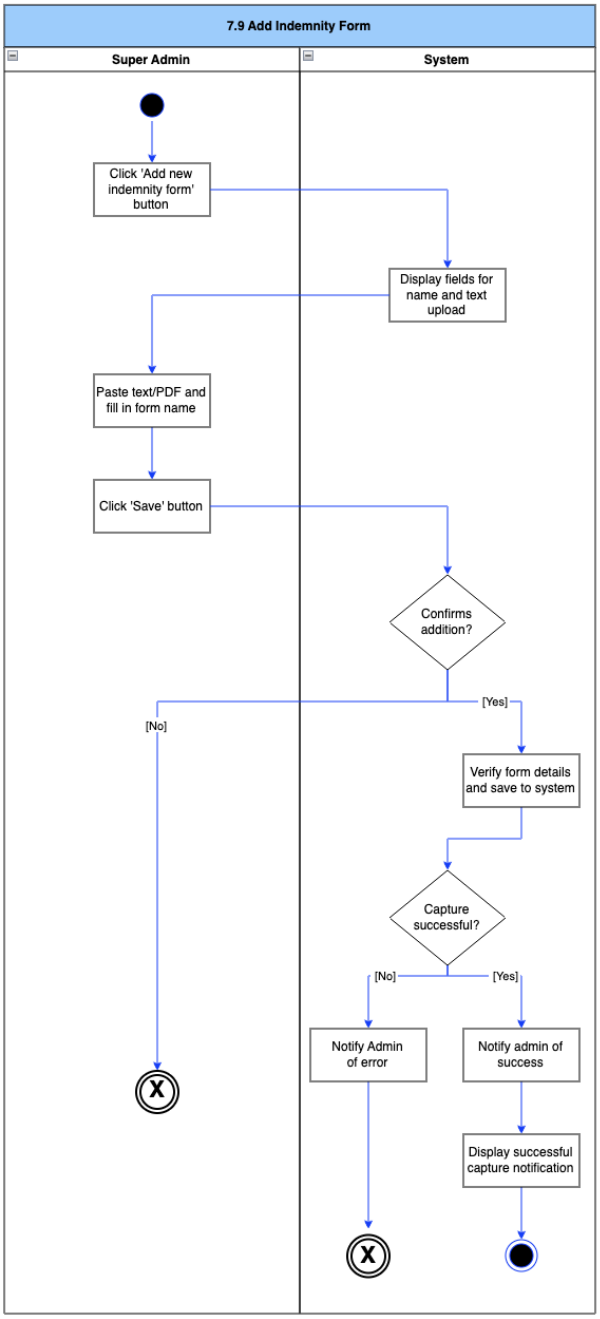


Figure 190 7.9 Add Indemnity Form - Activity Diagram

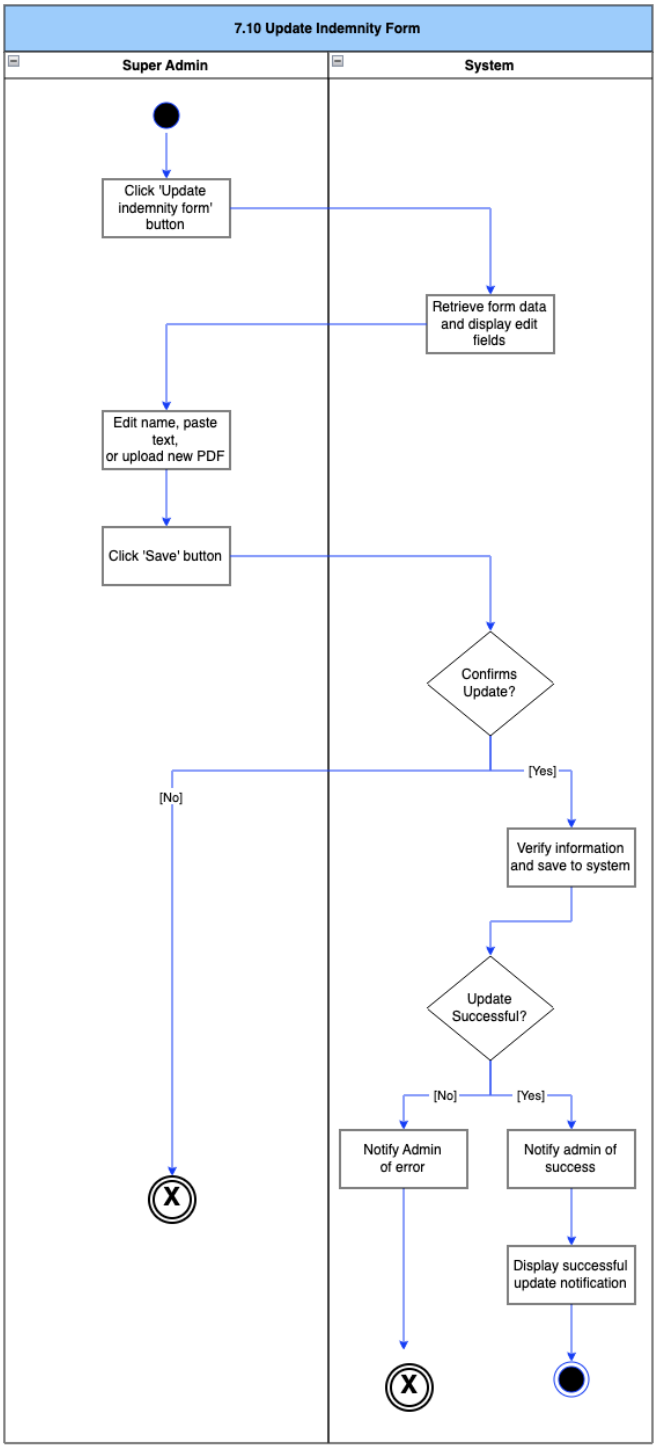


Figure 191 7.10 Update Indemnity Form - Activity Diagram

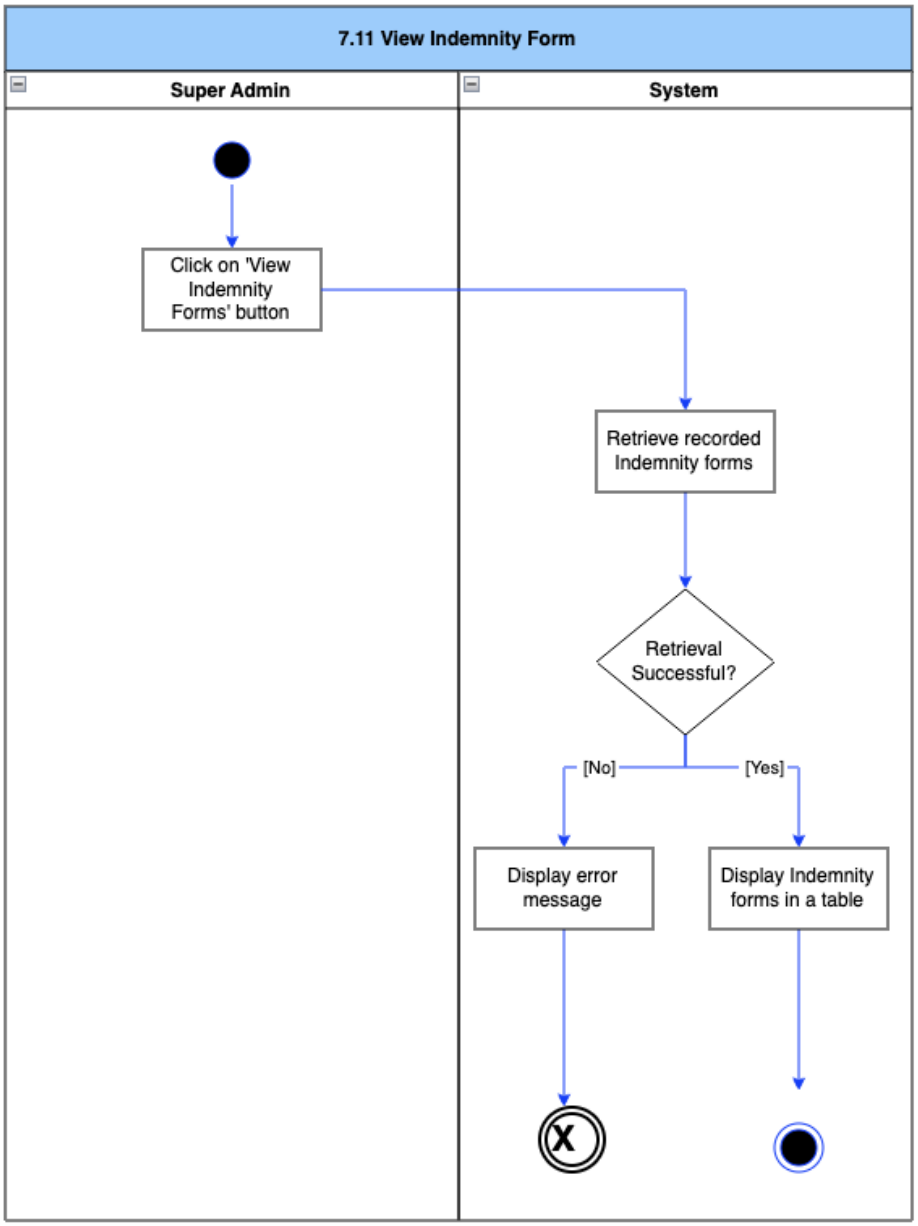


Figure 192 7.11 View Indemnity Form- Activity Diagram

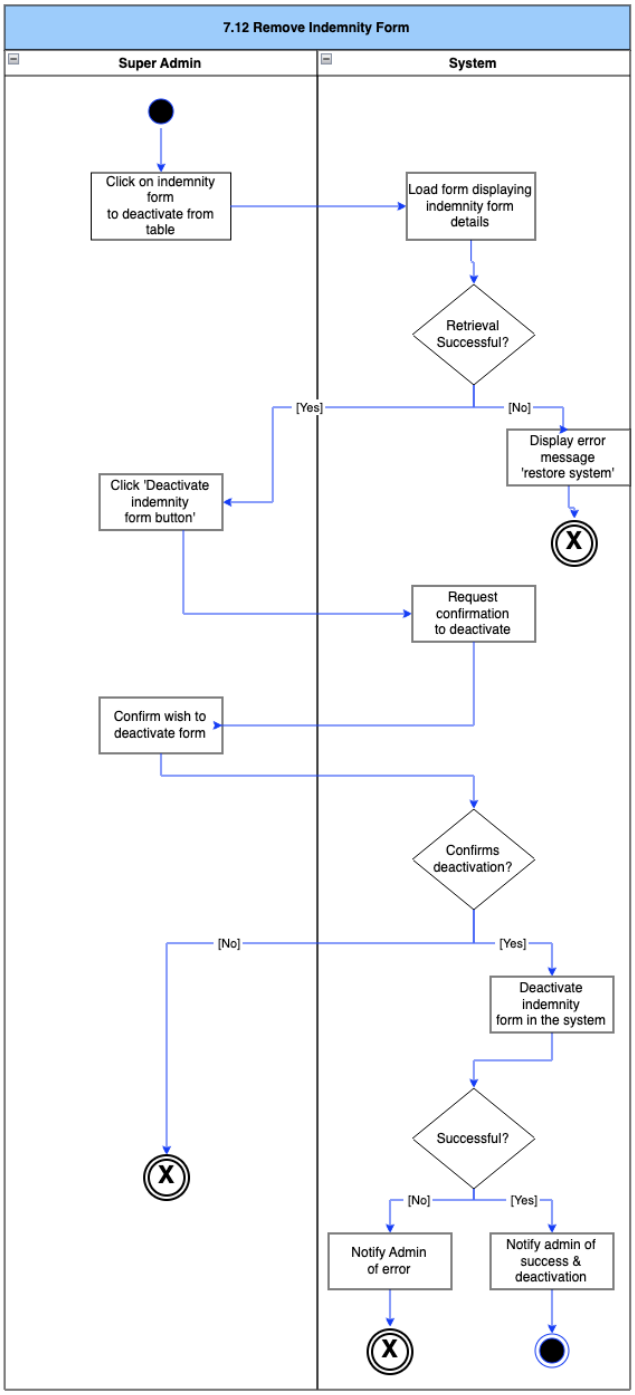


Figure 193 7.12 Remove Indemnity Form - Activity Diagram

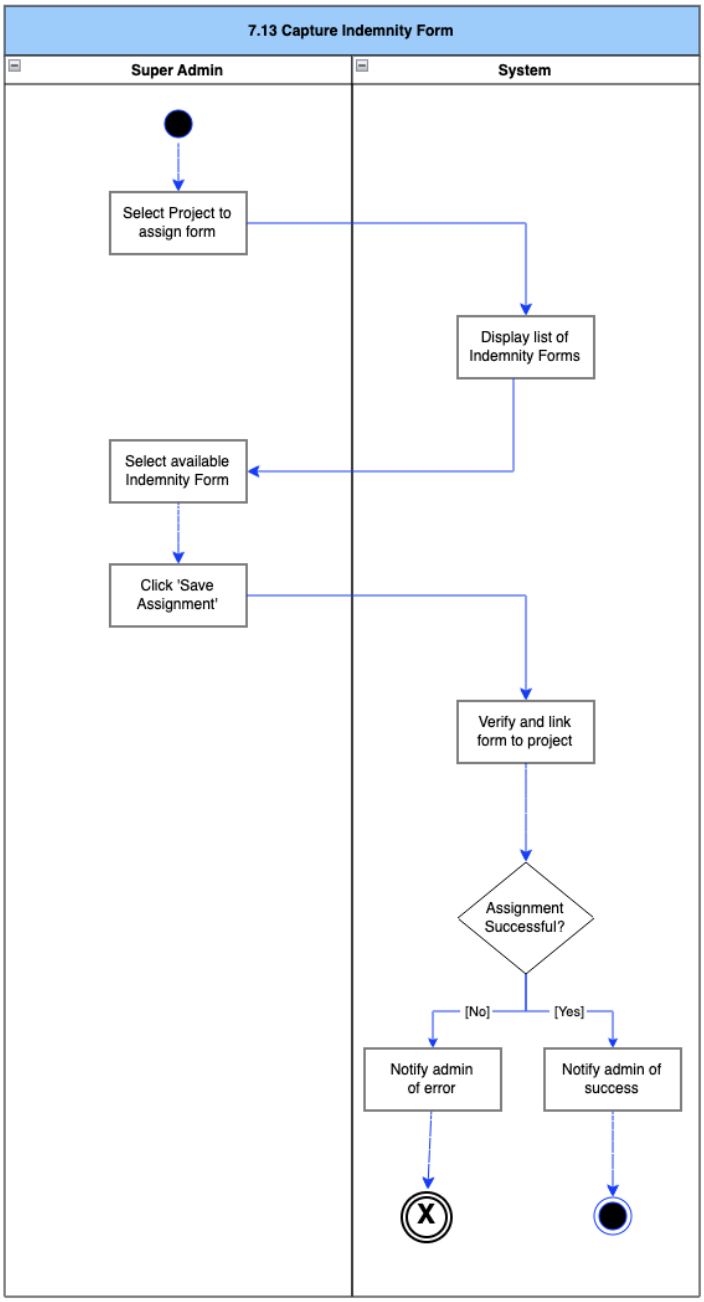


Figure 194 7.13 Capture Indemnity Form - Activity Diagram

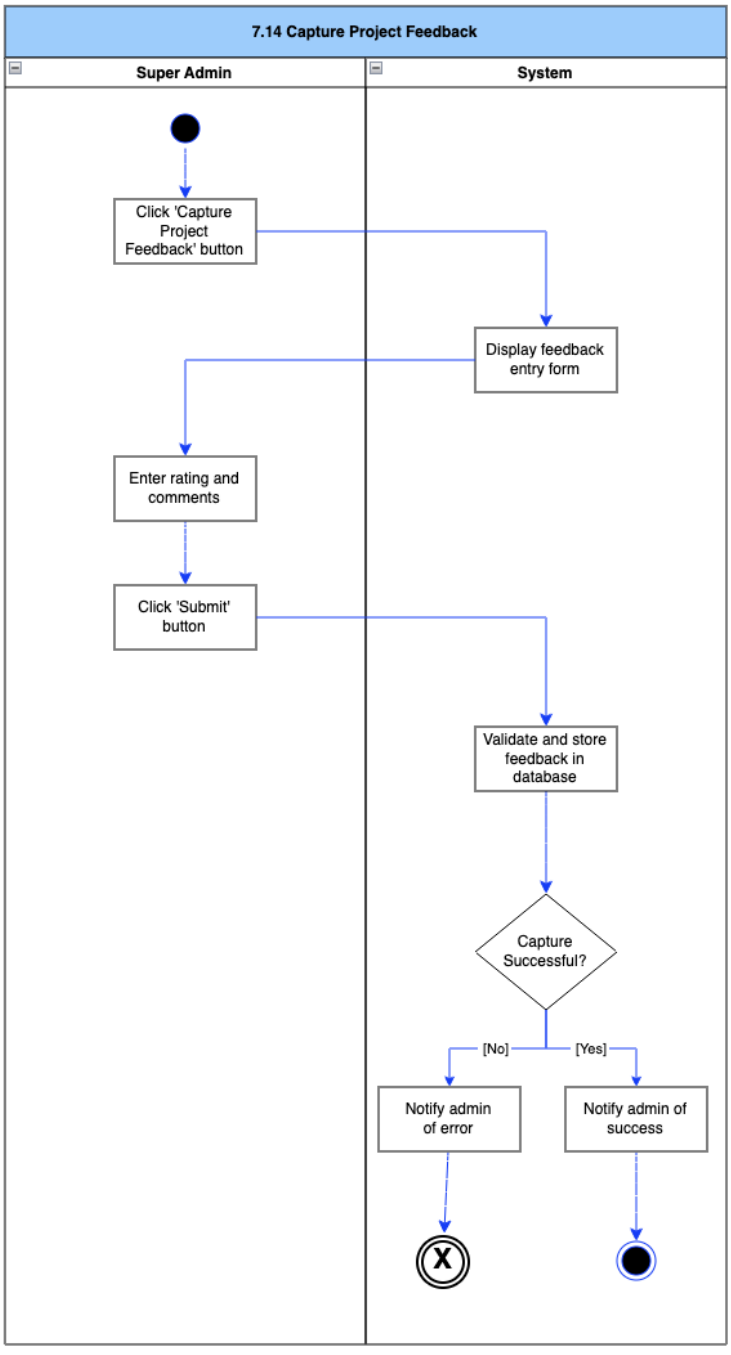


Figure 195 7.14 Capture Project Feedback - Activity Diagram

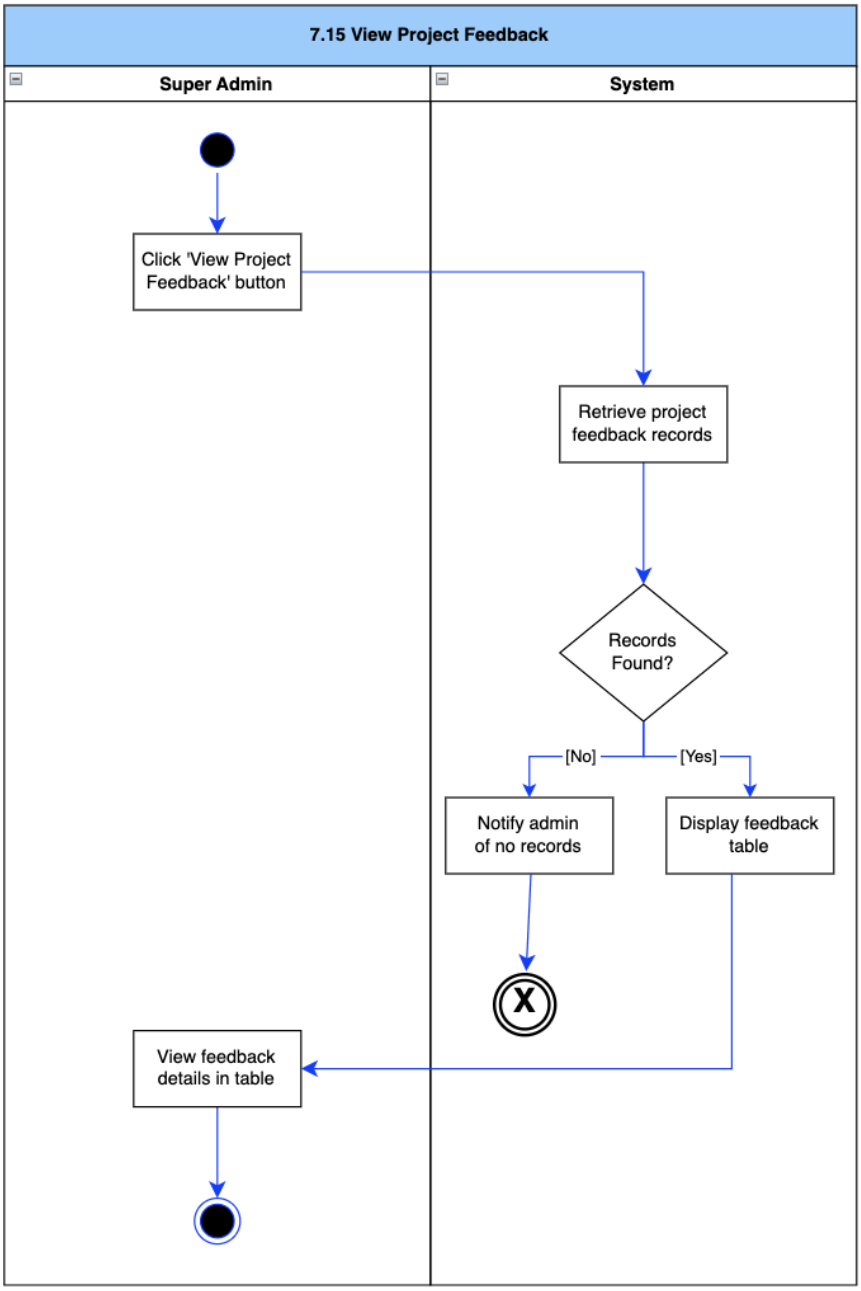


Figure 196 7.15 View Project Feedback - Activity Diagram

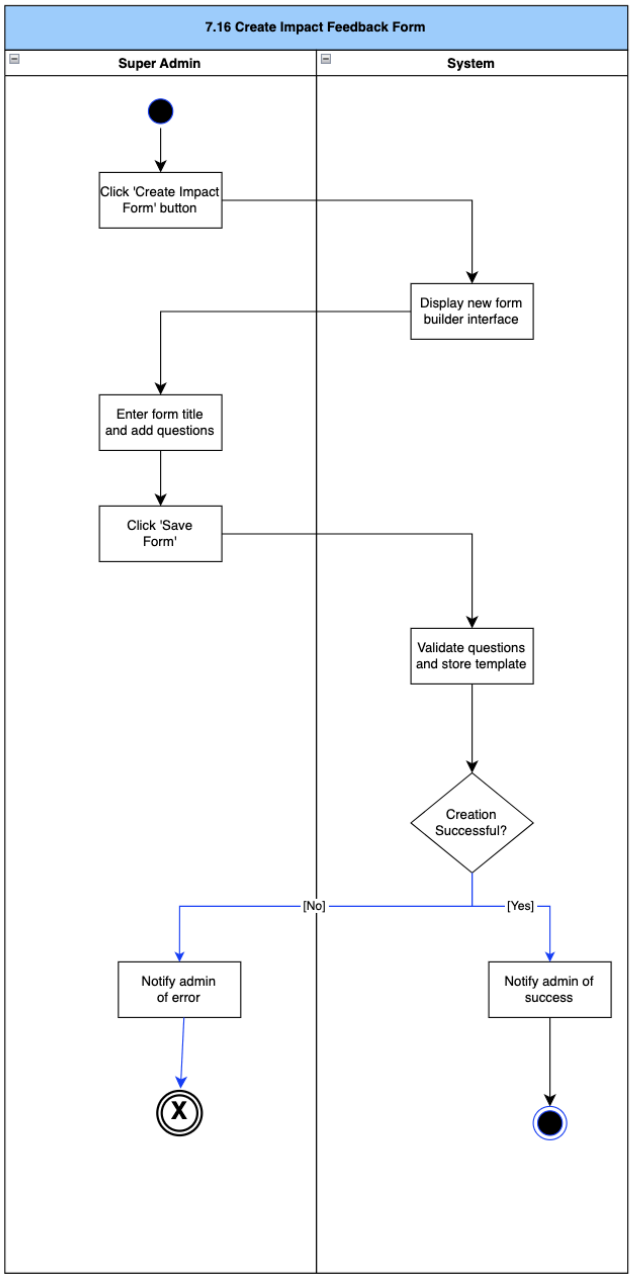


Figure 197 7.16 Create Project Feedback Form - Activity Diagram

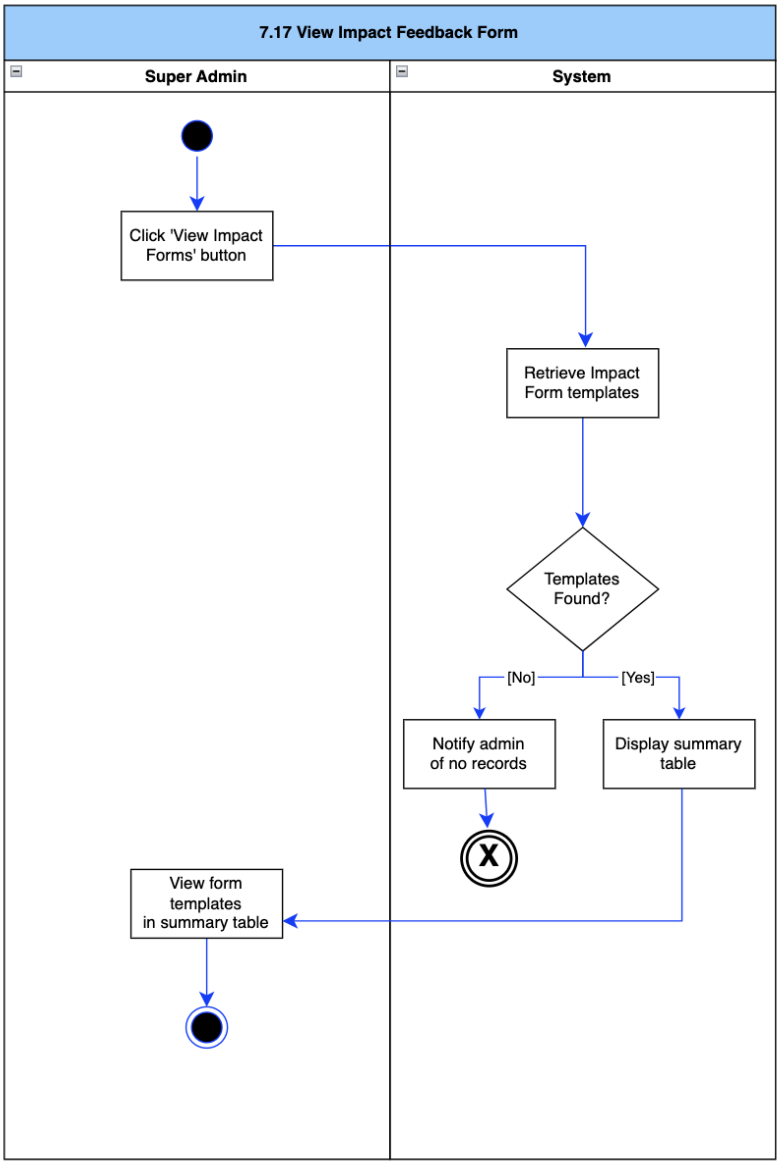


Figure 198 7.17 View Impact Feedback Form - Activity Diagram

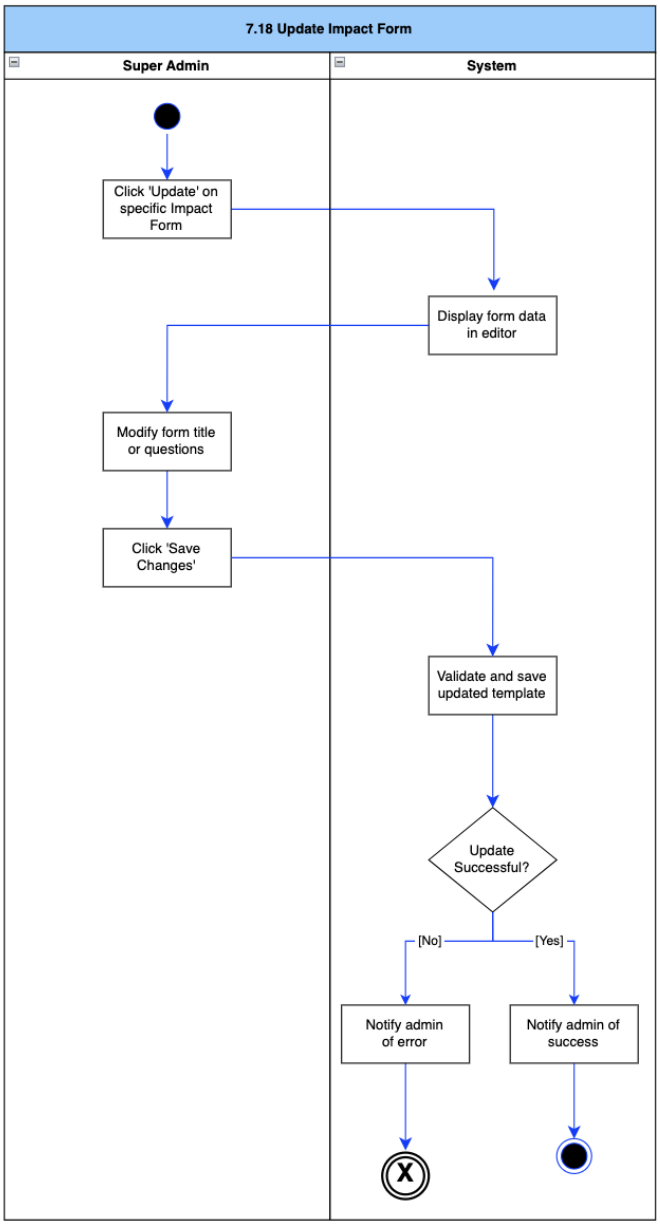


Figure 199 7.18 Update Impact Form - Activity Diagram

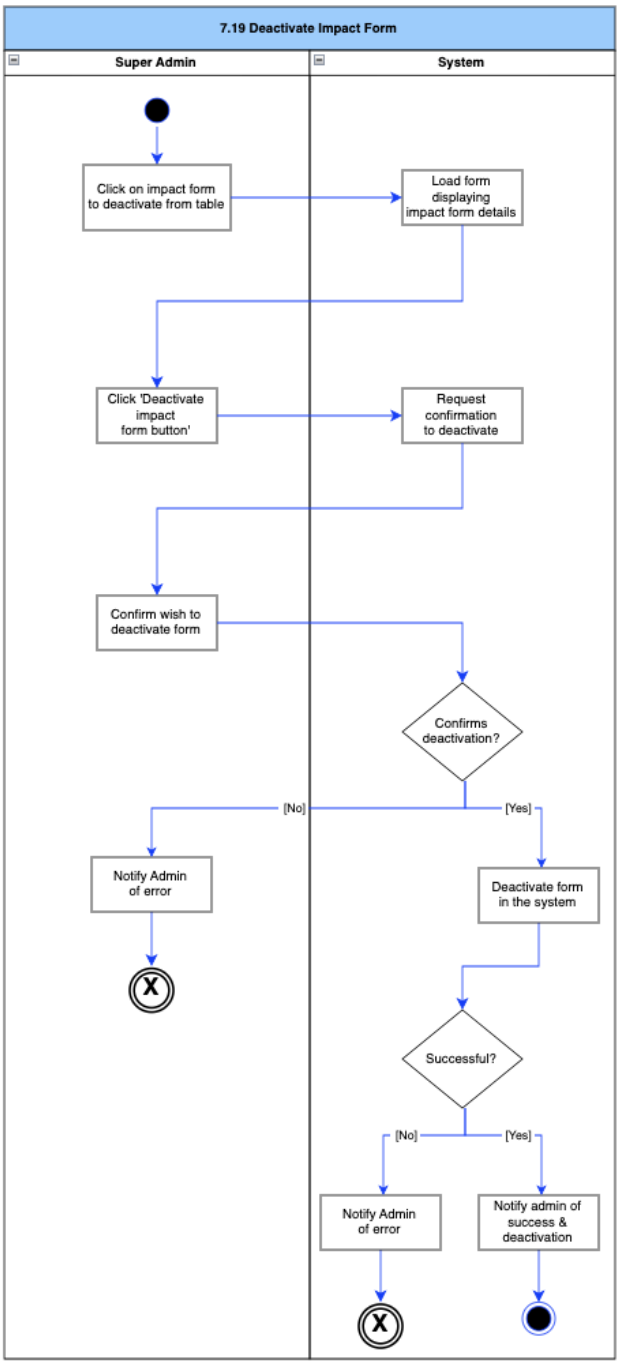


Figure 200 7.19 Deactivate Impact Form - Activity Diagram

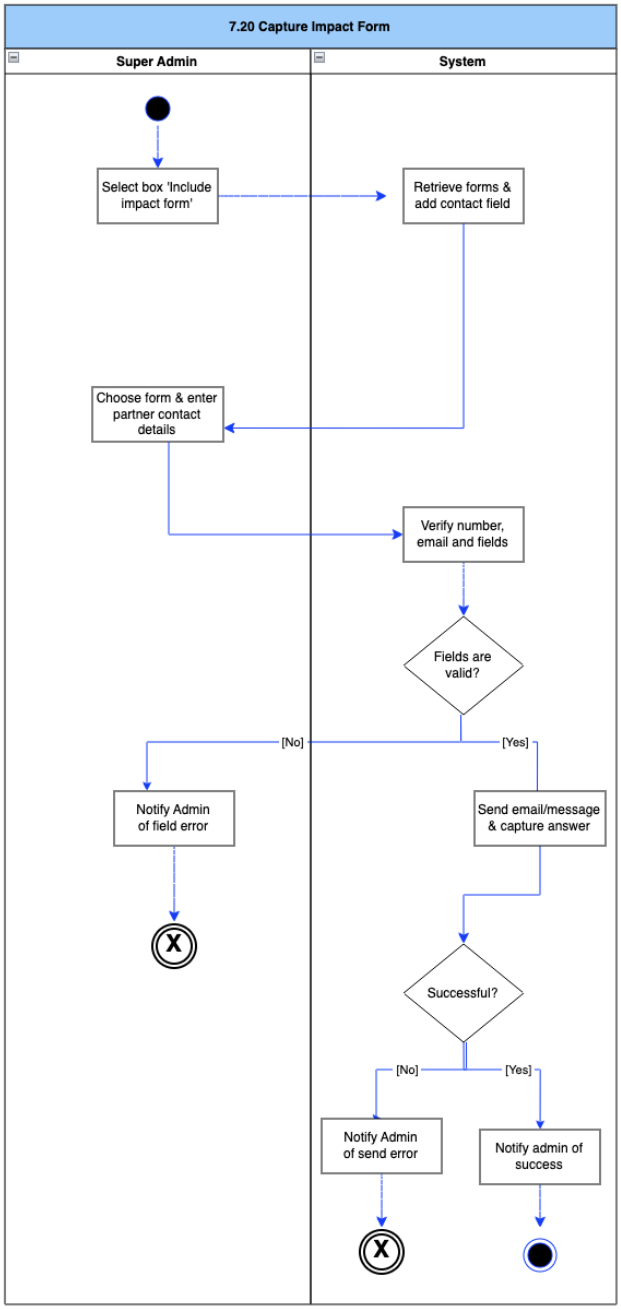


Figure 201 7.20 Capture Impact Form - Activity Diagram

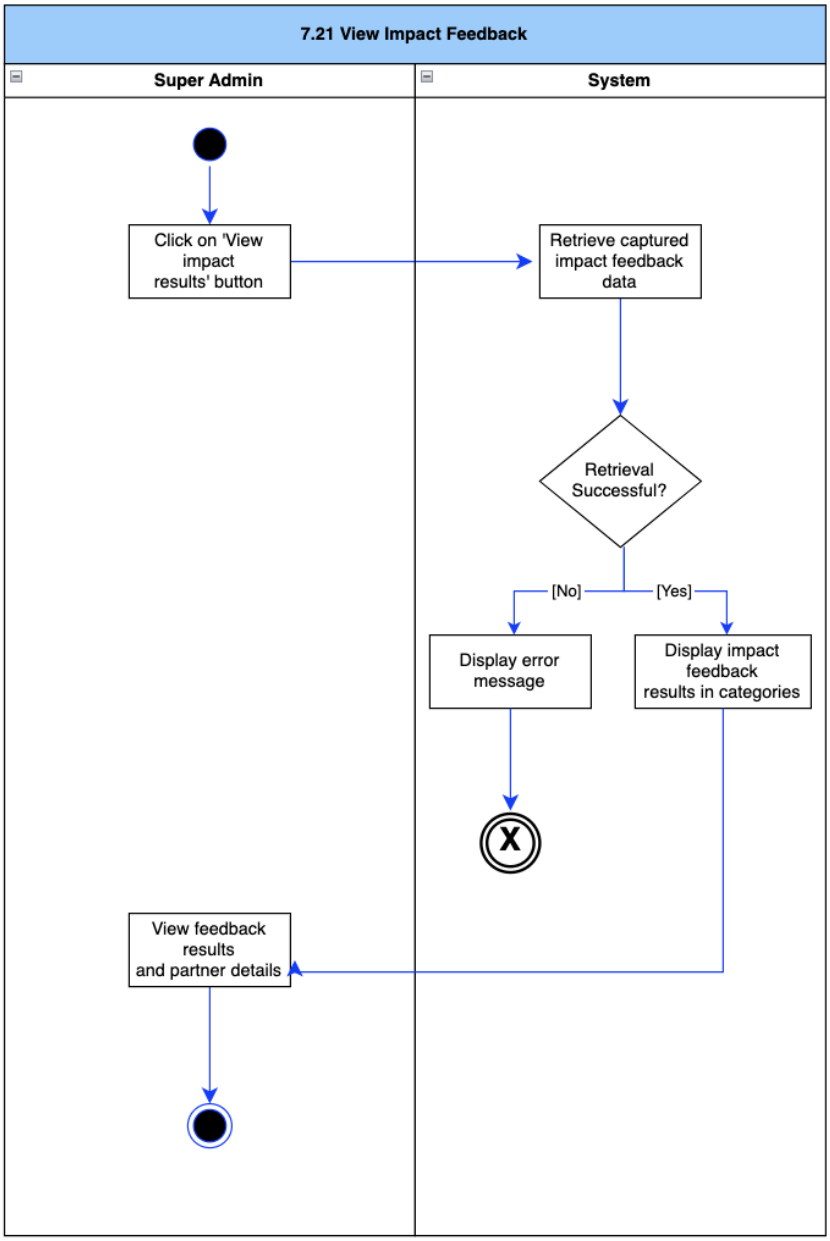


Figure 202 7.21 View Impact Form - Activity Diagram

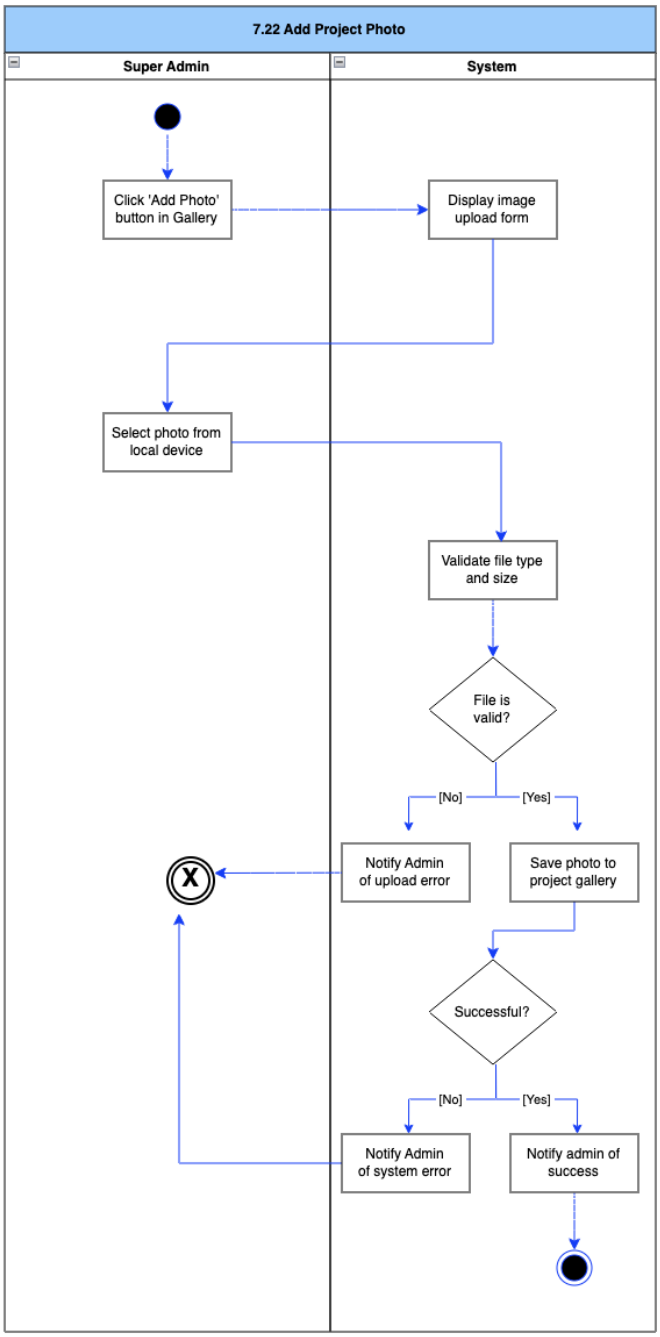


Figure 203 7.22 Add Project Photo - Activity Diagram

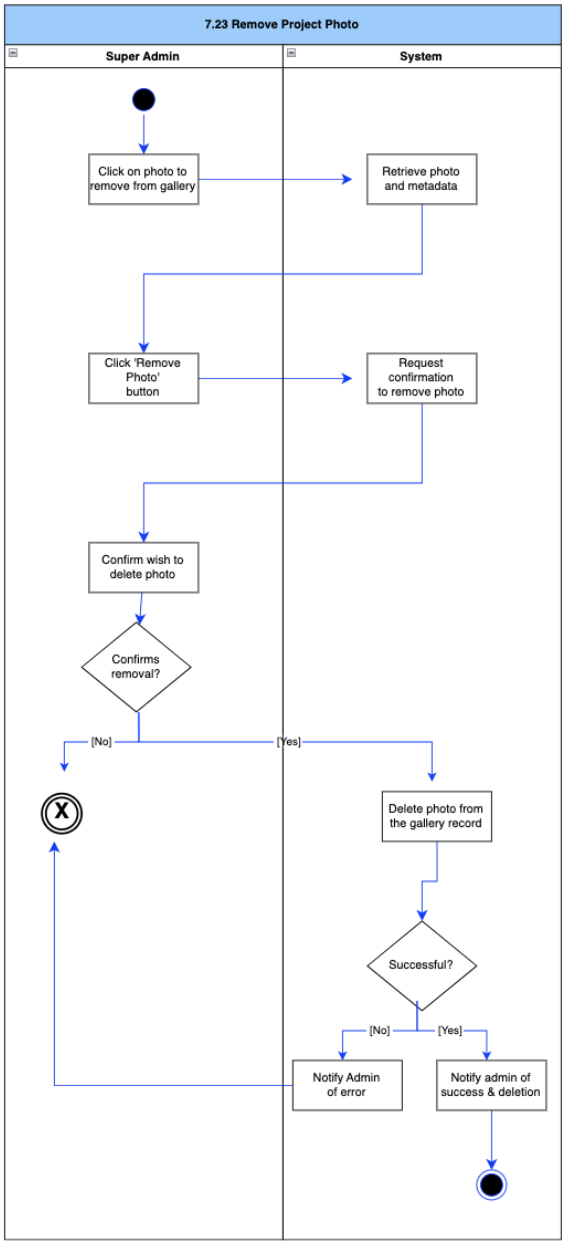


Figure 204 7.23 Remove Project Photo - Activity Diagram

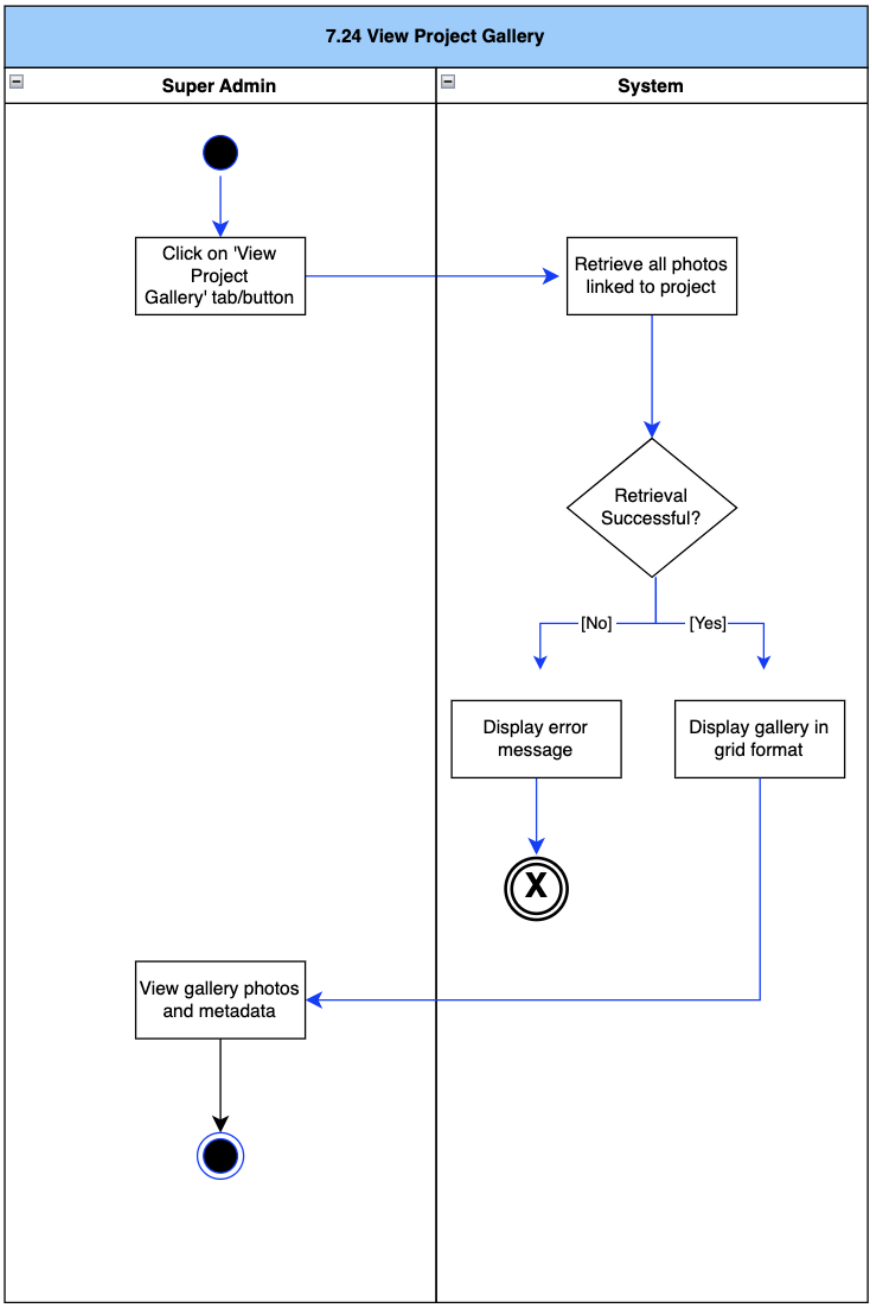


Figure 205 7.24 View Project Photo - Activity Diagram

2.10.8 K9 Subsystem Activity Diagrams

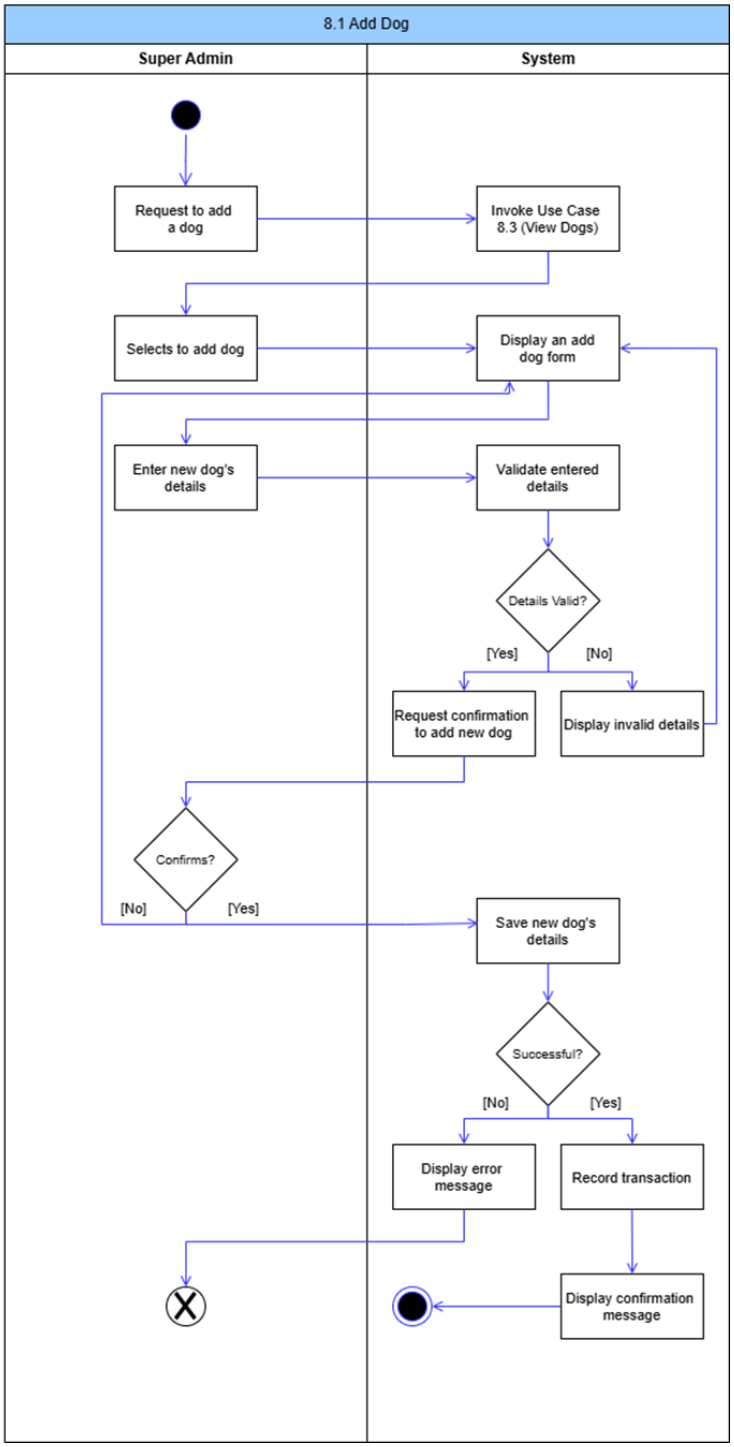


Figure 194 8.1 Add Dog - Activity Diagram

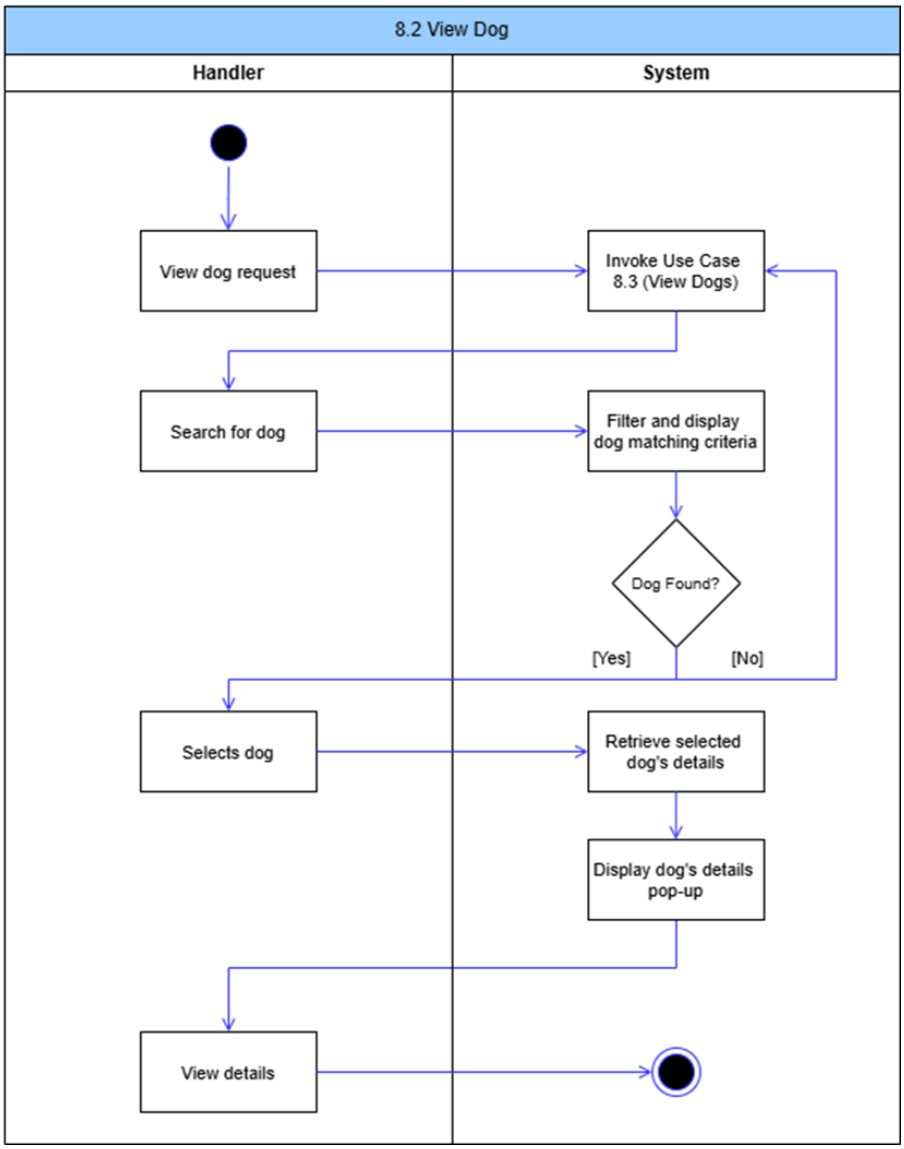


Figure 195 8.2 View Dog - Activity Diagram

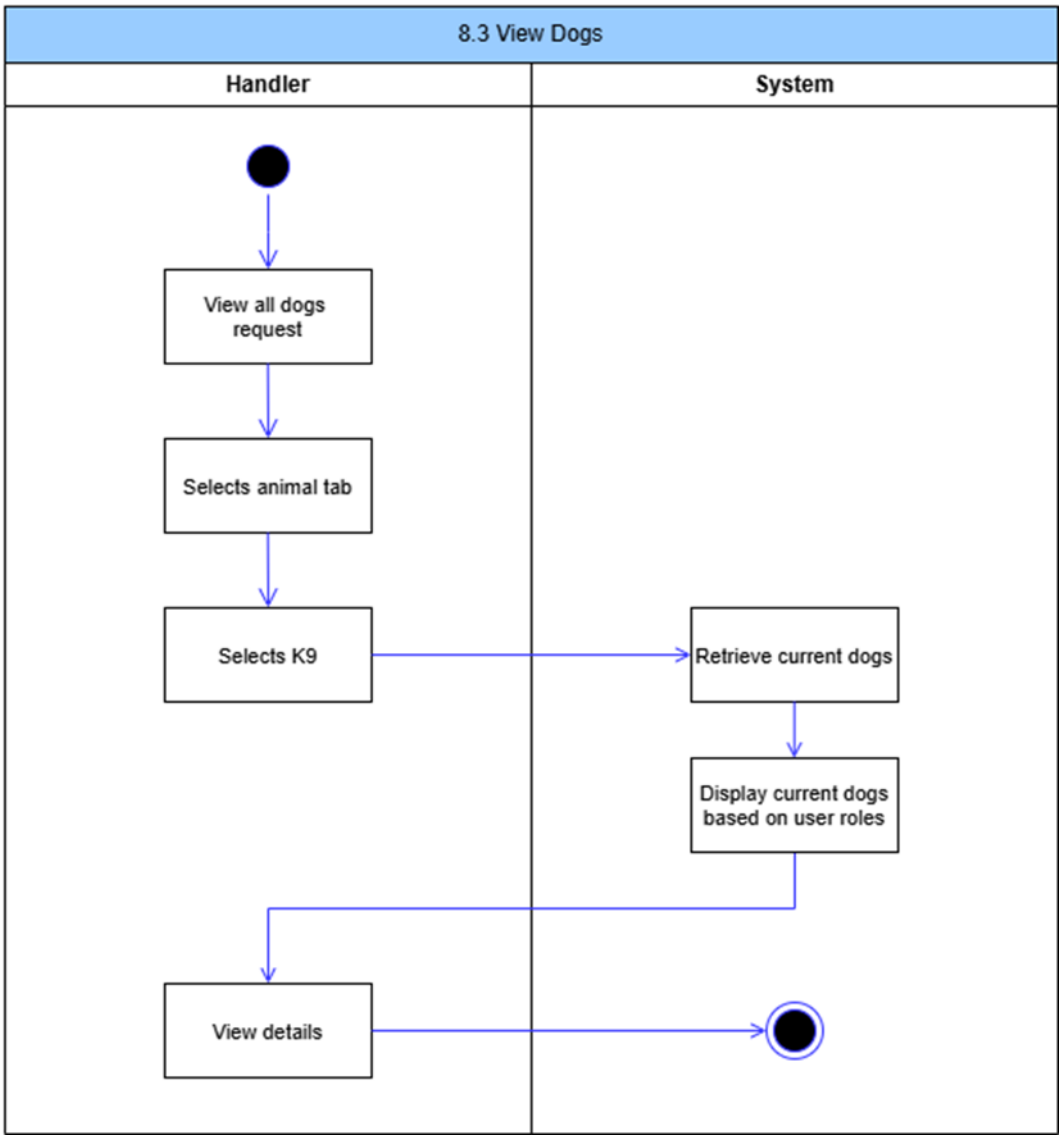


Figure 195 8.3 View Dogs - Activity Diagram

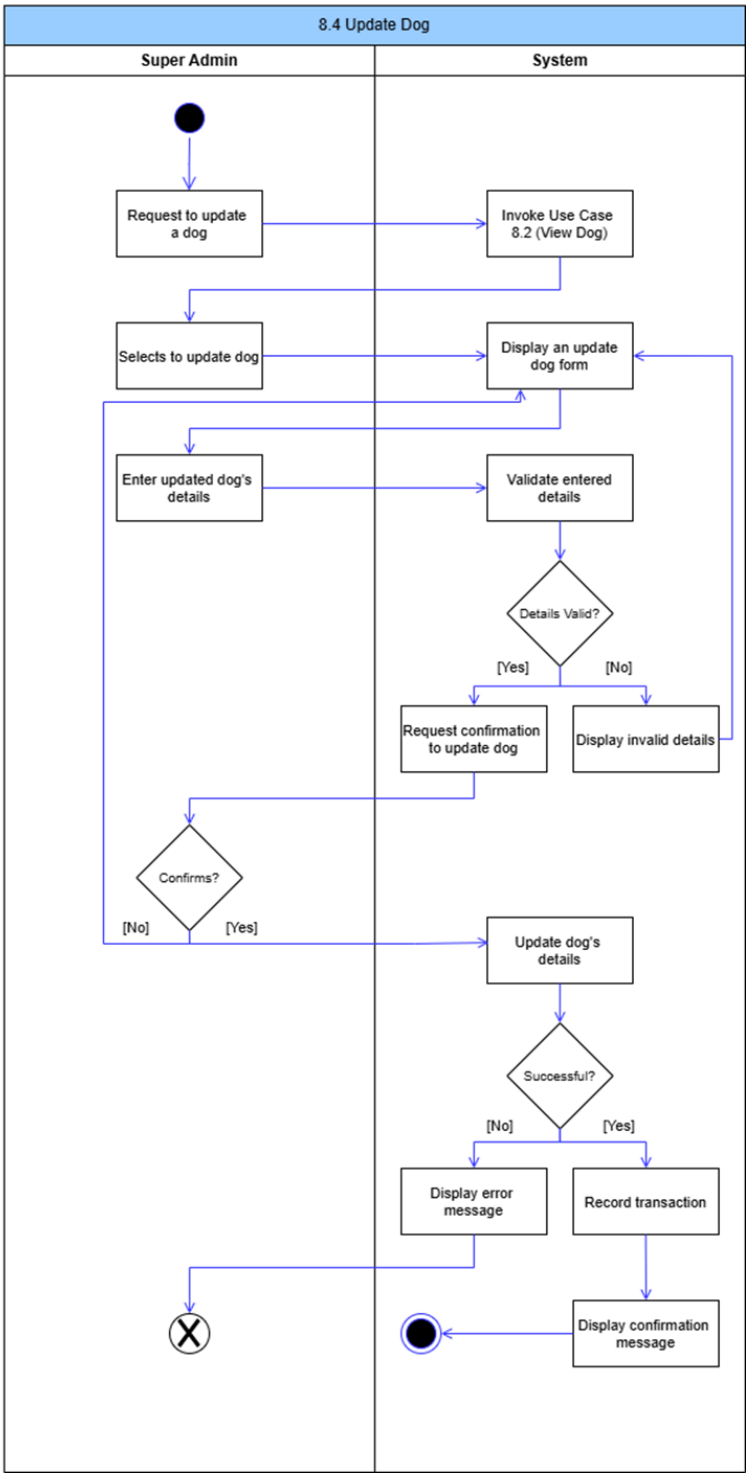


Figure 196 8.4 Update Dog - Activity Diagram

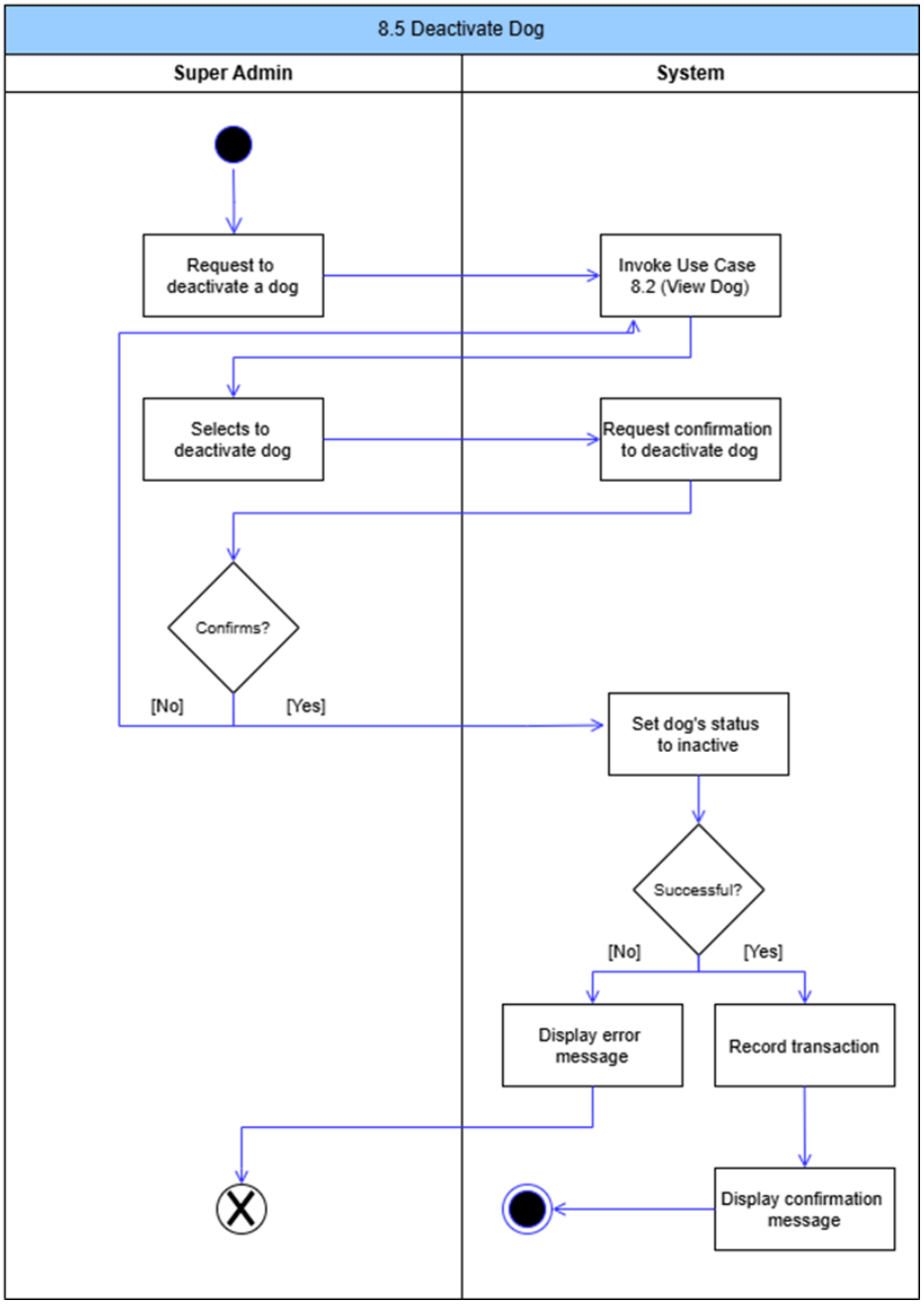


Figure 184 8.5 Deactivate Dog - Activity Diagram

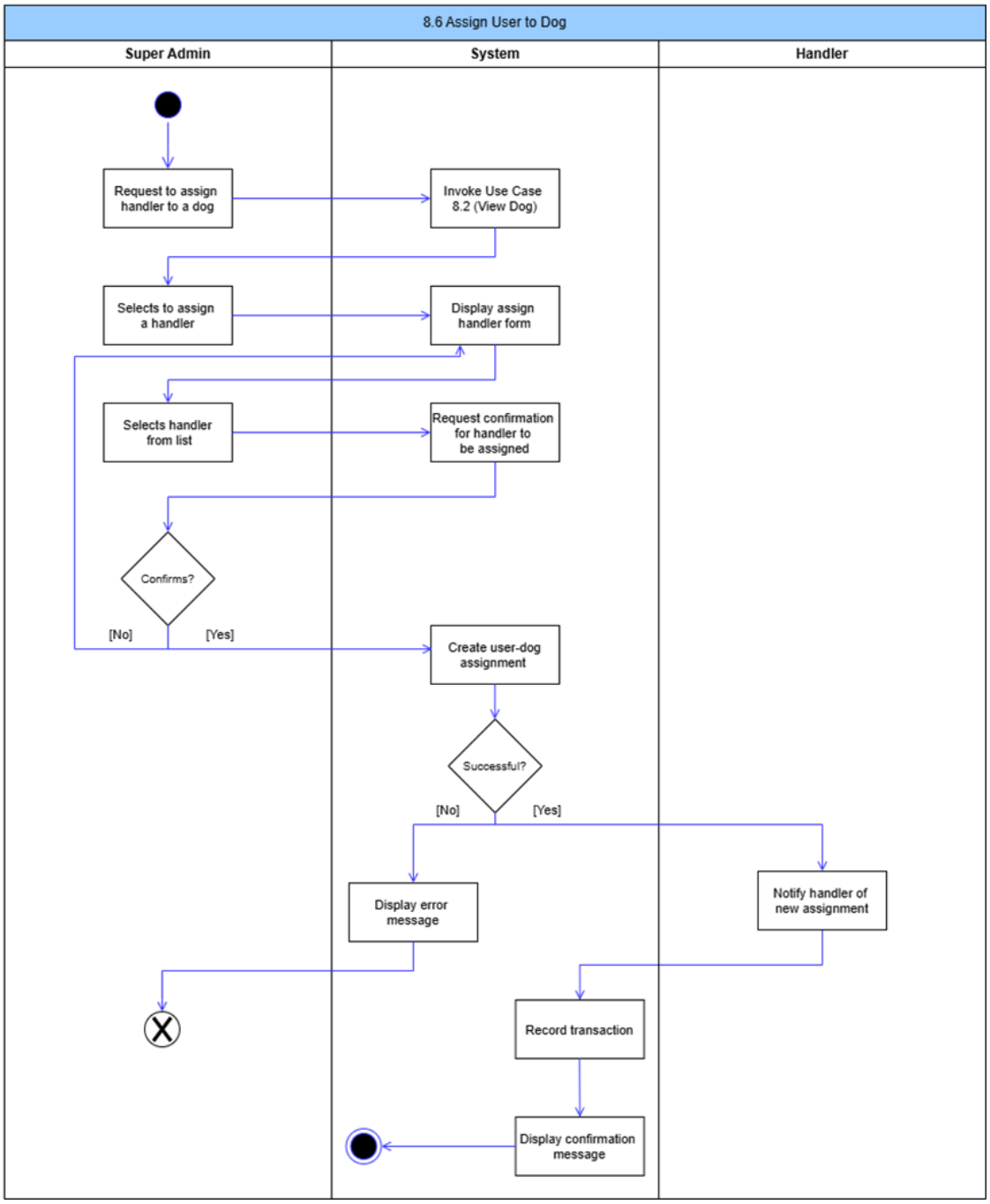


Figure 185 8.6 Assign User to Dog - Activity Diagram

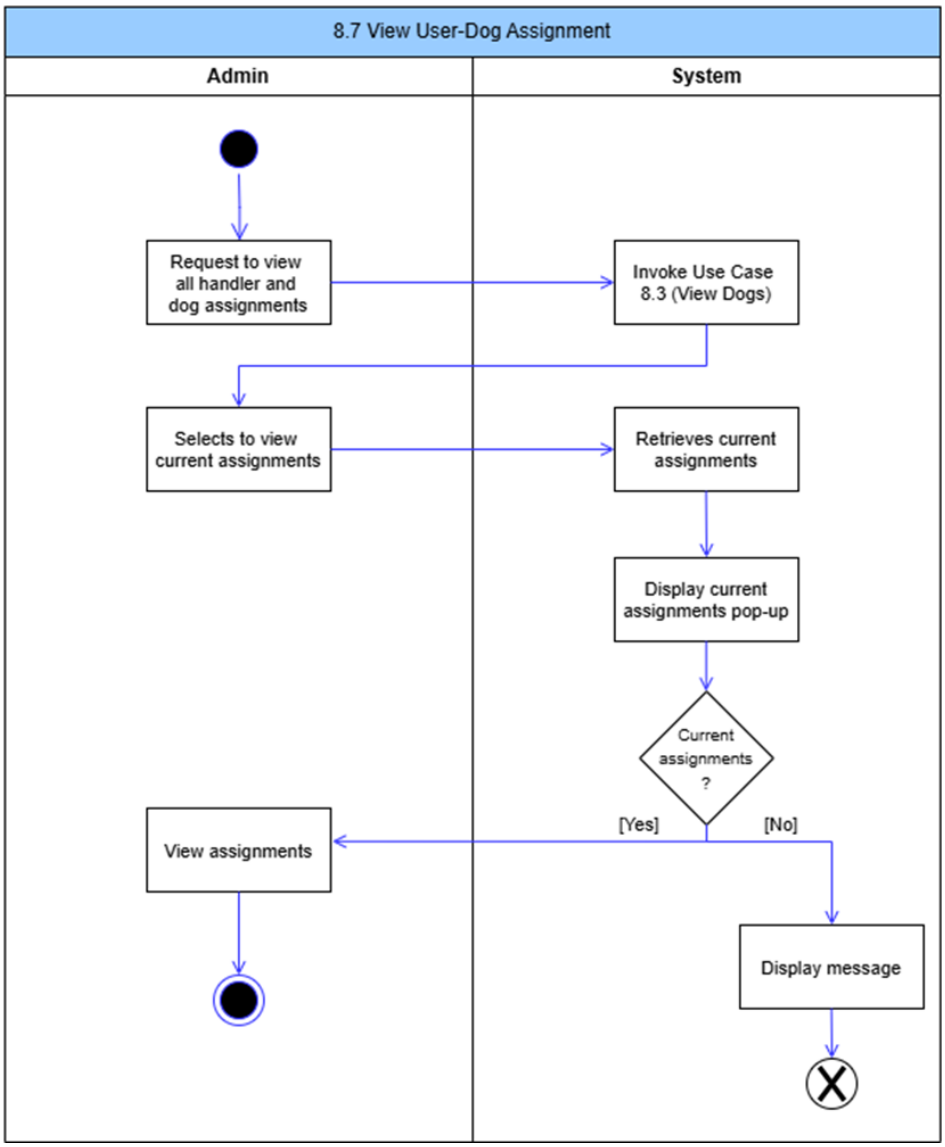


Figure 186 8.7 View User-Dog Assignment - Activity Diagram

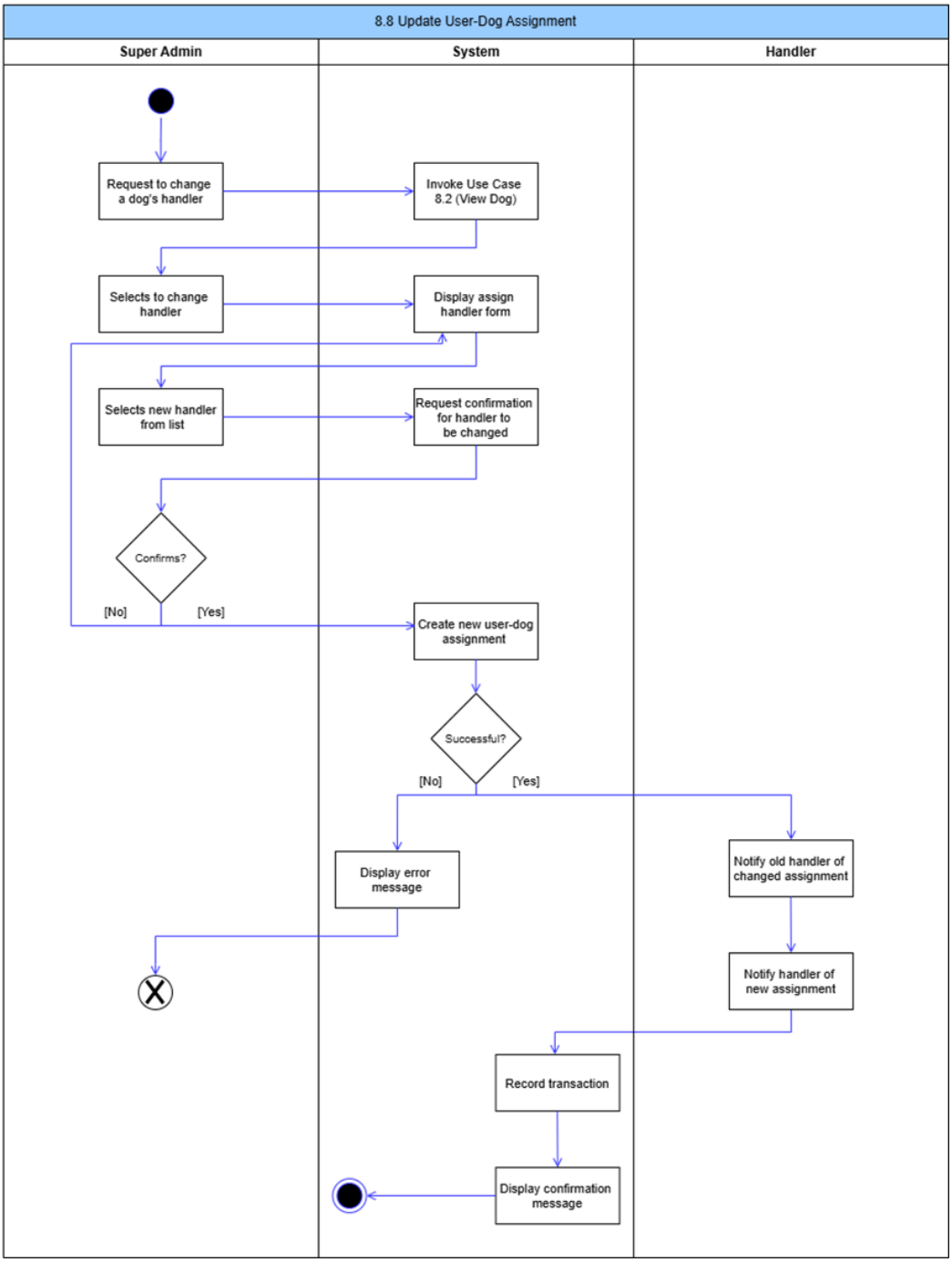


Figure 187 8.8 Update User-Dog Assigment - Activity Diagram

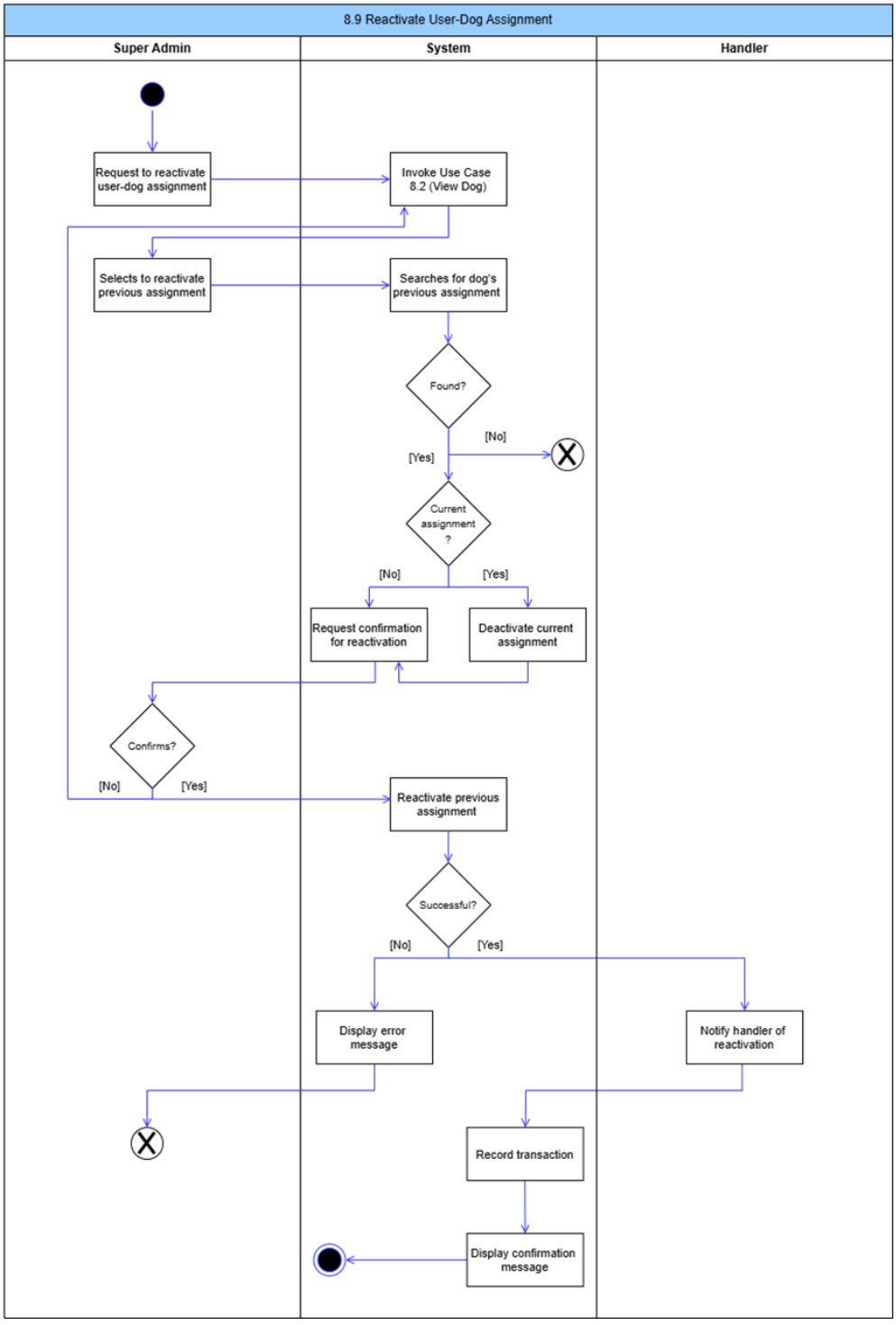


Figure 188 8.9 Reactivate User-Dog Assignment - Activity Diagram

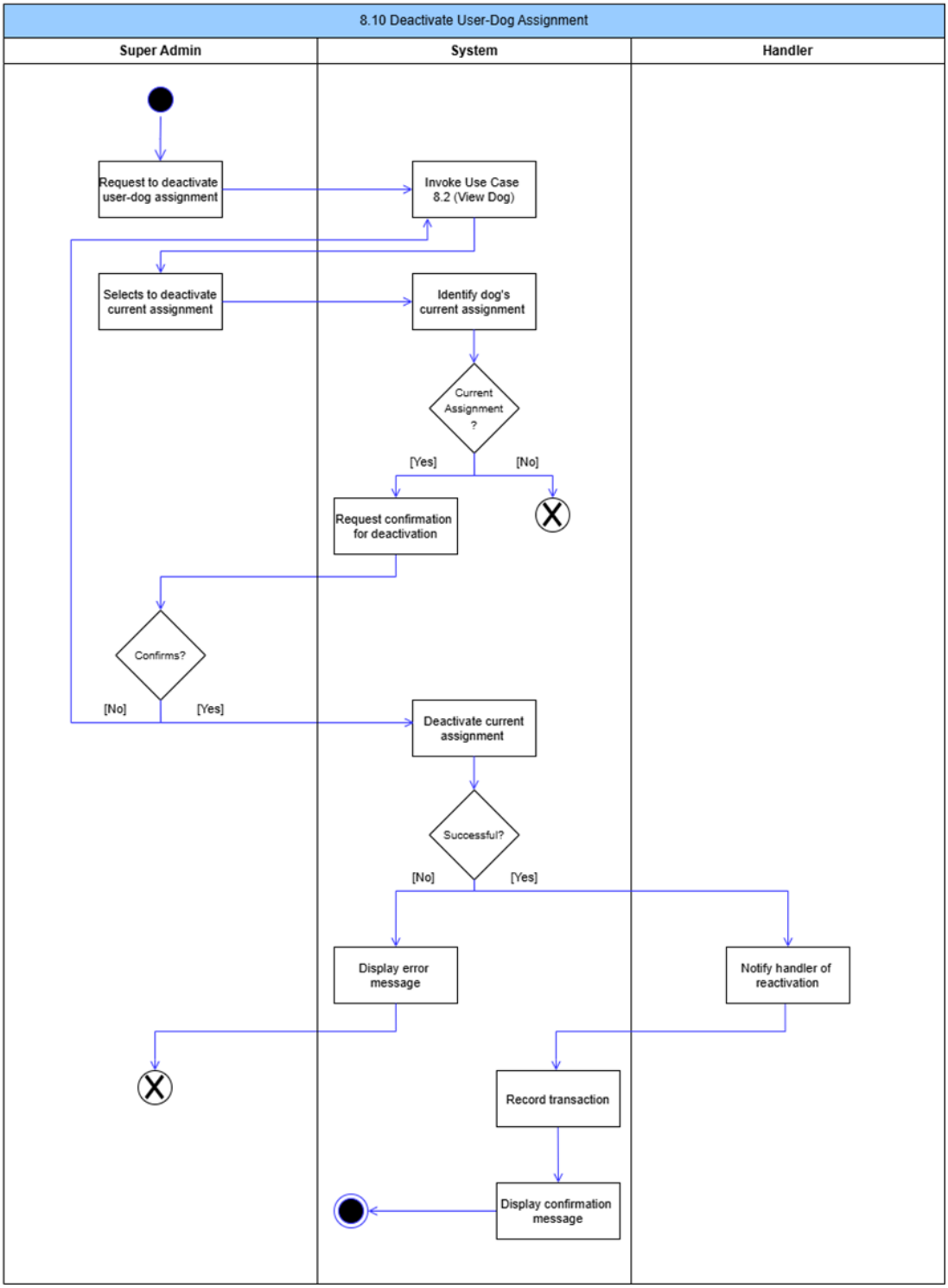


Figure 189 8.10 Deactivate User-Dog Assignment - Activity Diagram

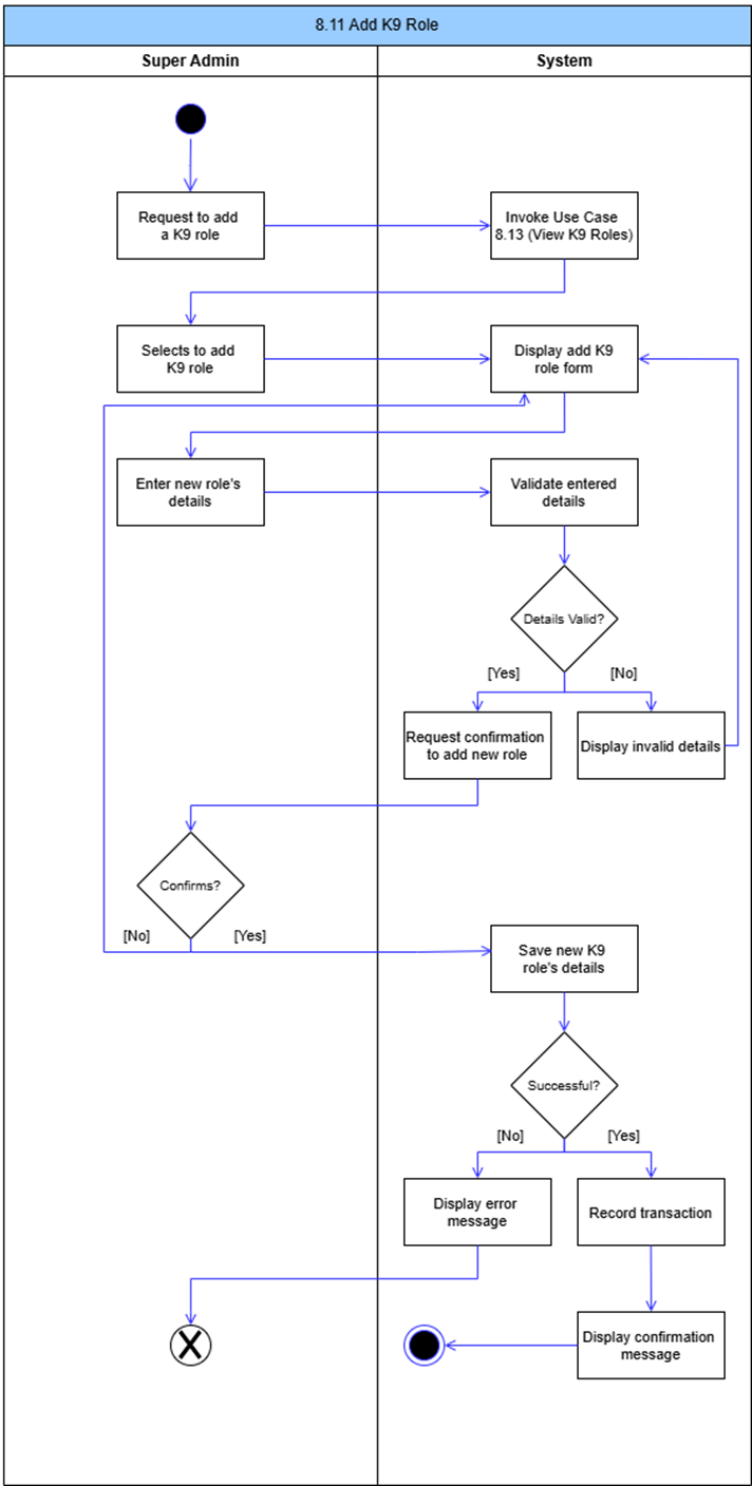


Figure 190 8.11 Add K9 Role - Activity Diagram

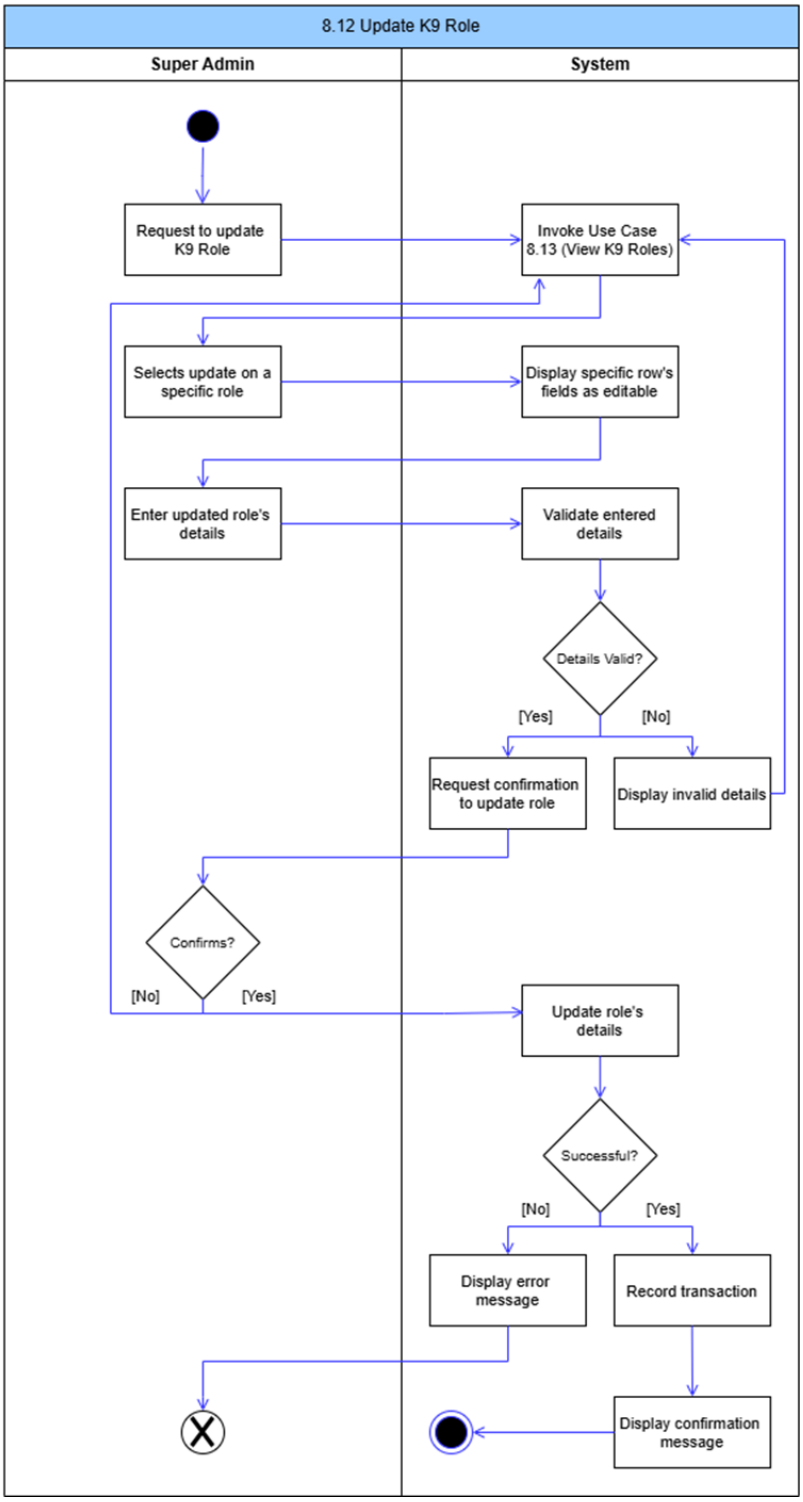


Figure 191 8.12 Update K9 Role - Activity Diagram

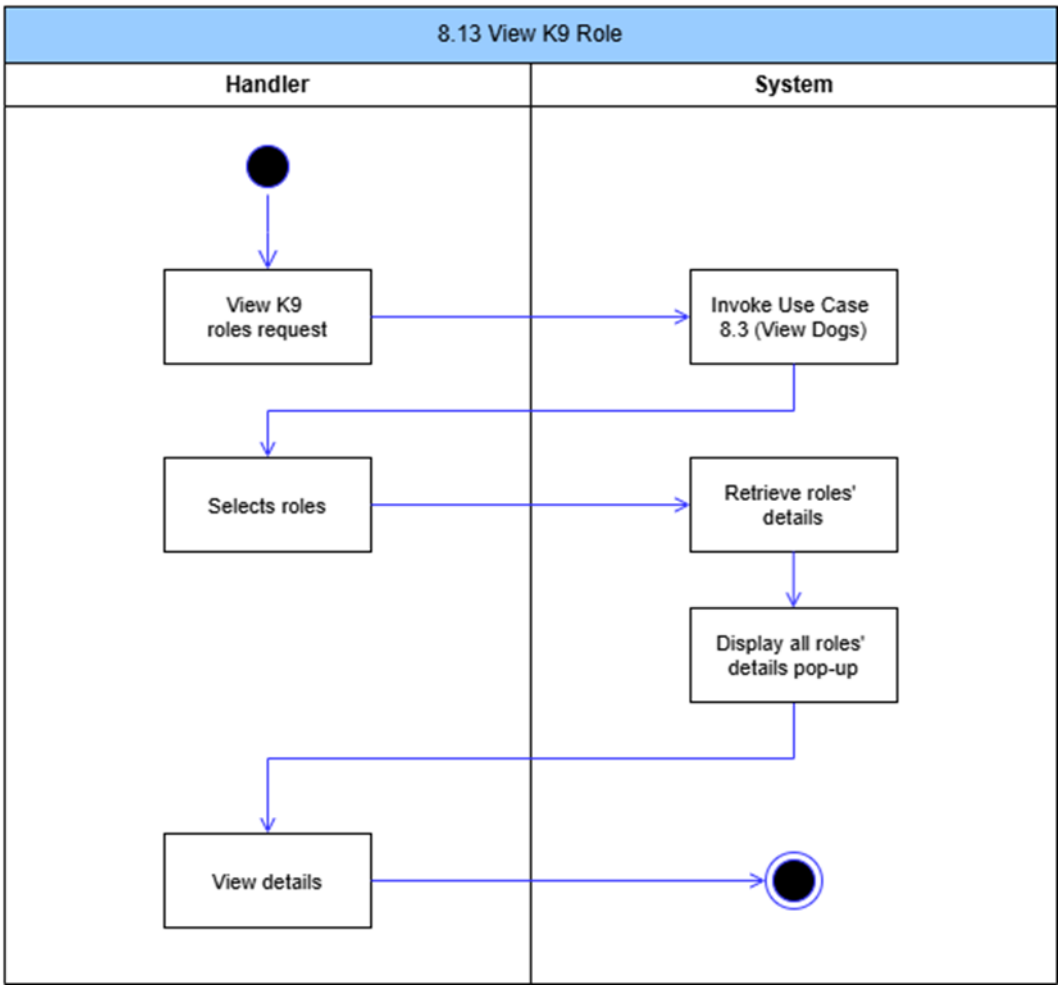


Figure 192 8.13 View K9 Role - Activity Diagram

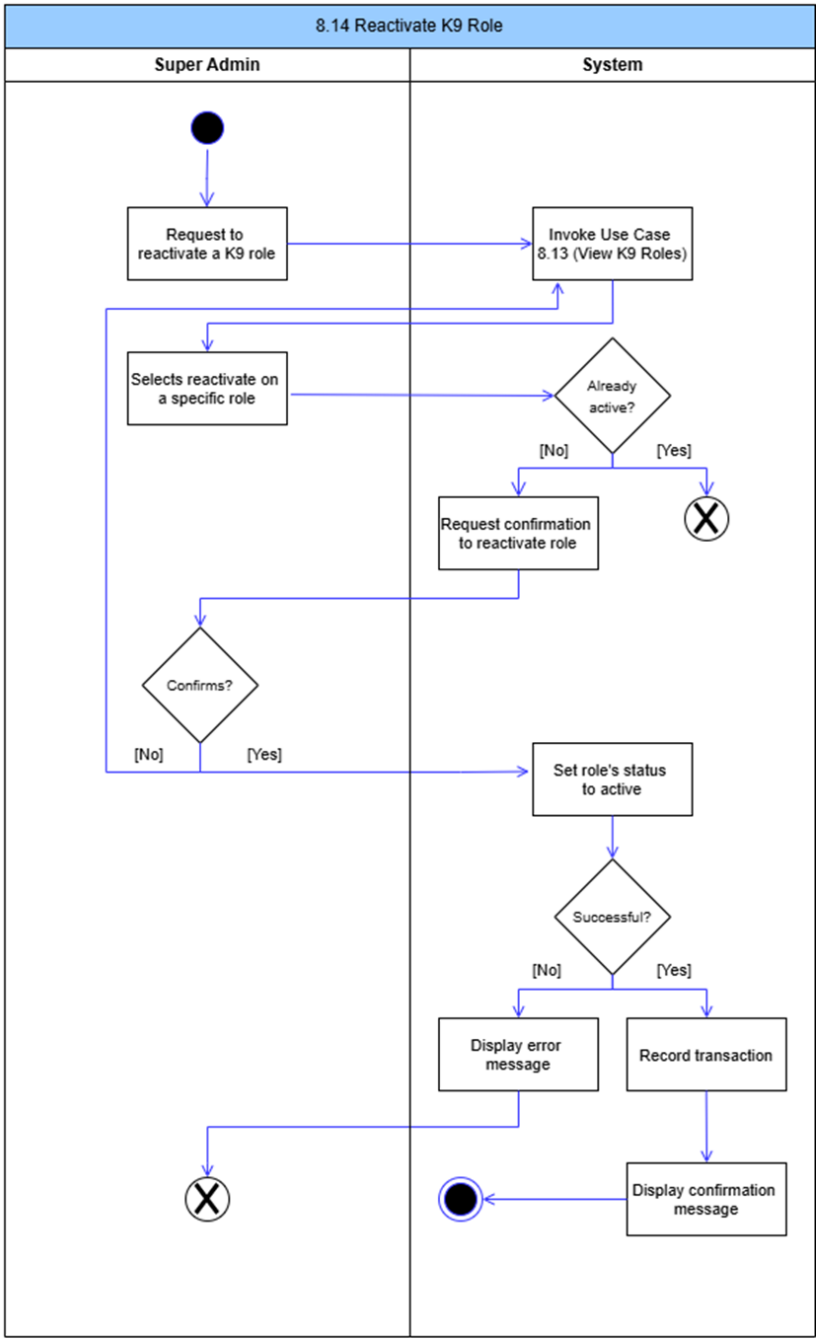


Figure 193 8.14 Reactivate K9 Role - Activity Diagram

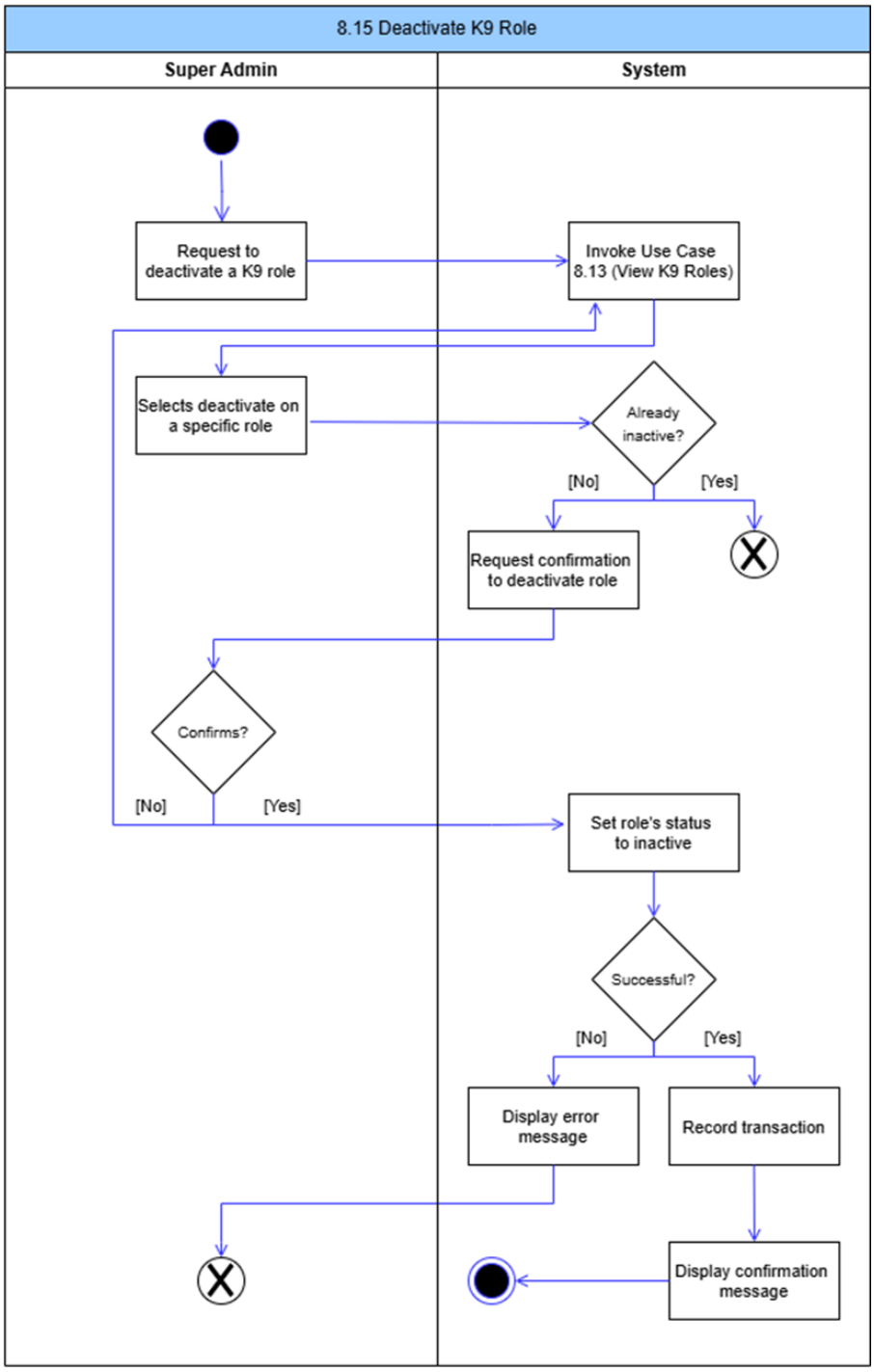


Figure 194 8.15 Deactivate K9 Role - Activity Diagram

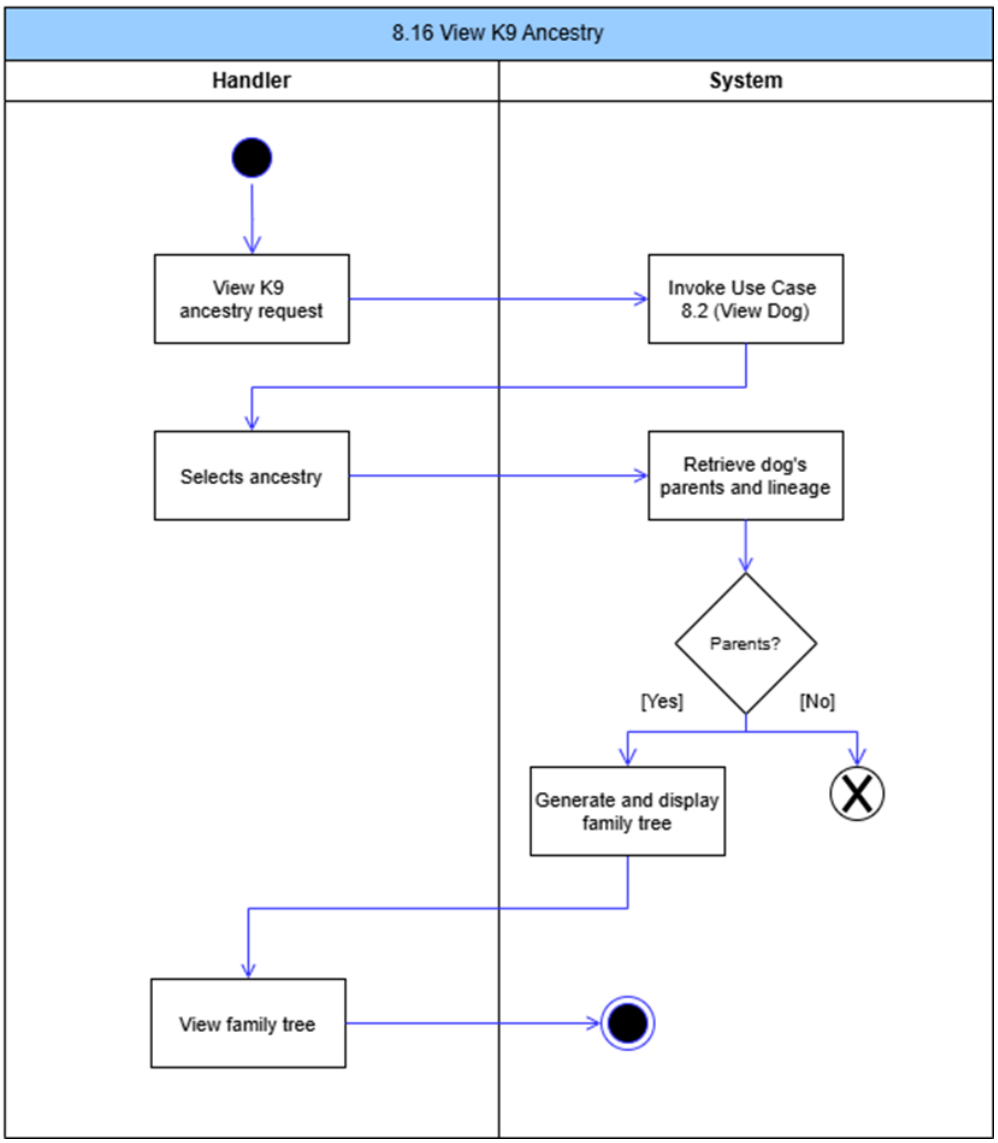


Figure 195 8.16 View K9 Ancestry - Activity Diagram

2.10.9 Rhino Subsystem Activity Diagrams

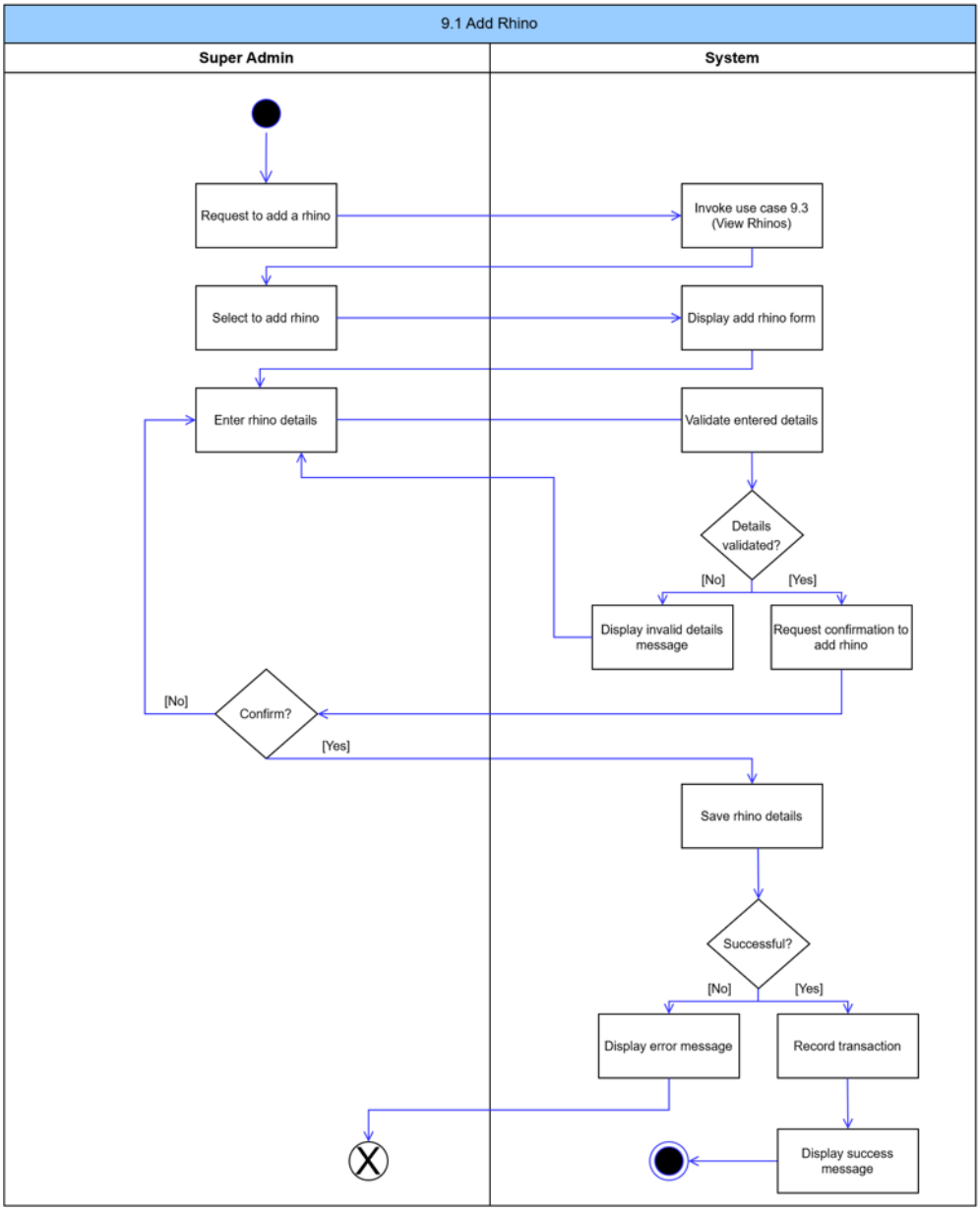


Figure 196 9.1 Add Rhino - Activity Diagram

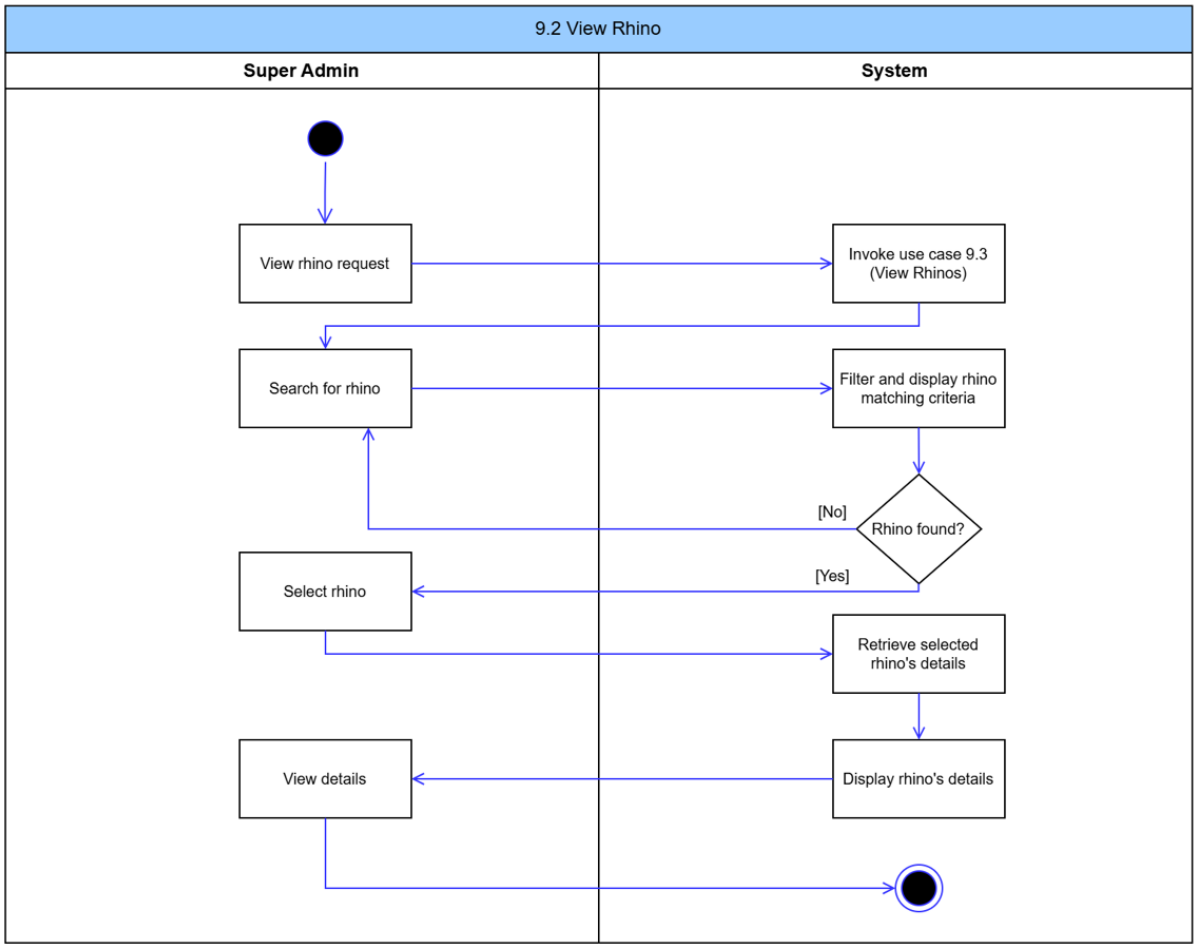


Figure 197 9.2 View Rhino - Activity Diagram

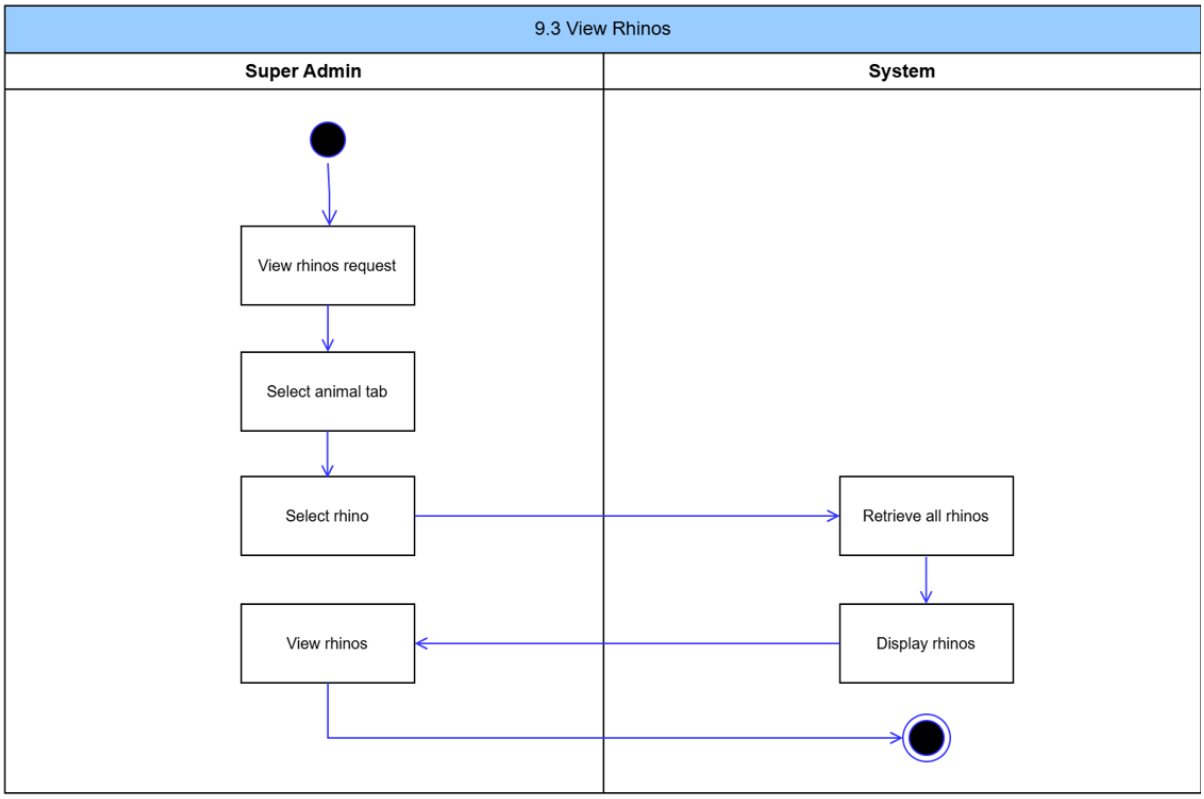


Figure 198 9.3 View Rhinos - Activity Diagram

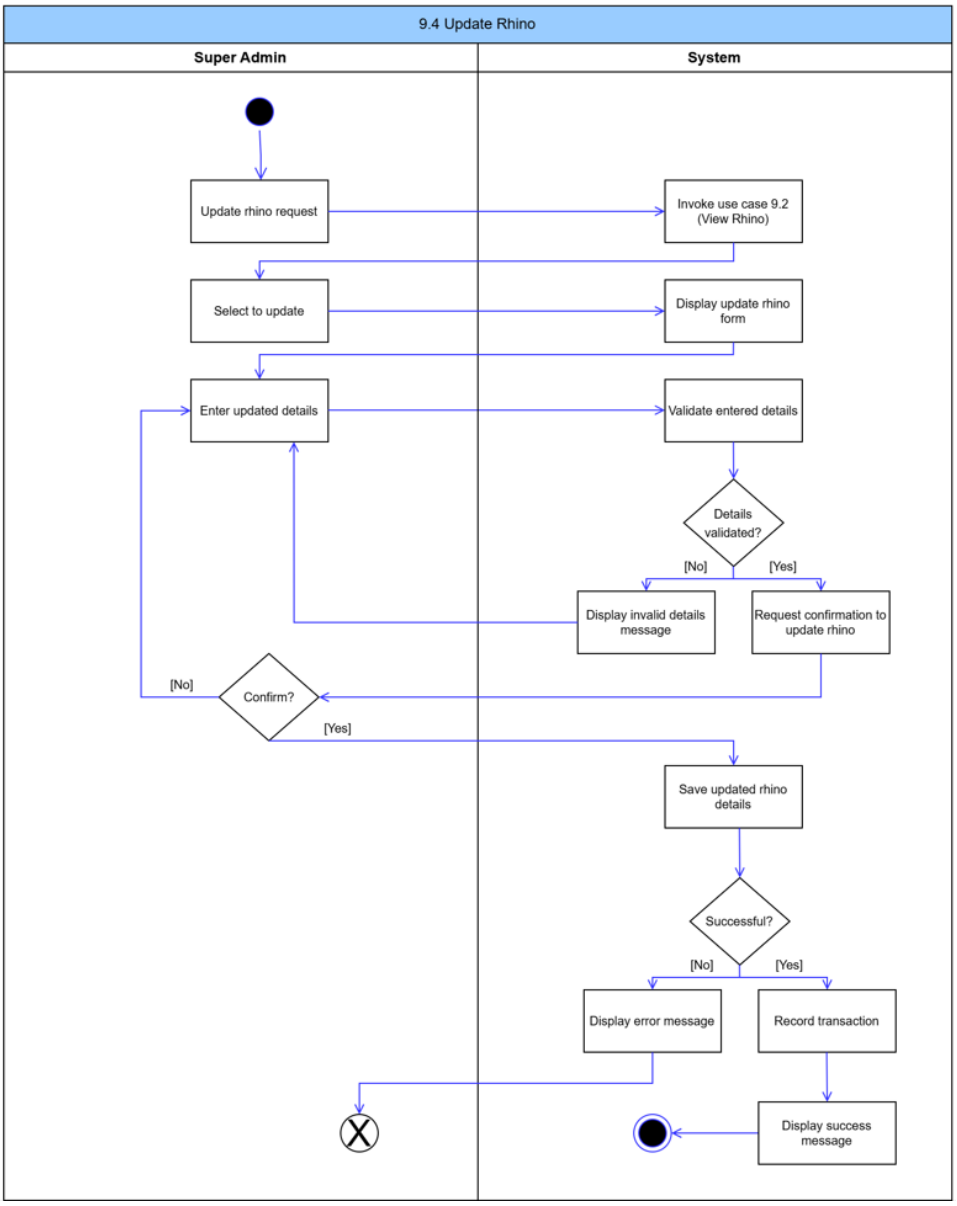


Figure 199 9.4 Update Rhino - Activity Diagram

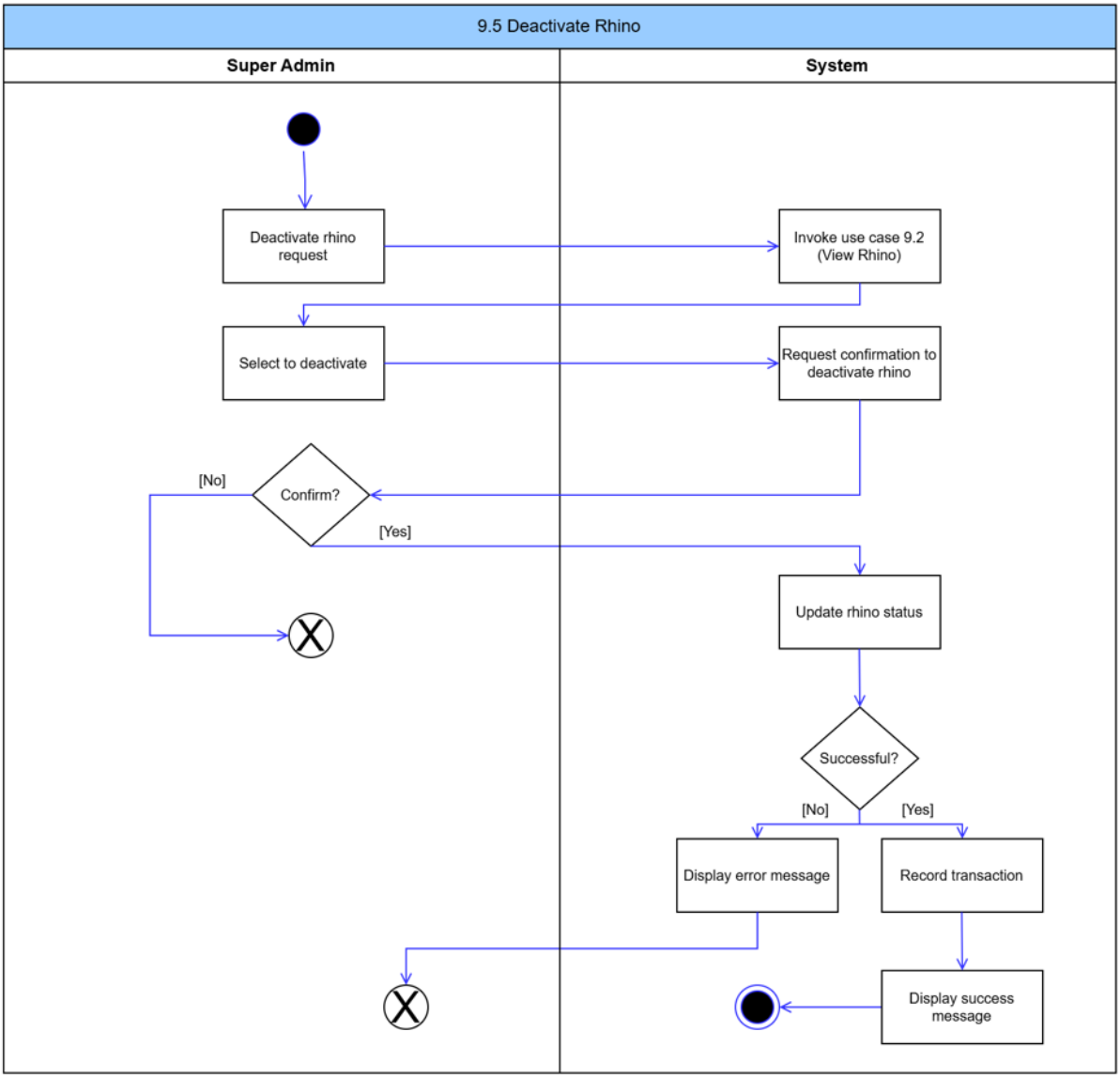


Figure 200 9.5 Deactivate Rhino - Activity Diagram

2.10.10 Elephant Subsystem Activity Diagrams

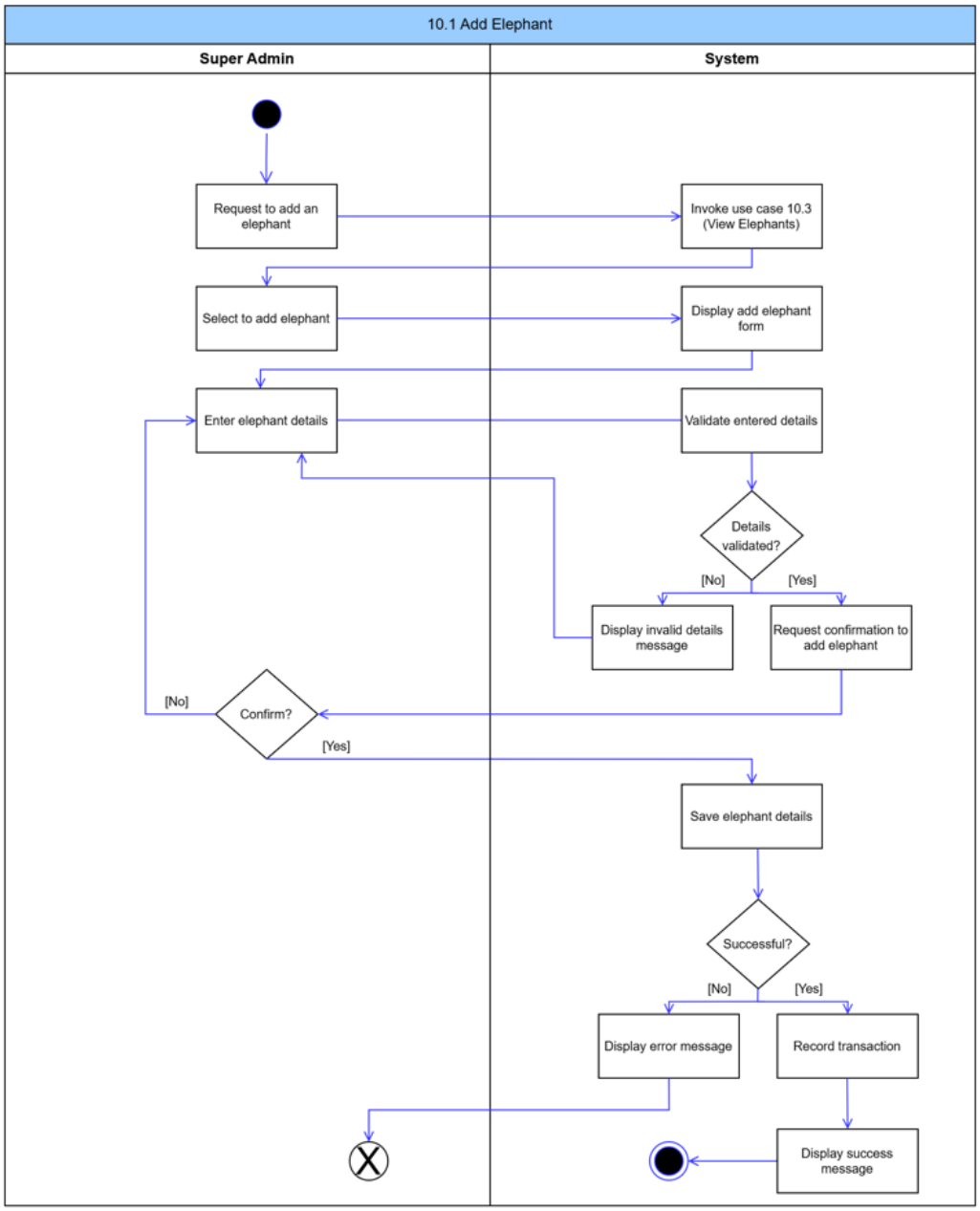


Figure 201 10.1 Add Elephant - Activity Diagram

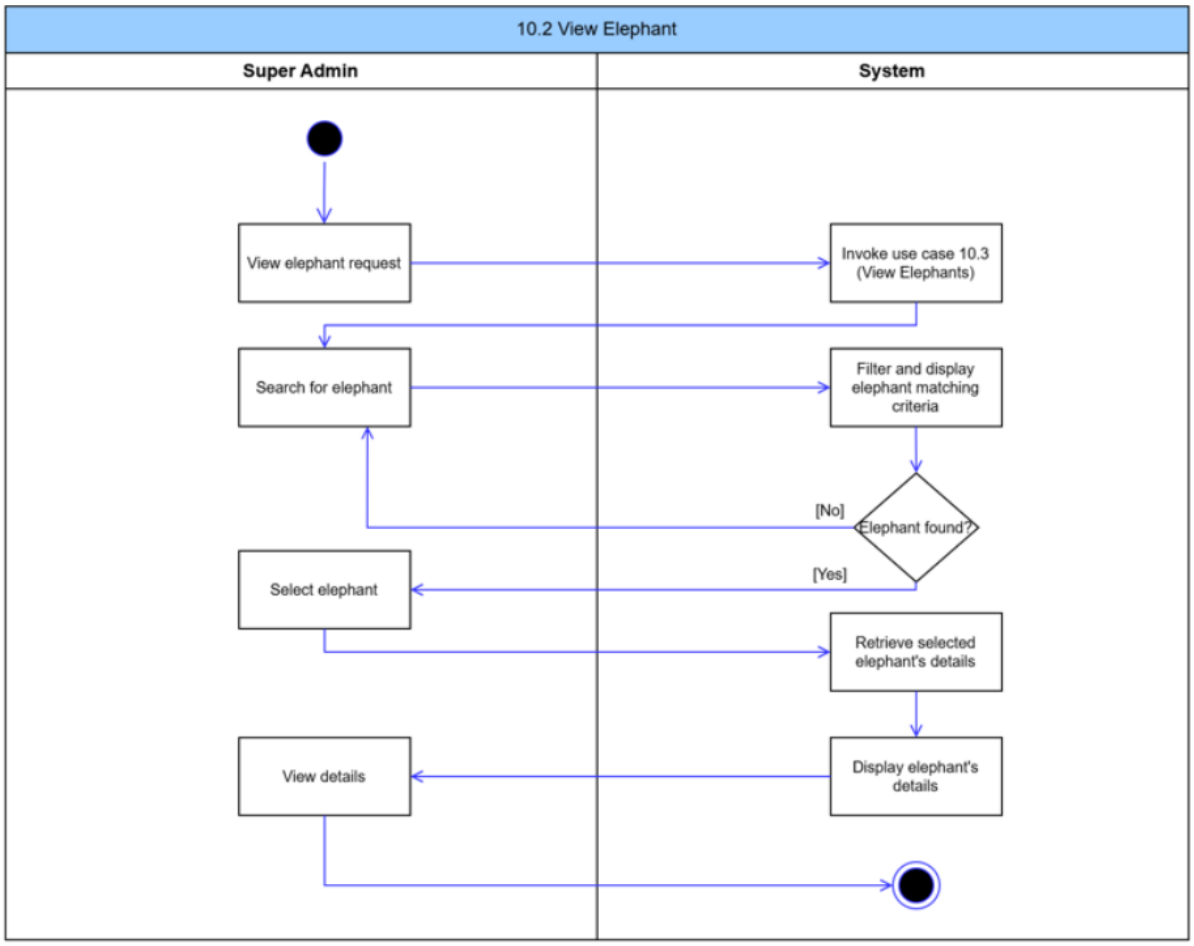


