

# ALTEAN SYSTEM NAME: PROCION PROCUREMENT SYSTEM



## THE TEAM: TEAM 11



WERNER SCHUTTE

EMAIL: U21797189@TUKS.CO.ZA  
CONTACT NUMBER: 0736179722  
STUDENT NUMBER: U21797189

EMIL WONIGKET

EMAIL: U19022949@TUKS.CO.ZA  
CONTACT NUMBER: 0820601750  
STUDENT NUMBER: U19022949

LEON COMBRINCK

EMAIL: U21514951@TUKS.CO.ZA  
CONTACT NUMBER: 0838430876  
STUDENT NUMBER: U21514951

BUPE CHINDONGO

EMAIL: U21566641@TUKS.CO.ZA  
CONTACT NUMBER: 0658793401  
STUDENT NUMBER: U21566641

JASON VAN DER MERWE

EMAIL: U21502987@TUKS.CO.ZA  
CONTACT NUMBER: 0793731393  
STUDENT NUMBER: U21502987



FROM LEFT TO RIGHT: BUPE CHINDONGO || EMIL WONIGKET || JASON VD MERWE || LEON COMBRINCK || WERNER SCHUTTE

## DESIGN & DEVELOPMENT: ITERATION 5

THIS DOCUMENT OUTLINES THE UPDATED FUNCTIONAL REQUIREMENTS LIST AS WELL AS NUMEROUS TECHNICAL DESIGNS FOR OUR CLIENT (MOYO BUSINESS ADVISORY) SUCH AS TECHNICAL USE CASE DIAGRAMS, TECHNICAL ERD, ACTIVITY, SEQUENCE AND DATAFLOW DIAGRAMS. THE PURPOSE OF THIS ITERATION IS TO IMPLEMENT THE FUNCTIONALITY INTO A PHYSICAL WORKING SOLUTION WHERE WE START TO DEVELOP THE SYSTEM THAT WE WILL BE IMPLEMENTING. IN THIS ITERATION WE WILL BE EXPANDING ON THE FOUNDATION THAT WE HAVE CREATED AND FURTHER IMPLEMENTING FUNCTIONS TO REACH 95% COMPLETION OF THE PROJECT

# MOYO

Driving Significance Together



## Contents

|  |    |
|--|----|
| Table of figures .....                               | 14 |
| 1.DOCUMENT INTRODUCTION .....                        | 23 |
| 2. Updated Functional requirments .....              | 23 |
| 2.1 Introduction .....                               | 23 |
| 2.2 Summary of Updated Functional Requirements ..... | 23 |
| 2.3 Functional requirement lisT .....                | 24 |
| Subsystem 1 – User Subsystem .....                   | 24 |
| Subsystem 2 – Admin Subsystem .....                  | 24 |
| Subsystem 3 – Procurement Subsystem .....            | 25 |
| Subsystem 4 – Finance subsystem .....                | 25 |
| Subsystem 5 – Inventory subsystem .....              | 25 |
| Subsystem 6 – Vendor subsystem.....                  | 25 |
| Subsystem 7 – Report Subsystem.....                  | 26 |
| 2.4 Functional requirements description .....        | 27 |
| Subsystem 1 – User Subsystem .....                   | 27 |
| Subsystem 2 – Admin subsystem .....                  | 31 |
| Subsystem 3 – Procurement Subsystem.....             | 50 |
| Subsystem 4 – Finance Subsystem.....                 | 57 |
| Subsystem 5 – Inventory Subsystem.....               | 63 |
| Subsystem 6 – Vendor Subsystem .....                 | 68 |
| Subsystem 7 – Report Subsystem.....                  | 77 |
| 2.5 Conclusion .....                                 | 82 |
| 2.    Functional requirement Client Sign OFF .....   | 84 |
| 1. Technical Use Case Diagram .....                  | 85 |
| 1.1 Introduction .....                               | 85 |
| 1.2 Use Case Diagrams .....                          | 85 |
| 1.2.1 Subsystem 1 - User Subsystem.....              | 86 |
| 1.2.2 Subsystem 2 – Admin Subsystem.....             | 87 |
| 1.2.3 Subsystem 3 - Procurement Subsystem.....       | 90 |
| 1.2.4 Subsystem 4 - Finance Subsystem.....           | 92 |
| 1.2.5 Subsystem 5 - Inventory Subsystem.....         | 93 |
| 1.2.6 Subsystem 6 - Vendor Subsystem .....           | 94 |
| 1.2.7 Subsystem 7 - Reporting Subsystem .....        | 95 |

|  |                                     |
|--|-------------------------------------|
| 1.3 Conclusion .....                                   | 95                                  |
| 2. Logical Use Case Narratives.....                    | 96                                  |
| 2.1 Introduction .....                                 | 96                                  |
| 2.2 Logical Use Case Narratives.....                   | 96                                  |
| Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35–2.36.....    | 96                                  |
| Use Case 1.7, 2.25-2.30 .....                          | 123                                 |
| Use Case 3.8-12 & 4.1-4.6.....                         | 137                                 |
| Use Case 3.1-3.1? .....                                | <b>Error! Bookmark not defined.</b> |
| Use Case 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4..... | 160                                 |
| Use Case 6.5 & 6.10-6.13.....                          | 201                                 |
| 2.3 Conclusion .....                                   | 236                                 |
| 3. Logical Context Diagram.....                        | 237                                 |
| 3.1 Introduction .....                                 | 237                                 |
| 3.2 Logical Context Diagrams .....                     | 237                                 |
| Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35–2.36.....    | 237                                 |
| Use Case 1.7-1.7 & 2.25-2.26 & 2.27-2.30 .....         | 238                                 |
| Use Case 3.8-4.2 & 4.3-4.10.....                       | 240                                 |
| Use Case 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4.....  | 242                                 |
| Use Case 3.5-3.7 & 6.5 & 6.10-6.13 .....               | 244                                 |
| 3.3 Conclusion .....                                   | 244                                 |
| 4. Screen Design .....                                 | 246                                 |
| 4.1 Introduction .....                                 | 246                                 |
| 4.2 Design principles & Example Wireframes .....       | 246                                 |
| Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35.....         | 246                                 |
| Use Case 2.9-2.16 .....                                | 254                                 |
| Use Case 2.21-2.24 & 4.3-4.6.....                      | 267                                 |
| 2.21 Create Mandate Limit Screen: .....                | 267                                 |
| 2.23 Update Mandate Limit Screen: .....                | 268                                 |
| 2.24 Delete Mandate Limit Screen: .....                | 268                                 |
| .....  | 268                                 |
| Successful Delete Mandate Limit Screen:.....           | 268                                 |
| Use Case 3.5-3.7 .....                                 | <b>Error! Bookmark not defined.</b> |
| 3.5 Approve Procurement Request: .....                 | 291                                 |
| 3.5 Approve Procurement Request: .....                 | 292                                 |
| 3.5 Approve Procurement Request: .....                 | 295                                 |

|   |                              |
|---|------------------------------|
| 4.3 Create Budget Allocation Screen: .....                                | 269                          |
| .....   | 269                          |
| 4.4 View Budget Allocation Screen:.....                                   | 269                          |
| .....   | 269                          |
| 4.5 Edit Budget Allocation Screen: .....                                  | 270                          |
| 4.6 Delete Budget Allocation Screen: .....                                | 270                          |
| .....   | 270                          |
| 4.6 Successful Delete Budget Allocation Screen:.....                      | 271                          |
| .....   | 271                          |
| 4.7 Create Budget Category Screen:.....                                   | 271                          |
| 4.8 View Budget Category Screen: .....                                    | 271                          |
| .....   | 271                          |
| 4.9 Edit Budget Category Screen:.....                                     | 272                          |
| .....   | 272                          |
| 4.10 Delete Budget Category Screen: .....                                 | 272                          |
| .....   | 272                          |
| 4.10 Successful Delete Budget Category Screen:.....                       | 272                          |
| .....   | 272                          |
| <b>Use Case 5.1-5.8 &amp; 5.9, 5.10 &amp; 1.1-1.4 &amp; 3.1-3.4</b> ..... | 273                          |
| 5.2 View Consumables Screen .....   | 274                          |
| <b>Use Case 6.5, 6.10-6.13</b> .....                                      | 291                          |
| 6.5 Approve Onboard Request .....   | 298                          |
| 6.10 Generate Due Diligence.....  | 304                          |
| 6.13 Update Approve Onboard Request .....                                 | 307                          |
| <b>4.3 Conclusion</b> .....   | 311                          |
| <b>5. Technical Use Case Narratives</b> .....                             | 312                          |
| 5.1 Introduction .....  | 312                          |
| 5.2 Technical Use Case Narratives.....                                    | 312                          |
| Use Case 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36 .....                      | 312                          |
| Use Case 1.7 & 2.25-2.30.....   | 354                          |
| Use Case 3.8 – 3.12 & 4.1-4.6.....  | 383                          |
| Use Case 3.5-3.7 .....  | Error! Bookmark not defined. |
| Use Case 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4.....                    | 427                          |
| Use Case 6.5. -6.4 & 6.6-6.9.....   | 542                          |
| <b>5.3 Conclusion</b> .....   | 643                          |

|  |                                     |
|--|-------------------------------------|
| 6. Technical Context Diagram.....                      | 644                                 |
| 6.1 Introduction .....                                 | 644                                 |
| 6.2 Technical Context Diagram.....                     | 644                                 |
| Use Case 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36 .....   | 644                                 |
| Use Case 1.7 & 2.25-2.26 & 2.27-2.30 .....             | 645                                 |
| Use Case 3.8-4.2 & 4.3-4.10.....                       | 646                                 |
| Use Case 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4.....  | 648                                 |
| Use Case 3.5-3.7 & 6.5 & 6.10-6.13 .....               | 650                                 |
| 6.3 Conclusion .....                                   | 650                                 |
| 7. Technical Primitive Level Diagram.....              | 651                                 |
| 7.1 Introduction .....                                 | 651                                 |
| 7.2 Technical Primitive Level Diagram.....             | 651                                 |
| Use Case 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36 .....   | 651                                 |
| Use Case 1.7 & 2.25-2.26 & 2.27-2.30 .....             | 662                                 |
| Use Case 3.8-3.11 & 4.1-4.6.....                       | 667                                 |
| Use Case 3.5-3.7 .....                                 | <b>Error! Bookmark not defined.</b> |
| Use Case 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4..... | 678                                 |
| Use Case 6.5, 6.10-6.13 .....                          | 696                                 |
| 7.3 Conclusion .....                                   | 705                                 |
| 8. Activity Diagram.....                               | 706                                 |
| 8.1 Introduction .....                                 | 706                                 |
| 8.2 Activity Diagram .....                             | 706                                 |
| Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35-2.36 .....   | 706                                 |
| Use Case 1.7 & 2.25-2.26 & 2.27-2.30 .....             | 716                                 |
| Use Case 4.3-4.6 & 4.1-4.2 & 3.8-3.12 .....            | 722                                 |
| Use case 3.5-3.7 .....                                 | <b>Error! Bookmark not defined.</b> |
| Use Case 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4..... | 733                                 |
| Use Case 6.5,6.10-6.13 .....                           | 751                                 |
| 8.3 Conclusion .....                                   | 759                                 |
| 9. Sequence Diagram .....                              | 760                                 |
| 9.1 Introduction .....                                 | 760                                 |
| 9.2 Sequence Diagram .....                             | 760                                 |
| Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35-2.36 .....   | 760                                 |
| Use Case 1.7 & 2.25-2.26 & 2.27-2.30 .....             | 765                                 |
| Use Case 3.8 – 3.12 & 4.1-4.6 .....                    | 769                                 |

|   |                              |
|---|------------------------------|
| Use Case 3.5-3.7 .....                                | Error! Bookmark not defined. |
| Use Case 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4..... | 777                          |
| Use Case 6.5,6.10-6.13 .....                          | 789                          |
| 9.3 Conclusion .....                                  | 792                          |
| 10. Updated Entity Relationship Diagram .....         | 793                          |
| 10.1 Introduction .....                               | 793                          |
| 10.2 Entity relationship Diagram .....                | 794                          |
| 10.3 Conclusion .....                                 | 794                          |
| 11. State Diagram .....                               | 795                          |
| 11.1 Introduction .....                               | 795                          |
| 11.2 State Diagram .....                              | 795                          |
| 11.3 Conclusion .....                                 | 797                          |
| 12.Updated Complexity matrix .....                    | 798                          |
| 12.1 introduction .....                               | 798                          |
| 12.2 complexity matrix .....                          | 798                          |
| 12.3 conclusion .....                                 | 800                          |
| 12. Document Conclusion .....                         | 801                          |
| 13. Team Sign off.....                                | 802                          |
| 14. Client Document Sign OFF .....                    | 803                          |

## Table of Tables

|  |    |
|--|----|
| Table 1: 1.1 Login functional requirement .....                              | 27 |
| Table 2: 1.2 Logout functional requirement.....                              | 27 |
| Table 3:1.3 Forgot password functional requirement .....                     | 28 |
| Table 4: 1.4 Update password functional requirement .....                    | 28 |
| Table 5:1.5 View profile functional requirement .....                        | 29 |
| Table 6: 1.6 Update profile functional requirement .....                     | 29 |
| Table 7:1.7 View user manual functional requirement .....                    | 30 |
| Table 8: 1.9 View Notifications.....   | 30 |
| Table 9:2.1 Create employee functional requirement. ....                     | 31 |
| Table 10:2.2 View employee functional requirement .....                      | 31 |
| Table 11:2.3 Update employee functional requirement.....                     | 32 |
| Table 12:2.4 Delete employee functional requirement.....                     | 33 |
| Table 13:2.5 Create employee roles functional requirement.....               | 33 |
| Table 14:2.6 View employee roles functional requirement.....                 | 34 |
| Table 15:2.7 Update employee roles functional requirement.....               | 34 |
| Table 16:2.8 Delete employee roles functional requirement .....              | 35 |
| Table 17:2.9 Create department functional requirement.....                   | 35 |
| Table 18:2.10 View department functional requirement.....                    | 36 |
| Table 19:2.11 Update department functional requirement .....                 | 36 |
| Table 20:2.12 Delete department functional requirement .....                 | 37 |
| Table 21:2.13 Create Branch functional requirement.....                      | 37 |
| Table 22:2.14 View Branch functional requirement.....                        | 38 |
| Table 23:2.15 Update Branch functional requirement.....                      | 39 |
| Table 24:2.16 Delete Branch functional requirement .....                     | 39 |
| Table 25: 2.17 View Delegation of Authority .....                            | 40 |
| Table 26:2.18 Assigning delegation of authority functional requirement ..... | 40 |
| Table 27:2.19 Accept delegation of authority functional requirement.....     | 41 |
| Table 28: 2.20 Revoke delegation of authority functional requirement .....   | 41 |
| Table 29:2.21 Create mandate limits functional requirement .....             | 42 |
| Table 30:2.22 View mandate limits functional requirement .....               | 42 |
| Table 31:2.23 Update mandate limits functional requirement .....             | 43 |
| Table 32:2.24 Delete mandate limits functional requirement .....             | 44 |
| Table 33:2.25 Backup system data functional requirement .....                | 44 |
| Table 34:2.26 Restore data to system functional requirement .....            | 44 |

|   |    |
|---|----|
| Table 35:2.27 Add help functional requirement.....      | 45 |
| Table 36:2.28 Update help functional requirement .....  | 45 |
| Table 37:2.29 Delete help functional requirement.....   | 46 |
| Table 38:2.30 View all help functional requirement..... | 46 |
| Table 39: 2.31 Create Admin .....                       | 47 |
| Table 40: 2.31 View Admin .....                         | 47 |
| Table 41: 2.33 Update Admin .....                       | 48 |
| Table 42: 2.34 Delete Admin.....                        | 48 |
| Table 43: 3.1 Create Procurement request .....          | 50 |
| Table 44: 3.2 View procurement request. ....            | 50 |
| Table 45: 3.3 Update procurement request.....           | 51 |
| Table 46: 3.4 Delete procurement request.....           | 51 |
| Table 47: 3.5 Approve Procurement Request.....          | 52 |
| Table 49: 3.6 Place procurement request. ....           | 53 |
| Table 50: 3.7 Management mandate approval.....          | 54 |
| Table 51: 3.8 Receive Procurement item.....             | 54 |
| Table 52: 3.9 Upload tax Invoice.....                   | 55 |
| Table 53: 3.10 Budget owner Requisition approval.....   | 56 |
| Table 54: 3.11 Upload receipt.....                      | 56 |
| Table 55: 3.12 Upload credit card Payment .....         | 57 |
| Table 56: 4.1 Finalize procurement requests. ....       | 58 |
| Table 57: 4.2 Upload proof of payment.....              | 58 |
| Table 58: 4.3 Create Budget allocation.....             | 59 |
| Table 59: 4.4 View budget allocation .....              | 60 |
| Table 60: 4.5 Update budget allocation .....            | 60 |
| Table 61: 4.6 Delete budget allocation.....             | 61 |
| Table 62: 4.7 Create Budget Category .....              | 61 |
| Table 63: 4.8 View Budget Category .....                | 62 |
| Table 64: 4.9 Update Budget Category.....               | 62 |
| Table 65: 4.10 Delete Budget Category .....             | 63 |
| Table 66: 5.1 Create inventory item .....               | 63 |
| Table 67: 5.2 View inventory item .....                 | 64 |
| Table 68: 53 Update inventory item .....                | 64 |
| Table 69: 5.4 Delete inventory item .....               | 65 |
| Table 70: 5.5 Create item category .....                | 65 |

|   |     |
|---|-----|
| Table 71: 5.6 View item category.....                               | 66  |
| Table 72: 5.7 Update Item Category .....                            | 66  |
| Table 73: 5.8 Delete Item Category .....                            | 67  |
| Table 74: 5.9 Export inventory details.....                         | 67  |
| Table 75: 5.10 Generate reorder stock notification. ....            | 67  |
| Table 76: 5.10 Complete stock take.....                             | 68  |
| Table 77: 6.1 Create Vendor onboard Request. ....                   | 69  |
| Table 78: 6.2 View Vendor onboard Request .....                     | 69  |
| Table 79: 6.3 Edit Vendor onboard request .....                     | 70  |
| Table 80: 6.4 Delete Vendor onboard Request.....                    | 71  |
| Table 81: 6.5 Approve onboard request.....                          | 72  |
| Table 82: 6.6 Create vendor .....                                   | 73  |
| Table 83: 6.7 View Vendor.....                                      | 74  |
| Table 84: 6.8 Update Vendor .....                                   | 74  |
| Table 85: 6.9 Delete Vendor .....                                   | 75  |
| Table 86: 6.10 Export due diligence vendor checklist .....          | 75  |
| Table 87: 6.11 Generate Sole Supplier Review Notification .....     | 76  |
| Table 88: 6.12 Generate BEE Expiry Certification Notification ..... | 76  |
| Table 89: 7.1 Generate approved vendor report.....                  | 78  |
| Table 90: 7.2 Generate BEE spend report.....                        | 79  |
| Table 91: 7.3 Generate Vendor Spend Report.....                     | 80  |
| Table 92: 7.4 Generate Consumable Inventory management report ..... | 80  |
| Table 93: 7.5 Generate Business Unit allocation report.....         | 81  |
| Table 94: 7.6 Generate Budget Variance Report .....                 | 82  |
| Table 95: UC 7.7 View Generate Report .....                         | 82  |
| Table 96: 2.1 Create Employee .....                                 | 100 |
| Table 97: 2.2 Search Employee.....                                  | 104 |
| Table 98: 2.3 Edit Employee .....                                   | 106 |
| Table 99: 2.4 Delete Employee.....                                  | 109 |
| Table 100: 2.5 Create Employee Roles .....                          | 113 |
| Table 101: 2.6 Search Employee Roles.....                           | 116 |
| Table 102: 2.7 Edit Employee Roles .....                            | 118 |
| Table 103: 2.8 Delete Employee Roles.....                           | 121 |
| Table 104: 2.31 Create Admin .....                                  | 123 |
| Table 105: 1.7 View User Manual .....                               | 124 |

|   |                                     |
|---|-------------------------------------|
| Table 106: 2.25 Backup System Data.....                 | 126                                 |
| Table 107: 2.26 Restore System Data .....               | <b>Error! Bookmark not defined.</b> |
| Table 108: 2.27 Add Help .....                          | 129                                 |
| Table 109: 2.28 Update Help .....                       | 132                                 |
| Table 110: 2.29 Delete Help .....                       | 134                                 |
| Table 111: 2.30 View All Help .....                     | 137                                 |
| Table 112: 3.8 Receive Procurement Item.....            | 139                                 |
| Table 113: 3.9 Upload Tax Invoice .....                 | 141                                 |
| Table 114: 3.10 Budget Owner requisition approval ..... | 143                                 |
| Table 115: 3.10 Upload Receipt.....                     | 145                                 |
| Table 115: 3.11 Upload Credit Card Payment .....        | 146                                 |
| Table 116: 3.5 View Pending Procurement Request .....   | 204                                 |
| Table 117: 4.1 Finalize Procurement Request.....        | 148                                 |
| Table 117: 4.2 Upload Proof of Payment .....            | <b>Error! Bookmark not defined.</b> |
| Table 117: 4.3 Create Budget Allocation .....           | 152                                 |
| Table 118: 4.4 Search Budget Allocation .....           | 155                                 |
| Table 119: 4.5 Update Budget Allocation.....            | 157                                 |
| Table 120: 4.6 Delete Budget Allocation.....            | 160                                 |
| Table 125: 5.1 Create Consumable .....                  | 163                                 |
| Table 126: 5.2 View Consumable .....                    | 165                                 |
| Table 127: 5.3 Update Consumable .....                  | 167                                 |
| Table 128: 5.4 Delete Consumable.....                   | 170                                 |
| Table 129: 5.5 Create Consumable Category.....          | 172                                 |
| Table 130: 5.6 View Consumable Categories .....         | 174                                 |
| Table 131: 5.7 Update Consumable Category .....         | 177                                 |
| Table 132: 5.8 Delete Consumable Category .....         | 179                                 |
| Table 132: 5.9 Export Inventory Details .....           | 181                                 |
| Table 132: 5.10 Complete Stock Take.....                | 183                                 |
| Table 132: 1.1 Login .....                              | 185                                 |
| Table 132: 1.2 Logout .....                             | 186                                 |
| Table 132: 1.3 Forgot Password .....                    | 189                                 |
| Table 132: 1.4 Update Password.....                     | 191                                 |
| Table 132: 3.1 Create Procurement Request.....          | 194                                 |
| Table 132: 3.2 View Procurement Request .....           | 196                                 |
| Table 132: 3.3 Update Procurement Request.....          | 199                                 |

|   |     |
|---|-----|
| Table 132: 3.4 Delete Procurement Request .....                       | 201 |
| Table 133: 6.5 Approve onboard request .....                          | 219 |
| Table 134: 6.10 Export Due Diligence Vendor Checklist .....           | 221 |
| Table 135: 6.11 Generate Sole Supplier Review Notification .....      | 224 |
| Table 136: 6.12 Generate BEE Certificate Expiry Notification .....    | 226 |
| Table 137: Use case 3.5-3.7, 6.5, 6.10-6.13 Logical Context .....     | 244 |
| Table 138: View Pending Procurement Requests.....                     | 291 |
| Table 139: Approve Procurement Request.....                           | 291 |
| Table 140: Successfully Reject Request Notification .....             | 291 |
| Table 141: Successfully Approve Request Notification .....            | 292 |
| Table 142: View Place Procurement Request Screen .....                | 292 |
| Table 143: Place Procurement Details.....                             | 293 |
| Table 144: Place Procurement Details alternatives .....               | 294 |
| Table 145: Successfully Placed Procurement Request Notification ..... | 294 |
| Table 146: View Flagged Procurement Details .....                     | 295 |
| Table 147: Approve Flagged Procurement Details .....                  | 296 |
| Table 148: Approve Flagged Procurement Details Notification.....      | 297 |
| Table 149: Reject Flagged Procurement Details Notification .....      | 297 |
| Table 150:Approve Onboard Request View Screen .....                   | 298 |
| Table 151: Approve Vendor Screen 1 .....                              | 298 |
| Table 152: Approve Vendor Screen 2 Alternative .....                  | 299 |
| Table 153: Approve Vendor - Create screen - Step 1 .....              | 299 |
| Table 154: Approve Vendor - Create screen - Step 2 .....              | 300 |
| Table 155: Approve Vendor - Create screen - Step 3 .....              | 300 |
| Table 156: Approve Vendor - Create screen - Step 4 .....              | 301 |
| Table 157: Approve Vendor - Create screen - Step 5 .....              | 301 |
| Table 158: Approve Vendor - Create screen - Step 6 .....              | 301 |
| Table 159: Approve Vendor - Create screen - Step 7 .....              | 302 |
| Table 160: Approve Vendor - Create screen - Step 8 .....              | 302 |
| Table 161: Approve Vendor - Create screen - Step 9 .....              | 302 |
| Table 162: Approve Vendor - Create screen - Step 10 .....             | 303 |
| Table 163: Created Due Diligence Checklist Success Notification ..... | 303 |
| Table 164: Management Sole Supplier Approval .....                    | 303 |
| Table 165: Successful rejection notification .....                    | 303 |
| Table 166: Successful Approved Onboard Request .....                  | 304 |

|  |                                     |
|--|-------------------------------------|
| Table 167: Generated Due Diligence Checklist .....                           | 304                                 |
| Table 168: Approve Vendor - Update Screen - Step 1.....                      | 307                                 |
| Table 169: Approve Vendor - Update Screen - Step 2.....                      | 307                                 |
| Table 170: Approve Vendor - Update Screen - Step 3.....                      | 307                                 |
| Table 171: Approve Vendor - Update Screen - Step 4.....                      | 308                                 |
| Table 172: Approve Vendor - Update Screen - Step 5.....                      | 308                                 |
| Table 173: Approve Vendor - Update Screen - Step 6.....                      | 308                                 |
| Table 174: Approve Vendor - Update Screen - Step 7.....                      | 309                                 |
| Table 175: Approve Vendor - Update Screen - Step 8.....                      | 310                                 |
| Table 176: Approve Vendor - Update Screen - Step 9.....                      | 310                                 |
| Table 177: Approve Vendor - Update Screen - Step 10.....                     | 310                                 |
| Table 178: Update Successful Notification .....                              | 310                                 |
| Table 179: 3.5 View Pending Procurement Request .....                        | 548                                 |
| Table 180: 3.6 Place Procurement Request .....                               | 563                                 |
| Table 181: View Flagged Procurement Request.....                             | 573                                 |
| Table 182: 6.5 Approve Onboard Request .....                                 | 598                                 |
| Table 183: 6.10 Export Due Diligence Vendor Checklist .....                  | 607                                 |
| Table 184: 6.11 Generate Sole Supplier Review Notification .....             | 609                                 |
| Table 185: 6.12 Generate BEE Certificate Expiry Notification .....           | 611                                 |
| Table 186: 6.13 Update Approved Onboard Request .....                        | 640                                 |
| Table 187: 6.13 Updated Approved Onboard Request .....                       | <b>Error! Bookmark not defined.</b> |
| Table 188: Use case 3.5-3.7, 6.5, 6.10-6.13 Technical Context .....          | 650                                 |
| Table 189: 3.5 Approve Procurement Request.....                              | 696                                 |
| Table 190: 3.6 Place Procurement Request .....                               | 697                                 |
| Table 191: 3.7 View Flagged Procurement Request.....                         | 698                                 |
| Table 191: 4.1 Finalize Procurement Request.....                             | 672                                 |
| Table 191: 4.2 Upload Proof of Payment .....                                 | 673                                 |
| Table 192: 6.1 Approve Onboard Request .....                                 | 699                                 |
| Table 193: 6.10 Generate Due Diligence Checklist .....                       | 700                                 |
| Table 194: 6.11 Generate Sole Supplier Performance Review Notification ..... | 701                                 |
| Table 195: 6.13 Generate BEE Expiry Notification .....                       | 702                                 |
| Table 196: 3.13 Update Approve Vendor Onboard Request .....                  | 703                                 |
| Table 197: 6.5 Approve Onboard Request .....                                 | <b>Error! Bookmark not defined.</b> |
| Table 198: 6.11 Generate Sole Supplier Performance Review Notification ..... | 756                                 |
| Table 199: 6.12 Generate BEE Expiry notification .....                       | 757                                 |

|  |     |
|--|-----|
| Table 200: 6.13 Update Approve Onboard Request .....     | 758 |
| Table 201: 3.5 Approve Procurement Request .....         | 789 |
| Table 202: 3.6 Place Procurement Request .....           | 789 |
| Table 203: 3.7 Approve Flagged Procurement Details ..... | 789 |
| Table 204: 6.5 Approve Onboard Request .....             | 790 |
| Table 205: Vendor State Diagram.....                     | 795 |
| Table 206: Onboard Request State Diagram .....           | 796 |
| Table 207: Procurement Request State Diagram .....       | 797 |
| Table 208: Procurement Details State Diagram.....        | 797 |

## TABLE OF FIGURES

|   |                                     |
|---|-------------------------------------|
| Figure 1 Use Case Diagram - Subsystem 1 .....                 | 86                                  |
| Figure 2 Use Case Diagram - Subsystem 2.....                  | 87                                  |
| Figure 3 Use Case Diagram - Subsystem 2 Continuation 1.....   | 88                                  |
| Figure 4 Use Case Diagram - Subsystem 2 Continuation 2.....   | 89                                  |
| Figure 5 Use Case Diagram - Subsystem 3.....                  | 90                                  |
| Figure 6 Use Case Diagram - Subsystem 3 Continuation.....     | 91                                  |
| Figure 7 Use Case Diagram - Subsystem 4.....                  | 92                                  |
| Figure 8 Use Case Diagram - Subsystem 5.....                  | 93                                  |
| Figure 9 Use Case Diagram - Subsystem 6.....                  | 94                                  |
| Figure 10 Use Case Diagram - Subsystem 7.....                 | 95                                  |
| Figure 11: Use Case 2.1-2.8 & 2.31-2.34 Logical Context.....  | 237                                 |
| Figure 12: Use Case 1.7-1.8 & 2.25-2.30 Logical Context.....  | 238                                 |
| Figure 13: Use Case 3.8-4.2 Logical Context .....             | 240                                 |
| Figure 14: Use Case 4.3-4.10 Logical Context .....            | 241                                 |
| Figure 15: Use Case 5.1-5.8 & 5.9, 5.10.....                  | 242                                 |
| Figure 15: Use Case 1.1-1.4 Logical Context .....             | 243                                 |
| Figure 15: Use Case 3.1-3.4 Logical Context .....             | 243                                 |
| Figure 16: Use Case 6.1-6.4 & 6.6-6.9 Logical Context.....    | <b>Error! Bookmark not defined.</b> |
| Figure 17: View Employee Screen.....                          | 246                                 |
| Figure 18: Create Employee Screen.....                        | <b>Error! Bookmark not defined.</b> |
| Figure 19: Edit Employee Screen .....                         | <b>Error! Bookmark not defined.</b> |
| Figure 20: Delete Employee Confirmation .....                 | 250                                 |
| Figure 21: View Role Screen .....                             | 251                                 |
| Figure 22: Create Role Screen .....                           | 252                                 |
| Figure 23: Edit Employee Roles.....                           | 253                                 |
| Figure 24: Delete Role Conformation.....                      | 253                                 |
| Figure 36: View User Manual Screens.....                      | 256                                 |
| Figure 38: Settings dropdown menu for Backup and Restore..... | 257                                 |
| Figure 39: Backup Screen.....                                 | 257                                 |
| Figure 39: Backup Loading Screen.....                         | 258                                 |
| Figure 39: Backup Successful message .....                    | 258                                 |
| Figure 39: Backup Screen.....                                 | <b>Error! Bookmark not defined.</b> |
| Figure 39: Restore Successful message .....                   | <b>Error! Bookmark not defined.</b> |

|   |                                     |
|---|-------------------------------------|
| Figure 40: View All Help Screen .....                       | 258                                 |
| Figure 41: Add Help Screen.....                             | 259                                 |
| Figure 42: Add Help Loading Screen .....                    | 260                                 |
| Figure 43: Add help Successful message .....                | 260                                 |
| Figure 44: Edit Help Screen .....                           | 261                                 |
| Figure 45: Edit Help Loading screen.....                    | 262                                 |
| Figure 46: Edit Help successful Message .....               | 262                                 |
| Figure 47: Delete Help Screen.....                          | 263                                 |
| Figure 48: Delete help successful message.....              | 264                                 |
| Figure 49: Validation Message for Create Branch .....       | <b>Error! Bookmark not defined.</b> |
| Figure 50: View All Help as Admin Screen .....              | 264                                 |
| Figure 50: View All Help as user Screen.....                | 265                                 |
| Figure 51: Help Video Screen.....                           | 266                                 |
| Figure 52: Help User Manual screen .....                    | 266                                 |
| Figure 56: Create Mandate Limit Screen .....                | 267                                 |
| Figure 57: Update Mandate Limit Screen .....                | 268                                 |
| Figure 58: Delete Mandate Limit Screen.....                 | 268                                 |
| Figure 59: Successful Delete Mandate Limit Screen .....     | 268                                 |
| Figure 60: Create Budget Allocation Screen .....            | 269                                 |
| Figure 61: View Budget Allocation Screen.....               | 269                                 |
| Figure 62: Edit Budget Allocation Screen .....              | 270                                 |
| Figure 63: Delete Budget Allocation Screen .....            | 270                                 |
| Figure 64: Successful Delete Budget Allocation Screen ..... | 271                                 |
| Figure 65 Create Budget Category Screen.....                | 271                                 |
| Figure 66 View Budget Category Screen.....                  | 271                                 |
| Figure 67 Edit Budget Category Screen .....                 | 272                                 |
| Figure 68 Delete Budget Category Screen .....               | 272                                 |
| Figure 69 Successful Delete Budget Category Screen .....    | 272                                 |
| Figure 70 Create Consumable Screen.....                     | 273                                 |
| Figure 71 Successful Create Notification .....              | 273                                 |
| Figure 72 Unsuccessful Create Notification .....            | 273                                 |
| Figure 73 View Consumables Screen.....                      | 274                                 |
| Figure 74 Update Consumable Screen.....                     | 274                                 |
| Figure 75 No Change Made Update Notification.....           | 275                                 |
| Figure 76 Successful Update Notification .....              | 275                                 |

|  |     |
|--|-----|
| Figure 77 Delete Consumable Dialog .....                           | 275 |
| Figure 78 Delete Notification.....                                 | 275 |
| Figure 79 Create Category Screen .....                             | 276 |
| Figure 80 Unsuccessful Create Notification .....                   | 276 |
| Figure 81 Success Create Notification.....                         | 276 |
| Figure 82 View Consumable Categories Screen .....                  | 277 |
| Figure 83 Update Consumable Category Screen .....                  | 277 |
| Figure 84 No Changes Made Notification .....                       | 278 |
| Figure 85 Successful Update Notification .....                     | 278 |
| Figure 86 Delete Category Prompt .....                             | 278 |
| Figure 87 Successful Delete Notification .....                     | 278 |
| Figure 88 Association Found Notification.....                      | 279 |
| Figure 88 Update Stock Screen.....                                 | 279 |
| Figure 88 Update Stock Chart.....                                  | 280 |
| Figure 88 View Consumable Screen with Export Details Button.....   | 281 |
| Figure 88 Export Details PDF .....                                 | 282 |
| Figure 88 Login Screen.....  | 283 |
| Figure 88 Login Failed Notification.....                           | 283 |
| Figure 88 Logout Screen .....                                      | 284 |
| Figure 88 Forgot Password Screen .....                             | 284 |
| Figure 88 Forgot Password failed Notification.....                 | 284 |
| Figure 88 Update Password Screen .....                             | 285 |
| Figure 88 Update Password Success Notification .....               | 285 |
| Figure 88 Update Password Success Notification .....               | 285 |
| Figure 88 Create procurement Request screen.....                   | 286 |
| Figure 88 Create procurement Request Approved Vendor screen .....  | 286 |
| Figure 88 Create procurement Request Other screen .....            | 287 |
| Figure 88 Create procurement Request Success Notification.....     | 287 |
| Figure 88 View procurement Request Screen .....                    | 288 |
| Figure 88 View procurement Request With table dropdown Screen..... | 288 |
| Figure 88 Update procurement Request Screen.....                   | 289 |
| Figure 88 Update procurement Request success Notification .....    | 289 |
| Figure 88 Delete procurement Request Screen.....                   | 289 |
| Figure 88 Delete procurement Request Success Notification .....    | 290 |
| Figure 130 1.7 View User Manual.....                               | 355 |

|   |                                     |
|---|-------------------------------------|
| Figure 132 2.25 Backup System Data .....                | 358                                 |
| Figure 133 2.26 Restore System Data.....                | <b>Error! Bookmark not defined.</b> |
| Figure 134 2.27 Add Help .....                          | 359                                 |
| Figure 135 2.28 Update Help .....                       | 365                                 |
| Figure 136 2.29 Delete Help .....                       | 372                                 |
| Figure 137 2.30 View All Help.....                      | 376                                 |
| Figure 138 3.8 Receive Procurement Item.....            | 387                                 |
| Figure 139 3.9 Upload Tax Invoice .....                 | 390                                 |
| Figure 140 3.10 Budget Owner Requisition Approval ..... | 395                                 |
| Figure 141 3.11 Upload Receipt .....                    | 398                                 |
| Figure 141 3.12 Upload Credit Card Payment .....        | 402                                 |
| Figure 141 4.1 Finalize Procurement Request.....        | 407                                 |
| Figure 141 4.2 Upload Proof of Payment.....             | <b>Error! Bookmark not defined.</b> |
| Figure 142 4.3 Create Budget Allocation .....           | 412                                 |
| Figure 143 4.4 View Budget Allocation .....             | 416                                 |
| Figure 144 4.5 Update Budget Allocation .....           | 423                                 |
| Figure 145 4.6 Delete Budget Allocation.....            | 427                                 |
| Figure 150 5.1 Create Consumable .....                  | 434                                 |
| Figure 151 5.2 View Consumables .....                   | 439                                 |
| Figure 152 Update Consumable .....                      | 448                                 |
| Figure 153 5.4 Delete Consumable .....                  | 452                                 |
| Figure 154 5.5 Create Consumable Category.....          | 458                                 |
| Figure 155 5.6 View Consumable Category.....            | 462                                 |
| Figure 156 5.7 Update Consumable Category.....          | 469                                 |
| Figure 157 5.8 Delete Consumable Category .....         | 474                                 |
| Figure 157 5.9 Delete Export Inventory Details.....     | 476                                 |
| Figure 157 5.10.Complete stock take .....               | 479                                 |
| Figure 157 1.1 Login .....                              | 484                                 |
| Figure 157 1.2 Logout.....                              | 490                                 |
| Figure 157 1.3 Forgot Password.....                     | 492                                 |
| Figure 157 1.4 Update Password.....                     | 499                                 |
| Figure 157 3.1 Create Procurement Request .....         | 505                                 |
| Figure 157 3.2 View Procurement Request .....           | 519                                 |
| Figure 157 3.3 Update Procurement Request.....          | 524                                 |
| Figure 157 3.4 Delete Procurement Request.....          | 537                                 |

|   |                                     |
|---|-------------------------------------|
| Figure 166 Use Case 2.1-2.8 & 2.31-2.34 Tech Context .....            | 644                                 |
| Figure 167 Use Case 1.7-1.8 & 2.25-2.26 & 2.27-2.30 Tech Context..... | 645                                 |
| Figure 168 3.8-4.2 Tech Context .....                                 | 646                                 |
| Figure 169 4.3-4.10 Tech Context .....                                | 647                                 |
| Figure 170 Use Case 5.1 - 5.8 & 5.9, 5.10 Tech Context .....          | 648                                 |
| Figure 170 Use Case 1.1-1.4 Tech Context.....                         | 649                                 |
| Figure 170 Use Case 3.1-3.4 Tech Context.....                         | 649                                 |
| Figure 171 Use Case 6.1 - 6.4 & 6.6 - 6.9 Tech Context .....          | <b>Error! Bookmark not defined.</b> |
| Figure 172 2.1 Create Employee Tech Prim .....                        | 652                                 |
| Figure 173 2.2 View Employee Tech Prim .....                          | 653                                 |
| Figure 174 2.3 Edit Employee Tech Prim .....                          | 654                                 |
| Figure 175 2.4 Delete Employee Tech Prim .....                        | 655                                 |
| Figure 176 2.5 Create Role Tech Prim .....                            | 656                                 |
| Figure 177 View Role Tech Prim .....                                  | 657                                 |
| Figure 178 2.7 Edit Employee Role Tech Prim .....                     | 658                                 |
| Figure 179 2.8 Delete Role Tech Prim.....                             | 659                                 |
| Figure 180 2.31 Create Admin Tech Prim.....                           | 660                                 |
| Figure 181 2.32 View Admin Tech Prim.....                             | 661                                 |
| Figure 184 1.7 View User Manual Tech Prim.....                        | 662                                 |
| Figure 186 2.25 Backup System Data Tech Prim .....                    | 662                                 |
| Figure 187 2.26 Restore System Data Tech Prim.....                    | <b>Error! Bookmark not defined.</b> |
| Figure 188 2.27 Add Help Tech Prim.....                               | 663                                 |
| Figure 189 2.28 Update Help Tech Prim.....                            | 664                                 |
| Figure 190 2.29 Delete Help Tech Prim.....                            | 665                                 |
| Figure 191 2.30 View All Help Tech Prim .....                         | 666                                 |
| Figure 192 3.8 Receive Procurement Item Tech Prim .....               | 667                                 |
| Figure 193 3.9 Upload Tax Invoice Tech Prim .....                     | 668                                 |
| Figure 194 3.10 Budget Owner Requisition Approval Tech Prim .....     | 668                                 |
| Figure 195 3.11 Upload Receipt Tech Prim .....                        | 669                                 |
| Figure 196 4.3 Create Budget Allocation Tech Prim .....               | 674                                 |
| Figure 197 4.4 View Budget Allocation Tech Prim .....                 | 675                                 |
| Figure 198 4.5 Update Budget Allocation Tech Prim .....               | 676                                 |
| Figure 199 4.6 Delete Budget Allocation Tech Prim .....               | 677                                 |
| Figure 204 5.1 Create Consumable Tech Prim.....                       | 678                                 |
| Figure 205 5.2 View Consumables Tech Prim .....                       | 679                                 |

|  |                                     |
|--|-------------------------------------|
| Figure 206 5.3 Update Consumable Tech Prim .....               | 680                                 |
| Figure 207 5.4 Delete Consumable Tech Prim .....               | 681                                 |
| Figure 208 5.5 Create Consumable Category Tech Prim .....      | 682                                 |
| Figure 209 5.6 View Consumable Categories Tech Prim .....      | 683                                 |
| Figure 210 5.7 Update Consumable Category Tech Prim .....      | 684                                 |
| Figure 211 5.8 Delete Consumable Category Tech Prim.....       | 685                                 |
| Figure 211 5.9 Export Inventory Details Tech Prim.....         | 686                                 |
| Figure 211 5.10 Update Stock Tech Prim .....                   | 687                                 |
| Figure 211 1.1 Login Tech Prim.....                            | 688                                 |
| Figure 211 1.2 Logout Tech Prim.....                           | 689                                 |
| Figure 211 1.3 Forgot Password Tech Prim .....                 | 690                                 |
| Figure 211 1.4 Update Password Tech Prim .....                 | 691                                 |
| Figure 211 3.1 Create Procurement Request Tech Prim .....      | 692                                 |
| Figure 211 3.2 View Procurement Request Tech Prim .....        | 693                                 |
| Figure 211 3.3 Update Procurement Request Tech Prim .....      | 694                                 |
| Figure 211 3.4 Delete Procurement Request Tech Prim .....      | 695                                 |
| Figure 219 6.9 Delete Vendor Tech Prim.....                    | 705                                 |
| Figure 220 2.1 Create Employee Activity Diagram .....          | 707                                 |
| Figure 221 2.2 View Employee Activity Diagram .....            | 708                                 |
| Figure 222 2.3 Edit Employee Activity Diagram.....             | 709                                 |
| Figure 223 2.4 Delete Employee Activity Diagram.....           | 710                                 |
| Figure 224 2.5 Create Role Activity Diagram.....               | 711                                 |
| Figure 225 2.6 View Role Activity Diagram.....                 | 712                                 |
| Figure 226 2.7 Edit Role Activity Diagram .....                | 713                                 |
| Figure 227 2.8 Delete Role Activity Diagram .....              | 714                                 |
| Figure 228 2.31 Create Admin Activity Diagram .....            | 715                                 |
| Figure 232 1.7 View User Manual Activity Diagram .....         | 716                                 |
| Figure 234 2.25 Backup System Data Activity Diagram.....       | 717                                 |
| Figure 235 2.26 Restore System Data Activity Diagram .....     | <b>Error! Bookmark not defined.</b> |
| Figure 236 2.27 Add Help Activity Diagram .....                | 718                                 |
| Figure 237 2.28 Update Help Activity Diagram .....             | 719                                 |
| Figure 238 2.29 Delete Help Activity Diagram .....             | 720                                 |
| Figure 239 2.30 View All Help Activity Diagram .....           | 721                                 |
| Figure 244 4.3 Create Budget allocation Activity Diagram ..... | 722                                 |
| Figure 245 4.4 View Budget Allocation Activity Diagram .....   | 723                                 |

|   |     |
|---|-----|
| Figure 246 4.5 Update Budget Allocation Activity Diagram.....           | 724 |
| Figure 247 4.6 Delete Budget Allocation Activity Diagram .....          | 725 |
| Figure 247 4.1 Finalize Procurement Request Activity Diagram .....      | 726 |
| Figure 247 4.2 Upload Proof Of Payment Activity Diagram .....           | 727 |
| Figure 247 3.8 Receive Procurement Item Activity Diagram.....           | 728 |
| Figure 247 3.9 Upload Tax Invoice Activity Diagram .....                | 729 |
| Figure 247 3.10 budget Owner Requisition Approval Activity Diagram..... | 730 |
| Figure 247 3.11 Upload receipt Activity Diagram .....                   | 731 |
| Figure 247 3.12 Upload Credit Card payment Diagram .....                | 732 |
| Figure 252 5.1 Create Consumable Activity Diagram .....                 | 733 |
| Figure 253 5.2 View Consumable Activity Diagram .....                   | 734 |
| Figure 254 5.3 Update Consumable Activity Diagram .....                 | 735 |
| Figure 255 5.4 Delete Consumable Activity Diagram.....                  | 736 |
| Figure 256 5.5 Create Consumable category Activity Diagram.....         | 737 |
| Figure 257 5.6 View Consumable category Activity Diagram.....           | 738 |
| Figure 258 5.7 Update Consumable Category Activity Diagram .....        | 739 |
| Figure 259 5.8 Delete Consumable Category Activity Diagram .....        | 740 |
| Figure 259 5.9. Export Inventory Details Activity Diagram .....         | 741 |
| Figure 259 5.10. Update Stock Activity Diagram.....                     | 742 |
| Figure 259 1.1. Login Activity Diagram .....                            | 743 |
| Figure 259 1.2. Logout Activity Diagram .....                           | 744 |
| Figure 259 1.3. Forgot Password Activity Diagram .....                  | 745 |
| Figure 259 1.4. Update Password Activity Diagram.....                   | 746 |
| Figure 259 3.1. Create Procurement Request Activity Diagram.....        | 747 |
| Figure 259 3.2. View Procurement Request Activity Diagram .....         | 748 |
| Figure 259 3.3. Update Procurement Request Activity Diagram.....        | 749 |
| Figure 259 3.4. Delete Procurement Request Activity Diagram .....       | 750 |
| Figure 268 2.1 Create Employee Sequence Diagram.....                    | 760 |
| Figure 269 2.2 View Employee Sequence Diagram.....                      | 761 |
| Figure 270 2.3 Edit Employee Sequence Diagram .....                     | 761 |
| Figure 271 2.4 Delete Employee Sequence Diagram .....                   | 762 |
| Figure 272 2.5 Create Role Sequence Diagram .....                       | 762 |
| Figure 273 2.6 View Role Sequence Diagram .....                         | 763 |
| Figure 274 2.7 Edit Role Sequence Diagram.....                          | 763 |
| Figure 275 2.8 Delete Role Sequence Diagram.....                        | 764 |

|   |                                     |
|---|-------------------------------------|
| Figure 276 2.31 Create Admin Sequence Diagram .....                     | 764                                 |
| Figure 280 1.7 View User Manual Sequence Diagram .....                  | 765                                 |
| Figure 282 2.25 Backup System data Sequence Diagram .....               | 766                                 |
| Figure 283 2.26 Restore System Data Sequence Diagram .....              | <b>Error! Bookmark not defined.</b> |
| Figure 284 2.27 Add Help Sequence Diagram.....                          | 766                                 |
| Figure 285 2.28 Update Help Sequence Diagram .....                      | 767                                 |
| Figure 286 2.29 Delete Help Sequence Diagram.....                       | 767                                 |
| Figure 287 2.30 View All Help Sequence Diagram .....                    | 768                                 |
| Figure 288 3.8 Receive Procurement Item Sequence Diagram .....          | 769                                 |
| Figure 289 3.9 Upload Tax Invoice Sequence Diagram.....                 | 770                                 |
| Figure 290 3.10 Budget Owner Requisition Approval Sequence Diagram..... | 771                                 |
| Figure 291 3.11 Upload Receipt Sequence Diagram.....                    | 771                                 |
| Figure 291 3.12 Upload Credit Card Payment Sequence Diagram.....        | 772                                 |
| Figure 291 4.1 Finalize Procurement Request Sequence Diagram .....      | 773                                 |
| Figure 291 4.2 Upload Proof Of Payment Sequence Diagram.....            | 773                                 |
| Figure 292 4.3 Create Budget Allocation Sequence Diagram.....           | 774                                 |
| Figure 293 4.4 View Budget Allocation Sequence Diagram.....             | 775                                 |
| Figure 294 4.5 Update Budget Allocation Sequence Diagram .....          | 776                                 |
| Figure 295 4.6 Delete Budget Allocation Sequence Diagram .....          | 776                                 |
| Figure 300 5.1 Create Consumable Sequence Diagram .....                 | 777                                 |
| Figure 301 5.2 View Consumable Sequence Diagram .....                   | 778                                 |
| Figure 302 5.3 Update Consumable Sequence Diagram.....                  | 779                                 |
| Figure 303 5.4 Delete Consumable Sequence Diagram .....                 | 780                                 |
| Figure 304 5.5 Create Consumable Category Sequence Diagram .....        | 780                                 |
| Figure 305 5.6 View Consumable Category Sequence Diagram .....          | 781                                 |
| Figure 306 5.7 Update Consumable Category Sequence Diagram .....        | 782                                 |
| Figure 307 5.8 Delete Consumable Category Sequence Diagram .....        | 783                                 |
| Figure 307 5.9.Export Inventory Details Sequence Diagram .....          | 783                                 |
| Figure 307 5.10 Update Stock Sequence Diagram.....                      | 784                                 |
| Figure 307 1.1 Login Sequence Diagram .....                             | 784                                 |
| Figure 307 1.2 Logout Sequence Diagram .....                            | 785                                 |
| Figure 307 1.3 Forgot Password Sequence Diagram .....                   | 785                                 |
| Figure 307 1.4 Update Password Sequence Diagram .....                   | 786                                 |
| Figure 307 3.1 Create Procurement Request Sequence Diagram .....        | 786                                 |
| Figure 307 3.2 View Procurement Request Sequence Diagram.....           | 787                                 |

**Iteration 5: Design & Development– Team 11**

|  |     |
|--|-----|
| Figure 307 3.3 Update Procurement Request Sequence Diagram ..... | 787 |
| Figure 307 3.4 Delete Procurement Request Sequence Diagram ..... | 788 |
| Figure 316 Technical ERD .....                                   | 794 |

# 1. DOCUMENT INTRODUCTION

In iteration 4 we focus on the Design and the development of the following Use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25-2.26, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system that we are developing. Here we try and gather insight about the updated functional requirements, Logical Use Case Narratives, Logical context diagrams for those narratives, Screen designs for those narratives, Technical Use Case Narrative, Technical Context diagram for those narratives, Technical primitive level diagram for those narratives, Activity diagram for those narratives, Sequence diagram for those narratives, State diagram for those narratives(only complex narratives that has a change in state involved), and Demonstrate Functionality of the use cases, and the updated complexity matrix with the updated ERD to help us develop the basis for the final system This will bring us closer towards our end goal of the final solution of the Procion system.

# 2. UPDATED FUNCTIONAL REQUIREMENTS

## 2.1 INTRODUCTION

In this section a list of the second and last updated functional requirements will be listed for our proposed system Procion. The second and last updated functional requirements are grouped into the following sub-systems: The user sub-system, Admin sub-system, Procurement sub-system, Finance sub-system, Inventory sub-system, Vendor sub-system and a Report sub-system.

## 2.2 SUMMARY OF UPDATED FUNCTIONAL REQUIREMENTS

**With regards to the to our functional requirements changes have been made to the following:**

- 2.18 Assign Delegation of Authority
- 2.10 Generate reorder stock notification (removed)
- 2.19 Edit Delegation of Authority
- 2.20 Revoke Access
- 2.35 Delete Delegation of Authority
- 2.36 Activate Delegation of Authority
- 1.8 View Help (Removed)
- 3.6 Upload requisition Quotes(Removed)
- 1.8 View Notification
- 2.27 Add Help
- 2.28 Update Help
- 2.29 Delete Help

## 2.30 View All Help

## 2.3 FUNCTIONAL REQUIREMENT LIST

## SUBSYSTEM 1 – USER SUBSYSTEM

- 1.1 Login
- 1.2 Logout
- 1.3 Forgot password.
- 1.4 Update password
- 1.5 View profile.
- 1.6 Update profile
- 1.7 View user manual.
- 1.8 View Notification

2.18 Assign Delegation of Authority.

2.19 Edit Delegation of Authority

2.20 Revoke Access

2.21 Create Mandate limits

2.22 View Mandate limits

2.23 Update Mandate limits

2.24 Delete Mandate limits

2.25 Backup system data

2.26 Restore data to system.

2.27 Add Help

2.28 Update Help

2.29 Delete Help

2.30 View all help

2.31 View Admin

2.32 Create Admin

2.33 Edit Admin

2.34 Delete Admin

2.35 Delete Delegation of Authority

2.36 Activate Delegation of Authority

## SUBSYSTEM 2 – ADMIN SUBSYSTEM

- 2.1 Create employee.
- 2.2 View employee
- 2.3 Edit employee
- 2.4 Delete employee.
- 2.5 Create employee roles.
- 2.6 View employee roles
- 2.7 Update employee roles
- 2.8 Edit employee roles.
- 2.9 Create Department
- 2.10 View Department
- 2.11 Edit Department
- 2.12 Delete Department
- 2.13 Create Branch
- 2.14 View Branch
- 2.15 Edit Branch
- 2.16 Delete Branch
- 2.17 View Delegation of Authority

### SUBSYSTEM 3 – PROCUREMENT

#### SUBSYSTEM

- 3.1 Create Procurement Request
- 3.2 View Procurement Request
- 3.3 Update Procurement Request
- 3.4 Delete Procurement Request
- 3.5 Approve Procurement Request
- 3.6 Place procurement details
- 3.7 Approve Flagged Procurement Details
- 3.8 Receive Procurement item
- 3.9 Upload tax Invoice
- 3.10 Budget owner Requisition approval
- 3.11 Upload receipt
- 3.12 Upload credit card Payment
- 3.13 View Procurement Details

### SUBSYSTEM 4 – FINANCE

#### SUBSYSTEM

- 4.1 Finalize procurement request
- 4.3 Create budget allocation.
- 4.4 View budget allocation
- 4.5 Update budget allocation
- 4.6 Delete budget allocation.
- 4.7 Create budget category
- 4.8 View budget category
- 4.9 Update budget category
- 4.10 Delete budget category

### SUBSYSTEM 5 – INVENTORY

#### SUBSYSTEM

- 5.1 Create Consumable
- 5.2 View Consumables
- 5.3 Update Consumable
- 5.4 Delete Consumable
- 5.5 Create Consumable category.
- 5.6 View Consumable categories
- 5.7 Update Consumable category.
- 5.8 Delete Consumable category.
- 5.9 Export inventory details
- 5.10 Complete Stock Take

### SUBSYSTEM 6 – VENDOR

#### SUBSYSTEM

- 6.1 Create Vendor onboard Request
- 6.2 View Vendor onboard Request
- 6.3 Update Vendor Onboard Request
- 6.4 Delete Vendor Onboard Request
- 6.5 Approve onboard request
- 6.6 Create vendor
- 6.7 View vendor
- 6.8 Update Vendor
- 6.9 Delete vendor
- 6.10 Export Due Diligence Vendor Checklist
- 6.11 Generate Sole Supplier Review Notification
- 6.12 Generate BEE Certificate Expiry Notification

**SUBSYSTEM 7 – REPORT****SUBSYSTEM**

7.1 Generate Approved Vendor Report.

7.2 Generate BEE Spend Report.

7.3 Generate Vendor Spend Report

7.4 Generate Consumable Inventory  
Management Report.

7.5 Generate Business Unit Allocation Report

7.6 Generate Budget Variance Report

7.7 View Generate Report

## 2.4 FUNCTIONAL REQUIREMENTS DESCRIPTION

### SUBSYSTEM 1 – USER SUBSYSTEM

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 1.1   |
| <b>Requirement name (use case name):</b>                 | Login   |
| <b>Requirement short description:</b>                    | The system should allow the user to log on to the system  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Each user will log on to the system with their username and password. The information is then validated using the system's database and will end either by providing user access to what their authority level allows or by displaying an error message indicating the entered details are incorrect.</p> <p><b>Constraint(s):</b></p> <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul> |
| <b>Business rules applicable to this requirement:</b>    | <ul style="list-style-type: none"> <li>• All passwords must be encrypted and hashed.</li> <li>• The administrative must have added user into the system.</li> </ul>   |
| <b>Revision date and Revision number:</b>                | 03/17/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 1: 1.1 Login functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 1.2   |
| <b>Requirement name (use case name):</b>                 | Logout  |
| <b>Requirement short description:</b>                    | The system should allow the user to log out of the system   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user will request to log out of the system. The system will then log the user out and the use case concludes once the user is returned to the login page successfully.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user requires internet access</li> <li>• User must be logged into the system.</li> <li>• The user must be able to log out from any page of the system.</li> </ul>                                  |
| <b>Revision date and revision Number:</b>                | 03/17/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 2: 1.2 Logout functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 1.3  |
| <b>Requirement name (use case name):</b>                 | Forgot password  |
| <b>Requirement short description:</b>                    | The system should allow the user to reset their password if they forgot their password.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user will request that he forgot his password. The system will prompt the user to enter the following details:           <ul style="list-style-type: none"> <li>• Username/ email</li> </ul>           The system will use two factor authentication to send a code to the user's email address. The system will prompt the user to enter the security code. Once the security code is entered the user will be prompted to enter a new password. The system will validate that the new password is not the same as the previous password. The system will notify the user that his password has been changed successfully.</p> <p><b>Constraint(s):</b><br/>           None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user requires internet access</li> <li>• User must be logged out of the system.</li> <li>• The user must be registered on the system.</li> <li>• The new password cannot be the same as any previous passwords set by the user.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/17/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 3:1.3 Forgot password functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 1.4   |
| <b>Requirement name (use case name):</b>                 | Update password   |
| <b>Requirement short description:</b>                    | The system should allow the user to reset their password  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user should be able to request a new password should they want to change their password. The user will request to change password in which the system will request their previous password, new password. The previous password will be validated using the system database if the validation is correct the password will be replaced. The user will then be notified of the successful password update.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user requires internet access</li> <li>• The user details must be verified before any changes are made.</li> <li>• The new password cannot be the same as any previous passwords set by the user.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/17/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 4: 1.4 Update password functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 1.5  |
| <b>Requirement name (use case name):</b>                 | View profile   |
| <b>Requirement short description:</b>                    | The system should allow the user to view their own profile.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user requests to view their profile. The system will retrieve the user's details from the database and display the following details:</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Name</li> <li>• Surname</li> <li>• Telephone Number</li> <li>• Email</li> <li>• Branch</li> <li>• Department</li> <li>• Mandate Limit</li> <li>• Profile Picture</li> </ul> <p>The complete user profile will be displayed to the user.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged in to the system</li> <li>• The user requires internet access.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/08/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 5:1.5 View profile functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 1.6  |
| <b>Requirement name (use case name):</b>                 | Edit profile   |
| <b>Requirement short description:</b>                    | Users should be able to view their own profile and edit some parts of their details.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user requests to update their profile. The user will provide the specific details that they wish to update. The system will capture these details and validate the captured data. If the data validation returns a success the system will capture the newly updated details on the database.</p> <p><b>Constraint(s):</b><br/>           Certain details cannot be updated without the proper access level.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged in to the system.</li> <li>• The user requires internet access.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/17/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 6: 1.6 Update profile functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 1.7  |
| <b>Requirement name (use case name):</b>                 | View user manual   |
| <b>Requirement short description:</b>                    | The system should allow users to view the user manual.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user will request to view the user manual. The user will navigate to the user manual using the system. This will provide users with the user manual that will help explain certain parts of the system and how to work with them.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged in to the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/21/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 7:1.7 View user manual functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 1.8   |
| <b>Requirement name (use case name):</b>                 | View Notifications  |
| <b>Requirement short description:</b>                    | The user of the system would like to view any notifications that specifically they might have.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The system would allow the user to view all the notifications that are specifically meant for them, this could be found by the user at the top right of the navbar at the bell icon. When the user clicks on the icon the system will display a screen list of notifications for that user. Once the user clicks on one item of the list the system will navigate them to the appropriate location within the system where they can complete the task of that notification. Example The user clicks on a notification that has to do with procurement status approval then the system will navigate that user to the procurement approval page for them to complete that task.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged in to the system.</li> <li>User must have a notification before they can click on a specific notification.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 04/10/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 8: 1.9 View Notifications

**SUBSYSTEM 2 – ADMIN SUBSYSTEM**

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.1  |
| <b>Requirement name (use case name):</b>                 | Create employee  |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to create a new employee account.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin with the right authority should be able to create a new employee account. Where they will need to provide all the information such as:</p> <ul style="list-style-type: none"> <li>• email address</li> <li>• phone number</li> <li>• name</li> <li>• surname</li> </ul> <p>The system will send the user an email containing a temporary login, allowing them to activate their account. The details are then validated by the system and added to the database.</p> <p><b>Constraint(s):</b><br/>A duplicate account cannot be made.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• All needed details must be provided.</li> <li>• A strong password should be given.</li> <li>• The user requires internet access.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 9:2.1 Create employee functional requirement.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.2  |
| <b>Requirement name (use case name):</b>                 | View employee  |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to view and search all employee accounts on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin with the right authority will request to either search or view all employee accounts on the system. The admin will select the employee account they would like to view. The system will then display all the related details for that account.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin with the right authority must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/18/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 10:2.2 View employee functional requirement

| Functional Requirement   | Explanation  |
|--|--|
| <b>Requirement number:</b>   | 2.3  |
| <b>Requirement name (use case name):</b>   | Edit employee  |
| <b>Requirement short description:</b>  | The system should allow the admin with the right authority to update an employee's account detail.                     |
| <b>Requirement detailed description and constraints:</b> <p><b>Description:</b><br/>The admin with the right authority will request to view all employee accounts on the system. The admin will select the employee account they would like to update. The system will then display all the related details for that account where the admin can then edit any of the details.</p> <ul style="list-style-type: none"> <li>• email address</li> <li>• phone number</li> <li>• name</li> <li>• surname</li> </ul> <p>Once finished the system will validate the new details and either successfully update the details in the system database or it will display an error message regarding the details. The system then notifies the employee that their account has been updated.</p> <p><b>Constraint(s):</b><br/>The admin must “view” the employee account first before they will be able to update it.</p> |  |
| <b>Business rules applicable to this Requirement:</b>  | <ul style="list-style-type: none"> <li>• The admin with the right authority must be logged into the system.</li> </ul> |
| <b>Revision date and revision Number:</b>  | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 11:2.3 Update employee functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.4  |
| <b>Requirement name (use case name):</b>                 | Delete employee  |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to delete an employee account.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin wishes to delete an employee profile. The admin views the profile they wish to delete and requests deletion. The system will validate that the employee can be deleted. If the validation returns a success the system will remove the employee profile from the database.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user with the right authority must be logged into the system.</li> <li>The employee must not be with the company anymore.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/18/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 12:2.4 Delete employee functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.5   |
| <b>Requirement name (use case name):</b>                 | Create employee roles   |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to add new employee roles   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Admin should be able to create new employee roles on the system. The admin will request to create new employee role. The system will request all relevant details</p> <ul style="list-style-type: none"> <li>Name</li> <li>Description</li> </ul> <p>The admin will provide the details and it will then be validated by the system. The system will notify the user if their role has been successfully added.</p> <p><b>Constraint(s):</b><br/>No duplication of roles will be allowed</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Only users with the correct authority will be able to view employee roles.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/18/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 13:2.5 Create employee roles functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.6  |
| <b>Requirement name (use case name):</b>                 | View employee roles  |
| <b>Requirement short description:</b>                    | The system should allow the admin to view all employee roles   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to view all the employee roles saved on the system. The system will prompt the user to enter the role that they wish to search for. The user will provide the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>The system will search the system database for the record, retrieve the data and display it to the user.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• Only users with the correct authority will be able to view employee roles</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/08/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 14.2.6 View employee roles functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.7  |
| <b>Requirement name (use case name):</b>                 | Edit employee roles  |
| <b>Requirement short description:</b>                    | The system should allow the admin to update an employee's roles.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user will request to update an employee role. The user will select the employee role they would like to update. The system will then display the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The system will capture these details and perform validation in order to ensure that the details can be updated. Once the validation returns a success the system will update the relevant details in the systems database. The system will then notify the user that their role has been updated.</p> <p><b>Constraint(s):</b><br/>           The admin must "view" the employee role first before they will be able to update it.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• Only users with the correct authority will be able to view employee roles</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/18/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 15.2.7 Update employee roles functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.8  |
| <b>Requirement name (use case name):</b>                 | Delete employee roles  |
| <b>Requirement short description:</b>                    | The system should allow an admin to delete an employee role.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to delete an employee role. The system will prompt the user to confirm if they are sure that they want to continue with the deletion. The system will validate the deletion request and if it returns a success the employee role will be removed from the database. The user will then be notified by the system that their role has been removed.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Only users with the correct authority will be able to delete employee roles</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/18/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 16.2.8 Delete employee roles functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.9  |
| <b>Requirement name (use case name):</b>                 | Create Department  |
| <b>Requirement short description:</b>                    | The system should allow the admin to create a Department section on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Admin should be able to create new department on to the system. The admin will request to create new department. The system will request the following details:</p> <ul style="list-style-type: none"> <li>Name</li> <li>Description</li> </ul> <p>The admin will provide the details and it will then be validated by the system. System will notify the admin that the department has successfully been created.</p> <p><b>Constraint(s):</b><br/>No duplicate departments allowed.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 17.2.9 Create department functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.10  |
| <b>Requirement name (use case name):</b>                 | View Department   |
| <b>Requirement short description:</b>                    | The system should allow an admin to view all the departments.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to view a department. The system will prompt the user to enter the following details:</p> <ul style="list-style-type: none"> <li>Name</li> </ul> <p>The system will search for the relevant record in the database, and it will retrieve the data associated with it. The system will display the department data to the user.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 18:2.10 View department functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.11   |
| <b>Requirement name (use case name):</b>                 | Edit Department  |
| <b>Requirement short description:</b>                    | The system should allow an admin to update a department.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to update a department on the system. The user will select the department they would like to update. The system will then display all the related details for that account where the admin can then edit any of the details.</p> <ul style="list-style-type: none"> <li>Name</li> <li>Description</li> </ul> <p>The system will capture these details and perform validation in order to ensure that the details can be updated. Once the validation returns a success the system will update the relevant details in the systems database. The system will then notify the user that their department has been updated.</p> <p><b>Constraint(s):</b><br/>Admin must first view all departments before they can select and update the required department.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 19:2.11 Update department functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.12   |
| <b>Requirement name (use case name):</b>                 | Delete Department  |
| <b>Requirement short description:</b>                    | The system should allow an admin to delete a department.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Admin should be able to delete a selected department from the system. The system will confirm whether to continue with the deletion or not. All information for the department will then be deleted. The system notifies the admin that the department has been successfully removed.</p> <p><b>Constraint(s):</b><br/>Admin must first view all departments before they can select and delete the required department.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 20:2.12 Delete department functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.13  |
| <b>Requirement name (use case name):</b>                 | Create Branch   |
| <b>Requirement short description:</b>                    | The system should allow an admin to create a branch on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Admin should be able to create new branch on to the system. The admin will request to create new branch. The system will request the following details:</p> <ul style="list-style-type: none"> <li>Name</li> <li>Street</li> <li>City</li> <li>Postal Code</li> <li>Province</li> </ul> <p>The admin will capture the details and it will then be validated by the system. If the validation returns a success the system will create the new branch in the database. System will notify the admin that the branch has successfully been created.</p> <p><b>Constraint(s):</b><br/>No duplicate branches allowed</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 21:2.13 Create Branch functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.14  |
| <b>Requirement name (use case name):</b>                 | View Branch   |
| <b>Requirement short description:</b>                    | The system should allow an admin to view all the branches on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user will request to view a branch on the system. The system will prompt the user to enter the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>The user will select the branch they would like to view. The system will retrieve the relevant data from the system database. The system will then display all the related details for that branch.</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Street</li> <li>• City</li> <li>• Postal Code</li> <li>• Province</li> </ul> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 22:2.14 View Branch functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.15   |
| <b>Requirement name (use case name):</b>                 | Edit Branch  |
| <b>Requirement short description:</b>                    | The system should allow an admin to update a branch  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to update a branch on the system. The user will select the branch they would like to update. The system will then display all the relevant details of the branch.</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Street</li> <li>• City</li> <li>• Postal Code</li> <li>• Province</li> </ul> <p>The system will capture these details and perform validation in order to ensure that the details can be updated. Once the validation returns a success the system will update the relevant details in the systems database. The system will then notify the user that their branch has been updated.</p> <p><b>Constraint(s):</b><br/>Admin must first view all branches before they can select and update the required branches.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 23:2.15 Update Branch functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.16   |
| <b>Requirement name (use case name):</b>                 | Delete Branch  |
| <b>Requirement short description:</b>                    | The system should allow an admin to delete a branch.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Admin will request to delete a selected branch from the system. The system will confirm whether to continue with the deletion or not. All information for the branch will then be deleted. The system notifies that the admin that the branch has been successfully removed.</p> <p><b>Constraint(s):</b><br/>Admin must first view all branches before they can select and delete the required branch.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/20/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 24:2.16 Delete Branch functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.17  |
| <b>Requirement name (use case name):</b>                 | View delegation of authority  |
| <b>Requirement short description:</b>                    | The admin should be able to view the list of all the delegation of authority that took place in the system.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin clicks on the delegation of authority tab then the system will navigate to the list of all the delegation of authority that took place in the system. The admin will then be able to click on the view button then the system will display the details of that specific delegation of authority.</p> <p><b>Constraint(s):</b><br/>           None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 04/10/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 25: 2.17 View Delegation of Authority

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.18   |
| <b>Requirement name (use case name):</b>                 | Assign delegation of authority   |
| <b>Requirement short description:</b>                    | The admin should be able to setup a delegation of authority  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to assign a delegation of authority. The admin will provide the details of when it will start and end, who will receive the authority and the completed Delegation of Authority document.</p> <p><b>Constraint(s):</b><br/>           Must be approved before anything authority is given.<br/>           The admin will not be able to complete this process until the completed Delegation of Authority document has been uploaded.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 24/07/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 26: 2.18 Assigning delegation of authority functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.19  |
| <b>Requirement name (use case name):</b>                 | Edit delegation request   |
| <b>Requirement short description:</b>                    | The admin should be able to setup a delegation of authority   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin with the right authority will request to view all delegations on the system. The admin will select the delegation they would like to update. The system will then display all the related details for that delegation where the admin can then edit any of the details.</p> <ul style="list-style-type: none"> <li>• Delegate user</li> <li>• From Date</li> <li>• To Date</li> <li>• Delegation Document</li> </ul> <p>Once finished the system will validate the new details and either successfully update the details in the system database or it will display an error message regarding the details.</p> <p><b>Constraint(s):</b><br/>           The admin must “view” the delegation first before they will be able to update it.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 24/07/2023<br>Version 4.0   |
| <b>Priority:</b>   | High  |

Table 27:2.19 Edit delegation of authority functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.20   |
| <b>Requirement name (use case name):</b>                 | Revoke Access  |
| <b>Requirement short description:</b>                    | The system will revoke a certain delegation of authority if the specified time period has passed   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The system request to revoke a certain delegation of authority. The system will access the delegation of authority data in the systems database to continuously check and compare the date values given. If the end date has passed the system will revoke the access and authority of the delegated user back to their original access level. The system will then notify both the requester and the receiver about the shift in delegation of authority.</p> <p><b>Constraint(s):</b><br/>           Time period set must be passed.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 24/07/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 28: 2.20 Revoke Access functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.21  |
| <b>Requirement name (use case name):</b>                 | Create mandate limits   |
| <b>Requirement short description:</b>                    | The system should allow the admin to set mandate limits for different departments for handling procurements without management approval   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to create a new mandate limit for a department. The system will request the user to select a role and provide the mandated limit. The required details are provided</p> <ul style="list-style-type: none"> <li>• Amount</li> <li>• Department</li> </ul> <p>The system will validate that the roles have no other set mandate limits and then update it as needed. The system then displays a success message for the user.</p> <p><b>Constraint(s):</b><br/>           Only departments that does not have a non-default mandate limit should be displayed.</p> |
| <b>Business rules applicable to this requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be registered on the system.</li> <li>• The user must have authority to access the create mandate limits.</li> <li>• The different departments must be stored in the database</li> </ul>   |
| <b>Revision date and Revision number:</b>                | 03/19/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 29:2.21 Create mandate limits functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.22   |
| <b>Requirement name (use case name):</b>                 | View Mandate limits  |
| <b>Requirement short description:</b>                    | The system should allow the admin to view the different employees with a mandate limited.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to view all the mandated limits created for employees. The system will retrieve all required details from the relevant tables and display all the mandate limits.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be registered on the system.</li> <li>• The user must have the correct authority to view mandate limits</li> </ul>  |
| <b>Revision date and Revision number:</b>                | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 30:2.22 View mandate limits functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.23  |
| <b>Requirement name (use case name):</b>                 | Update Mandate limits   |
| <b>Requirement short description:</b>                    | The system should allow the admin to update the mandated limits in any employee.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to update a mandate limit for an employee. The system prompt the user to select the mandate limit that they wish to update. The user will select the limit and provide the following details:</p> <ul style="list-style-type: none"> <li>• Amount</li> <li>• Date</li> </ul> <p>The system will validate the details that need to be updated. If the validation returns a success the details will be updated on the system. The system finally displays a success message for the user.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be registered on the system.</li> <li>• The user must have authority to access the create mandate limits.</li> </ul>   |
| <b>Revision date and Revision number:</b>                | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

*Table 31.2.23 Update mandate limits functional requirement*

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.24  |
| <b>Requirement name (use case name):</b>                 | Delete Mandate limits   |
| <b>Requirement short description:</b>                    | The system should allow the admin to delete any mandated limits of an employee or group.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to delete a mandate limit for an employee. The system will request the selection of an employee for the deletion of the mandate limit. The system will validate the request in order to ensure that the data can be deleted. Once the validation returns a success the data will be removed from the system. The system finally displays a success message for the user.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this requirement:</b>    | <ul style="list-style-type: none"> <li>• The administrative must have added user into the system</li> </ul>   |
| <b>Revision date and Revision number:</b>                | 05/18/2023<br>Version 3.0   |

|                  |      |
|------------------|------|
| <b>Priority:</b> | High |
|------------------|------|

Table 32:2.24 Delete mandate limits functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.25   |
| <b>Requirement name (use case name):</b>                 | Backup system data   |
| <b>Requirement short description:</b>                    | The admin should be able to make a backup of all the data on the system.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin should be following a scheduled time for making a backup of the system data.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/19/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 33:2.25 Backup system data functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.26  |
| <b>Requirement name (use case name):</b>                 | Restore data to system  |
| <b>Requirement short description:</b>                    | The admin should be able to restore all data to the system using a backup.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin request to restore system to a previous version. The system will then restore all data to the point that it was last backed up.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/19/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 34:2.26 Restore data to system functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.27  |
| <b>Requirement name (use case name):</b>                 | Add Help  |
| <b>Requirement short description:</b>                    | The admin should be able to add a help section.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to create a help section. The system will prompt the user to enter the following:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> <li>• Category</li> </ul> <p>The system will capture these details and create the help function on the system.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• Only the admin can make use of the add help function</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 07/24/2023<br>Version 3.0   |
| <b>Priority:</b>   | Medium  |

Table 35:2.27 Add help functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.28   |
| <b>Requirement name (use case name):</b>                 | Update Help  |
| <b>Requirement short description:</b>                    | The admin should be able to update a help section.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin will request to update a help section. The system will prompt the user to update any of the following:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> <li>• Category</li> </ul> <p>The system will capture these details and update the help function on the system's database.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• Only the admin can make use of update help.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 07/24/2023<br>Version 3.0  |
| <b>Priority:</b>   | Medium   |

Table 36:2.28 Update help functional requirement

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.29   |
| <b>Requirement name (use case name):</b>                 | Delete Help  |
| <b>Requirement short description:</b>                    | The admin should be able to Delete a help section.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to delete a help section. The system will prompt the user to confirm that they want to delete the specified help section. The system will validate the request and once it retrieves a success the help section along with the following details will be deleted and removed from the system:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> <li>• Category</li> </ul> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• Only the admin can make use of Delete help.</li> <li>• The admin must confirm the deletion of the help.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 07/24/2023<br>Version 3.0  |
| <b>Priority:</b>   | Medium   |

Table 37:2.29 Delete help functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.30  |
| <b>Requirement name (use case name):</b>                 | View all help   |
| <b>Requirement short description:</b>                    | The admin should be able to view all the help sections.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin will request to view a help section. The system will prompt the user to select a help section that they would like to view. The system will retrieve the help section from the system's database and display it to the user along with the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> <li>• Category</li> </ul> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• Only the admin can make use of View all help.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 07/24/2023<br>Version 3.0   |
| <b>Priority:</b>   | Medium  |

Table 38:2.30 View all help functional requirement

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.31  |
| <b>Requirement name (use case name):</b>                 | Create admin  |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to create a new admin account.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin with the right authority should be able to create a new admin account. Where they will need to provide all the information such as:</p> <ul style="list-style-type: none"> <li>• email address</li> <li>• phone number</li> <li>• name</li> <li>• surname</li> </ul> <p>The details are then validated by the system and added to the database.</p> <p><b>Constraint(s):</b><br/>A duplicate account cannot be made.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin must be logged into the system.</li> <li>• All needed details must be provided.</li> <li>• A strong password should be given.</li> <li>• The user requires internet access.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 39: 2.31 Create Admin

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.32   |
| <b>Requirement name (use case name):</b>                 | View admin   |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to view and search all admin accounts on the system.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The admin with the right authority will request to either search or view all admin accounts on the system. The admin will select the admin account they would like to view. The system will then display all the related details for that account.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin with the right authority must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 40: 2.31 View Admin

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.33  |
| <b>Requirement name (use case name):</b>                 | Edit admin  |
| <b>Requirement short description:</b>                    | The system should allow the admin with the right authority to update an admin's account detail.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin with the right authority will request to view all admin accounts on the system. The admin will select the admin account they would like to update. The system will then display all the related details for that account where the admin can then edit any of the details.</p> <ul style="list-style-type: none"> <li>• email address</li> <li>• phone number</li> <li>• name</li> <li>• surname</li> </ul> <p>Once finished the system will validate the new details and either successfully update the details in the system database or it will display an error message regarding the details. The system then notifies the admin that their account has been updated.</p> <p><b>Constraint(s):</b><br/>           The admin must "view" the admin account first before they will be able to update it.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The admin with the right authority must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 2.34  |
| <b>Requirement name (use case name):</b>                 | Delete admin  |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to delete an admin account.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin wishes to delete an admin profile. The admin views the profile they wish to delete and requests deletion. The system will validate that the admin can be deleted. If the validation returns a success the system will remove the admin profile from the database.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user with the right authority must be logged into the system.</li> <li>• The admin must not be with the company anymore.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 42: 2.34 Delete Admin

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.35   |
| <b>Requirement name (use case name):</b>                 | Delete Delegation of Authority   |
| <b>Requirement short description:</b>                    | The system should allow an admin with the right authority to delete a delegation request.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The admin wishes to delete a delegation request. The admin views the delegation they wish to delete and requests deletion. The system will validate that the delegation can be deleted. If the validation returns a success the system will remove the delegation request from the database.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user with the right authority must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 24/07/2023<br>Version 1.0  |
| <b>Priority:</b>   | High   |

Table 43: 2.35 Delete Delegation of Authority

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 2.36   |
| <b>Requirement name (use case name):</b>                 | Activate Delegation of Authority   |
| <b>Requirement short description:</b>                    | The system will activate a certain delegation of authority if the specified time period begins   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The system request to activate a certain delegation of authority. The system will access the delegation of authority data in the systems database to continuously check and compare the date values given. If the start date has been reached the system will activate the access and authority of the delegated user back to the additional temporary access level.</p> <p><b>Constraint(s):</b><br/>           Time period set must begin.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The admin must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 24/07/2023<br>Version 1.0  |
| <b>Priority:</b>   | High   |

Table 44: 2.36 Activate Delegation of Authority

**SUBSYSTEM 3 – PROCUREMENT SUBSYSTEM**

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.1  |
| <b>Requirement name (use case name):</b>                 | Create procurement request   |
| <b>Requirement short description:</b>                    | Allows user to make a new request for procurement.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user can request to make a new request for procurement. The system will request the following details:</p> <ul style="list-style-type: none"> <li>• Vendor name (It can be an approved vendor selected from a dropdown list, or a other vendor from which the name should be specified)</li> <li>• Quote options</li> </ul> <p>The system will validate the request. In the case of a “other” vendor that was selected, the system will validate the request to check if that vendor has been used more than 2 times. Once the request returns a success the request will be stored in the database and given the status of “Submitted” is assigned.</p> <p><b>Constraint(s):</b><br/>           None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 45: 3.1 Create Procurement request.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.2  |
| <b>Requirement name (use case name):</b>                 | View procurement request   |
| <b>Requirement short description:</b>                    | User will be able to view a request for procurement which has been made.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user requests to view a procurement request. When the view button is pressed, the procurement request should be displayed, which will detail the</p> <ul style="list-style-type: none"> <li>• Vendor name</li> <li>• Quote options</li> </ul> <p>The system will search the systems database and retrieve the relevant procurement request records and display it to the user where the user will be able to view the Procurement request details that have been stored on the system.</p> <p><b>Constraint(s):</b><br/>           At least one procurement request must be present on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 46: 3.2 View procurement request.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.3  |
| <b>Requirement name (use case name):</b>                 | Update procurement request   |
| <b>Requirement short description:</b>                    | The user should be able to update a request for procurement that has not yet been approved or rejected.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user will request to update a procurement request. The user will select the request they would like to update. The system will then display all the related details for that account where the user can then edit any of the details.</p> <ul style="list-style-type: none"> <li>• Vendor name</li> <li>• Quote options</li> </ul> <p>Once finished the system will validate the new details and either successfully updated the details in the system database or it will display an error message regarding the details.</p> <p><b>Constraint(s):</b><br/>The request for procurement should be temporary disabled from being rejected or approved.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 47: 3.3 Update procurement request

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 3.4   |
| <b>Requirement name (use case name):</b>                 | Delete procurement request  |
| <b>Requirement short description:</b>                    | The user should be able to delete a request for procurement that has not yet been approved or rejected.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user will request to delete a procurement request. The user will select the request they would like to delete. The system will prompt the user to confirm the deletion of the request. The request of deletion will be validated and if the validation returns a success, all information for the request will then be deleted. The system notifies that the request has been successfully removed.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 48: 3.4 Delete procurement request.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.5  |
| <b>Requirement name (use case name):</b>                 | Approve Procurement Request  |
| <b>Requirement short description:</b>                    | The system should allow procurement request created to be approved or rejected.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The Budget owner requests to approve or reject a request. The system will prompt the user to select the request they would like to approve or reject. The user will select the request and subsequently either approve or reject it. Once the request has been accepted or rejected the relevant buyer will be notified that the approval has been made</p> <p><b>Constraint(s):</b><br/>A procurement request must exist, and at least 1 quotes must be uploaded for this procurement request.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• Budget Owner should be logged into the system</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 07/26/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

*Table 49: 3.5 Approve Procurement Request*

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 3.6   |
| <b>Requirement name (use case name):</b>                 | Place procurement details   |
| <b>Requirement short description:</b>                    | The system should allow the user to place order details into the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user request to place order details. The system will request the following details:</p> <ul style="list-style-type: none"> <li>• Buyer Name</li> <li>• Buyer Email</li> <li>• Item Type</li> <li>• Consumable item/Asset Name</li> <li>• Quantity/Description</li> <li>• Account Code</li> <li>• Payment Type</li> <li>• Deposit Amount</li> <li>• Deposit Due Date</li> <li>• Full payment due date</li> <li>• Receipt Document</li> <li>• Total amount</li> <li>• Proof of payment</li> <li>• Comment</li> </ul> <p>The provided details are then validated by the system and then saved into the database. The system then displays a success message for the user.</p> <p><b>Constraint(s):</b><br/>           The procurement request must have been approved.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 07/23/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 50: 3.6 Place procurement request.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.7  |
| <b>Requirement name (use case name):</b>                 | Approve Flagged Procurement Details  |
| <b>Requirement short description:</b>                    | The system will flag a procurement details when its cost is over a mandated limit and allow the *budget owner* to review request to either reject or accept request.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The system flags a procurement for going over mandated limit and notify the budget owner of the procurement request. The budget owner will request to review request. The system will display the procurement request and request that the user provide either accept or reject input. The input is provided by the user and either a reject or approved notification is sent to the requestor.</p> <p><b>Constraint(s):</b><br/>The procurement request must be larger than the requestor mandate limit.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user must be logged in the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 51: 3.7 Management mandate approval

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.8  |
| <b>Requirement name (use case name):</b>                 | Receive Procurement item   |
| <b>Requirement short description:</b>                    | The user should be allowed to update stock items for when receiving procurement items.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user will request to add the stock received on the system. The system will retrieve the data from the procurement request and prompt the user to enter the quantity received. The user will enter the following details relative to the procurement request item:</p> <ul style="list-style-type: none"> <li>Amount</li> </ul> <p>The system will capture the relevant details and the amount will be saved on the system's database. An on-screen notification will be displayed to the user to inform them of the successful stock capture.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> <li>The request must be approved.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 52: 3.8 Receive Procurement item.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.9  |
| <b>Requirement name (use case name):</b>                 | Upload tax Invoice   |
| <b>Requirement short description:</b>                    | A user should be allowed to upload a tax invoice.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user would like to upload a tax invoice for a procurement request. The system will request the user to provide the required file. The user will select the file they want to upload from their device onto the system. The system will then save the files with into the database. The system will display a success message if the file has been successfully added to the database.</p> <p><b>Constraint(s):</b><br/>           The system will not allow the user to continue with the procurement process until the invoice has been uploaded to the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 53: 3.9 Upload tax Invoice

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.10   |
| <b>Requirement name (use case name):</b>                 | Budget owner Requisition approval  |
| <b>Requirement short description:</b>                    | The system should allow the budget owner to review the requisition request and approve or reject it.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user request to review the requisition request. The system will display all the details from the procurement request. The budget owner will verify the uploaded invoice, the total amount due, and the selected budget line. After verification the budget owner will either accept or reject the procurement request and notify the appropriate parties.</p> <p>If the request is approved the system will notify the Finance department to continue with payment.</p> <p>If the request is rejected the buyer will be notified to make the requested changes.</p> <p><b>Constraint(s):</b><br/>           None</p> |

|   |   |
|---|---|
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> <li>The request must be approved.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 03/16/2023<br>Version 2.0   |
| <b>Priority:</b>                                      | High  |

Table 54: 3.10 Budget owner Requisition approval

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.11   |
| <b>Requirement name (use case name):</b>                 | Upload receipt   |
| <b>Requirement short description:</b>                    | A user should be allowed to upload a receipt.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to upload a receipt for a procurement request. The system will request the user to provide the required file. The user will select the file they want to upload from their device onto the system. The system will then save the file into the database. The system will display a success message if the file has been successfully added to the database.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 55: 3.11 Upload receipt

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 3.12   |
| <b>Requirement name (use case name):</b>                 | Upload credit card Payment   |
| <b>Requirement short description:</b>                    | A user should be allowed to upload a credit card payment.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to upload a credit card payment for a procurement request. The system will request the user to provide the required file. The user will select the file they want to upload from their device onto the system. The system will then save the file into the database. The system will display a success message if the file has been successfully added to the database.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The request must have a status of approved</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |

|                  |      |
|------------------|------|
| <b>Priority:</b> | High |
|------------------|------|

Table 56: 3.12 Upload credit card Payment

---

## SUBSYSTEM 4 – FINANCE SUBSYSTEM

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 4.1  |
| <b>Requirement name (use case name):</b>                 | Finalize procurement requests  |
| <b>Requirement short description:</b>                    | The user should be able to search or view all invoices on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to finalize a procurement request. The user would select the procurement request that is awaiting payment. The request would be reviewed and paid outside of the system. The user will then upload the proof of payment (optional) and finalize the procurement request. The system will then automatically deduct the amount of the invoice from the requesters procurement items specific budget line. A notification would then be sent to the requester that their procurement request has been finalized.</p> <p><b>Constraint(s):</b></p> |

|   |  |
|---|--|
|   | Procurement request must exist beforehand  |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> <li>• A procurement item request must be approved.</li> <li>• Invoice must be uploaded to the system</li> </ul> |
| <b>Revision date and revision Number:</b>             | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>                                      | High   |

Table 57: 4.1 Finalize procurement requests.

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 4.2   |
| <b>Requirement name (use case name):</b>                 | Upload proof of payment   |
| <b>Requirement short description:</b>                    | The user should be able upload proof of payment   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to upload a proof of payment for their specific procurement request. The user will select the file they want to upload from their device onto the system. The system will store the proof of payment on the system database.</p> <p><b>Constraint(s):</b><br/>The File that the user should upload must be the proof of payment of their specific procurement request.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> <li>• The user must have the right authority to upload a POP.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/08/2023<br>Version 1.0   |
| <b>Priority:</b>   | High  |

Table 58: 4.2 Upload proof of payment

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 4.3  |
| <b>Requirement name (use case name):</b>                 | Create budget allocation   |
| <b>Requirement short description:</b>                    | The user should be able to create a new budget allocation for a department.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to add a budget allocation to the system. The system will gather the following details.</p> <ul style="list-style-type: none"> <li>• Date</li> <li>• Total</li> <li>• Year for budget</li> <li>• Department name</li> </ul> |

|   |   |
|---|---|
|   | <p>The system will capture the data on the system and save it in the systems database. The system will notify the relevant budget owner that the budget has been created.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> <li>• The user must have the right authority to create a budget allocation.</li> </ul>   |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>                                      | High  |

Table 59: 4.3 Create Budget allocation

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 4.4  |
| <b>Requirement name (use case name):</b>                 | View budget allocation   |
| <b>Requirement short description:</b>                    | The user should be able to view all budget allocation for the different departments.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user requests to view all budget allocation for the different departments. The system will retrieve the following details:</p> <ul style="list-style-type: none"> <li>• Date</li> <li>• Total</li> <li>• Year for budget</li> <li>• Department name</li> </ul> <p>This information will be displayed to the user.</p> <p><b>Constraint(s):</b><br/>None</p> |

|   |   |
|---|---|
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> <li>The user must have the right authority to access budget allocation.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 2.0   |
| <b>Priority:</b>                                      | High  |

Table 60: 4.4 View budget allocation

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 4.5   |
| <b>Requirement name (use case name):</b>                 | Update budget allocation  |
| <b>Requirement short description:</b>                    | The user should be able update the budget allocation  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user will request to view all budget allocations on the system. The user will select the request they would like to update. The user has the option to update the relevant details on the system which includes:</p> <ul style="list-style-type: none"> <li>Adding and removing of line items</li> <li>Changing figures</li> </ul> <p>The relevant changes will be stored on the system's database.</p> <p><b>Constraint(s):</b><br/>           There must be a budget allocation created on the system</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged into the system.</li> <li>The user must have the right authority to access budget allocation.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 61: 4.5 Update budget allocation

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 4.6   |
| <b>Requirement name (use case name):</b>                 | Delete budget allocation  |
| <b>Requirement short description:</b>                    | The system should allow a user to delete a budget allocation.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user would like to delete a budget allocation. The user would view the budget that they would like to remove. Once the user has chosen to remove the budget allocation, the system will validate the request. If the validation returns a success the user will be notified that the budget has been successfully deleted and the data will be removed from the system.</p> <p><b>Constraint(s):</b><br/>           There must be a budget created on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user must be logged into the system.</li> <li>The user must have the right authority to access budget allocation.</li> </ul>   |

|   |                           |
|---|---------------------------|
| <b>Revision date and revision Number:</b> | 05/18/2023<br>Version 2.0 |
| <b>Priority:</b>                          | High                      |

Table 62: 4.6 Delete budget allocation.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 4.7  |
| <b>Requirement name (use case name):</b>                 | Create budget category   |
| <b>Requirement short description:</b>                    | The user should be able to create a new budget category.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to add a budget category to the system. The system will prompt the user to enter the following details for the budget category:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• The user must have the right authority to access budget category.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 1.0  |
| <b>Priority:</b>   | High   |

Table 63: 4.7 Create Budget Category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 4.8   |
| <b>Requirement name (use case name):</b>                 | View budget category  |
| <b>Requirement short description:</b>                    | The user should be able to view all budget categories on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user requests to view all budget categories. The system will retrieve the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>This information will be displayed to the user.</p> <p><b>Constraint(s):</b></p> |

|   |   |
|---|---|
|   | None  |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> <li>• The user must have the right authority to access budget category.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 1.0   |
| <b>Priority:</b>                                      | High  |

Table 64: 4.8 View Budget Category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 4.9   |
| <b>Requirement name (use case name):</b>                 | Update budget category  |
| <b>Requirement short description:</b>                    | The user should be able update the budget category  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user will request to view all budget categories on the system. The user will select the category they would like to update. The user has the option to update the relevant details on the system which includes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The relevant changes will be stored on the system's database.</p> <p><b>Constraint(s):</b><br/>           There must be a budget category created on the system</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged into the system.</li> <li>• The user must have the right authority to access budget category.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 1.0   |
| <b>Priority:</b>   | High  |

Table 65: 4.9 Update Budget Category

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 4.10   |
| <b>Requirement name (use case name):</b>                 | Delete budget category   |
| <b>Requirement short description:</b>                    | The system should allow a user to delete a budget category.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user would like to delete a budget category. The user would view the budget category that they would like to remove. Once the user has chosen to remove the budget category, the system will validate the request. If the validation returns a success the user will be notified that the budget category has been successfully deleted and the data will be removed from the system.</p> <p><b>Constraint(s):</b><br/>There must be a budget category created on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• The user must have the right authority to access budget category.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 1.0  |
| <b>Priority:</b>   | High   |

Table 66: 4.10 Delete Budget Category

## SUBSYSTEM 5 – INVENTORY SUBSYSTEM

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.1   |
| <b>Requirement name (use case name):</b>                 | Create Consumable   |
| <b>Requirement short description:</b>                    | The system should allow a procurement consumable item to be created   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The use case describes the event where the user will want to create a new consumable. The system will prompt the user to enter the product details. Including:</p> <ul style="list-style-type: none"> <li>• Item name</li> <li>• Item description</li> <li>• Item Category</li> </ul> <p>Once this is complete and the item is saved the item will be added to the system's database.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged in to the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>18/08/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 67: 5.1 Create inventory item

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 5.2  |
| <b>Requirement name (use case name):</b>                 | View Consumables   |
| <b>Requirement short description:</b>                    | The system should allow the user to view consumables on the system   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The use case describes the event where a user would want to search for a consumable. The item can be searched for with the following details:</p> <ul style="list-style-type: none"> <li>• Item name</li> <li>• ItemID</li> </ul> <p>The system will search through all of the records in the database and retrieve the correct item and display it to the user based on their search criteria.</p> <p><b>Constraint(s):</b><br/>           There needs to be an inventory item created on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged in to the system</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 68: 5.2 View inventory item

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 5.3  |
| <b>Requirement name (use case name):</b>                 | Update Consumable  |
| <b>Requirement short description:</b>                    | The system should allow the user to update a consumable  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The use case describes the event where a user would want to update certain consumable. The user will view the item that they wish to update. They will enter the relevant details that they wish to change and once updated the data will be validated on the system. If the validation returns a success, the details will be saved on the systems database.</p> <p><b>Constraint(s):</b><br/>           There needs to be an inventory item on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 69: 5.3 Update inventory item

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.4   |
| <b>Requirement name (use case name):</b>                 | Delete Consumable   |
| <b>Requirement short description:</b>                    | The system should allow the user to delete a consumable   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>This use case describes the event where the user requests to delete a consumable. The user will select the inventory item they want to delete. The system will ask the user for confirmation of deletion. The system will validate the request of deletion. If the validation returns a success the system will remove the procurement inventory item from the system.</p> <p><b>Constraint(s):</b><br/>An inventory item needs to be on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged in to the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 70: 5.4 Delete inventory item

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 5.5  |
| <b>Requirement name (use case name):</b>                 | Create Consumable Category   |
| <b>Requirement short description:</b>                    | The system should allow a category to be created   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The use case describes the event where a user wants to create a category on the system. The system will prompt the user to enter the category details such as:</p> <ul style="list-style-type: none"> <li>Category name</li> <li>Category Description</li> </ul> <p>Once this is complete and the category is saved the category will be added to the system's database.</p> <p><b>Constraint(s):</b><br/>None.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user must be logged in to the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 71: 5.5 Create item category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.6   |
| <b>Requirement name (use case name):</b>                 | View Consumable categories  |
| <b>Requirement short description:</b>                    | The system should allow the user to view a consumable category  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The use case describes the event where a user would want to search for a specific category. The item can be searched for with the following details:</p> <ul style="list-style-type: none"> <li>• category name</li> <li>• Category ID</li> </ul> <p>The system will search through all of the records in the database and retrieve the correct category and display it to the user based on their search criteria.</p> <p><b>Constraint(s):</b><br/>           There needs to be an inventory item category created on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged in to the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 72: 5.6 View item category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.7   |
| <b>Requirement name (use case name):</b>                 | Update Consumable Category  |
| <b>Requirement short description:</b>                    | The system should allow the user to update a consumable category  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The use case describes the event where a user would want to update a certain category. The user will view the category that they wish to update. They will enter the relevant details that they wish to change and once updated the data will be validated on the system. If the validation returns a success, the details will be saved on the systems database.</p> <p><b>Constraint(s):</b><br/>           There needs to be an inventory item category on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• User must be logged in to the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 73: 5.7 Update Item Category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.8   |
| <b>Requirement name (use case name):</b>                 | Delete Consumable Category  |
| <b>Requirement short description:</b>                    | The system should allow the user to delete a consumable Category  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           This use case describes the event where the user requests to delete an consumable category. The user will select the category they want to delete. The system will ask the user for confirmation of deletion. The system will validate the request of deletion. If the validation returns a success the system will remove the procurement inventory category from the system.</p> <p>.</p> <p><b>Constraint(s):</b><br/>           An inventory item category needs to be on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>User must be logged in to the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 18/08/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 74: 5.8 Delete Item Category

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 5.9   |
| <b>Requirement name (use case name):</b>                 | Export inventory details  |
| <b>Requirement short description:</b>                    | The system will allow the user to export the details of the procurement item inventory to a CSV file  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The use case describes the event where the user would like to export the details contained in the procurement item inventory system (Item name, stock on hand) to a .csv file in the structure supported by Xero. The user will export the inventory list and the .csv will be generated as response.</p> <p><b>Constraint(s):</b><br/>           There must be inventory loaded on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user with the right authority must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/08/2023<br>Version 1.0   |
| <b>Priority:</b>   | High  |

Table 75: 5.9 Export inventory details

Table 76: 5.10 Generate reorder stock notification.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 5.10   |
| <b>Requirement name (use case name):</b>                 | Complete stock take  |
| <b>Requirement short description:</b>                    | The system should allow the user to complete a stock take.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The use case describes the event where the user completes a stock take, and the stock levels of different consumables are entered and updated. The user will select the items for which they intend to perform a stock take, and the system will prompt them to enter the amount of stock which exists for these items, after which this data will be captured and saved to the system database.</p> <p><b>Constraint(s):</b><br/>There must be inventory loaded on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Only the appropriate users will be able to perform a stock take</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/08/2023<br>Version 1.0  |
| <b>Priority:</b>   | High   |

Table 77: 5.10 Complete stock take

---

## SUBSYSTEM 6 – VENDOR SUBSYSTEM

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.1  |
| <b>Requirement name (use case name):</b>                 | Create Vendor onboard Request  |
| <b>Requirement short description:</b>                    | The system should allow the user to create a vendor onboard request.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The system should allow the user to create a vendor onboard request. The system should allow the user to select between a normal vendor or a sole supplier.</p> <p>In the case of a normal vendor: The user would have had to collect 3 quotes from 3 different companies. The system will prompt the user to upload the documents from their computer's file system. The user will be prompted to enter the company name and email related to each quote. The user will then confirm the request which will notify the Procurement department for approval.</p> <p>In the case of a sole supplier: The user will provide the company name and reason for this vendor to be approved with the option of providing a quote. The user will then confirm the request which will notify the Managing director for approval.</p> |

|   |  |
|---|--|
|   | <b>Constraint(s):</b><br>A number of 3 Quotes and company names must be added to the request for the user to be able to proceed with the process if the sole supplier option is NOT selected.  |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• The business requires a minimum of 3 quotes for a vendor to be onboarded in the case of a normal vendor.</li> <li>• Sole suppliers do not need to have quotes as supporting documents but require the managing director's approval</li> </ul> |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>                                      | High   |

Table 78: 6.1 Create Vendor onboard Request.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.2  |
| <b>Requirement name (use case name):</b>                 | View Vendor onboard Request  |
| <b>Requirement short description:</b>                    | The system should allow the user to view the vendor onboard request  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           Under the vendor section, the requester (user) should be able view their current vendor onboard requests. When the view button is pressed, the request should be displayed along with the uploaded quotes, company names, and the current status of the request.</p> <p><b>Constraint(s):</b><br/>           An existing request needs to have been created.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• Only the user can update their own request.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 79: 6.2 View Vendor onboard Request

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.3  |
| <b>Requirement name (use case name):</b>                 | Update Vendor onboard Request  |
| <b>Requirement short description:</b>                    | The system should allow the requester (user) to update their vendor onboard request.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The requester(user) requests to edit a vendor onboard request. Under the vendor section, the requester (user) should be able update their current vendor onboard requests. The system will prompt the user to enter the details that they wish to update. The user would update the details and validation will be performed. If the validation returns a success the details will be updated on the system.</p> |
|  | <p>The system will notify the Procurement department of the update and request approval.</p> <p><b>Constraint(s):</b><br/>           An existing request needs to have been created.<br/>           A number of 3 Quotes and company names and emails must be added to the request for the user to be able to proceed with the process.</p>  |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• Only the user can update their own request.</li> <li>• The business requires a minimum of 3 quotes for a vendor to be onboarded.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | High   |

Table 80: 6.3 Edit Vendor onboard request

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.4   |
| <b>Requirement name (use case name):</b>                 | Delete Vendor onboard Request   |
| <b>Requirement short description:</b>                    | The system should allow the requester (user) to remove their vendor onboard request from the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user wishes to delete a specific onboard request. The user would select the specific request that they would like to delete. The user will delete the specific onboard request, it will be validated on the system to ensure that there are no associated records to the request and if the validations succeed the data will be removed from the system.</p> <p><b>Constraint(s):</b><br/>           The user must first view the vendor onboard requests before they will be able to remove a request from the system.<br/>           An existing request needs to have been created.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user must be logged into the system.</li> <li>• Only the user can remove their own request.</li> </ul>   |

|   |                           |
|---|---------------------------|
| <b>Revision date and revision Number:</b> | 03/16/2023<br>Version 2.0 |
| <b>Priority:</b>                          | High                      |

Table 81: 6.4 Delete Vendor onboard Request

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.5   |
| <b>Requirement name (use case name):</b>                 | Approve onboard request   |
| <b>Requirement short description:</b>                    | The system should allow the GRC officer to approve a vendor onboarding request.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The GRC officer requests to approve a vendor onboard request. The GRC officer will select the request that they wish to approve. The system will display the following details which relates to the onboard request:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• 3 quotes</li> </ul> <p>The GRC officer will approve the correct company on the system. The system will prompt the GRC officer to add the vendor's name, BEE rating, BEE certificate, and BEE expiration date to a Vendor Setup form.</p> <p>In the case of a sole supplier: The managing director will request to view the onboard request on the system. The following detail will be displayed:</p> <ul style="list-style-type: none"> <li>• Vendor Name</li> </ul> |

|   |   |
|---|---|
|   | <ul style="list-style-type: none"> <li>• Relevant Reason</li> <li>• Optional quote</li> </ul> <p>The managing director will approve the setup request. The system will notify the GRC officer to add the vendor's name, BEE rating, BEE certificate, and BEE expiration date to a Vendor Setup form.</p> <p>In both cases of a sole supplier and a normal vendor the system will send a notification, along with the uncompleted Vendor Setup form containing the vendor's name, BEE rate, BEE certificate, and BEE expiration date, to the buyer/requester notifying them that they can proceed with the vendor setup process.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• The GRC officer/ managing director must be logged into the system.</li> <li>• The vendor's name, BEE rating, BEE certificate, and BEE expiration date must be added to the form approval process to be completed.</li> </ul>   |
| <b>Revision date and revision Number:</b>             | 03/16/2023<br>Version 2.0   |
| <b>Priority:</b>                                      | High  |

Table 82: 6.5 Approve onboard request

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.6   |
| <b>Requirement name (use case name):</b>                 | Create vendor   |
| <b>Requirement short description:</b>                    | The system should allow a vendor to be added on the system once the Vendor onboard request has been approved.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user wishes to create a vendor on the system. The user will select the vendor that they have requested and that has been approved and they will provide the following details:</p> <p><b>Company Contact Information</b></p> <ul style="list-style-type: none"> <li>• Telephone</li> <li>• Fax</li> <li>• Email</li> <li>• Point of Contact Title</li> <li>• Point of Contact Name</li> <li>• Contact Phone Number</li> <li>• Registered Address</li> <li>• Website URL</li> <li>• Contact Email</li> </ul> <p><b>Company Overview Information</b></p> |

|   |   |
|---|---|
|   | <ul style="list-style-type: none"> <li>• Details of goods and services</li> <li>• Vat registration number</li> <li>• Vat Registration Document</li> <li>• Company Registration Number</li> <li>• Income tax Number</li> <li>• Tax clearance certificate</li> <li>• Signed Agreement Details</li> <li>• Signed Agreement Document</li> <li>• Insurance Cover Details</li> <li>• Insurance Cover Document</li> <li>• Payment Terms Details</li> <li>• License/ Accreditation Number and document</li> </ul> <p><b>Bank Name</b></p> <ul style="list-style-type: none"> <li>• Bank Name</li> <li>• Branch Code</li> <li>• Account Name</li> <li>• Account Type</li> <li>• Account Number</li> <li>• Bank Contact Name</li> <li>• Bank Contact Phone Number</li> <li>• Bank stamped confirmation letter</li> </ul> <p>The system will take these details and save them on the system's database.</p> <p><b>Constraint(s):</b></p> <ul style="list-style-type: none"> <li>• The system will not allow the user to proceed if they have not uploaded the accompanying documents to the following fields:</li> <li>• Company Registration Number and document</li> <li>• Income Tax Number and document</li> </ul> |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>• The user must be logged in to the system.</li> <li>• The vendor will only be able to be created once the onboarding request has been approved</li> </ul>   |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>                                      | High  |

Table 83: 6.6 Create vendor

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.7   |
| <b>Requirement name (use case name):</b>                 | View Vendor   |
| <b>Requirement short description:</b>                    | The system should allow a specific vendor and the details associated with the vendor to be viewed.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user wishes to view a specific vendor. The user will search for the user using the following details:</p> <ul style="list-style-type: none"> <li>• Company name</li> </ul> <p>The system will search the systems database and retrieve the relevant vendor record and display it to the user where the user will be able to view the vendor details that have been stored on the system.</p> <p>The user will also be able to view more detail by clicking a view button next to a specific vendor record.</p> <p><b>Constraint(s):</b><br/>           There needs to be a vendor added on the system</p> |

|   |   |
|---|---|
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>The user needs to be logged in to the system.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 05/18/2023<br>Version 3.0   |
| <b>Priority:</b>                                      | High  |

Table 84: 6.7 View Vendor

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.8  |
| <b>Requirement name (use case name):</b>                 | Update Vendor  |
| <b>Requirement short description:</b>                    | The system should allow the user to update a specific vendor on the system.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user wishes to update a specific vendor. The user will select the vendor they wish to update and provide the updated details. The system will capture and validate the updated details. Finally, a success notification is then displayed and return the user back to the view vendor screen.</p> <p>.</p> <p><b>Constraint(s):</b><br/>There needs to be a vendor added onto the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user needs to be logged in to the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 05/18/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 85: 6.8 Update Vendor

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.9  |
| <b>Requirement name (use case name):</b>                 | Delete Vendor  |
| <b>Requirement short description:</b>                    | The system should allow the user to delete a specific vendor record  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user wishes to delete a specific vendor. The user would select the specific vendor that they would like to delete. The user will delete the specific vendor, it will be validated on the system to ensure that there are no associated records to the request and if the validations succeed the data will be removed from the system.</p> <p>.</p> <p><b>Constraint(s):</b><br/>There needs to be a vendor added onto the system</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>The user needs to be logged into the system</li> <li>Only a user with the correct access level will be allowed to remove a vendor from the system</li> </ul>  |

|   |                           |
|---|---------------------------|
| <b>Revision date and revision Number:</b> | 03/16/2023<br>Version 2.0 |
| <b>Priority:</b>                          | High                      |

Table 86: 6.9 Delete Vendor

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.10  |
| <b>Requirement name (use case name):</b>                 | Export due diligence vendor checklist   |
| <b>Requirement short description:</b>                    | The system should allow the user to export a due diligence vendor checklist.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The user would like to export a vendor due diligence checklist. The user would search for the vendor by using the following fields:</p> <ul style="list-style-type: none"> <li>• Company name</li> </ul> <p>The user would select the vendor and export the checklist which will be downloaded in a structured pdf format.</p> <p><b>Constraint(s):</b><br/>           The due diligence checklist for the specific vendor needs to be completed.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• The user needs to be logged in to the system.</li> <li>• The user with the correct access level will be able to export the due diligence checklist</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0   |
| <b>Priority:</b>   | High  |

Table 87: 6.10 Export due diligence vendor checklist

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.11  |
| <b>Requirement name (use case name):</b>                 | Generate Sole Supplier Review Notification  |
| <b>Requirement short description:</b>                    | The system should generate a notification to indicate that a Sole supplier should be reviewed.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           Notifications should be generated by the system which indicate that a Sole supplier should undergo a performance review. This notification will be sent on the first day of the month in which the vendor was first added to the system.</p> <p><b>Constraint(s):</b><br/>           None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 03/19/2023<br>Version 1.0   |
| <b>Priority:</b>   | Medium  |

Table 88: 6.11 Generate Sole Supplier Review Notification

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 6.12   |
| <b>Requirement name (use case name):</b>                 | Generate BEE Certificate Expiry Notification.  |
| <b>Requirement short description:</b>                    | The system should generate a notification to indicate the expiry of a BEE certificate  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>Notifications should be generated by the system which indicate that the expiry date of a BEE certificate which has been added to the system is nearing. This should be done 1 week before, 3 weeks before, as well as once the expiry date for the certificate has officially been reached. This notification should be sent to the GRC officer via email.</p> <p><b>Constraint(s):</b><br/>At least 1 BEE certificate should be present on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 03/16/2023<br>Version 2.0  |
| <b>Priority:</b>   | Medium   |

Table 89: 6.12 Generate BEE Expiry Certification Notification

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.13  |
| <b>Requirement name (use case name):</b>                 | Update Approve onboard request  |
| <b>Requirement short description:</b>                    | The system should allow the user to update the due diligence checklist and other details related to it  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The GRC officer requests to update a vendor onboard request due diligence checklist details. The system will prompt the GRC officer to add the vendor's BEE rating, BEE certificate, and BEE expiration date, Due diligence checklist and POPI related details.</p> <p>The system will then capture and validate the updated details. Finally, a success notification is then displayed and return the user back to the View Approved or Pending onboard request screen.</p> <p><b>Constraint(s):</b><br/>None</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>None</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 07/29/2023<br>Version 1.0   |
| <b>Priority:</b>   | Medium  |

Table 90: 6.13 Update Approve Onboard Request

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 6.14  |
| <b>Requirement name (use case name):</b>                 | View Approved Or Pending Onboard Request  |
| <b>Requirement short description:</b>                    | The system should allow the user to view approved or pending onboard request.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user wishes to view approved or pending onboard request. The user will search for the user using the following details:</p> <ul style="list-style-type: none"> <li>• Onboard request details</li> </ul> <p>The system will search the systems database and retrieve the relevant onboard request records and display it to the user where the user will be able to view the onboard request details and add, remove or update any due diligence details.</p> <p><b>Constraint(s):</b><br/>There needs to be a onboard request that has been created and not rejected on the system</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Revision date and revision Number:</b>                | 07/29/2023<br>Version 1.0   |
| <b>Priority:</b>   | Medium  |

Table 91: 6.14 View Approved or Pending Onboard Request

---

#### SUBSYSTEM 7 – REPORT SUBSYSTEM

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 7.1   |
| <b>Requirement name (use case name):</b>                 | Generate approved vendor Report   |
| <b>Requirement short description:</b>                    | The system should allow the user to generate a list of the approved vendors.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The GRC officer request to generate an approved vendor report. The system will generate the report which includes a table list of all the vendors which have the status of “approved”. The system will display the option to download the report in a PDF format.</p> <p><b>Constraint(s):</b><br/>There must be at one vendor with at least one procurement item they sell on the system.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>• Head of governance and compliance (Arina van er Merwe) must be logged into the system.</li> <li>• Only Head of governance and compliance (Arina van er Merwe) can generate this report.</li> </ul>   |

|   |                           |
|---|---------------------------|
| <b>Revision date and revision Number:</b> | 04/10/2023<br>Version 3.0 |
| <b>Priority:</b>                          | High                      |

Table 92: 7.1 Generate approved vendor report.