Figure 202 10.2 View Elephant - Activity Diagram

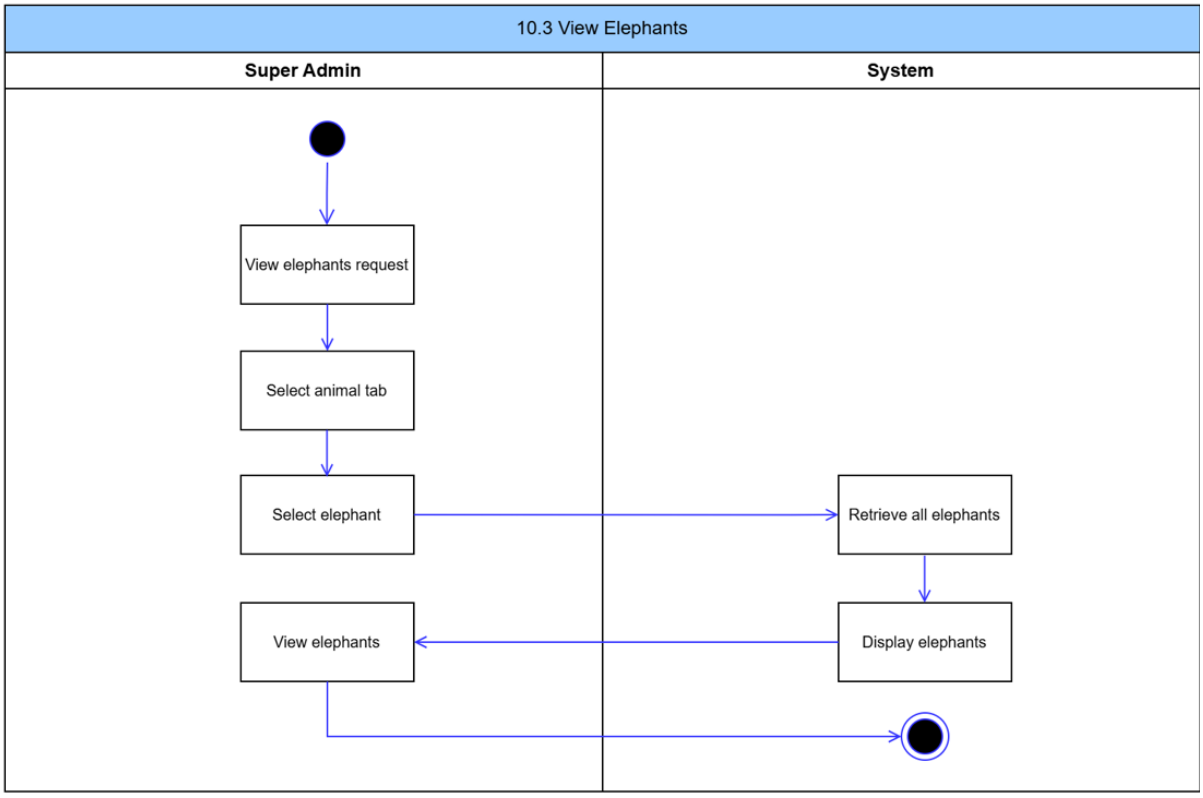


Figure 203 10.3 View Elephants - Activity Diagram

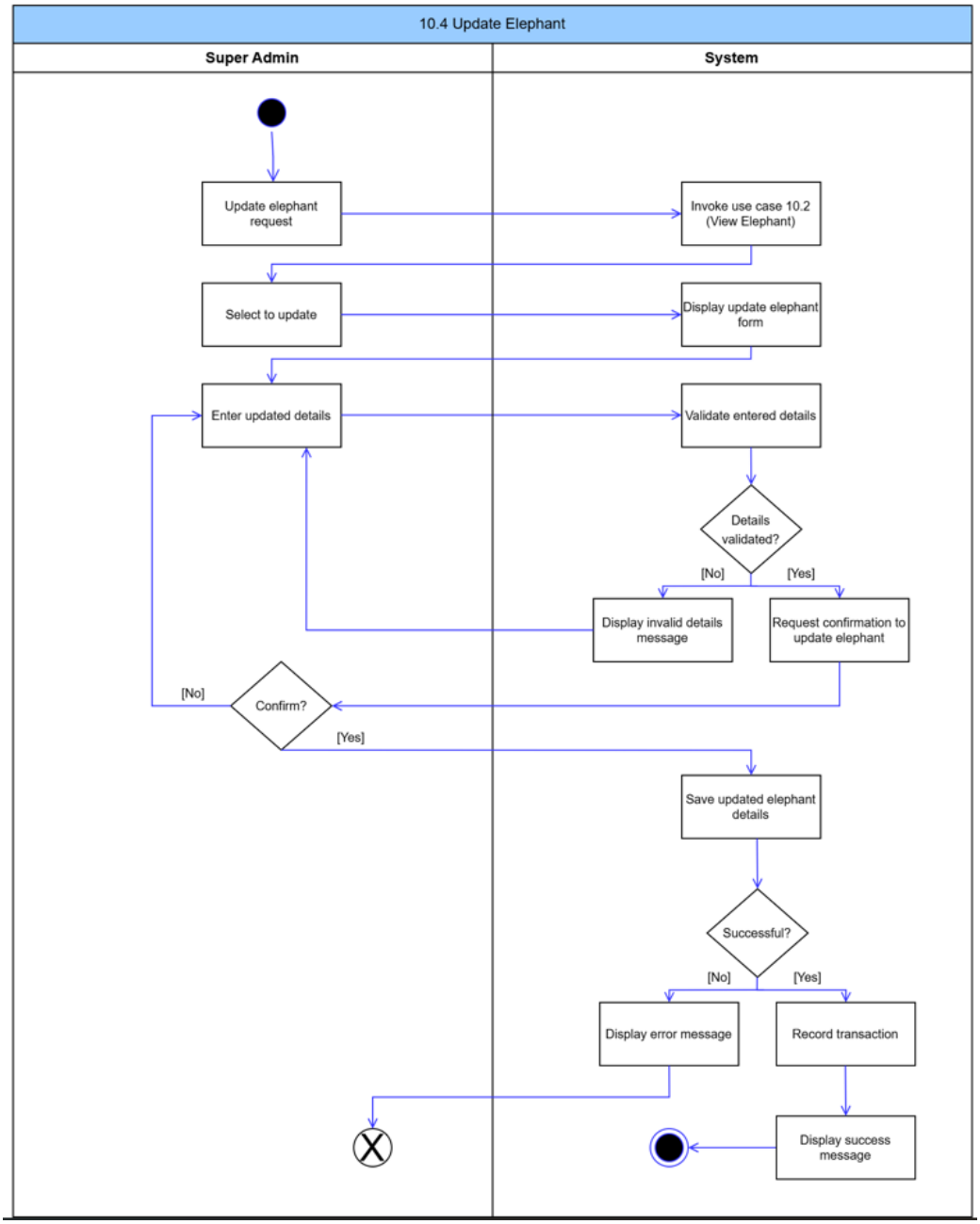


Figure 204 10.4 Update Elephant - Activity Diagram

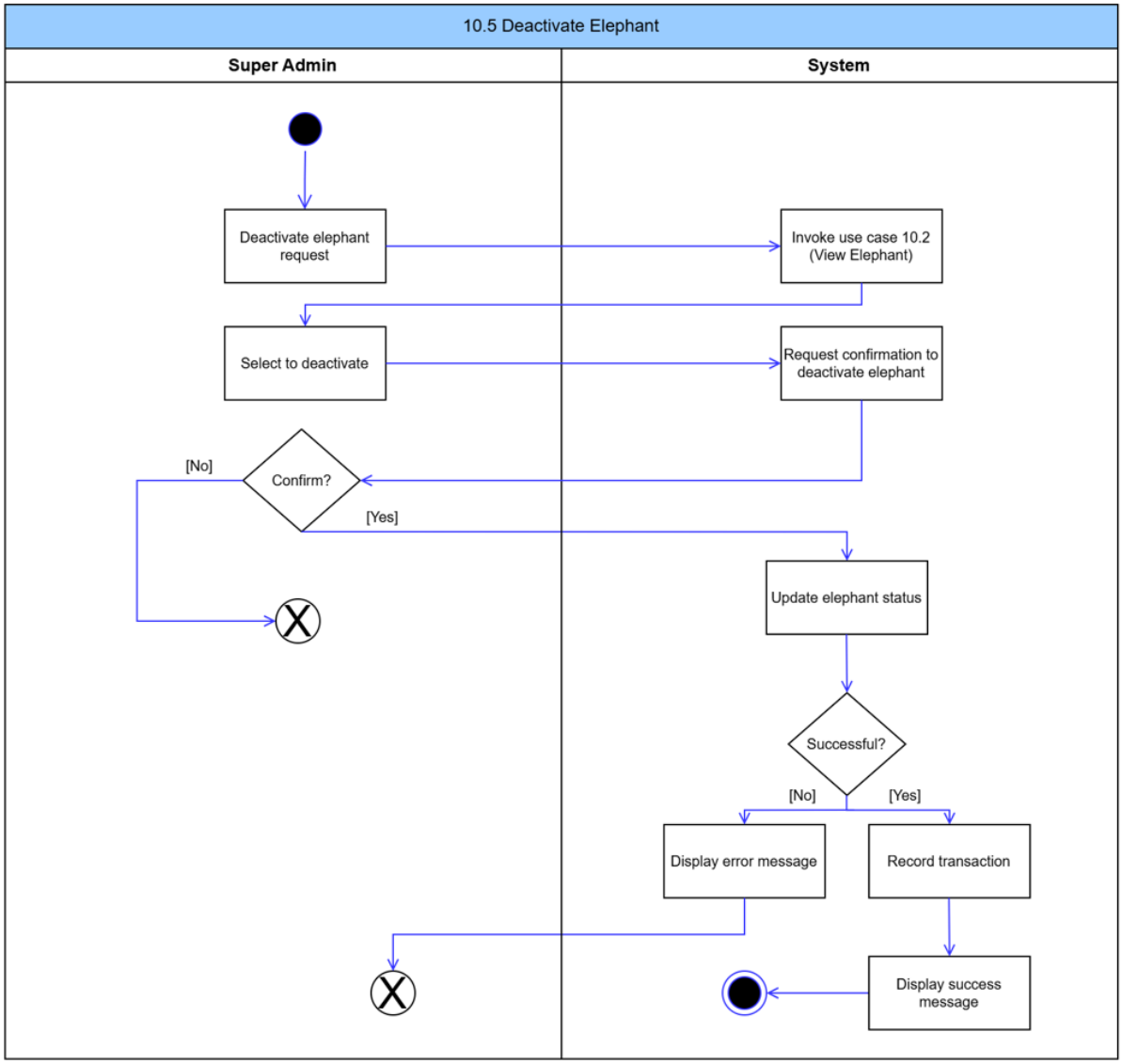


Figure 205 10.5 Deactivate Elephant - Activity Diagram

2.10.11 Message Subsystem Activity Diagrams

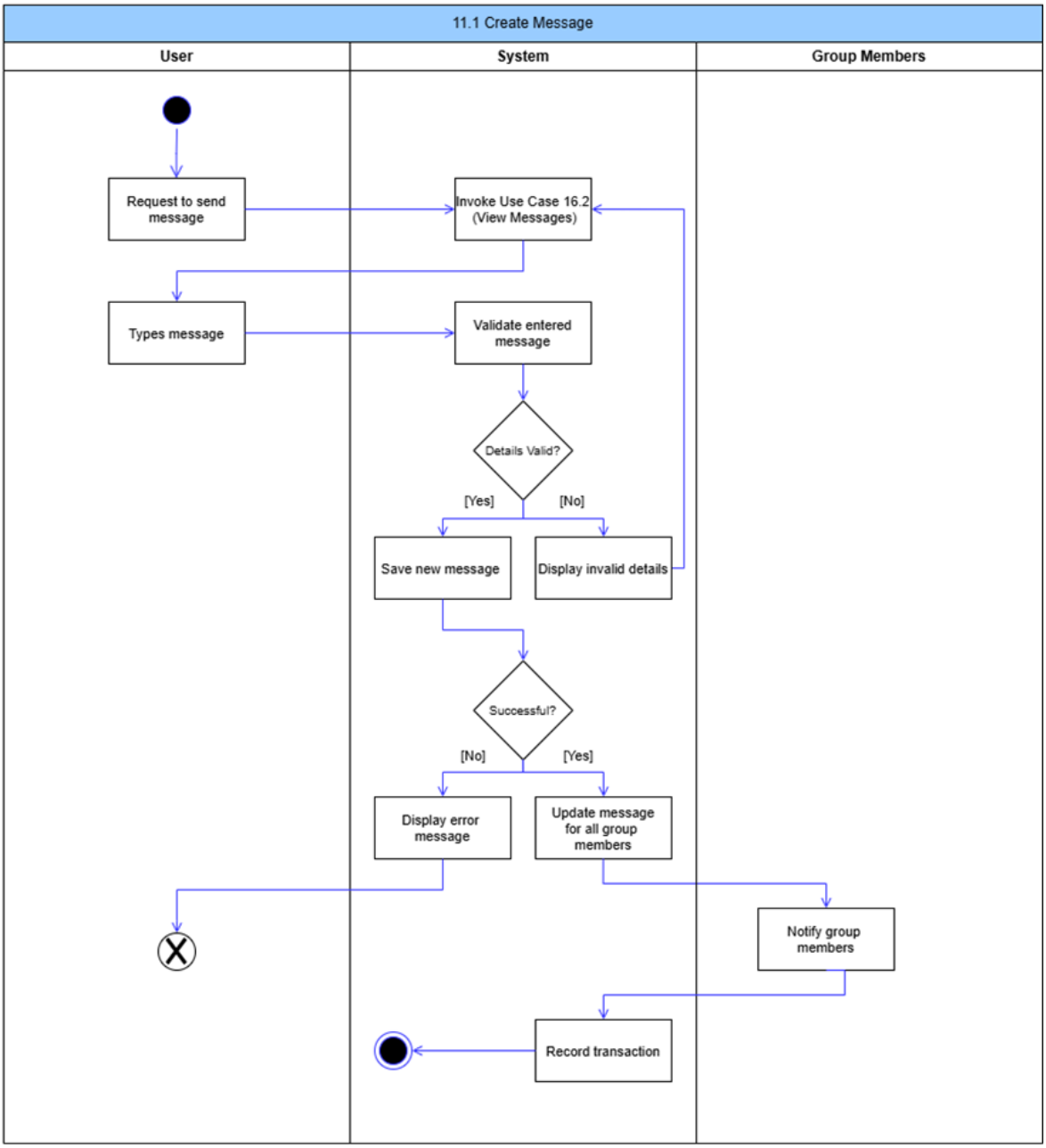


Figure 206 11.1 Create Message - Activity Diagram

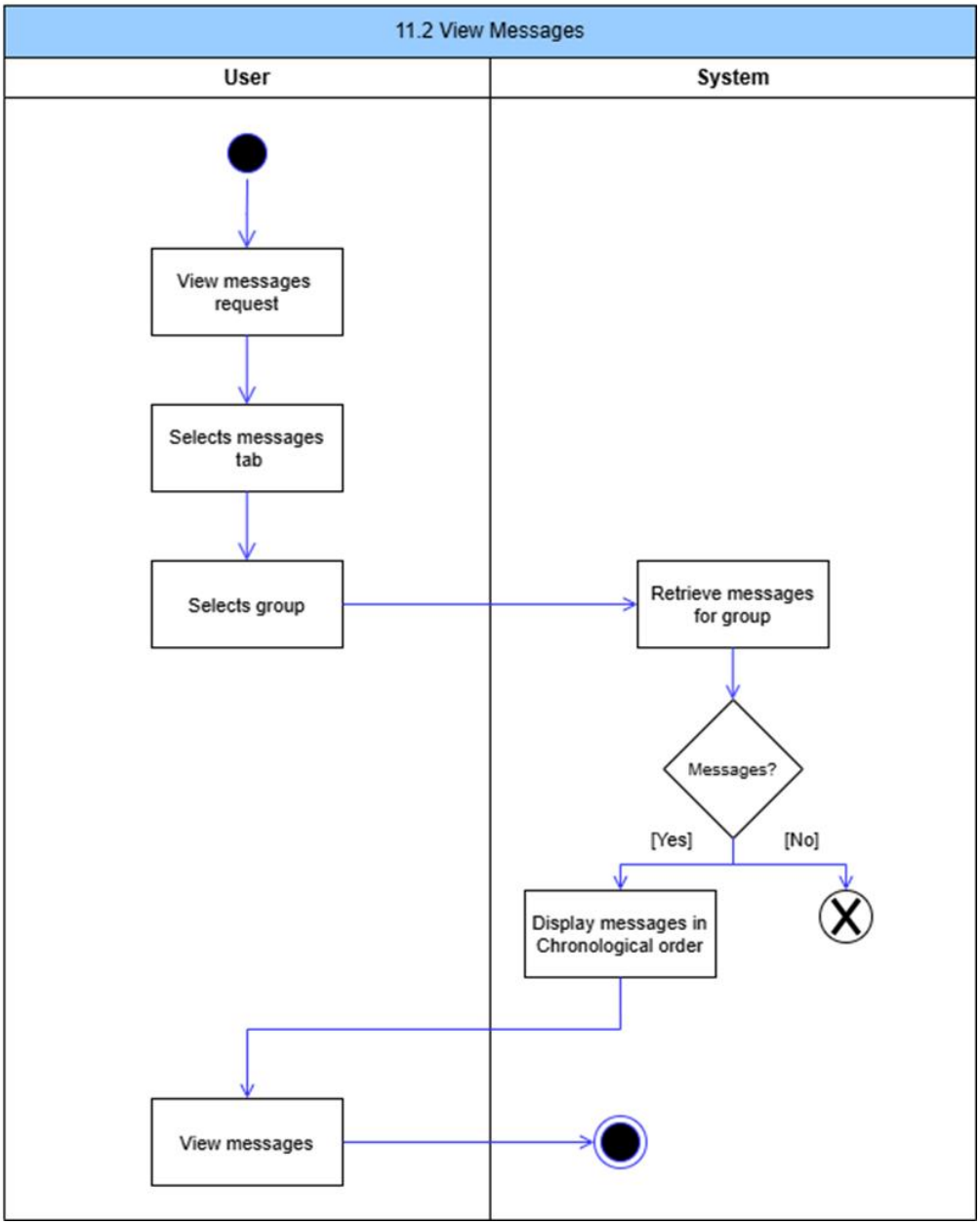


Figure 207 11.2 View Messages - Activity Diagram

2.10.12 Report Subsystem Activity Diagrams

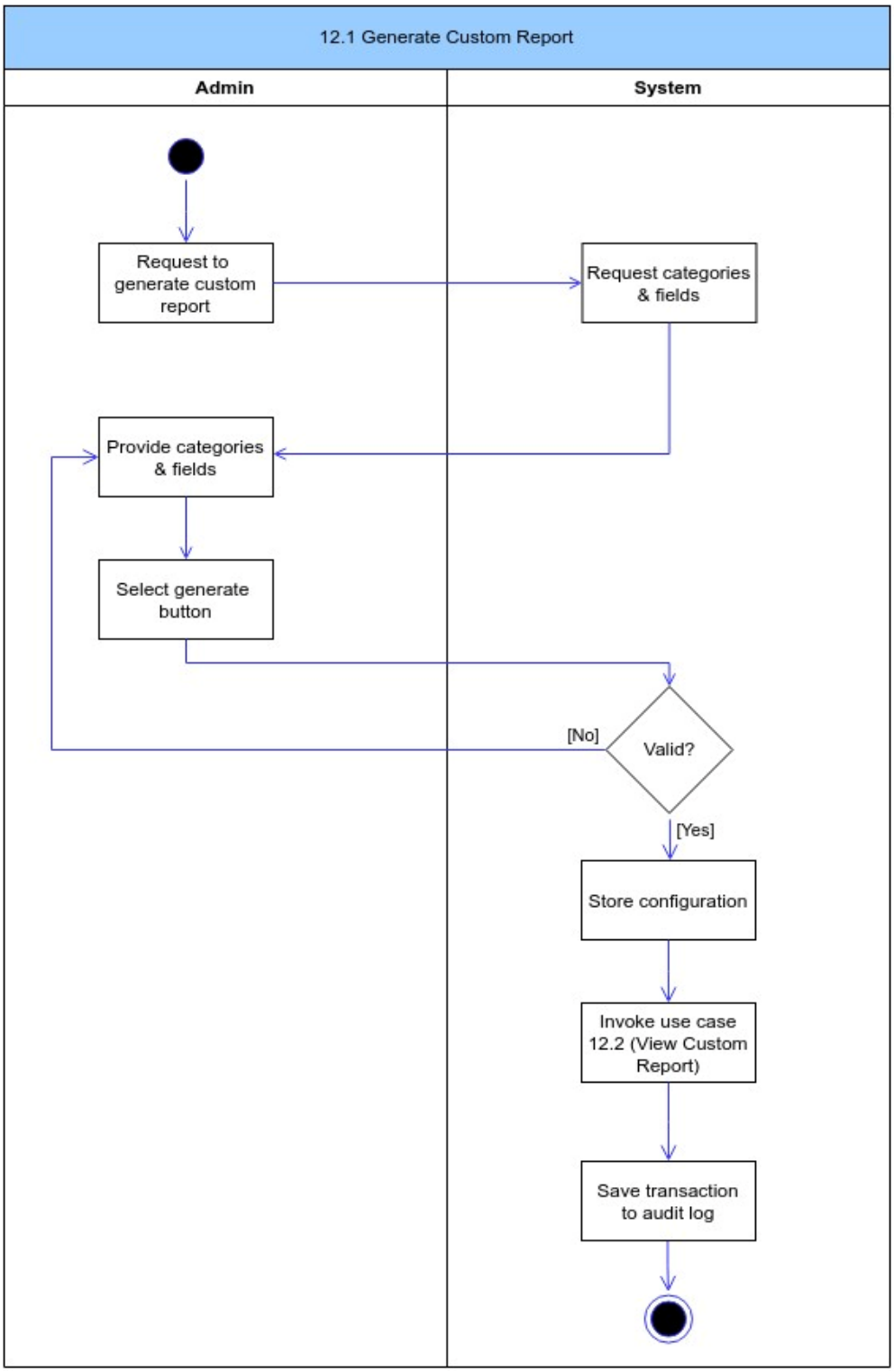


Figure 208 12.1 Generate Custom Report - Activity Diagram

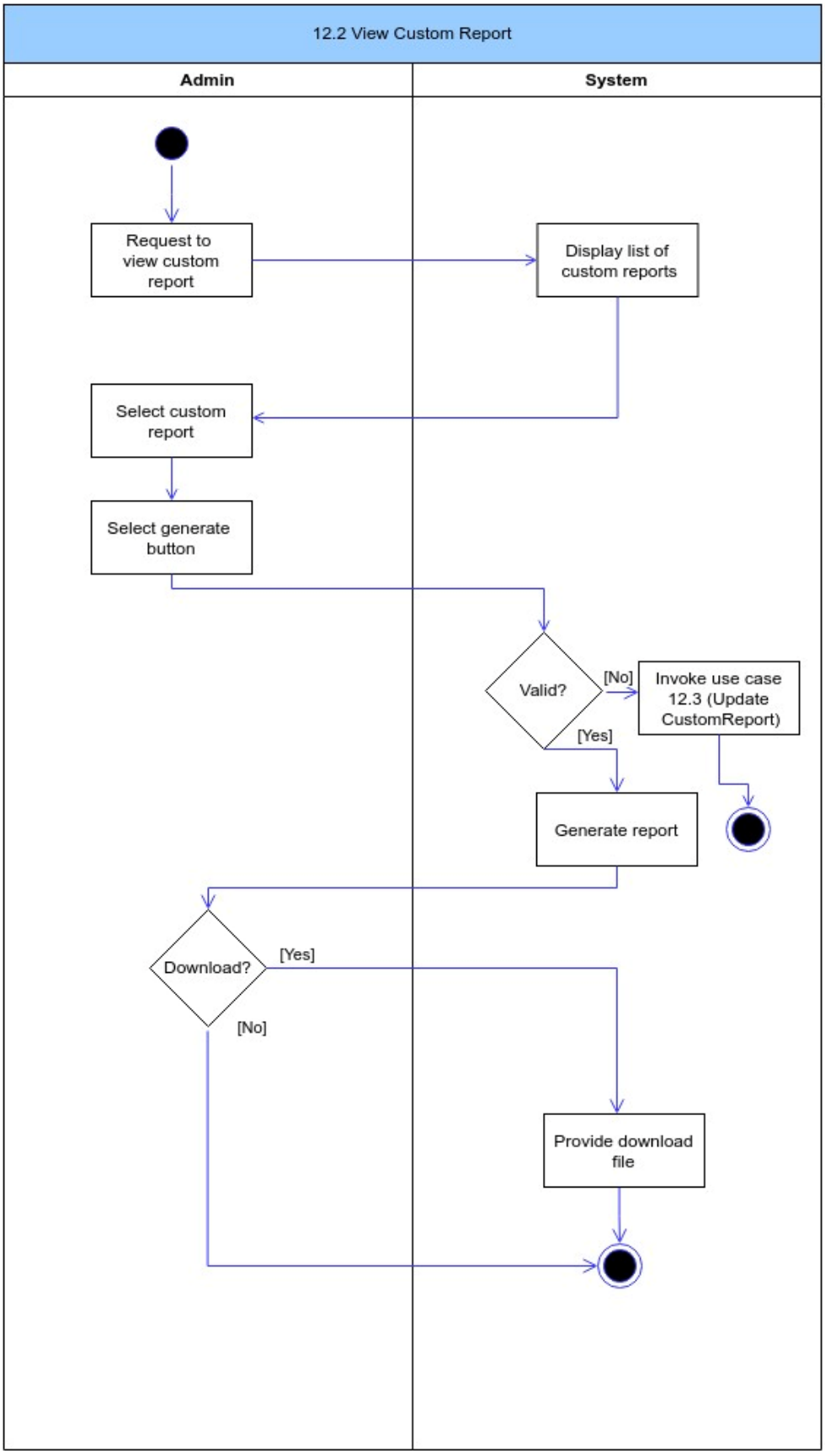


Figure 209 12.2 View Custom Report - Activity Diagram

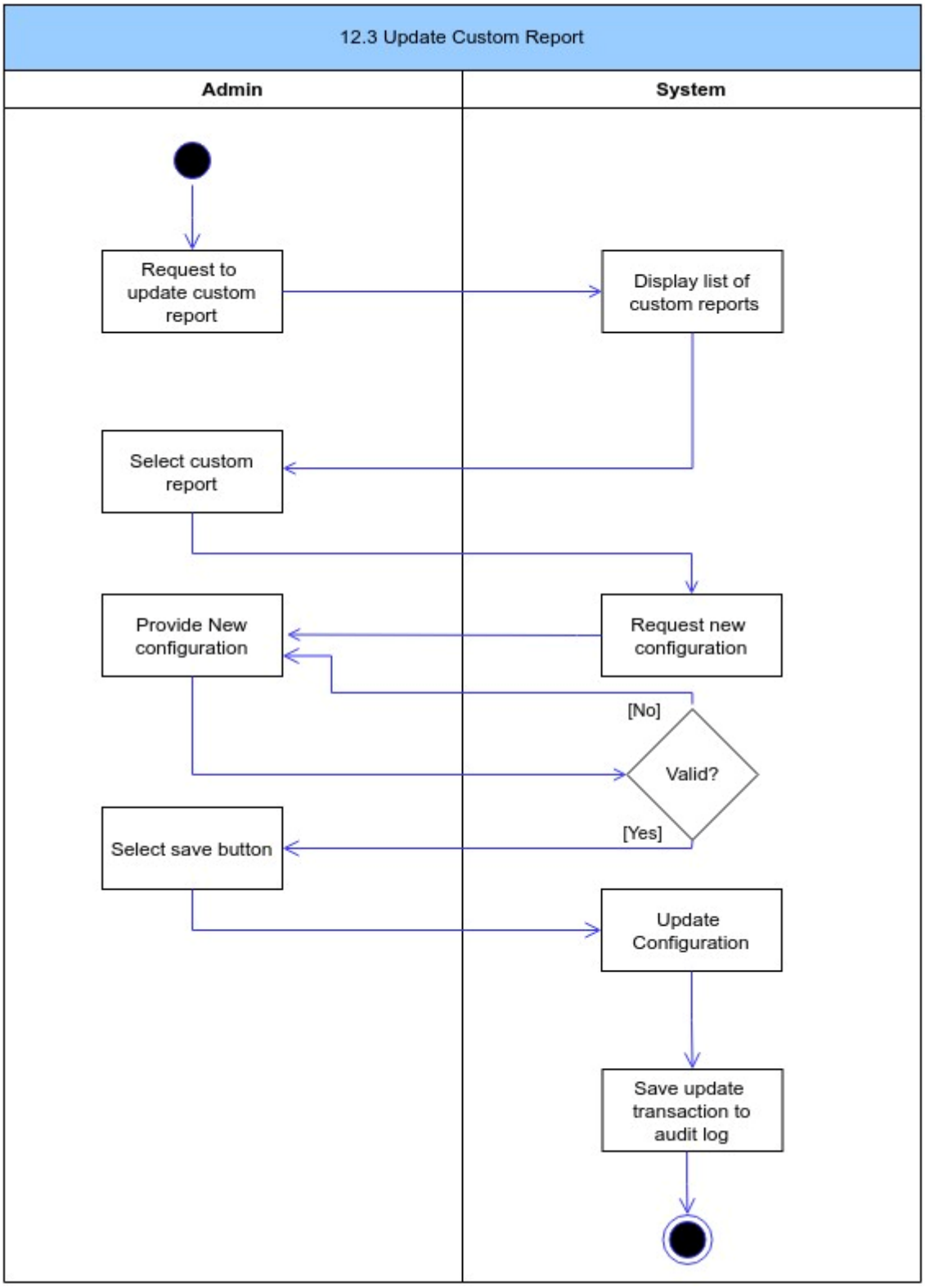


Figure 210 12.3 Update Custom Report - Activity Diagram

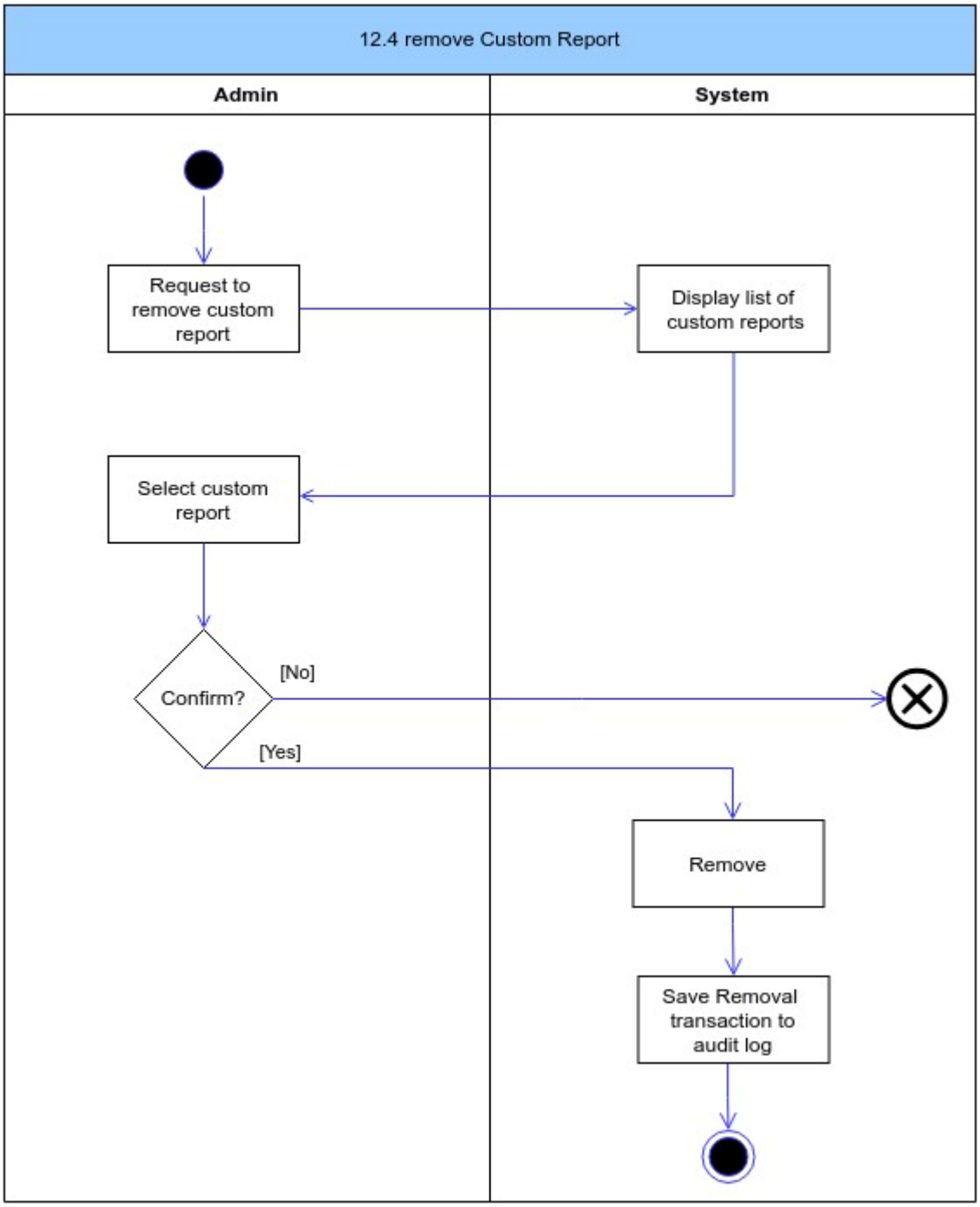


Figure 211 12.4 Remove Custom Report - Activity Diagram

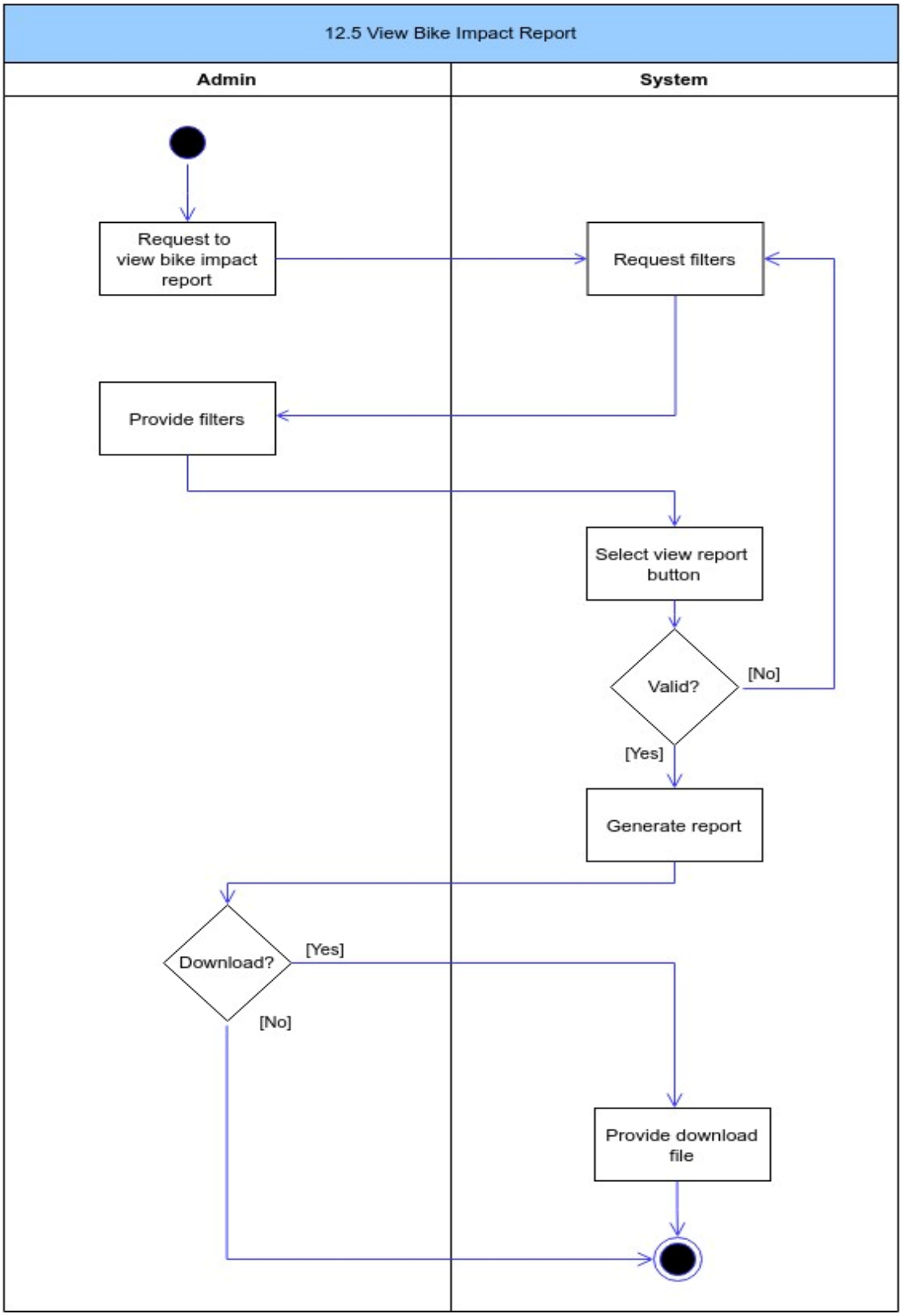


Figure 212 12.5 View Bike Impact Report - Activity Diagram

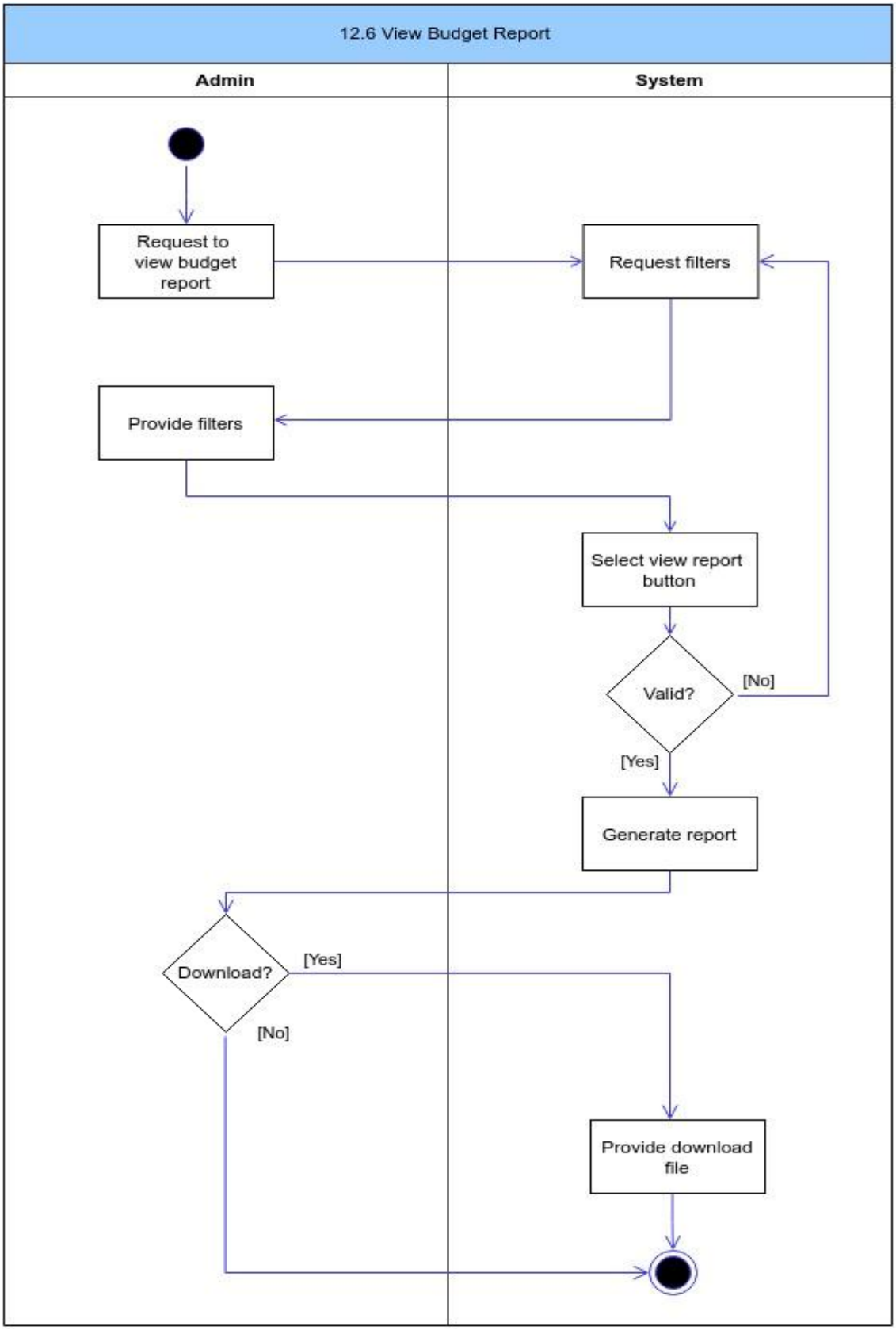


Figure 213 12.6 View Budget Report - Activity Diagram

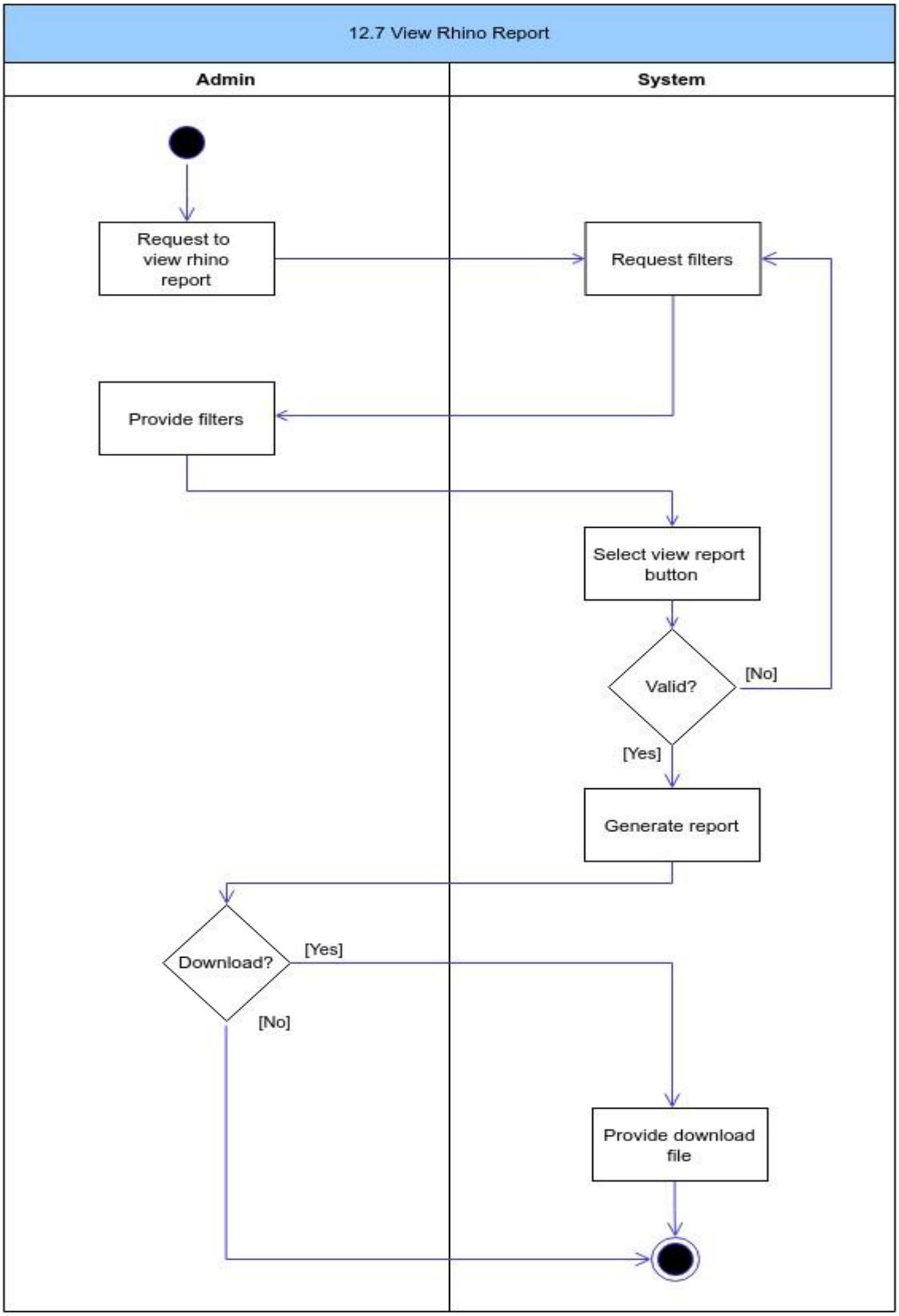


Figure 214 View Rhino Report - Activity Diagram

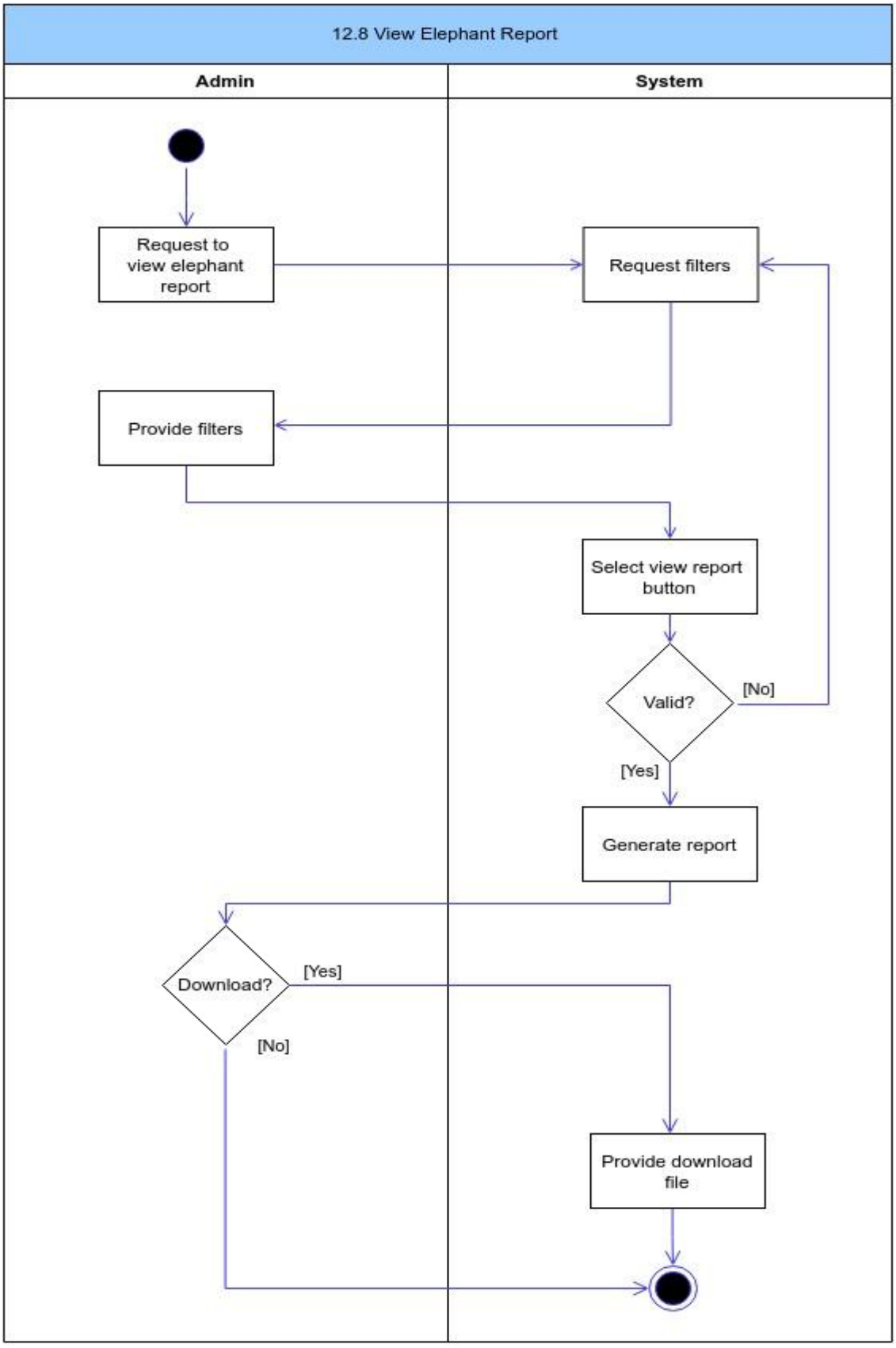


Figure 215 12.8 View Elephant Report - Activity Diagram

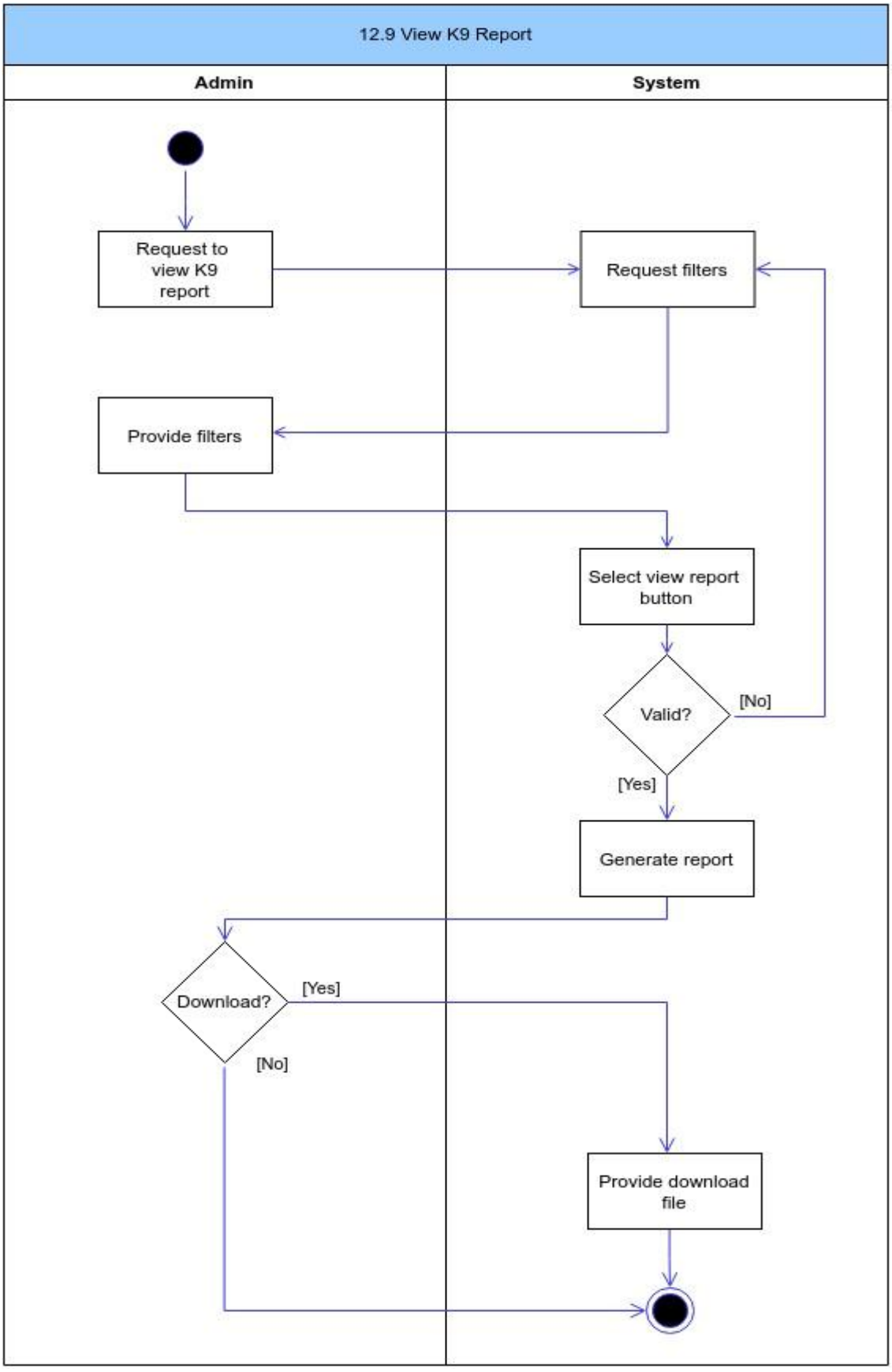


Figure 216 12.9 View K9 Report - Activity Diagram

2.10.13 Machine Learning Subsystem Activity Diagrams

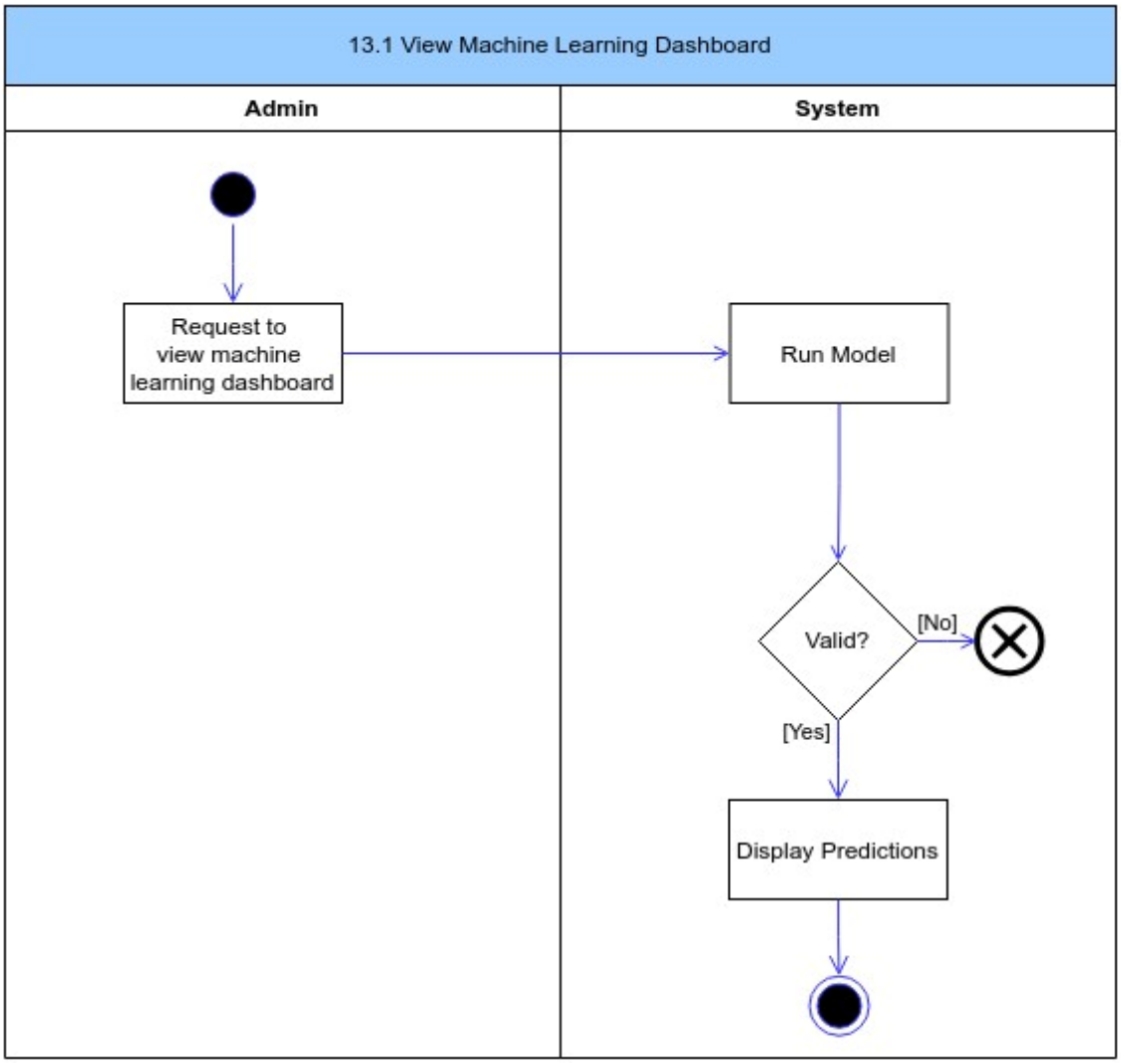


Figure 217 13.1 View Machine Learning Dashboard - Activity Diagram

2.10.14 System Subsystem Activity Diagrams

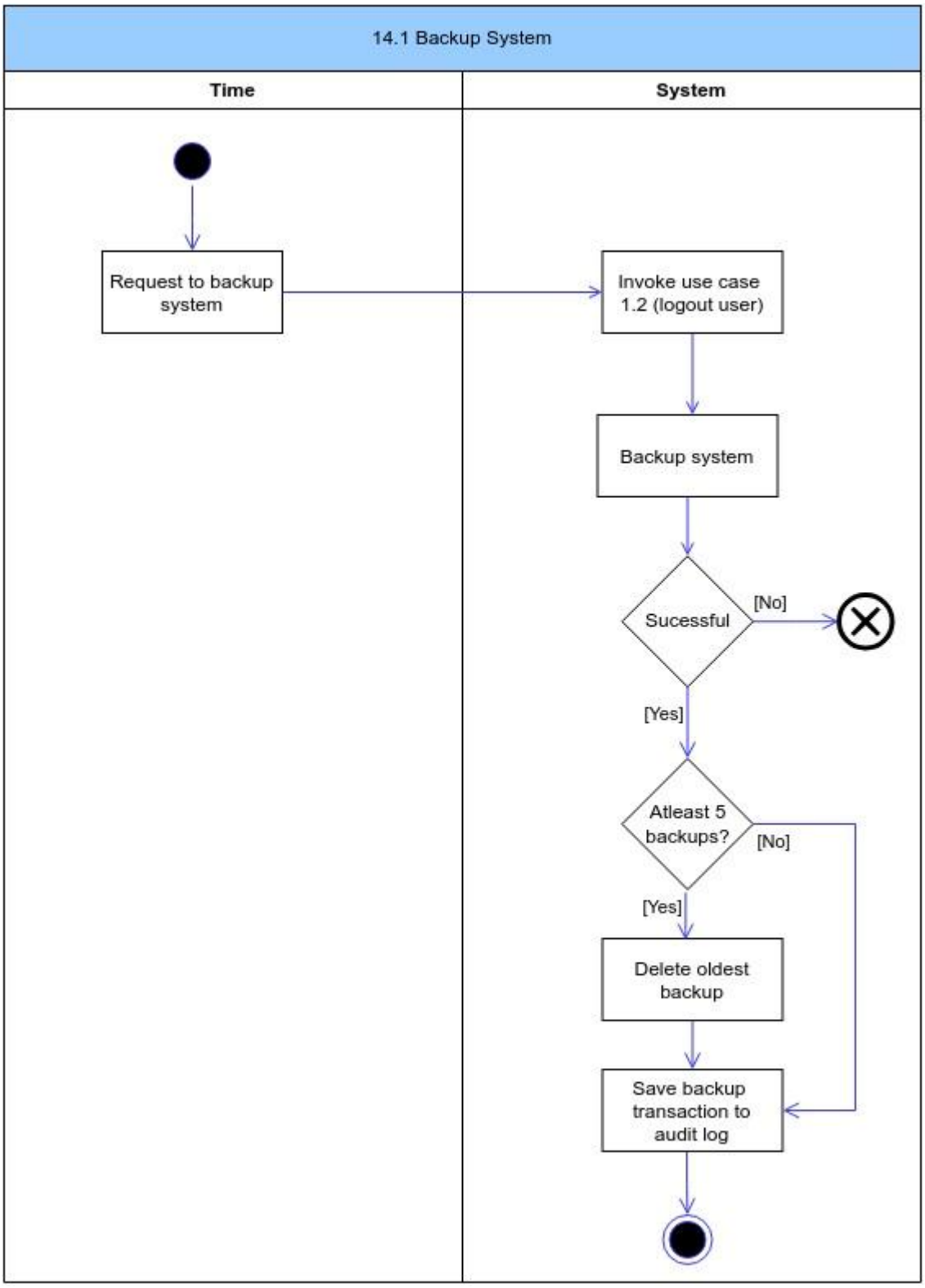


Figure 218 14.1 Backup System - Activity Diagram

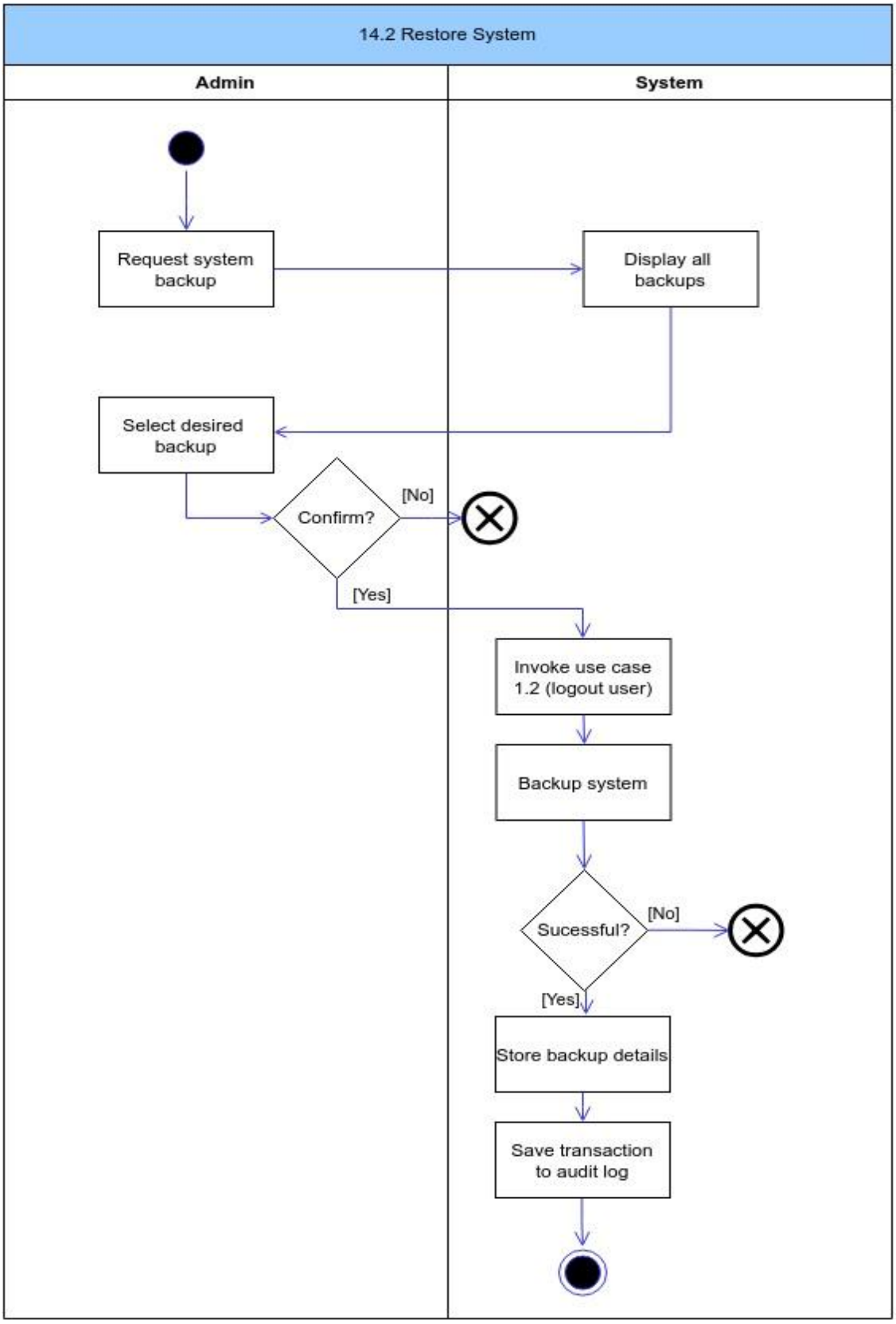


Figure 219 14.2 Restore System - Activity Diagram

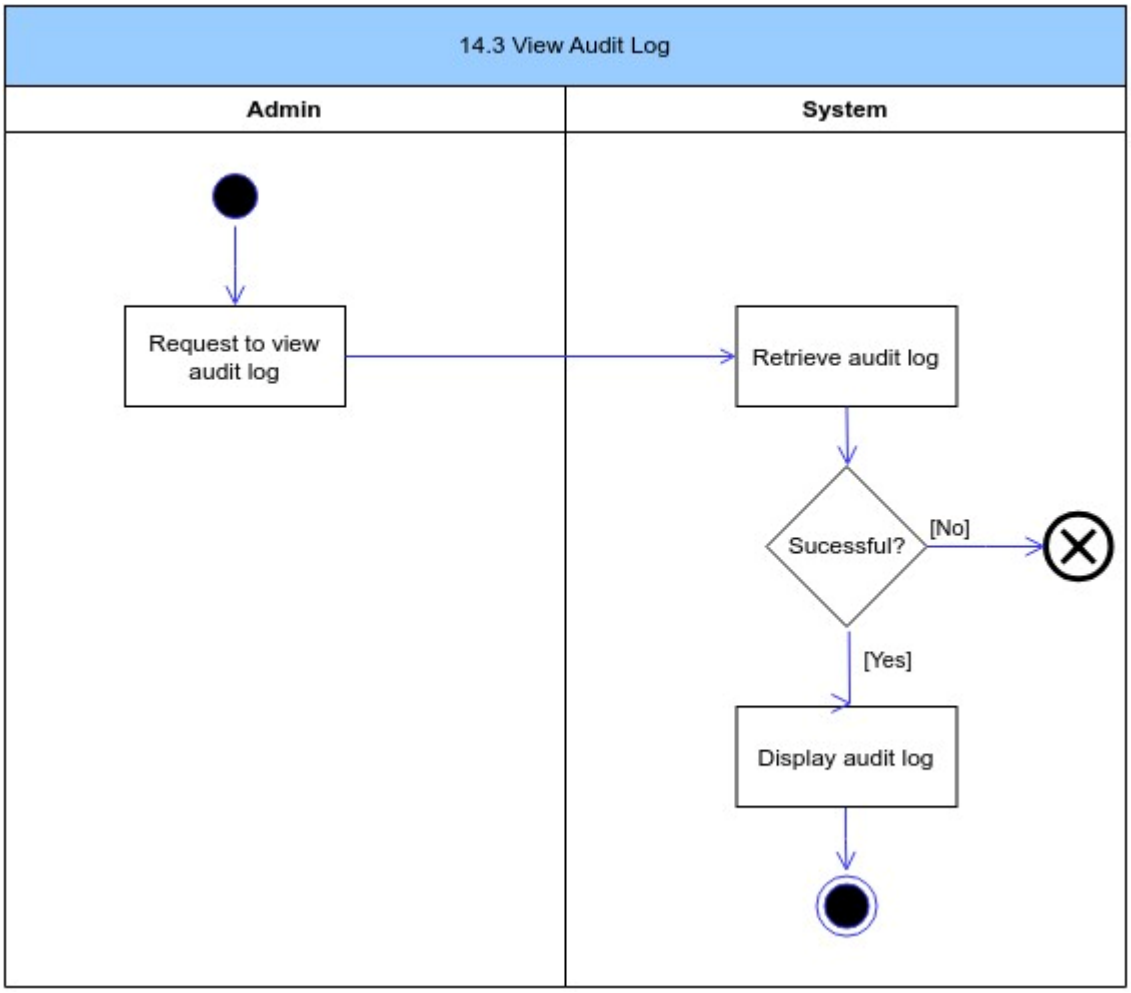


Figure 220 14.3 View Audit Log - Activity Diagram

2.10.15 Budget Subsystem Activity Diagrams

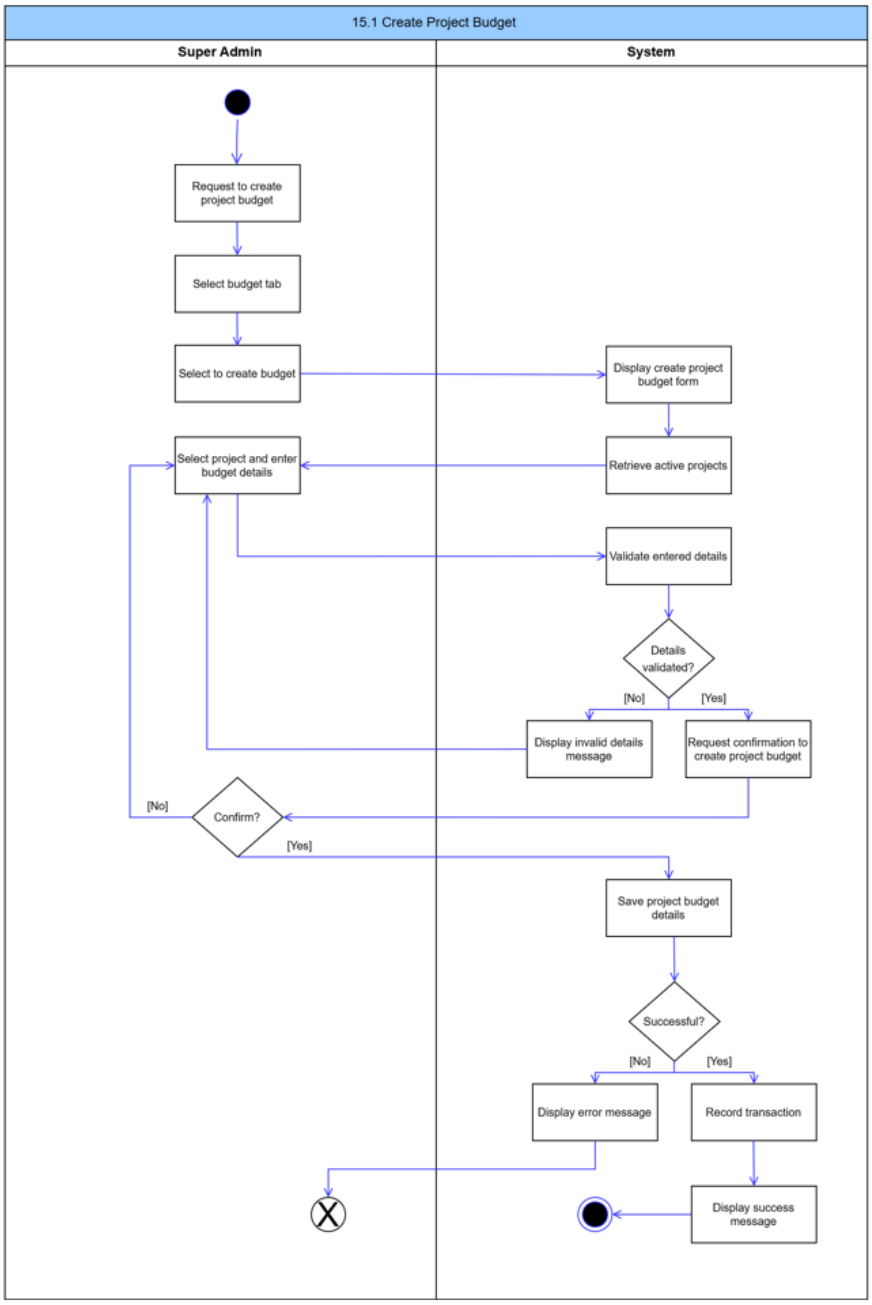


Figure 221 15.1 Create Project Budget - Activity Diagram

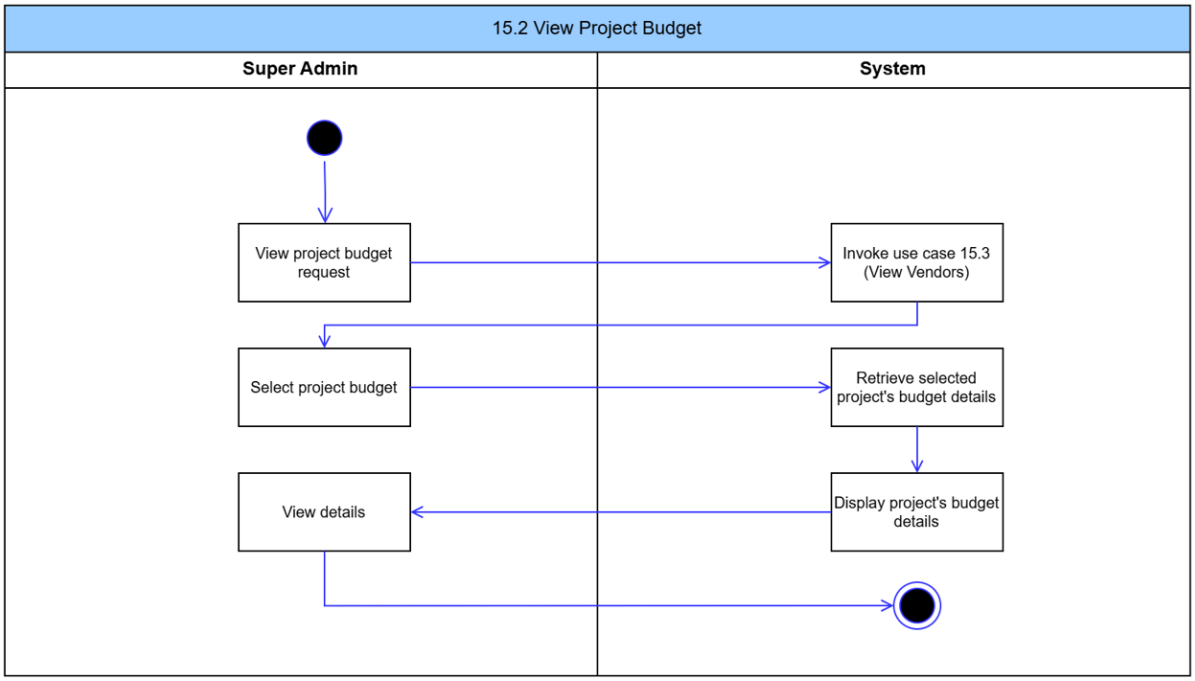


Figure 222 15.2 View Project Budget - Activity Diagram

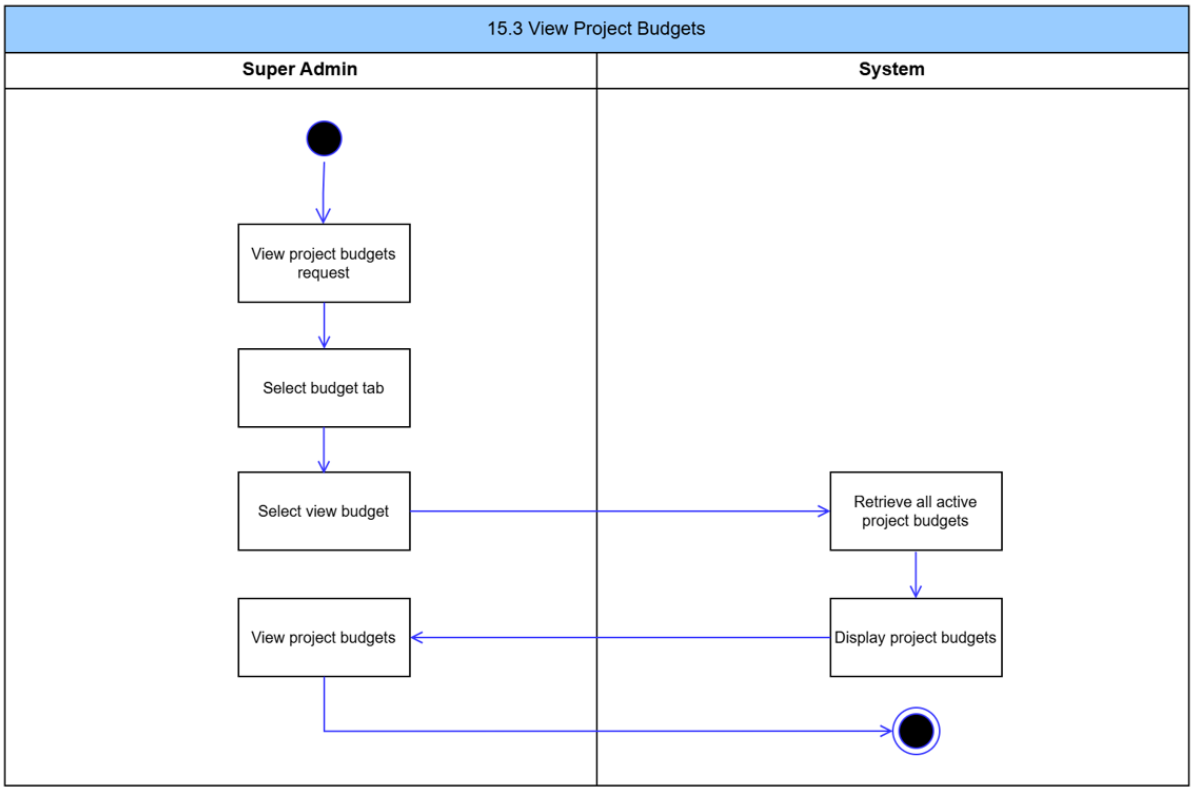


Figure 223 15.3 View Project Budgets - Activity Diagram

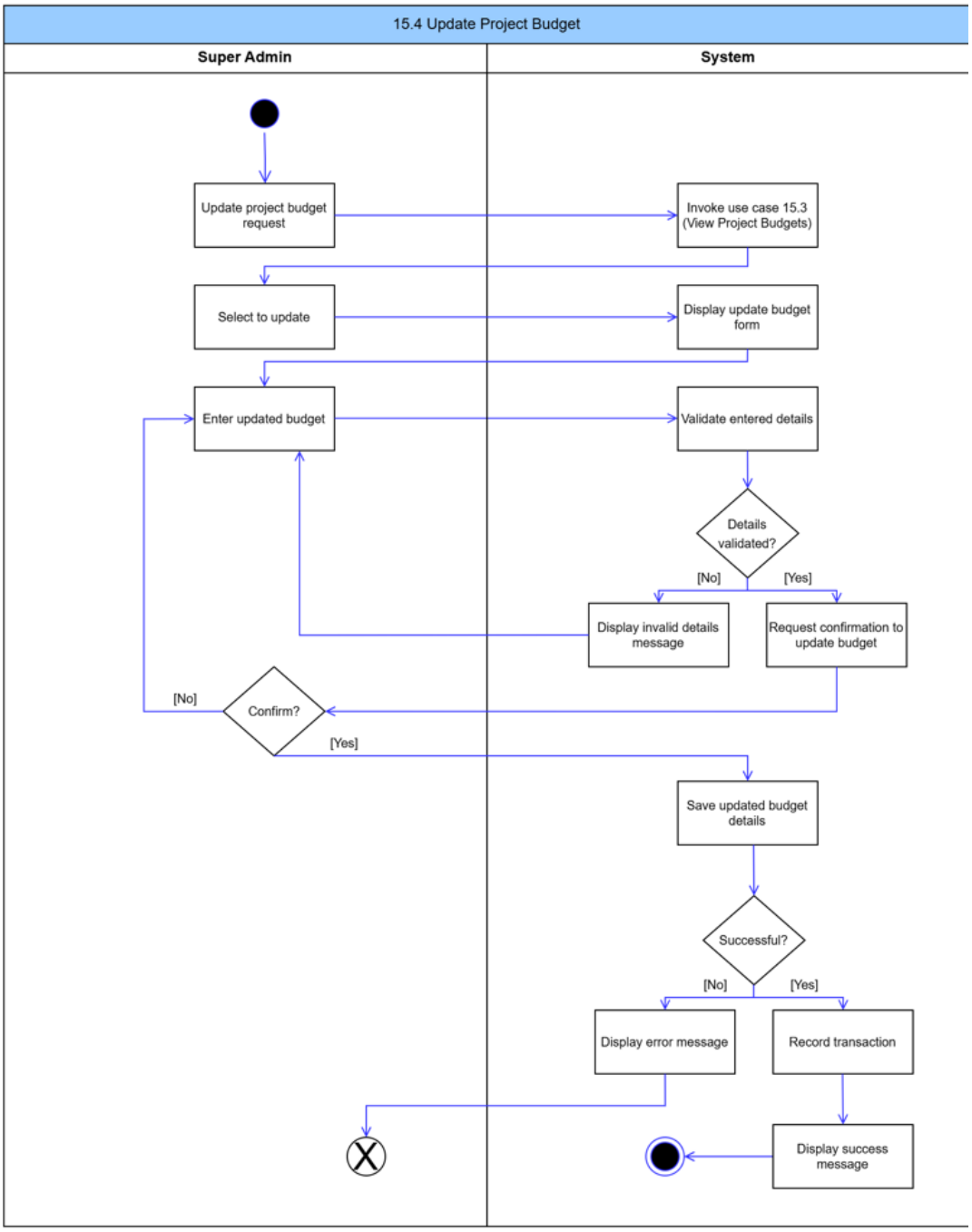


Figure 224 15.4 Update Project Budget - Activity Diagram

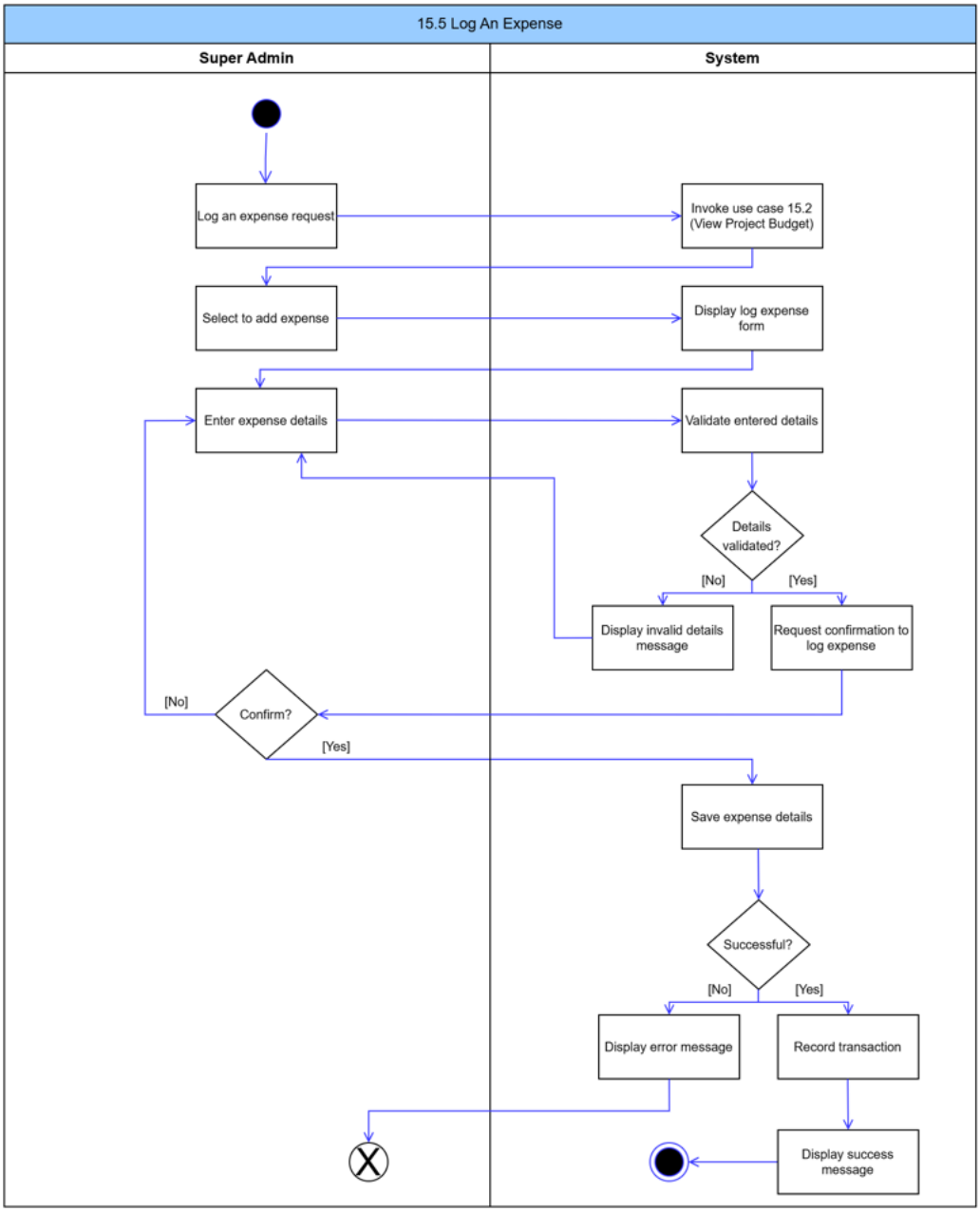


Figure 225 15.5 Log An Expense - Activity Diagram

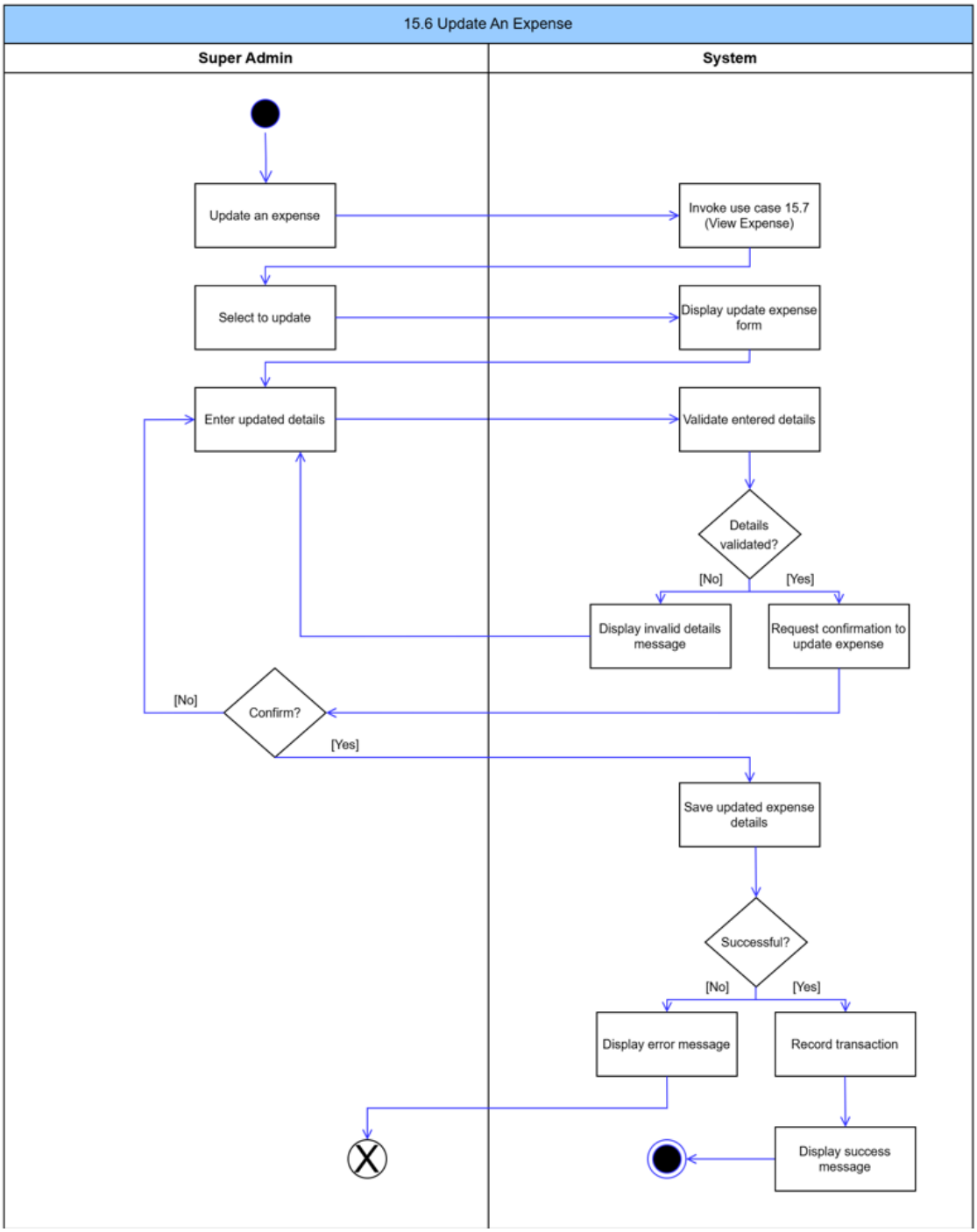


Figure 226 15.6 Update An Expense - Activity Diagram

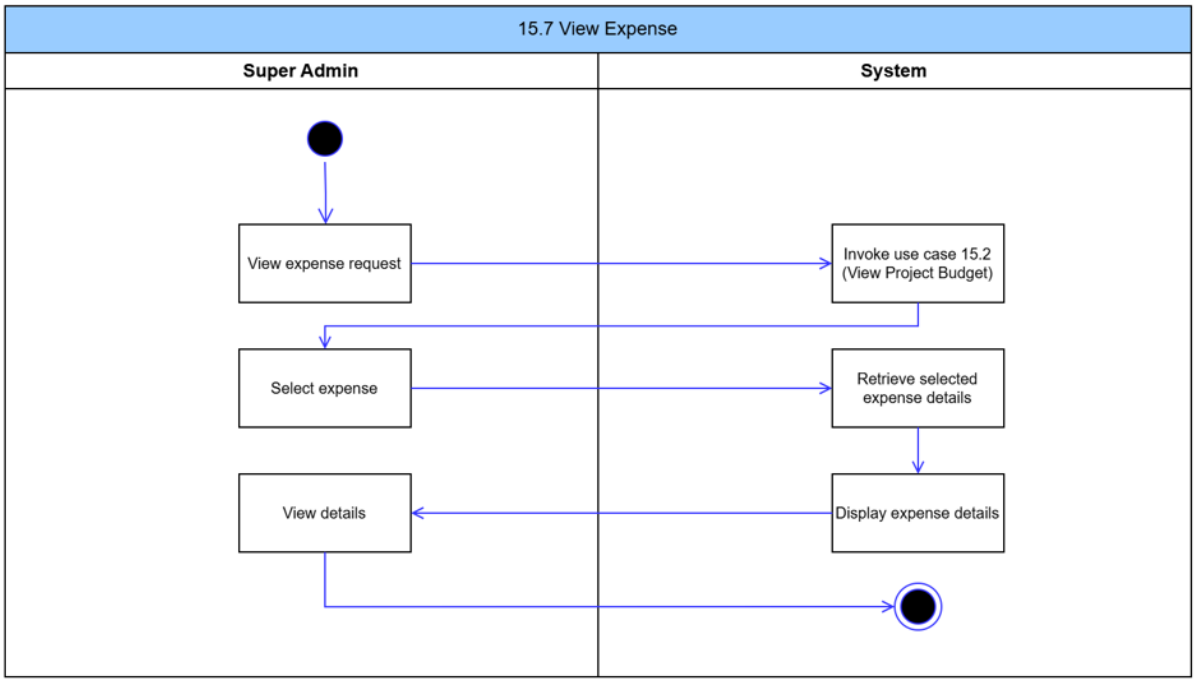


Figure 227 15.7 View Expense - Activity Diagram

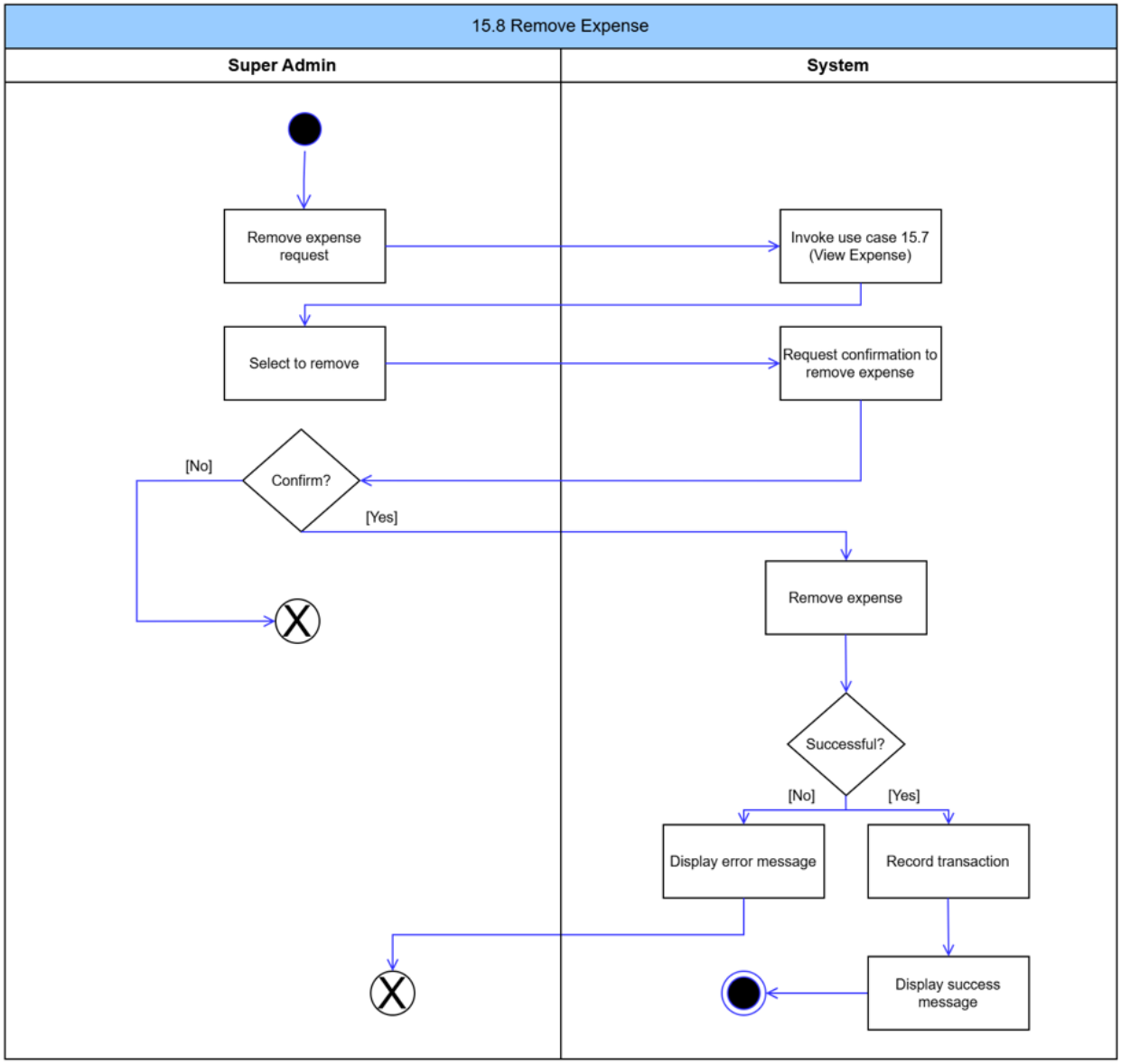


Figure 228 15.8 Remove Expense - Activity Diagram

2.11 Context Diagrams

2.11.1 User Subsystem Context Diagram

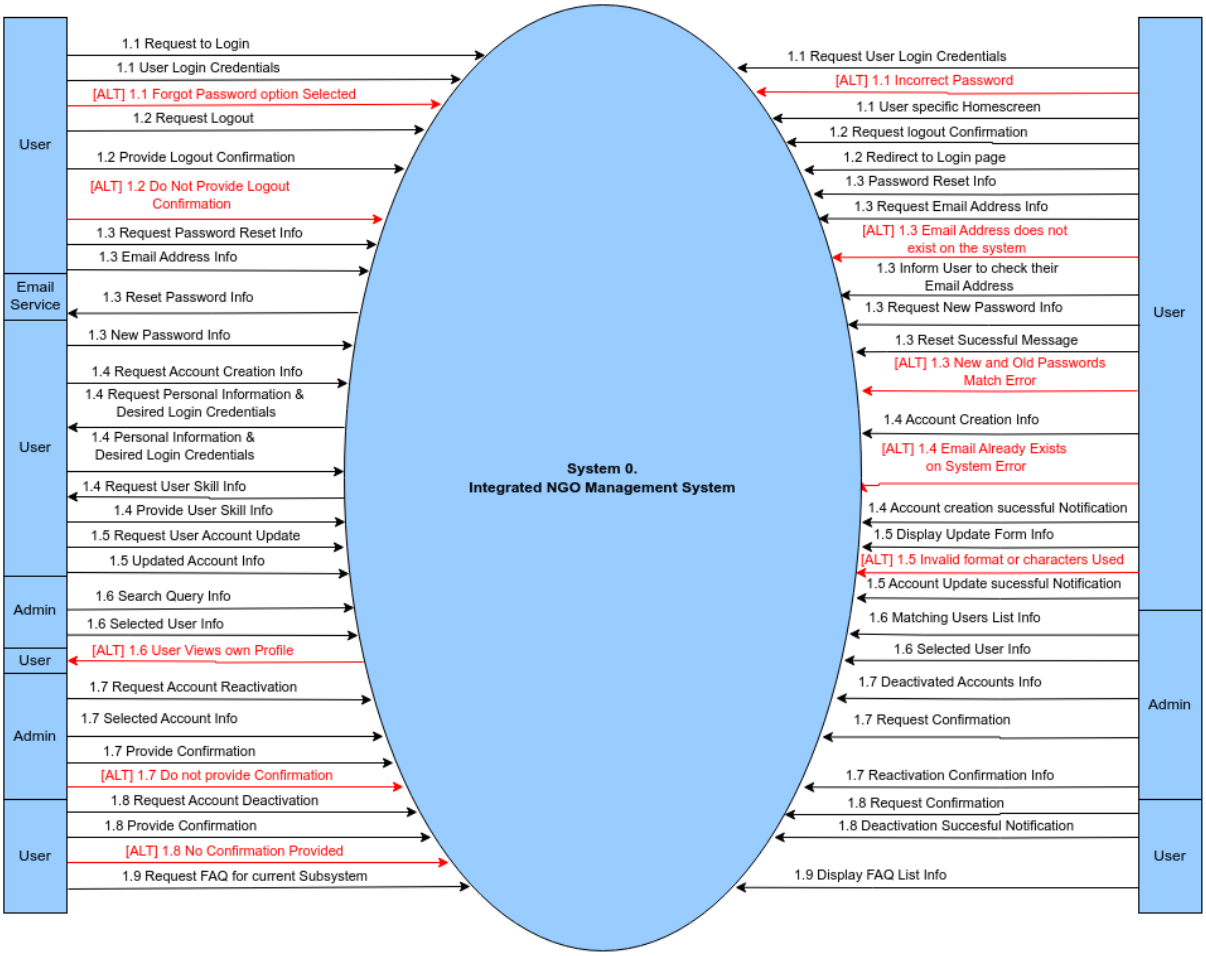


Figure 229: User Subsystem Context Diagram

2.11.2 Admin Subsystem Context Diagram

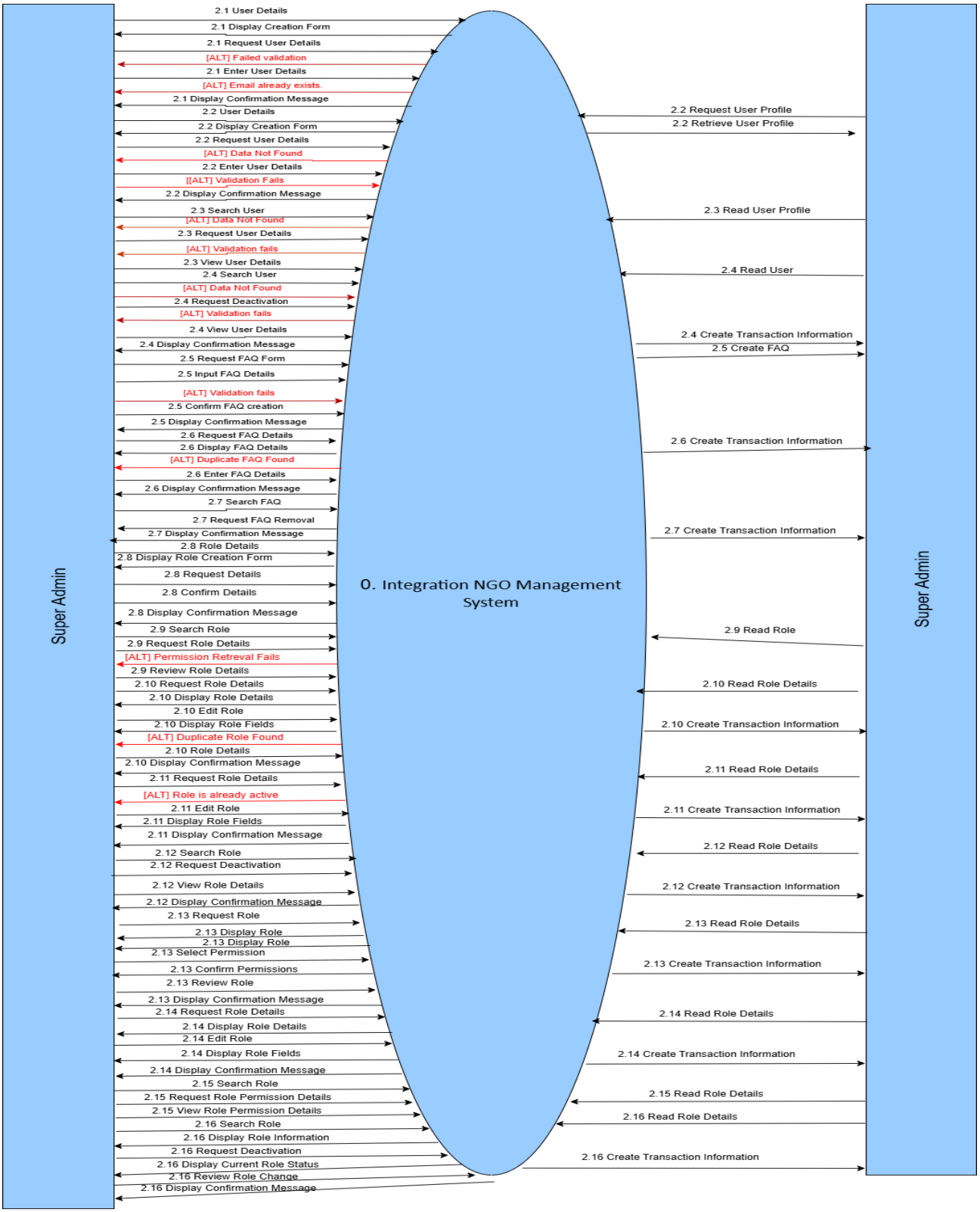


Figure 230 Admin Subsystem Context Diagram

2.11.3 Inventory Subsystem Context Diagram

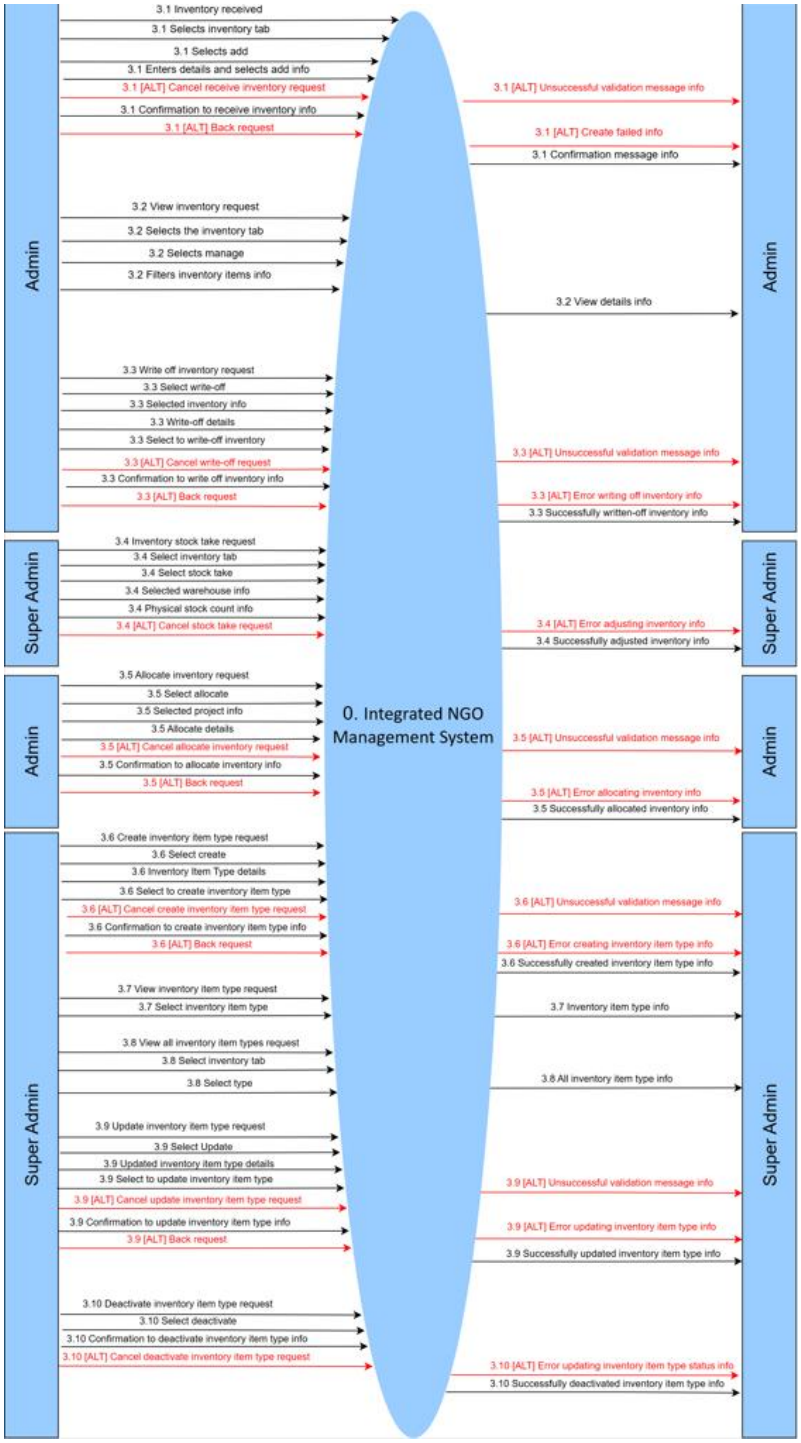


Figure 231: Inventory Subsystem Context Diagram

2.11.4 Group Subsystem Context Diagram

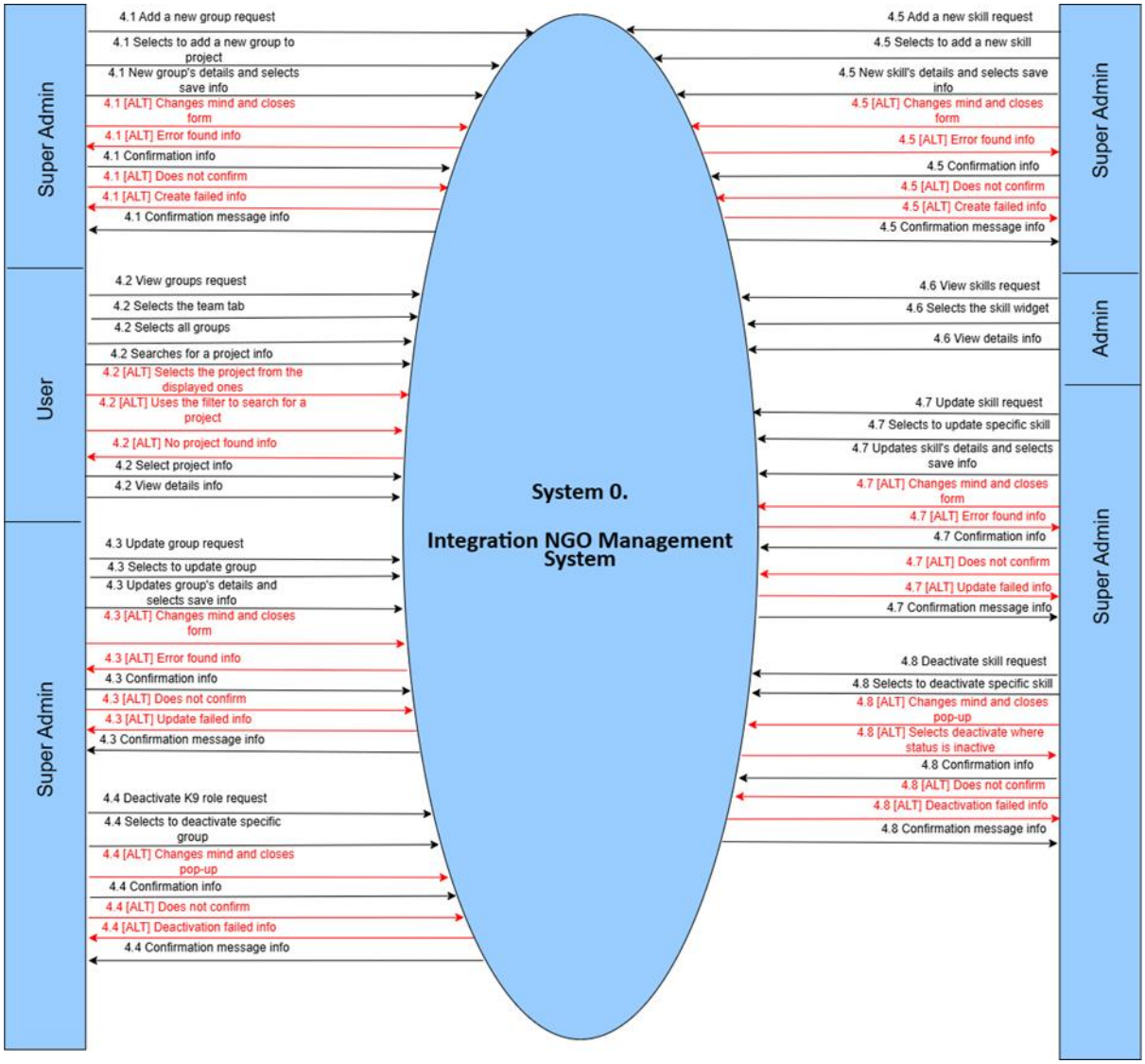


Figure 232: Group Subsystem Context Diagram

2.11.5 Warehouse Subsystem Context Diagram

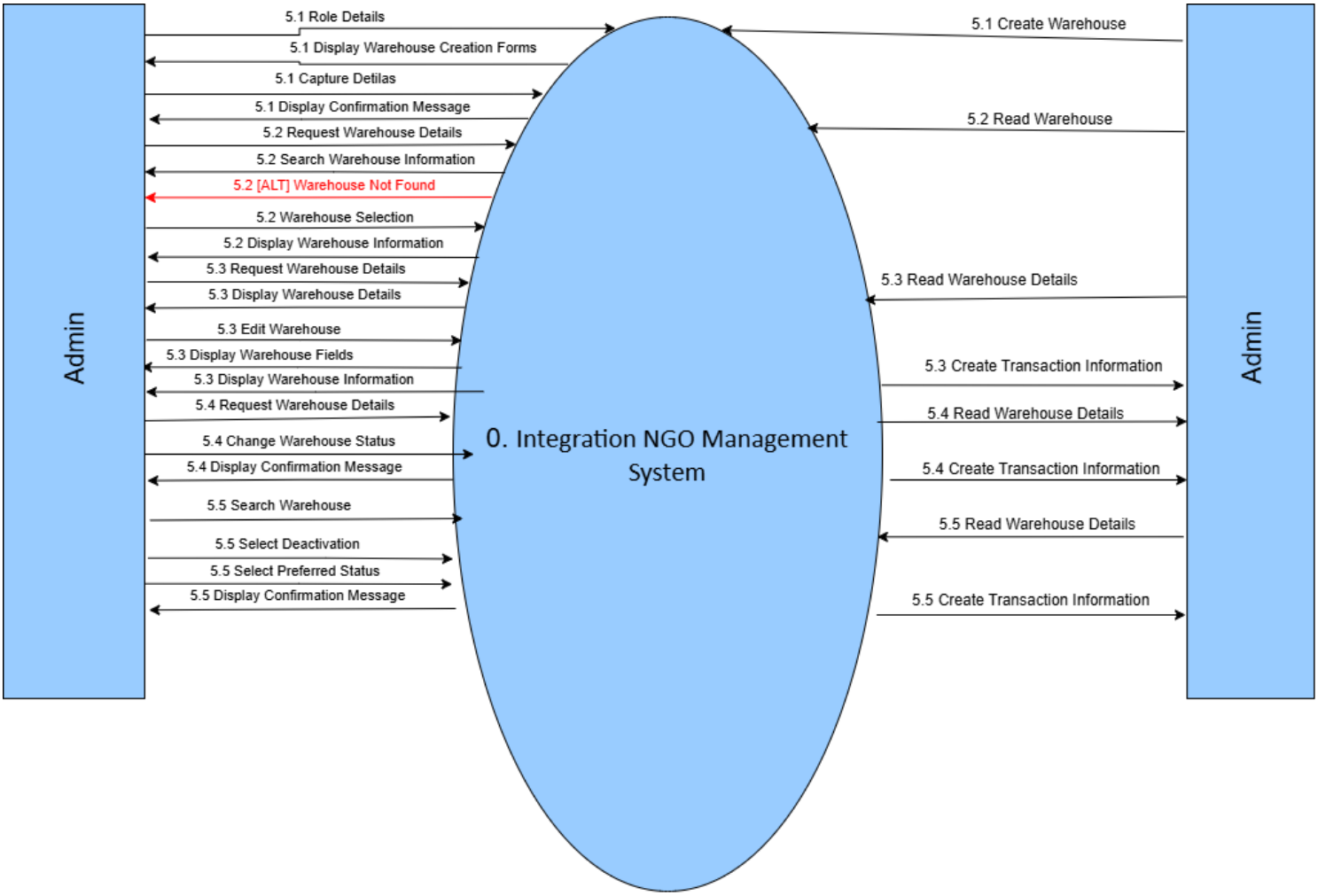


Figure 233 Warehouse Subsystem Context Diagram

2.11.6 Vendor Subsystem Context Diagram

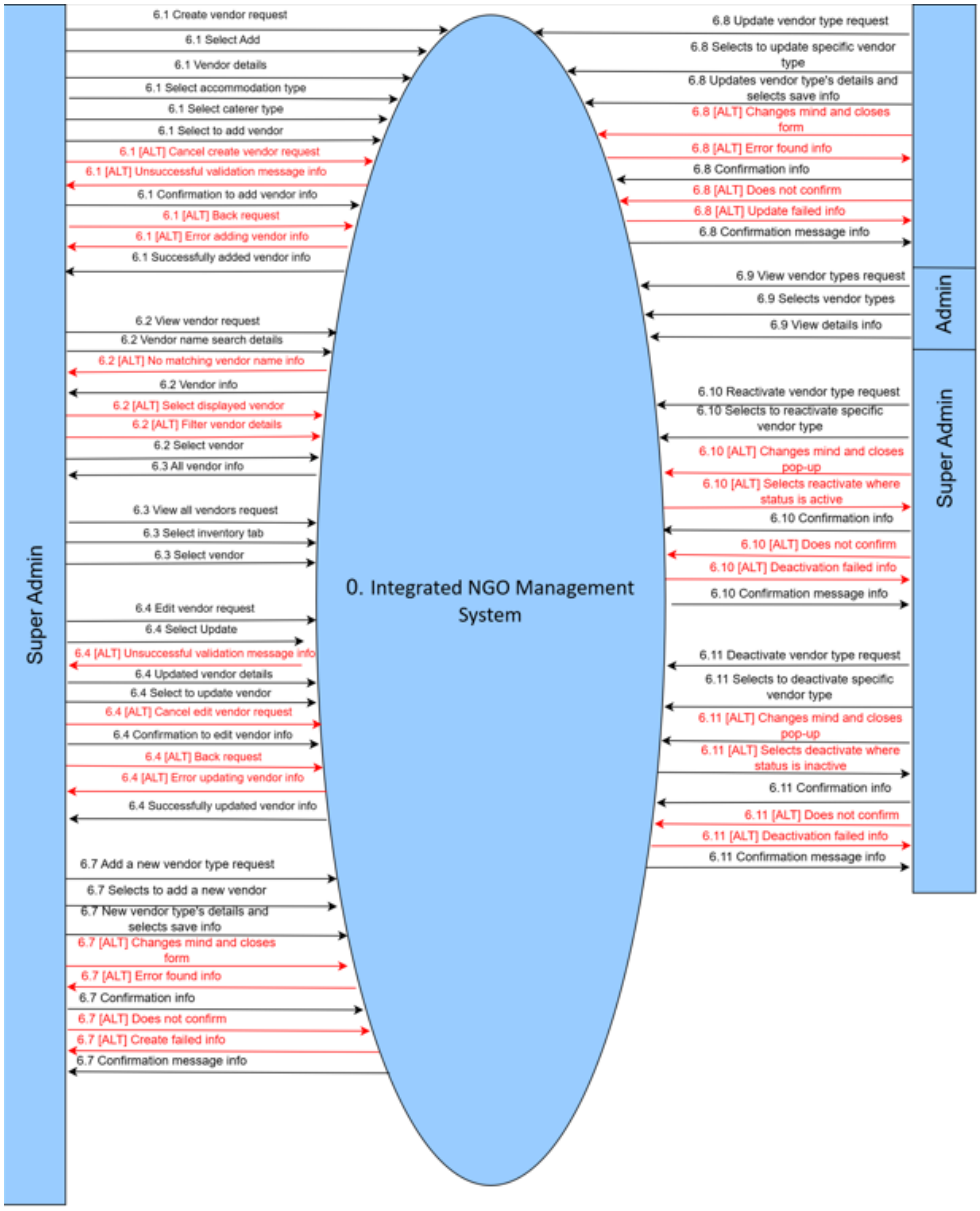


Figure 234: Vendor Subsystem Context Diagram

2.11.7 Project Management Subsystem Context Diagram

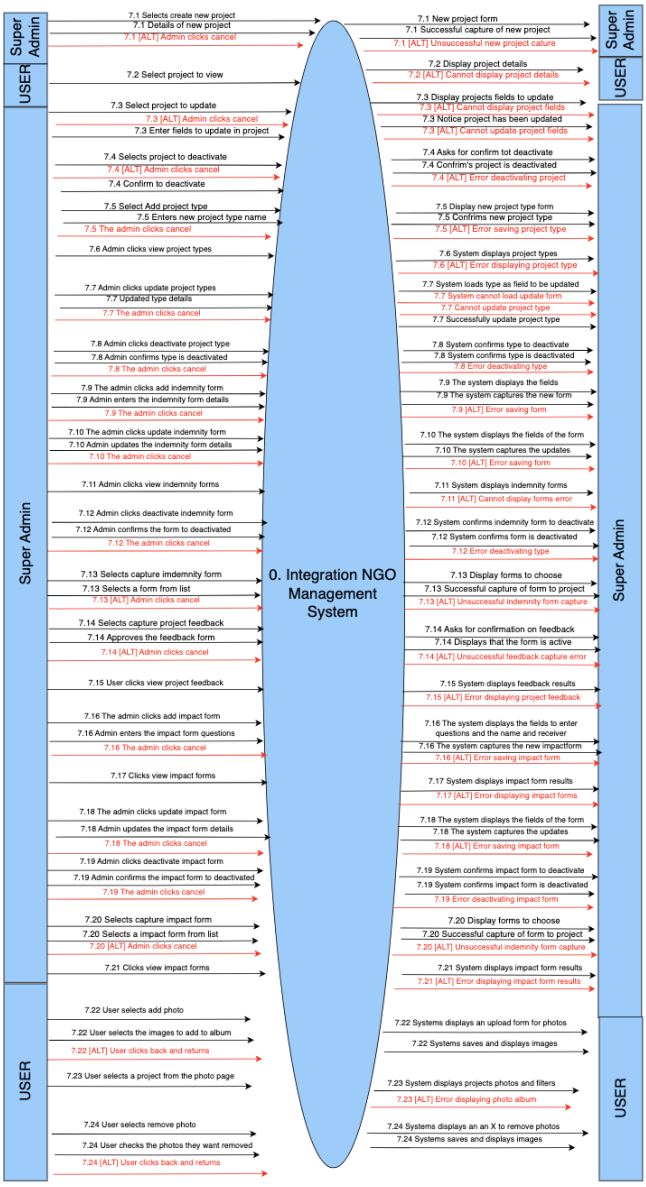


Figure 235: Project Management Subsystem Context Diagram

2.11.8 K9 Subsystem Context Diagram

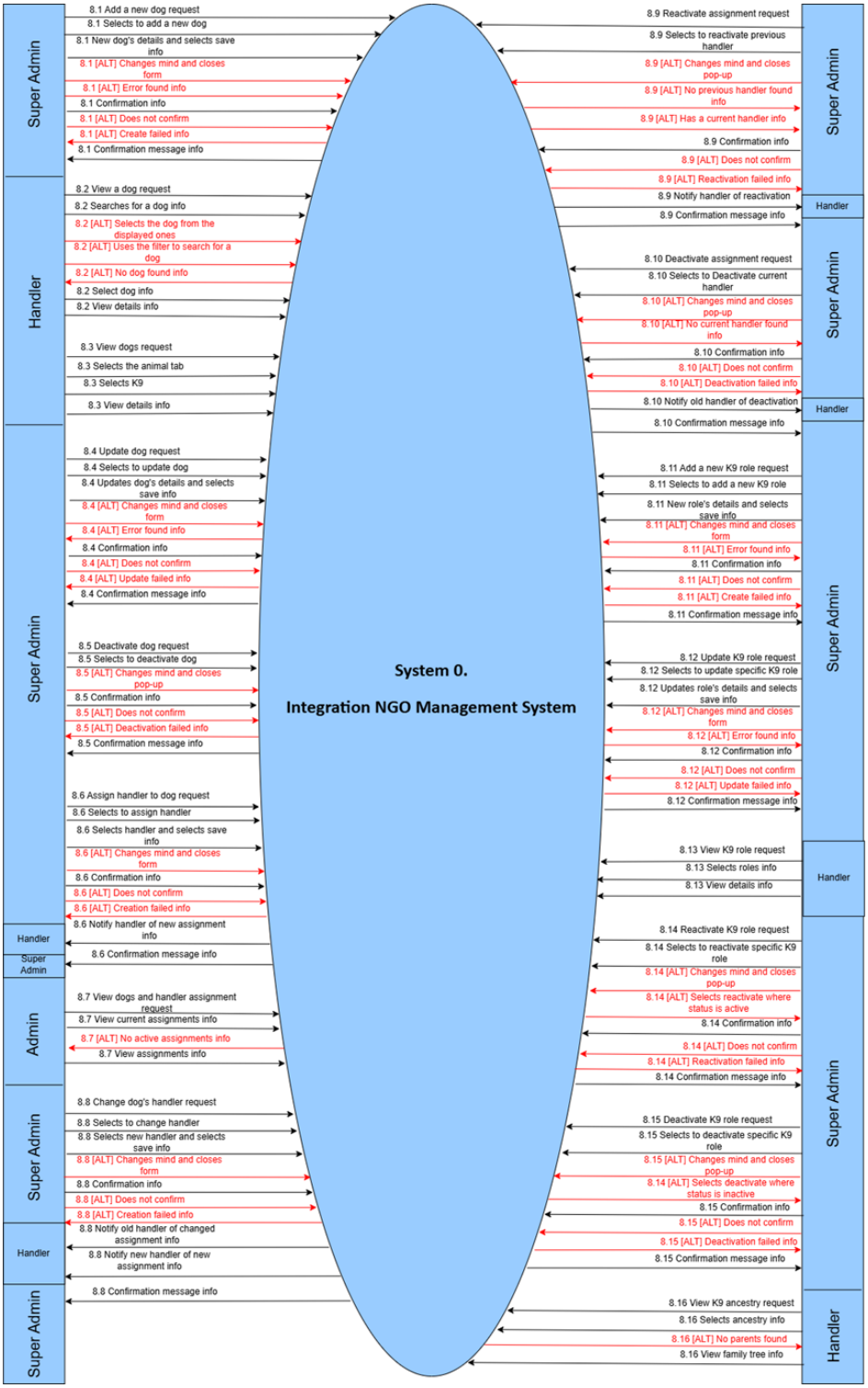


Figure 236: K9 Subsystem Context Diagram

2.11.9 Rhino Subsystem Context Diagram

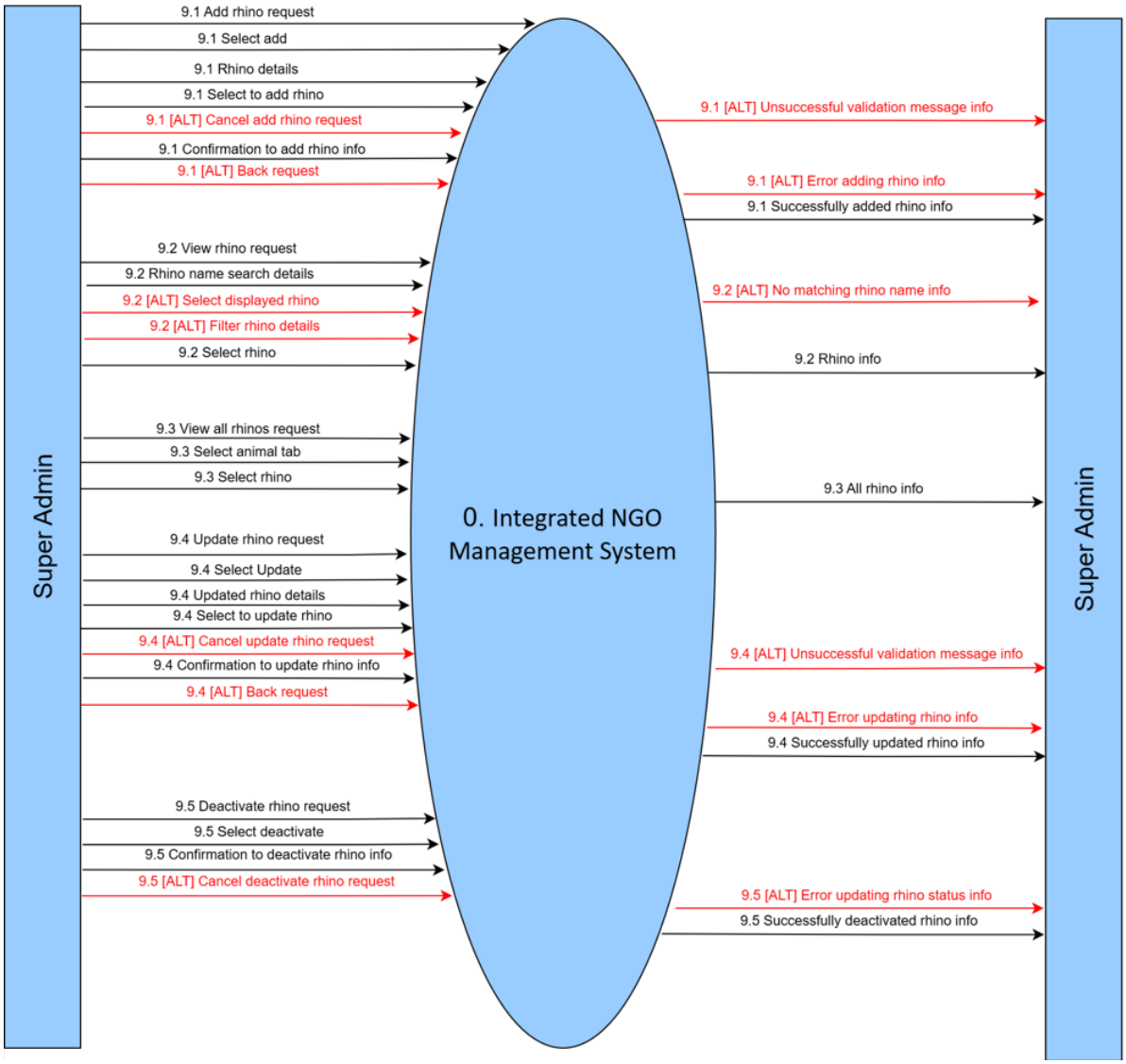


Figure 237: Rhino Subsystem Context Diagram

2.11.10 Elephant Subsystem Context Diagram

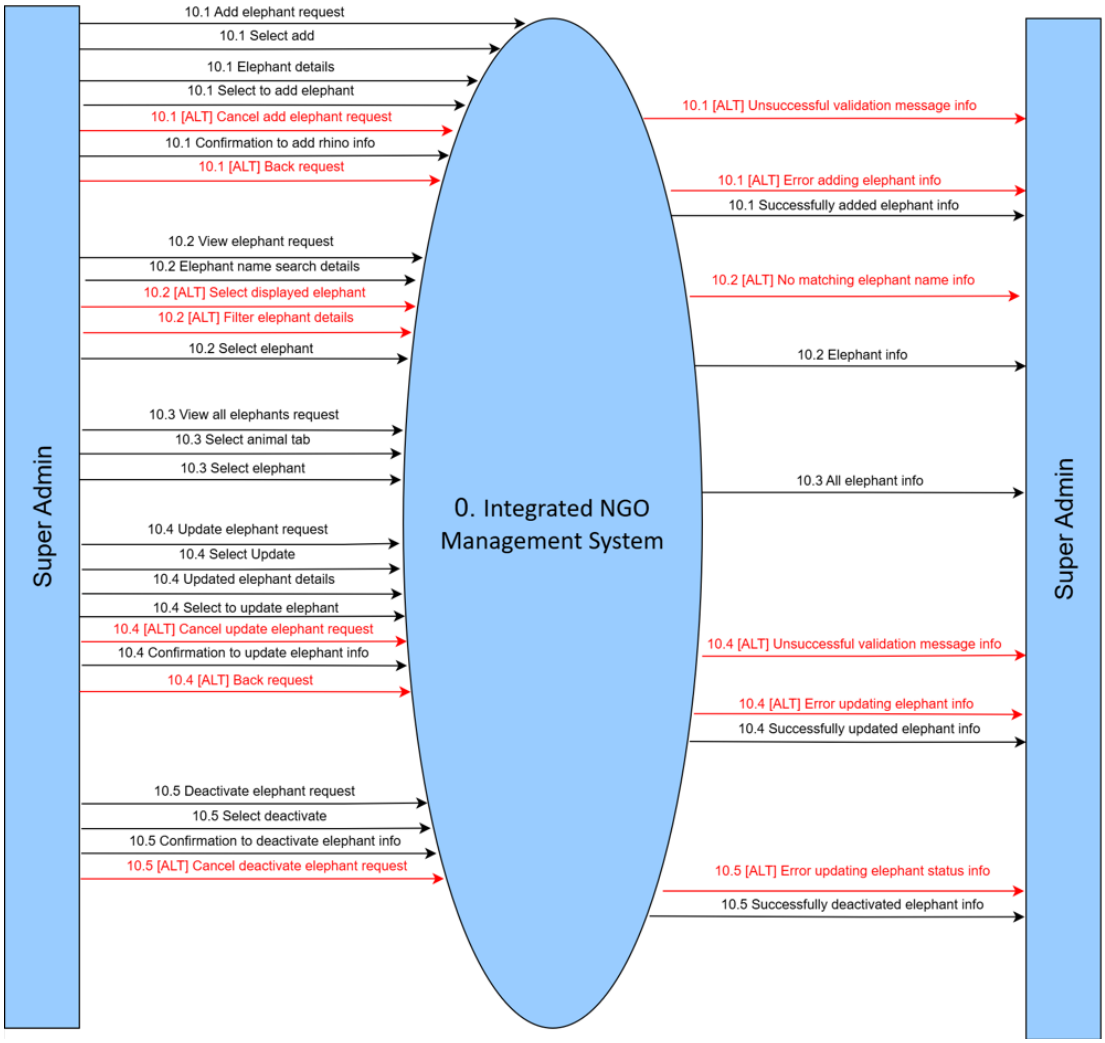


Figure 238: Elephant Subsystem Context Diagram

2.11.11 Messages Subsystem Context Diagram

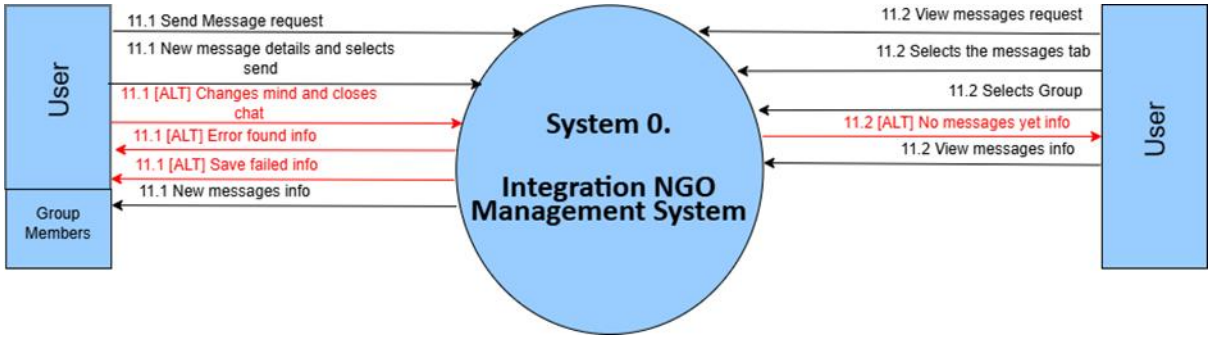


Figure 239: Messages Subsystem Context Diagram

2.11.12 Report Subsystem Context Diagram

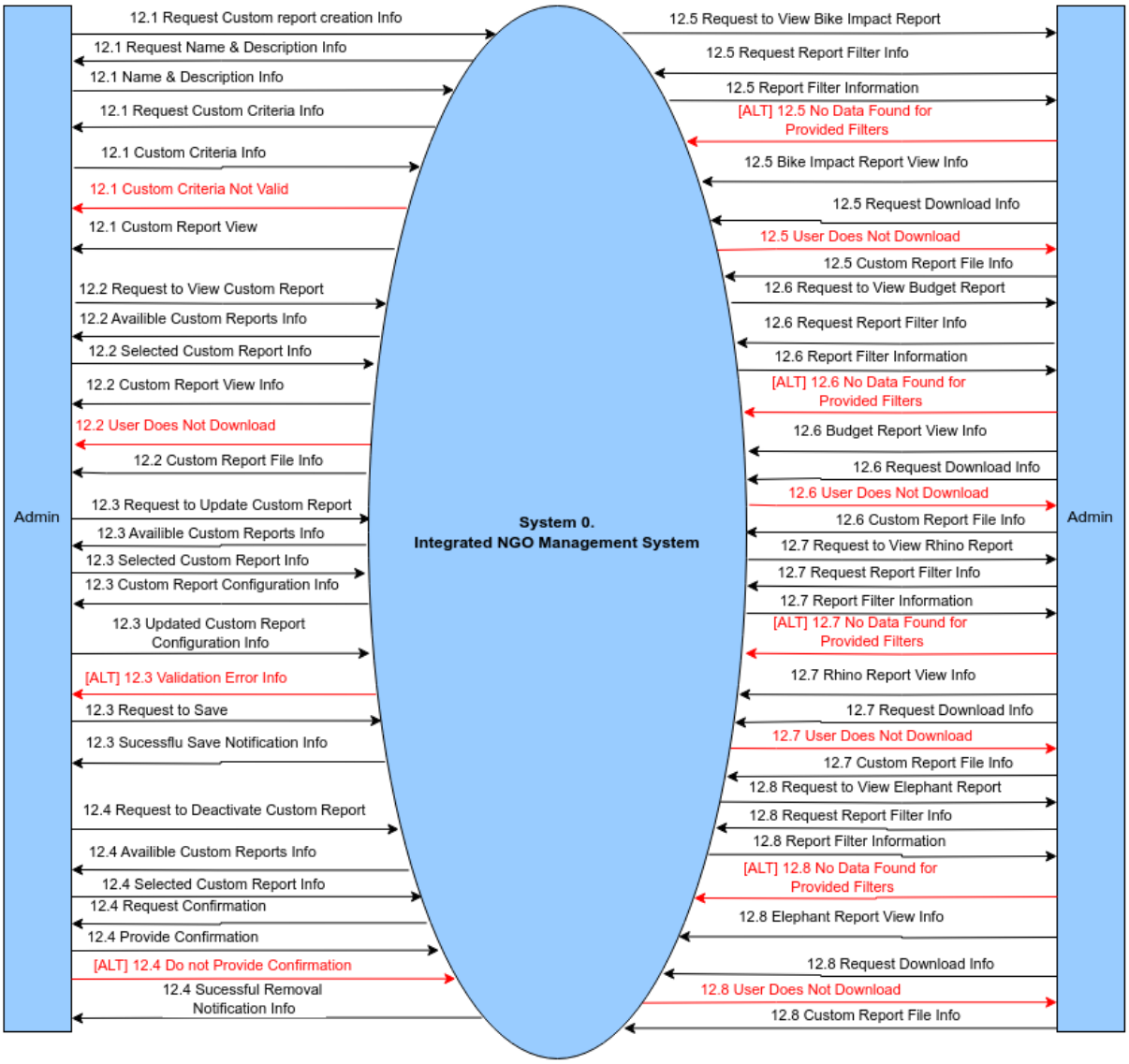


Figure 240: Report Subsystem Context Diagram

2.11.13 Machine Learning Subsystem Context Diagram

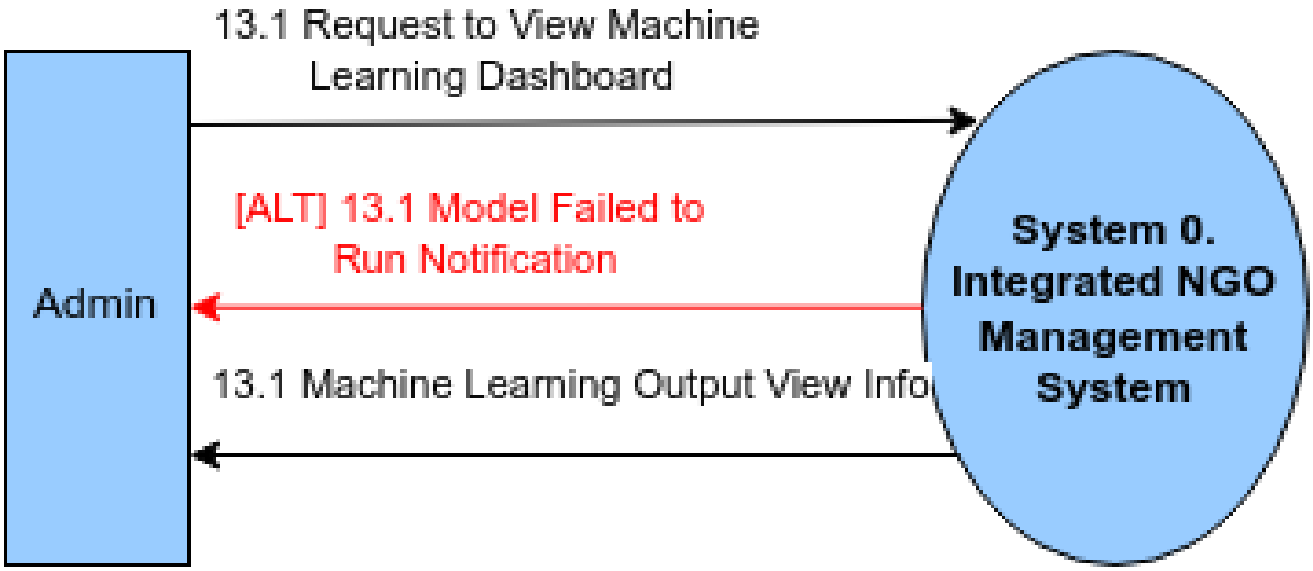


Figure 241: Machine Learning Subsystem Context Diagram

2.11.14 System Subsystem Context Diagram

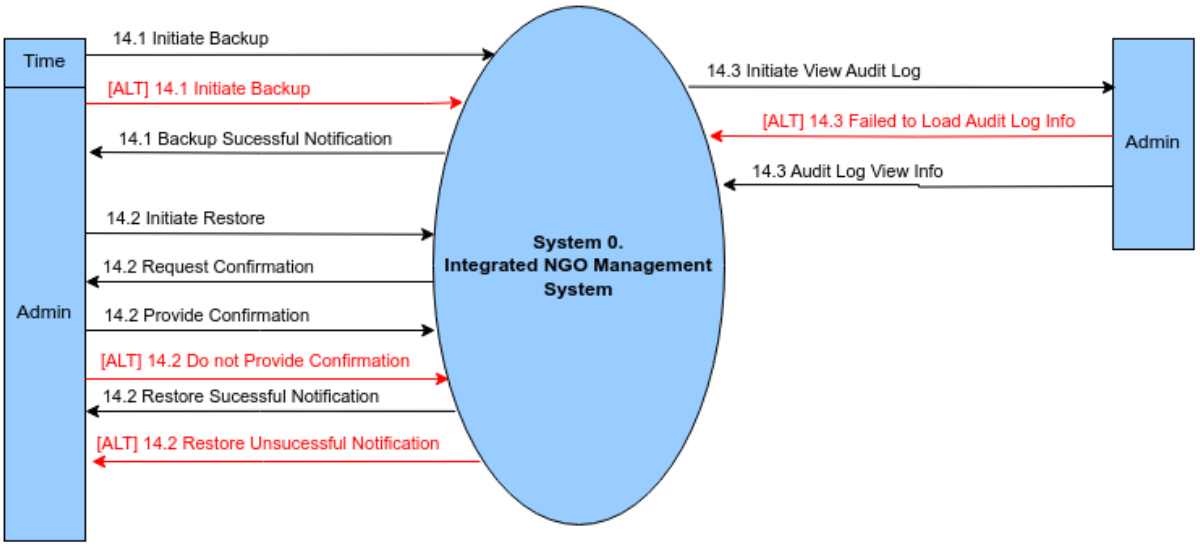


Figure 242: System Subsystem Context Diagram

2.11.15 Budget Subsystem Context Diagram

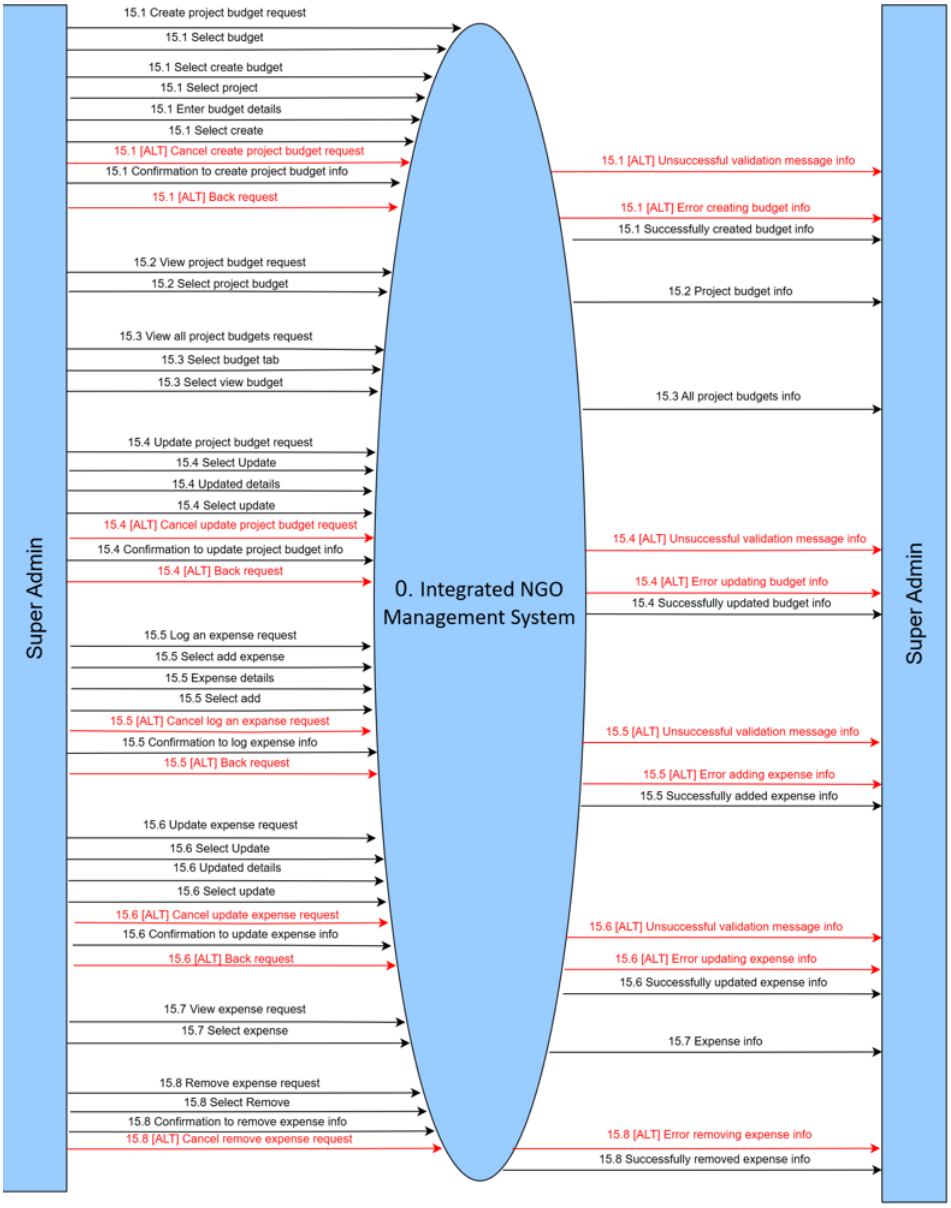


Figure 243: Budget Subsystem Context Diagram

2.12 UML Class Diagram

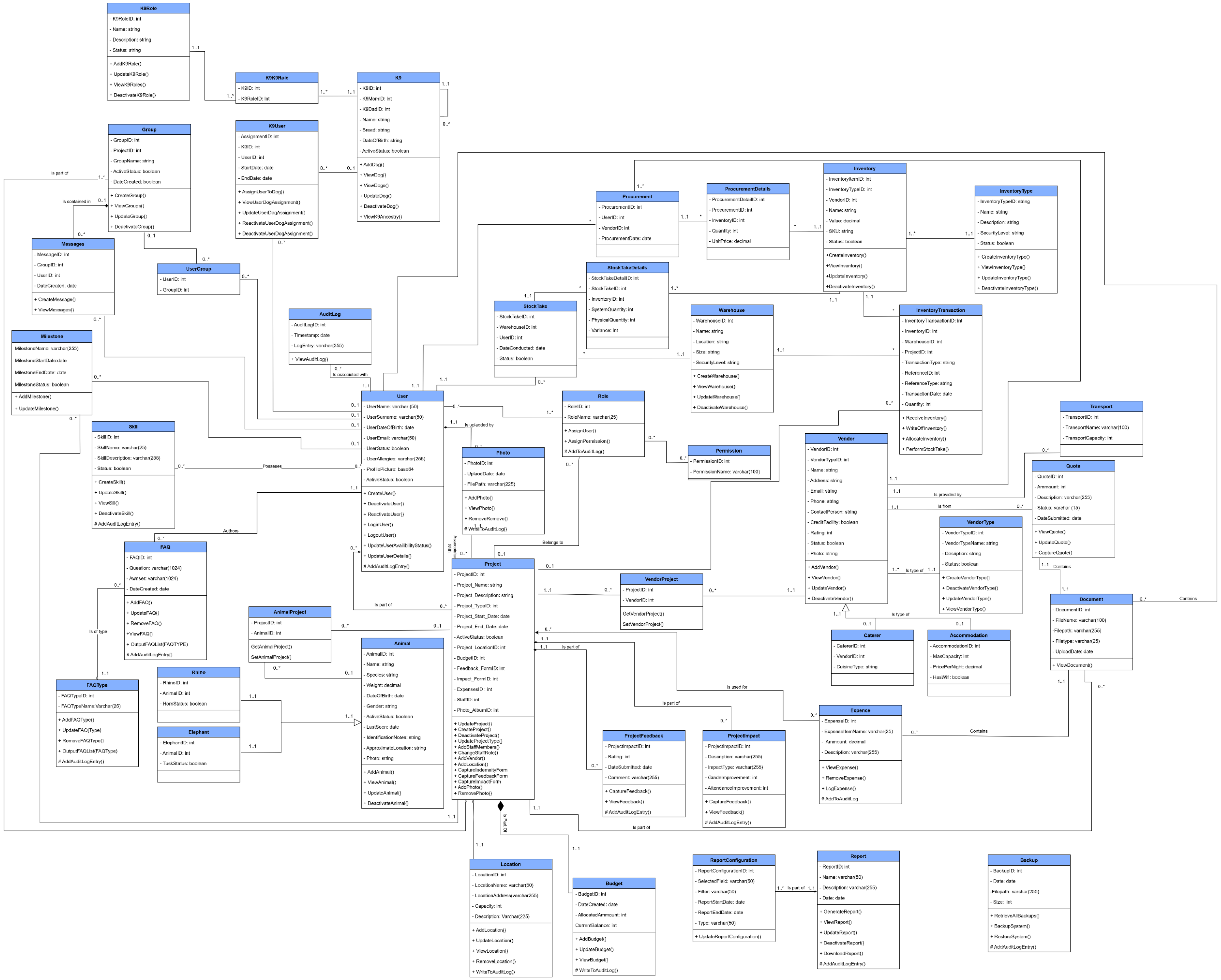


Figure 244 Class Diagram

2.13 Validation of Functional Specifications Against the Business Requirements

2.13.1 User Subsystem

User Subsystem		
Requirement	Use Case	Entities
1.1 User Login	1.1 User Login	User
1.2 User Logout	1.2 User Logout	-
1.3 Reset Password	1.3 Reset Password	User
1.4 Create User	1.4 Create User	User, UserSkill, AuditLog
1.5 Update User	1.5 Update User	User, UserSkill, AuditLog
1.6 View User	1.6 View User	User, UserSkill
1.7 Reactivate User	1.7 Reactivate User	User, AuditLog
1.8 Deactivate User	1.8 Deactivate User	User, AuditLog
1.9 View FAQ	1.9 View FAQ	FAQ, FAQType

Table 265: User Subsystem Validation

2.13.2 Admin Subsystem

Admin Subsystem		
Requirement	Use Case	Entities
2.1 Create User Profile	2.1 Create User Profile	User, User Skill, AuditLog
2.2 Update User Profile	2.2 Update User Profile	User, User Skill, Audit Log
2.3 View User Profile	2.3 View User Profile	User, UserSkill
2.4 Deactivate User Profile	2.4 Deactivate User Profile	User, AuditLog
2.5 Add FAQ	2.5 Add FAQ	User, FAQ, FAQType, AuditLog
2.6 Update FAQ	2.6 Update FAQ	User, FAQ, FAQType, AuditLog
2.7 Remove FAQ	2.7 Remove FAQ	User, FAQ, AuditLog
2.8 Create Role	2.8 Create Role	User, Role, AuditLog
2.9 View Role	2.9 View Role	Role
2.10 Update Role	2.10 Update Role	Role, AuditLog
2.11 Reactivate Role	2.11 Reactivate Role	Role, AuditLog
2.12 Deactivate Role	2.12 Deactivate Role	Role, AuditLog
2.13 Create Role Permission	2.13 Create Role Permission	RolePermissions, AuditLog
2.14 Update Role Permission	2.14 Update Role Permission	RolePermissions, AuditLog
2.15 View Role Permission	2.15 View Role Permission	Role, RolePermissions,
2.16 Deactivate Role Permission	2.16 Deactivate Role Permission	RolePermissions, AuditLog

Table 266: Admin Subsystem Validation

2.13.3 Inventory Subsystem

Inventory Subsystem		
Requirement	Use Case	Entities
3.1 Receive Inventory Item	3.1 Receive Inventory Item	InventoryTransaction, Procurement, ProcurementDetails
3.2 View Inventory	3.2 View Inventory	Inventory, InventoryType, InventoryTransaction
3.3 Write-Off Inventory	3.3 Write-Off Inventory	InventoryTransaction, AuditLog
3.4 Inventory Stock Take	3.4 Inventory Stock Take	StockTake, StockTakeDetails, InventoryTransaction
3.5 Allocate Inventory	3.5 Allocate Inventory	InventoryTransation, AuditLog
3.6 Create Inventory Item Type	3.6 Create Inventory Item Type	RolePermission, AuditLog
3.7 View Inventory Item Type	3.7 View Inventory Item Type	InventoryType
3.8 View Inventory Item Types	3.8 View Inventory Item Types	InventoryType
3.9 Update Inventory Item Type	3.9 Update Inventory Item Type	InventoryType, AuditLog
3.10 Deactivate Inventory Item Type	3.10 Deactivate Inventory Item Type	InventoryType, AuditLog

Table 267: Inventory Subsystem Validation

2.13.4 Group Subsystem

Group Subsystem		
Requirement	Use Case	Entities
4.1 Create Group	4.1 Create Group	UserGroup, AuditLog