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 7.2  |
| <b>Requirement name (use case name):</b>                 | Generate BEE spend report  |
| <b>Requirement short description:</b>                    | The system should allow to generate a BEE spending report.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The owner, financial department, business enablement department and governance and compliance department requests to generate a BEE spend report. The system will state the “Generated on:” followed by the date that the report was generated on and then the “Period:” followed by the period that the report was based on. The system will generate an overall BEE spend summary table, a MBA BEE spend summary table, a VEN BEE spend summary table, a ENG BEE spend summary table, a MTS BEE spend summary table, a CPT BEE spend summary table, with all table containing columns ranging from 1-8, a non-compliant, and a total column as well as the following rows procurement spend, procurement %, and procurement entitlement. This is followed by a total procurement entitlement BEE level. This data will be retrieved from the system’s database. The system will display the option to the stakeholders to download the report in a structured PDF format.</p> <p><b>Constraint(s):</b></p> |

|   |   |
|---|---|
|   | Only The owner, financial department, business enablement department and governance and compliance department will be able to generate this report. |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>Stakeholders must be logged into the system.</li> </ul>  |
| <b>Revision date and revision Number:</b>             | 04/10/2023<br>Version 3.0   |
| <b>Priority:</b>                                      | High  |

*Table 93: 7.2 Generate BEE spend report.*

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 7.3  |
| <b>Requirement name (use case name):</b>                 | Generate Vendor Spend Report   |
| <b>Requirement short description:</b>                    | The system should allow to generate a vendor spending report.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The financial department and governance and compliance department requests to generate a vendor spend report. The system will generate a table of the vendors with the following Information: Supplier, account code, account name, budget, branch, level, amount spent, count. This will be retrieved from the system's database. The system will display the option to all the previously mentioned stakeholders to download the specified report in a structured PDF format.</p> <p><b>Constraint(s):</b><br/>           Business enablement department will not be able to generate this report.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Users of the financial department and governance and compliance department must be logged out of the system.</li> </ul>   |

|   |                           |
|---|---------------------------|
| <b>Revision date and revision Number:</b> | 04/10/2023<br>Version 3.0 |
| <b>Priority:</b>                          | High                      |

Table 94: 7.3 Generate Vendor Spend Report

| Functional Requirement                                   | Explanation  |
|--|--|
| <b>Requirement number:</b>                               | 7.4  |
| <b>Requirement name (use case name):</b>                 | Generate Consumable Inventory management report  |
| <b>Requirement short description:</b>                    | The system should be able to generate a consumable inventory management report.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The financial department and Business enablement department requests to generate the consumable inventory management report. The system will state the “Generated on:” followed by the date that the report was generated on and then the “Period:” followed by the time period that the report was based on. The system will generate a table with the following columns with the respective information in each column Item name and ranging from January–December, as well as the following rows based on consumable inventory. This is then followed by a bar chart based on consumable inventory on hand per month. The system will display the option to all the previously mentioned stakeholders to download the report in a structured PDF format.</p> <p><b>Constraint(s):</b><br/>           The governance and compliances department will not be able to see this report.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Users of the financial department and Business enablement department must be logged out of the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 04/10/2023<br>Version 3.0  |
| <b>Priority:</b>   | High   |

Table 95: 7.4 Generate Consumable Inventory management report

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 7.5   |
| <b>Requirement name (use case name):</b>                 | Generate Business Unit allocation Report  |
| <b>Requirement short description:</b>                    | The system should allow to generate a business unit allocation spend report   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The financial department, Business enablement department and governance and compliance department requests to generate the business unit allocation report. The system will state the “Department:” followed by the department chosen to do the report on. The system will generate a table about the business unit spending budget category based on that specific department's spending. The system will generate a bar chart that clearly indicates what is the spending report of each budget unit is within the department chosen. The system will display the option to all the previously mentioned stakeholders to download the report in a structured PDF format.</p> <p><b>Constraint(s):</b></p> |

|   |   |
|---|---|
|   | Only the financial department, Business enablement department and governance and compliance department can generate this report   |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>Users of the financial department and Business enablement department and governance and compliance department must be logged out of the system.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 04/10/2023<br>Version 3.0   |
| <b>Priority:</b>                                      | High  |

*Table 96: 7.5 Generate Business Unit allocation report.*

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 7.6   |
| <b>Requirement name (use case name):</b>                 | Generate Budget Variance Report   |
| <b>Requirement short description:</b>                    | The system should allow to generate budget variance report.   |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>           The financial department and Business enablement department requests to generate the budget variance report. The system will state the “Generated on:” followed by the date that the report was generated on and then the “Period:” followed by the time period that the report was based on. The system will generate a table with the following columns with the respective information in each column: Month, budget category, budget amount (R), actual spent (R), variance (R), variance (%). This is followed by a line chart to show total expenses on a month-based period (Month by month for a year). Then followed by a bar chart that shows total expenses on a category-based period (each budget category over a year). The system will display the option to all the previously mentioned stakeholders to download the report in a structured PDF format.</p> <p><b>Constraint(s):</b></p> |

|   |  |
|---|--|
|   | Governance and compliances will not be able to view this report.   |
| <b>Business rules applicable to this Requirement:</b> | <ul style="list-style-type: none"> <li>Users of the financial department and Business enablement department must be logged out of the system.</li> </ul> |
| <b>Revision date and revision Number:</b>             | 04/10/2023<br>Version 3.0  |
| <b>Priority:</b>                                      | High   |

Table 97: 7.6 Generate Budget Variance Report

| Functional Requirement                                   | Explanation   |
|--|---|
| <b>Requirement number:</b>                               | 7.7   |
| <b>Requirement name (use case name):</b>                 | View Generate Report.   |
| <b>Requirement short description:</b>                    | The system should allow the user to view all the reports that they can generate.  |
| <b>Requirement detailed description and constraints:</b> | <p><b>Description:</b><br/>The user wishes to view which reports they would be able to generate. The system will then navigate to the view Generate report page where all the report that that user can generate will be.</p> <p><b>Constraint(s):</b><br/>Users will only be able to view reports that they would be able to generate based on their role's authority.</p> |
| <b>Business rules applicable to this Requirement:</b>    | <ul style="list-style-type: none"> <li>Users must be logged into the system.</li> </ul>   |
| <b>Revision date and revision Number:</b>                | 04/10/2023<br>Version 3.0   |
| <b>Priority:</b>   | High  |

Table 98: UC 7.7 View Generate Report

## 2.5 CONCLUSION

This Concludes the updated requirement list for the Procion system. The updated requirement list with the following subsystems: The user sub-system, Admin sub-system, Procurement sub-system, Finance sub-system, Inventory sub-system, Vendor sub-system and a Report sub-system with the description and explanation of each.



## 2. FUNCTIONAL REQUIREMENT CLIENT SIGN OFF

PROCION SYSTEM: Design and development (Updated Functional Requirements)

Business Name: MOYO Business Advisory

I, Vumile Gumbi, Hereby confirm that I have read the list of updated functional requirements and agree that the information contained therein is complete, accurate and meets the requirements of the project.

**BY SIGNING I CONFIRM THAT:**

1. I have reviewed the changes made to the functional requirements
2. I accept the changes as complete and satisfactory

| Signed by: |   |
|------------|---|
| Name:      | Vumile Gumbi  |
| Date:      | 28/07/2023  |
| Signature: |  |

# 1. TECHNICAL USE CASE DIAGRAM

## 1.1 INTRODUCTION

In this section we provide a visual representation of the technical use case diagrams for each sub system. This clearly states each actor and the role that they play within the Procion system.

## 1.2 USE CASE DIAGRAMS

### 1.2.1 SUBSYSTEM 1 - USER SUBSYSTEM

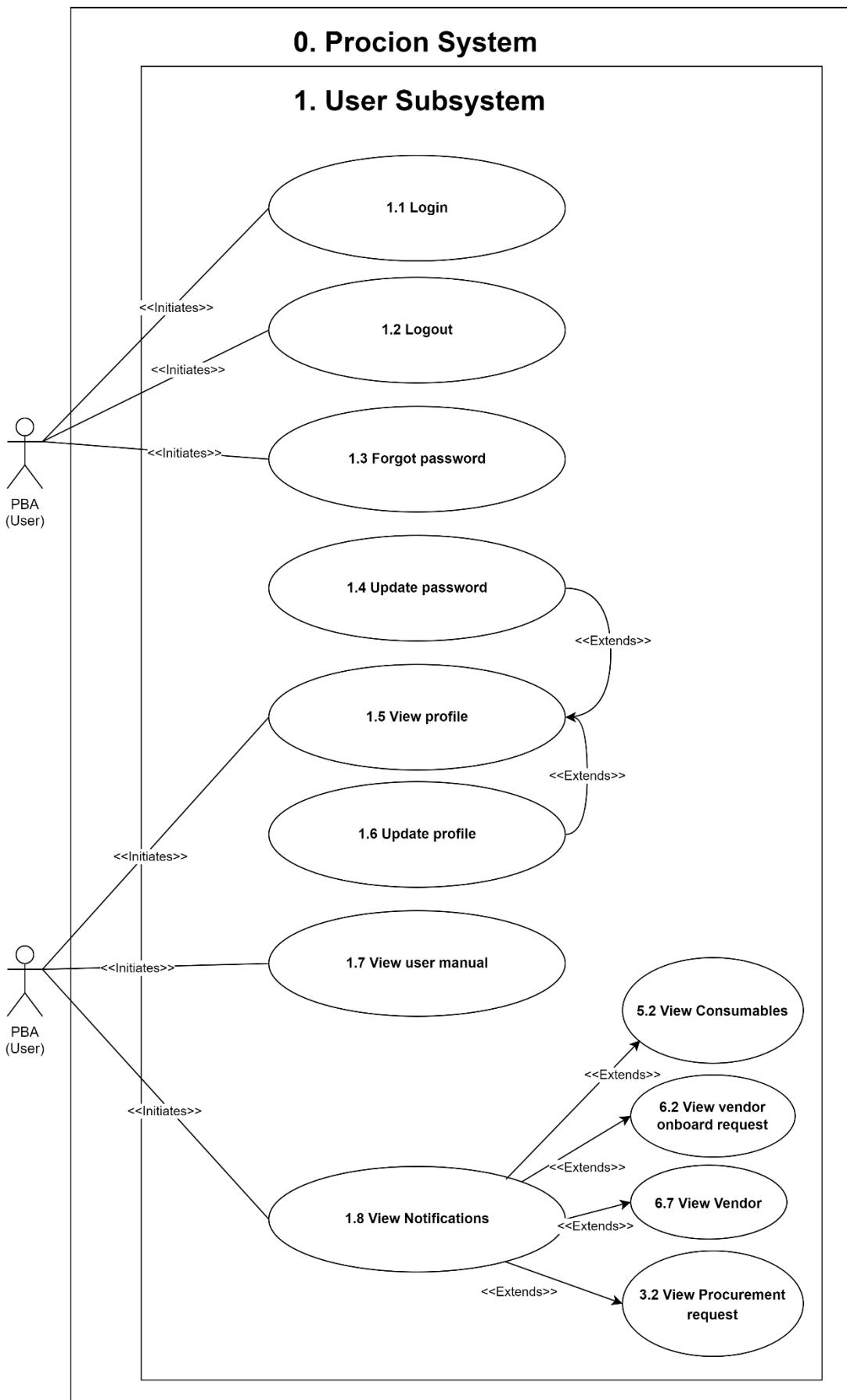


Figure 1 Use Case Diagram - Subsystem 1

## 1.2.2 SUBSYSTEM 2 – ADMIN SUBSYSTEM

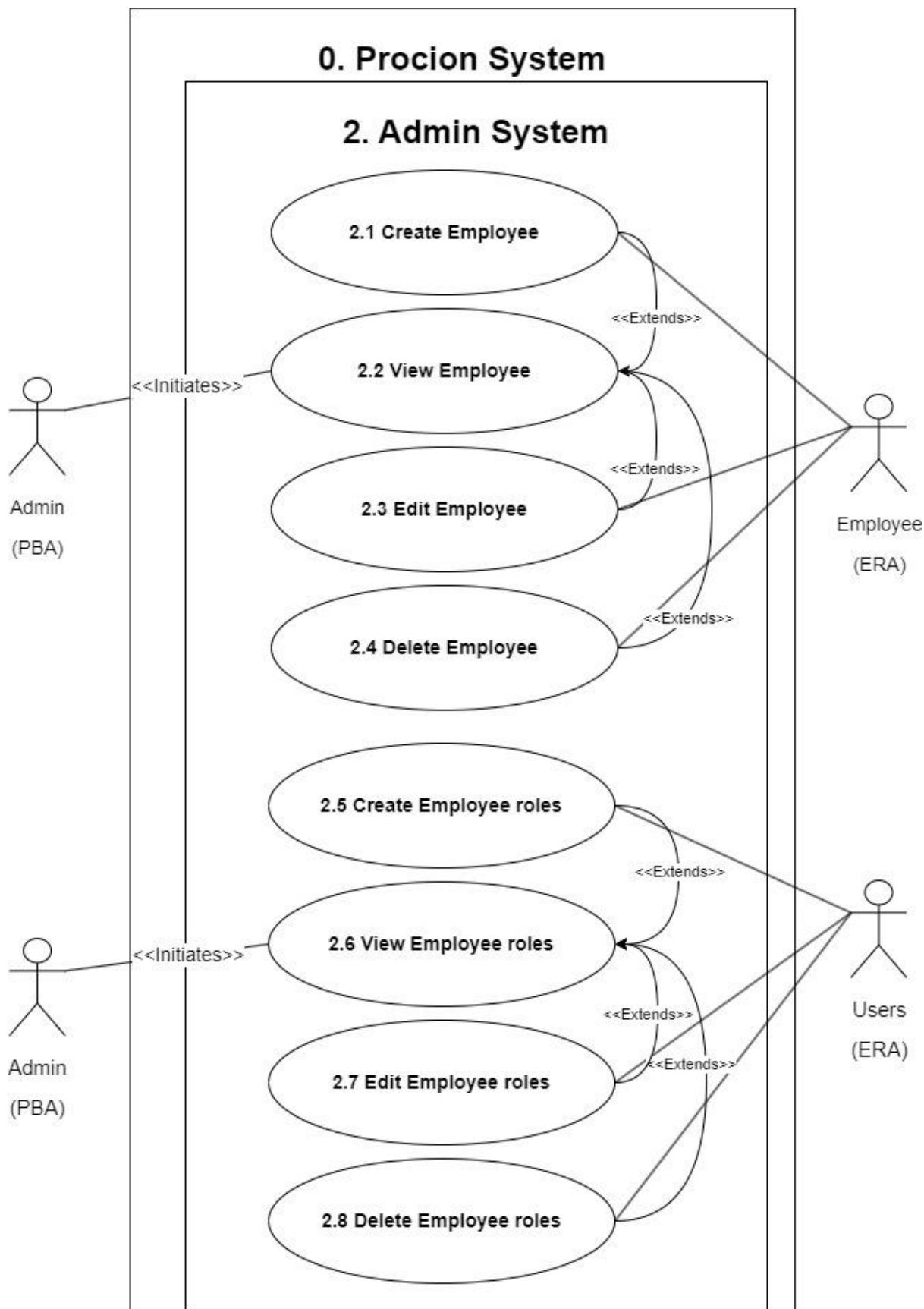


Figure 2 Use Case Diagram - Subsystem 2

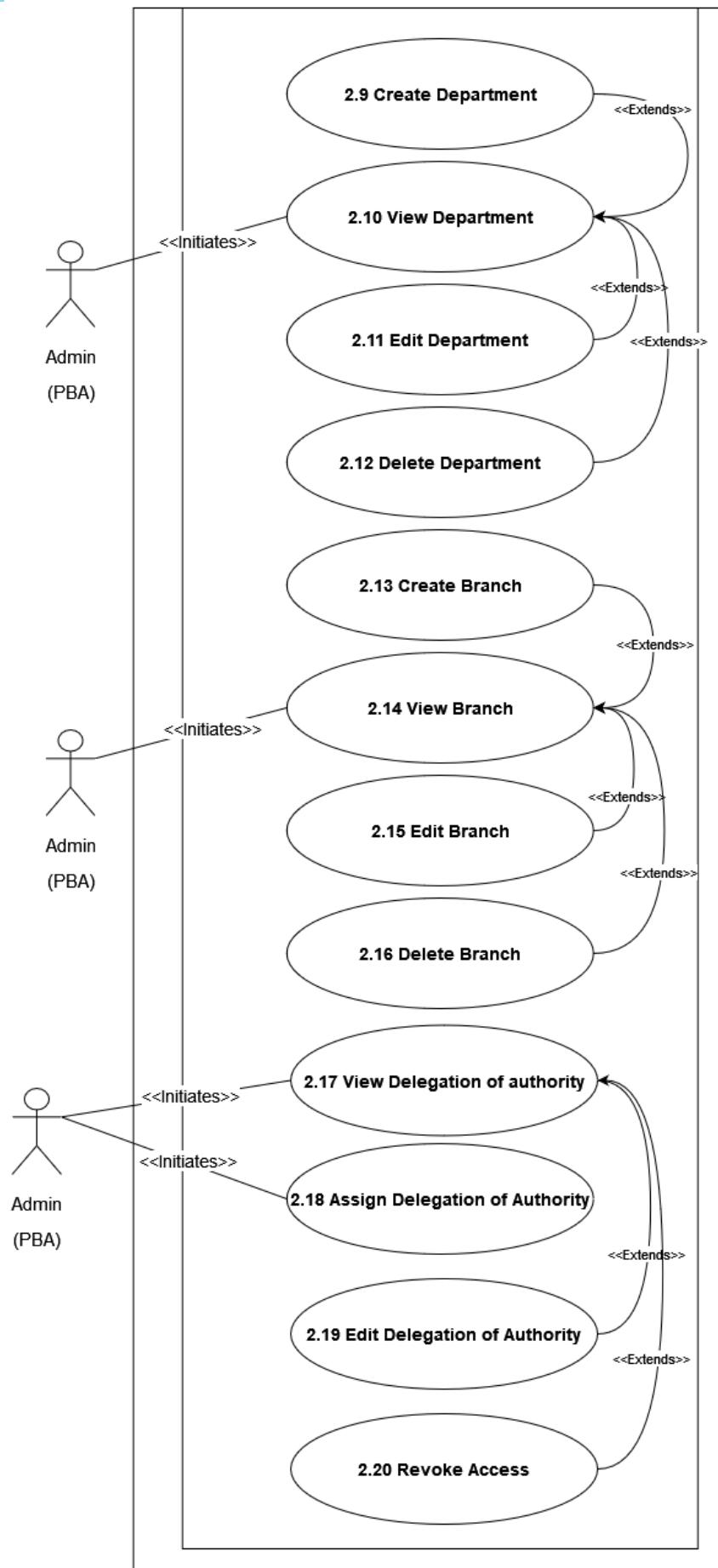


Figure 3 Use Case Diagram - Subsystem 2 Continuation 1

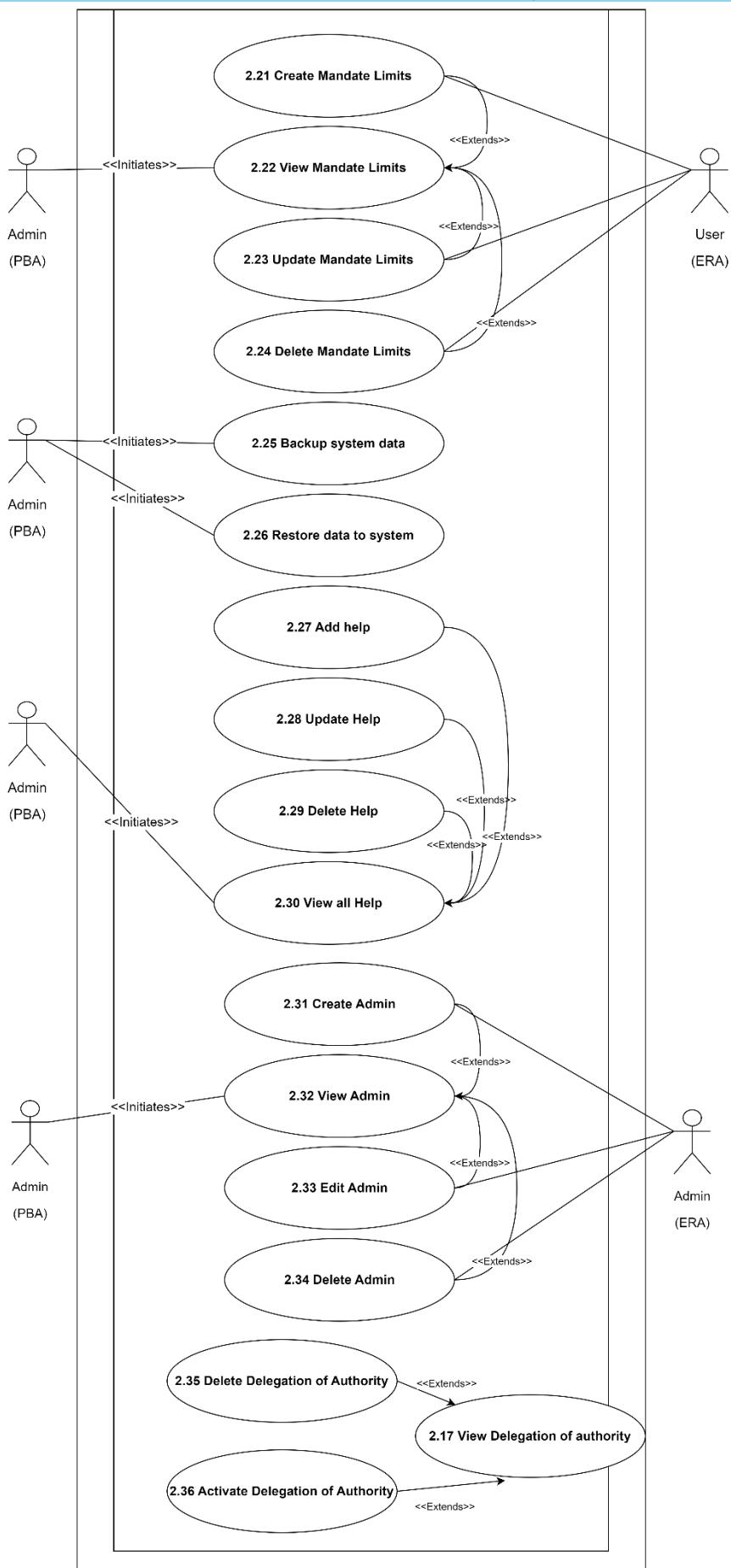


Figure 4 Use Case Diagram - Subsystem 2 Continuation 2

### 1.2.3 SUBSYSTEM 3 - PROCUREMENT SUBSYSTEM

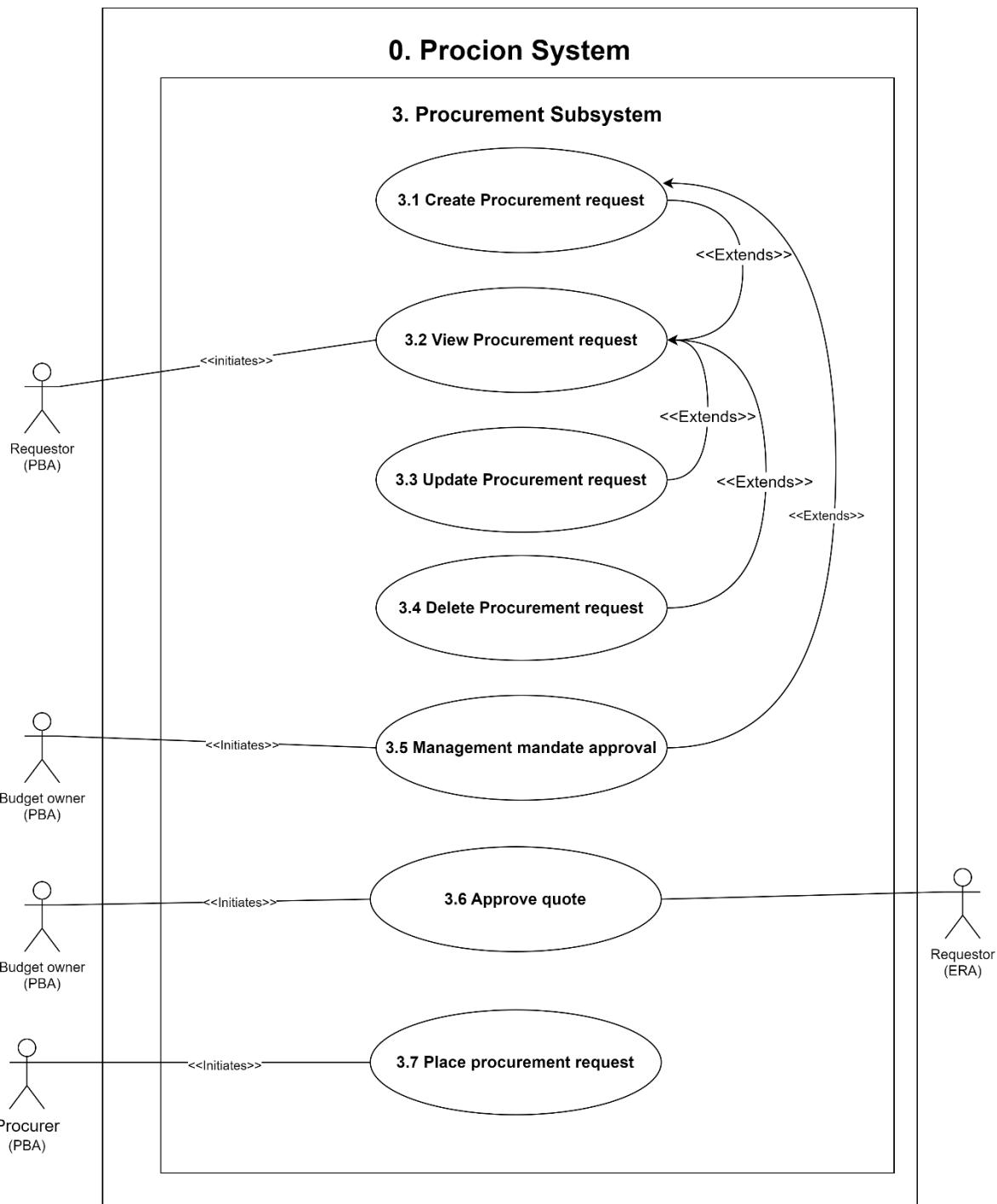


Figure 5 Use Case Diagram - Subsystem 3

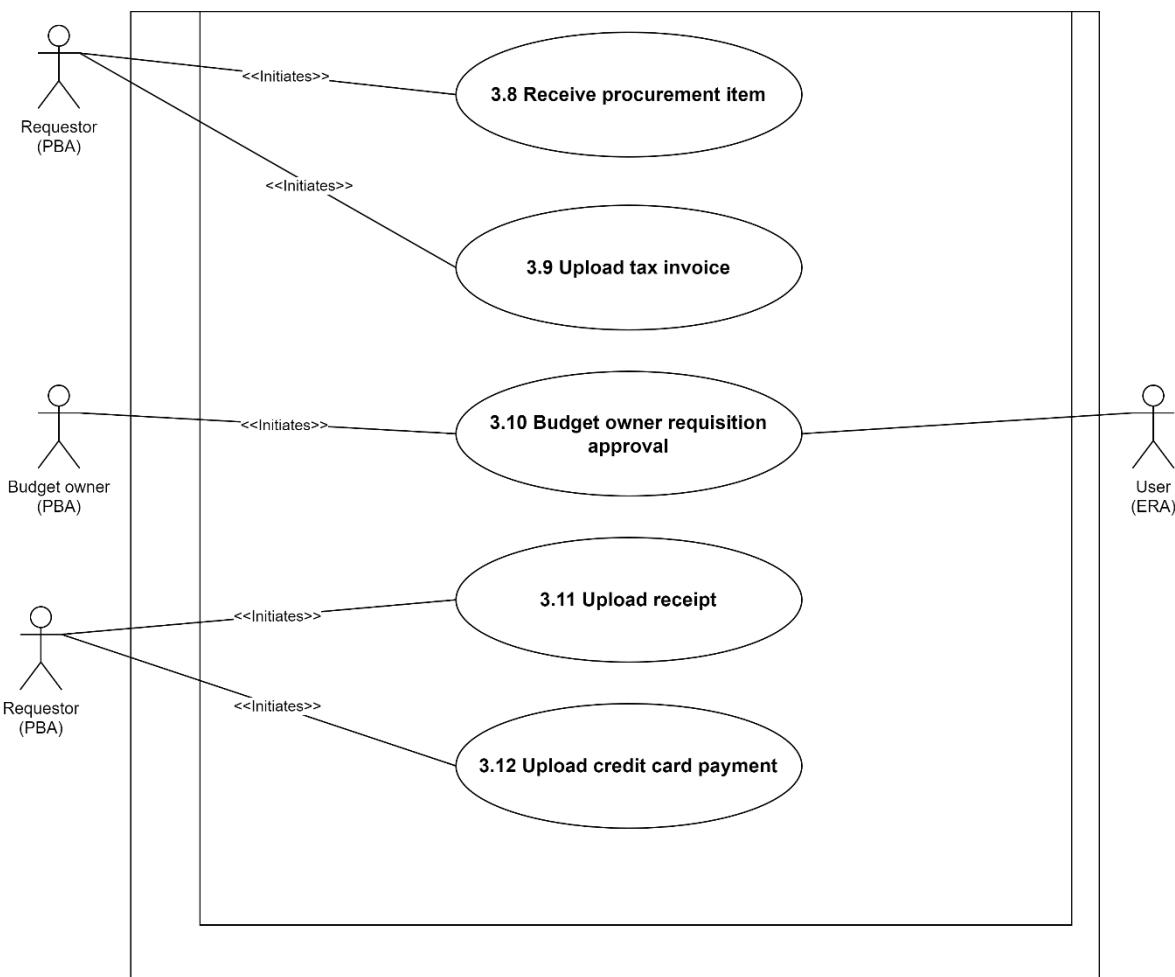
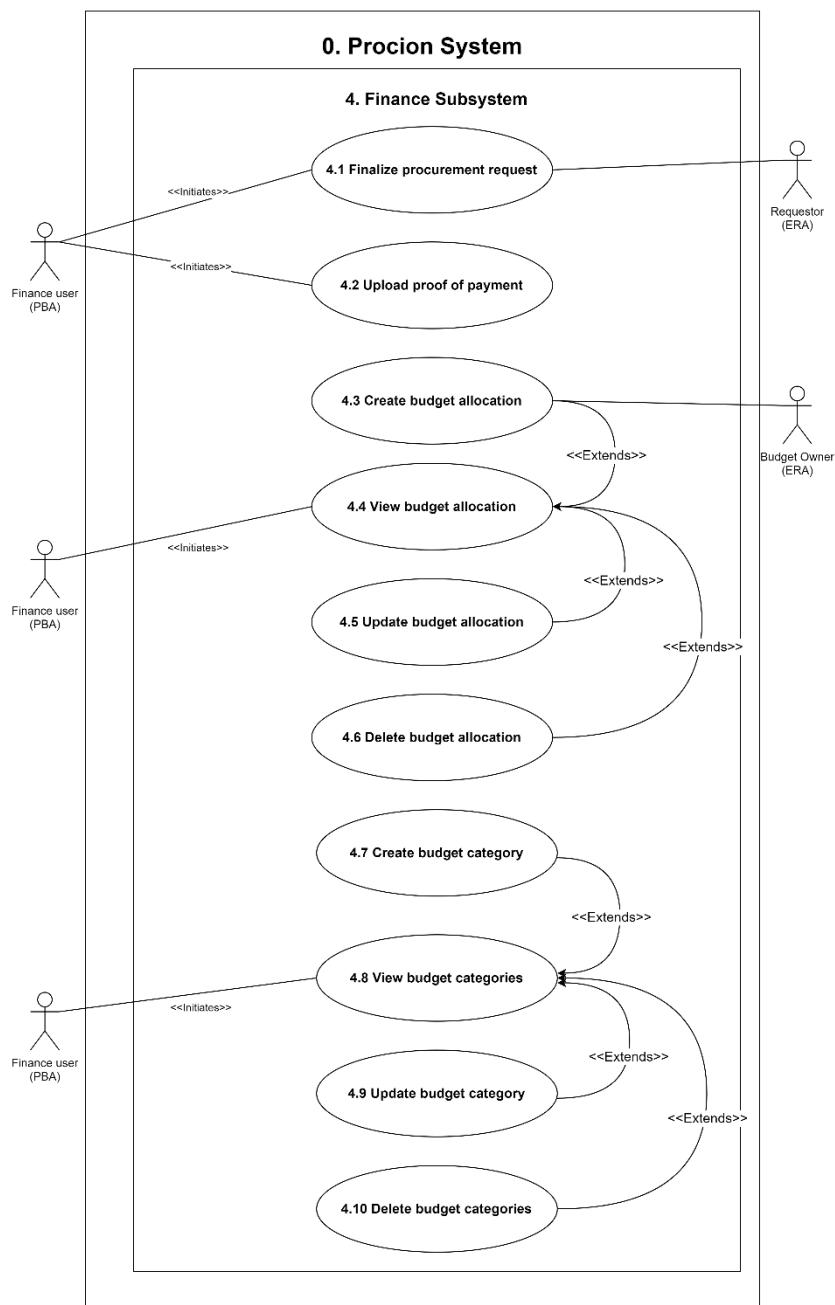
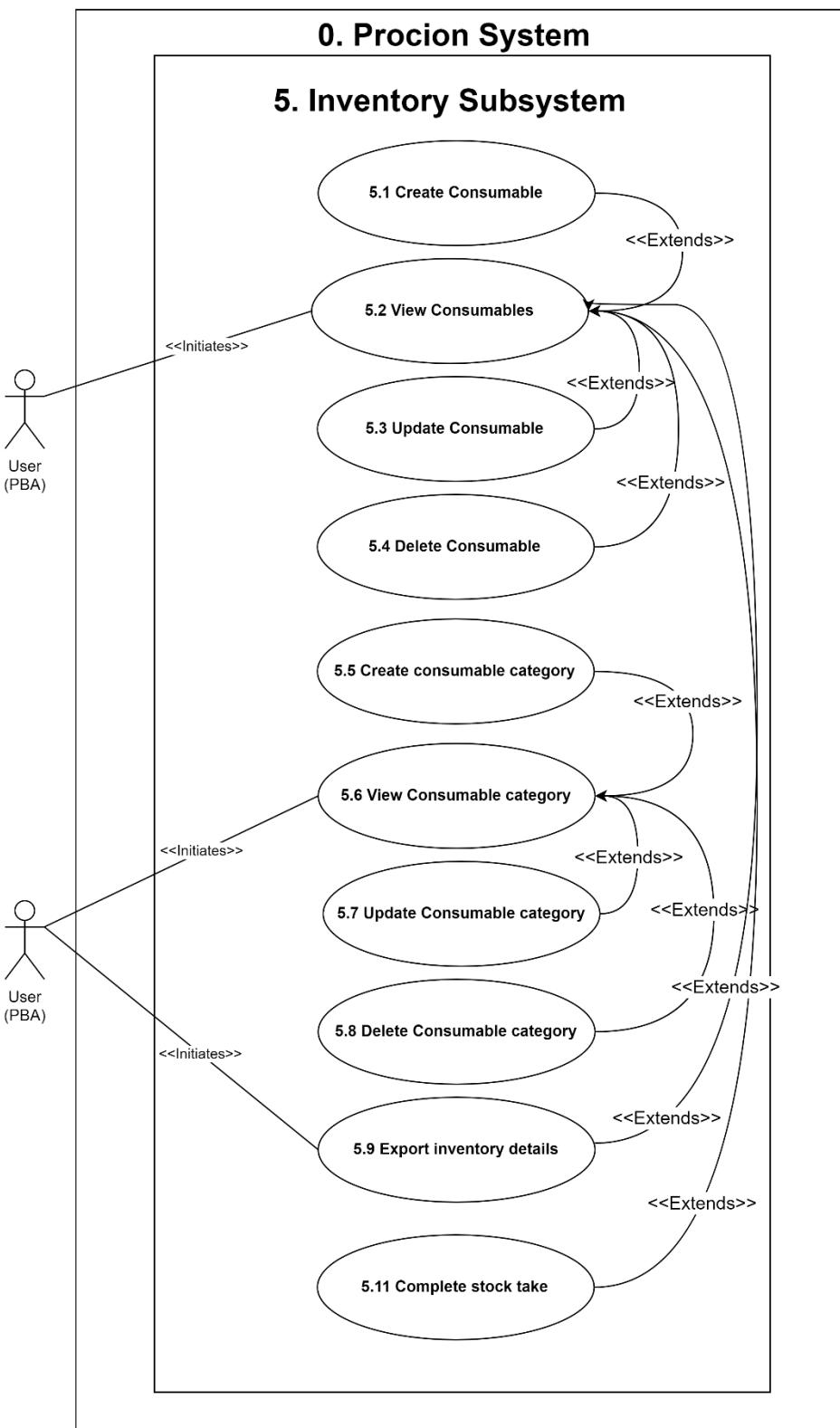


Figure 6 Use Case Diagram - Subsystem 3 Continuation

**1.2.4 SUBSYSTEM 4 - FINANCE SUBSYSTEM**

*Figure 7 Use Case Diagram - Subsystem 4*

**1.2.5 SUBSYSTEM 5 - INVENTORY SUBSYSTEM**

*Figure 8 Use Case Diagram - Subsystem 5*

### 1.2.6 SUBSYSTEM 6 - VENDOR SUBSYSTEM

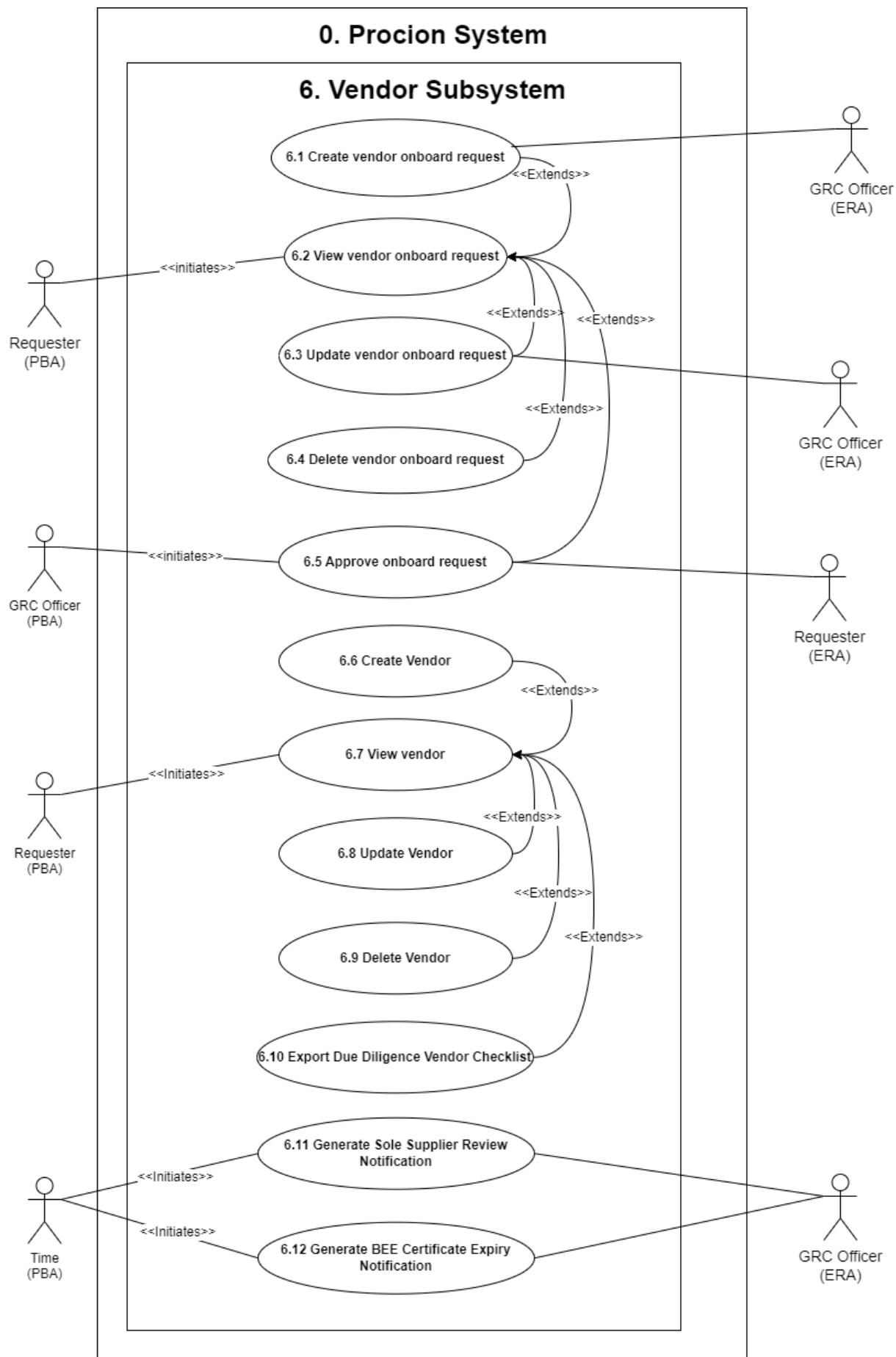


Figure 9 Use Case Diagram - Subsystem 6

### 1.2.7 SUBSYSTEM 7 - REPORTING SUBSYSTEM

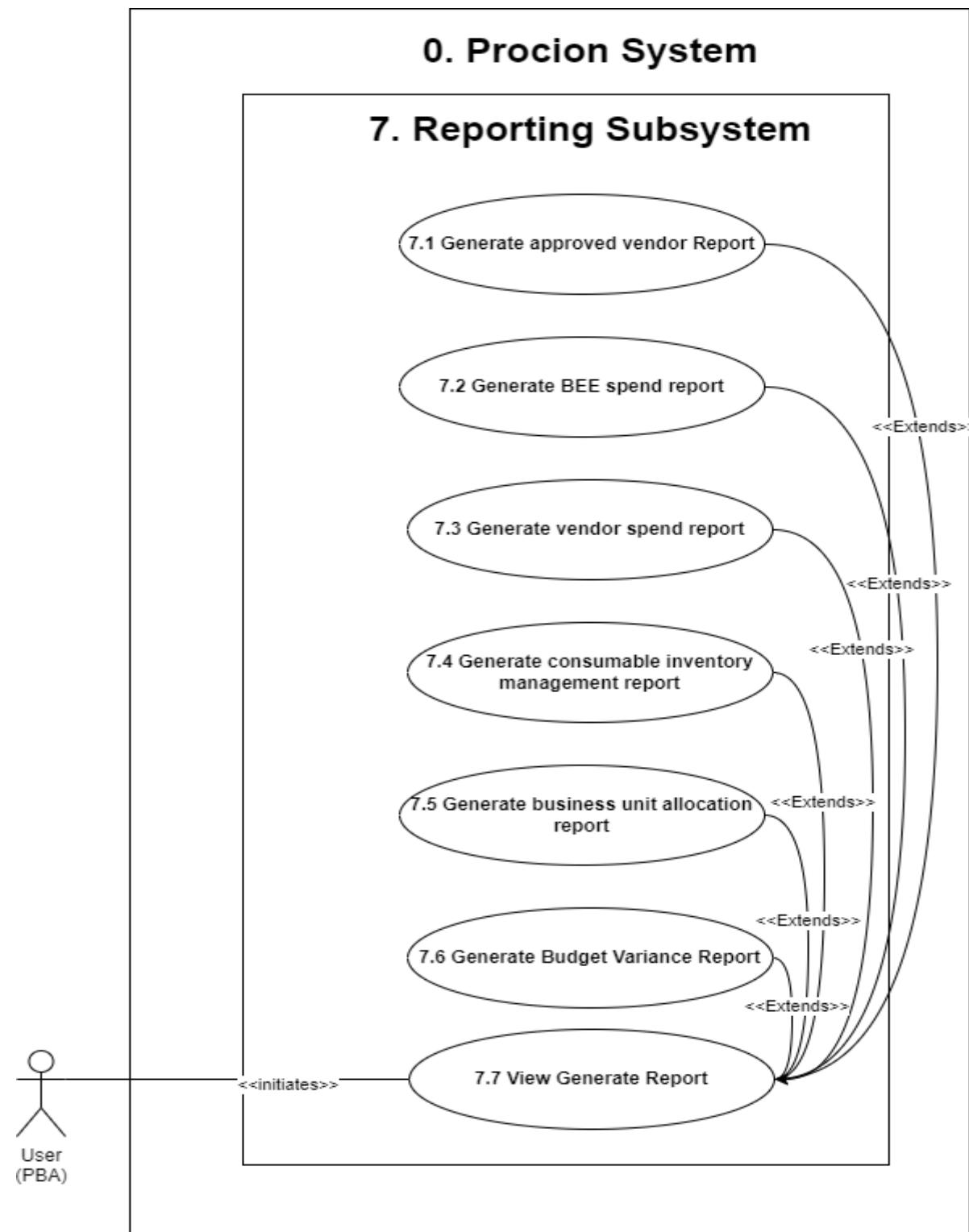


Figure 10 Use Case Diagram - Subsystem 7

### 1.3 CONCLUSION

This concludes the Use Case Diagrams of the Procion system. This will help give a better understanding of how the Procion system will operate.

## 2. LOGICAL USE CASE NARRATIVES

### 2.1 INTRODUCTION

In this section we will discuss the logical narrative for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system.

### 2.2 LOGICAL USE CASE NARRATIVES

#### USE CASE 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35–2.36

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: |                  |              |
| Author: Emil Wonigkeit                     | Date: 23/07/2023 | Version: 1.0 |

| Use case name:                 | Search Profile  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 1.5   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | User  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | This use case describes the event where the user wishes to view their user details on the Procion system. |  |

|                                  | <p>The use case begins when the user requests to view their user details on the system. The system displays the users profile accompanied with their details.</p> <p>The use case concludes once the user views their profile and details</p> |   |
|----------------------------------|---|---|
| <b>Pre-condition:</b>            | The user must be logged into the system.  |   |
| <b>Trigger:</b>                  | The user requests to view their profile on the system   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                  | <p><b>Step 1:</b> The user requests to view their profile on the system.</p>  | <p><b>Step 2:</b> The system retrieves all the employees from the <b>Employee</b> entity as well as the role, department, branch, mandate limit, and username of each employee by using the foreign keys User_Id, Branch_ID, Department_ID, Mandate_ID in the <b>Employee</b> entity.</p> <p>The <b>Employee</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• EmployeeName</li> <li>• EmployeeSurname</li> <li>• Email</li> <li>• CellPhone_Num</li> </ul> <p>The role linked to the employee is retrieved by making use of the foreign key User_Id linked to the primary key User_Id in the <b>User</b> entity, which contains the Role_ID foreign key which is linked to the Role_ID primary key in the <b>Role</b> entity.</p> |

|  |   |
|--|---|
|  | <p>The <b>Role</b> entity contains the following attributes:</p> <ul style="list-style-type: none"><li>• Name</li></ul> <p>The username linked to the employee is retrieved by making use of the foreign key User_Id linked to the primary key User_Id in the <b>User</b> entity.</p> |
|  | <p>The <b>User</b> entity contains the following attributes:</p> <ul style="list-style-type: none"><li>• Username</li></ul>   |
|  | <p>The branch linked to the employee is retrieved by making use of the foreign key Branch_ID linked to the primary key Branch_ID in the <b>Branch</b> entity.</p>   |
|  | <p>The <b>Branch</b> entity contains the following attributes:</p> <ul style="list-style-type: none"><li>• Name</li></ul>   |
|  | <p>The department linked to the employee is retrieved by making use of the foreign key Department_ID linked to the primary key Department _ID in the <b>Department</b> entity.</p>  |
|  | <p>The <b>Department</b> entity contains the following attributes:</p> <ul style="list-style-type: none"><li>• Name</li></ul>   |

|                           |  |  |
|---------------------------|--|--|
|                           |  | <p>The mandate limit linked to the employee is retrieved by making use of the foreign key Mandate_ID linked to the primary key Mandate_ID in the <b>Mandate_Limit</b> entity</p> <p>The <b>Mandate_Limit</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Amount</li> </ul> <p><b>[ALT]</b></p>  |
|                           |  | <p><b>Step 3:</b> The system displays the user's details to the user.</p>  |
| <b>Alternate courses:</b> |  | <p><b>ALT Step 2:</b> The system retrieves all the admins from the <b>Admin</b> entity as well as the role of each admin by using the foreign key User_Id in the <b>Admin</b> entity.</p> <p>The <b>Admin</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• AdminName</li> <li>• AdminSurname</li> <li>• Email</li> <li>• CellPhone_Num</li> </ul> <p>The role linked to the admin is retrieved by making use of the foreign key User_Id linked to the primary key User_Id in the <b>User</b> entity, which contains the Role_ID foreign key which is linked to the Role_ID primary key in the <b>Role</b> entity.</p> <p>The <b>Role</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p><b>Proceed to step 3.</b></p> |
| <b>Conclusion:</b>        |  | The system displays the user's profile.  |

|  |  |
|--|--|
| <b>Post-condition:</b>                               | The system displays the user's profile.  |
| <b>Business rules</b>                                |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires a laptop or desktop.</li> <li>• The user requires internet access.</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 99: 1.5 View Profile

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Emil Wonigkeit      Date: 23/07/2023      Version: 1.0 |                  |              |
| Author: Emil Wonigkeit   | Date: 23/07/2023 | Version: 1.0 |

|                                       |              |  |
|---------------------------------------|--------------|--|
| <b>Use case name:</b>                 | Edit Profile | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 1.6          | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High         | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo         | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User         |  |
| <b>Primary system actor</b>           | None         |  |
| <b>Other participating actors:</b>    | None         |  |
| <b>Other interested stakeholders:</b> | None         |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Description:</b>              | <p>This use case describes the event where the user wishes to edit their profile on the Procion system.</p> <p>The use case begins when the user requests to edit their profile on the system. The system will prompt the user to enter the required profile details into the provided fields. The user will enter the details and submit them, which will then be captured by the system.</p> <p>The use case concludes once the updated user details are added to the <b>User and Employee</b> tables in the database.</p> |   |
| <b>Pre-condition:</b>            | <p>The user must be logged into the system.</p>  |   |
| <b>Trigger:</b>                  | <p>The user requests to edit their profile on the system</p>   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The user requests to edit their profile on the system   | <b>Step 2:</b> The system invokes Use Case 1.5 “Search Profile”.  |
|                                  |  | <b>Step 3:</b> The system will prompt the user to enter the following details: <ul style="list-style-type: none"> <li>• Profile Picture</li> <li>• Name</li> <li>• Surname</li> <li>• Email</li> <li>• Cell Phone Number</li> </ul> |
|                                  | <b>Step 4:</b> The user enters the required details and submits.<br><b>[ALT]</b>   | <b>Step 5:</b> The system validates the submitted data, that all the fields are completed, and the user does not already exist.<br><b>[ALT]</b>   |
|                                  |  | <b>Step 6:</b> The system inserts the entered details into the <b>Employee and User</b> tables.   |

|                           |  |
|---------------------------|--|
|                           | <p><b>Employee:</b></p> <ul style="list-style-type: none"> <li>• EmployeeID</li> <li>• User_Id</li> <li>• Department_ID</li> <li>• Branch_ID</li> <li>• Mandate_ID</li> <li>• EmployeeName</li> <li>• EmployeeSurname</li> <li>• CellPhone_Num</li> <li>• Email</li> </ul> <p><b>User:</b></p> <ul style="list-style-type: none"> <li>• User_Id</li> <li>• Role_ID</li> <li>• Username</li> </ul> <p>The system will auto increment the User_Id and EmployeeID by incrementing the last added ID.</p> <p>The User_Id, Department_ID, Branch_ID and Mandate_ID is used as foreign keys in the <b>Employee</b> entity.</p> <p>The Role_ID is used as a foreign key in the <b>User</b> entity.</p> <p>The system generates a username for the employee.</p> <p>The system displays a success card notification.</p> <p><b>[ALT]</b></p> |
| <b>Alternate courses:</b> | <b>ALT Step 4:</b> The user chooses not to edit their profile. <b>Terminate use case.</b>  |

|                    |  |
|--------------------|--|
|                    | <p><b>ALT Step 5a:</b> The validation of the fields is invalid. The system displays an appropriate error message at the related field. <b>Return to Step 4.</b></p>  |
|                    | <p><b>ALT Step 5b:</b> The user already exists on the system. The system displays an error card notification. <b>Return to Step 4.</b></p>   |
|                    | <p><b>ALT Step 6a:</b> The system failed to update the user on the database. The system displays an appropriate error card notification. <b>Terminate use case.</b></p>  |
|                    | <p><b>ALT Step 6b:</b> The system inserts the entered details into the <b>Admin</b> and <b>User</b> tables.</p> <p><b>Admin:</b></p> <ul style="list-style-type: none"> <li>• Admin_ID</li> <li>• User_Id</li> <li>• AdminName</li> <li>• AdminSurname</li> <li>• CellPhone_Num</li> <li>• Email</li> </ul> <p><b>User:</b></p> <ul style="list-style-type: none"> <li>• User_Id</li> <li>• Role_ID</li> <li>• Username</li> </ul> <p>The User_Id is used as a foreign key in the <b>Admin</b> entity.</p> <p>The Role_ID is used as a foreign key in the <b>User</b> entity.</p> <p>The system generates a username for the admin.</p> <p>The system displays a success card notification</p> |
| <b>Conclusion:</b> | The user has been updated on the system and a success message is displayed.  |

|  |  |
|--|--|
| <b>Post-condition:</b>                               | An existing record has been updated in the <b>User and Employee</b> table.   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires a laptop or desktop.</li> <li>• The user requires internet access.</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 100: 1.6 Edit Profile

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Emil Wonigkeit      Date: 23/07/2023      Version: 1.0 |                  |              |
| Author: Emil Wonigkeit   | Date: 23/07/2023 | Version: 1.0 |

| <b>Use case name:</b>                 | Search Notifications | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|----------------------|--|
| <b>Use case id:</b>                   | 1.8                  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High                 | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo                 | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User                 |  |
| <b>Primary system actor</b>           | None                 |  |
| <b>Other participating actors:</b>    | None                 |  |
| <b>Other interested stakeholders:</b> | None                 |  |

| <b>Description:</b>              | <p>This use case describes the event where the user wishes to view their notifications on the system.</p> <p>The use case begins when the user requests to view their notifications on the system. The system loads and displays all the notifications.</p> <p>The use case concludes once the user views the notifications.</p> |  |
|----------------------------------|--|--|
| <b>Pre-condition:</b>            | <p>The user must be logged into the system.</p>  |  |
| <b>Trigger:</b>                  | <p>The user requests to view their notifications on the system</p>   |  |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>  |
|                                  | <p><b>Step 1:</b> The user requests to view their notifications on the system.</p>   | <p><b>Step 2:</b> The system retrieves all the notifications from the <b>Notification</b> entity as well as the type of each notification by using the foreign key Notification_Type_ID in the <b>Notification</b> entity.</p> <p>The <b>Notification</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Send_Date</li> </ul> <p>The notification type linked to the notification is retrieved by making use of the foreign key Notification_Type_ID linked to the primary key Notification_Type_ID in the <b>Notification_Type</b> entity.</p> <p>The <b>Notification_Type</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |

|  |  |
|--|--|
|  | <ul style="list-style-type: none"> <li>• Description</li> </ul>  |
|  | <b>Step 3:</b> The system displays the notifications to the admin.   |
| <b>Alternate courses:</b>                            | None   |
| <b>Conclusion:</b>                                   | The system displays the users' notifications   |
| <b>Post-condition:</b>                               | The system displays the users' notifications   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires a laptop or desktop.</li> <li>• The user requires internet access.</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