4.2 View Group	4.2 View Group	UserGroup
4.3 Update Group	4.3 Update Group	UserGroup, AuditLog
4.4 Deactivate Group	4.4 Deactivate Group	UserGroup, AuditLog
4.5 Create Skill	4.5 Create Skill	Skill, AuditLog
4.6 View Skill	4.6 View Skill	Skill
4.7 Update Skill	4.7 Update Skill	Skill, AuditLog
4.8 Deactivate Skill	4.8 Deactivate Skill	Skill, AuditLog

Table 268: Group Subsystem Validation

2.13.5 Warehouse Subsystem

Warehouse Subsystem		
Requirement	Use Case	Entities
5.1 Create Warehouse	5.1 Create Warehouse	Warehouse, AuditLog
5.2 View Warehouse	5.2 View Warehouse	Warehous
5.3 Update Warehouse	5.3 Update Warehouse	Warehouse, AuditLog
5.4 Reactivate Warehouse	5.4 Reactivate Warehouse	Warehouse, AuditLog
5.5 Deactivate Warehouse	5.5 Deactivate Warehouse	Warehouse, AuditLog

Table 269: Warehouse Subsystem Validation

2.13.6 Vendor Subsystem

Vendor Subsystem		
Requirement	Use Case	Entities
6.1 Create Vendor	6.1 Create Vendor	Vendor, VendorType,Accomodation, Caterer,Transport, AuditLog
6.2 View Vendor	6.2 View Vendor	Vendor
6.3 View Vendors	6.3 View Vendors	Vendor
6.4 Edit Vendor	6.4 Edit Vendor	Vendor, AuditLog
6.5 Reactivate Vendor	6.5 Reactivate Vendor	Vendor, AuditLog
6.6 Deactivate Vendor	6.6 Deactivate Vendor	VendorAuditlog
-	6.7 Create Vendor Type	VendorType, AuditLog
-	6.8 Update Vendor Type	VendorType, AuditLog
-	6.9 View Vendor Types	VendorType,
-	6.10 Reactivate Vendor Type	Warehouse, AuditLog
-	6.11 Deactivate Vendor type	VendorType, AuditLog

Table 270: Vendor Subsystem Validation

2.13.7 Project Management Subsystem

Project Management Subsystem		
Requirement	Use Case	Entities
7.1 Create Project	7.1 Create Project	ProjectUser, Project, ProjectGroup, ProjectLocation, ProjectAnimal, AuditLog
7.2 View Project	7.2 View Project	ProjectGroup, ProjectLocation, ProjectAnimal
7.3 Update Project	7.3 Update Project	ProjectUser, Project, ProjectGroup, ProjectLocation, ProjectAnimal, AuditLog
7.4 Deactivate Project	7.4 Deactivate Project	ProjectUser, Project, ProjectGroup, ProjectLocation, ProjectAnimal, AuditLog
7.5 Create Project Type	7.5 Create Project Type	ProjectType, AuditLog
7.6 View Project Type	7.6 View Project Type	ProjectType
7.7 Update Project Type	7.7 Update Project Type	ProjectType, AuditLog
7.8 Deactivate Project Type	7.8 Deactivate Project Type	ProjectType, AuditLog
2.2.7.9 Add Indemnity Form	2.2.7.9 Add Indemnity Form	Document, AuditLog
7.10 Update Indemnity Form	7.10 Update Indemnity Form	Document, AuditLog
7.11 View Indemnity Form	7.11 View Indemnity Form	Document
7.12 Remove Indemnity Form	7.12 Remove Indemnity Form	Document, AuditLog
7.13 Capture Indemnity Form	7.13 Capture Indemnity Form	ProjectIndemnity
14 Capture Project Feedback	14 Capture Project Feedback	ProjectFeedback
7.15 View Project Feedback	7.15 View Project Feedback	ProjectFeedback

Project Management Subsystem		
Requirement	Use Case	Entities
7.16 Create Impact Feedback Form	7.16 Create Impact Feedback Form	ProjectFeedback, AuditLog
7.17 View Impact Feedback Form	7.17 View Impact Feedback Form	ProjectFeedback
7.18 Update Impact Feedback Form	7.18 Update Impact Feedback Form	ImpactFeedback,AuditLog
7.19 Deactivate Impact Feedback Form	7.19 Deactivate Impact Feedback Form	ImpactFeedback,AuditLog
7.20 Capture Impact Feedback	7.20 Capture Impact Feedback	ImpactFeedback
7.21 View Impact Feedback	7.21 View Impact Feedback	ImpactFeedback
7.22 Add Photo	7.22 Add Photo	Photo, AuditLog
7.23 View Photo	7.23 View Photo	Photo
7.24 Delete Photo	7.24 Delete Photo	Photo, AuditLog

Table 271:Project Management Subsystem Validation

2.13.8 K9 Subsystem

K9 Subsystem		
Requirement	Use Case	Entities
8.1 Add Dog	8.1 Add Dog	K9, AuditLog
8.2 View Dog	8.2 View Dog	K9
8.3 View All Dogs	8.3 View All Dogs	K9
8.4 Update Dog	8.4 Update Dog	K9, AuditLog
8.5 Deactivate Dog	8.5 Deactivate Dog	K9, AuditLog
8.6 Assign User to Dog	8.6 Assign User to Dog	K9User, AuditLog
8.7 View User-Dog Assignment	8.7 View User-Dog Assignment	K9User
8.8 Update User-Dog Assignment	8.8 Update User-Dog Assignment	K9User, AuditLog
8.9 Reactivate User-Dog Assignment	8.9 Reactivate User-Dog Assignment	K9User, AuditLog
8.10 Deactivate User-Dog Assignment	8.10 Deactivate User-Dog Assignment	K9User, AuditLog
8.11 Add K9 Role	8.11 Add K9 Role	K9Role, AuditLog
8.12 Update K9 Role	8.12 Update K9 Role	K9Role, AuditLog
8.13 View K9 Role	8.13 View K9 Role	K9Role
8.14 Reactivate K9 Role	8.14 Reactivate K9 Role	K9Role, AuditLog
8.15 Deactivate K9 Role	8.15 Deactivate K9 Role	K9Role, AuditLog
8.16 View K9 Ancestry	8.16 View K9 Ancestry	K9

Table 272: K9 Subsystem Validation

2.13.9 Rhino Subsystem

User Subsystem		
Requirement	Use Case	Entities
9.1 Add Rhino	9.1 Add Rhino	Animal, Rhino, AuditLog

User Subsystem		
Requirement	Use Case	Entities
9.2 View Rhino	9.2 View Rhino	Animal, Rhino
9.3 View Rhinos	9.3 View Rhinos	Animal, Rhino
9.4 Update Rhino	9.4 Update Rhino	Animal, Rhino, AuditLog
9.5 Deactivate Rhino	9.5 Deactivate Rhino	Animal, Rhino, AuditLog

Table 273: Rhino Subsystem Validation

2.13.10 Elephant Subsystem

Elephant Subsystem		
Requirement	Use Case	Entities
10.1 Add Elephant	10.1 Add Elephant	Animal, Elephant, AuditLog
10.2 View Elephant	10.2 View Elephant	Animal, Elephant
10.3 View Elephants	10.3 View Elephants	Animal, Elephant
10.4 Update Elephant	10.4 Update Elephant	Animal, Elephant, AuditLog
10.5 Deactivate Elephant	10.5 Deactivate Elephant	Animal, Elephant, AuditLog

Table 274: Elephant Subsystem Validation

2.13.11 Message Subsystem

Message Subsystem		
Requirement	Use Case	Entities
1.1 Create Message	1.1 Create Message	User, UserGroup, Group Message
1.2 View Message	1.2 View Message	User, UserGroup, Group Message

Table 275: Message Subsystem Validation

2.13.12 Report Subsystem

Report Subsystem		
Requirement	Use Case	Entities
12.1 Generate Custom Report	12.1 Generate Custom Report	Report, ReportConfiguration
12.2 View Custom Report	12.2 View Custom Report	Report, ReportConfiguration
12.3 Update Custom Report	12.3 Update Custom Report	Report, ReportConfiguration
12.4 Remove Custom Report	12.4 Remove Custom Report	Report, ReportConfiguration
12.5 View Bike Impact Report	12.5 View Bike Impact Report	InventoryTransaction, ProjectImpact
12.6 View Budget Report	12.6 View Budget Report	Budget, ProjectBudget, Expense
12.7 View Rhino Report	12.7 View Rhino Report	Animal, Rhino
12.8 View Elephant Report	12.8 View Elephant Report	Animal, Elephant
12.9 View K9 Report	12.9 View K9 Report	K9, K9User, K9Role

Table 276: Report Subsystem Validation

2.13.13 Machine Learning Subsystem

Machine Learning Subsystem		
Requirement	Use Case	Entities
13.1 View Machine Learning Dashboard	3.1 View Machine Learning Dashboard	-

Table 277: Machine Learning Subsystem Validation

2.13.14 System Subsystem

System Subsystem		
Requirement	Use Case	Entities
14.1 Backup System	14.1 Backup System	Backup
14.2 Restore System	14.2 Restore System	Backup
14.3 View Audit Log	14.3 View Audit Log	AuditLog

Table 278: System Subsystem Validation

2.13.15 Budget Subsystem

Budget Subsystem		
Requirement	Use Case	Entities
15.1 Create Project Budget	15.1 Create Project Budget	ProjectBudget, AuditLog
15.2 View Project Budget	15.2 View Project Budget	ProjectBudget
15.3 View Project Budgets	15.3 View Project Budgets	ProjectBudget
15.4 Update Project Budget	15.4 Update Project Budget	ProjectBudget, AuditLog
15.5 Log An Expense	15.5 Log An Expense	Expense, AuditLog
15.6 Update An Expense	15.6 Update An Expense	Expense, AuditLog
15.7 View Expense	15.7 View Expense	Expense
15.8 Remove Expense	15.8 Remove Expense	Expense, AuditLog

Table 279: Budget Subsystem Validation

2.14 Conclusion