*Table 101: 1.8 View Notifications*

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: |                  |              |
| Author: Emil Wonigkeit                     | Date: 23/07/2023 | Version: 1.0 |

| Use case name: | Search Delegation of Authority | USE CASE TYPE  |
|----------------|--------------------------------|--|
| Use case id:   | 2.17                           | Business Requirements: <input type="checkbox"/>      |
| Priority:      | High                           | System Analysis: <input checked="" type="checkbox"/> |
| Source:        | Moyo                           | System Design: <input type="checkbox"/>              |

|                                       |   |   |
|---------------------------------------|---|---|
| <b>Primary business actor</b>         | Admin   |   |
| <b>Primary system actor</b>           | None  |   |
| <b>Other participating actors:</b>    | None  |   |
| <b>Other interested stakeholders:</b> | None  |   |
| <b>Description:</b>                   | <p>This use case describes the event where the admin wishes to view the delegations on the Procion system.</p> <p>The use case begins when the admin requests to view the delegations on the system. The system loads all the delegations. The admin will enter the username of the delegating employee into the search field. The system will filter the list to match the entered search criteria.</p> <p>The use case concludes once the admin views the searched delegation(s).</p> |   |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.   |   |
| <b>Trigger:</b>                       | The admin requests to view the delegations on the system  |   |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                       | <p><b>Step 1:</b> The admin requests to view the delegations on the system.</p>   | <p><b>Step 2:</b> The system retrieves all the delegations from the <b>Delegation_Of_Authority</b> entity as well as the status and delegate username of each delegation by using the foreign keys User_Id and Delegation_Status_ID in the <b>Delegation_Of_Authority</b> entity.</p> |

|  |  |
|--|--|
|  | <p>The <b>Delegation_Of_Authority</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Delegation_ID</li> <li>• User_Id</li> <li>• Admin_ID</li> <li>• Delegation_Status_ID</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation_Document</li> <li>• DelegatingParty</li> </ul> <p>The username linked to the delegate employee is retrieved by making use of the foreign key User_Id linked to the primary key User_Id in the <b>User</b> entity.</p> <p>The <b>User</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Username</li> </ul> <p>The Delegation_Status linked to the delegation is retrieved by making use of the foreign key Delegation_Status_ID linked to the primary key Delegation_Status_ID in the <b>Delegation_Status</b> entity.</p> <p>The <b>Delegation_Status</b> entity contains the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |
|  | <p><b>Step 3:</b> The system displays the delegations to the admin.</p>  |
|  | <p><b>Step 4:</b> The system prompts the admin to enter the search criteria.</p>   |

|  |  |  |
|--|--|--|
|  | <b>Step 5:</b> The admin enters the search criteria.   | <b>Step 6:</b> The system filters the retrieved delegation array to match the entered search criteria by comparing it to the DelegatingParty attribute.<br><br>The system displays the search results. |
| <b>Alternate courses:</b>                            | None   |  |
| <b>Conclusion:</b>                                   | The system displays the delegations that matched the entered search criteria   |  |
| <b>Post-condition:</b>                               | The system displays the delegations that matched the entered search criteria   |  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• Only the admin can view the delegations</li> </ul>  |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The admin requires a laptop or desktop.</li> <li>• The admin requires internet access.</li> </ul> |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |

*Table 102: 2.17 View Delegation of Authority*

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System:<br><br>Author: Emil Wonigkeit      Date: 23/07/2023      Version: 1.0 |                  |              |
| Author: Emil Wonigkeit   | Date: 23/07/2023 | Version: 1.0 |

| Use case name:                 | Assign Delegation of Authority  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 2.18  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | Admin   |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>This use case describes the event where the admin would like to delegate a user's authority to another user for a set period of time on the Procion system.</p> <p>The use case begins when the admin requests to assign a new delegation of authority. The system will prompt the admin to enter the required details into the provided fields. The admin will enter the details and submit them, which will then be captured by the system.</p> <p>The use case concludes once the new delegation is added to the <b>Delegation_of_Authority</b> and <b>Temporary_Access</b> tables in the database.</p> |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Pre-condition:</b>            | The admin must be logged into the system.                                      |   |
| <b>Trigger:</b>                  | The admin requests to assign a new delegation of authority.                    |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The admin requests to assign a new delegation of authority.     | <b>Step 2:</b> The system invokes Use Case 2.2 “View Employee”.   |
|                                  |  | <b>Step 3:</b> The system will prompt the admin to enter the following details: <ul style="list-style-type: none"> <li>• Delegate Username</li> <li>• From Date</li> <li>• To Date</li> <li>• Delegation Document</li> </ul>  |
|                                  | <b>Step 4:</b> The admin enters the required details and submits.<br><br>[ALT] | <b>Step 5:</b> The system validates the submitted data and that all the fields are completed.<br><br>[ALT]  |
|                                  |  | <b>Step 6:</b> The system saves the document to the backend of the system and saves the file path in the <b>Delegation_Document</b> entity.<br><br>The system inserts the entered details into the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> tables.<br><br><b>Delegation_Of_Authority:</b> <ul style="list-style-type: none"> <li>• Delegation_ID</li> <li>• User_Id</li> <li>• Admin_ID</li> <li>• Delegation_Status_ID</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation_Document</li> <li>• DelegatingParty</li> </ul> |

|                           |   |
|---------------------------|---|
|                           | <p><b>Temporary_Access:</b></p> <ul style="list-style-type: none"> <li>• Temporary_Access_ID</li> <li>• Delegation_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>The system will auto increment the Delegation_ID and Temporary_Access_ID by incrementing the last added ID.</p> <p>The User_Id, Admin_ID and Delegation_Status_ID is used as foreign keys in the <b>Delegation_Of_Authority</b> entity.</p> <p>The Delegation_ID is used as a foreign key in the <b>Temporary_Access</b> entity.</p> <p>The system displays a success card notification.</p> <p><b>[ALT]</b></p> |
| <b>Alternate courses:</b> | <p><b>ALT Step 4:</b> The admin chooses not to assign a new delegation of authority. <b>Terminate use case.</b></p> <p><b>ALT Step 5a:</b> The validation of the fields is invalid. The system displays an appropriate error message at the related field. <b>Return to Step 4.</b></p> <p><b>ALT Step 6:</b> The system failed to add the delegation to the database. The system displays an appropriate error card notification. <b>Terminate use case.</b></p>   |
| <b>Conclusion:</b>        | The delegation has been assigned and added to the system and a success message is displayed.  |

|  |  |
|--|--|
| <b>Post-condition:</b>                               | A new record has been added to the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> tables.                                  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>Only the admin can assign a delegation of authority to the system</li> </ul>                    |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>The admin requires a laptop or desktop.</li> <li>The admin requires internet access.</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |

*Table 103: 2.18 Assign Delegation of Authority*

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Emil Wonigkeit      Date: 23/07/2023      Version: 1.0 |                  |              |
| Author: Emil Wonigkeit   | Date: 23/07/2023 | Version: 1.0 |

|                                       |                              |  |
|---------------------------------------|------------------------------|--|
| <b>Use case name:</b>                 | Edit Delegation of Authority | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 2.19                         | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High                         | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo                         | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | Admin                        |  |
| <b>Primary system actor</b>           | None                         |  |
| <b>Other participating actors:</b>    | None                         |  |
| <b>Other interested stakeholders:</b> | None                         |  |

|                                  |   |   |
|----------------------------------|---|---|
| <b>Description:</b>              | <p>This use case describes the event where the admin wishes to edit a delegation on the Procion system.</p> <p>The use case begins when the admin requests to edit a delegation on the system. The system will prompt the admin to enter the required delegation details into the provided fields. The admin will enter the details and submit them, which will then be captured by the system.</p> <p>The use case concludes once the updated delegation is added to the <b>Delegation_Of_Authority</b> table in the database.</p> |   |
| <b>Pre-condition:</b>            | <p>The admin must be logged into the system.</p>  |   |
| <b>Trigger:</b>                  | <p>The admin requests to edit a delegation on the system</p>  |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                  | <p><b>Step 1:</b> The admin requests to edit a delegation on the system</p>   | <p><b>Step 2:</b> The system invokes Use Case 2.17 “Search Delegation of Authority”.</p>  |
|                                  |   | <p><b>Step 3:</b> The system will prompt the admin to enter the following details:</p> <ul style="list-style-type: none"> <li>• Delegate Username</li> <li>• From Date</li> <li>• To Date</li> <li>• Delegation Document</li> </ul> |
|                                  | <p><b>Step 4:</b> The admin enters the required details and submits.</p> <p>[ALT]</p>   | <p><b>Step 5:</b> The system validates the submitted data and that all the fields are completed.</p> <p>[ALT]</p>   |
|                                  |   | <p><b>Step 6:</b> The system inserts the entered details into the <b>Delegation_Of_Authority</b> table.</p> <p><b>Delegation_Of_Authority:</b></p>  |

|  |   |  |
|--|---|--|
|  |   | <ul style="list-style-type: none"> <li>• Delegation_ID</li> <li>• User_Id</li> <li>• Admin_ID</li> <li>• Delegation_Status_ID</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation_Document</li> <li>• DelegatingParty</li> </ul> <p>The User_Id, Admin_ID and Delegation_Status_ID is used as foreign keys in the <b>Delegation_Of_Authority</b> entity.</p> <p>The system displays a success card notification.</p> <p><b>[ALT]</b></p> |
| <b>Alternate courses:</b>                            | <p><b>ALT Step 4:</b> The admin chooses not to edit a delegation. <b>Terminate use case.</b></p> <p><b>ALT Step 5a:</b> The validation of the fields is invalid. The system displays an appropriate error message at the related field. <b>Return to Step 4.</b></p> <p><b>ALT Step 6:</b> The system failed to update the delegation on the database. The system displays an appropriate error card notification. <b>Terminate use case.</b></p> |  |
| <b>Conclusion:</b>                                   | <p>The delegation has been updated on the system and a success message is displayed.</p>  |  |
| <b>Post-condition:</b>                               | <p>An existing record has been updated in the <b>Delegation_Of_Authority</b> table.</p>   |  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• Only the admin can edit delegations on the system</li> </ul>   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The admin requires a laptop or desktop.</li> <li>• The admin requires internet access.</li> </ul>  |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |

**Open issues:**

- None

Table 104: 2.19 Edit Delegation of Authority

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: |                  |              |
| Author: Emil Wonigkeit                     | Date: 23/07/2023 | Version: 1.0 |

| Use case name:                 | Revoke Access   | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 2.20  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | <ul style="list-style-type: none"> <li>• Time</li> <li>• [ALT] Admin</li> </ul>   |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>This use case describes the event where a user's temporary access is revoked after the delegation period has passed.</p> <p>The use case begins where the system checks the to_Date of a delegation request and compares it to the current date. Should the current date fall outside of the delegation period the system will change the delegation requests status to revoked and remove the temporary_access details from the <b>Temporary_Access</b> entity in the database.</p> |  |

|                                  | The use case concludes once the status of the delegation request has been set to “revoked” and the temporary_access details are removed in the <b>Temporary_Access</b> entities in the database.   |                      |                         |  |   |  |   |  |  |  |   |
|----------------------------------|--|----------------------|-------------------------|--|---|--|---|--|--|--|---|
| <b>Pre-condition:</b>            | A delegation request must be created.<br>The Procion system must be running.<br>The recursive check function must be initiated.  |                      |                         |  |   |  |   |  |  |  |   |
| <b>Trigger:</b>                  | The system runs the recursive check function at 00:00.   |                      |                         |  |   |  |   |  |  |  |   |
| <b>Typical course Of events:</b> | <table border="1"> <thead> <tr> <th><b>Actors Action</b></th><th><b>Systems response</b></th></tr> </thead> <tbody> <tr> <td></td><td><b>Step 1:</b> The system runs the recursive check function at 00:00.</td></tr> <tr> <td></td><td><b>Step 2:</b> The system compares the to_Date attribute, of all the delegations stored on the system, to the current date.</td></tr> <tr> <td></td><td><b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Revoke” status.</td></tr> <tr> <td></td><td><b>Step 4:</b> The system deletes the temporary access linked to the delegation by using the Delegation_ID foreign key.</td></tr> </tbody> </table> | <b>Actors Action</b> | <b>Systems response</b> |  | <b>Step 1:</b> The system runs the recursive check function at 00:00. |  | <b>Step 2:</b> The system compares the to_Date attribute, of all the delegations stored on the system, to the current date. |  | <b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Revoke” status. |  | <b>Step 4:</b> The system deletes the temporary access linked to the delegation by using the Delegation_ID foreign key. |
| <b>Actors Action</b>             | <b>Systems response</b>  |                      |                         |  |   |  |   |  |  |  |   |
|                                  | <b>Step 1:</b> The system runs the recursive check function at 00:00.  |                      |                         |  |   |  |   |  |  |  |   |
|                                  | <b>Step 2:</b> The system compares the to_Date attribute, of all the delegations stored on the system, to the current date.  |                      |                         |  |   |  |   |  |  |  |   |
|                                  | <b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Revoke” status.   |                      |                         |  |   |  |   |  |  |  |   |
|                                  | <b>Step 4:</b> The system deletes the temporary access linked to the delegation by using the Delegation_ID foreign key.  |                      |                         |  |   |  |   |  |  |  |   |
| <b>Alternate courses:</b>        | <p><b>ALT:</b></p> <p><b>Step 1:</b> The admin requests to revoke the temporary access of a delegation.</p> <p><b>Step 2:</b> The system invokes Use Case 2.17 “Search Delegation of Authority”.</p> <p><b>Step 3:</b> The system will prompt the admin to select the delegation whose temporary access they wish to revoke.</p> <p><b>Step 4:</b> The admin selects the delegation whose temporary access should be revoked.</p>  |                      |                         |  |   |  |   |  |  |  |   |

|  |  |
|--|--|
|  | <p><b>Step 5:</b> The system prompts the admin to confirm that he/she wants to revoke the selected delegations temporary access.</p> <p><b>Step 6:</b> The admin confirms the revoke.</p> <p><b>Proceed to step 2.</b></p> |
| <b>Conclusion:</b>                                   | The Delegation_Status has been updated in the <b>Delegation_of_Authority entity</b> and the temporary access has been removed from the <b>Temporary_Access table</b> .   |
| <b>Post-condition:</b>                               | The Delegation_Status has been updated in the <b>Delegation_of_Authority entity</b> and the temporary access has been removed from the <b>Temporary_Access table</b> .   |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>Only the admin can revoke access of delegations</li> </ul>  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>The admin requires a laptop or desktop.</li> <li>The admin requires internet access.</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |

Table 105: 2.20 Revoke Access

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: |                  |              |
| Author: Emil Wonigkeit                     | Date: 23/07/2023 | Version: 1.0 |

|                       |                                |  |
|-----------------------|--------------------------------|--|
| <b>Use case name:</b> | Delete Delegation of Authority | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>   | 2.35                           | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>      | High                           | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>        | Moyo                           | System Design: <input type="checkbox"/>              |

|                                       |  |   |
|---------------------------------------|--|---|
| <b>Primary business actor</b>         | Admin  |   |
| <b>Primary system actor</b>           | None   |   |
| <b>Other participating actors:</b>    | None   |   |
| <b>Other interested stakeholders:</b> | None   |   |
| <b>Description:</b>                   | <p>This use case describes the event where the admin wishes to remove a delegation from the Procion system.</p> <p>The use case begins when the admin requests to remove a delegation from the system. The system will prompt the admin to select the delegation they wish to delete. The admin selects the delegation after which the system will prompt the admin to confirm the deletion. The admin confirms the deletion.</p> <p>The use case concludes once the delegation is removed from the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> tables in the database.</p> |   |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.  |   |
| <b>Trigger:</b>                       | The admin requests to delete a delegation from the system  |   |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                       | <b>Step 1:</b> The admin requests to delete a delegation from the system   | <b>Step 2:</b> The system invokes Use Case 2.17 “Search Delegation of Authority”.             |
|                                       |  | <b>Step 3:</b> The system will prompt the admin to select the delegation they wish to delete. |

|  |  |  |
|--|--|--|
|  | <p><b>Step 4:</b> The admin selects the delegation to be deleted.</p> <p><b>[ALT]</b></p>  | <p><b>Step 5:</b> The system prompts the admin to confirm that he/she wants to delete the selected delegation.</p>   |
|  | <p><b>Step 6:</b> The admin confirms the deletion.</p> <p><b>[ALT]</b></p>   | <p><b>Step 7:</b> The temporary access linked to the delegation will be deleted by using the foreign key Delegation_ID in the <b>Temporary_Access</b> table.</p> <p>The system finally deletes the delegation from the <b>Delegation_Of_Authority</b> table using the Delegation_ID.</p> <p><b>[ALT]</b></p> |
| <b>Alternate courses:</b>  |  | <p><b>ALT Step 4:</b> The admin chooses not to delete a delegation. <b>Terminate use case.</b></p>   |
| <b>ALT Step 6:</b> The admin chooses not to delete a delegation. <b>Terminate use case.</b>  |  |  |
| <b>ALT Step 7:</b> The system failed to delete the delegation from the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> table. An error card notification is displayed. <b>Terminate Use Case.</b> |  |  |
| <b>Conclusion:</b>   | The delegation has been removed from the system and a success message is displayed.  |  |
| <b>Post-condition:</b>   | The selected delegation details have been removed from the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> table.           |  |
| <b>Business rules</b>  | <ul style="list-style-type: none"> <li>Only the admin can remove delegations from the system</li> </ul>                                |  |
| <b>Implementation constraints and specifications</b>   | <ul style="list-style-type: none"> <li>The admin requires a laptop or desktop.</li> <li>The admin requires internet access.</li> </ul> |  |
| <b>Assumptions:</b>  | . None   |  |

|              |        |
|--------------|--------|
| Open issues: | • None |
|--------------|--------|

Table 106: 2.35 Delete Delegation of Authority

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: |                  |              |
| Author: Emil Wonigkeit                     | Date: 23/07/2023 | Version: 1.0 |

| Use case name:                 | Activate Delegation of Authority  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 2.36  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | Time  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>This use case describes the event where a delegation request is set to active.</p> <p>The use case begins where the system checks the from_Date of a delegation request and compares it to the current date. Should the current date fall inside of the delegation period the system will change the delegation requests status to Active.</p> |  |

|  | The use case concludes once the status of the delegation request has been set to “Active”.   |                      |                         |  |   |  |   |  |  |
|--|--|----------------------|-------------------------|--|---|--|---|--|--|
| <b>Pre-condition:</b>                                | A delegation request must be created.<br>The Procion system must be running.<br>The recursive check function must be initiated.  |                      |                         |  |   |  |   |  |  |
| <b>Trigger:</b>                                      | The system runs the recursive check function at 00:00.   |                      |                         |  |   |  |   |  |  |
| <b>Typical course Of events:</b>                     | <table border="1"> <thead> <tr> <th><b>Actors Action</b></th><th><b>Systems response</b></th></tr> </thead> <tbody> <tr> <td></td><td><b>Step 1:</b> The system runs the recursive check function at 00:00.</td></tr> <tr> <td></td><td><b>Step 2:</b> The system compares the from_Date attribute, of all the delegations stored on the system, to the current date.</td></tr> <tr> <td></td><td><b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Active” status.</td></tr> </tbody> </table> | <b>Actors Action</b> | <b>Systems response</b> |  | <b>Step 1:</b> The system runs the recursive check function at 00:00. |  | <b>Step 2:</b> The system compares the from_Date attribute, of all the delegations stored on the system, to the current date. |  | <b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Active” status. |
| <b>Actors Action</b>                                 | <b>Systems response</b>  |                      |                         |  |   |  |   |  |  |
|  | <b>Step 1:</b> The system runs the recursive check function at 00:00.  |                      |                         |  |   |  |   |  |  |
|  | <b>Step 2:</b> The system compares the from_Date attribute, of all the delegations stored on the system, to the current date.  |                      |                         |  |   |  |   |  |  |
|  | <b>Step 3:</b> The system updates the Delegation_Status_ID attribute in the Delegation_of_Authority entity to the ID of the “Active” status.   |                      |                         |  |   |  |   |  |  |
| <b>Alternate courses:</b>                            | None   |                      |                         |  |   |  |   |  |  |
| <b>Conclusion:</b>                                   | The Delegation_Status has been updated in the <b>Delegation_of_Authority entity</b>  |                      |                         |  |   |  |   |  |  |
| <b>Post-condition:</b>                               | The Delegation_Status has been updated in the <b>Delegation_of_Authority entity</b>  |                      |                         |  |   |  |   |  |  |
| <b>Business rules</b>                                | None   |                      |                         |  |   |  |   |  |  |
| <b>Implementation constraints and specifications</b> | None   |                      |                         |  |   |  |   |  |  |
| <b>Assumptions:</b>                                  | • None   |                      |                         |  |   |  |   |  |  |

|              |        |
|--------------|--------|
| Open issues: | • None |
|--------------|--------|

Table 107: 2.36 Activate Delegation of Authority

### USE CASE 1.7, 2.25 & 2.27-2.30

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 1 User |                  |              |
| Author: Leon Combrinck                            | Date: 2023/07/10 | Version: 1.0 |

| Use case name:                 | View User Manual  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 1.7   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor:        | User (PBA)  |  |
| Primary system actor:          | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | In this use case the User would like to view the User Manual. The User will thus request the system to view the User Manual. The system will then open/download the User Manual file as a PDF on the Users device allowing the user to view the manual. |  |
| Pre-condition:                 | The User must be logged into the system.  |  |

|  |   |  |
|--|---|--|
| <b>Trigger:</b>                                      | The User requests to View the User Manual.  |  |
| <b>Typical course Of events:</b>                     | <b>Actor Action</b>   | <b>System Response</b>   |
|  | <b>Step 1:</b> The admin requests to View the User Manual.  | <b>Step 2:</b> The system opens the User Manual PDF on the user's browser on an alternative tab allowing the user to view the Manual |
| <b>Alternate courses:</b>                            | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |
| <b>Conclusion:</b>                                   | The User Manual is open on the user's browser available for viewing.  |  |
| <b>Post-condition:</b>                               | The users are provided with the user manual that will help explain certain parts of the system and how to work with them. |  |
| <b>Business rules</b>                                | All users of the system are allowed to View the user's manual.  |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |

Table 108: 1.7 View User Manual

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                       |                    |  |
|-----------------------|--------------------|--|
| <b>Use case name:</b> | Backup system data | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>   | 2.25               | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>      | High               | System Analysis: <input checked="" type="checkbox"/> |

|                                       |   |   |
|---------------------------------------|---|---|
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>   |
| <b>Primary business actor:</b>        | Admin (PBA)   |   |
| <b>Primary system actor:</b>          | None  |   |
| <b>Other participating actors:</b>    | None  |   |
| <b>Other interested stakeholders:</b> | None  |   |
| <b>Description:</b>                   | <p>In the use case the admin will request the system to make a backup of the systems data. The admin will thus request the make a backup on the system. The system will then request confirmation. The system will then make a backup of the current data on the system and save the backup file with timestamp on the admins device after the confirmation has been received. The system will then display a successful backup notification.</p> |   |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.   |   |
| <b>Trigger:</b>                       | The admin requests to back-up the system data.  |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b><br><b>Step 1:</b> Admin requests to back-up the system data.<br><br><b>Step 3:</b> The Admin Provides Confirmation to create system backup.<br><br><b>[ALT]</b>   | <b>System Response</b><br><b>Step 2:</b> The system will request Backup Confirmation from the Admin.<br><br><b>Step 4:</b> The system retrieves the data form the system database and will create a backup of the current data on the system and store this backed-up file with file path and timestamp of when the backup was made on the admins device. |

|  |  |
|--|--|
|  | <b>Step 5:</b> The system will display a success notification regarding the successful backup to the admin.<br><br>[ALT]   |
| <b>Alternate courses:</b>                            | <b>ALT-Step 3:</b> The Admin denies Backup Confirmation. The use case ends.<br><br><b>ALT-Step 5:</b> The system displays a unsuccessful notification regarding the unsuccessful backup to the admin. The use case ends. |
| <b>Conclusion:</b>                                   | A backup of the system data will be made, and a success notification is displayed.   |
| <b>Post-condition:</b>                               | A Backed-up file of the system data will be on the admins device.  |
| <b>Business rules</b>                                | Only the admin can backup the data on the system.  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 109: 2.25 Backup System Data

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                       |          |                      |
|-----------------------|----------|----------------------|
| <b>Use case name:</b> | Add Help | <b>USE CASE TYPE</b> |
|-----------------------|----------|----------------------|

|                                       |   |   |
|---------------------------------------|---|---|
| <b>Use case id:</b>                   | 2.27  | Business Requirements: <input type="checkbox"/>                     |
| <b>Priority:</b>                      | Medium  | System Analysis: <input checked="" type="checkbox"/>                |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>                             |
| <b>Primary business actor:</b>        | Admin (PBA)   |   |
| <b>Primary system actor:</b>          | None  |   |
| <b>Other participating actors:</b>    | None  |   |
| <b>Other interested stakeholders:</b> | None  |   |
| <b>Description:</b>                   | <p>In this use case the admin would like to add a help section. The admin will thus request the system to add a help section. The system will then prompt the admin to enter all the help fields required. The admin will then enter all the required fields and save the entered fields.</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will capture the information and the newly created help section will be added and displayed to the <b>help</b> table in the database.</p> |   |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.   |   |
| <b>Trigger:</b>                       | The admin requests to add Help.   |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>   | <b>System Response</b>  |
|                                       | <b>Step 1:</b> Admin requests to add a new help section on the system.  | <b>Step 2:</b> The system invokes the use case 2.30 “View all Help” |

|                           |   |   |
|---------------------------|---|---|
|                           |   | <p><b>Step 3:</b> System will prompt the admin to enter the required fields with the following details:</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul>   |
|                           | <p><b>Step 4:</b> The Admin will enter the required fields and saves the details.</p> <p>[ALT]</p>  | <p><b>Step 5:</b> The system validates the details that was entered to ensure everything is correct.</p> <p>[ALT]</p>   |
|                           |   | <p><b>Step 6:</b> The system will add the entered help details in the <b>Help</b> table with the following details:</p> <p><b>Help</b></p> <ul style="list-style-type: none"> <li>• HelpID</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p><b>Help Category</b></p> <ul style="list-style-type: none"> <li>• Category</li> </ul> <p>The system will automatically increment the HelpID (increase by 1) to ensure each Help has a unique ID.</p> <p>[ALT]</p> |
|                           |   | <p><b>Step 7:</b> The system will then display a successful notification to the admin.</p>  |
| <b>Alternate courses:</b> | <p><b>ALT-Step 4:</b> The admin can alternatively request to not save the field but rather cancel. <b>System will Terminates usecase.</b></p> <p><b>ALT-Step 5:</b> The validation of the field is invalid this will display an invalid inputs message to the admin. <b>System will return to step 4.</b></p> |   |

|  |  |
|--|--|
|  | <b>ALT-Step 6:</b> The system validates and finds that the to be added Help details is a duplicate in the <b>Help</b> table. System will display a Duplicate Help message to the admin. <b>System will return to step 4.</b> |
| <b>Conclusion:</b>                                   | The newly created Help will be added to the system and a success notification is displayed.  |
| <b>Post-condition:</b>                               | A new record is created in the <b>Help</b> table.  |
| <b>Business rules</b>                                | Only the admin can Add Help to the system.   |
| <b>Implementation constraints and specifications</b> | No duplicate helps allowed.  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 110: 2.27 Add Help

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                                |             |  |
|--------------------------------|-------------|--|
| <b>Use case name:</b>          | Update Help | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>            | 2.28        | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>               | medium      | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                 | Moyo        | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b> | Admin (PBA) |  |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the admin would like to update a help section. The admin will thus request the system to update a help. The system will then prompt the admin to select the specific help fields they would like to update. The admin will then enter all the required fields and save the entered fields.</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will capture the updated information and the newly updated Help will be added and displayed to the <b>Help</b> table in the database.</p> |  |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.  |  |
| <b>Trigger:</b>                       | The admin requests to update a Help section.   |  |
| <b>Typical course Of events:</b>      | <b>Actor Action</b><br><b>Step 1:</b> The admin requests to update a help section on the system.   | <b>System Response</b><br><b>Step 2:</b> The system invokes the use case 2.30 “View all Help”  |
|                                       |  | <b>Step 3:</b> System will prompt the admin to enter the required fields with the following details: <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> |

|  |  |   |
|--|--|---|
|  | <p><b>Step 4:</b> The Admin will enter the required fields and saves the details.</p> <p>[ALT]</p> | <p><b>Step 5:</b> The system validates the details that was entered to ensure everything is correct.</p> <p>[ALT]</p>   |
|  |  | <p><b>Step 6:</b> The system will update the entered help details in the <b>Help</b> table with the following details:</p> <p><b>Help</b></p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p><b>Help Category</b></p> <ul style="list-style-type: none"> <li>• Category</li> </ul> <p>[ALT]</p> |
|  |  | <p><b>Step 7:</b> The system will then display a successful notification to the admin.</p>  |
| <b>Alternate courses:</b>  |  | <p><b>ALT-Step 4:</b> The admin can alternatively request to not save the field but rather cancel. <b>System will Terminates usecase.</b></p>   |
| <b>ALT-Step 5:</b> The validation of the field is invalid this will display an invalid inputs message to the admin. <b>System will return to step 4.</b>                                       |  |   |
| <b>ALT-Step 6:</b> The system fails to update the Help details on the <b>Help</b> table. System will display a failed notification message to the admin. <b>System will terminate usecase.</b> |  |   |
| <b>Conclusion:</b>   | The updated Help details will be updated on the system and a success notification is displayed.    |   |
| <b>Post-condition:</b>   | A record is updated in the <b>Help</b> table.  |   |
| <b>Business rules</b>  | Only the admin can update Help to the system.  |   |

|  |                                     |
|--|-------------------------------------|
| <b>Implementation constraints and specifications</b> | No duplicate Help sections allowed. |
| <b>Assumptions:</b>                                  | • None                              |
| <b>Open issues:</b>                                  | • None                              |

Table 111: 2.28 Update Help

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

| <b>Use case name:</b>                 | Delete Help   | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|---|--|
| <b>Use case id:</b>                   | 2.29  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | medium  | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | Admin (PBA)   |  |
| <b>Primary system actor:</b>          | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | In this use case the admin would like to delete a Help section. The admin will thus request the system to delete help. The system will then prompt the admin to the specific help they would like to delete. The admin then selects the help they would like to delete. The system then prompts the admin to confirm if they really do want to delete that specific help. The admin confirms the request, |  |

|                                  | and the system deletes that help from the <b>Help</b> table on the system.  |   |
|----------------------------------|---|---|
| <b>Pre-condition:</b>            | The admin must be logged into the system.   |   |
| <b>Trigger:</b>                  | The admin requests to delete a Help.  |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>   | <b>System Response</b>  |
|                                  | <b>Step 1:</b> The admin requests to delete a Help section on the system.   | <b>Step 2:</b> The system invokes the use case 2.30 “View all Help”   |
|                                  |   | <b>Step 3:</b> The system prompts the admin to select the help they would like to delete.   |
|                                  | <b>Step 4:</b> The admin selects the help they would like to delete.<br><br>[ALT]   | <b>Step 5:</b> The system prompts the admin to confirm the deletion of that specific help.  |
|                                  | <b>Step 6:</b> The admin confirms the deletion of that specific help.<br><br>[ALT]  | <b>Step 7:</b> The system then removes that help form the system in the <b>Help</b> table.<br><br>A Successful deletion notification will display for the admin notifying them of the successful deletion.<br><br>[ALT] |
| <b>Alternate courses:</b>        | <b>ALT-Step 4:</b> The admin does not want to delete the specified department. <b>Terminate use case.</b>   |   |
|                                  | <b>ALT-Step 6:</b> The admin does not want to delete the specified help thus the admin would rather cancel the deletion. <b>Terminate use case.</b> |   |

|  |  |
|--|--|
|  | <b>ALT-Step 7a:</b> The system failed to delete the help from <b>Help</b> table. A failed deletion notification will display for the admin notifying them of the failed deletion. <b>Terminate use case.</b> |
| <b>Conclusion:</b>                                   | The help has been deleted of the system and successful deletion notification displays.   |
| <b>Post-condition:</b>                               | The requested deleted field has been removed from the <b>Help</b> table.   |
| <b>Business rules</b>                                | Only the admin can delete help on the system   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 112: 2.29 Delete Help

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                                |               |  |
|--------------------------------|---------------|--|
| <b>Use case name:</b>          | View All Help | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>            | 2.30          | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>               | medium        | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                 | Moyo          | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b> | User (PBA)    |  |
| <b>Primary system actor:</b>   | None          |  |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the User would like to view all the helps on the system. The User will request the system to view all the help on the system. The system will then load all the helps that are currently on the system and display them on the help table with the following details:</p> <ul style="list-style-type: none"> <li>• HelpID</li> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will then prompt the user to enter the search field and at the search bar. The user will then enter the search field at the search bar. The system will then filter the list of helps based on the users search criteria. The system will then display the list of the searched criteria. The user will then request the system to view the help video. The system will then display the help video according to the user's request.</p> |  |
| <b>Pre-condition:</b>                 | The user must be logged into the system.   |  |
| <b>Trigger:</b>                       | The user requests to view all help.  |  |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b>   |
|                                       | <p><b>Step 1:</b> The user requests to view all the helps on the system.</p>   | <p><b>Step 2:</b> The system will retrieve all the helps from the <b>Help</b> table and <b>Help_Category</b> table with the following details:</p> <ul style="list-style-type: none"> <li>• HelpID</li> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> |

|                           |  |   |
|---------------------------|--|---|
|                           |  | The system will display all the retrieved Help details in the <b>Help</b> table.  |
|                           |  | <b>Step 3:</b> The system will prompt the user to enter the search criteria.  |
|                           | <b>Step 4:</b> The user will enter their search criteria and searches.<br><br>[ALT]  | <b>Step 5:</b> The system then filters through the retrieved help list to match the help list with the search criteria of the user by comparing the following attributes: <ul style="list-style-type: none"> <li>• Name</li> </ul> The system then displays the matching fields.<br><br>[ALT] |
|                           |  | <b>Step 6:</b> The system will prompt the user to view the help video and/or user manual.   |
|                           | <b>Step 7:</b> The user will request to view the video.<br><br>[ALT]   | <b>Step 8:</b> The system will display the help video according to video they requested.  |
| <b>Alternate courses:</b> | <b>ALT-Step 4:</b> The user does not enter any search criteria. The use case proceeds to step 6.   |   |
|                           | <b>ALT-Step 5:</b> The search criteria does not match with anything in the help list since there is no such help data on the system thus the system will display a blank <b>Help</b> table with no matching fields. <b>The usecase ends.</b> |   |
|                           | <b>ALT-Step 7:</b> The user requests to view the User Manual. The use case extends to 1.7 “View User Manual”.  |   |
| <b>Conclusion:</b>        | The system displays the help video that matches with the user's request.   |   |
| <b>Post-condition:</b>    | The user receives the help they requested from the help video.   |   |

|  |  |
|--|--|
| <b>Business rules</b>                                | All the users can View all Help to the system. |
| <b>Implementation constraints and specifications</b> | • None   |
| <b>Assumptions:</b>                                  | • None   |
| <b>Open issues:</b>                                  | • None   |

Table 113: 2.30 View All Help

---

### USE CASE 3.8-12 & 4.1-4.6

|                             |                    |              |
|-----------------------------|--------------------|--------------|
| System Name: Procion System |                    |              |
| Sub-System:                 |                    |              |
| Author: Bupe Chindongo      | Date: 25 July 2023 | Version: 1.0 |

| Use case name:                 | Receive procurement item | USE CASE TYPE  |
|--------------------------------|--------------------------|--|
| Use case id:                   | 3.8                      | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High                     | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo                     | System Design: <input type="checkbox"/>              |
| Primary business actor         | Requester                |  |
| Primary system actor           | None                     |  |
| Other participating actors:    | None                     |  |
| Other interested stakeholders: | None                     |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Description:</b>              | This use case describes the event whereby a requester on the system would like to receive procurement items. The system will prompt the requester to update the stock items which have been received, and the requester will update the stock to match with what has been received.  |   |
| <b>Pre-condition:</b>            | The requester has to be logged in to the system  |   |
| <b>Trigger:</b>                  | The requester requests to receive procurement items  |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The requester requests to receive procurement items   | <b>Step 2:</b> The system retrieves the consumable where the Consumable_ID of the Consumable table is equal to the Consumable_ID of the Procurement_Consumable table.                                       |
|                                  |  | <b>Step 3:</b> The system prompts the requester to enter the amount into the number selector  |
|                                  | <b>Step 4:</b> The requester will enter the amount into the field  | <b>Step 5:</b> The system validates that the input has been filled with valid data. <b>[ALT]</b>  |
|                                  | <b>Step 6:</b> The requester clicks on the “Save” button<br><b>[ALT]</b>   | <b>Step 7:</b> The system will save the amount entered into the <b>Consumable</b> entity and update the quantity on hand accordingly. The system will then generate a success notification.<br><b>[ALT]</b> |
| <b>Alternate courses:</b>        | <p><b>[ALT] Step 5:</b> The data entered is invalid. The requester will not be able to click on the save button. Return to Step 3.</p> <p><b>[ALT] Step 6:</b> The requester clicks on the cancel button. Terminate use case.</p> <p><b>[ALT] Step 7:</b> The <b>Consumable</b> entity was unable to be updated. Display an unsuccessful notification.</p> |   |

|  |  |
|--|--|
| <b>Conclusion:</b>                                   | The quantity on hand of the consumable has been updated accordingly and the success notification is displayed. |
| <b>Post-condition:</b>                               | A record in the <b>Consumable</b> entity has been updated.   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

*Table 114: 3.8 Receive Procurement Item*

|  |                    |              |
|--|--------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Bupe Chindongo      Date: 25 July 2023      Version: 1.0 |                    |              |
| Author: Bupe Chindongo   | Date: 25 July 2023 | Version: 1.0 |

|                                    |                    |  |
|------------------------------------|--------------------|--|
| <b>Use case name:</b>              | Upload Tax Invoice | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                | 3.9                | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                   | High               | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                     | Moyo               | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>      | Requester          |  |
| <b>Primary system actor</b>        | None               |  |
| <b>Other participating actors:</b> | None               |  |

|                                |   |   |
|--------------------------------|---|---|
| Other interested stakeholders: | None  |   |
| Description:                   | The use case describes the event whereby a requester would like to upload a tax invoice for a procurement request. The requester will provide the required file to the system, and the system will save the file into its database. The system will display a notification that the file has been successfully added to the database. |   |
| Pre-condition:                 | The requester has to be logged in to the system   |   |
| Trigger:                       | The requester requests to upload a tax invoice for a procurement request  |   |
| Typical course Of events:      | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                | <b>Step 1:</b> The requester requests to upload a tax invoice   | <b>Step 2:</b> The system prompts the requester to select a file to upload  |
|                                | <b>Step 3:</b> The requester selects the file to be uploaded  | <b>Step 4:</b> The system validates the file type. <b>[ALT]</b>   |
|                                | <b>Step 5:</b> The requester clicks on the “Save” button<br><br>[ALT]   | <b>Step 6:</b> The system will save the file into the correct folder and file path in the API, after which it will display a success notification.<br><br>[ALT] |
| Alternate courses:             | <p><b>[ALT] Step 4:</b> The file entered is invalid. The requester will not be able to click on the save button. Return to Step 3.</p> <p><b>[ALT] Step 5:</b> The requester clicks on the cancel button. Terminate use case.</p> <p><b>[ALT] Step 6:</b> The file was unable to be added. Display an unsuccessful notification.</p>  |   |
| Conclusion:                    | The file has been added to the backend of the system and a success notification is displayed.   |   |
| Post-condition:                | A file has been added to the correct directory in the API.  |   |

|  |  |
|--|--|
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

*Table 115: 3.9 Upload Tax Invoice*

|  |                    |              |
|--|--------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Bupe Chindongo      Date: 25 July 2023      Version: 1.0 |                    |              |
| Author: Bupe Chindongo   | Date: 25 July 2023 | Version: 1.0 |

|                                       |                                   |  |
|---------------------------------------|-----------------------------------|--|
| <b>Use case name:</b>                 | Budget owner requisition approval | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 3.10                              | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High                              | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo                              | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | Budget owner                      |  |
| <b>Primary system actor</b>           | None                              |  |
| <b>Other participating actors:</b>    | None                              |  |
| <b>Other interested stakeholders:</b> | None                              |  |

|                                  |   |   |
|----------------------------------|---|---|
| <b>Description:</b>              | This use case describes the event whereby a budget owner would like to approve a requisition request. The budget owner will request to view the requisition request, after which the system will display all the details related to the relevant procurement request. The budget owner will verify that the details of the request are in order, such as the invoice has been uploaded, the total that is due, the budget line that has been selected, after which the budget |   |
| <b>Pre-condition:</b>            | The requester has to be logged in to the system   |   |
| <b>Trigger:</b>                  | The requester requests to receive procurement items   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                  | <p><b>Step 1:</b> The budget owner requests to review a requisition request.</p>  | <p><b>Step 2:</b> The system loads the relevant requisition request which has the status of “unapproved” by retrieving the requisition request where the Procurement_ID of the selected request by the budget owner is equal to the Procurement_Request_ID of the Procurement_Request entity.</p> |
|                                  | <p><b>Step 3:</b> The budget owner reviews the requisition request and clicks the “Approve” button.</p> <p>[ALT]</p>  | <p><b>Step 4:</b> The system updates the <b>Procurement_Request</b> entity and sets the Requisition_Status to approved. A success notification is displayed.</p> <p>[ALT]</p>   |
| <b>Alternate courses:</b>        | <p>[ALT] <b>Step 3:</b> The budget owner clicks on the “cancel” button. Terminate use case.</p> <p>[ALT] <b>Step 4:</b> The requisition status could not be updated. Display an unsuccessful notification. Terminate use case.</p>  |   |
| <b>Conclusion:</b>               | The system updates the Requisition_Status in the Procurement_Request entity and displays a success notification.  |   |
| <b>Post-condition:</b>           | A record in the <b>Procurement_Request</b> entity is updated.   |   |
| <b>Business rules</b>            | None  |   |

|   |  |
|---|--|
| Implementation constraints and specifications | <ul style="list-style-type: none"> <li>None</li> </ul> |
| Assumptions:                                  | None   |
| Open issues:                                  | None   |

Table 116: 3.10 Budget Owner requisition approval

|                             |                    |              |
|-----------------------------|--------------------|--------------|
| System Name: Procion System |                    |              |
| Sub-System:                 |                    |              |
| Author: Bupe Chindongo      | Date: 25 July 2023 | Version: 1.0 |

|                                |   |  |
|--------------------------------|---|--|
| Use case name:                 | Upload receipt  | USE CASE TYPE  |
| Use case id:                   | 3.11  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | Requester   |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | The use case describes the event whereby a requester would like to upload a receipt for a procurement request. The requester will provide the required file to the system, and the system will save the file into its database. The system will display a notification that the file has been successfully added to the database. |  |

|  |  |  |
|--|--|--|
| <b>Pre-condition:</b>                                | The requester has to be logged in to the system  |  |
| <b>Trigger:</b>                                      | The requester requests to upload a receipt for a procurement request   |  |
| <b>Typical course Of events:</b>                     | <b>Actors Action</b>   | <b>Systems response</b>  |
|  | <b>Step 1:</b> The requester requests to upload a receipt  | <b>Step 2:</b> The system prompts the requester to select a file to upload   |
|  | <b>Step 3:</b> The requester selects the file to be uploaded   | <b>Step 4:</b> The system validates the file type. <b>[ALT]</b>  |
|  | <b>Step 5:</b> The requester clicks on the “Save” button<br><br><b>[ALT]</b>   | <b>Step 6:</b> The system will save the file into the correct folder and file path in the API, after which it will display a success notification.<br><br><b>[ALT]</b> |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4:</b> The file entered is invalid. The requester will not be able to click on the save button. Return to Step 3.</p> <p><b>[ALT] Step 5:</b> The requester clicks on the cancel button. Terminate use case.</p> <p><b>[ALT] Step 6:</b> The file was unable to be added. Display an unsuccessful notification.</p> |  |
| <b>Conclusion:</b>                                   | The file has been added to the backend of the system and a success notification is displayed.  |  |
| <b>Post-condition:</b>                               | A file has been added to the correct directory in the API.   |  |
| <b>Business rules</b>                                | None   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | None   |  |

|                     |      |
|---------------------|------|
| <b>Open issues:</b> | None |
|---------------------|------|

Table 117: 3.10 Upload Receipt

|                             |                    |              |
|-----------------------------|--------------------|--------------|
| System Name: Procion System |                    |              |
| Sub-System:                 |                    |              |
| Author: Bupe Chindongo      | Date: 25 July 2023 | Version: 1.0 |

| <b>Use case name:</b>                 | Upload Credit Card Payment  | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|---|--|
| <b>Use case id:</b>                   | 3.12  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | Requester   |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | The use case describes the event whereby a requester would like to upload a credit card payment for a procurement request. The requester will provide the required file to the system, and the system will save the file into its database. The system will display a notification that the file has been successfully added to the database. |  |
| <b>Pre-condition:</b>                 | The requester has to be logged in to the system   |  |
| <b>Trigger:</b>                       | The requester requests to upload a credit card payment for a procurement request  |  |

| Typical course Of events:                            | Actors Action  | Systems response   |
|--|--|--|
|  | <b>Step 1:</b> The requester requests to upload a credit card payment  | <b>Step 2:</b> The system prompts the requester to select a file to upload   |
|  | <b>Step 3:</b> The requester selects the file to be uploaded   | <b>Step 4:</b> The system validates the file type. <b>[ALT]</b>  |
|  | <b>Step 5:</b> The requester clicks on the “Save” button<br><br><b>[ALT]</b>   | <b>Step 6:</b> The system will save the file into the correct folder and file path in the API, after which it will display a success notification.<br><br><b>[ALT]</b> |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4:</b> The file entered is invalid. The requester will not be able to click on the save button. Return to Step 3.</p> <p><b>[ALT] Step 5:</b> The requester clicks on the cancel button. Terminate use case.</p> <p><b>[ALT] Step 6:</b> The file was unable to be added. Display an unsuccessful notification.</p> |  |
| <b>Conclusion:</b>                                   | The file has been added to the backend of the system and a success notification is displayed.  |  |
| <b>Post-condition:</b>                               | A file has been added to the correct directory in the API.   |  |
| <b>Business rules</b>                                | None   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | None   |  |
| <b>Open issues:</b>                                  | None   |  |

Table 118: 3.11 Upload Credit Card Payment

|  |                    |              |
|--|--------------------|--------------|
| System Name: Procion System<br>Sub-System: |                    |              |
| Author: Bupe Chindongo                     | Date: 25 July 2023 | Version: 1.0 |

| Use case name:                 | Finalize Procurement request   | USE CASE TYPE  |
|--------------------------------|--|--|
| Use case id:                   | 4.1  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo   | System Design: <input type="checkbox"/>              |
| Primary business actor         | Finance user   |  |
| Primary system actor           | None   |  |
| Other participating actors:    | None   |  |
| Other interested stakeholders: | None   |  |
| Description:                   | <p>This use case describes the event whereby a finance user would like to finalize a procurement request. The finance user will request to view the procurement request, after which the system will display all the details related to the relevant procurement request. The finance user will verify that the details of the procurement request are in order, after which the finance user will approve the procurement request. After the request has been approved, requester will be notified that their procurement request has been finalized.</p> |  |
| Pre-condition:                 | The finance user has to be logged in to the system   |  |
| Trigger:                       | The finance user requests to finalize a procurement request.   |  |

| Typical course Of events:                            | Actors Action  | Systems response   |
|--|--|--|
|  | <b>Step 1:</b> The finance user requests to finalize a procurement request.  | <b>Step 2:</b> The system retrieves the procurement item where the Procurement_ID of the Procurement table is equal to the Procurement_ID of the procurement request selected by the finance user.                       |
|  |  | <b>Step 3:</b> The system prompts the finance user to finalize the procurement request.  |
|  | <b>Step 4:</b> The finance user clicks on the “Finalize” button. <b>[ALT]</b>  | <b>Step 5:</b> The system updates the Procurement_Payment_Status of the procurement request to indicate that the procurement request has been finalized. The system displays a success notification.<br><br><b>[ALT]</b> |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 4:</b> The finance user clicks on the “Cancel” button. Terminate use case.<br><br><b>[ALT] Step 5:</b> The relevant record could not be updated. Display an unsuccessful notification. Terminate use case. |  |
| <b>Conclusion:</b>                                   | The Procurement_Payment_Status of the procurement request has been updated to indicate that the procurement request has been finalized, and a success notification is displayed.   |  |
| <b>Post-condition:</b>                               | A record in the <b>Procurement_Details</b> entity has been updated.  |  |
| <b>Business rules</b>                                | None   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | None   |  |
| <b>Open issues:</b>                                  | None   |  |

Table 119: 4.1 Finalize Procurement Request

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 4 Finance |                  |              |
| Author: Bupe Chindongo                               | Date: 2023/07/30 | Version: 1.5 |

| Use case name:                 | Create Budget Allocation  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 4.3   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor:        | User  |  |
| Primary system actor:          | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>In this use case the user would like to create a budget allocation. The user will thus request the system to create a budget allocation. The system will then prompt the user to enter all the budget allocation fields required. The user will then enter all the required fields and save the entered fields.</p> <ul style="list-style-type: none"> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul> <p>The system will capture the information and the newly created budget allocation will be added and displayed to the <b>Budget_Allocation</b> table in the database. The system will then route the user to the <b>Budget_Lines</b> screen for the created budget allocation. On this screen the user will fill in all of the required budget line fields and then save these fields to create a new budget line associated with a budget allocation.</p> |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Pre-condition:</b>            | The user must be logged into the system.   |   |
| <b>Trigger:</b>                  | The user requests to create a budget allocation.   |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>  | <b>System Response</b>  |
|                                  | <b>Step 1:</b> user requests to create a new allocation category on the system.              | <b>Step 2:</b> The system invokes the use case 4.8 “Search Budget Allocations”  |
|                                  |  | <b>Step 3:</b> System will prompt the user to enter the required fields with the following details: <ul style="list-style-type: none"> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul>   |
|                                  | <b>Step 4:</b> The user will enter the required fields and save the details.<br><b>[ALT]</b> | <b>Step 5:</b> The system validates the details that were entered to ensure everything is correct.<br><b>[ALT]</b>  |
|                                  |  | <b>Step 6:</b> The system will add the entered budget allocation details in the <b>Budget Allocation</b> table with the following details:<br><br><b>Budget_Allocation</b> <ul style="list-style-type: none"> <li>• Budget_ID</li> <li>• Department_ID</li> <li>• Date_Created</li> <li>• Year</li> <li>• Total</li> </ul> The system will automatically increment the Budget_ID to ensure each budget allocation has a unique ID. The system will then route the user to the <b>Budget Lines</b> screen.<br><b>[ALT]</b> |

|                           |  |  |
|---------------------------|--|--|
|                           | <p><b>Step 7:</b> The user requests to add a new budget line to the system</p> <p>[ALT]</p>      | <p><b>Step 8:</b> System will prompt the user to enter the required fields with the following details:</p> <ul style="list-style-type: none"> <li>• Account Code</li> <li>• Budget Category</li> <li>• Month</li> <li>• ActualAmt</li> <li>• BudgetAmt</li> </ul>  |
|                           | <p><b>Step 9:</b> The user will enter the required fields and save the details.</p> <p>[ALT]</p> | <p><b>Step 10:</b> The system validates the details that were entered to ensure everything is correct.</p> <p>[ALT]</p>  |
|                           |  | <p><b>Step 11:</b> The system will add the entered budget allocation details in the <b>Budget Line</b> table with the following details:</p> <p><b>Budget_Allocation</b></p> <ul style="list-style-type: none"> <li>• AccountCode</li> <li>• Budget_Category_ID</li> <li>• Budget_ID</li> <li>• Month</li> <li>• ActualAmt</li> <li>• BudgetAmt</li> <li>• Variance</li> </ul> <p>The system will then route the user to the <b>Budget Lines</b> screen.</p>       |
| <b>Alternate courses:</b> |  | <p><b>ALT-Step 4:</b> The user can alternatively request to not save the field but rather cancel. <b>System will Terminates usecase.</b></p> <p><b>ALT-Step 5:</b> The validation of the field is invalid this will display an invalid inputs message to the admin. <b>System will return to step 4.</b></p> <p><b>ALT-Step 6:</b> The system validates and find that the to be added budget allocation details is a duplicate in the <b>Budget_Allocation</b></p> |

|  |   |
|--|---|
|  | table. System will display a Duplicate budget allocation message to the user. <b>System will return to step 4.</b>  |
|  | <b>ALT-Step 7:</b> The user can alternatively request to not add a new budget line, but instead return to the budget allocation screen by clicking “Back”. <b>Terminate use case.</b> |
|  | <b>ALT-Step 9:</b> The user can alternatively request to not save the field but rather cancel. <b>System will Terminates usecase.</b>   |
|  | <b>ALT-Step 10:</b> The validation of the field is invalid this will display an invalid inputs message to the admin. <b>System will return to step 9.</b>                             |
| <b>Conclusion:</b>                                   | The newly created budget allocation and budget line will be added to the system and a success notification is displayed.  |
| <b>Post-condition:</b>                               | A new record is created in the <b>Budget_Allocation as well as the Budget_Line</b> table.   |
| <b>Business rules</b>                                | Only the user can create budget allocations on the system.  |
| <b>Implementation constraints and specifications</b> | No duplicate budget allocations allowed.  |
| <b>Assumptions:</b>                                  | None  |
| <b>Open issues:</b>                                  | None  |

Table 120: 4.3 Create Budget Allocation

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 4 Finance |                  |              |
| Author: Bupe Chindongo                               | Date: 2023/05/15 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Search Budget Allocation   | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 4.4  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | User   |  |
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the user would like to view all the budget allocations on the system. The user will request the system to view all the budget allocations on the system. The system will then load all the budget allocations that are currently on the system and display them on the <b>Budget Allocations</b> table with the following details:</p> <ul style="list-style-type: none"> <li>• Budget_ID</li> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> <li>• Buttons for budget lines, edit, and delete</li> </ul> <p>The system will then prompt the user to enter the search field and at the search bar. The user will then enter the search field at the search bar. The system will then filter the list of budget allocations based on the users search criteria. The system will then display the list of the searched criteria.</p> |  |
| <b>Pre-condition:</b>                 | The user must be logged into the system.   |  |
| <b>Trigger:</b>                       | The user requests to view all budget allocations.  |  |
|                                       | <b>Actor Action</b>  | <b>System Response</b>                               |

|   |  |
|---|--|
| <b>Typical course Of events:</b>  | <p><b>Step 1:</b> The user requests to view all the budget allocations on the system.</p> <p><b>Step 2:</b> The system will retrieve all the budget allocations from the <b>Budget_Allocation</b> table with the following details:</p> <ul style="list-style-type: none"> <li>• Budget_ID</li> <li>• Department_ID</li> <li>• Date_Created</li> <li>• Year</li> <li>• Total</li> </ul> <p>The system will display all the retrieved branch details in the <b>Budget Allocation</b> table.</p> |
|   | <p><b>Step 3:</b> The system will prompt the user to enter the search criteria.</p>  |
| <p><b>Step 4:</b> The user will enter their search criteria and search.</p> | <p><b>Step 5:</b> The system then filters through the retrieved budget allocations list to match the budget allocations list with the search criteria of the admin by comparing the year attribute.</p> <p>The system then displays the matching fields.</p> <p><b>[ALT]</b></p>   |
| <b>Alternate courses:</b>   | <p><b>ALT-Step 5:</b> The search criteria does not match with anything in the budget allocations list since there is no such budget allocation data on the system thus the system will display a blank <b>Budget Allocation</b> table with no matching fields. <b>The usecase ends.</b></p>  |
| <b>Conclusion:</b>  | <p>The system displays the budget allocation list that matches with the users search criteria.</p>   |
| <b>Post-condition:</b>  | <p>The matching <b>Budget Allocation</b> table fields are displayed.</p>   |
| <b>Business rules</b>   | <p>Only the user can View budget allocations on the system.</p>  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul> |

Table 121: 4.4 Search Budget Allocation

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 4 Finance |                  |              |
| Author: Bupe Chindongo                               | Date: 2023/07/29 | Version: 1.1 |

| Use case name:                 | Update Budget Allocation  | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 4.5   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor:        | User  |  |
| Primary system actor:          | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | In this use case the user would like to update a budget allocation. The user will thus request the system to update a budget allocation. The system will then prompt the user to select the specific budget allocation fields they would like to update. The user will then enter all the required fields and save the entered fields. <ul style="list-style-type: none"> <li>Department</li> <li>Date</li> </ul> |  |

|   | <ul style="list-style-type: none"> <li>• Year</li> <li>• Total</li> </ul> <p>The system will capture the updated information and the newly updated budget allocation will be added and added to the <b>Budget_Allocation</b> table in the database.</p>  |                     |                        |   |   |  |   |   |   |  |  |
|---|--|---------------------|------------------------|---|---|--|---|---|---|--|--|
| <b>Pre-condition:</b>   | The user must be logged into the system.   |                     |                        |   |   |  |   |   |   |  |  |
| <b>Trigger:</b>   | The user requests to update a budget allocation.   |                     |                        |   |   |  |   |   |   |  |  |
| <b>Typical course Of events:</b>  | <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;"><b>Actor Action</b></th><th style="text-align: left;"><b>System Response</b></th></tr> </thead> <tbody> <tr> <td><b>Step 1:</b> The user requests to update a budget allocation on the system.</td><td><b>Step 2:</b> The system invokes the use case 4.8 “Search Budget Allocation”</td></tr> <tr> <td></td><td> <b>Step 3:</b> System will prompt the user to enter the required fields with the following details:           <ul style="list-style-type: none"> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul> </td></tr> <tr> <td> <b>Step 4:</b> The user will enter the required fields and saves the details.<br/> <b>[ALT]</b> </td><td> <b>Step 5:</b> The system validates the details that was entered to ensure everything is correct.<br/> <b>[ALT]</b> </td></tr> <tr> <td></td><td> <b>Step 6:</b> The system will update the entered budget allocation details in the <b>Budget Allocation</b> table with the following details:<br/> <br/> <b>Budget_Allocation</b> <ul style="list-style-type: none"> <li>• Department</li> </ul> </td></tr> </tbody> </table> | <b>Actor Action</b> | <b>System Response</b> | <b>Step 1:</b> The user requests to update a budget allocation on the system. | <b>Step 2:</b> The system invokes the use case 4.8 “Search Budget Allocation” |  | <b>Step 3:</b> System will prompt the user to enter the required fields with the following details: <ul style="list-style-type: none"> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul> | <b>Step 4:</b> The user will enter the required fields and saves the details.<br><b>[ALT]</b> | <b>Step 5:</b> The system validates the details that was entered to ensure everything is correct.<br><b>[ALT]</b> |  | <b>Step 6:</b> The system will update the entered budget allocation details in the <b>Budget Allocation</b> table with the following details:<br><br><b>Budget_Allocation</b> <ul style="list-style-type: none"> <li>• Department</li> </ul> |
| <b>Actor Action</b>   | <b>System Response</b>   |                     |                        |   |   |  |   |   |   |  |  |
| <b>Step 1:</b> The user requests to update a budget allocation on the system.                 | <b>Step 2:</b> The system invokes the use case 4.8 “Search Budget Allocation”  |                     |                        |   |   |  |   |   |   |  |  |
|   | <b>Step 3:</b> System will prompt the user to enter the required fields with the following details: <ul style="list-style-type: none"> <li>• Department</li> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul>  |                     |                        |   |   |  |   |   |   |  |  |
| <b>Step 4:</b> The user will enter the required fields and saves the details.<br><b>[ALT]</b> | <b>Step 5:</b> The system validates the details that was entered to ensure everything is correct.<br><b>[ALT]</b>  |                     |                        |   |   |  |   |   |   |  |  |
|   | <b>Step 6:</b> The system will update the entered budget allocation details in the <b>Budget Allocation</b> table with the following details:<br><br><b>Budget_Allocation</b> <ul style="list-style-type: none"> <li>• Department</li> </ul>   |                     |                        |   |   |  |   |   |   |  |  |

|  |  |  |
|--|--|--|
|  |  | <ul style="list-style-type: none"> <li>• Date</li> <li>• Year</li> <li>• Total</li> </ul> <p>[ALT]</p>                                       |
|  |  | <p><b>Step 7:</b> The system will then display a successful notification to the user.</p>  |
| <b>Alternate courses:</b>  |  | <p><b>ALT-Step 4:</b> The user can alternatively request to not save the field but rather cancel. <b>System will Terminates usecase.</b></p> |
| <b>ALT-Step 5:</b> The validation of the field is invalid this will display an invalid inputs message to the user. <b>System will return to step 4.</b>  |  |  |
| <b>ALT-Step 6:</b> The system fails to update the budget allocation details on the <b>Budget Allocation</b> table. <b>System will terminate usecase.</b> |  |  |
| <b>Conclusion:</b>   | The updated budget category details will be updated on the system and a success notification is displayed. |  |
| <b>Post-condition:</b>   | A record is updated in the <b>Budget_Allocation</b> table.   |  |
| <b>Business rules</b>  | Only the user can update budget allocation to the system.  |  |
| <b>Implementation constraints and specifications</b>   | No duplicate budget allocation allowed.  |  |
| <b>Assumptions:</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Open issues:</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |

Table 122: 4.5 Update Budget Allocation

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 4 Finance |                  |              |
| Author: Bupe Chindongo                               | Date: 2023/05/15 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Delete Budget Allocation   |  |
| <b>Use case id:</b>                   | 4.6  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | User   |  |
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the user would like to delete a budget allocation. The user will thus request the system to delete a budget allocation. The system will then prompt the user to the specific budget allocation they would like to delete. The user then selects the budget allocation they would like to delete. The system then prompts the user to confirm if they really do want to delete that specific budget allocation. The user confirms the request, and the system deletes that budget allocation from the <b>Budget_Allocation</b> table on the system.</p> |  |
| <b>Pre-condition:</b>                 | The user must be logged into the system.   |  |
| <b>Trigger:</b>                       | The user requests to delete a budget allocation.   |  |
|                                       | <b>Actor Action</b>  | <b>System Response</b>                               |

|                                  |   |  |
|----------------------------------|---|--|
| <b>Typical course Of events:</b> | <b>Step 1:</b> The user requests to delete a budget allocation on the system.     | <b>Step 2:</b> The system invokes the use case 4.9 “Search Budget Allocation”  |
|                                  |   | <b>Step 3:</b> The system prompts the user to select the budget allocation they would like to delete.                |
|                                  | <b>Step 4:</b> The user selects the budget allocation they would like to delete.  | <b>Step 5:</b> The system prompts the user to confirm the deletion of that specific budget allocation.               |
|                                  | <b>[ALT]</b>  |  |
|                                  | <b>Step 6:</b> The user confirms the deletion of that specific budget allocation. |  |
|                                  | <b>[ALT]</b>  | <b>Step 7:</b> The system then removes that budget allocation from the system in the <b>Budget_Allocation</b> table. |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |
|                                  |   |  |

|  |  |
|--|--|
| <b>Conclusion:</b>                                   | The budget allocation has been deleted and successful deletion notification displays.            |
| <b>Post-condition:</b>                               | The requested deleted budget allocation has been removed from the <b>Budget_Allocation</b> table |
| <b>Business rules</b>                                | Only the user can delete budget allocation on the system   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

*Table 123: 4.6 Delete Budget Allocation*


---

#### USE CASE 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System: 5 Inventory |                   |              |
| Author: Jason van der Merwe                            | Date: 13 May 2023 | Version: 1.0 |

|                               |                   |  |
|-------------------------------|-------------------|--|
| <b>Use case name:</b>         | Create Consumable | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>           | 5.1               | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>              | High              | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                | Moyo              | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b> | User              |  |
| <b>Primary system actor</b>   | None              |  |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to create a new consumable item. The use case begins with the admin requesting to create a new consumable on the system. The user will input all of the relevant details that they wish for the consumable, the system will validate these details. The system will save the new consumable on the system's database in the <b>Consumable</b> entity and notifies the user thus the use case concludes.</p> |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>The user must have an internet connection</p>  |  |
| <b>Trigger:</b>                       | The user requests to add a consumable item to the system   |  |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>   | <b>Systems response</b>  |
|                                       | <b>Step 1:</b> The user requests to add a consumable item to the system.   | <b>Step 2:</b> The system invokes UC 5.2: View Consumables   |
|                                       |  | <p><b>Step 3:</b> The system will prompt the user to enter the following details for the consumable which includes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Category</li> <li>• Minimum Reorder Quantity</li> <li>• Maximum Reorder Quantity</li> <li>• On Hand</li> </ul> |
|                                       | <p><b>Step 4:</b> The user will provide the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On-Hand amount</li> <li>• Minimum Reorder Quantity</li> <li>• Maximum- Reorder quantity</li> <li>• Category</li> </ul> <p>The user submits</p>  | <p><b>Step 5:</b> The system validates that all fields have been entered and is in the correct format</p> <p>[ALT]</p>   |

|                    |  |   |
|--------------------|--|---|
|                    | [ALT]  |   |
|                    |  | <p><b>Step 6:</b> The system then inserts the following attributes into the <b><u>Consumable</u></b> entity:</p> <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Name</li> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_quantity</li> <li>• Consumable_Category_ID</li> </ul> <p>The Consumable_Category_ID is a foreign key in the <b><u>Consumable</u></b> entity</p> <p>The <b>Consumable_ID</b> is auto incremented in the database by taking the previous <b>Consumable_ID</b> and adds 1 to it</p> |
|                    |  | <p><b>Step 7:</b> The system displays a success notification to inform the user that the consumable has been created successfully</p>   |
| Alternate courses: | [ALT] <b>Step 4:</b> The user decides not to add a consumable. The use case ends   |   |
|                    | [ALT] <b>Step 5a:</b> The validation of the input fields have not succeeded. The user is appropriately notified. Return to Step 4                                      |   |
|                    | [ALT] <b>Step 5b:</b> The system has validated the consumable and has found duplicates. The user is notified with the appropriate error notification. Return to step 4 |   |
| Conclusion:        | The consumable has been added on the system and a notification is displayed to the user  |   |
| Post-condition:    | A new consumable is added to the <b><u>Consumable</u></b> entity   |   |
| Business rules     | A consumable cannot be duplicated in the same category   |   |

|  |   |
|--|---|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>The user requires internet access</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                            |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                            |

Table 124: 5.1 Create Consumable

|  |                   |              |
|--|-------------------|--------------|
| <p style="text-align: center;">System Name: Procion System<br/>Sub-System: 5 Inventory</p> |                   |              |
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0 |

| Use case name:                 | View Consumables   | USE CASE TYPE  |
|--------------------------------|--|--|
| Use case id:                   | 5.2  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo   | System Design: <input type="checkbox"/>              |
| Primary business actor         | User   |  |
| Primary system actor           | None   |  |
| Other participating actors:    | None   |  |
| Other interested stakeholders: | None   |  |
| Description:                   | The use case describes the event where the user will want to view all of the Consumables saved on the system. The use case begins with the user requesting to view all of the consumables saved on the system. The system will load all of the Consumables. The user will enter details to search for and the system will filter the consumable records to match the search criteria. The use case concludes when the filtered search results is displayed to the user |  |

|                                      |  |   |
|--------------------------------------|--|---|
| <b>Pre-condition:</b>                | The user has to be logged in to the system.  |   |
| <b>Trigger:</b>                      | The user requests to view the consumables on the system                              |   |
| <b>Typical course<br/>Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                      | <b>Step 1:</b> The user requests to view all of the Consumables saved on the system. | <b>Step 2:</b> The system retrieves all of the consumables from the <b>Consumable</b> entity as well as the category selected from the <b>Consumable Category</b> entity by means of the foreign key <b>Consumable_Category_ID</b> in the Consumable entity which links to the <b>Consumable Category</b> entity by means of its primary key: <b>Consumable_Category_ID</b> |
|                                      |  | <b>Step 3:</b> The system displays the consumables to the user  |
|                                      |  | <b>Step 4:</b> The system prompts the user to provide their search criteria   |
|                                      | <b>Step 5:</b> The user provides their search criteria                               | <b>Step 6:</b> The system filters dynamically by comparing the search criteria of the user against the <b>Name</b> attribute contained in the <b>Consumable</b> entity.<br><br>The system displays the search results to the user   |
| <b>Alternate courses:</b>            | None   |   |
| <b>Conclusion:</b>                   | The system displays the filtered consumables to the user                             |   |
| <b>Post-condition:</b>               | The system displays the consumables to the user.                                     |   |

|  |  |
|--|--|
| <b>Business rules</b>                                | A consumable cannot be duplicated in the same category   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

Table 125: 5.2 View Consumable

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe   Date: 13 May 2023   Version: 1.0 |                   |              |
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Update Consumable  | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 5.3  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | The use case describes the event where the user will want to update an existing consumable item. The use case begins with a user requesting to update a existing consumable on the system. The system will prompt the user to provide the details that they wish to update. The user will enter the relevant details which will be captured on the system. |  |

|                                  |   |   |
|----------------------------------|---|---|
|                                  | The use case concludes when the consumable is updated in the <b><u>Consumable</u></b> entity on the system.   |   |
| <b>Pre-condition:</b>            | <ul style="list-style-type: none"> <li>• The user has to be logged in to the system.</li> <li>• There must be at least one consumable stored on the system</li> </ul>   |   |
| <b>Trigger:</b>                  | The user requests to update a consumable item.  |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The user requests to update a consumable item.   | <b>Step 2:</b> The system invokes <b>UC 5.2: View Consumables</b>   |
|                                  |   | <b>Step 3:</b> The system will prompt the user to enter the following details for the consumable which includes: <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• Category</li> <li>• Minimum Reorder Quantity</li> <li>• Maximum Reorder Quantity</li> <li>• On Hand</li> </ul> |
|                                  | <b>Step 4:</b> The user will provide the following details: <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On-Hand amount</li> <li>• Minimum Reorder Quantity</li> <li>• Maximum- Reorder quantity</li> <li>• Category</li> </ul> The user submits<br><b>[ALT]</b> | <b>Step 5:</b> The system validates that all fields have been entered and is in the correct format<br><b>[ALT]</b>  |
|                                  |   | <b>Step 6:</b> The system then updates the following attributes into the <b><u>Consumable</u></b> entity: <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Name</li> </ul>   |

|  |  |   |
|--|--|---|
|  |  | <ul style="list-style-type: none"> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity</li> <li>• Consumable_Category_ID</li> </ul> <p>The Consumable_Category_ID is a foreign key in the <b>Consumable</b> entity</p> |
|  |  | <p><b>Step 7:</b> The system displays a success notification to inform the user that the consumable has been updated successfully</p>   |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4:</b> The user decides not to update a consumable. The use case ends</p>   |   |
|  | <p><b>[ALT] Step 5a:</b> The validation of the input fields have not succeeded. The user is appropriately notified. Return to Step 4</p> <p><b>[ALT] Step 5b:</b> The system has validated the consumable and has found duplicates. The user is notified with the appropriate error notification. Return to step 4</p> |   |
| <b>Conclusion:</b>                                   | <p>The consumable has been updated on the system and a notification is displayed to the user</p>   |   |
| <b>Post-condition:</b>                               | <p>A existing consumable is updated on the system</p>  |   |
| <b>Business rules</b>                                | <p>A consumable cannot be duplicated in the same category</p>  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |   |
| <b>Assumptions:</b>                                  | <p>None</p>  |   |
| <b>Open issues:</b>                                  | <p>None</p>  |   |

Table 126: 5.3 Update Consumable

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe    Date: 13 May 2023    Version: 1.0 |                   |              |
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Delete Consumable   | <b>USE CASE TYPE</b>                                       |
| <b>Use case id:</b>                   | 5.4   | Business Requirements: <input type="checkbox"/>            |
| <b>Priority:</b>                      | High  | System Analysis: <input checked="" type="checkbox"/>       |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>                    |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | <p>The user requests to delete the consumable. The use case begins with the user requesting to delete a consumable on the system. The system will prompt the select the consumable that they would like to delete. The user will select the consumable and they will be prompted to confirm deletion. The user will confirm the deletion of the consumable. The consumable will be removed from the <b>Consumable</b> entity. The user is displayed with a success notification notifying them that the consumable has been successfully deleted.</p> |  |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.   |  |
| <b>Trigger:</b>                       | The user requests to delete a consumable.   |  |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>  | <b>Systems response</b>                                    |
|                                       | <b>Step 1:</b> The user requests to delete a consumable item.   | <b>Step 2:</b> The system invokes UC 5.2: View Consumables |

|  |  |   |
|--|--|---|
|  |  | <b>Step 3:</b> The system will prompt the user to select a consumable that they wish to delete.   |
|  | <b>Step 4:</b> The user selects a consumable to be deleted<br><b>[ALT]</b>                   | <b>Step 5:</b> The system prompts the user to provide confirmation that they would like to delete a consumable  |
|  | <b>Step 6:</b> The user confirms that they would like to delete a consumable<br><b>[ALT]</b> | <b>Step 7:</b> The system deletes/ removes the selected consumable from the system's database in the <b>Consumable</b> entity which will remove the following details from the selected consumable: <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Name</li> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_quantity</li> <li>• Consumable_Category_ID</li> </ul> |
|  |  | <b>Step 8:</b> The system notifies the user that the consumable has been deleted successfully   |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 4:</b> The user chooses not to delete the consumable. The use case ends        |   |
|  | <b>[ALT] Step 6:</b> The user denies the deletion confirmation. The use case ends            |   |
| <b>Conclusion:</b>                                   | The chosen consumable has been deleted/ removed from the system's database                   |   |
| <b>Post-condition:</b>                               | The chosen consumable is removed from the <b>Consumable</b> entity                           |   |
| <b>Business rules</b>                                | None   |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>        |   |

|                     |  |
|---------------------|--|
| <b>Assumptions:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Table 127: 5.4 Delete Consumable

|   |                   |              |
|---|-------------------|--------------|
| <p style="text-align: center;">System Name: Procion System</p> <p style="text-align: center;">Sub-System:</p> |                   |              |
| Author: Jason van der Merwe   | Date: 13 May 2023 | Version: 1.0 |

| Use case name:                 | Create Consumable Category   | USE CASE TYPE  |
|--------------------------------|--|--|
| Use case id:                   | 5.5  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo   | System Design: <input type="checkbox"/>              |
| Primary business actor         | User   |  |
| Primary system actor           | None   |  |
| Other participating actors:    | None   |  |
| Other interested stakeholders: | None   |  |
| Description:                   | <p>The use case describes the event where the user will want to create a new consumable category. The use case begins with the admin requesting to create a new consumable category on the system. The user will input all of the relevant details that they wish for the consumable category, the system will validate these details. The system will save the new consumable category on the system's database in the <b>Consumable Category</b> entity and notifies the user thus the use case concludes.</p> |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Pre-condition:</b>            | The user has to be logged in to the system.  |   |
| <b>Trigger:</b>                  | The user requests to add a consumable category to the system   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The user requests to add a consumable category to the system.   | <b>Step 2:</b> The system invokes UC 5.6: View Consumable Categories  |
|                                  |  | <b>Step 3:</b> The system will prompt the user to enter the following details for the consumable category which includes: <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul>   |
|                                  | <b>Step 4:</b> The user will provide the following details: <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> The user submits<br><b>[ALT]</b> | <b>Step 5:</b> The system validates that all fields have been entered and is in the correct format<br><b>[ALT]</b>  |
|                                  |  | <b>Step 6:</b> The system then inserts the following attributes into the <b><u>Consumable Category</u></b> entity: <ul style="list-style-type: none"> <li>• Consumable_Category_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>The <b>Consumable_Category_ID</b> is auto incremented in the database by taking the previous <b>Consumable_Category_ID</b> and adds 1 to it</p> |

|  |  |   |
|--|--|---|
|  |  |   |
|  |  | <b>Step 7:</b> The system displays a success notification to inform the user that the consumable category has been created successfully |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4:</b> The user decides not to add a consumable category. The use case ends</p> <p><b>[ALT] Step 5a:</b> The validation of the input fields have not succeeded. The user is appropriately notified. Return to Step 4</p> <p><b>[ALT] Step 5b:</b> The system has validated the consumable category and has found duplicates. The user is notified with the appropriate error notification. Return to step 4</p> |   |
| <b>Conclusion:</b>                                   | The consumable category has been added on the system and a notification is displayed to the user   |   |
| <b>Post-condition:</b>                               | A new consumable category is added to the <b><u>Consumable Category</u></b> entity   |   |
| <b>Business rules</b>                                | A consumable Category cannot be duplicated   |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |   |

Table 128: 5.5 Create Consumable Category

System Name: Procion System

Sub-System:

Author: Jason van der Merwe

Date: 13 May 2023

Version: 1.0

| Use case name:                 | View Consumable Categories  | USE CASE TYPE   |
|--------------------------------|---|---|
| Use case id:                   | 5.6   | Business Requirements: <input type="checkbox"/>   |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/>  |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>   |
| Primary business actor         | User  |   |
| Primary system actor           | None  |   |
| Other participating actors:    | None  |   |
| Other interested stakeholders: | None  |   |
| Description:                   | The use case describes the event where the user will want to view all of the Consumable categories saved on the system. The use case begins with the user requesting to view all of the consumable categories saved on the system. The system will load all of the Consumable categories. The user will enter details to search for and the system will filter the consumable category records to match the search criteria. The use case concludes when the filtered search results is displayed to the user |   |
| Pre-condition:                 | The user has to be logged in to the system.   |   |
| Trigger:                       | The user requests to view the consumables on the system   |   |
| Typical course Of events:      | Actors Action   | Systems response  |
|                                | Step 1: The user requests to view all of the Consumable categories saved on the system.   | Step 2: The system retrieves all of the consumable categories from the <u>Consumable Category</u> |

|  |  |  |
|--|--|--|
|  |  | <b>Step 3:</b> The system displays the consumable categories to the user   |
|  |  | <b>Step 4:</b> The system prompts the user to provide their search criteria  |
|  | <b>Step 5:</b> The user provides their search criteria             | <b>Step 6:</b> The system filters dynamically by comparing the search criteria of the user against the <b>Name</b> attribute contained in the <b>Consumable Category</b> entity.<br><br>The system displays the search results to the user |
| <b>Alternate courses:</b>                            | None   |  |
| <b>Conclusion:</b>                                   | The system displays the filtered consumable categories to the user |  |
| <b>Post-condition:</b>                               | The system displays the consumable categories to the user.         |  |
| <b>Business rules</b>                                | A consumable category cannot be duplicated in the same category    |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>           |  |
| <b>Assumptions:</b>                                  | None   |  |
| <b>Open issues:</b>                                  | None   |  |

Table 129: 5.6 View Consumable Categories

System Name: Procion System

Sub-System:

Author: Jason van der Merwe

Date: 13 May 2023

Version: 1.0

|                                       |  |   |
|---------------------------------------|--|---|
| <b>Use case name:</b>                 | <b>Update Consumable Category</b>  |   |
| <b>Use case id:</b>                   | 5.7  | Business Requirements: <input type="checkbox"/>                   |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/>              |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>                           |
| <b>Primary business actor</b>         | User   |   |
| <b>Primary system actor</b>           | None   |   |
| <b>Other participating actors:</b>    | None   |   |
| <b>Other interested stakeholders:</b> | None   |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to update an existing consumable item. The use case begins with a user requesting to update a existing consumable category on the system. The system will prompt the user to provide the details that they wish to update. The user will enter the relevant details which will be captured on the system.</p> <p>The use case concludes when the consumable category is updated in the <b>Consumable Category</b> entity on the system.</p> |   |
| <b>Pre-condition:</b>                 | <ul style="list-style-type: none"> <li>• The user has to be logged in to the system.</li> <li>• There must be at least one consumable category stored on the system</li> </ul>   |   |
| <b>Trigger:</b>                       | The user requests to update a consumable item.   |   |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                       | <b>Step 1:</b> The user requests to update a consumable category item.   | <b>Step 2:</b> The system invokes UC 5.2: <b>View Consumables</b> |

|                           |   |  |
|---------------------------|---|--|
|                           |   | <p><b>Step 3:</b> The system will prompt the user to enter the following details for the consumable which includes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul>                            |
|                           | <p><b>Step 4:</b> The user will provide the following details:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The user submits</p> <p>[ALT]</p>   | <p><b>Step 5:</b> The system validates that all fields have been entered and is in the correct format</p> <p>[ALT]</p>   |
|                           |   | <p><b>Step 6:</b> The system then updates the following attributes into the <b><u>Consumable Category</u></b> entity:</p> <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Name</li> <li>• Description</li> </ul> |
|                           |   | <p><b>Step 7:</b> The system displays a success notification to inform the user that the consumable category has been updated successfully</p>   |
| <b>Alternate courses:</b> | <p>[ALT] <b>Step 4:</b> The user decides not to update a consumable category. The use case ends</p>   |  |
|                           | <p><b>[ALT] Step 5a:</b> The validation of the input fields have not succeeded. The user is appropriately notified. Return to Step 4</p> <p><b>[ALT] Step 5b:</b> The system has validated the consumable category and has found duplicates. The user is notified with the appropriate error notification. Return to step 4</p> |  |
| <b>Conclusion:</b>        | <p>The consumable category has been updated on the system and a notification is displayed to the user</p>   |  |
| <b>Post-condition:</b>    | <p>A existing consumable category is updated on the system</p>  |  |

|  |  |
|--|--|
| <b>Business rules</b>                                | A consumable category cannot be duplicated on the system |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

Table 130: 5.7 Update Consumable Category

|  |  |  |
|--|--|--|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe   Date: 13 May 2023   Version: 1.0 |  |  |
|  |  |  |

|                                       | <b>DELETE CONSUMABLE CATEGORY</b>  | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|--|--|
| <b>Use case id:</b>                   | 5.8  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | The user requests to delete the consumable category. The use case begins with the user requesting to delete a consumable category on the system. The system will prompt the user select the consumable category that they would like to delete. The user will select the consumable category and the system will validate the deletion. The user will be prompted to confirm deletion. The user will confirm the |  |

|                                  |  |   |
|----------------------------------|--|---|
|                                  | deletion of the consumable category. The consumable category will be removed from the <b><u>Consumable Category</u></b> entity. The user is displayed with a success notification notifying them that the consumable category has been successfully deleted. |   |
| <b>Pre-condition:</b>            | The user has to be logged in to the system.  |   |
| <b>Trigger:</b>                  | The user requests to delete a consumable category.   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The user requests to delete a consumable category.  | <b>Step 2:</b> The system invokes UC 5.6: View Consumable Categories  |
|                                  |  | <b>Step 3:</b> The system will prompt the user to select a consumable category that they wish to delete.  |
|                                  | <b>Step 4:</b> The user selects a consumable category to be deleted [ALT]  | <b>Step 5:</b> The system validates the association of the consumable category with regards to the consumable. The system prompts the user to provide confirmation that they would like to delete a consumable category [ALT]   |
|                                  | <b>Step 6:</b> The user confirms that they would like to delete a consumable category [ALT]  | <b>Step 6:</b> The system deletes/ removes the selected consumable category from the system's database in the <b><u>Consumable Category</u></b> entity which will remove the following details from the selected consumable category: <ul style="list-style-type: none"> <li>• Consumable_Category_ID</li> <li>• Name</li> <li>• Description</li> </ul> |
|                                  |  | <b>Step 7:</b> The system notifies the user that the consumable category has been deleted successfully  |
| <b>Alternate courses:</b>        | [ALT] <b>Step 4:</b> The user chooses not to delete the consumable category. The use case ends   |   |

|  |  |
|--|--|
|  | <b>[ALT] Step 5:</b> The validation confirms the existence of at least one association between the selected consumable category to be deleted and a consumable. The user is appropriately notified and the use case ends |
|  | <b>[ALT] Step 6:</b> The user denies the deletion confirmation. The use case ends  |
| <b>Conclusion:</b>                                   | The chosen consumable category has been deleted/ removed from the system's database  |
| <b>Post-condition:</b>                               | The chosen consumable category is removed from the <b>Consumable Category</b> entity   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 131: 5.8 Delete Consumable Category

|   |  |  |
|---|--|--|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 25 July 2023      Version: 1.0 |  |  |
|   |  |  |

|                               |                          |  |
|-------------------------------|--------------------------|--|
| <b>Use case name:</b>         | Export Inventory Details | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>           | 5.9                      | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>              | High                     | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                | Moyo                     | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b> | User                     |  |
| <b>Primary system actor</b>   | None                     |  |

|                                       |  |   |
|---------------------------------------|--|---|
| <b>Other participating actors:</b>    | None   |   |
| <b>Other interested stakeholders:</b> | None   |   |
| <b>Description:</b>                   | <p>The use case describes the event where a user would like to export the inventory details of all consumables of the system</p> <p>The user Requests to export the inventory details. The system retrieves all of the data stored for the consumables on the system along with their respective categories. The system generates the Inventory details report and displays it to the user</p> |   |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.  |   |
| <b>Trigger:</b>                       | The user requests to export the inventory details  |   |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                       | <b>Step 1:</b> The user requests to export the inventory details   | <b>Step 2:</b> The system retrieves all of the records contained in the <b>Consumables Entity</b> and the <b>Consumable_Category entity</b> |
|                                       | <b>Step 3:</b> The user selects the inventory item to be updated   | <b>Step 4:</b> The system generates inventory details report based on the data retrieved from the system and displays it to the user        |
| <b>Alternate courses:</b>             | <b>None</b>  |   |
| <b>Conclusion:</b>                    | The user is displayed the Inventory details report   |   |
| <b>Post-condition:</b>                | The user is displayed the Inventory details report   |   |
| <b>Business rules</b>                 | None   |   |
| <b>Implementation constraints and</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>  |   |

|                       |  |
|-----------------------|--|
| <b>specifications</b> |  |
| <b>Assumptions:</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Table 132: 5.9 Export Inventory Details

|   |  |  |
|---|--|--|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 25 July 2023      Version: 1.0 |  |  |
|---|--|--|

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Complete Stock Take  | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 5.10   | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | The use case describes the event where a user requests to update the stock levels for a inventory item. The system prompts the user to provide the inventory item for which they will like to update the stock amount. The user selects the inventory item and the system prompts the user to provide the new stock amount that they will like to add. The user provides the details and the Inventory item on hand amount and |  |

|                                      |   |   |
|--------------------------------------|---|---|
|                                      | the consumable history entity is updated with the latest stock amount.<br>The system displays a successful update notification to the user                  |   |
| <b>Pre-condition:</b>                | The user has to be logged in to the system.   |   |
| <b>Trigger:</b>                      | The user requests to update the inventory stock amount of a specific inventory item   |   |
| <b>Typical course<br/>Of events:</b> | <b>Actors Action</b>  | <b>Systems response</b>   |
|                                      | <b>Step 1:</b> The user requests to update the inventory stock amount of a specific inventory item  | <b>Step 2:</b> The system prompts the user to select the inventory item for which they will like to update the amount   |
|                                      | <b>Step 3:</b> The user selects the inventory item to be updated  | <b>Step 4:</b> The system prompts the user to provide the following details <ul style="list-style-type: none"><li>• Stock amount</li></ul>  |
|                                      | <b>Step 5:</b> The user provides the following details: <ul style="list-style-type: none"><li>• Stock amount</li></ul> The user submits<br><br><b>[ALT]</b> | <b>Step 6:</b> The system captures the stock amount and updates the StockAmt attribute in the <b>Consumable_History entity</b> along with the On_Hand attribute in the <b>Consumable</b> entity for the specific consumable where the Consumable_ID in the <b>Consumable</b> entity is equal to the Consumable_ID in the <b>Consumable_History Entity</b> |
|                                      |   | <b>Step 7:</b> The system displays a successful update notification for the user to indicate that the Consumable On hand amount and the Consumable History stock amount has been updated successfully   |
| <b>Alternate courses:</b>            | <b>[ALT] Step 5:</b> The user does not wish to provide the details, The use case ends   |   |

|  |   |
|--|---|
| <b>Conclusion:</b>                                   | The user is notified of the successful stock on hand capture                          |
| <b>Post-condition:</b>                               | The Consumable history and on hand amount is updated on the system                    |
| <b>Business rules</b>                                | None  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                              |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                              |

Table 133: 5.10 Complete Stock Take

|  |       |               |
|--|-------|---------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 7 July 2023      Version: 1.0 |       |               |
| Use case name:   | Login | USE CASE TYPE |

|                                    |      |  |
|------------------------------------|------|--|
| <b>Use case id:</b>                | 1.1  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                   | High | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                     | Moyo | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>      | User |  |
| <b>Primary system actor</b>        | None |  |
| <b>Other participating actors:</b> | None |  |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The use case describes the event a user wants to log in to the system. The user will be prompted to enter their username/ email and password on the login screen. The system will check the systems database for any temporary access levels for the user. The user's credentials will be validated in the system's database. The use case concludes once the system has generated a token for the user and the user is logged in to the system</p> |  |
| <b>Pre-condition:</b>                 | The user has loaded the “Login” screen   |  |
| <b>Trigger:</b>                       | The user Requests to Login to the system.  |  |
| <b>Typical course Of events:</b>      | <b>Actors Action</b><br><b>Step 1:</b> The user Requests to Login to the system.<br><br><b>Step 3:</b> The user provides the following details<br><ul style="list-style-type: none"> <li>• Username</li> <li>• Password</li> </ul> The user submits <b>[ALT]</b>   | <b>Systems response</b><br><b>Step 2:</b> The system prompts the user to enter the following details:<br><ul style="list-style-type: none"> <li>• Username or email</li> <li>• Password</li> </ul> <b>Step 4:</b> The system validates that the details entered is in the correct format<br><b>[ALT]</b>   |
|                                       |  | <b>Step 5:</b> The system then retrieve the user from the <b>User Entity</b> and the Temporary access details from the <b>Delegation_Of_Authority entity</b> where the User_ID in the <b>Delegation_Of_Authority Entity</b> is equal to the User_ID in the <b>User table</b><br><br>The system will validate the User;s email/username and password against the details in the User entity And return the temporary access details of the user |

|  |  |
|--|--|
|  | <b>Step 6:</b> A token is generated for the user the user is logged in to the system.  |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 3:</b> The user has forgot their password. <b>Invoke UC 1.3 Forgot password</b></p> <p><b>[ALT] Step 4:</b> The validation of the input fields have not succeeded. The user is appropriately notified. Return to Step 3</p> |
| <b>Conclusion:</b>                                   | The system generates a token and the user is logged in to the system   |
| <b>Post-condition:</b>                               | The user is logged in to the system  |
| <b>Business rules</b>                                |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 134: 1.1 Login

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 7 July 2023      Version: 1.0 |                   |              |
| Author: Jason van der Merwe  | Date: 7 July 2023 | Version: 1.0 |

|                       |        |  |
|-----------------------|--------|--|
| <b>Use case name:</b> | Logout | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>   | 1.2    | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>        | Moyo   | System Design: <input type="checkbox"/>              |

|  |  |  |
|--|--|--|
| <b>Primary business actor</b>                        | User   |  |
| <b>Primary system actor</b>                          | None   |  |
| <b>Other participating actors:</b>                   | None   |  |
| <b>Other interested stakeholders:</b>                | None   |  |
| <b>Description:</b>                                  | The use case describes the event where a user will want to log out of the system. The system captures the Logout request of the user. The user's token is destroyed and the user is redirected to the “Login” Screen |  |
| <b>Pre-condition:</b>                                | The user has to be logged in to the system.  |  |
| <b>Trigger:</b>                                      | The user requests to log out of the system   |  |
| <b>Typical course Of events:</b>                     | <b>Actors Action</b>   | <b>Systems response</b>  |
|  | <b>Step 1:</b> The user requests to log out of the system.   | <b>Step 2:</b> The system destroys the token and logs the user out of the system |
| <b>Alternate courses:</b>                            | None   |  |
| <b>Conclusion:</b>                                   | The token is destroyed and the user is logged out of the system  |  |
| <b>Post-condition:</b>                               | The user is logged out of the system   |  |
| <b>Business rules</b>                                | None   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | None   |  |
| <b>Open issues:</b>                                  | None   |  |

Table 135: 1.2 Logout

|  |                    |              |
|--|--------------------|--------------|
| System Name: Procion System<br>Sub-System: |                    |              |
| Author: Jason van der Merwe                | Date: 25 July 2023 | Version: 1.0 |

| Use case name:                 | Forgot Password  | USE CASE TYPE  |
|--------------------------------|--|--|
| Use case id:                   | 1.3  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo   | System Design: <input type="checkbox"/>              |
| Primary business actor         | User   |  |
| Primary system actor           | None   |  |
| Other participating actors:    | None   |  |
| Other interested stakeholders: | None   |  |
| Description:                   | The use case describes the event where a user has forgotten their password. The user will request to reset their password. The system will prompt the user to enter their email address. The user will provide their email address and the system will verify the account existence. The system will generate a new password for the user and the new password along with their username will be sent to the relevant email address. |  |
| Pre-condition:                 | The user must be registered on the system  |  |
| Trigger:                       | The user requests to reset their password  |  |
|                                | Actors Action  | Systems response                                     |

|                                  |  |  |
|----------------------------------|--|--|
| <b>Typical course Of events:</b> | <b>Step 1:</b> The user requests to reset their password   | <b>Step 2:</b> The system prompts the user to provide the following details: <ul style="list-style-type: none"> <li>• Email</li> </ul>   |
|                                  | <b>Step 3:</b> The system will prompt the user to enter the following details: <ul style="list-style-type: none"> <li>• Email</li> </ul><br>The user submits   | <b>Step 4:</b> The system validates that the users Email is in the required format <b>[ALT]</b>  |
|                                  |  | <b>Step 5:</b> The system validates the Email provided against the Email Attribute in the <b>Employee</b> entity <b>[ALT]</b>  |
|                                  |  | <b>Step 6:</b> The system Generates a new password and updates the Password Attribute for the User in the <b>User</b> Entity where the User_ID attribute in the <b>User</b> entity is equal to the User_ID foreign key in the <b>Employee</b> entity |
|                                  |  | <b>Step 7:</b> The system will generate a email providing the user with their username and new password  |
|                                  |  | <b>Step 8:</b> The system notifies the user of the successful password reset   |
| <b>Alternate courses:</b>        | <b>[ALT] Step 4:</b> The email provided is in the incorrect format. Return to step 3   |  |
|                                  | <b>[ALT] Step 5:</b> No Employee has been found. The system validates the email against the Email attribute in the <b>Admin</b> entity. The system Generates a new password and updates the Password Attribute for the User in the <b>User</b> Entity where the User_ID attribute in the <b>User</b> entity is equal to the User_ID foreign key in the <b>Admin</b> entity |  |
| <b>Conclusion:</b>               | The system notifies the user of the successful password reset  |  |
| <b>Post-condition:</b>           | The users password is reset  |  |
| <b>Business rules</b>            |  |  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

Table 136: 1.3 Forgot Password

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 13 May 2023      Version: 1.0 |                   |              |
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0 |

| Use case name:                 | Update Password   | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 1.4   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor         | User  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | The user requests to update their password. The user will be prompted to provide their existing password and the new password. The system will validate the existing password and update existing password to the new password entered. A notification is generated to indicate the |  |

|                                  |  |   |
|----------------------------------|--|---|
|                                  | password has been updated successfully and the user is logged out of the system  |   |
| <b>Pre-condition:</b>            | The user has to be logged in to the system.  |   |
| <b>Trigger:</b>                  | The user requests to update their password   |   |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                  | <b>Step 1:</b> The user requests to update their password  | <b>Step 2:</b> The system prompts the user to provide the following: <ul style="list-style-type: none"><li>• Current Password</li><li>• New Password</li></ul>                                    |
|                                  | <b>Step 3:</b> The user provides the following information <ul style="list-style-type: none"><li>• Current Password</li><li>• New Password</li></ul> | <b>Step 4:</b> The system validates the users input and ensures that it is in the correct format.   |
|                                  |  | <b>Step 5:</b> The system will validate the password provided against the Password attribute in the <b>User</b> entity where the User_ID is equal to the User_ID stored in the token <b>[ALT]</b> |
|                                  |  | <b>Step 6:</b> The system notifies the user that the password has been updated successfully and sends an email to the user providing them with their username and new password                    |
|                                  |  | <b>Step 7:</b> The user is logged out of the system   |
| <b>Alternate courses:</b>        | <b>[ALT] Step 5:</b> The user password validation is failed. The User is notified of the unsuccessful password update, Return to Step 3              |   |
| <b>Conclusion:</b>               | The user is logged out of the system   |   |

|  |   |
|--|---|
| <b>Post-condition:</b>                               | The user's password is updated  |
| <b>Business rules</b>                                | None  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                              |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                              |

Table 137: 1.4 Update Password

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 13 May 2023      Version: 1.0 |                   |              |
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0 |

|                                       |                                   |  |
|---------------------------------------|-----------------------------------|--|
| <b>Use case name:</b>                 | <b>Create Procurement Request</b> | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 3.1                               | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High                              | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo                              | System Design: <input type="checkbox"/>              |
| <b>Primary business actor</b>         | User                              |  |
| <b>Primary system actor</b>           | None                              |  |
| <b>Other participating actors:</b>    | None                              |  |
| <b>Other interested stakeholders:</b> | None                              |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Description:</b>              | <p>The use case describes the event where a user would like to create a procurement request. The user will be prompted to enter the details for their request. The system will validate the information added and save the request in the systems database. The system will capture the files provided and upload it in the systems directory. The system will capture the file paths saved and create a new entry in the systems database for the newly added procurement quotes. The system Generates a notification for the user indicating that the procurement request has been generated successfully.</p> |   |
| <b>Pre-condition:</b>            | <p>The user has to be logged in to the system.</p>   |   |
| <b>Trigger:</b>                  | <p>The user requests to add a procurement request to the system</p>  |   |
| <b>Typical course Of events:</b> | <p><b>Actors Action</b></p> <p><b>Step 1:</b> The user requests to add a procurement request to the system</p>   | <p><b>Systems response</b></p> <p><b>Step 2:</b> The approved Vendors is retrieved through the Foreign key Vendor_Status_ID in the <b>Vendor</b> Entity which is equal to the Vendor_Status_ID in the <b>Vendor_Status</b> entity where the Name attribute of the <b>Vendor_Status</b> Entity is set to <i>approved</i></p> |
|                                  |  | <p><b>Step 3:</b> The system prompts the user to provide the following details for a approved vendor:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor Name</li> <li>• Procurement Quote</li> </ul>   |
|                                  | <p><b>Step 3:</b> The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor Name</li> <li>• Procurement Quote</li> </ul> <p>The user submits<br/><b>[ALT]</b></p>  | <p><b>Step 4:</b> The system validates the users response and ensures that it is in the correct format. <b>[ALT]</b></p>  |

|                           |  |  |
|---------------------------|--|--|
|                           |  | <p><b>Step 5:</b> The system inserts the following into the <b>Procurement_Request</b> entity which contains the following details:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID: Incremented ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>[ALT]</p>  |
|                           |  | <p><b>Step 6:</b> The system captures the Files uploaded by the user and creates a new directory based on the VendorName and Filename. The system uploads the files</p> <p>[ALT]</p>   |
|                           |  | <p><b>Step 7:</b> The system retrieves the paths from the newly uploaded files and inserts the new procurement quote data in the <b>Procurement_Request_Quote entity</b> which contains the following details:</p> <ul style="list-style-type: none"> <li>• Quote_ID: Incremented new ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>[ALT]</p> |
|                           |  | <p><b>Step 8:</b> The system Notifies the user of the successful creation of the procurement request</p>   |
| <b>Alternate courses:</b> | <p>[ALT] <b>Step 3:</b> The user requests to provide the details for another vendor. The system prompts the user to enter the following details:</p> |  |

|  |  |
|--|--|
|  |  |
|  | <b>[ALT] Step 4:</b> The inputs provided is not in the correct format Return to step 3           |
|  | <b>Step 4:</b> The system has encountered an error. The user is notified of the creation failure |
|  | <b>Step 5:</b> The system has encountered an error. The user is notified of the creation failure |
|  | <b>Step 6:</b> The system has encountered an error. The user is notified of the creation failure |
| <b>Conclusion:</b>                                   | The user is notified of the successful procurement request creation                              |
| <b>Post-condition:</b>                               | A new procurement request and procurement request quote is added to the system                   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires internet access</li> </ul>            |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 138: 3.1 Create Procurement Request

|  |                    |              |
|--|--------------------|--------------|
| System Name: Procion System<br>Sub-System: |                    |              |
| Author: Jason van der Merwe                | Date: 25 July 2023 | Version: 1.0 |

|                       |                                 |   |
|-----------------------|---------------------------------|---|
| <b>Use case name:</b> | <b>View Procurement Request</b> | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>   | 3.2                             | Business Requirements: <input type="checkbox"/> |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Priority:</b>                      | High  | System Analysis: <input checked="" type="checkbox"/>   |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>  |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | <p>The use case describes the event where a user would like to view the procurement requests stored on the system. The system will display all of the procurement requests along with each request's quotes to the user. The system will prompt the user to provide their search criteria and the system will dynamically filter through the procurement requests and display it to the user.</p> |  |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.   |  |
| <b>Trigger:</b>                       | The user requests to view the procurement requests stored on the system   |  |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>  | <b>Systems response</b>  |
|                                       | <p><b>Step 1:</b> The user requests to view the procurement requests stored on the system</p>   | <p><b>Step 2:</b> The system will retrieve all of the records from the Procurement_Request entity along with the associated quotes from the Procurement_Quote entity where the Procurement_Request_ID variable in the Procurement_Request entity is equal to the Procurement_Request_ID foreign key in the Procurement_Request_Quote entity.</p> |

|  |   |  |
|--|---|--|
|  |   | <b>Step 3:</b> The system displays the Results to the user   |
|  |   | <b>Step 4:</b> The system prompts the user to provide their search criteria  |
|  | <b>Step 5:</b> The user provides their search criteria            | <b>Step 6:</b> The system filters dynamically by comparing the search criteria of the user against the <b>Name</b> attribute contained in the <b>Procurement_Request</b> entity.<br><br>The system displays the search results to the user |
| <b>Alternate courses:</b>                            | None  |  |
| <b>Conclusion:</b>                                   | The system displays the filtered procurement requests to the user |  |
| <b>Post-condition:</b>                               | The system displays the filtered procurement requests to the user |  |
| <b>Business rules</b>                                |   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>          |  |
| <b>Assumptions:</b>                                  | None  |  |
| <b>Open issues:</b>                                  | None  |  |

Table 139: 3.2 View Procurement Request

|  |                   |              |
|--|-------------------|--------------|
| System Name: Procion System<br>Sub-System: |                   |              |
| Author: Jason van der Merwe                | Date: 13 May 2023 | Version: 1.0 |

|                                       |  |   |
|---------------------------------------|--|---|
| <b>Use case name:</b>                 | Update Procurement Request   | <b>USE CASE TYPE</b>  |
| <b>Use case id:</b>                   | 3.3  | Business Requirements: <input type="checkbox"/>   |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/>  |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>   |
| <b>Primary business actor</b>         | User   |   |
| <b>Primary system actor</b>           | None   |   |
| <b>Other participating actors:</b>    | None   |   |
| <b>Other interested stakeholders:</b> | None   |   |
| <b>Description:</b>                   | The use case describes the event where a user will like to update a procurement request on the system. The system retrieves the current data from the systems database for the procurement request and the quotes that the user would like to update. The user will provide their details. The system will update the relevant records in the database with the new details received. The system generates a successful update notification for the user |   |
| <b>Pre-condition:</b>                 | <ul style="list-style-type: none"> <li>• The user has to be logged in to the system.</li> <li>• There must be atleast one procurement request stored on the system</li> </ul>  |   |
| <b>Trigger:</b>                       | The user requests to update a procurement request  |   |
| <b>Typical course Of events:</b>      | <b>Actors Action</b>   | <b>Systems response</b>   |
|                                       | <b>Step 1:</b> The user requests to update a procurement request   | <b>Step 2:</b> The system prompts the user to provide the following details: <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Replacement Quote</li> </ul> |

|  |   |  |
|--|---|--|
|  | <p><b>Step 3:</b> The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Replacement Quote</li> </ul> <p>The user submits</p> | <p><b>Step 4:</b> The system validates the details entered by the user to ensure that it is in the correct format [ALT]</p>  |
|  |   | <p><b>Step 5:</b> The system captures the details entered by the user and updates the relevant record in the <b>Procurement_Request</b> entity along with the relevant quotes in the <b>Procurement_Request_Quote</b> entity where the Procurement_Request_ID in the <b>Procurement_Request</b> entity is equal to the Procurement_Request_ID attribute in the <b>Procurement_Request_Quote</b> entity [ALT]</p> |
|  |   | <p><b>Step 6:</b> The system displays a successful update notification to the user to inform them that the Procurement request has been updated successfully</p>   |
| <b>Alternate courses:</b>                            | <p>[ALT] <b>Step 4:</b> The inputs provided is not in the correct format Return to step 3</p>   |  |
|  | <p>[ALT] <b>Step 5:</b> The update has failed. The user is displayed with a unsuccessful update notification</p>  |  |
| <b>Conclusion:</b>                                   | <p>The user is notified of the successful procurement request update</p>  |  |
| <b>Post-condition:</b>                               | <p>A existing procurement request and associated procurement request quotes is updated on the system</p>  |  |
| <b>Business rules</b>                                | <p>None</p>   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |

|                     |      |
|---------------------|------|
| <b>Assumptions:</b> | None |
| <b>Open issues:</b> | None |

Table 140: 3.3 Update Procurement Request

|   |  |  |
|---|--|--|
| System Name: Procion System<br>Sub-System:<br>Author: Jason van der Merwe      Date: 25 July 2023      Version: 1.0 |  |  |
|---|--|--|

| Use case name:                 | Delete Procurement Request   | USE CASE TYPE  |
|--------------------------------|--|--|
| Use case id:                   | 3.4  | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo   | System Design: <input type="checkbox"/>              |
| Primary business actor         | User   |  |
| Primary system actor           | None   |  |
| Other participating actors:    | None   |  |
| Other interested stakeholders: | None   |  |
| Description:                   | The user requests to delete a procurement request. The system will prompt the user to select a procurement request that they wish to delete. The user selects a procurement request that they wish to delete. The system will remove the Procurement Quotes in the database which is associated with the procurement request to delete. The system will also remove all of the files stored on the system associated with the procurement request. The user is displayed with a successful deletion notification |  |

|                                  |  |  |
|----------------------------------|--|--|
| <b>Pre-condition:</b>            | The user has to be logged in to the system.  |  |
| <b>Trigger:</b>                  | The user requests to delete a Procurement request  |  |
| <b>Typical course Of events:</b> | <b>Actors Action</b>   | <b>Systems response</b>  |
|                                  | <b>Step 1:</b> The user requests to delete a Procurement request                         | <b>Step 2:</b> The system prompts the user to select a procurement request that they wish to delete.   |
|                                  | <b>Step 3:</b> The user selects a procurement request to be deleted                      | <b>Step 4:</b> The system prompts the user for confirmation of deletion of the procurement request   |
|                                  | <b>Step 5:</b> The user confirms the procurement request deletion<br><b>[ALT]</b>        | <b>Step 6:</b> The system deletes the Procurement Request from the <b>Procurement_Request</b> entity along with the associated quotes in the <b>Procurement_Request_Ques</b> entity where the <b>Procurement_Request_ID</b> attribute in the <b>Procurement_Request</b> entity is equal to the <b>Procurement_Request_ID</b> in the <b>Procurement_Request_Ques</b> entity |
|                                  |  | <b>Step 7:</b> The system displays a successful delete notification to inform the user of the successful deletion of the procurement request   |
| <b>Alternate courses:</b>        | <b>[ALT] Step 5:</b> The user denies the procurement request deletion. The use case ends |  |
| <b>Conclusion:</b>               | The user is notified of the successful procurement request deletion                      |  |
| <b>Post-condition:</b>           | The Procurement request along with its respective quotes is removed from the system      |  |
| <b>Business rules</b>            | None   |  |

|  |   |
|--|---|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>The user requires internet access</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                            |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                            |

Table 141: 3.4 Delete Procurement Request

---

USE CASE 3.5-3.8, 6.5 & 6.10-6.14

|  |                  |              |
|--|------------------|--------------|
| System Name: Procision System<br>Sub-System: 3 Procurement |                  |              |
| Author: Werner Schutte                                     | Date: 2023/07/23 | Version: 1.0 |

| Use case name:                 | Pending Procurement Request   | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 3.5   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | high  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor:        | User (PBA)  |  |
| Primary system actor:          | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>The use case describes the events where a procurement request was created and is awaiting approval. The user would like to approve or reject a newly created request. The system will request that the user provide selection details. The user will provide the selections details. The system will then display the following details:</p> <ul style="list-style-type: none"> <li>• Company Name</li> <li>• Company Email</li> <li>• Description</li> <li>• Preferred Quote</li> <li>• Comparative 1 (if vendor is not from approved list)</li> <li>• Comparative 2 (if vendor is not from approved list)</li> </ul> <p>The system will prompt the user to provide accept or rejection details. The user will provide the required detail.</p> |  |

|                                  |   |   |
|----------------------------------|---|---|
| <b>Pre-condition:</b>            | The system must contain procurement request.  |   |
| <b>Trigger:</b>                  | The user would like to accept or reject a request.  |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b><br><b>Step 1:</b> The user request to accept or reject a procurement request. | <b>System Response</b><br><b>Step 2:</b> The system will retrieve all the related procurement request from the following tables:<br><b>Procurement_Request</b> table:<br><ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Description</li> </ul> <b>Procurement_Request_Quote:</b><br><ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• PreferredQuote</li> </ul> <b>Requisition_Status:</b><br><ul style="list-style-type: none"> <li>• Requisition_Status_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>The 3 tables are linked by Procurement_Request_ID</p> <p>The system will prompt the user to provide accept or reject details</p> |
|                                  | <b>Step 4:</b> The Requestor Provides accept details.<br><b>[ALT]</b>                             | <b>Step 5:</b> The system responds by updating the Requisition_Status_ID in the <b>Procurement_Request</b> table based on the provided details  |

|  |   |
|--|---|
|  |   |
|  | <b>Step 6:</b> The system then displays a success notification based on the provided details and notifies the employee that created the request.  |
| <b>Alternate courses:</b>                            | <p><b>ALT-Step 4a:</b> The user provides rejection details. <b>Proceed to step 5.</b></p> <p><b>ALT-Step 4b:</b> The user decides to not provide details. <b>Use case terminates.</b></p> |
| <b>Conclusion:</b>                                   | The system displays an notification based on provided details.  |
| <b>Post-condition:</b>                               | The procurement request Requisition_Status_ID is updated to either rejected or accept.  |
| <b>Business rules</b>                                | Only users that have the correct role can view and approve or reject the request on the system.   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

*Table 142: 3.5 Pending Procurement Request*

System Name: Procion System

Sub-System: 3 Procurement

Author: Werner Schutte

Date: 2023/07/23

Version: 1.0

| Use case name:                 | Place procurement Details   | USE CASE TYPE  |
|--------------------------------|---|--|
| Use case id:                   | 3.6   | Business Requirements: <input type="checkbox"/>      |
| Priority:                      | high  | System Analysis: <input checked="" type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input type="checkbox"/>              |
| Primary business actor:        | User (PBA)  |  |
| Primary system actor:          | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>The user would like to place a procurement request. The system will prompt the user to provide the following details:</p> <ul style="list-style-type: none"> <li>• Buyer Name</li> <li>• Buyer Email</li> <li>• Item Type</li> <li>• Select Consumable Item (if Item Type = Consumables)</li> <li>• Quantity (if Item Type = Consumables)</li> <li>• Asset Name (if Item Type = Assets)</li> <li>• Description of Asset (if Item Type = Assets)</li> <li>• Account Code</li> <li>• Payment Type</li> <li>• Deposit (True or False)</li> <li>• Deposit Amount (Deposit = true)</li> <li>• Deposit Due Date (Deposit = true)</li> <li>• Full Payment Made (True or False)</li> <li>• Paid On Date (Full Payment Made = true)</li> <li>• Upload Receipt (Full Payment Made = true)</li> <li>• Proof of payment (True or False)</li> <li>• Proof of payment document (Proof of payment = true)</li> <li>• Total Amount</li> <li>• Total Amount Due Date</li> </ul> |  |

|                                  | <ul style="list-style-type: none"> <li>Comments</li> </ul> <p>The user provides the required details. The system validates and request confirmation details. The user provides the confirmation detail. The system then creates the placed procurement request details.</p> |   |
|----------------------------------|---|---|
| <b>Pre-condition:</b>            | The user must be logged into the system.  |   |
| <b>Trigger:</b>                  | The user would like to place a procurement request.   |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>   | <b>System Response</b>  |
|                                  | <p><b>Step 1:</b> The user would like to place a procurement request.</p>   | <p><b>Step 2:</b> The system displays and prompts the user to provide the following details:</p> <ul style="list-style-type: none"> <li>Buyer Name</li> <li>Buyer Email</li> <li>Item Type</li> <li>Select Consumable Item (if Item Type = Consumables)</li> <li>Quantity (if Item Type = Consumables)</li> <li>Asset Name (if Item Type = Assets)</li> <li>Description of Asset (if Item Type = Assets)</li> <li>Account Code</li> <li>Payment Type</li> <li>Deposit (True or False)</li> <li>Deposit Amount (Deposit = true)</li> <li>Deposit Due Date (Deposit = true)</li> <li>Full Payment Made (True or False)</li> <li>Paid On Date (Full Payment Made = true)</li> <li>Upload Receipt (Full Payment Made = true)</li> <li>Proof of payment (True or False)</li> <li>Proof of payment document (Proof of payment = true)</li> <li>Total Amount</li> <li>Total Amount Due Date</li> <li>Comments</li> </ul> |
|                                  | <p><b>Step 3:</b> The user will provide the required details</p>  | <p><b>Step 4:</b> The system will validate the inputted details and request confirmation details from user.</p>   |

|  |   |
|--|---|
|  | [ALT]   |
| <p><b>Step 5:</b> The user provides the confirmation details.</p> <p>[ALT]</p> | <p><b>Step 6:</b> The system responds by adding the following details as attribute into:</p> <p><b>PROCUREMENT_DETAILS</b> table:</p> <ul style="list-style-type: none"> <li>• Employee_ID (based on the user placing the procurement request)</li> <li>• Procurement_Status_ID (Either set to 1 or 3 based on whether procurement gets flagged)</li> <li>• Sign_Off_Status (set as default to 1)</li> <li>• Procurement_Payment_Status_ID (based on whether payment has been partially, fully or still needs payment)</li> <li>• Account_Code (based on account code selected by user)</li> <li>• Payment_Method_ID (based on payment method selected by user)</li> <li>• Item_Type (either assets or consumables)</li> <li>• Buyer_Name</li> <li>• Buyer Email</li> <li>• Deposit_Required (Boolean value)</li> <li>• Full_Payment_Due_Date (Boolean value)</li> <li>• Total_Amount</li> <li>• Payment_Made</li> <li>• Comment</li> <li>• Proof_Of_Payment_Required (Boolean value)</li> </ul> <p><b>DEPOSIT</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Deposit_Due_Date</li> <li>• Deposit_Amount</li> <li>• Amount_Outstanding (based on total amount subtracted by deposit amount)</li> </ul> <p><b>PAYMENT MADE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Paid_On_Date</li> <li>• Receipt_Upload</li> </ul> <p><b>PROOF_OF_PAYMENT</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Proof_of_Payment_Doc</li> </ul> |

|                           |   |
|---------------------------|---|
|                           | <p><b>PROCUREMENT_CONSUMABLE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID (PK,FK)</li> <li>• Consumable_ID (PK,FK)</li> <li>• Quantity</li> </ul> <p><b>VENDOR_CONSUMABLE</b> table:</p> <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Vendor_ID</li> <li>• Vendor_Consumable_ID</li> </ul> <p><b>PROCUREMENT_ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID</li> <li>• Asset_ID</li> </ul> <p><b>ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p><b>VENDOR_ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Asset_ID</li> <li>• Vendor_ID</li> <li>• Vendor_Asset_ID</li> </ul> |
|                           | <p><b>Step 7:</b> The system then displays a success notification to the user.</p>  |
| <b>Alternate courses:</b> | <p><b>ALT-Step 4:</b> Validation error is display to the user. <b>Return to step 3.</b></p> <p><b>ALT-Step 5:</b> The user does not want to continue with place procurement request. <b>Use Case Terminates.</b></p>  |
| <b>Conclusion:</b>        | The procurement details will be added on the system and a success notification is displayed.  |
| <b>Post-condition:</b>    | Record is created in the <b>PROCUREMENT_DETAILS</b> table, <b>DEPOSIT</b> table, <b>PAYMENT_MADE</b> table, <b>PROOF_OF_PAYMENT</b> table, <b>PROCUREMENT_CONSUMABLE</b> table, <b>VENDOR_CONSUMABLE</b> table, <b>PROCUREMENT_ASSET</b> table, <b>ASSET</b> table and <b>VENDOR_ASSET</b> table  |

|  |  |
|--|--|
| <b>Business rules</b>                                | Only the user with a role that has the authority can place a procurement request on to the system. |
| <b>Implementation constraints and specifications</b> | None   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 143: Place Procurement Details