This section of the document provided the logical design for the new and proposed system. It contained an updated functional requirements list, their functional requirements descriptions, logical use case diagram, table relationship diagram (ERD), functional decomposition diagram, logical use case narratives, logical context diagram, logical primitive level diagrams, activity diagrams, validation, and the UML class diagram.

3. Technical Design

3.1 Introduction

This section of the document provides the technical design for the new and proposed system. It contains proposed design principles, example wireframes, technical specifications, and an updated complexity matrix.

3.2 Proposed Design Principles

Code	Element	Text/Content	Formatting
EL1	System Logo	Image	Positioned top-left in navbar; consistent branding
EL2	User Avatar	Image	Circular icon, top-right; clickable for profile/actions
PC1	Primary Colour (Purple)	Colour	Hex: #6C63FF; used for buttons, highlights, progress bars
PC2	Secondary Colour (Green)	Colour	Hex: #4CAF50; used for status indicators (Active)
SC1	Background Colour (Light Grey)	Colour	Hex: #F5F5F5; main background
SC2	Card Background	Colour	Hex: #FFFFFF; rounded corners with subtle shadow
SC3	Text Primary	Colour	Hex: #333333; headings and main content
SC4	Text Secondary	Colour	Hex: #777777; descriptions and metadata
H1	Page Heading	Text	Font: Sans-serif; Bold; Size: 24–28px
H2	Section Heading	Text	Font: Sans-serif; Semi-Bold; Size: 18–20px
PT	Paragraph Text	Text	Font: Sans-serif; Regular; Size: 14–16px
LT	Label Text (Forms)	Text	Font: Sans-serif; Medium; Size: 12–14px
IT	Input Field Text	Text	Font: Sans-serif; Regular; Size: 14px
IFT	Input Field	Component	Rounded corners; light border; focus highlight in primary colour
BTN1	Primary Button (Sign in)	Component	Background: #6C63FF; white text; rounded; hover effect
BTN2	Secondary Button (Social Login)	Component	Light background; border; icon + text aligned left
NAV	Navigation Bar	Component	Fixed top; horizontal layout; icons + labels
NAVC	Navigation Active State	Colour	Highlight with primary colour + subtle background
CB	Card Component (Project Card)	Component	Rounded corners; shadow; padding; structured layout
TAG	Status Tags (Active, Done, etc.)	Component	Pill-shaped; colour-coded (Green = Active)
PB	Progress Bar	Component	Thin bar; primary colour fill; percentage indicator
FIL	Filter Chips (All, Active, etc.)	Component	Rounded pills; selected state uses primary colour
DIV	Divider	Component	Thin horizontal line; light grey

Code	Element	Text/Content	Formatting
LINK	Hyperlinks (e.g., “Forgot password?”)	Text	Primary colour; underline on hover
ICON	Icons	SVG	Consistent style; size ~16–24px; neutral colour
BG1	Login Background Image	Image	Full screen; grayscale; blurred for contrast
MOD	Login Card (Modal-style)	Component	Centred; white background; shadow; padding

Table 280 Design Principles Table

3.3 Example Wireframes

3.3.1 Sign-In

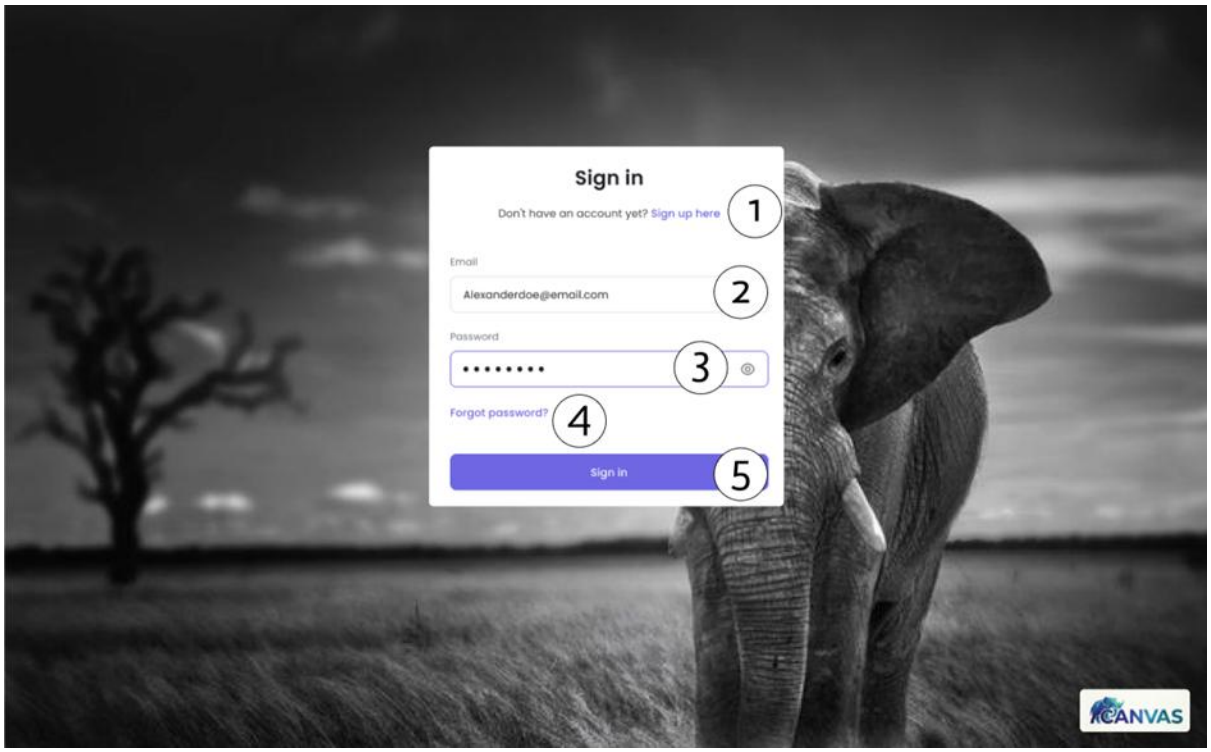


Figure 245 Sign-In Screen

#	Item Name	Item Description	Design Principle
1	Sign up here link	Enables the user to sign up if they do not have an account.	Font: Actor, Regular Size: 14 Colour: FFFFFF
2	Email input field	Enables the user to enter their email.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
3	Password input field	Enables the user to enter their password.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
4	Forgot password link	Enables the user to change their forgotten password.	Font: Actor, Regular Size: 14 Colour: FFFFFF
5	Sign in button	Enables the user to sign in to their account.	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 10

3.3.2 Navbar

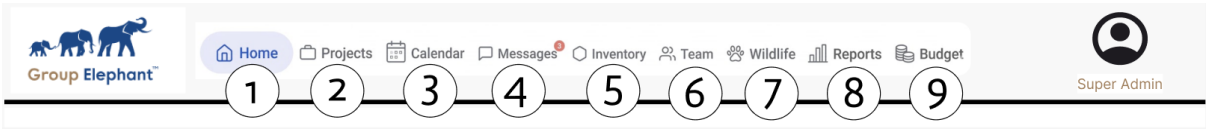


Figure 246: Navbar

#	Item Name	Item Description	Design Principle
1	Home	When selected, the system will redirect the user to the "Home" dashboard.	Font: Actor, Regular Size: 14 Colour: E7E9F7
2	Projects	When selected, the system will redirect the user to the "Projects" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