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 3 Procurement |                  |              |
| Author: Werner Schutte                                   | Date: 2023/07/23 | Version: 1.0 |

| <b>Use case name:</b>                 | Flagged Procurement Request | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|-----------------------------|--|
| <b>Use case id:</b>                   | 3.7                         | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | high                        | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo                        | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | User (PBA)                  |  |
| <b>Primary system actor:</b>          | None                        |  |
| <b>Other participating actors:</b>    | None                        |  |
| <b>Other interested stakeholders:</b> | None                        |  |

|                                  |   |   |
|----------------------------------|---|---|
| <b>Description:</b>              | <p>The use case describes the event where the user would like to approve or reject a placed procurement request that has with total amount exceeding the person who placed it mandate limit.</p> <p>The use case begins when the system request selection detail. The user will provide selection details. The system will load the selected procurement details and prompt the requestor for accept or rejection details. When the requestor provides the accept or rejection details the system will respond by updating the procurement request. If successful it will display a notification based on provided details.</p> |   |
| <b>Pre-condition:</b>            | The user must be logged into the system.  |   |
| <b>Trigger:</b>                  | The user would like to accept or reject a flagged procurement request.  |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>   | <b>System Response</b>  |
|                                  | <p><b>Step 1:</b> The user would like to accept or reject a flagged procurement request.</p>  | <p><b>Step 2:</b> The system responds by retrieving the following details:</p> <p><b>PROCUREMENT_DETAILS</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID</li> <li>• Employee_ID</li> <li>• Procurement_Status_ID</li> <li>• Sign_Off_Status</li> <li>• Procurement_Payment_Status_ID</li> <li>• Account_Code</li> <li>• Payment_Method_ID</li> <li>• Item_Type (either assets or consumables)</li> <li>• Buyer_Name</li> <li>• Buyer Email</li> <li>• Deposit_Required (Boolean value)</li> <li>• Full_Payment_Due_Date (Boolean value)</li> <li>• Total_Amount</li> <li>• Payment_Made</li> <li>• Comment</li> <li>• Proof_Of_Payment_Required (Boolean value)</li> </ul> |

|                           |   |
|---------------------------|---|
|                           | <p><b>DEPOSIT</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Deposit_Due_Date</li> <li>• Deposit_Amount</li> <li>• Amount_Outstanding</li> </ul> <p><b>PAYMENT_MADE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Paid_On_Date</li> <li>• Receipt_Upload</li> </ul> <p><b>PROOF_OF_PAYMENT</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Proof_of_Payment_Doc</li> </ul> <p><b>PROCUREMENT_CONSUMABLE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID (PK,FK)</li> <li>• Consumable_ID (PK,FK)</li> <li>• Quantity</li> <li>• Procurement_Consumable_ID</li> </ul> <p><b>PROCUREMENT_ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID</li> <li>• Asset_ID</li> <li>• Procurement_Asset_ID</li> </ul> <p><b>ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Asset_ID</li> <li>• Name</li> <li>• Description</li> </ul> |
|                           | <p><b>Step 3:</b> The system prompts the user to provide accept or rejection details.</p>   |
|                           | <p><b>Step 4:</b> The requestor provides accept or rejection details.<br/><b>[ALT]</b></p> <p><b>Step 5:</b> The system responds by updating Procurement_Payment_Status_ID of <b>PROCUREMENT_DETAILS</b> table based on the provided details.</p>   |
|                           | <p><b>Step 6:</b> The system then displays a notification based on the provided details</p>   |
| <b>Alternate courses:</b> | <p><b>ALT-Step 4:</b> The user does not want to continue with the use case.<br/><b>Use case terminates.</b></p>   |

|  |   |
|--|---|
| <b>Conclusion:</b>                                   | The system displays a notification based on provided details.                               |
| <b>Post-condition:</b>                               | The procurement request Requisition_Status_ID is updated to either rejected or accept.      |
| <b>Business rules</b>                                | Only the user with the right role can approve or reject a procurement request on the system |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>                                    |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                                    |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                                    |

Table 144: Flagged Procurement Request

System Name: Procion System

Sub-System: 6 Vendor

Author: Werner Schutte

Date: 2023/07/22

Version: 1.0

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Approve onboard request  | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 6.5  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | GRC Officer (PBA)  |  |
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the GRC Officer would like to approve or reject a vendor from the available onboard request. The user will request to approve or reject a selected onboard request. The system will prompt the user to select one of the available vendors. The user provides the selection details. The system then prompts the user to provide the following details:</p> <ul style="list-style-type: none"> <li>• BEE Rating</li> <li>• BEE Certificate</li> <li>• BEE expiration date</li> <li>• DUE_DILIGENCE</li> <li>• POPI</li> <li>• CONTRACTED_PARTNER_TYPE</li> </ul> <p>The requestor will then provide the details. The system then captures and validates the details. The approved vendor's status is updated in the <b>Vendor_Status</b> table and the related details are added or updated in the <b>Vendor</b> table, <b>Vendor_POPIA</b> table, <b>Due_Diligence</b> table, <b>POPI</b> table. Finally, a success notification is displayed to the GRC Officer that the request has been approved and available for the requestor to add the final details to the vendor.</p> |  |
| <b>Pre-condition:</b>                 | The GRC Officer must be logged into the system.  |  |

|                                  |   |   |
|----------------------------------|---|---|
| <b>Trigger:</b>                  | GRC Officer would like to approve or reject a vendor from the available onboard request     |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>   | <b>System Response</b>  |
|                                  | <p><b>Step 1:</b> The user wants to approve or reject a Vendor from an onboard request.</p> | <p><b>Step 2:</b> The system responds by requesting the user to select an available onboard request</p>   |
|                                  | <p><b>Step 3:</b> The user will provide the onboard request select detail.</p>              | <p><b>Step 4:</b> The system responds by retrieving the following details from the related tables:</p> <p><b>Vendor</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> <li>• PreferredVendor</li> </ul> <p><b>Onboard_Request</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Onboard_Request_ID (FK)</li> <li>• Vendor_ID</li> <li>• Quotes</li> <li>• User_ID</li> <li>• Status_ID</li> </ul> <p><b>Vendor</b> table is linked to the <b>Onboard_Request</b> table via the <b>Vendor_ID</b> attribute.</p> <p>The following details are then displayed:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Email</li> <li>• Quotes</li> <li>• Preferred Vendor</li> </ul> |

|  |  |  |
|--|--|--|
|  |  | <p>The system prompts the user to select a vendor from the selected onboard request.</p> <p>[ALT]</p>  |
|  | <p><b>Step 5:</b> The user will provide the vendor selection detail. [ALT]</p>     | <p><b>Step 6:</b> The system then prompts the user to provide the following details:</p> <ul style="list-style-type: none"> <li>• BEE Rating</li> <li>• BEE Certificate</li> <li>• BEE expiration date</li> <li>• DUE_DILIGENCE Checklist</li> <li>• POPI</li> </ul>   |
|  | <p><b>Step 7:</b> The user will provide the selected vendor information. [ALT]</p> | <p><b>Step 8:</b> The system saves or updates the following details by storing it as attributes in the following tables:</p> <p><b>Onboard_Request:</b></p> <ul style="list-style-type: none"> <li>• Onboard_Request_Id (PK)</li> <li>• User_Id (PK,FK)</li> <li>• Vendor_ID (PK,FK)</li> <li>• Status_ID (FK)</li> <li>• Quotes</li> </ul> <p><b>Vendor:</b></p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Vendor_Status_ID</li> <li>• Number_Of_Times_Used (default value set to 0)</li> <li>• Sole_Supplier_Provided</li> </ul> <p><b>Vendor_BEE:</b></p> <ul style="list-style-type: none"> <li>• BEE_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> </ul> <p><b>DUE_DILIGENCE:</b></p> <ul style="list-style-type: none"> <li>• Due_Diligence_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> </ul> |

- Income\_Tax\_Number\_Provided
- Tax\_Clearance\_Certificate\_Provided
- Vat\_Number\_Provided
- Vat\_Reg\_Certificate\_Provided
- Company\_Reg\_Doc\_Provided
- Letter\_Of\_Good\_Standing\_Provided
- B\_BBEE\_Certificate\_Provided
- Direcor\_Details\_Provided
- Company\_Resolution\_Agreement\_Provided
- Subcontractor\_Name\_Provided
- Company\_Details\_Provided
- Individual\_Details\_Provided
- General\_Liability\_Insurance\_Present
- Cyber\_Insurance\_Present
- Proffesional\_Indemnity\_Insurance\_Present
- Other\_Insurance\_Required
- Licenses\_Required
- Accreditation\_Required
- Proffesional\_Membership\_Required
- Business\_Continuity\_Present
- Disaster\_Recovery\_Plan\_Present
- POPI\_Present
- Data\_Security\_Breaches\_Present
- Site\_Visits\_Present
- Information\_Security\_Policy\_Present
- Privacy\_Policy\_Present\_Present
- Data\_Retention\_Destruction\_Policy\_Present
- Anti\_Bribery\_Corruption\_Policy\_Present
- Ethics\_Policy\_Present
- Conflict\_Of\_Interest\_Policy\_Present
- Customer\_Complaints\_Policy\_Present
- Business\_References\_Present

#### **POPI:**

- POPI\_ID (PK)
- Contracted\_Partner\_Type\_ID (FK)
- Due\_Diligence\_ID (FK)
- Personal\_Data\_Purpose
- DataProcessing\_JointController\_Agreement
- Confidentiality\_Importance\_Highlighted
- Contract\_Audits\_Provisions\_Provided
- Activity\_Liability\_Present
- Third\_Party\_Data\_Processing\_Provisioned
- Contract\_End\_Data\_Management\_Provided
- Personal\_Data\_Processing\_Details\_Present
- Processing\_Activities\_Certification\_Held

#### **Insurnace:**

- General\_Liability\_Insurance\_Present\_Document
- Cyber\_Insurance\_Present\_Document

|                    |   |
|--------------------|---|
|                    | <ul style="list-style-type: none"> <li>• Professional_Indemnity_Insurance_Present_Document</li> <li>• Other_Insurance_Required_Document</li> </ul> <p>The Vendor_ID is linked to the associative entities:</p> <ul style="list-style-type: none"> <li>• Vendor_BEE</li> <li>• DUE_DILIGENCE</li> <li>• POPI</li> <li>• Insurance</li> </ul> <p>where their related primary keys:</p> <ul style="list-style-type: none"> <li>• BEE_ID</li> <li>• Due_Diligence_ID</li> <li>• POPI_ID</li> <li>• InsuranceID</li> </ul> <p>is auto incremented by 1.</p>  |
|                    | <p><b>Step 9:</b> The system displays a notification indicating the onboard request has been successfully approved with the required details.</p> <p>[ALT]</p>  |
|                    | <p><b>Step 10:</b> The system generates a notification to send to the user who created the onboard request.</p>   |
| Alternate courses: | <p><b>ALT-Step 4a:</b> The selected onboard request contains a sole supplier. <b>Proceed to step 6.</b></p> <p><b>ALT-Step 4b:</b> The managing director wants to approve or reject a sole supplier onboard request. The system responds by retrieving the following details from the related tables:</p> <p><b>Vendor</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> </ul> <p><b>Onboard_Request</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Onboard_Request_ID (PK)</li> <li>• Vendor_ID (PK,FK)</li> <li>• Quotes</li> </ul> |

- User\_ID (PK,FK)
- Status\_ID (FK)

**Sole\_Supplier** table containing the following attributes:

- Sole\_Supplier\_ID (PK)
- Vendor\_ID (FK)
- MD\_Approval
- Date
- Reason

**Vendor** table is linked to the **Onboard\_Request** table and **Sole\_Supplier** table via the Vendor\_ID attribute.

The following details are then displayed:

- Vendor Name
- Vendor Email
- Reason
- Optional Quote

The system prompts the user to approve or reject the onboard request. **Proceed to Step 5.**

**ALT-Step 5a:** The managing director provides the onboard request approval or rejection detail. **Proceed to step 9.**

**ALT-Step 5b:** The managing director cancels providing any details. **Use case terminates.**

**ALT-Step 5c:** The GRC Officer does not want to continue with selected onboard request. **Return to step 1.**

**ALT-Step 5c:** The GRC Officer wants to reject the onboard request. **Use case terminates.**

**ALT Step 7:** The user does not want to continue with the selected vendor. **Return to step 4.**

**ALT Step 9:** The managing director provided the approval or rejection of onboard request details. The system responds by updating the attribute “MD\_Approval” in:

**Sole\_Supplier** table containing the following attributes:

- Sole\_Supplier\_ID (PK)
- Vendor\_ID (FK)

|  |   |
|--|---|
|  | <ul style="list-style-type: none"> <li>• MD_Approval</li> <li>• Date</li> <li>• Reason</li> </ul> <p><b>Use case terminate.</b></p>   |
| <b>Conclusion:</b>                                   | A vendor has been approved from an onboard request and is ready for the next step to be officially added to the system.   |
| <b>Post-condition:</b>                               | The vendor obtained all the required detail and has been added in the <b>Vendor_BEE</b> table, <b>Vendor_POPIA</b> table, <b>DUE_DILIGENCE</b> table and <b>POPI</b> table. |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• Only the user that has the correct role can approve or reject a onboard request.</li> </ul>  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

Table 145: 6.5 Approve onboard request

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                       |                                       |  |
|-----------------------|---------------------------------------|--|
| <b>Use case name:</b> | Export due diligence vendor checklist | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>   | 6.10                                  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>      | high                                  | System Analysis: <input checked="" type="checkbox"/> |

|                                       |  |   |
|---------------------------------------|--|---|
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>   |
| <b>Primary business actor:</b>        | User (PBA)   |   |
| <b>Primary system actor:</b>          | None   |   |
| <b>Other participating actors:</b>    | None   |   |
| <b>Other interested stakeholders:</b> | None   |   |
| <b>Description:</b>                   | <p>The user would like to generate a due diligence checklist for a specific vendor. The system will prompt the user to select a vendor. The user will provide the vendor selection detail. The system responds by displaying a basic view of the vendor information and request the user to generate the due diligence checklist. The user provides the required detail. The system responds by generating a pdf file with the required due diligence checklist details and displaying it in a new tab for the user.</p> |   |
| <b>Pre-condition:</b>                 | The user must be logged into the system.   |   |
| <b>Trigger:</b>                       | The user would like to export a vendor's due diligence checklist   |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b><br><b>Step 1:</b> The user would like to export a vendor's due diligence checklist.  | <b>System Response</b><br><b>Step 2:</b> System responds by requesting the user to select a approved vendor   |
|                                       | <b>Step 3:</b> The user provides the approved vendor selection details.  | <b>Step 4:</b> The system responds by displaying the following details of the approved vendor: <ul style="list-style-type: none"> <li>• Name</li> <li>• Email</li> <li>• Quote</li> <li>• PreferredVendor</li> <li>• Generate Due Dilligence checklist link</li> </ul> and prompts the user to provide selection action details |

|  |  |  |
|--|--|--|
|  | <b>Step 5:</b> The user provides the selection action details.<br><br>[ALT]  | <b>Step 6:</b> The system responds by generating a pdf file with the relevant due diligence checklist detail with relevant POPI checklist (if POPI was added). |
| <b>Alternate courses:</b>                            | <b>ALT-Step 6:</b> The user does not want to continue with the generate due diligence checklist. <b>Use case terminates</b>      |  |
| <b>Conclusion:</b>                                   | The system generates a pdf file and can be downloaded onto the user's computer.  |  |
| <b>Post-condition:</b>                               | The user can download the recently opened pdf file and view the required due diligence checklist details of the relevant vendor. |  |
| <b>Business rules</b>                                | Only a user with the right role can have access to this.   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |

*Table 146: 6.10 Export Due Diligence Vendor Checklist*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                       |  |                      |
|-----------------------|--|----------------------|
| <b>Use case name:</b> | Generate Sole Supplier Review Notification | <b>USE CASE TYPE</b> |
|-----------------------|--|----------------------|

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case id:</b>                   | 6.11   | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | High   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | Time (PBA)   |  |
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The system must generate a Monthly performance review notification of each sole supplier on the system. The system will retrieve the following details:</p> <p><b>Vendor</b> table:</p> <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Vendor_Status_ID</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> </ul> <p><b>Sole_Supplier</b> table:</p> <ul style="list-style-type: none"> <li>• Sole_supplier_ID</li> <li>• Vendor_ID</li> <li>• MD_Aproval</li> <li>• Date</li> <li>• Reason</li> </ul> |  |
| <b>Pre-condition:</b>                 | The system must contain a sole supplier  |  |
| <b>Trigger:</b>                       | The system wants to generate a monthly performance review notification of each sole supplier on the system.  |  |
|                                       | <b>Actor Action</b>  | <b>System Response</b>                               |

|                                  |  |
|----------------------------------|--|
| <b>Typical course Of events:</b> | <p><b>Step 1:</b> The system responds by retrieving the following details from the:</p> <p><b>Vendor</b> table:</p> <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Vendor_Status_ID</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> </ul> <p><b>Sole_Supplier</b> table:</p> <ul style="list-style-type: none"> <li>• Sole_supplier_ID</li> <li>• Vendor_ID</li> <li>• MD_Aproval</li> <li>• Date</li> <li>• Reason</li> </ul> <p>[ALT]</p> |
|                                  | <p><b>Step 2:</b> The system then responds by filtering the vendor details to only those that are sole supplier and adding the following details to as attributes to the <b>NOTIFICATION</b> table:</p> <ul style="list-style-type: none"> <li>• User_ID (default set to GRC Officer's userID)</li> <li>• Name</li> <li>• Send_Date (date set to beginning of the month)</li> </ul>  |
|                                  | <p><b>Step 3:</b> The system will then display the new notification to the GRC Officer</p>   |
| <b>Alternate courses:</b>        | <p><b>ALT-Step 1:</b> If the system can't find any sole suppliers. <b>Use case terminates.</b></p>   |
| <b>Conclusion:</b>               | <p>The new notifications are added, and displayed for the GRC Officer</p>  |
| <b>Post-condition:</b>           | <p>The Sole Supplier Review Notification has been generated on the system.</p>   |

|  |  |
|--|--|
| <b>Business rules</b>                                | Vendors must be a sole supplier                          |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |

*Table 147: 6.11 Generate Sole Supplier Review Notification*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/22 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Generate BEE Certificate Expiry Notification | <b>USE CASE TYPE</b>                                 |
| <b>Use case id:</b>                   | 6.12   | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | high   | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | Time (PBA)                                   |  |
| <b>Primary system actor:</b>          | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |

|                                  |  |   |
|----------------------------------|--|---|
| <b>Description:</b>              | The GRC Officer has added an approved vendor with BEE details. The system will respond by setting a date to generate 3 weeks before, 1 week before and the day of the expiry of the BEE details. It will then display a notification to the GRC Officer. |   |
| <b>Pre-condition:</b>            | The GRC Officer must have accepted and filled in the details of an approved vendor.  |   |
| <b>Trigger:</b>                  | The system has reached 3 weeks before, 1 week before or the day of the expiry of the BEE details.  |   |
| <b>Typical course Of events:</b> | <b>Actor Action</b>  | <b>System Response</b>  |
|                                  |  | <p><b>Step 1:</b> The system retrieves the following details from the:</p> <p><b>VENDOR_BEE</b> table:</p> <ul style="list-style-type: none"> <li>• BEE_ID</li> <li>• Vendor_Detail_ID</li> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> </ul>   |
|                                  |  | <p><b>Step 2:</b> The system responds by taking the date value and calculating a year from then for 3 weeks before, 1 week before and the day of its expiry.</p> <p>[ALT]</p>   |
|                                  |  | <p><b>Step 3:</b> The system then saves it into the database in temporary tables which when the a specific date is reach will add the following details to the:</p> <p><b>NOTIFICATION</b> table:</p> <ul style="list-style-type: none"> <li>• User_ID (default set to GRC Officer's userID)</li> <li>• Name</li> <li>• Send_Date (date set to 3 weeks before, 1 week before or date of BEE details expiry date)</li> </ul> |

|  |   |   |
|--|---|---|
|  |   | <b>Step 4:</b> The system then display the BEE Certificate Expiry Notification. |
| <b>Alternate courses:</b>                            | <b>ALT-Step 2:</b> Vendor Details were updated. <b>Return to step 1</b>   |   |
| <b>Conclusion:</b>                                   | The newly created notifications details will be added to the system and is displayed.   |   |
| <b>Post-condition:</b>                               | A new record is created in the <b>NOTIFICATION</b> table: <ul style="list-style-type: none"> <li>• User_ID (default set to GRC Officer's userID)</li> <li>• Name</li> <li>• Send_Date (date set to 3 weeks before, 1 week before or date of BEE details expiry date)</li> </ul> |   |
| <b>Business rules</b>                                | None  |   |
| <b>Implementation constraints and specifications</b> | None  |   |
| <b>Assumptions:</b>                                  | None  |   |
| <b>Open issues:</b>                                  | None  |   |

*Table 148: 6.12 Generate BEE Certificate Expiry Notification*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/22 | Version: 1.0 |

|                       |                                |                      |
|-----------------------|--------------------------------|----------------------|
| <b>Use case name:</b> | Update Approve onboard request | <b>USE CASE TYPE</b> |
|-----------------------|--------------------------------|----------------------|

|                                       |   |   |
|---------------------------------------|---|---|
| <b>Use case id:</b>                   | 6.13  | Business Requirements: <input type="checkbox"/>   |
| <b>Priority:</b>                      | high  | System Analysis: <input checked="" type="checkbox"/>  |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>   |
| <b>Primary business actor:</b>        | GRC Officer (PBA)   |   |
| <b>Primary system actor:</b>          | None  |   |
| <b>Other participating actors:</b>    | None  |   |
| <b>Other interested stakeholders:</b> | None  |   |
| <b>Description:</b>                   | <p>In this use case the GRC Officer would like to Edit an approved vendor's:</p> <ul style="list-style-type: none"> <li>• Due Diligence Checklist</li> <li>• BEE details</li> <li>• POPI Details</li> </ul> <p>The requestor will then provide the updated details. The system then captures and validates the details. The approved vendor related details are updated in the <b>Vendor_BEE</b> table, <b>Due_Diligence</b> table, <b>POPI</b> table. Finally, a success notification is displayed to the GRC Officer that the request has been Updated.</p> |   |
| <b>Pre-condition:</b>                 | The requestor must be logged into the system.   |   |
| <b>Trigger:</b>                       | The GRC Officer requests to update approved onboard request vendor related details.   |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b><br><br><b>Step 1:</b> The user selects an approved onboard request they wish to update  | <b>System Response</b><br><br><b>Step 2:</b> The system retrieves the following details in the following tables:<br><b>Vendor:</b> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> </ul> |

- Email
- Vendor\_Status\_ID
- Number\_Of\_Times\_Used  
(default value set to 0)
- Sole\_Supplier\_Provided
- PreferredVendor

**Vendor\_BEE:**

- BEE\_ID (PK)
- Vendor\_ID (FK)
- BEE\_Level
- BEE\_Certificate
- Date

**DUE\_DILIGENCE:**

- Due\_Diligence\_ID (PK)
- Vendor\_ID (FK)
- Due\_Diligence\_Doc
- Mutual\_Nda\_Signed
- Basic\_Company\_Info\_Provided
- Group\_Structure\_Provided
- Income\_Tax\_Number\_Provided
- Tax\_Clearance\_Certificate\_Provided
- Vat\_Number\_Provided
- Vat\_Reg\_Certificate\_Provided
- Company\_Reg\_Doc\_Provided
- Letter\_Of\_Good\_Standing\_Provided
- B\_BBEE\_Certificate\_Provided
- Direcor\_Details\_Provided
- Company\_Resolution\_Agreement\_Provided
- Subcontractor\_Name\_Provided
- Company\_Details\_Provided
- Individual\_Details\_Provided
- General\_Liability\_Insurance\_Present
- Cyber\_Insurance\_Present
- Proffesional\_Indemnity\_Insurance\_Present

- Other\_Insurance\_Required
- Licenses\_Required
- Accreditation\_Required
- Professional\_Membership\_Required
- Business\_Continuity\_Present
- Disaster\_Recovery\_Plan\_Present
- POPI\_Present
- Data\_Security\_Breaches\_Present
- Site\_Visits\_Present
- Information\_Security\_Policy\_Present
- Privacy\_Policy\_Present\_Present
- Data\_Retention\_Destruction\_Policy\_Present
- Anti\_Bribery\_Corruption\_Policy\_Present
- Ethics\_Policy\_Present
- Conflict\_Of\_Interest\_Policy\_Present
- Customer\_Complaints\_Policy\_Present
- Business\_References\_Present

#### POPI:

- POPI\_ID (PK)
- Contracted\_Partner\_Type\_ID (FK)
- Due\_Diligence\_ID (FK)
- Personal\_Data\_Purpose
- DataProcessing\_JointController\_Agreement
- Confidentiality\_Importance\_Highlighted
- Contract\_Audits\_Provisions\_Provided
- Activity\_Liability\_Present
- Third\_Party\_Data\_Processing\_Provisioned
- Contract\_End\_Data\_Management\_Provided
- Personal\_Data\_Processing\_Details\_Present
- Processing\_Activities\_Certification\_Held

|  |  |
|--|--|
|  | <p><b>Insurance:</b></p> <ul style="list-style-type: none"> <li>• General_Liability_Insurance_Present_Document</li> <li>• Cyber_Insurance_Present_Document</li> <li>• Professional_Indemnity_Insurance_Present_Document</li> <li>• Other_Insurance_Required_Document</li> </ul> <p>The Vendor_ID is linked to the associative entities:</p> <ul style="list-style-type: none"> <li>• <b>Vendor_BEE</b></li> <li>• <b>DUE_DILIGENCE</b></li> <li>• <b>POPI</b></li> <li>• <b>Insurance</b></li> </ul> <p>where their related primary keys:</p> <ul style="list-style-type: none"> <li>• BEE_ID</li> <li>• Due_Diligence_ID</li> <li>• POPI_ID</li> <li>• InsuranceID</li> </ul> <p>is auto incremented by 1.</p> <p>And prompts the user to provide updated details</p> |
|  | <p><b>Step 3:</b> The user updates the details that they wish to change.</p>   |
|  | <p><b>Step 4:</b> The system successfully validates the details provided by the requestor by checking that all the required details are correctly entered. <b>[ALT]</b></p>  |
|  | <p><b>Step 5:</b> The system request that the requestor confirms the changes.</p>  |
|  | <p><b>Step 6:</b> The requestor confirms that the request can be updated.<br/><b>[ALT]</b></p>   |
|  | <p><b>Step 7:</b> The system validates and updates the following details by storing it as attributes in the following tables:</p>  |

|  |   |
|--|---|
|  | <p><b>Vendor_BEE:</b></p> <ul style="list-style-type: none"> <li>• BEE_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> </ul> <p><b>DUE_DILIGENCE:</b></p> <ul style="list-style-type: none"> <li>• Due_Diligence_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Direcor_Details_Provided</li> <li>• Company_Resolution_Agreement_Provided</li> <li>• Subcontractor_Name_Provided</li> <li>• Company_Details_Provided</li> <li>• Individual_Details_Provided</li> <li>• General_Liability_Insurance_Present</li> <li>• Cyber_Insurance_Present</li> <li>• Proffesional_Indemnity_Insurance_Present</li> <li>• Other_Insurance_Required</li> <li>• Licenses_Required</li> <li>• Accreditation_Required</li> <li>• Proffesional_Membership_Required</li> </ul> |
|--|---|

- Business\_Continuity\_Present
- Disaster\_Recovery\_Plan\_Present
- POPI\_Present
- Data\_Security\_Breaches\_Present
- Site\_Visits\_Present
- Information\_Security\_Policy\_Present
- Privacy\_Policy\_Present\_Present
- Data\_Retention\_Destruction\_Policy\_Present
- Anti\_Bribery\_Corruption\_Policy\_Present
- Ethics\_Policy\_Present
- Conflict\_Of\_Interest\_Policy\_Present
- Customer\_Complaints\_Policy\_Present
- Business\_References\_Present

#### **POPI:**

- POPI\_ID (PK)
- Contracted\_Partner\_Type\_ID (FK)
- Due\_Diligence\_ID (FK)
- Personal\_Data\_Purpose
- DataProcessing\_JointController\_Agreement
- Confidentiality\_Importance\_Highlighted
- Contract\_Audits\_Provisions\_Provided
- Activity\_Liability\_Present
- Third\_Party\_Data\_Processing\_Provisioned
- Contract\_End\_Data\_Management\_Provided
- Personal\_Data\_Processing\_Details\_Present
- Processing\_Activities\_Certification\_Held

#### **Insurance:**

- General\_Liability\_Insurance\_Present\_Document
- Cyber\_Insurance\_Present\_Document

|                                       |  |
|---------------------------------------|--|
|                                       | <ul style="list-style-type: none"> <li>• Professional_Indemnity_Insurance_Present_Document</li> <li>• Other_Insurance_Required_Document</li> </ul> <p>The Vendor_ID is linked to the associative entities:</p> <ul style="list-style-type: none"> <li>• <b>Vendor_BEE</b></li> <li>• <b>DUE_DILIGENCE</b></li> <li>• <b>POPI</b></li> <li>• <b>Insurance</b></li> </ul> <p>where their related primary keys:</p> <ul style="list-style-type: none"> <li>• BEE_ID</li> <li>• Due_Diligence_ID</li> <li>• POPI_ID</li> <li>• InsuranceID</li> </ul> <p>is auto incremented by 1.</p> |
|                                       | <p><b>Step 8:</b> The system notifies the user that the request has been successfully updated. <b>[ALT]</b></p>  |
| <b>Alternate courses:</b>             | <p><b>ALT-Step 4:</b> The entered information is not correct. The system responds by notifying the user what information is possibly incorrect or not filled in. <b>Return to step 3</b></p> <p><b>ALT-step-6:</b> The user cancels changes. <b>Use case terminates.</b></p> <p><b>ALT-step-8:</b> The system did not successfully update the onboard request. The system notifies the requestor that there was an error with the updating of the request. <b>Use case terminates</b></p>  |
| <b>Conclusion:</b>                    | The details will be updated on the system and a success notification is displayed.   |
| <b>Post-condition:</b>                | The details have been updated in the <b>Vendor_BEE</b> table, <b>DUE_DILIGENCE</b> table and <b>Insurance</b> table, POPI table.   |
| <b>Business rules</b>                 | Only the GRC Officer can update the request the system.  |
| <b>Implementation constraints and</b> | None   |

|                       |        |
|-----------------------|--------|
| <b>specifications</b> |        |
| <b>Assumptions:</b>   | • None |
| <b>Open issues:</b>   | • None |

Table 149: 6.13 Update Approve Onboard Request

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/24 | Version: 1.0 |

| <b>Use case name:</b>                 | Search Approved Or Pending onboard request.   | <b>USE CASE TYPE</b>                                 |
|---------------------------------------|---|--|
| <b>Use case id:</b>                   | 6.14  | Business Requirements: <input type="checkbox"/>      |
| <b>Priority:</b>                      | high  | System Analysis: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo  | System Design: <input type="checkbox"/>              |
| <b>Primary business actor:</b>        | User (PBA)  |  |
| <b>Primary system actor:</b>          | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | The use case describes the event where the user would like to view approved or pending onboard request. |  |

|   | <p>The use case begins when the user request to view the approved or pending onboard request. The system will load all onboard request details and filter it as required.</p> <p>Use case concludes when the user is able to view all the onboard request as needed.</p>  |                     |                        |   |  |
|---|---|---------------------|------------------------|---|--|
| <b>Pre-condition:</b>   | The user must be logged into the system.  |                     |                        |   |  |
| <b>Trigger:</b>   | The user request to view approved or pending onboard request  |                     |                        |   |  |
| <b>Typical course Of events:</b>  | <table border="1"> <thead> <tr> <th><b>Actor Action</b></th><th><b>System Response</b></th></tr> </thead> <tbody> <tr> <td> <p><b>Step 1:</b> The user request to view approved or pending onboard request.</p> </td><td> <p><b>Step 2:</b> The system responds by retrieving the following details:</p> <p><b>Vendor</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> <li>• PreferredVendor</li> </ul> <p><b>Onboard_Request</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Onboard_Request_ID (FK)</li> <li>• Vendor_ID</li> <li>• Quotes</li> <li>• User_ID</li> <li>• Status_ID</li> </ul> <p><b>Vendor</b> table is linked to the <b>Onboard_Request</b> table via the Vendor_ID attribute.</p> <p>The following details are then displayed:</p> </td></tr> </tbody> </table> | <b>Actor Action</b> | <b>System Response</b> | <p><b>Step 1:</b> The user request to view approved or pending onboard request.</p> | <p><b>Step 2:</b> The system responds by retrieving the following details:</p> <p><b>Vendor</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> <li>• PreferredVendor</li> </ul> <p><b>Onboard_Request</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Onboard_Request_ID (FK)</li> <li>• Vendor_ID</li> <li>• Quotes</li> <li>• User_ID</li> <li>• Status_ID</li> </ul> <p><b>Vendor</b> table is linked to the <b>Onboard_Request</b> table via the Vendor_ID attribute.</p> <p>The following details are then displayed:</p> |
| <b>Actor Action</b>   | <b>System Response</b>  |                     |                        |   |  |
| <p><b>Step 1:</b> The user request to view approved or pending onboard request.</p> | <p><b>Step 2:</b> The system responds by retrieving the following details:</p> <p><b>Vendor</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Vendor_ID (PK)</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplierProvided</li> <li>• PreferredVendor</li> </ul> <p><b>Onboard_Request</b> table containing the following attributes:</p> <ul style="list-style-type: none"> <li>• Onboard_Request_ID (FK)</li> <li>• Vendor_ID</li> <li>• Quotes</li> <li>• User_ID</li> <li>• Status_ID</li> </ul> <p><b>Vendor</b> table is linked to the <b>Onboard_Request</b> table via the Vendor_ID attribute.</p> <p>The following details are then displayed:</p>  |                     |                        |   |  |

|  |  |   |
|--|--|---|
|  |  | <ul style="list-style-type: none"> <li>• Name</li> <li>• Email</li> <li>• Quotes</li> <li>• Preferred Vendor</li> </ul> <p>The system prompts the user to either filter displayed data or to provide selection details.</p> <ul style="list-style-type: none"> <li>•</li> </ul> |
|  | <b>Step 3:</b> The user provides required details. | <b>Step 4:</b> The system responds by filtering displayed data or by displaying the correct screen  |
| <b>Alternate courses:</b>                            |  |   |
| <b>Conclusion:</b>                                   |  | The system displays a new screen or the same screen that has been filtered.   |
| <b>Post-condition:</b>                               |  | The user has an updated screen.   |
| <b>Business rules</b>                                |  | None  |
| <b>Implementation constraints and specifications</b> |  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  |  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  |  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

*Table 150: 6.14 Search Approved Or Pending Onboard Request*

## 2.3 CONCLUSION

This concludes the Logical Use case Narratives section of the Procion system.

## 3. LOGICAL CONTEXT DIAGRAM

### 3.1 INTRODUCTION

In this section we will display the visual representation of the logical Context diagrams for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system.

### 3.2 LOGICAL CONTEXT DIAGRAMS

#### USE CASE 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35–2.36

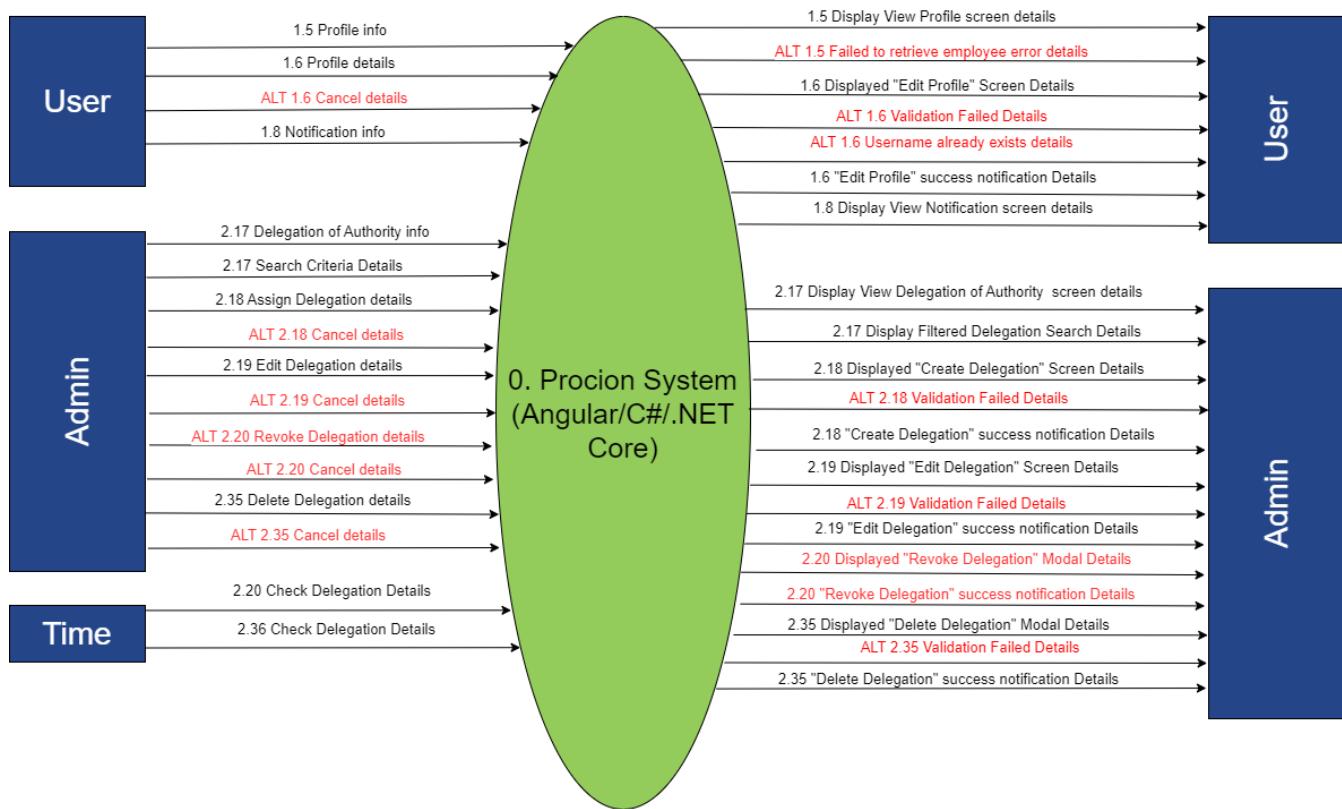


Figure 11: Use Case 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35–2.36Logical Context

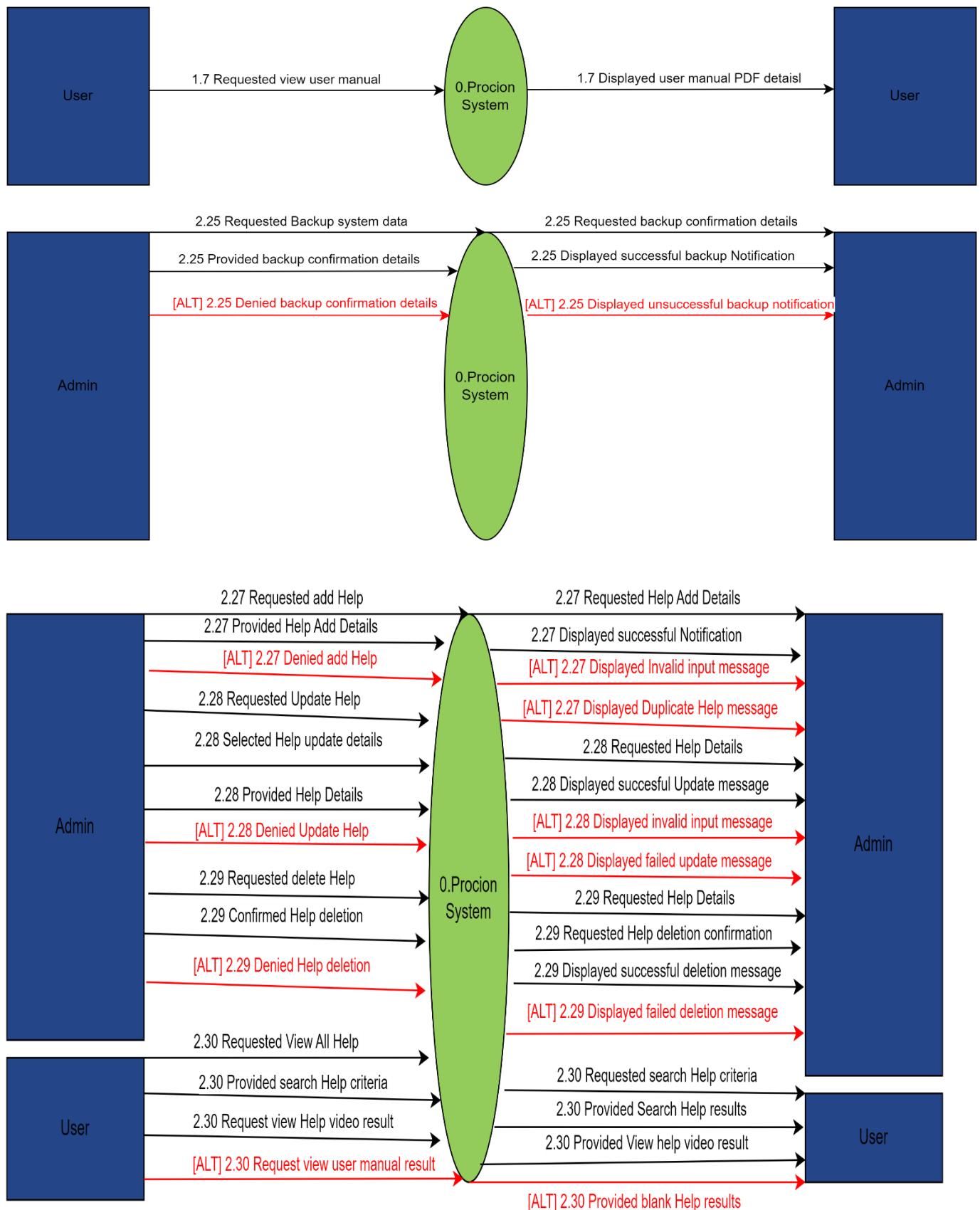
**USE CASE 1.7-1.7 & 2.25 & 2.27-2.30**


Figure 12: Use Case 1.7-1.8 &amp; 2.25-2.30 Logical Context



### USE CASE 3.8-4.1 & 4.3-4.10

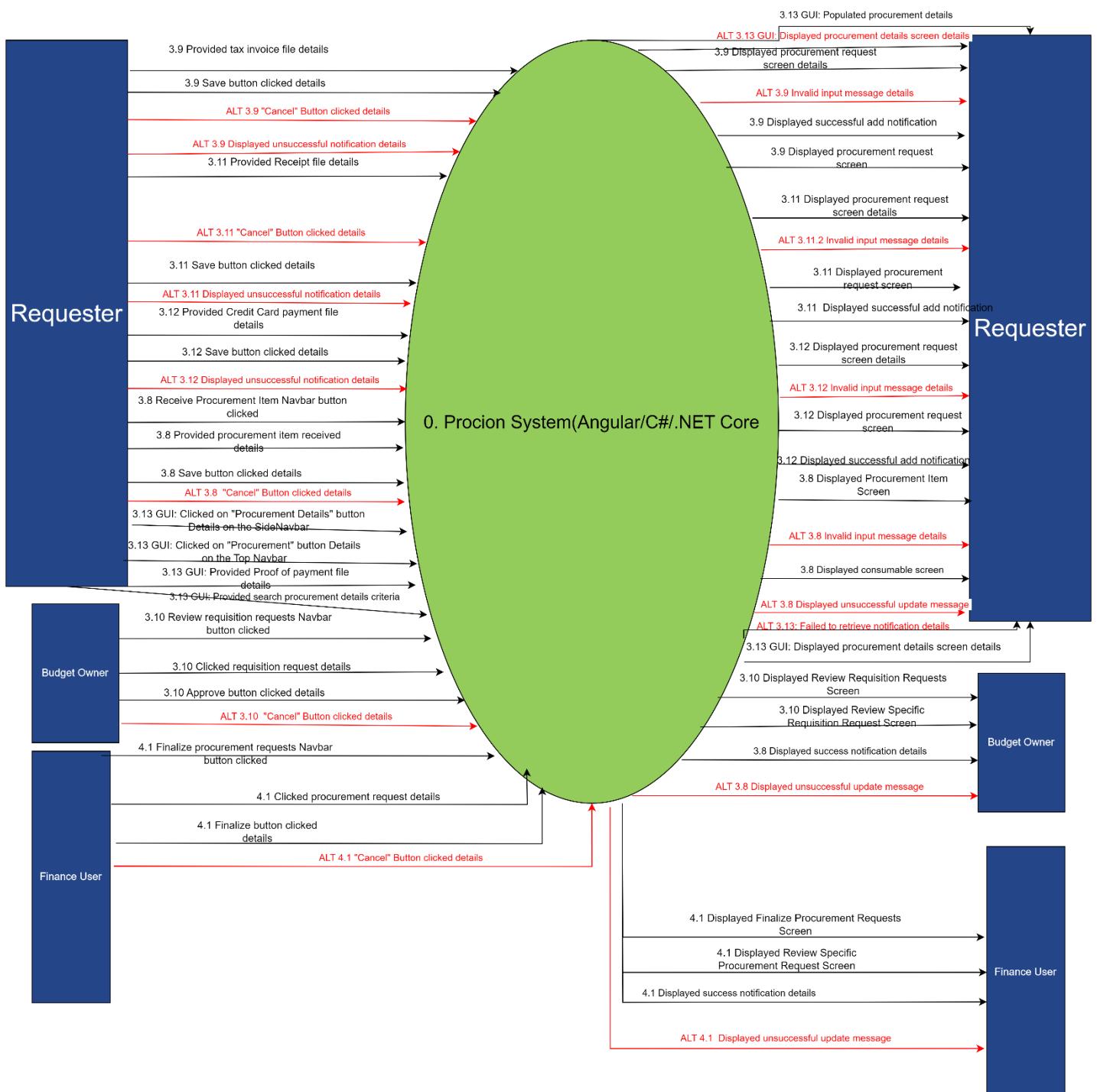


Figure 13: Use Case 3.8-4.2 Logical Context

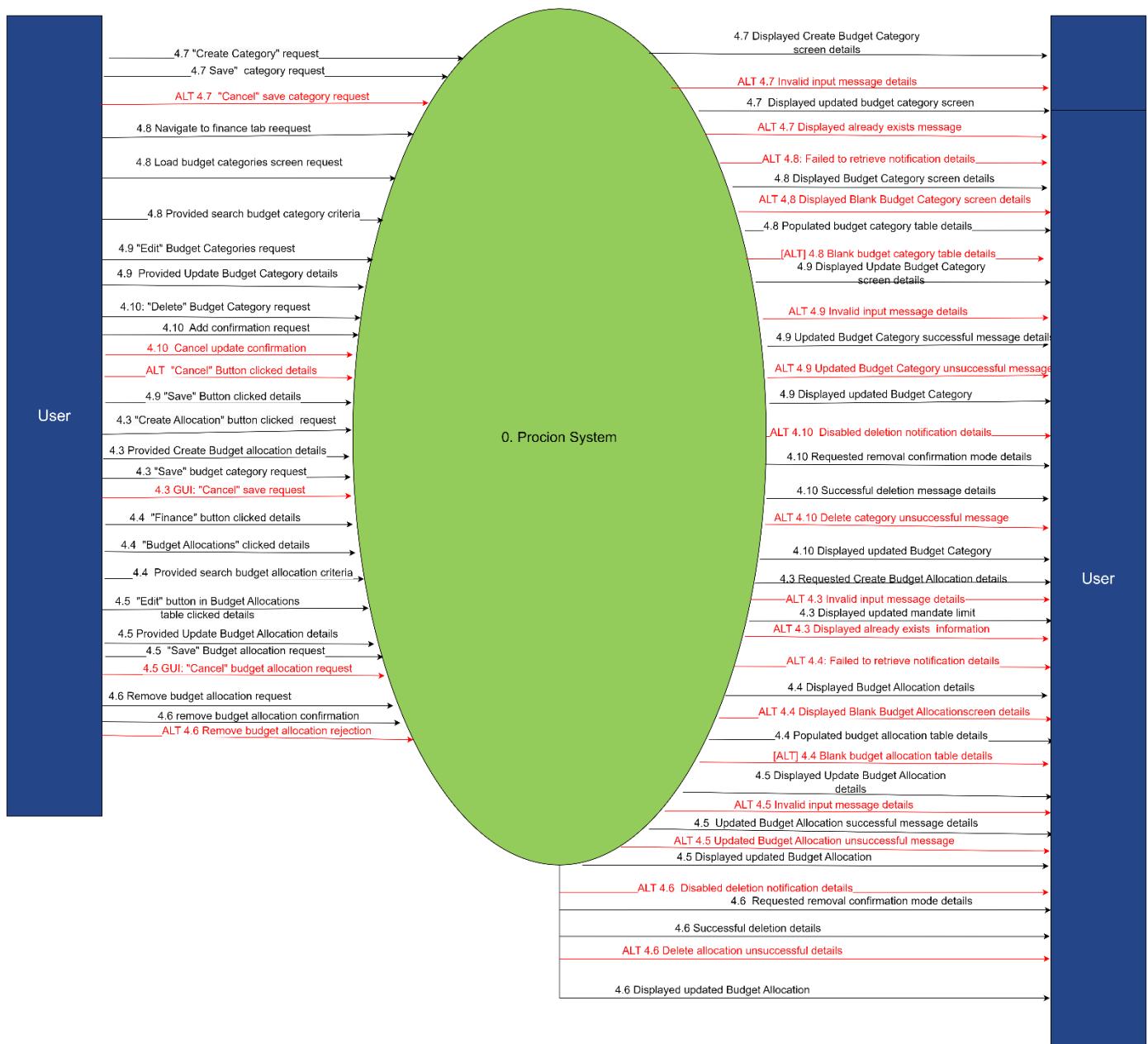
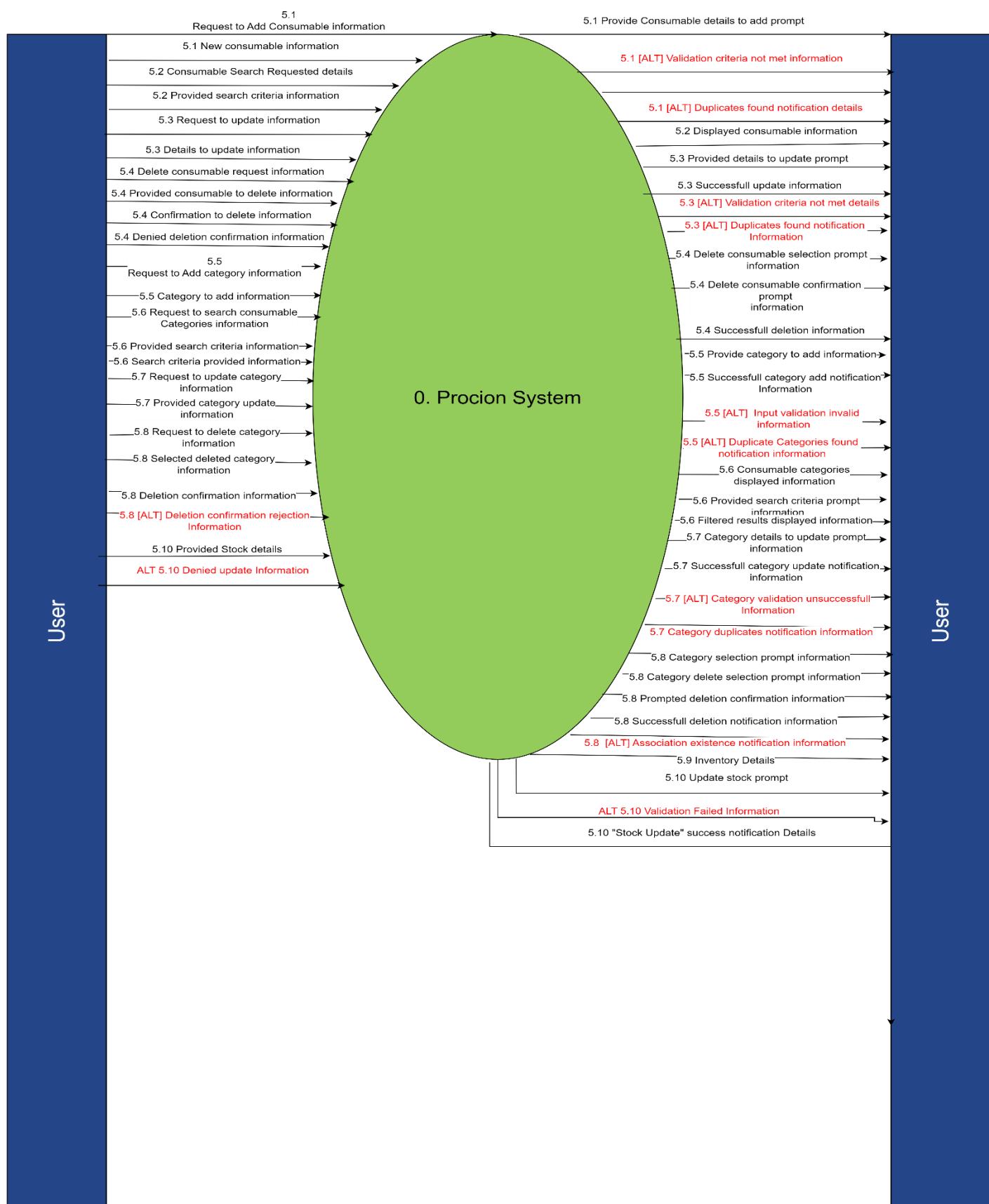