3	Calendar	When selected, the system will redirect the user to the "Calendar" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
4	Messages	When selected, the system will redirect the user to the "Messages" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
5	Inventory	When selected, the system will redirect the user to the "Inventory" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
6	Team	When selected, the system will redirect the user to the "Team" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
7	Wildlife	When selected, the system will redirect the user to the "Wildlife" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
8	Reports	When selected, the system will redirect the user to the "Reports" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7
9	Budget	When selected, the system will redirect the user to the "Budget" screen.	Font: Actor, Regular Size: 14 Colour: E7E9F7

3.3.3 Rhino CRUD

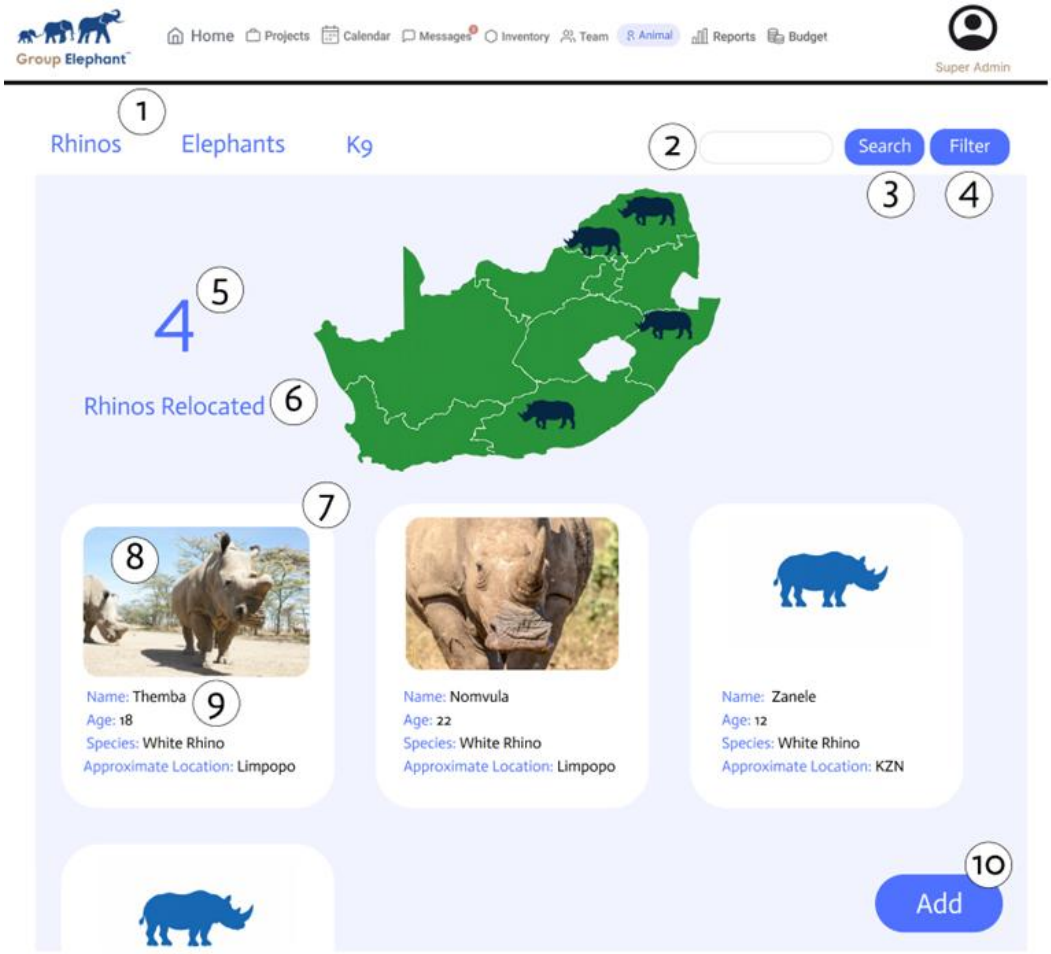


Figure 247 Rhino View Screen

#	Item Name	Item Description	Design Principle
1	Rhinos	Navbar option in the Animal tab.	Font: Actor, Regular Size: 32 Colour: 5071FF
2	Search bar	Enter search criteria (Rhino name).	Corner Radius: 20 Border: D9D9D9
3	Search button	Clicked after entering search criteria.	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 20
4	Filter button	Allows filtering of rhinos based on specific criteria (Species).	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 20
5	Relocated rhinos counter	Counts the number of rhinos relocated	Font: Actor, Regular Size: 96 Colour: 5071FF

6	Rhinos Relocated label	Displays the text “Rhinos Relocated”.	Font: Actor, Regular Size: 32 Colour: 5071FF
7	Rhino card	Displays a rhino with general information and can be clicked to display the “View Rhino” popup.	Corner Radius: 50 Colour: FFFFFF
8	Card Image	Displays the rhino’s image or default image.	Corner Radius: 20 Size: 292/194
9	Card details	Details on the card.	Font: Actor, Regular Size: 20 Colour: 5071FF, 000000
10	Add button	Clicked to add a rhino. Displays the “Add Rhino” form.	Font: Actor, Regular Size: 36 Colour: 5071FF Corner Radius: 40



Figure 248 Add Rhino Form

#	Item Name	Item Description	Design Principle
1	Add header	Describes the form's purpose.	Font: Inter, Regular Size: 40 Colour: FFFFFF
2	Cancel button	When selected, closes the form.	Red circle White "X" inside.
3	Rhino name input field.	Enables the user to enter the rhino's name.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
4	Rhino species input field.	Enables the user to enter the rhino's species.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
5	Rhino weight input field.	Enables the user to enter the rhino's weight.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
6	Rhino age numeric up/down.	Enables the user to enter the rhino's age.	Corner Radius: 10 Border: D9D9D9

			Fill: FFFFFF
7	Rhino gender dropdown.	Enables the user to enter the rhino's gender.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
8	Rhino horn status dropdown.	Enables the user to enter whether the rhino has a horn or not.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
9	Last seen date picker.	Enables the user to select the date when the rhino was last seen.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
10	Identification notes rich textbox.	Enables the user to enter the rhino's unique identification notes.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
11	Rhino picture upload.	Enables the user to upload the rhino's picture.	Picture: 283/225 Text: Inter, regular Colour: 000000 Size: 24
12	Rhino approximate location dropdown.	Enables the user to enter the rhino's approximate location.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
13	Create button	Adds the rhino to the system when clicked.	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 40