Figure 14: Use Case 4.3-4.10 Logical Context

**USE CASE 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4**

**Figure 15: Use Case 5.1-5.8 & 5.9, 5.10**

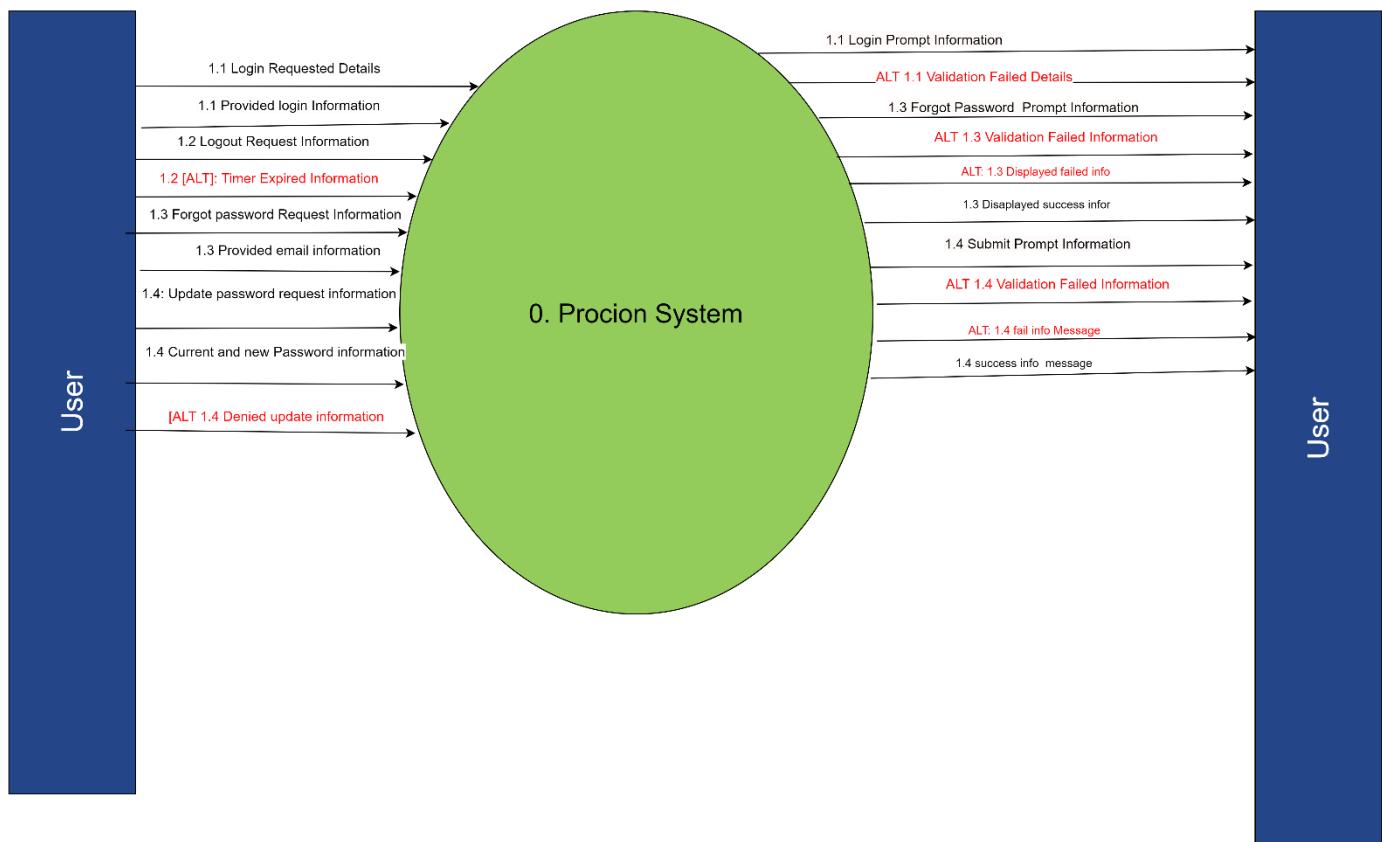


Figure 16: Use Case 1.1-1.4 Logical Context

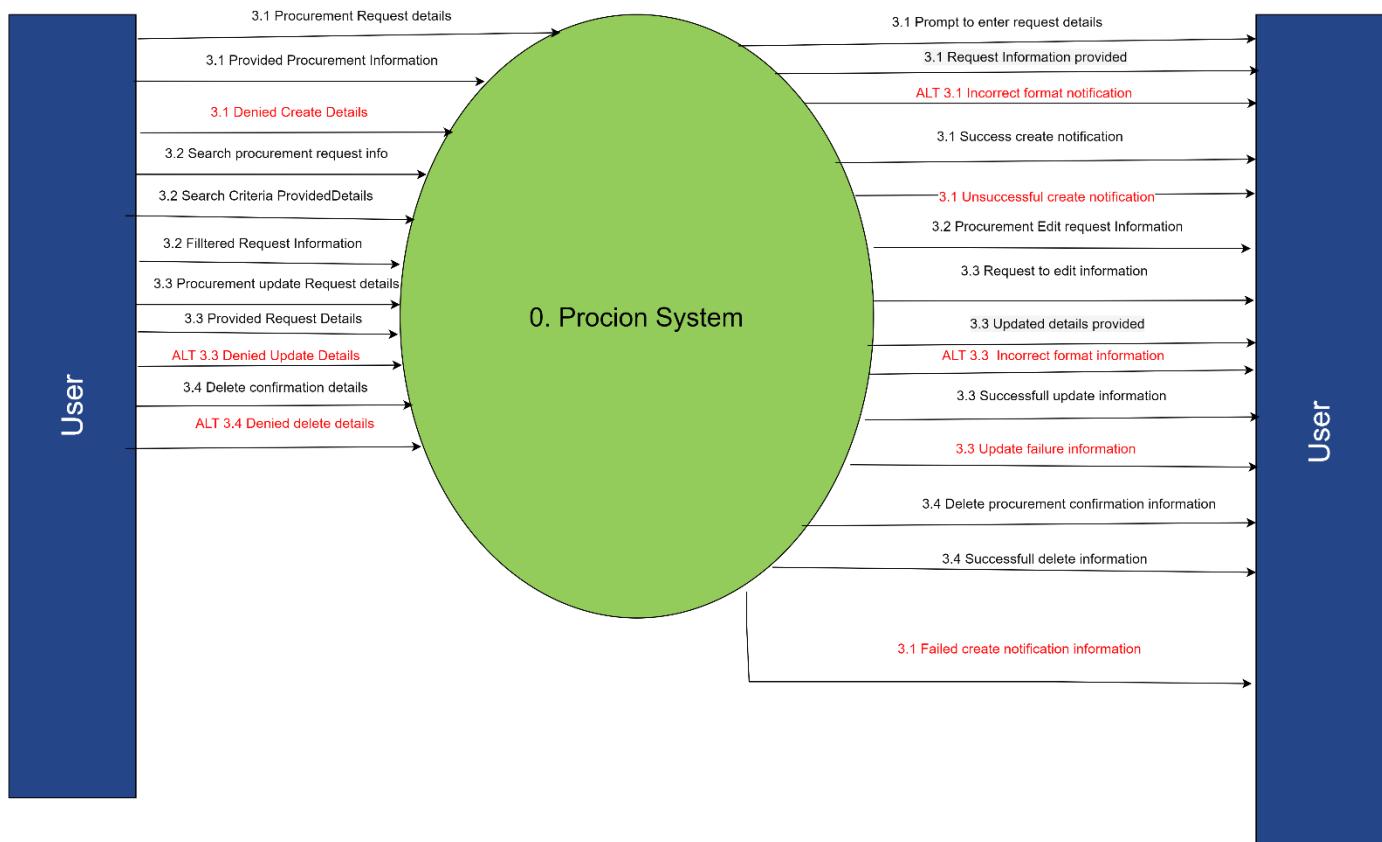


Figure 17: Use Case 3.1-3.4 Logical Context

## USE CASE 3.5-3.7 & 6.5 & 6.10-6.14

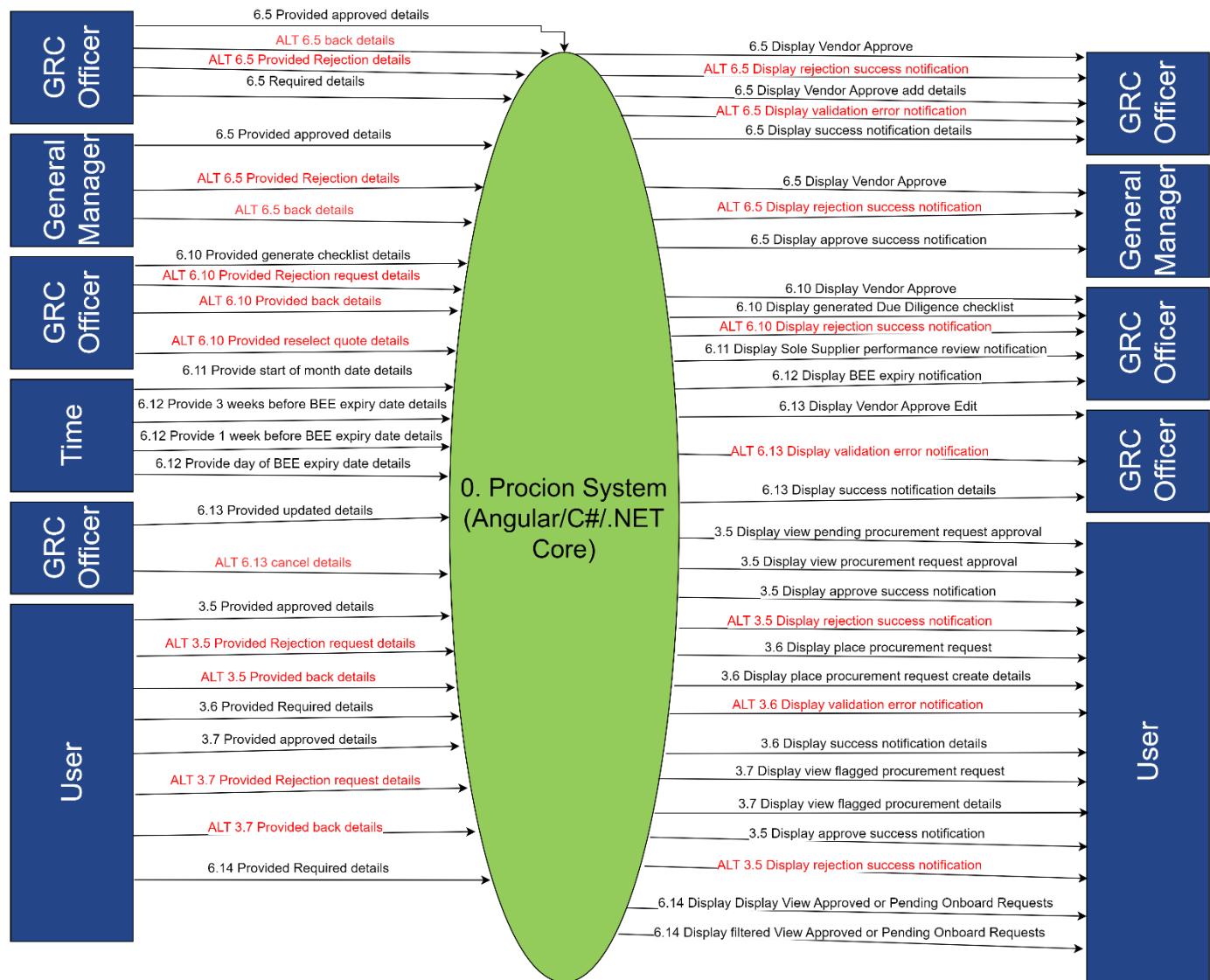


Table 151: Use case 3.5-3.7, 6.5, 6.10-6.14 Logical Context

### 3.3 CONCLUSION

This Concludes the visual representation of the Logical Context diagrams for the Procion system.



## 4. SCREEN DESIGN

### 4.1 INTRODUCTION

This section contains the design principles for the example input and output designs including Cruds from the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) the navigation bar to depict the principles that will be used for all the screens in the Procion System.

### 4.2 DESIGN PRINCIPLES & EXAMPLE WIREFRAMES

#### USE CASE 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35

##### 1.5 View Profile (Non-Admin Role)

The screenshot shows a user profile page for 'Werner Schutte'. At the top, there is a navigation bar with the MOYO logo, a home icon, and links for Administration, Inventory, Finance, Procurement, Vendor, Reports, a bell icon, and a user profile icon. Below the navigation bar is a large profile picture of a man with a surprised expression. Below the picture, the name 'Werner Schutte' and title 'Managing Director' are displayed. There are two buttons: 'Edit' and 'Update Password'. A blue box contains the following data:

|                |                         |
|----------------|-------------------------|
| Username:      | WernerSchutte574        |
| Email:         | emilwonigkeit@gmail.com |
| Phone Number:  | 094 574 1854            |
| Mandate Limit: | 10000                   |
| Department:    | Business Enablement     |
| Branch:        | Pretoria                |

Figure 18: View Profile

### 1.5 View Profile (Admin Role)

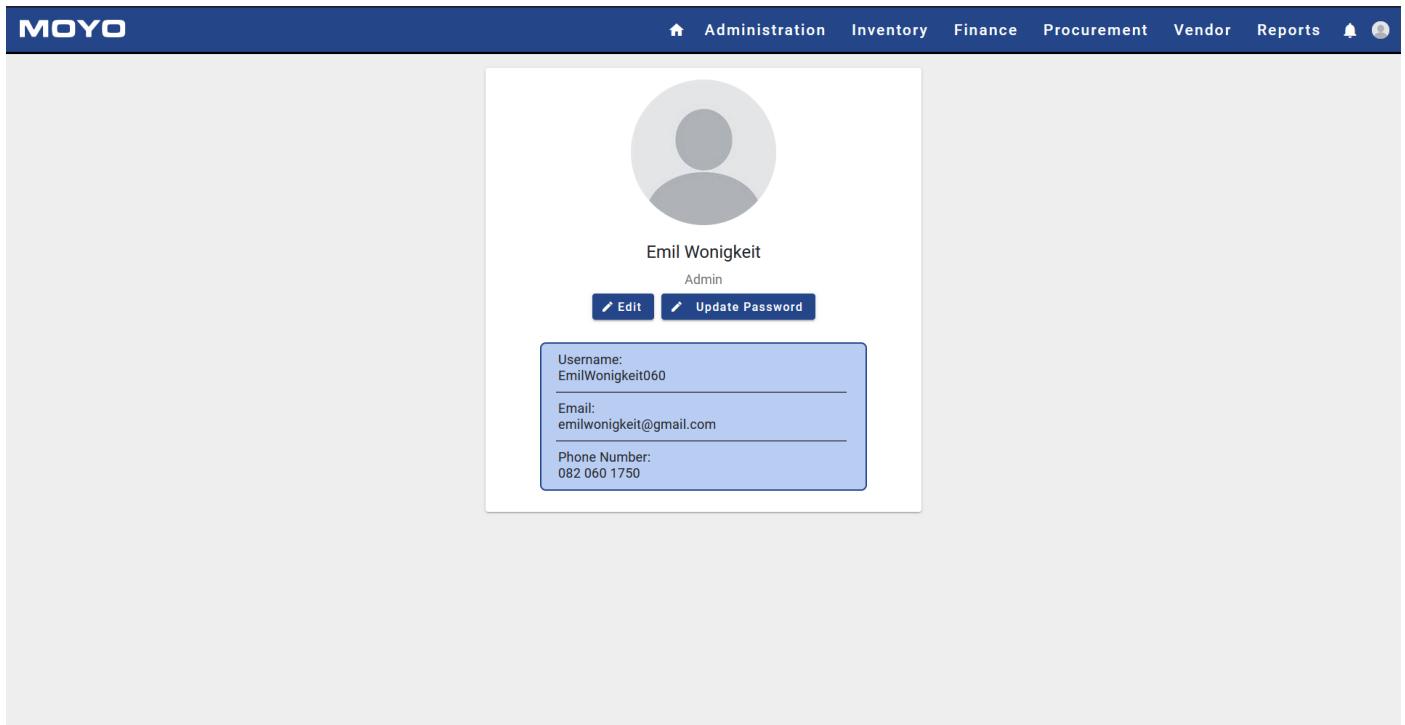


Figure 19: View Profile (Admin)

### 1.6 Edit Profile

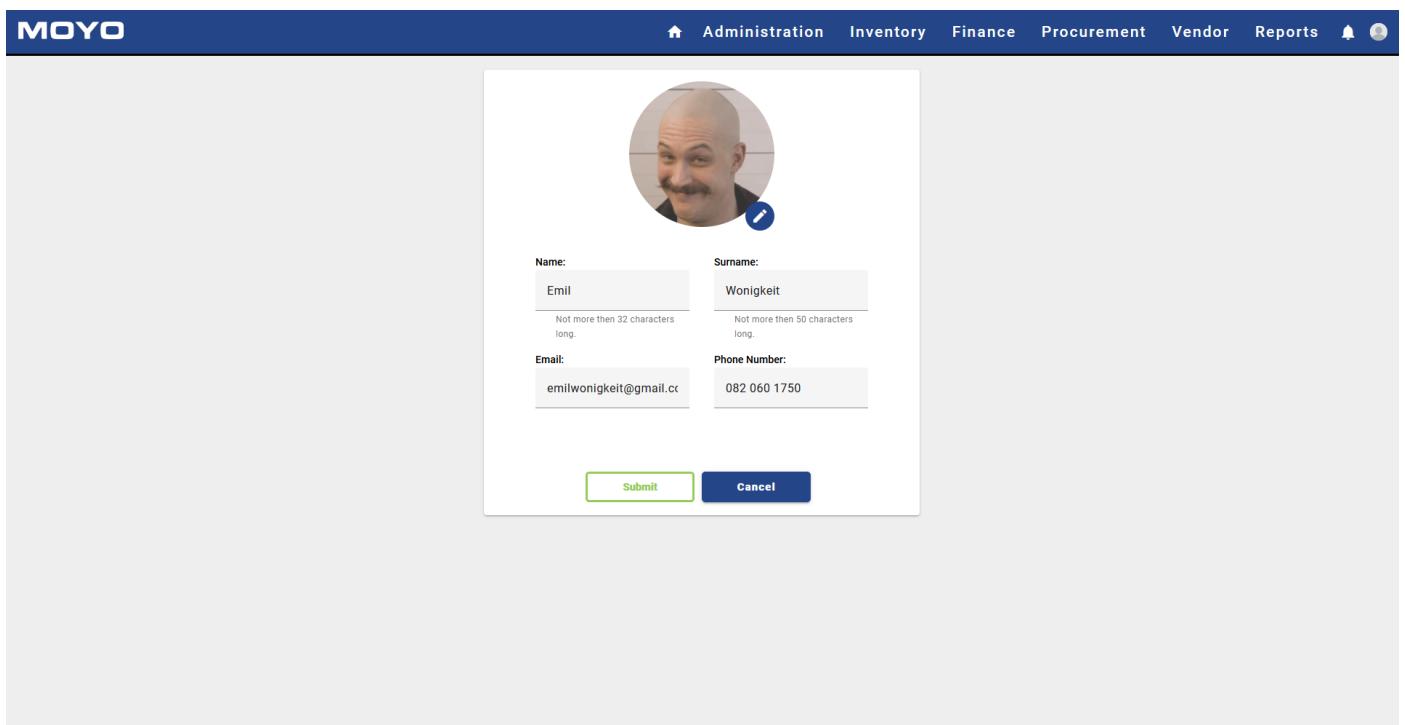


Figure 20: Edit Profile

## Cropper Module

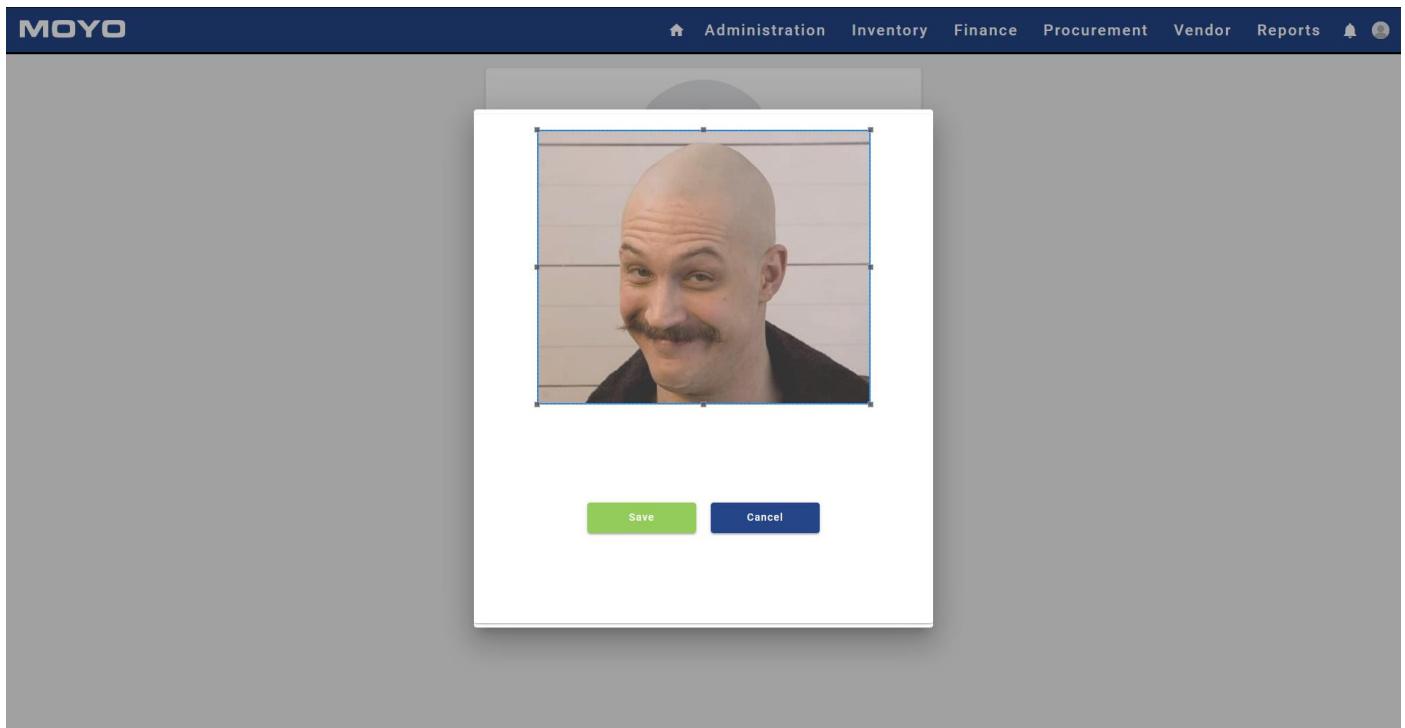


Figure 21: Cropper Module

## Update Success Message

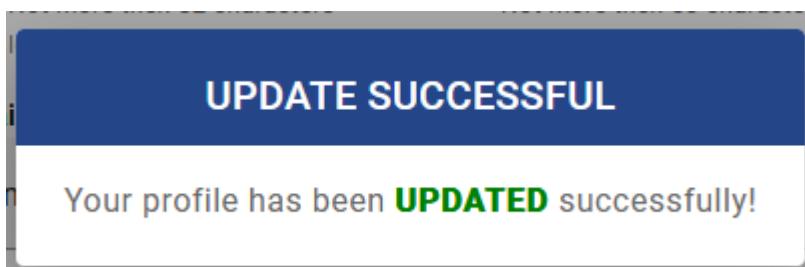


Figure 22: Update Success Message

## User Exists Error Notification



Figure 23: User Exists Error Notification

### 1.8 View Notification (No Temp Access)

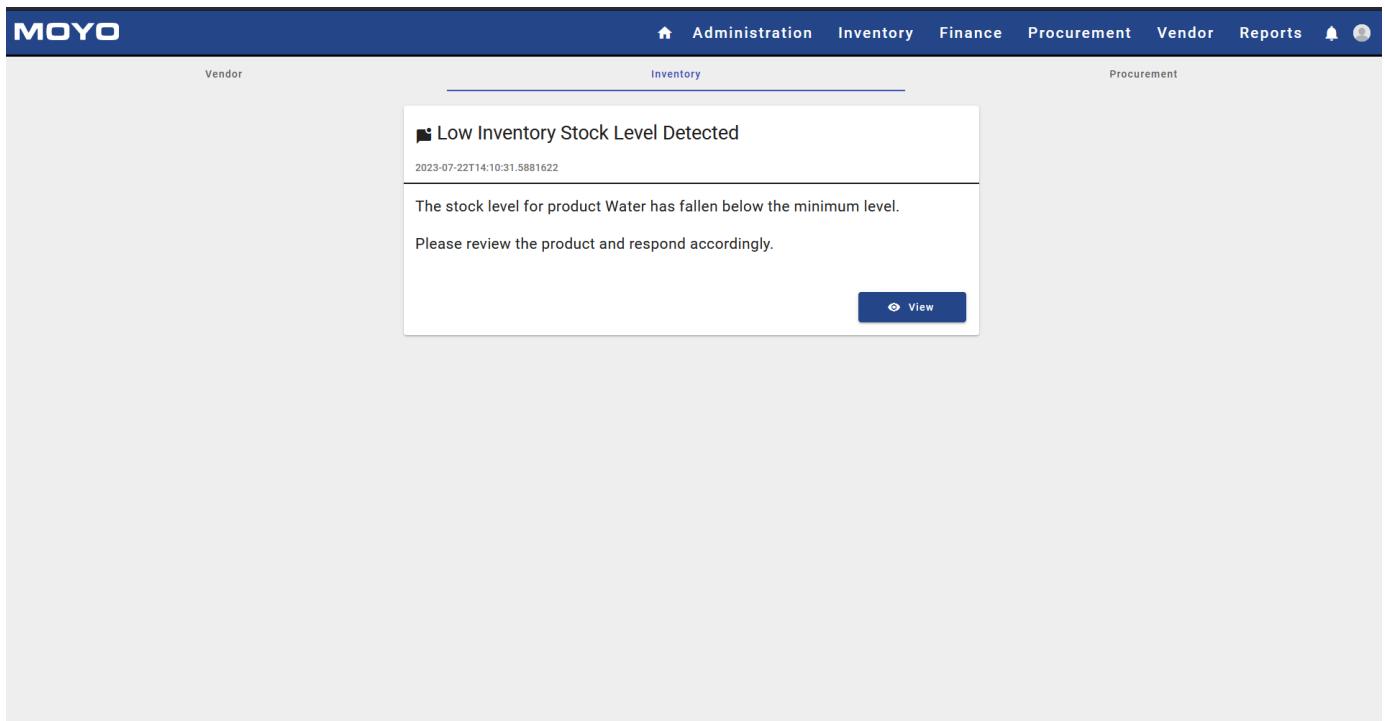


Figure 24: View Notification (No Temp Access)

### 1.8 View Notification (With Temp Access)

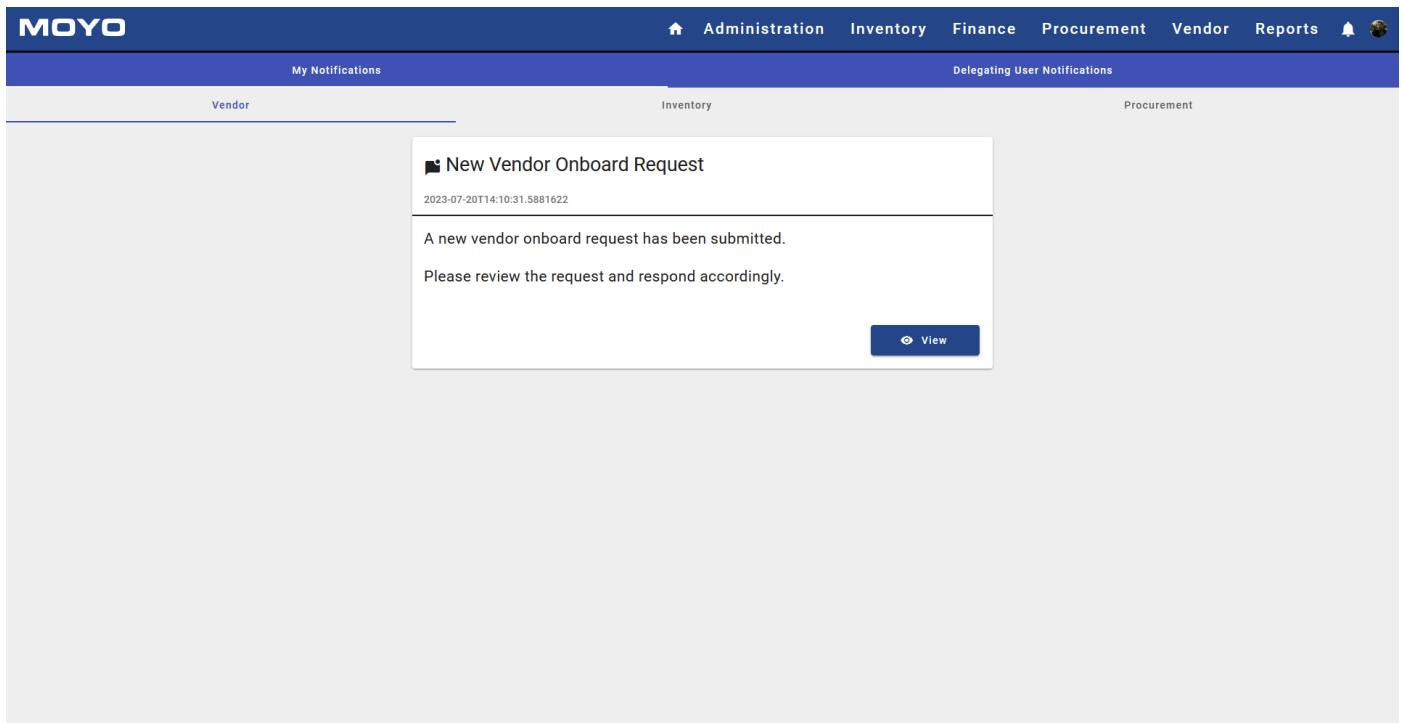
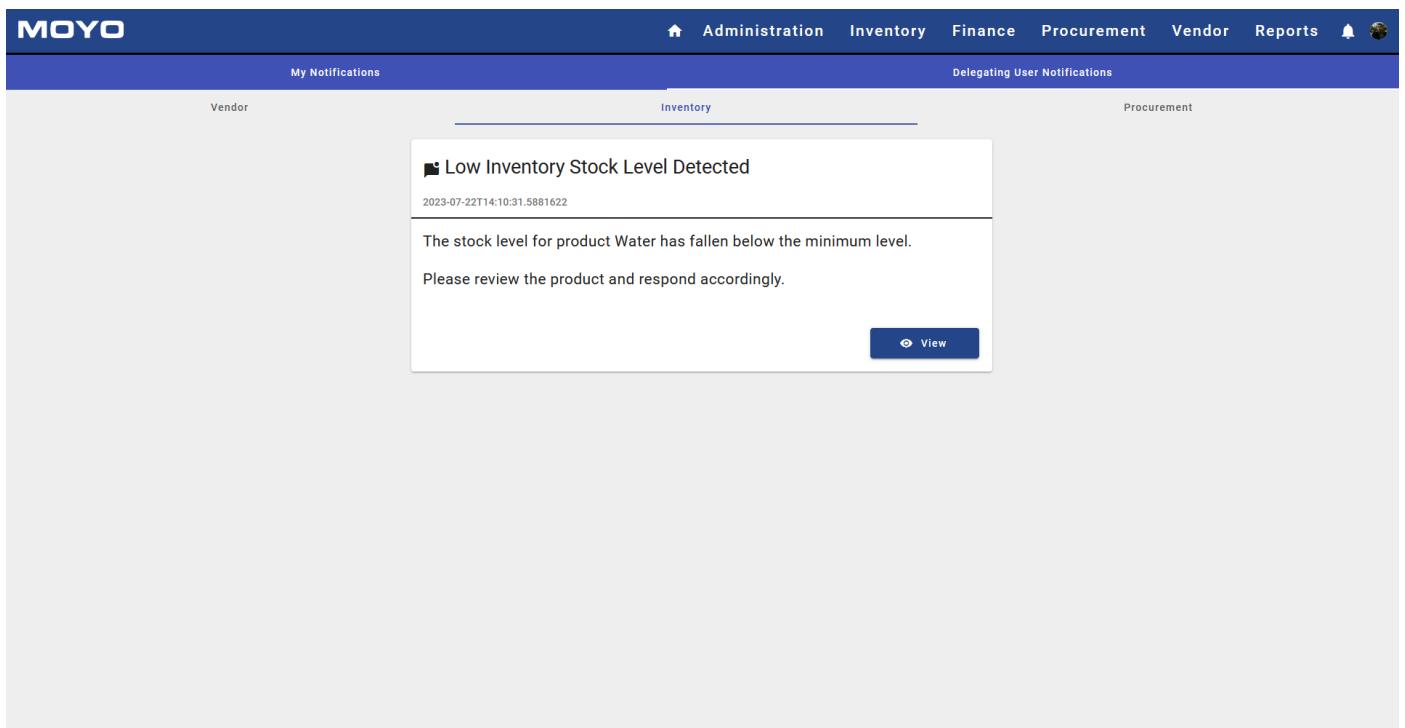


Figure 25: View Notification



### 2.17 View Delegation of Authority

The screenshot shows a table titled "Manage Delegation of Authority" with the following data:

| No. | Delegating Party    | Delegate            | Start Date | End Date   | Delegation Form                | Status   | Edit                  | Delete                  | Revoke Access                  |
|-----|---------------------|---------------------|------------|------------|--------------------------------|----------|-----------------------|-------------------------|--------------------------------|
| 1   | WernerSchutte574    | JasonvanderMerwe420 | 25/07/2023 | 31/07/2023 | <a href="#">Appendix B.pdf</a> | Inactive | <button>Edit</button> | <button>Delete</button> | <button>Revoke Access</button> |
| 2   | JasonvanderMerwe420 | EmilWoniigkeit060   | 20/07/2023 | 23/07/2023 | <a href="#">Appendix B.pdf</a> | Revoked  | <button>Edit</button> | <button>Delete</button> | <button>Revoke Access</button> |
| 3   | LeonCombrinck691    | WernerSchutte574    | 19/07/2023 | 24/07/2023 | <a href="#">Appendix A.pdf</a> | Active   | <button>Edit</button> | <button>Delete</button> | <button>Revoke Access</button> |

Figure 26: View Delegation of Authority

### 2.18 Assign Delegation of Authority

**Assign Delegation**

Delegating Party: WernerSchutte574

Delegate: JasonvanderMerwe420  
Not more than 50 characters long.

Date Range: 25/07/2023 – 31/07/2023

Delegation Document: Appendix B.pdf

Save Cancel

Figure 27: Assign Delegation of Authority

**Assign Delegation**

Delegating Party: WernerSchutte574

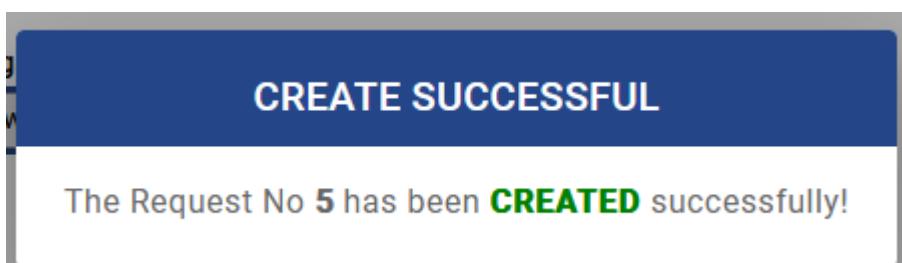
Delegate: JasonvanderMerwe420  
Not more than 50 characters long.

Date Range: 25/07/2023 – 31/07/2023

Delegation Document: Appendix B.pdf

Save Cancel

### Create Success Notification

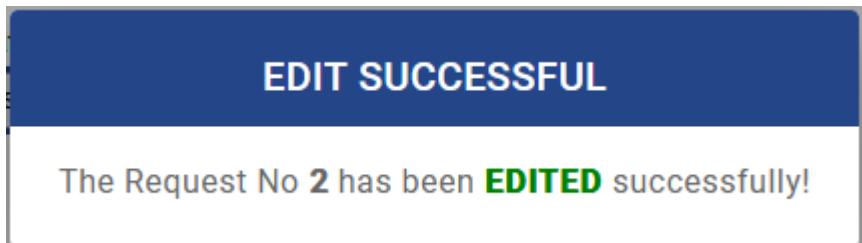


## 2.19 Edit Delegation of Authority

The screenshot shows a software application window titled "Edit Delegation". At the top, there is a navigation bar with the MOYO logo, a home icon, and menu items: Administration, Inventory, Finance, Procurement, Vendor, Reports, and a notifications icon. The main content area is titled "Edit Delegation". It contains several input fields: "Delegating Party" (WernerSchutte574), "Delegate" (JasonvanderMerwe420), "Date Range" (25/07/2023 – 31/07/2023), and "Delegation Document" (Browse... No file selected). At the bottom are two buttons: "Save" (highlighted in green) and "Cancel".

Figure 28: Edit Delegation of Authority

## Edit Success Notification



## 2.20 Revoke Access

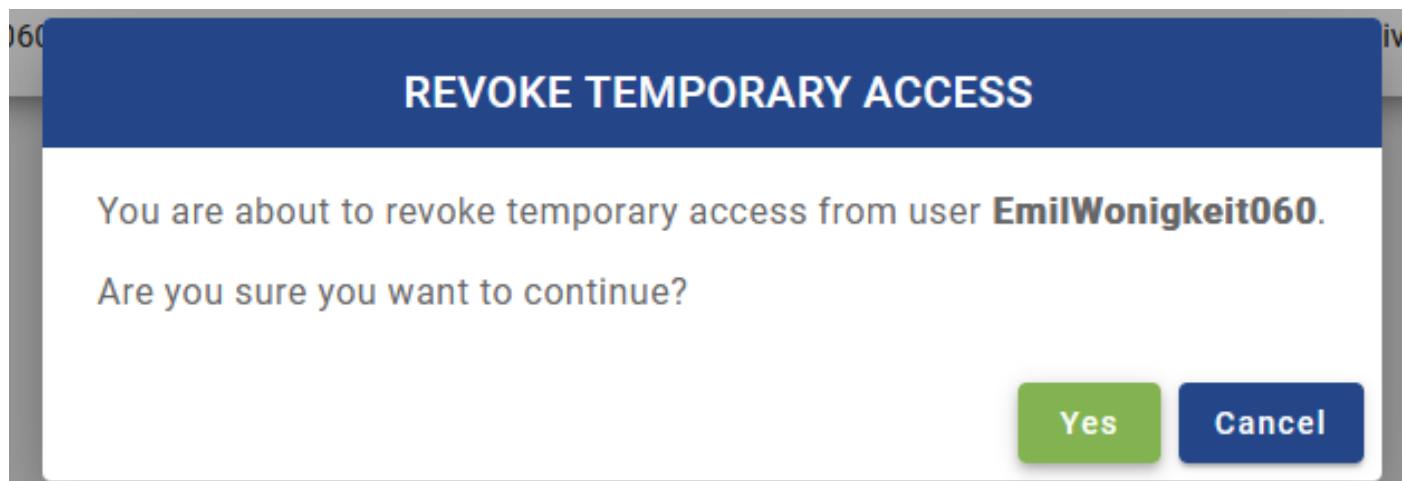
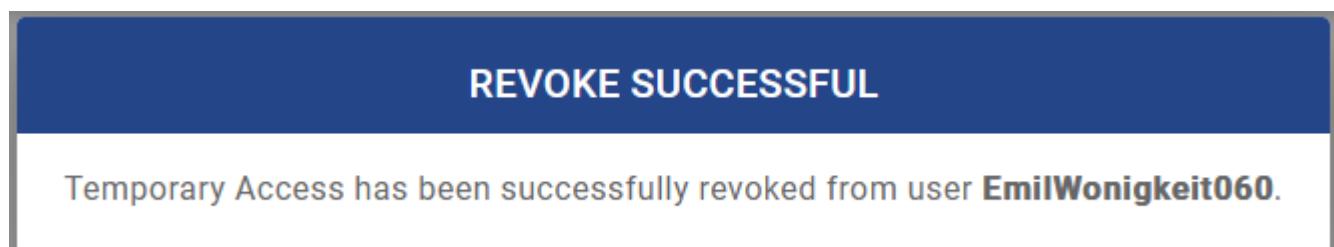


Figure 29: Revoke Access

## Revoke Success Notification



## 2.35 Delete Delegation of Authority

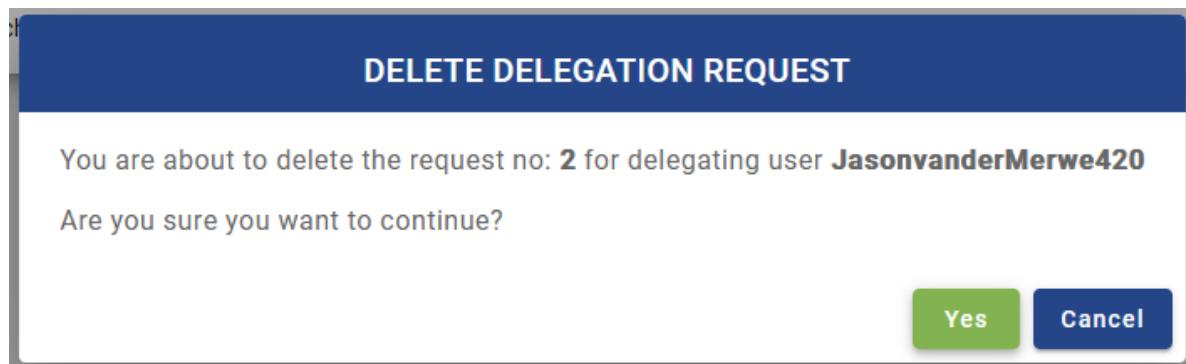
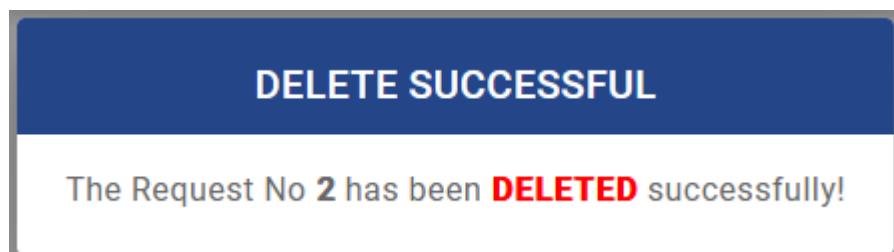


Figure 30: Delete Delegation of Authority

## Delete Success Notification



USE CASE 1.7-1.8 & 2.25 & 2.27-2.30

### 1.7 Admin View All Help Screen

| No. | Category       | Name                   | Description                            | Video   | User Manual                                |
|-----|----------------|------------------------|--|---|--|
| 1   | Administration | DeleteMandateLimit     | Process of deleting Mandate Limit      | <a href="#">Clip 2.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> |
| 2   | Vendor         | CreateVendor           | Process of creating Vendor             | <a href="#">NoSQL Tutorial for Beginners - Introduction to NoSQL Databases - NoSQL Databases Tutorial.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| 3   | Procurement    | EditProcurementRequest | Process of editing procurement request | <a href="#">SQL vs NoSQL or MySQL vs MongoDB.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> |
| 4   | Inventory      | ViewBudgetLines        | Process of viewing Budget Lines        | <a href="#">Types of NoSQL Databases.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> |

### 1.7 User View All Help Screen

| Category       | Name               | Description                       | Video                      | User Manual                                |
|----------------|--------------------|-----------------------------------|----------------------------|--|
| Finance        | DeleteDepartment   | Process of deleting Department    | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Administration | DeleteMandateLimit | Process of deleting Mandate Limit | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Vendor         | CreateVendor       | Process of creating Vendor        | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |

1.7 When you click the “View User Manual” button:



Figure 31: View User Manual Screens

### 2.25-2.26 Settings side-nav to Backup and Restore

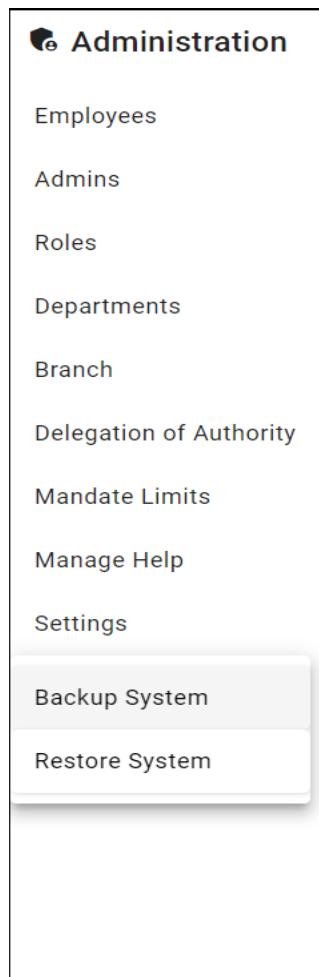


Figure 32: Settings dropdown menu for Backup and Restore

### 2.25 When the “Backup” button is clicked on Settings dropdown:

The screenshot shows the 'Manage Admins' screen within the MOYO application. The sidebar on the left remains the same as in Figure 32. A modal dialog box is centered on the screen with the title 'BACKUP SYSTEM DATA'. The dialog contains the message: 'You are about to create a Backup of the systems current data! Are you sure you want to continue?'. At the bottom of the dialog are two buttons: 'Yes' (green) and 'Cancel' (blue).

Figure 33: Backup Screen

2.25 When the “Yes” button is clicked and is still LOADING:

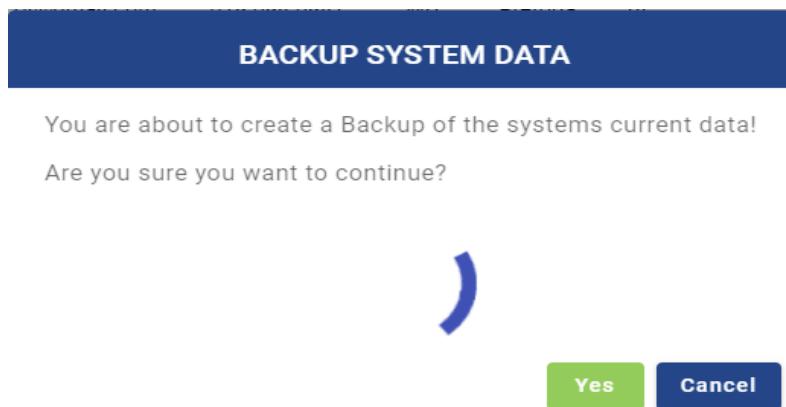


Figure 34: Backup Loading Screen

2.25 When the Backup is Created Successfully:

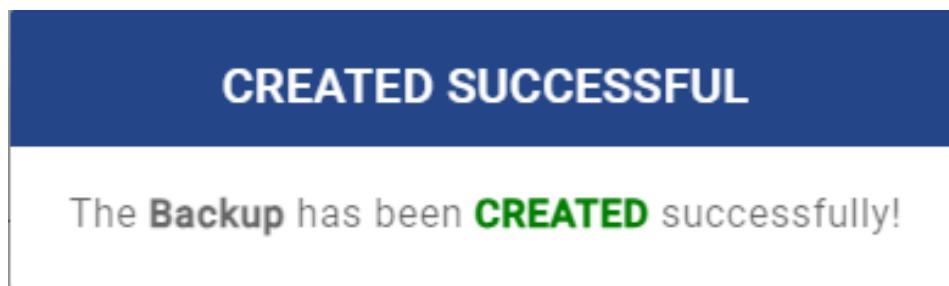


Figure 35: Backup Successful message

2.30 Admin View All Help Screen

| No. | Category       | Name                   | Description                            | Video   | User Manual                                |                      |                        |
|-----|----------------|------------------------|--|---|--|----------------------|------------------------|
| 1   | Administration | DeleteMandateLimit     | Process of deleting Mandate Limit      | <a href="#">Clip 2.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 2   | Vendor         | CreateVendor           | Process of creating Vendor             | <a href="#">NoSQL Tutorial for Beginners - Introduction to NoSQL Databases - NoSQL Databases Tutorial.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 3   | Procurement    | EditProcurementRequest | Process of editing procurement request | <a href="#">SQL vs NoSQL or MySQL vs MongoDB.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 4   | Inventory      | ViewBudgetLines        | Process of viewing Budget Lines        | <a href="#">Types of NoSQL Databases.mp4</a>  | <a href="#">Procurement Manual (1).pdf</a> | <a href="#">Edit</a> | <a href="#">Delete</a> |

Figure 36: View All Help Screen

### 2.30 User View All Help Screen

| Category       | Name               | Description                       | Video                      | User Manual                                |
|----------------|--------------------|-----------------------------------|----------------------------|--|
| Finance        | DeleteDepartment   | Process of deleting Department    | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Administration | DeleteMandateLimit | Process of deleting Mandate Limit | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Vendor         | CreateVendor       | Process of creating Vendor        | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |

Figure 37: View All Help Screen

### 2.27 Add Help:

### Add Help

---

**Name:**

Not more than 32 characters long.

**Description:**

Not more than 50 characters long.

---

**Help Video:**

Choose File
Clip 2.mp4

**User Manual:**

Choose File
Procurement..anual (1).pdf

---

**Category**

Choose one\*  
 Administration

---

Save
Cancel

Figure 38: Add Help Screen

2.27 Add Help loading Screen:

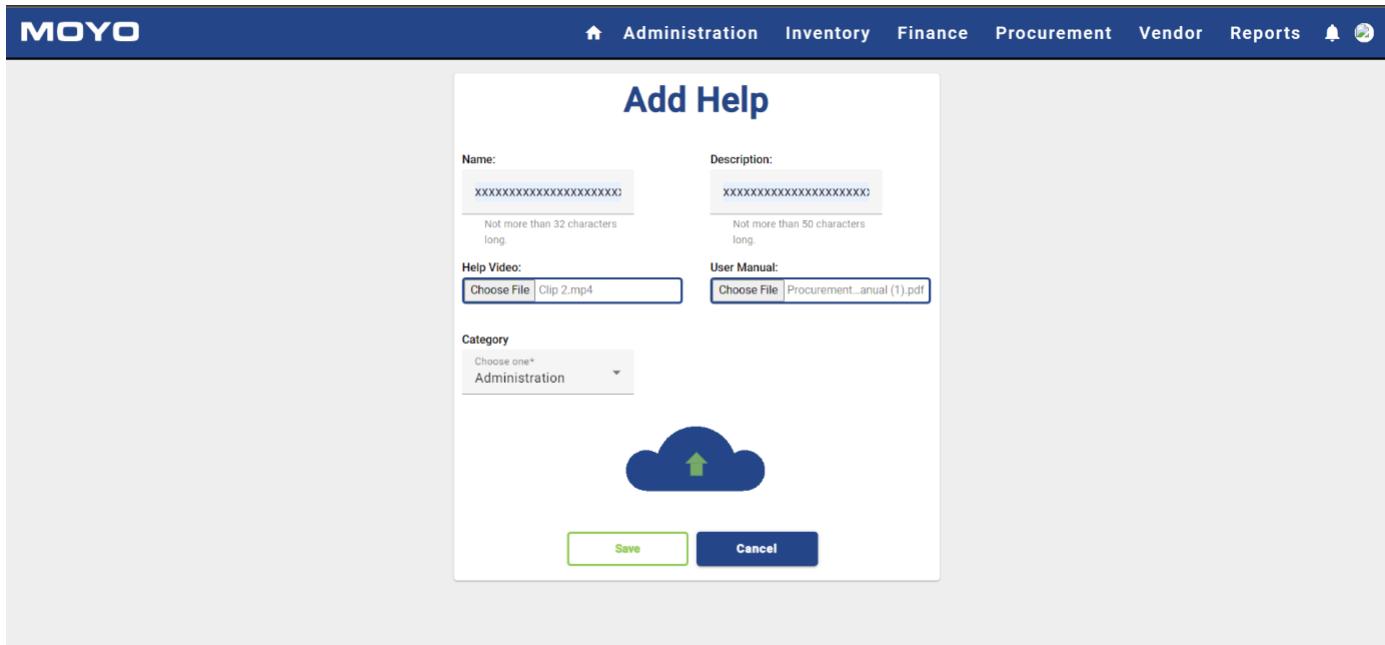


Figure 39: Add Help Loading Screen

2.27 Add Help successful Message:

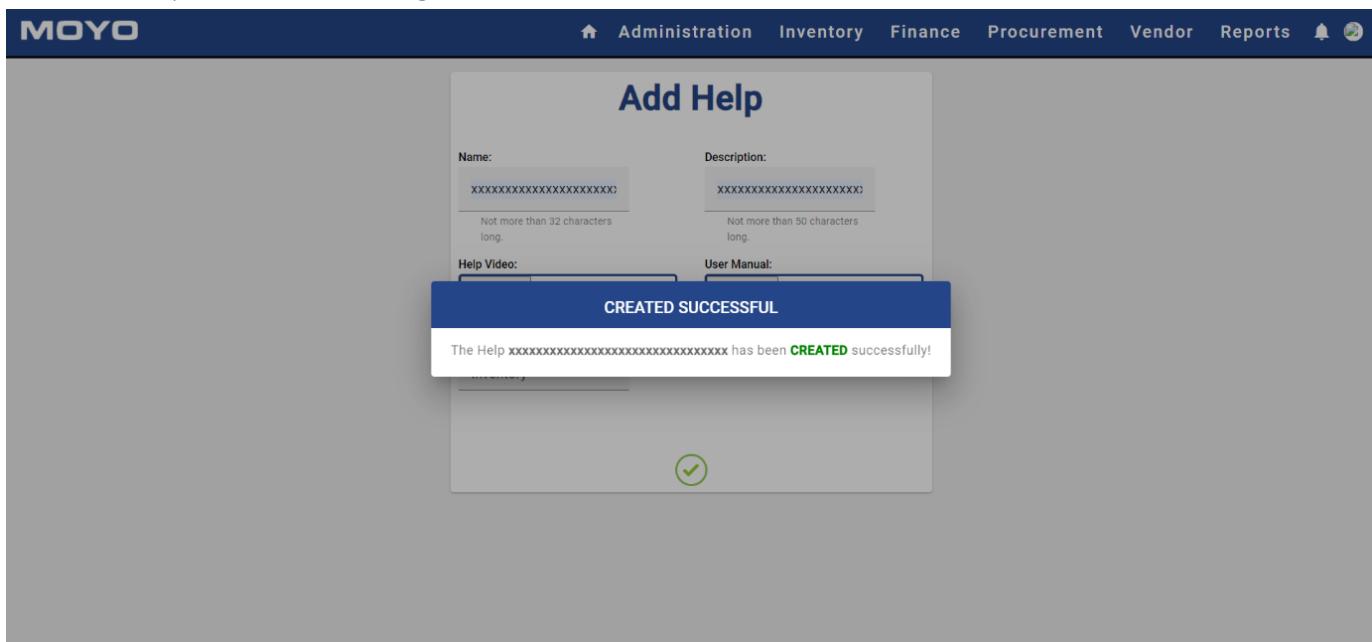


Figure 40: Add help Successful message

2.27 Add Help Unsuccessful Message:

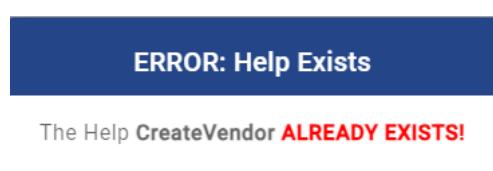


Figure 41: Add help Unsuccessful message

2.28 Edit Help screen:

The screenshot shows the MOYO software interface with a dark blue header bar. The header contains the MOYO logo on the left and navigation links for Administration, Inventory, Finance, Procurement, Vendor, Reports, and a bell icon on the right.

The main content area is titled "Edit Help". It features several input fields and buttons:

- Name:** A text input field containing "xxxxxxxxxxxxxxxxxxxxxx". Below it is a placeholder: "Not more than 32 characters long."
- Description:** A text input field containing "xxxxxxxxxxxxxxxxxxxxxx". Below it is a placeholder: "Not more than 50 characters long."
- Help Video:** A file upload button labeled "Choose File" with the file name "Clip 2.mp4".
- User Manual:** A file upload button labeled "Choose File" with the file name "Procurement...anual (1).pdf".
- Category:** A dropdown menu labeled "Choose one\*" with "Inventory" selected.
- Buttons:** At the bottom are two buttons: a green "Save" button and a dark blue "Cancel" button.

Figure 42: Edit Help Screen

### 2.28 Edit Help Loading screen:

The screenshot shows the 'Edit Help' form. It includes fields for 'Name' (containing 'xxxxxxxxxxxxxxxxxxxx'), 'Description' (containing 'xxxxxxxxxxxxxxxxxxxx'), 'Help Video' (with a file chosen: 'Data Wareho...Big Data.mp4'), 'User Manual' (with a file chosen: 'Procurement...anual (1).pdf'), 'Category' (set to 'Administration'), and a central 'Cloud' icon with an upward arrow. At the bottom are 'Save' and 'Cancel' buttons.

Figure 43: Edit Help Loading screen

### Edit Help Successful Message:

The screenshot shows the 'Edit Help' form again, but with a prominent blue 'EDIT SUCCESSFUL' banner at the top. Below the banner, a message reads 'The Help xxxxxxxxxxxxxxxxxxxxxxx has been EDITED successfully!' with a green checkmark icon. The rest of the form fields are visible but appear identical to Figure 43.

Figure 44: Edit Help successful Message

### 2.28 Edit Help Unsuccessful Message:

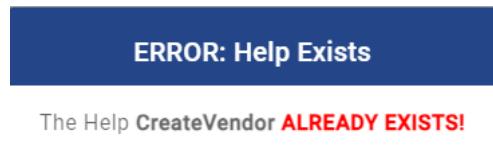


Figure 45: Edit Help successful Message

### 2.29 Delete Help screen:

The screenshot shows the "Manage Help" screen. On the left is a sidebar with navigation links: Administration, Employees, Admins, Roles, Departments, Branch, Delegation of Authority, Mandate Limits, Manage Help, and Settings. The main area displays a table of help entries. One entry, "DeleteMandateLimit", is selected. A modal dialog titled "DELETE HELP" is overlaid on the screen, asking "You are about to delete the Help: DeleteMandateLimit. Are you sure you want to continue?". It has "Yes" and "Cancel" buttons.

| No. | Category       | Name               | Description                       | Video      | User Manual                                |
|-----|----------------|--------------------|-----------------------------------|------------|--|
| 11  | Administration | DeleteMandateLimit | Process of deleting Mandate Limit | Clin 2.mp4 | <a href="#">Procurement Manual (1).pdf</a> |

Figure 46: Delete Help Screen

2.29 Delete Help successful message:

| No. | Category       | Name               | Description                       | Video      | User Manual                |
|-----|----------------|--------------------|-----------------------------------|------------|----------------------------|
| 11  | Administration | DeleteMandateLimit | Process of deleting Mandate Limit | Clip 2.mp4 | Procurement Manual (1).pdf |

Figure 47: Delete help successful message

2.30 View All Help with Admin Privileges:

| No. | Category       | Name                   | Description                            | Video   | User Manual                | Edit                  | Delete                  |
|-----|----------------|------------------------|--|---|----------------------------|-----------------------|-------------------------|
| 1   | Administration | DeleteMandateLimit     | Process of deleting Mandate Limit      | Clip 2.mp4  | Procurement Manual (1).pdf | <button>Edit</button> | <button>Delete</button> |
| 2   | Vendor         | CreateVendor           | Process of creating Vendor             | NoSQL Tutorial for Beginners - Introduction to NoSQL Databases - NoSQL Databases Tutorial.mp4 | Procurement Manual (1).pdf | <button>Edit</button> | <button>Delete</button> |
| 3   | Procurement    | EditProcurementRequest | Process of editing procurement request | SQL vs NoSQL or MySQL vs MongoDB.mp4  | Procurement Manual (1).pdf | <button>Edit</button> | <button>Delete</button> |
| 4   | Inventory      | ViewBudgetLines        | Process of viewing Budget Lines        | Types of NoSQL Databases.mp4  | Procurement Manual (1).pdf | <button>Edit</button> | <button>Delete</button> |

Figure 48: View All Help as Admin Screen

2.30 View All Help without Admin Privileges:

| Category       | Name               | Description                       | Video                      | User Manual                                |
|----------------|--------------------|-----------------------------------|----------------------------|--|
| Finance        | DeleteDepartment   | Process of deleting Department    | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Administration | DeleteMandateLimit | Process of deleting Mandate Limit | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |
| Vendor         | CreateVendor       | Process of creating Vendor        | <a href="#">Clip 2.mp4</a> | <a href="#">Procurement Manual (1).pdf</a> |

Figure 49: View All Help as user Screen

2.30 View help Video link:

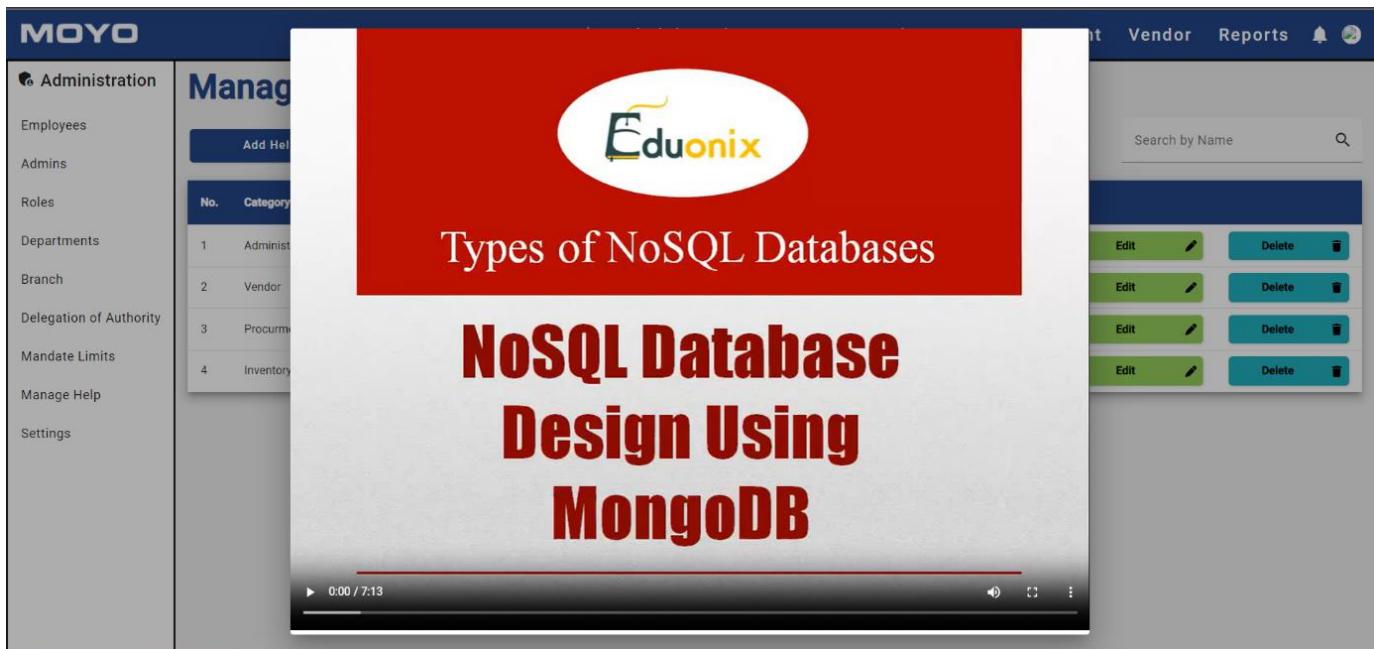


Figure 50: Help Video Screen

2.30 View help User Manual link:



Figure 51: Help User Manual screen

## USE CASE 2.21-2.24 &amp; 4.3-4.6

2.21 Create Mandate Limit Screen:

# Create Mandate Limit

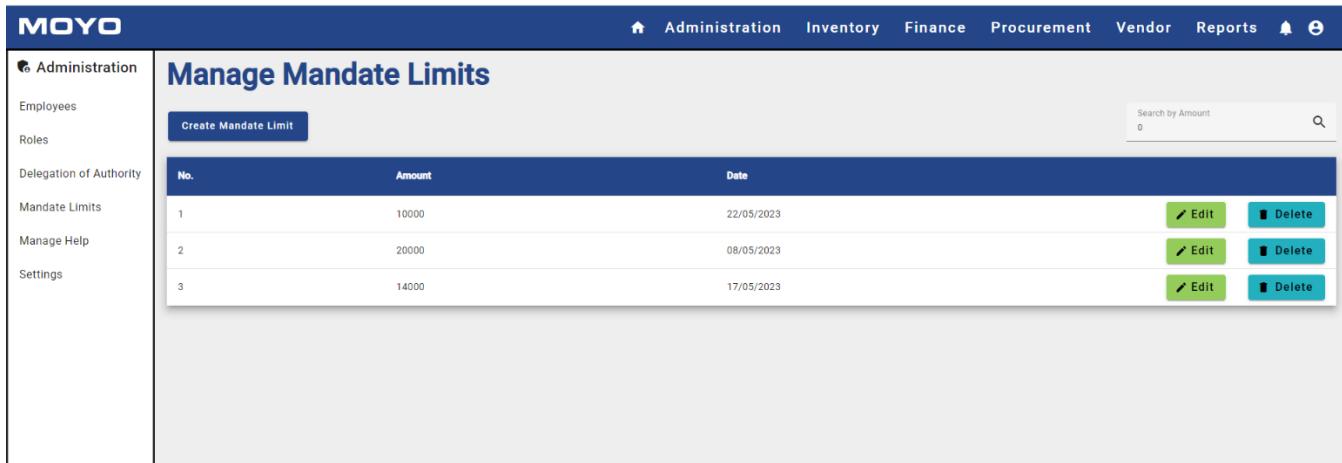
Amount:

Date:  

**Save** **Cancel**

Figure 52: Create Mandate Limit Screen

2.22 View Mandate Limit Screen:



| No. | Amount | Date       | Edit | Delete |
|-----|--------|------------|------|--------|
| 1   | 10000  | 22/05/2023 |      |        |
| 2   | 20000  | 08/05/2023 |      |        |
| 3   | 14000  | 17/05/2023 |      |        |

2.23 Update Mandate Limit Screen:

# Update Mandate Limit

Amount:

Date:

Figure 53: Update Mandate Limit Screen

2.24 Delete Mandate Limit Screen:

## DELETE MANDATE LIMIT

You are about to delete the Mandate Limit with amount: **14000**

Are you sure you want to continue?

Figure 54: Delete Mandate Limit Screen

Successful Delete Mandate Limit Screen:

## DELETE SUCCESSFUL

The mandate limit with amount **14000** has been **DELETED** successfully!

Figure 55: Successful Delete Mandate Limit Screen

## 4.3 Create Budget Allocation Screen:

# Create Budget Allocation

Date:

07/05/2023

Total:

400000

Year:

2023

Department:

Choose one\*

BE

**Save****Cancel**

Figure 56: Create Budget Allocation Screen

## 4.4 View Budget Allocation Screen:

|  |  | Manage Budget Allocations |            |              |      |                |              |      |        |
|--|--|---------------------------|------------|--------------|------|----------------|--------------|------|--------|
|  |  | Create Allocation         |            |              |      | Search by Year |              |      |        |
|  |  | No.                       | Department | Date Created | Year | Total          |              |      |        |
|  |  | 1                         | BE         | 08/05/2023   | 2023 | 40000          | Budget Lines | Edit | Delete |
|  |  | 2                         | BE         | 19/05/2023   | 2023 | 15000          | Budget Lines | Edit | Delete |

Figure 57: View Budget Allocation Screen

## 4.5 Edit Budget Allocation Screen:

# Edit Budget Allocation

Date Created: 08/05/2023

Total: 40000

Year: 2023

Department: Choose one\*

Figure 58: Edit Budget Allocation Screen

## 4.6 Delete Budget Allocation Screen:

## DELETE BUDGET ALLOCATION

You are about to delete the Budget Allocation with total: **15000** for year 2023  
Are you sure you want to continue?

Figure 59: Delete Budget Allocation Screen

### 4.6 Successful Delete Budget Allocation Screen:



Figure 60: Successful Delete Budget Allocation Screen

### 4.7 Create Budget Category Screen:

The screenshot shows a form titled "Create Budget Category". It has two input fields: "Name:" containing "XXXXXXXXXXXXXX" and "Description:" containing "XXXXXXXXXXXXXXXXXXXXXX". Below each field is a validation message: "Not more than 32 characters long." and "Not more than 50 characters long." respectively. At the bottom are "Save" and "Cancel" buttons.

Figure 61 Create Budget Category Screen

### 4.8 View Budget Category Screen:

The screenshot shows a table titled "Manage Budget Categories". The table has columns: No., Name, and Description. A single row is shown with "No." 5, "Name" "XXXXXXXXXXXXXX", and "Description" "XXXXXXXXXXXXXX". At the bottom right of the table are "Edit" and "Delete" buttons.

Figure 62 View Budget Category Screen

## 4.9 Edit Budget Category Screen:

# Edit Budget Category

Name:

Not more than 32 characters long.

Description:

Not more than 50 characters long.

**Save** **Cancel**

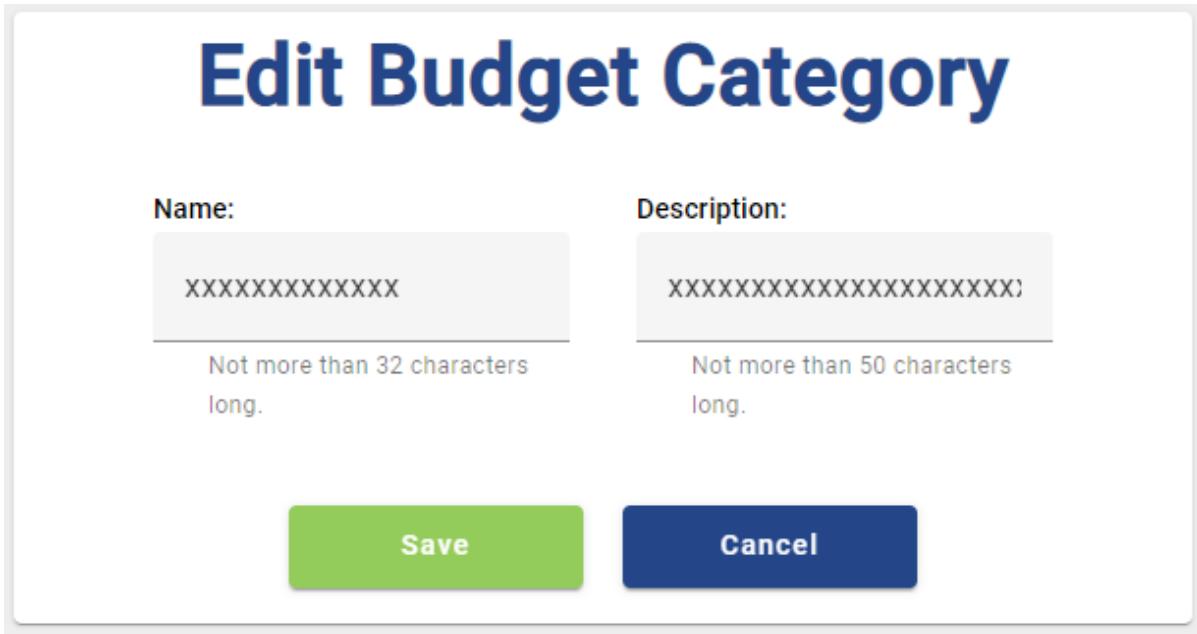


Figure 63 Edit Budget Category Screen

## 4.10 Delete Budget Category Screen:

## DELETE BUDGET CATEGORY

You are about to delete the Budget Category: XXXXXXXXXXXXXXXX

Are you sure you want to continue?

**Yes** **Cancel**

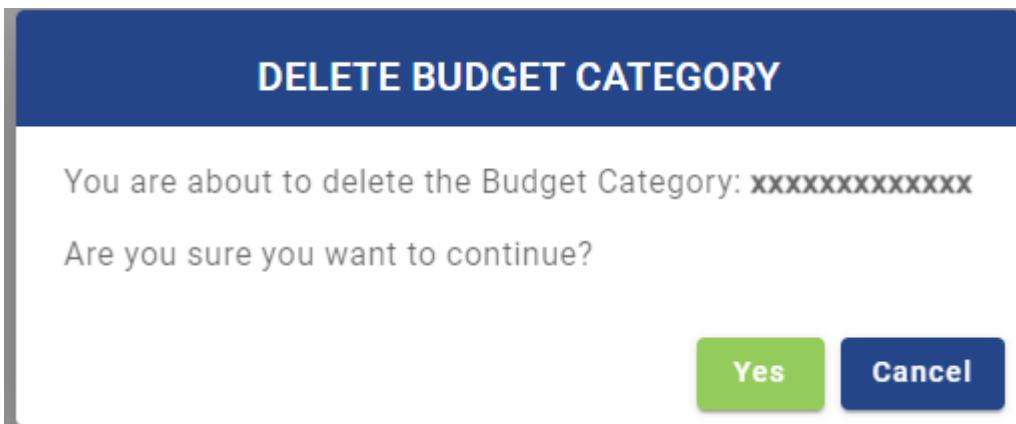


Figure 64 Delete Budget Category Screen

## 4.10 Successful Delete Budget Category Screen:

## DELETE SUCCESSFUL

The budget category XXXXXXXXXXXXXXX has been **DELETED** successfully!

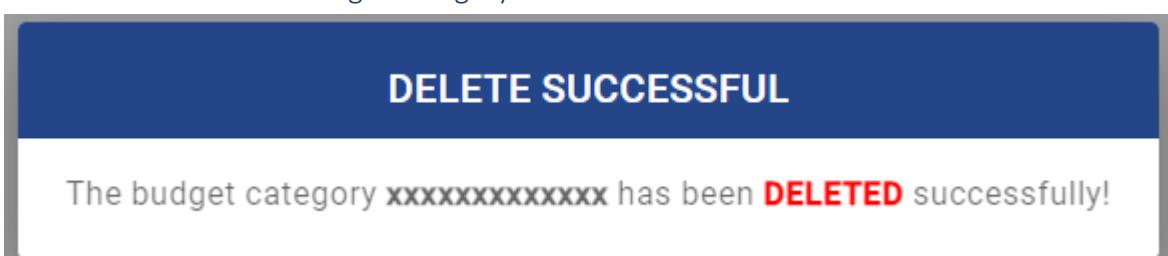


Figure 65 Successful Delete Budget Category Screen

USE CASE 5.1-5.8 &amp; 5.9, 5.10 &amp; 1.1-1.4 &amp; 3.1-3.4

## 5.1 Create Consumable Screen

### Create Consumable

Name:  Description:   
Not more than 32 characters long.

Category:  Minimum Reorder Quantity:

Maximum Reorder Quantity:  On Hand:

**Save** **Cancel**

Figure 66 Create Consumable Screen

## 5.1 Successful create notification

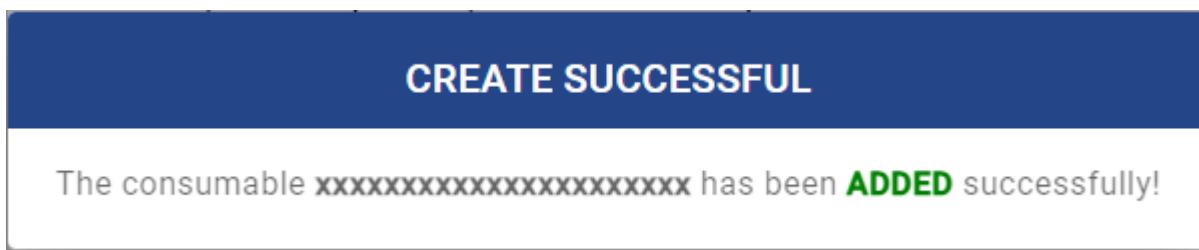


Figure 67 Successful Create Notification

## 5.1 Unsuccessful create notification

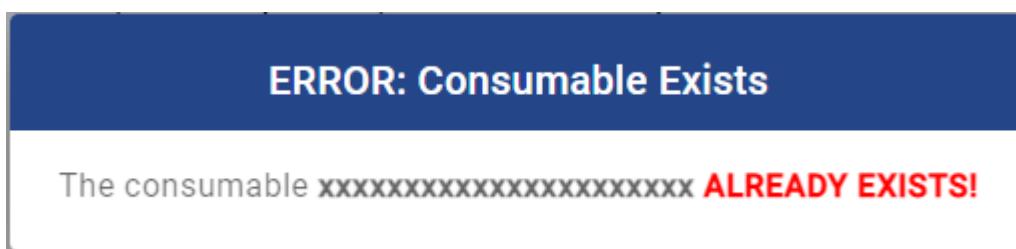


Figure 68 Unsuccessful Create Notification

### 5.2 View Consumables Screen

The screenshot shows the 'View Consumables' page. On the left sidebar, under 'Inventory', there are links for 'View Consumables', 'View Consumable Categories', and 'View Assets'. The main area has a title 'View Consumables' and a 'Create Consumable' button. A search bar at the top right says 'Search by Name...'. Below is a table with one row:

| Name  | Description | On Hand | Minimum Reorder Quantity | Maximum Reorder Quantity | Category |
|-------|-------------|---------|--------------------------|--------------------------|----------|
| Water | Bonaqua     | 2       | 50                       | 100                      | Kitchen  |

At the bottom right of the table are 'Edit' and 'Delete' buttons.

Figure 69 View Consumables Screen

### 5.3 Update Consumables Screen

The screenshot shows the 'Update Consumable' form. It has two columns of input fields:

- Name:** Water (maxlength: 32 characters)
- Description:** Bonaqua (maxlength: 50 characters)
- Category:** Kitchen (dropdown menu)
- Minimum Reorder Quantity:** 50
- Maximum Reorder Quantity:** 100
- On Hand:** 2

At the bottom are 'Save' and 'Cancel' buttons.

Figure 70 Update Consumable Screen

## 5.3 No changes made edit notification

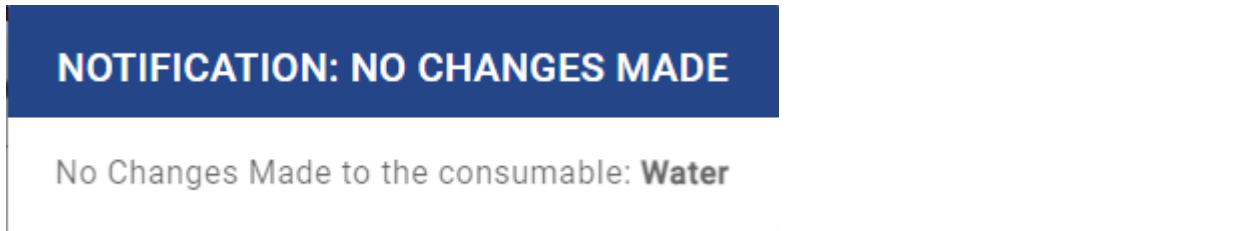


Figure 71 No Change Made Update Notification

## 5.3 Successful edit notification



Figure 72 Successful Update Notification

## 5.4 Delete Consumable dialog

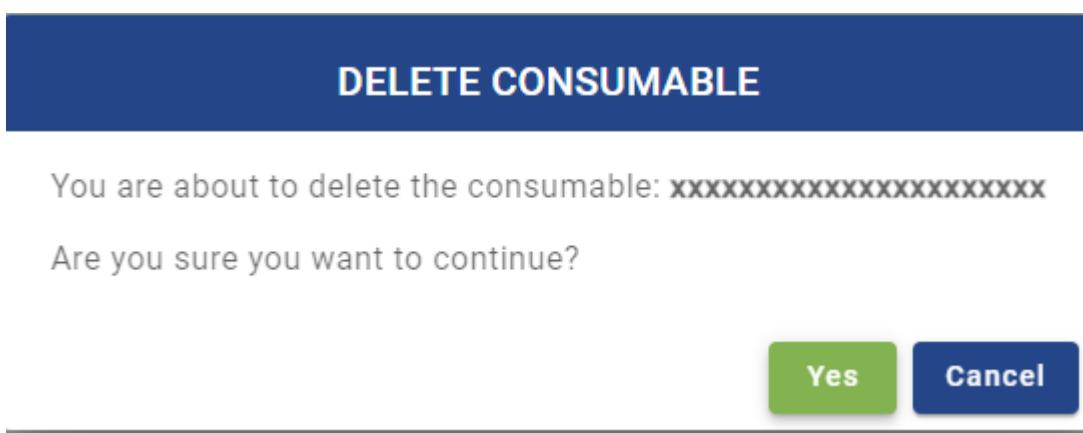


Figure 73 Delete Consumable Dialog

## 5.4 Delete notification



Figure 74 Delete Notification

## 5.5 Create category screen

# Create Consumable Category

Name:

Not more than 32 characters long.

Description:

Not more than 50 characters long.

**Save**    **Cancel**

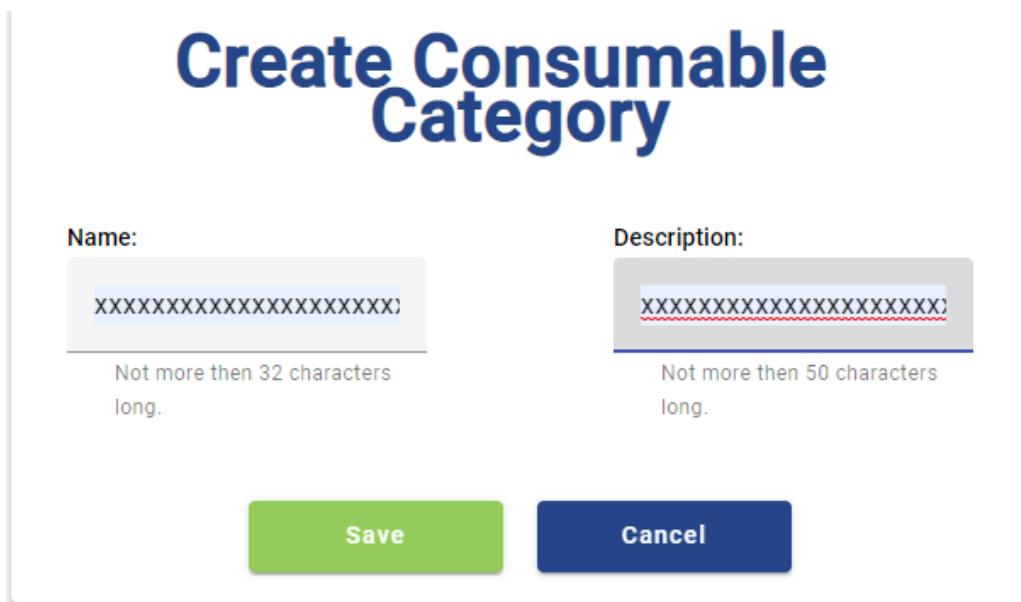


Figure 75 Create Category Screen

## 5.5 Unsuccessful create notification

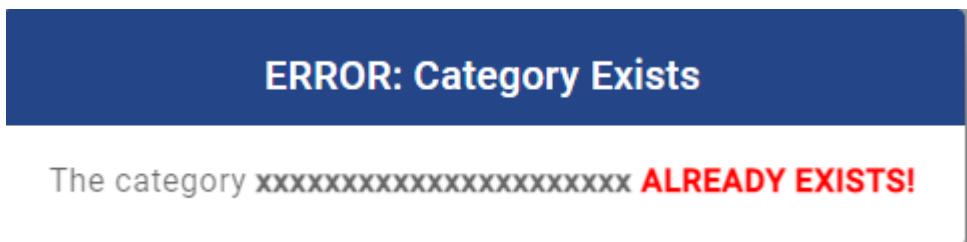


Figure 76 Unsuccessful Create Notification

## 5.5 Successful create notification

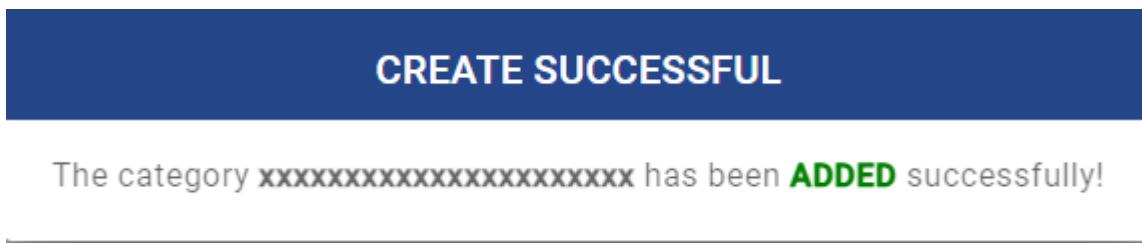


Figure 77 Success Create Notification

### 5.6 View Consumable categories screen

| Name    | Description |
|---------|-------------|
| Kitchen | Mykitchen   |

Figure 78 View Consumable Categories Screen

### 5.7 Update consumable category screen

## Update Consumable Category

|                                      |  |
|--------------------------------------|--|
| <b>Name:</b>                         | <b>Description:</b>                    |
| <input type="text" value="Kitchen"/> | <input type="text" value="Mykitchen"/> |
| Not more than 32 characters long.    | Not more than 50 characters long.      |
| <b>Save</b>                          | <b>Cancel</b>                          |

Figure 79 Update Consumable Category Screen

## 5.7 No changes made notification



Figure 80 No Changes Made Notification

## 5.7 Successful edit notification



Figure 81 Successful Update Notification

## 5.8 Delete category prompt

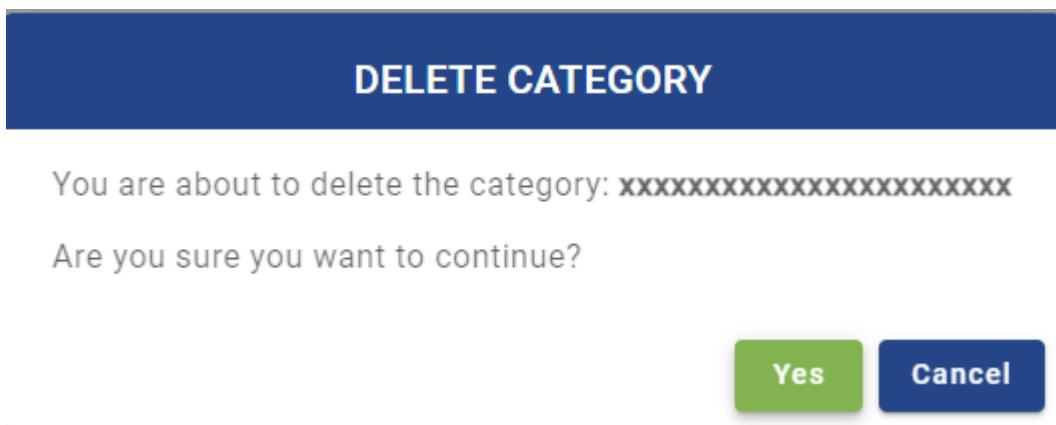


Figure 82 Delete Category Prompt

## 5.8 Successful delete notification

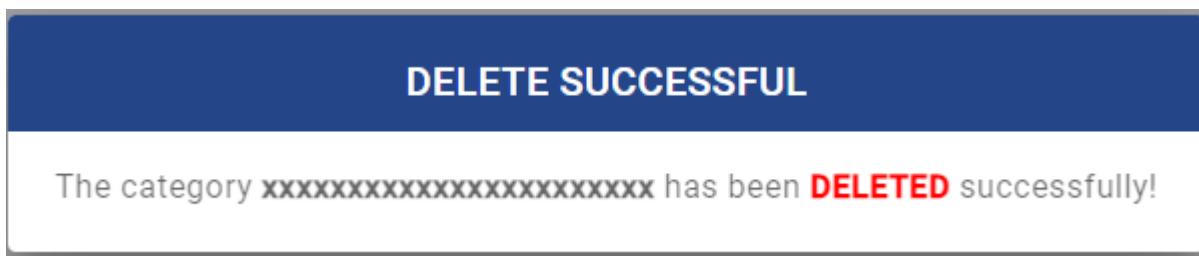


Figure 83 Successful Delete Notification

5.8 Association found notification

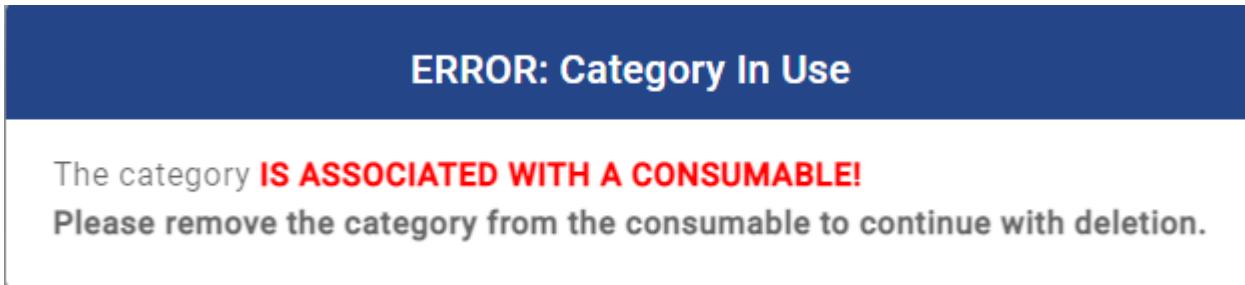


Figure 84 Association Found Notification

5.9 Update Stock



Figure 85 Update Stock Screen

## 5.9 Update Stock Chart

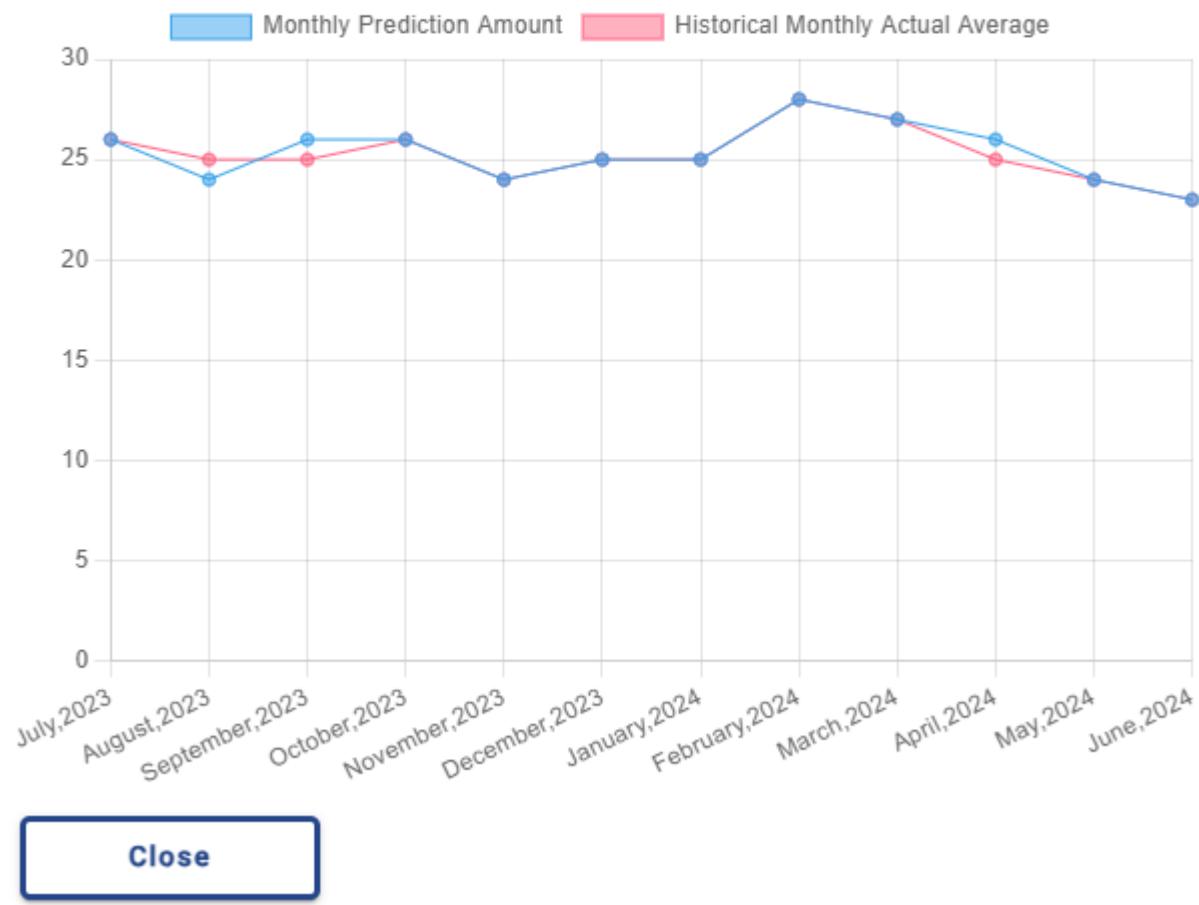
[Update Stock](#)[View Predictions](#)**Coffee Stock Level Predictions For The Next 12 Months**

Figure 86 Update Stock Chart

## 5.10 View Consumable Screen with Export Details Button

The screenshot shows the MOYO software's 'View Consumables' screen. The left sidebar has 'Inventory' selected. The main area title is 'View Consumables'. It features two buttons: 'Create Consumable' and 'Export Details'. A search bar at the top right says 'Search by Name...' with a magnifying glass icon. Below is a table with the following data:

| Name  | Description | On Hand | Minimum Reorder Quantity | Maximum Reorder Quantity | Category | Actions  |
|-------|-------------|---------|--------------------------|--------------------------|----------|--|
| Water | Bonaqua     | 2       | 50                       | 100                      | Kitchen  | <button>Edit</button> <button>Stock</button> <button>Delete</button> |

Figure 87 View Consumable Screen with Export Details Button

## 5.10 Export Details PDF

| <b>Consumable Inventory Details Report</b> |          |                           |                           |         |
|--|----------|---------------------------|---------------------------|---------|
| <b><u>Generated On: 07/26/2023</u></b>     |          |                           |                           |         |
| <b><u>Details Per Consumable Item</u></b>  |          |                           |                           |         |
| Name                                       | Category | Minimum Reorder- Quantity | Maximum Reorder- Quantity | On-Hand |
| Water                                      | Kitchen  | 50                        | 100                       | 6       |
| Coffee                                     | Kitchen  | 19                        | 9                         | 40      |

---

*\*\*End of Report\*\**

Figure 88 Export Details PDF

## 1.1 Login Screen

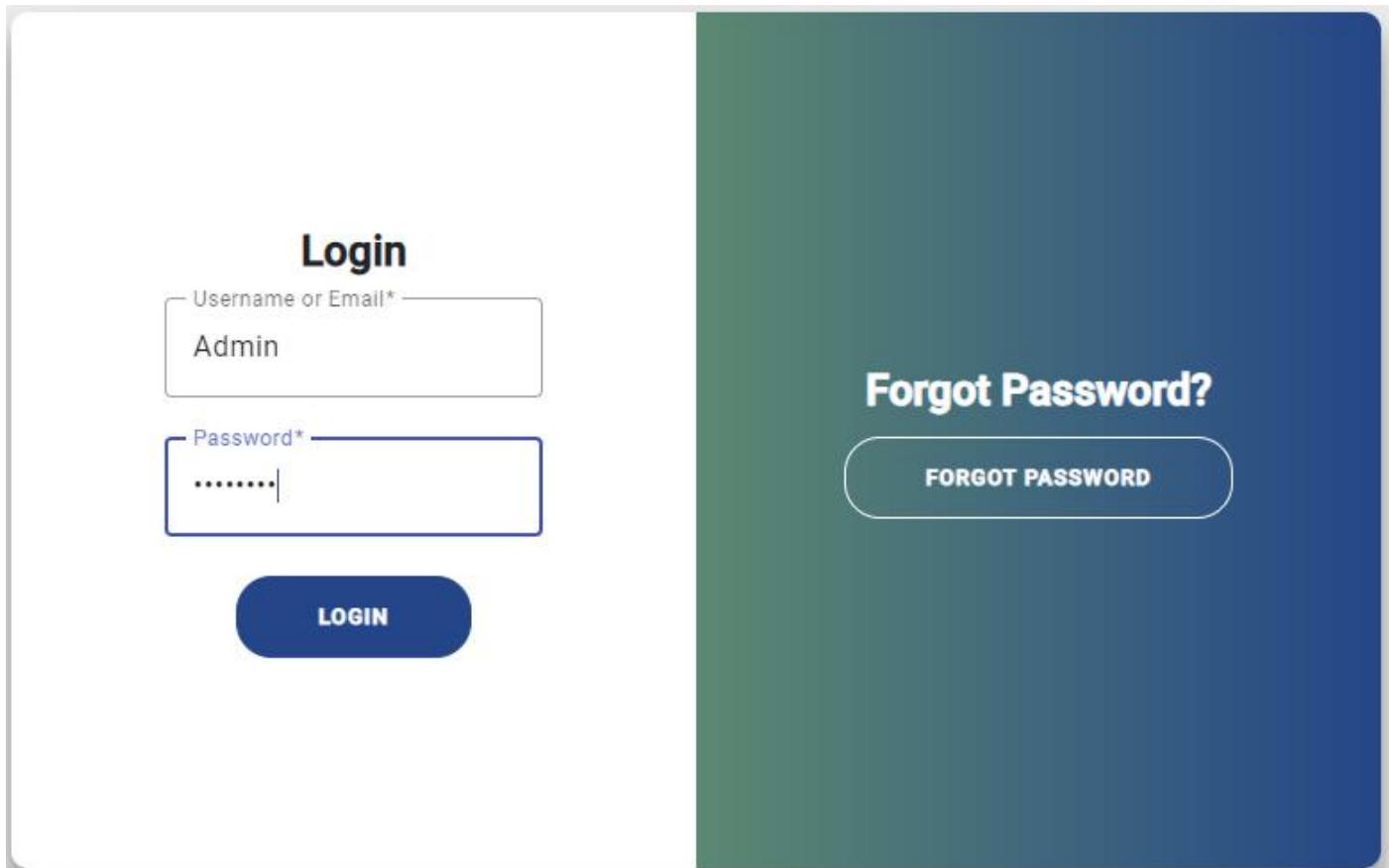


Figure 89 Login Screen

## 1.1 Login Failed Notification

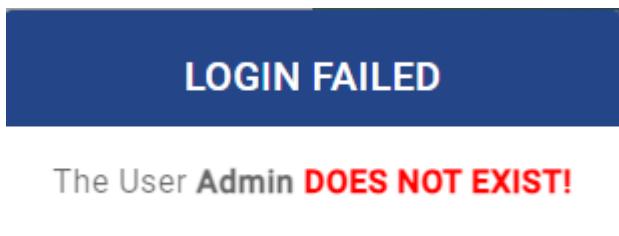


Figure 90 Login Failed Notification

## 1.2 Logout Screen

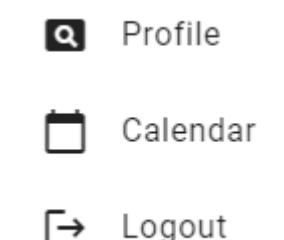


Figure 91 Logout Screen

## 1.3 Forgot Password Screen

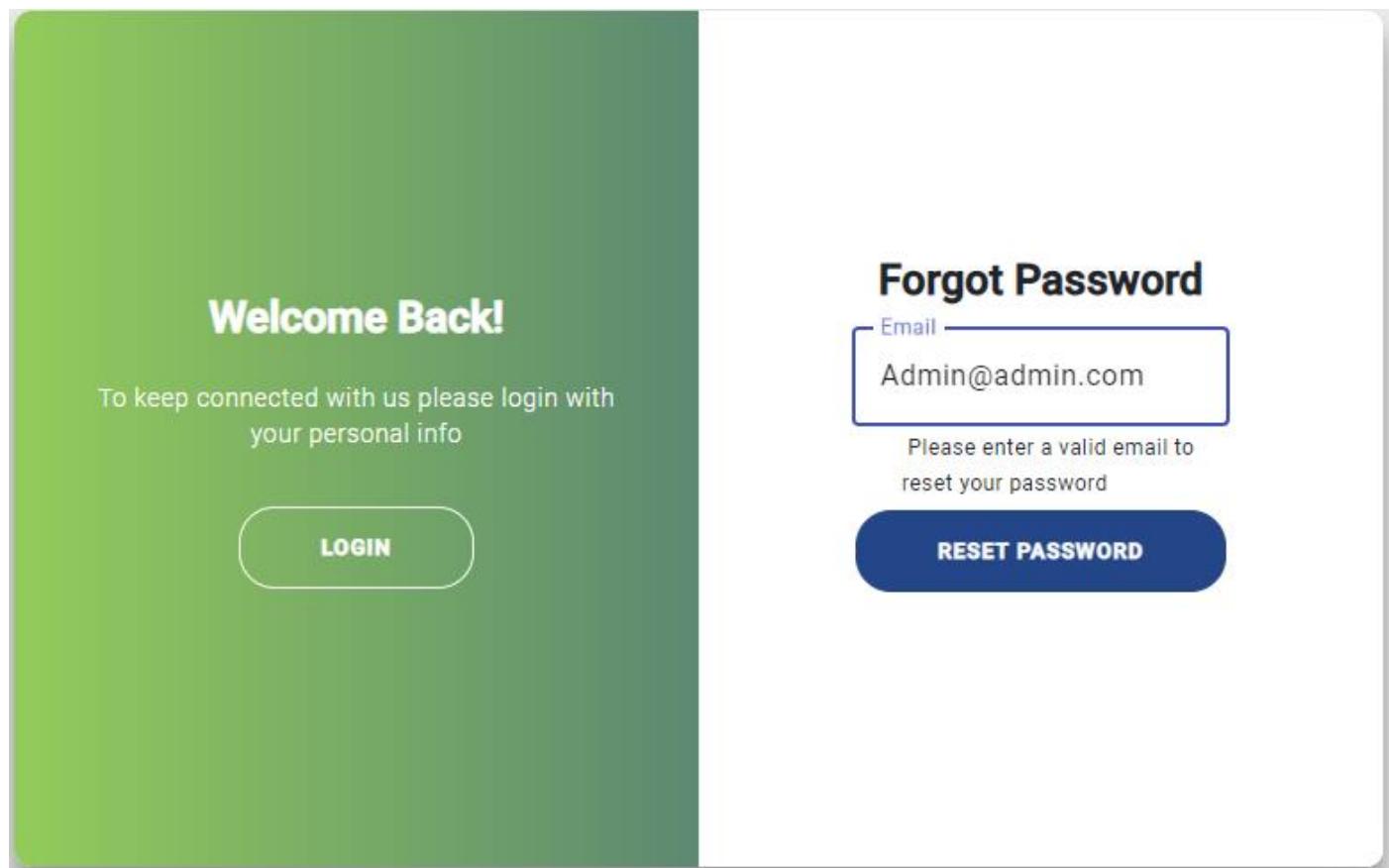


Figure 92 Forgot Password Screen

## 1.3 Forgot Password Failed Notification

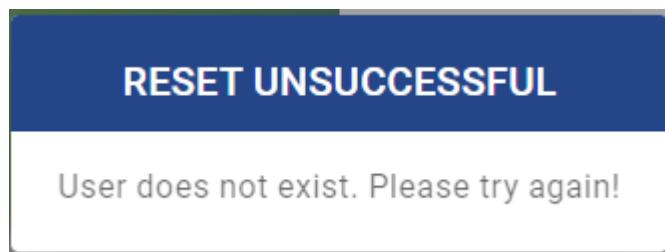


Figure 93 Forgot Password failed Notification

## 1.4 Update Password Screen

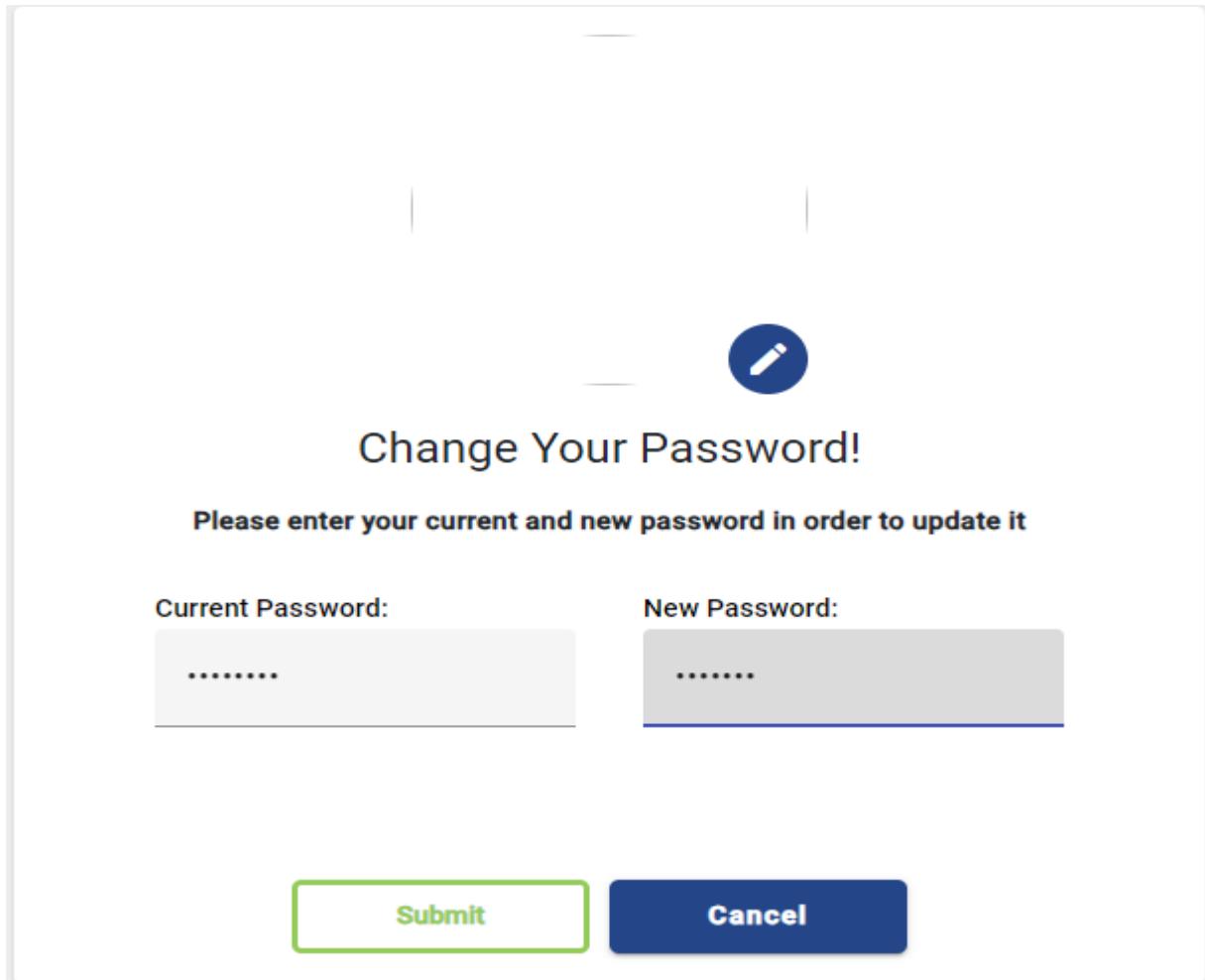


Figure 94 Update Password Screen

## 1.4 Update Password Success Notification

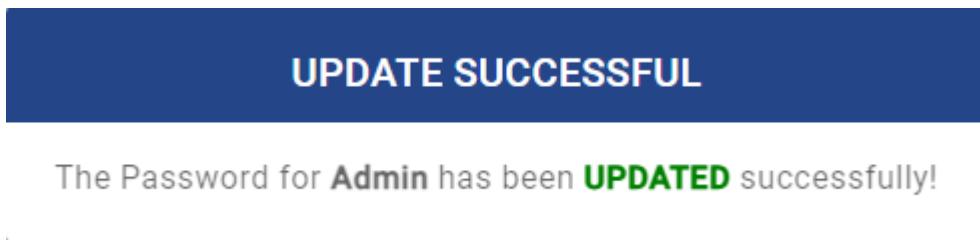


Figure 95 Update Password Success Notification

## 1.4 Update Password Unsuccessful Notification



Figure 96 Update Password Success Notification

## 3.1 Create Procurement Request screen

## Create Procurement Request

Approved Vendor  Other

Request Name:

Not more than 50 characters long.

Request Description:

Not more than 50 characters long.

Vendor:

Select\*

The Procurement Quote:

No file chosen

Figure 97 Create procurement Request screen

## 3.1 Create Procurement Request Approved Vendor screen

## Create Procurement Request

Approved Vendor  Other

Request Name:

Request for x

Not more than 50 characters long.

Request Description:

Some description

Not more than 50 characters long.

Vendor:

Select\*

The Procurement Quote:

WST 212.pdf

Figure 98 Create procurement Request Approved Vendor screen

### 3.1 Create Procurement Request Other screen

# Create Procurement Request

Approved Vendor  Other

|   |   |
|---|---|
| Request Name:   | Request Description:  |
| Request for x   | Some Description  |
| Not more than 50 characters long.   |   |
| Vendor Name:  | Vendor Email:   |
| Makro   | makro@makro   |
| Not more than 32 characters long.   |   |
| Preferred Quote:  | Comparative Quote 1:  |
| <input type="button" value="Choose File"/> Iteration 4.pdf                | <input type="button" value="Choose File"/> Client Memo ...rsion 4_4.pdf |
| Comparative Quote 2:  |   |
| <input type="button" value="Choose File"/> Late registrat...ocument.pdf   |   |
| <input type="button" value="Save"/> <input type="button" value="Cancel"/> |   |

Figure 99 Create procurement Request Other screen

### 3.1 Create Procurement Request Success Notification



Figure 100 Create procurement Request Success Notification

### 3.2 View Procurement Request Screen

| Name                      | Description                       | User  | Vendor                         | Status            | Action                                      |
|---------------------------|-----------------------------------|-------|--------------------------------|-------------------|---|
| Procurement Request for x | A new company procurement request | Admin | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx | Approval Required | <a href="#">Edit</a> <a href="#">Delete</a> |

Figure 101 View procurement Request Screen

### 3.2 View Procurement Request With table dropdown Screen

| Name                      | Description                       | User          | Vendor   | Status            | Action                                      |
|---------------------------|-----------------------------------|---------------|--|-------------------|---|
| Procurement Request for x | A new company procurement request | Admin         | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx                           | Approval Required | <a href="#">Edit</a> <a href="#">Delete</a> |
| Request for x             | Some Description                  | Admin         | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx                           | Approval Required | <a href="#">Edit</a> <a href="#">Delete</a> |
| Quote ID                  | Date Uploaded                     | Vendor Status | Vendor Quote   | Preferred Quote   |   |
| 2                         | 2023/07/26                        | In Progress   | <a href="#">Client Memo of Agreement Version 4_4.pdf</a> | true              |   |
| Request for y             | asdasd                            | Admin         | Makro  | Approval Required | <a href="#">Edit</a> <a href="#">Delete</a> |

Figure 102 View procurement Request With table dropdown Screen

## 3.3 Update Procurement Request Screen

## Edit Procurement Request

Vendor Type: In Progress

Request Name:  Not more than 50 characters long.

Request Description:  Not more than 50 characters long.

Vendor:  Replace Quote:  Choose File No file chosen  
Provide Quote to replace.

| Your Quotes | Quote |
|-------------|-------|
| Quote 1     | Test  |

**Save** **Cancel**

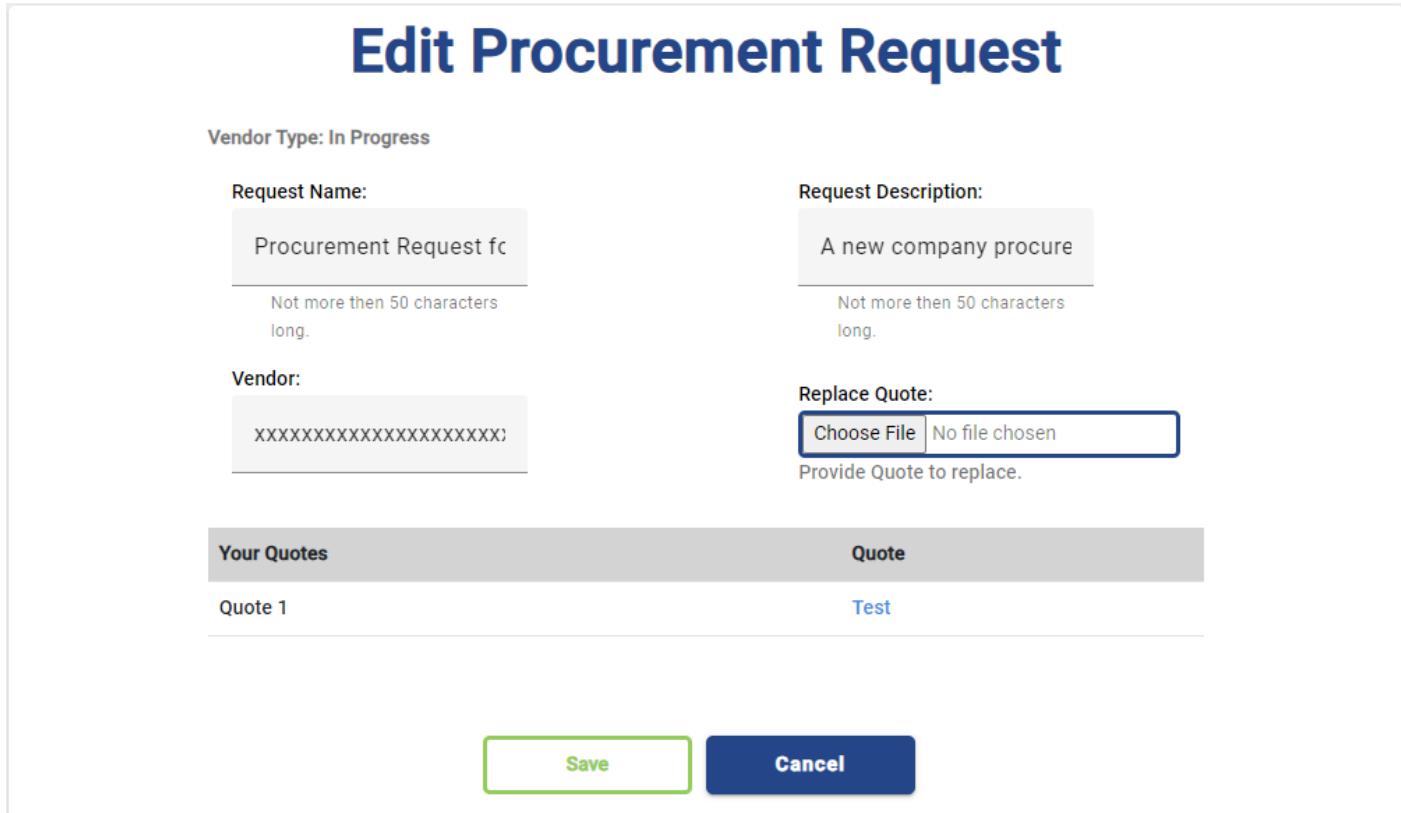


Figure 103 Update procurement Request Screen

## 3.3 Update Procurement Request Success Notification



Figure 104 Update procurement Request success Notification

## 3.4 Delete Procurement Request Screen

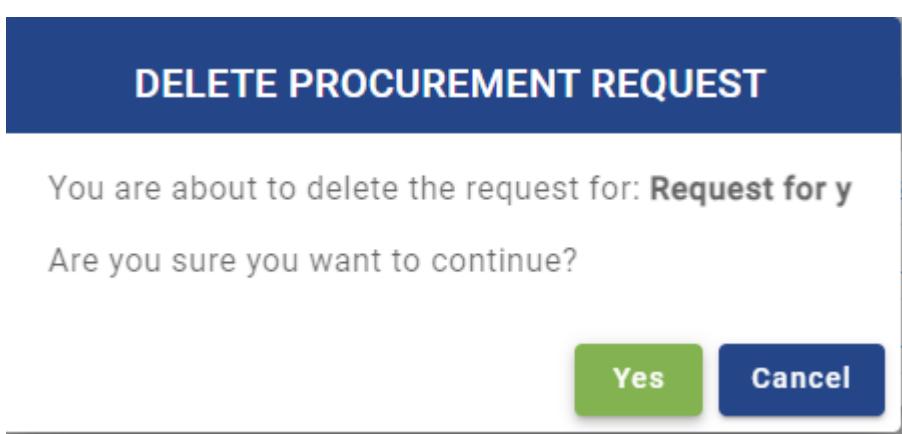


Figure 105 Delete procurement Request Screen

## 3.4 Delete Procurement Request Success Notification

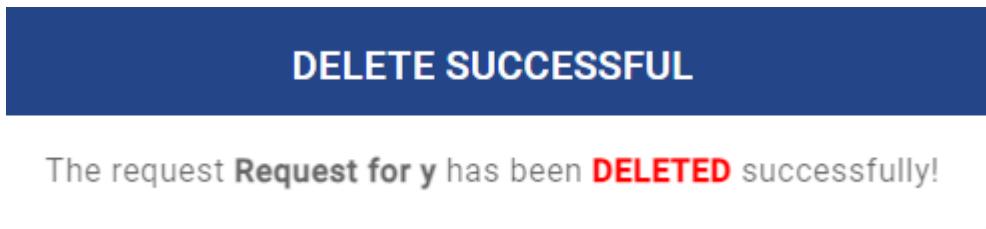


Figure 106 Delete procurement Request Success Notification

USE CASE 3.5-3.7, 6.