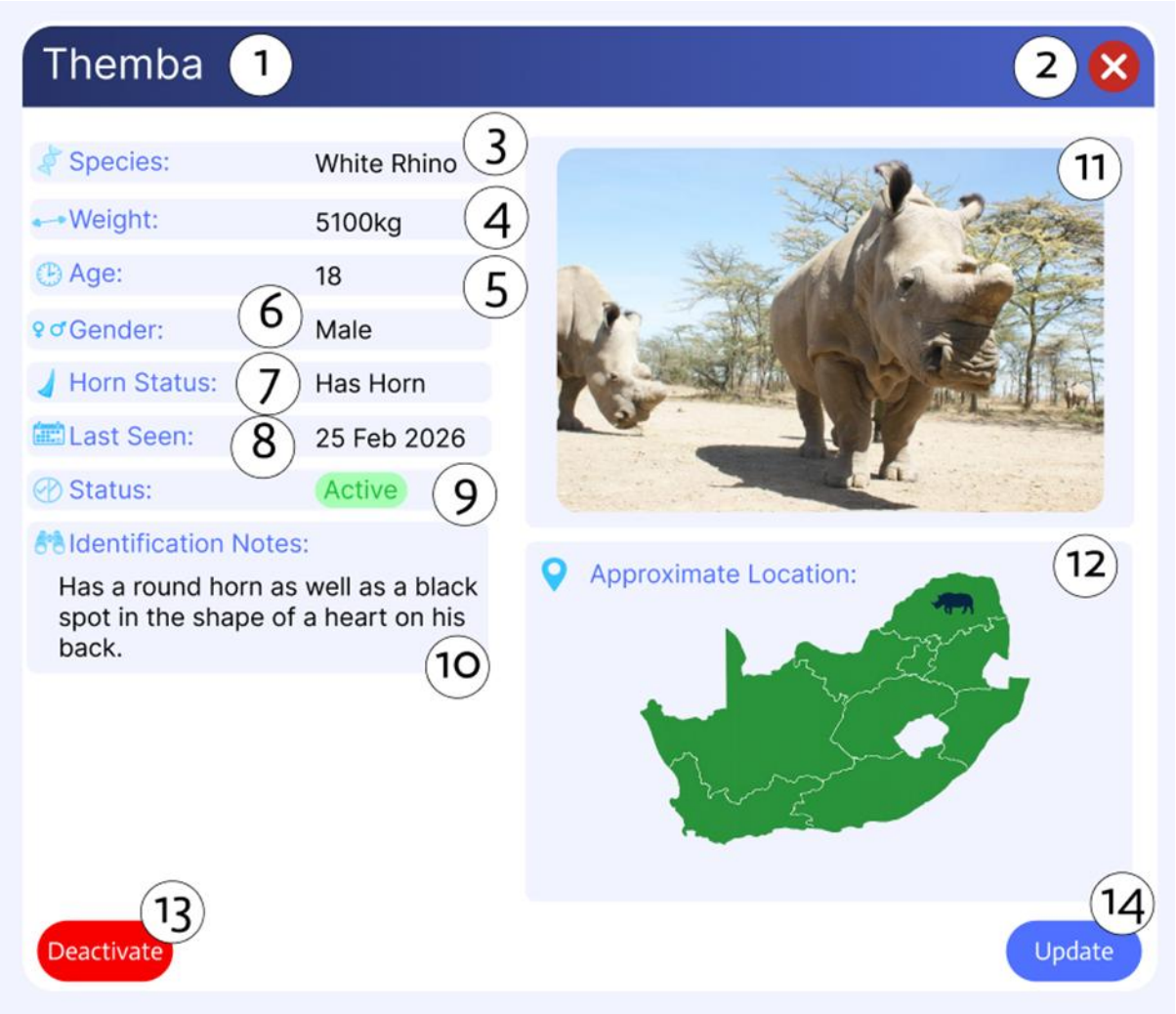


Figure 249 View Rhino Screen

#	Item Name	Item Description	Design Principle
1	Popup header	Indicates the name of the rhino selected.	Font: Inter, Regular Size: 40 Colour: FFFFFF
2	Cancel button	When selected, closes the popup.	Red circle White “X” inside.
3	Rhino species label	Contains rhino’s species.	Font: Inter, Regular Size: 24 Colour: 000000
4	Rhino weight label	Contains rhino’s weight.	Font: Inter, Regular Size: 24 Colour: 000000
5	Rhino age label	Contains rhino’s age.	Font: Inter, Regular Size: 24 Colour: 000000
6	Rhino gender label	Contains rhino’s gender.	Font: Inter, Regular Size: 24

			Colour: 000000
7	Rhino has horn label.	Contains rhino's horn status.	Font: Inter, Regular Size: 24 Colour: 000000
8	Last seen label.	Contains when rhino was last seen.	Font: Inter, Regular Size: 24 Colour: 000000
9	Rhino status label	Contains rhino's status.	Font: Inter, Regular Size: 24 Colour: A8FFB5
10	Rhino identification notes label	Contains rhino's identification notes.	Font: Inter, Regular Size: 24 Colour: 000000
11	Rhino image	Displays rhino's image	Corner radius: 20 Size: 520/346
12	Rhino approximate location map.	Displays rhino's approximate location.	Size: 554/324
13	Deactivate button	Deactivates rhino when clicked.	Font: Actor, Regular Size: 24 Colour: F90000 Corner Radius: 40
14	Update button	Displays the "Update Rhino" form.	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 40

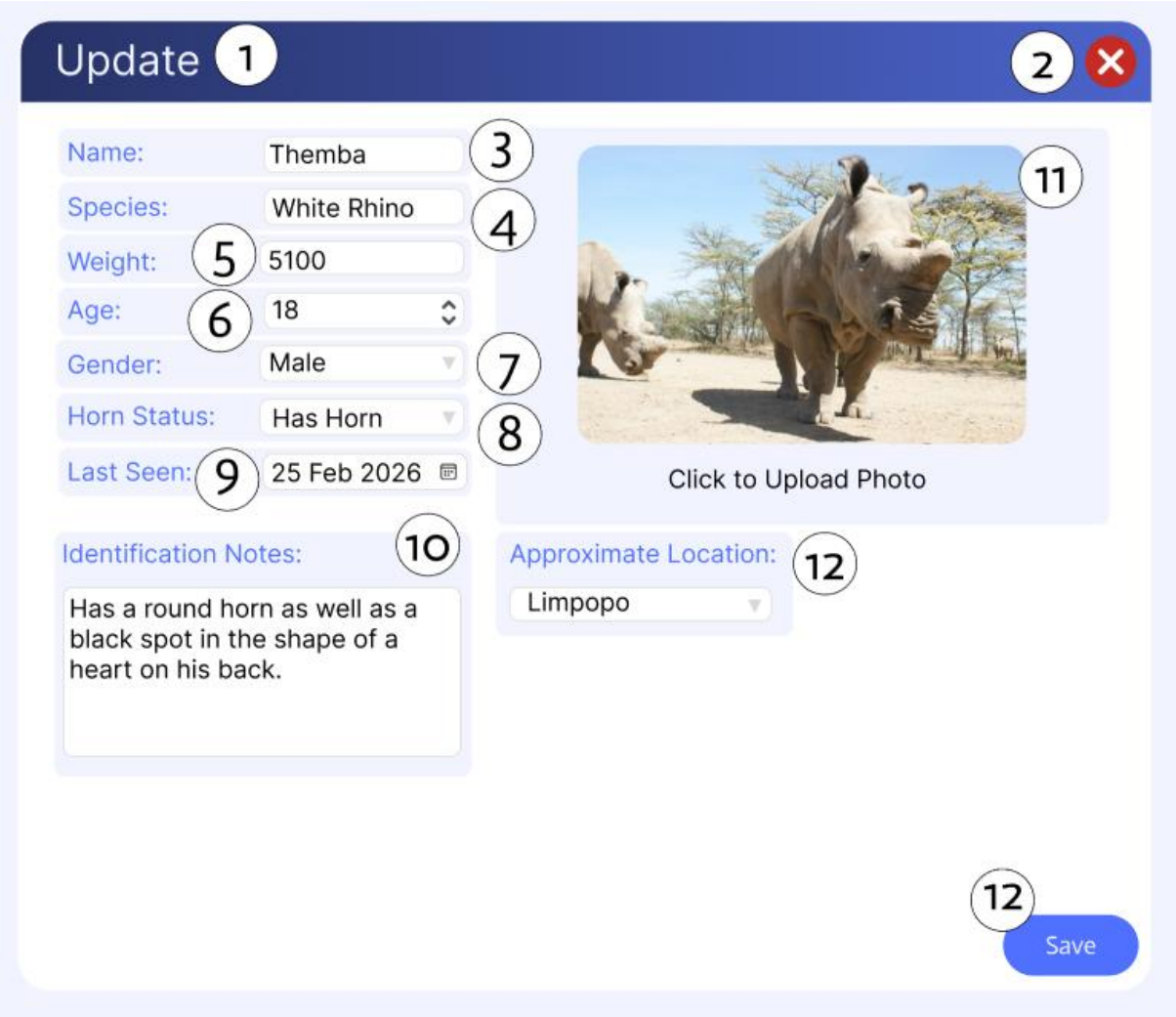


Figure 250 Update Rhino Form

#	Item Name	Item Description	Design Principle
1	Update header	Describes the form's purpose.	Font: Inter, Regular Size: 40 Colour: FFFFFF
2	Cancel button	When selected, closes the form.	Red circle White "X" inside.
3	Rhino name input field.	Enables the user to update the rhino's name.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
4	Rhino species input field.	Enables the user to update the rhino's species.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
5	Rhino weight input field.	Enables the user to update the rhino's weight.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF

6	Rhino age numeric up/down.	Enables the user to update the rhino's age.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
7	Rhino gender dropdown.	Enables the user to update the rhino's gender.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
8	Rhino horn status dropdown.	Enables the user to update whether the rhino has a horn or not.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
9	Last seen date picker.	Enables the user to update the date when the rhino was last seen.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
10	Identification notes rich textbox.	Enables the user to update the rhino's unique identification notes.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
11	Rhino picture.	Enables the user to update the rhino's picture.	Picture: 427/284 Text: Inter, regular Colour: 000000 Size: 24
12	Rhino approximate location dropdown.	Enables the user to update the rhino's approximate location.	Corner Radius: 10 Border: D9D9D9 Fill: FFFFFF
13	Save button	Saves the rhino's updated details to the system when clicked.	Font: Actor, Regular Size: 24 Colour: 5071FF Corner Radius: 40



Figure 251 Success Message

#	Item Name	Item Description	Design Principle
1	Success message	Displays when an action is performed successfully.	Colour: A8FFB5 Corner radius: 80
2	Message label	Contains action details.	Font: Actor, Regular Size: 36 Colour: 20BB37

3.3.4 Report

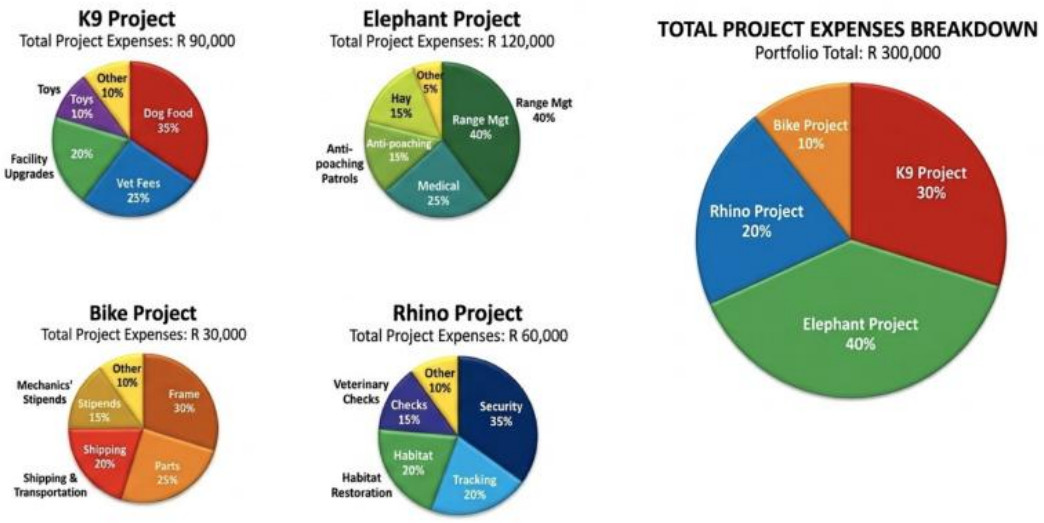


Figure 252 Reports Screen

3.3.5 Messaging

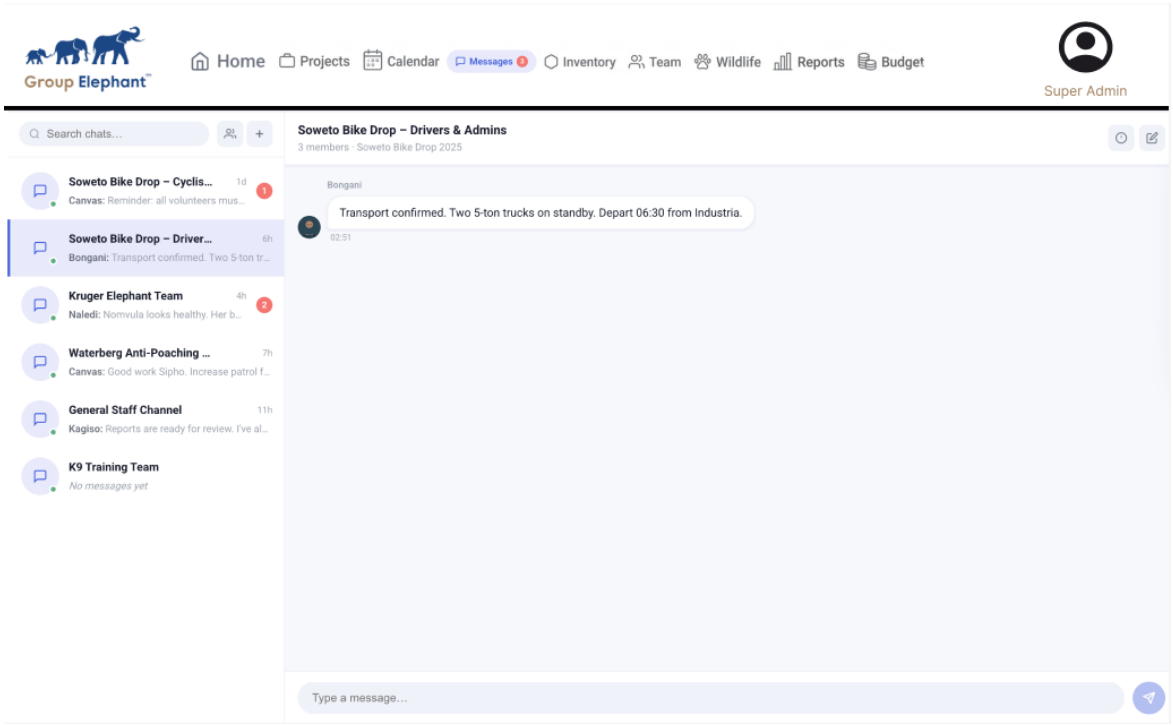


Figure 253 Messaging Screen

3.3.6 Calendar

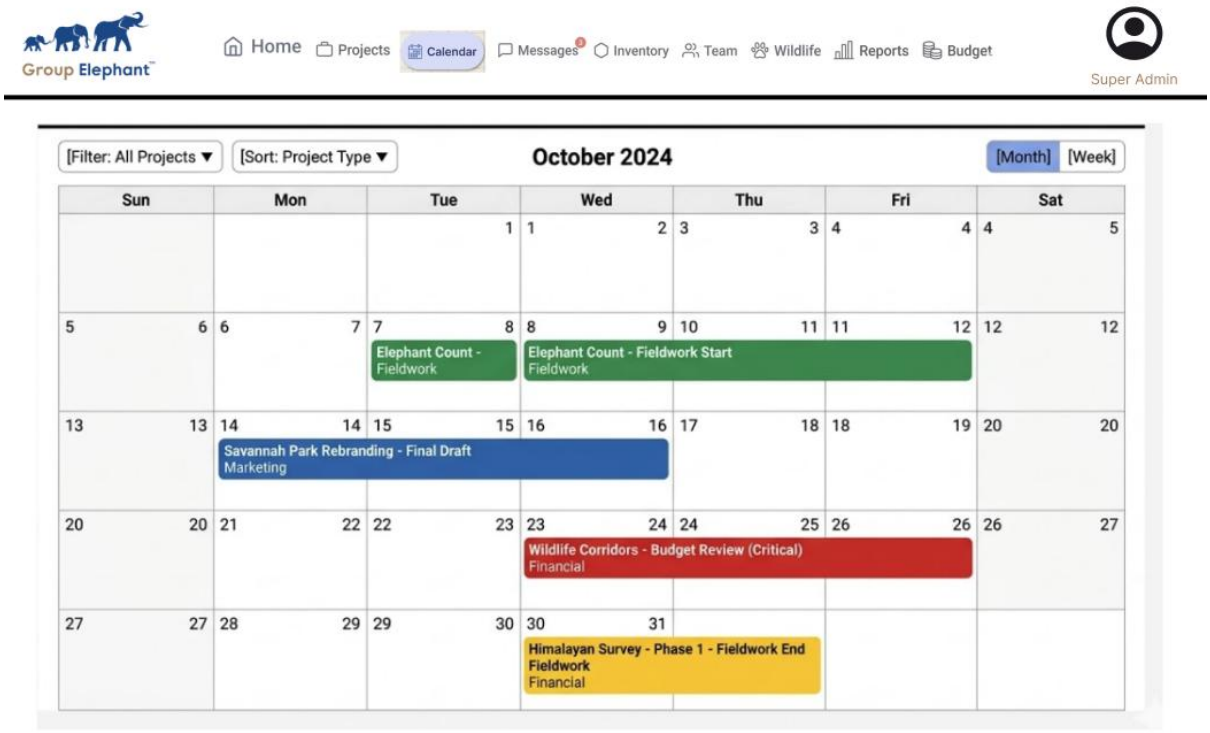


Figure 254 Calendar Screen

3.3.7 Dashboard

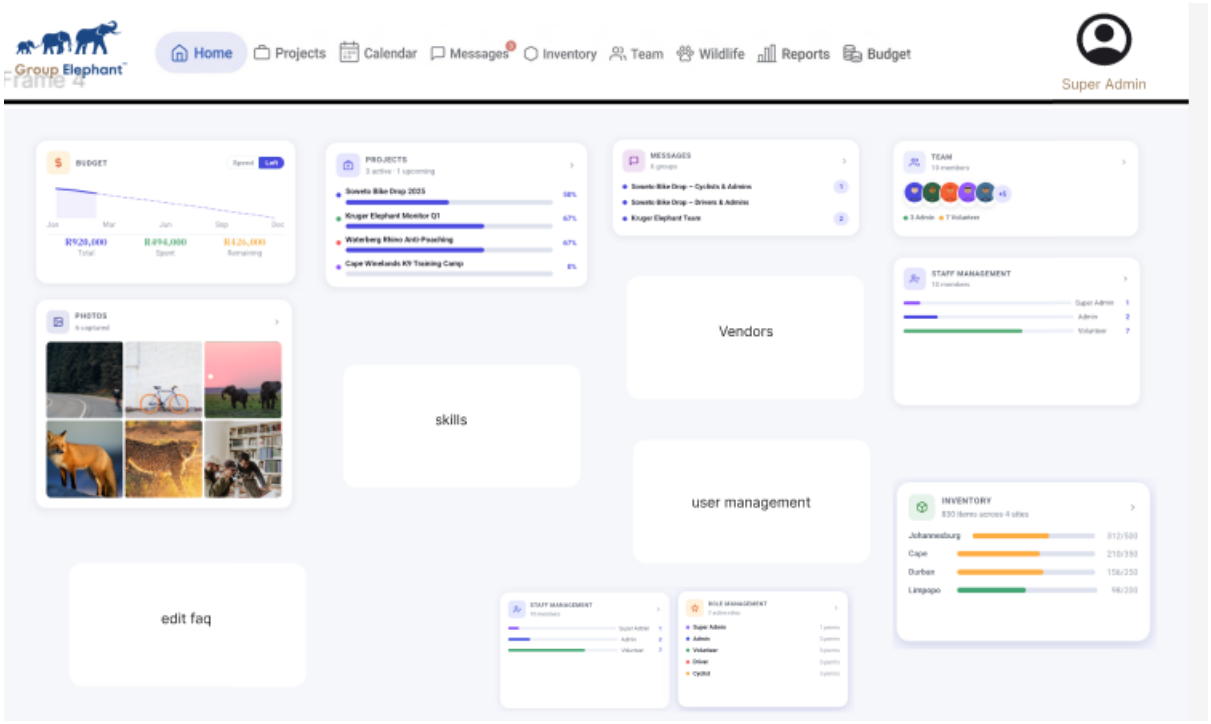


Figure 255 Dashboard Screen

3.4 Technical Specifications

3.4.1 Introduction

In this section we will discuss the technologies we plan to use in our project, why we chose them, and how they interact with one another. Basic outlines on the implementation of select technologies will also be provided.

3.4.2 ASP.NET Core Web API (C#)

ASP.NET Core is a high-performance, open-source, cross-platform framework for building modern, scalable web applications and APIs. We have selected as the foundation for the Integrated NGO Project Portfolio Management System (INPPMS) due to its advanced capabilities and essential features, including:

- **Cross-Platform Compatibility:** ASP.NET Core enables the development of both web-based and mobile-responsive applications, ensuring that Project Managers, Volunteers, and Super Admins can access the system seamlessly across devices and operating systems.
- **Cloud-Enabled Architecture**: The framework supports cloud-based deployment, allowing the system to be hosted on Microsoft Azure. This ensures scalability, enhanced data security, and high availability.
- **High Flexibility and Integration**: ASP.NET Core seamlessly integrates with Angular (frontend), MSSQL (database), JWT (authentication), Heatmap.js (visualisations), and Python-based AI/ML libraries, ensuring a modern, intuitive user experience while maintaining a robust back-end for secure and efficient data management.

Why ASP.NET Core fits into our system

The C# Web API functions as the application layer (business logic tier) in the three-tier architecture proposed for INPPMS. It handles all core functionalities: project creation/tracking, budget & expense management, volunteer/staff assignment (with AI recommendations), notifications/alerts, reporting & analytics (including AI forecasts), and preferred vendor suggestions. The API encapsulates business logic within structured controllers, services, and repositories, ensuring efficient processing of data for the Angular frontend and MSSQL database. The following figure illustrates this architectural design.

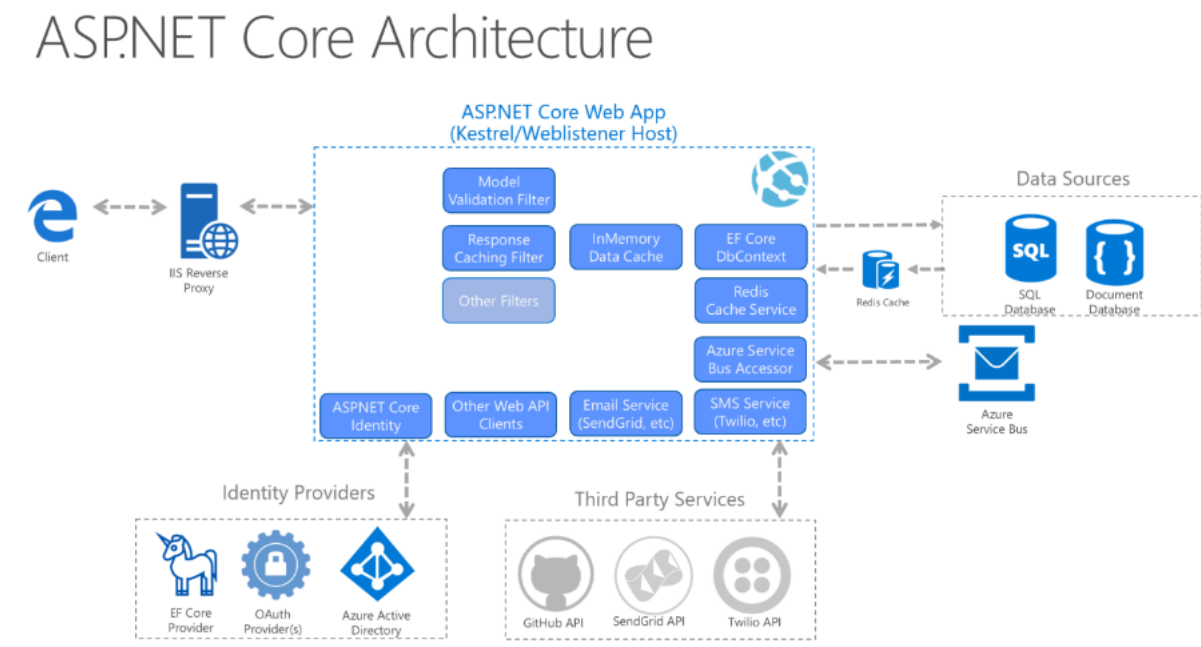


Figure 256 ASP.NET Core Architecture

Interaction with the database

Integrating the Web API with Microsoft SQL Server (MSSQL) is a seamless process. The integration leverages Entity Framework Core to manage database operations efficiently. To enable seamless interaction between the API and the database, the following NuGet packages are required:

- **Microsoft.EntityFrameworkCore:** Provides core ORM (Object-Relational Mapping) functionality for database interactions.
- **Microsoft.EntityFrameworkCore.SqlServer:** Enables SQL Server as the database provider for EF Core.
- **Microsoft.EntityFrameworkCore.Tools:** Provides command-line tools for database management, including migrations and updates.

Database-First Approach

We have adopted a Database-First approach with Entity Framework Core. This means the database schema is designed and created first in Microsoft SQL Server, allowing full control over table structures, relationships, constraints, indexes, and stored procedures before generating the C# entity classes and DbContext.

Integrating the Web API with Microsoft SQL Server (MSSQL) is a seamless process. The integration leverages Entity Framework Core to reverse-engineer the existing database into C# entity models and a DbContext class. Scaffolding can be completed using the GUI interfaces of most IDEs (Integrated Development Environments (such as Visual Studio)

Once scaffolding is complete, the generated DbContext is registered in Program.cs using dependency injection, and the Web API controllers and services can immediately query and manipulate data using LINQ. Any future changes to the database schema can be reflected by re-running the scaffold command.

3.4.3 Angular Framework

Angular is a powerful, component-based TypeScript framework developed by Google for building dynamic, single-page applications (SPAs). It has been chosen as the frontend technology for INPPMS because of its robust structure, two-way data binding, and excellent support for enterprise-scale applications.

How We Will Use Angular in Our Application

Angular will serve as the presentation layer for INPPMS, providing a structured approach to building a dynamic, responsive, and interactive user interface. We will leverage Angular's core features, including:

- **Modules:** Organising the application into reusable and maintainable units (e.g., ProjectModule, BudgetModule, ReportModule).
- **Components:** Building UI elements such as dashboards, project timelines and volunteer assignment interfaces.
- **Routing:** Enabling navigation between authentication screens, dashboard, project management, budget tracking, reporting, and mobile-responsive views.
- **Services:** Managing API calls to the C# Web API for business logic.
- **Pipes & Directives:** Transforming and formatting data dynamically (e.g., date formatting for deadlines, currency for budgets).

Angular Architecture

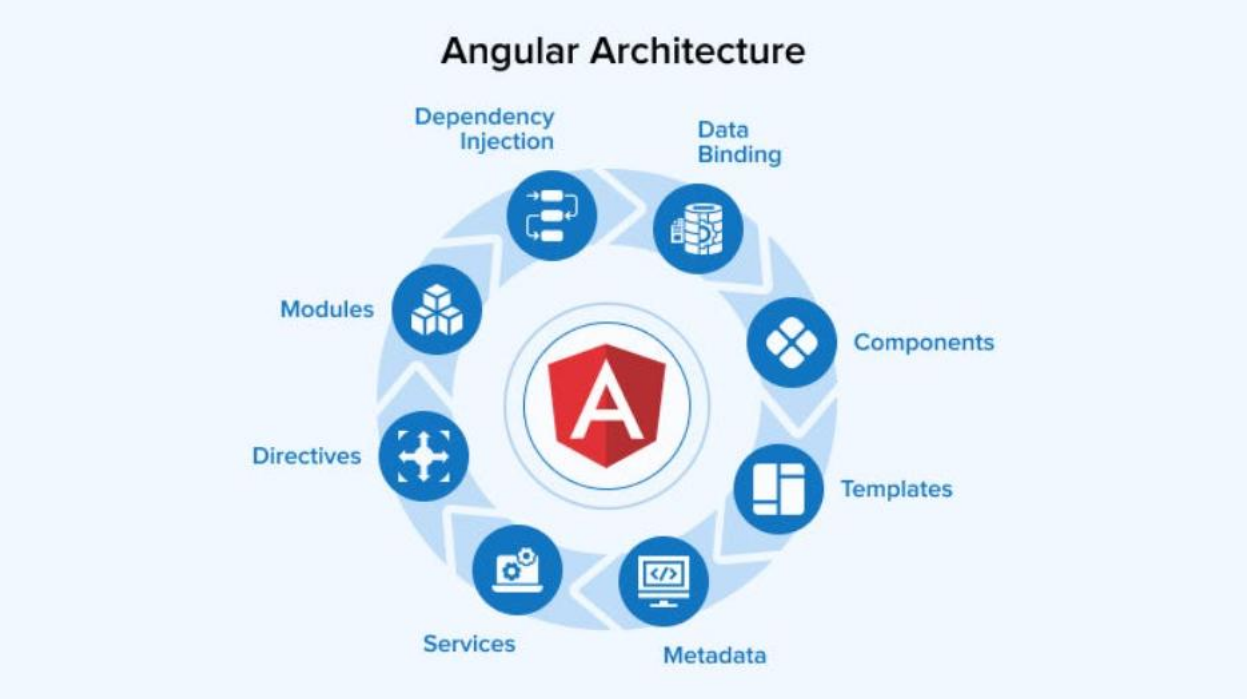


Figure 257 Angular Architecture

Angular will also integrate with Heatmap.js to render interactive South African geographic heatmaps visualising project impact, resource allocation, and initiative density across provinces (e.g., rhino relocation hotspots or bike drop-off areas).

3.4.4 Microsoft SQL Server (MSSQL)

Microsoft SQL Server (MSSQL) is a robust relational database management system designed for high-performance data storage, retrieval, and management. It has been selected as the data layer for INPPMS due to its reliability, security features, and seamless integration with ASP.NET Core.

Advantages of MSSQL

- **High-Speed Query Processing:** Optimised for handling large datasets of projects, budgets, volunteers, and historical data used by AI models.
- **Data Integrity and Consistency:** Enforces constraints (PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK) and ACID compliance.
- **Integration with Other Technologies:** Works natively with C#/.NET, Entity Framework Core, and Azure.

The MSSQL database will store all INPPMS entities. Setup will use SQL Server Management Studio (SSMS) with commands such as CREATE TABLE, ALTER TABLE, and constraints to maintain referential integrity between tables.

Connecting MSSQL to ASP.NET Core Web API

Connection is configured via appsettings.json and Program.cs using EF Core, with connection pooling for performance.

3.4.5 JSON Web Tokens (JWT) for Authentication & Authorisation

JSON Web Tokens (JWT) provide a compact, URL-safe means of securely transmitting information between parties as a JSON object. JWTs are used for authentication and authorisation in INPPMS.

Why JWT fits our system

- Token-based authentication.
- Supports role-based access control.
- Integrates seamlessly with ASP.NET Core middleware for validating tokens on every protected API request.

Implementation

JWTs are generated upon successful login (using user credentials) and include claims for role-based permissions. The Angular frontend stores the token securely and includes it in the header for all subsequent API calls.

3.4.6 Secure Password Hashing

Password security is critical for protecting confidential user data in INPPMS. We will implement industry-standard cryptographic hashing using .NET's built-in libraries.

- **Algorithm**: PBKDF2 (Password-Based Key Derivation Function 2) with HMAC-SHA256
- **Implementation in ASP.NET Core**: Using Microsoft.AspNetCore.Identity.

This ensures credentials are never stored in plain text and remain secure even if the database is compromised.

3.4.7 Heatmap.js for Geographic Visualisations

Heatmap.js is a lightweight, flexible JavaScript library for creating dynamic, interactive heatmaps on web canvases or maps. It has been selected to fulfil the requirement for visual dashboards and geographic insights in INPPMS.

Why Heatmap.js fits our system

- **Project Heatmaps over South African Map:** Visualises density and impact of NGO initiatives across South African provinces.
- **Integration:** Easily embedded in Angular components.
- **Performance:** Lightweight and customisable for large datasets of project locations and metrics.
-

Usage Example

Heatmap.js will consume JSON data from the C# Web API (project coordinates, impact scores, budget status) and render colour-coded heat layers, allowing stakeholders to quickly identify geographic trends.

3.4.8 Microsoft Azure for Cloud Hosting & Deployment

Microsoft Azure is a comprehensive cloud computing platform offering scalable infrastructure, databases, and monitoring services. It has been chosen to meet the deployment and environmental requirements.

Alignment with Our Project

- **Scalability & Flexibility**: Handles growing numbers of projects, volunteers, and AI computations.
- **Global Reach & Security**: South Africa region data centres ensure low latency and compliance with data protection standards.
- **Cost-Effectiveness**: Pay-as-you-go model.

Utilisation in Our Project:

- **Azure App Service**: Hosts the ASP.NET Core Web API.
- **Azure SQL Database**: Managed MSSQL instance for the data layer.

- **Azure Application Insights**: Monitors performance, API usage, and errors across environments.

3.4.9 SendGrid (Email Notification Service)

SendGrid, a Twilio product, is a cloud-based email delivery platform renowned for its high deliverability rates and scalability. It has been selected as the email service for the Integrated NGO Project Portfolio Management System (INPPMS) due to its robust feature set and seamless integration with the .NET technology stack.

Key Reasons for Choosing SendGrid

- Excellent official C# / .NET library with full async support.
- Built-in email templates for consistent branding.
- Event webhook support for tracking email opens, clicks, and bounces.
- Generous free tier.
-

Integration with ASP.NET Core

The SendGrid .NET SDK will be installed via NuGet. A dedicated EmailService class will be created in the application layer to handle all email-related operations. This service will be injected using dependency injection and called from controllers or background services when notifications are triggered.

Example usage in C#:

```
public async Task SendBudgetAlertAsync(string recipientEmail, string
projectName, decimal overrunAmount)
{
   var msg = new SendGridMessage();
   msg.SetFrom("no-reply@ieppms.groupelephant.com", "IEPPMS - ERP");
   msg.AddTo(recipientEmail);
   msg.SetTemplateId("d-template-budget-overrun");
   msg.SetTemplateData(new { ProjectName = projectName,
OverrunAmount = overrunAmount });

   await _sendGridClient.SendEmailAsync(msg);
}
```

3.4.10 scikit-learn (Artificial Intelligence and Predictive Analytics)

scikit-learn is a powerful, open-source Python machine learning library that provides simple and efficient tools for predictive data analysis. It has been chosen as the primary machine learning library for the AI-driven features in INPPMS.

Why scikit-learn is the best fit

The AI requirements are primarily based on historical project data (budgets, expenses, team performance, vendor outcomes) rather than complex deep learning or image processing. scikit-learn excels in this domain because of its:

- Outstanding support for regression models.
- Effective recommendation capabilities.
- Easy model training, evaluation, and deployment on tabular/structured data.

While TensorFlow or PyTorch are available, scikit-learn is significantly more appropriate and lightweight for the type of structured prediction and recommendation tasks required by the project.

AI Features to be Implemented

1. **Budget Overrun Prediction** – Using regression models (Random Forest Regressor / XGBoost) trained on previous project data, current expenses, duration, and other relevant features.
2. **Project Risk Forecasting** – Classification models to predict delay or failure risk and suggest mitigation steps.
3. **Optimal Team Recommendation** – Similarity matching and ranking algorithms based on skills, availability, and past involvement.
4. **Vendor & Catering Suggestions** – Recommendation engine based on historical cost, reliability, project type, and budget constraints.

Integration Strategy

Because the main backend is ASP.NET Core (C#), a hybrid architecture will be used:

- A lightweight Python FastAPI microservice will host the scikit-learn models.
- The C# Web API will make HTTP calls to the Python service endpoints whenever predictions or recommendations are needed.
- Models will be trained on historical project data stored in MSSQL and periodically retrained (When an administrator opens the Machine Learning Dashboard).

Example model training (Python):

```
from sklearn.ensemble import RandomForestRegressor
import pandas as pd

# Load historical project data from MSSQL
df = pd.read_sql("SELECT * FROM ProjectHistory", connection)
X = df[['budget_allocated', 'duration_days', 'team_size',
'risk_factors']]
y = df['actual_overrun']

model = RandomForestRegressor(n_estimators=100)
model.fit(X, y)
# Save model for FastAPI service
import joblib
joblib.dump(model, 'budget_model.pkl')
```

3.4.11 Google Maps Platform API

The Google Maps Platform API (specifically the **Distance Matrix API** and **Directions API**) is a robust, real-time service provided by Google. It has been selected for the INPPMS to enable accurate distance calculations and estimated travel times between project locations, volunteer bases, vendor sites, and resource warehouses across South Africa.

Key Reasons for Choosing Google Maps Platform

- Accurate, real-time distance and travel-time calculations that account for current traffic, road conditions, and South African road networks.
- Good reliability with usage quotas suitable for an NGO-scale system.
- Official .NET client libraries and simple REST endpoints that integrate seamlessly with ASP.NET Core.

How Google Maps API will be used in INPPMS

This service directly supports several core requirements from the project proposal:

- **Project Planning:** Display realistic travel time between multiple initiative locations.
- **Preferred Vendors:** When suggesting venues, accommodation, or catering, factor in travel distance to project locations.
- **Dashboard Visualisation:** Complement the Heatmap.js South African project heatmaps by providing route overlays and travel metrics on interactive maps.
-

Integration with ASP.NET Core Web API

The Google Maps .NET client library will be installed via NuGet. A dedicated Maps class will be created in the application layer and injected via dependency injection. API keys will be securely stored in Azure Key Vault and never exposed to the frontend.

Example usage in C#:

```
public async Task<TravelInfo> GetTravelDetailsAsync(string
originAddress, string destinationAddress)
{
   var request = new DistanceMatrixRequest
   {
       Origins = new[] { originAddress },
       Destinations = new[] { destinationAddress },
       TravelMode = TravelMode.Driving,
       Units = Units.Metric
   };

   var response = await
_googleMapsClient.GetDistanceMatrixAsync(request);

   return new TravelInfo
   {
       DistanceKm = response.Rows[0].Elements[0].Distance.Value /
1000.0,
       DurationMinutes = response.Rows[0].Elements[0].Duration.Value
/ 60.0,
       DurationText = response.Rows[0].Elements[0].Duration.Text
   };
}
```

3.4.12 Conclusion

In this section we explored the technologies we plan to use in our implementation of the INPPM System, including the backend, frontend, database, authorisation technologies and API integrations.

3.5 Complexity Matrix

Description Of Functionality	Score
For online applications: Responsive web design (using HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices.)	3
Appropriate use of tabs/links	3
Appropriate use of grids/tables	3
Use and update at least two graphs in an appropriate business context. For example, a dashboard indicating the status of goods sold, services rendered, staff allocation to work etc. These graphs are NOT the graphs for a report and do not have to be printed in a PDF as per section 3.	4
The storage and display of graphical information, like photos, is for a good business reason. For example, a profile picture or a product image.	3
Working e-mail automatically generated from the database in an appropriate business context	2
At least one use of a tree to display data from the database. This needs to be set up in the database. For example: Province, City, Suburb OR Product category, product type, and product.	3
Able to dynamically modify a data tree structure and in doing so adjusting the data in the database. As per line 19, however, from the front end you need to be able to modify it by adding/updating/deleting the records.	4
At least one use of a calendar view of data (not a date/time picker; not a plug-in such as Google calendar)	3
Uploading a file into the system for appropriate business reasons. Can be PDF or an additional image.	3
The use of audio/video in an appropriate business context. This can be a YouTube video or a help video.	3
At least one use of an administrator in a configurable timer in an appropriate business context. For example, OTP session timer or logout session timer - these timers need to be adjustable from the front end.	3
At least 30 tables used (4 member groups), or 40 tables used (5 member groups)	6
Full referential integrity on all table relationships	6
At least 3 simple list reports in a reporting tool (no control breaks, no graphs, single table) displayed on screen and generated as PDF with logo, date generated, person generated. Reports should be useful to allow organisation to make decisions.	3
At least 2 transactional reports with 2 or more control breaks (with heading and calculated values/totals, multiple tables) displayed on screen and generated as PDF with logo, date generated, person generated. Reports should be useful to allow organisation to make decisions. (have sub totals and full totals)	6
At least 1 report with adjustable criteria displayed on screen and generated as PDF with logo, date generated, person generated. Reports should be useful to allow organisation to make decisions.	3
At least 1 management report using a graph displayed on screen and generated as PDF with logo, date generated, person generated. Reports should be useful to allow organisation to make decisions.	3

Description Of Functionality	Score
All data that can change in future should not be hard coded but maintained in a sub-module of the system (e.g. Lookup tables)	6
Some business rules are not hard coded but maintained in a sub-module of the system and should work when implemented in the system. For example, discount, VAT, response time etc. These need to be in a table that can be updated. It should be applied in the code.	6
All system-generated errors are trapped, and consistent, user-friendly error messages are displayed	6
Appropriate data validation on all input fields. This includes dates (example: cannot be born in the future), numbers (no negative quantities or prices).	6
At least one menu item or other control that opens a complete help document (HTML, PDF, Help-file)	3
Extensive context-sensitive help. E.g. calling Help on a specific screen/function will automatically open the specific help for that screen/function.	6
Search facility on Help	3
Extensive use of hints for example place holders or tool tips.	3
Logon screen with user ID and password and fixed user profiles	3
Applying two factor authentication with applicable business reasons via an OTP not via a link.	3
Encrypted passwords in database	1
Flexible user profiles (i.e. you can dynamically add user profiles that will enable/disable access to certain parts of the system). For example, you can add a new role with new permissions for the role, or you can adjust permissions for an existing role. It does not refer to profiles for example a client vs and employee role.	6
An audit trail of all transactions in the system showing at least date, time, user, transaction type, critical data (such as amount and quantity of transaction). Who did what when on the system. Link the table to the user's table to capture ALL transactions.	6
Able to search the audit trail on any of the following: date, user, transaction type	3
For an online application: Deployment of application to a publicly accessible web server such as Azure.	3
For a mobile application: Deployment to an App Market place (such as the PlayStore or the AppStore)	6
Deployment of the database to a remote database server	3
Able to open Word or Excel and automatically place data in it based on the parameters provided (with a good business reason). For example, importing products and prices into the system.	6
XML or JSON: At least 1 XML or JSON file for Importing or Exporting of data (with good business reason). For example, JSON can be used to interact with another system.	3
Simple Link to an external INPUT device using plug-and-play technology, such as a swipe card reader, bar code reader, etc. or a native component such as a QR reader, a GPS component, etc.	3
Loose Link to an external INPUT device using specific device software. Data or images must seamlessly be stored in the database, but device specific software is visible to the user. (This could include a digital camera, scanner, voice recording device, thump print reader, etc.)	6

Description Of Functionality	Score
Tight Link to an external INPUT device using specific device software. Data or images must seamlessly be stored in the database, but device specific software is not visible to the user. (This could include a digital camera, scanner, voice recording device, thumb print reader, etc.)	9
Integrate an existing web service into your application (with good business reasons. For example, google API or weather forecast (with a good business reason)	3
Appropriate business use of static views on an alternative platform such as mobile application with a good business reason. For example, an about us page on your mobile app.	3
The use of a data layer to facilitate interaction between your database and your business layer	3
Comprehensive use of stored procedures and/or triggers and/or jobs.	3
Complete automated unit testing for five use cases.	5
Any very advanced innovative addition to the system (e.g. machine learning, AI, block chain, text mining, IOT, etc.) If you implement a chatbot, it needs to have advance functionalities that you have developed and trained yourself.	5
Total	174

Table 281 Complexity Matrix

3.6 Conclusion

This section of the document provided the technical design for the new and proposed system. It contained proposed design principles, example wireframes, technical specifications, and an updated complexity matrix.

5. Sign-off by Client

Sign-Off Confirmation:

I, the undersigned, have reviewed the contents of this document and approved it as complete and accurate. I acknowledge that the information provided meets the requirements agreed upon with team 35, Canvas.

Dané Coetzee

Client Name: _____

Signature: _____

05 April 2026

Date: _____

5. Sign-off by Team

Team member:

Name: Alexander Steven

Date: 05 April 2025

Signature: 

Team member:

Name: Amahle Khoza

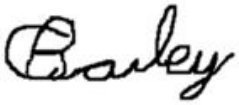
Date: 05 April 2025

Signature: 

Team member:

Name: Christopher Bailey

Date: 05 April 2025

Signature: 

Team member:

Name: Nicholas Bailey

Date: 05 April 2025

Signature: 

Team member:

Name: Stephan Bredenhann

Date: 05 April 2025

Signature:

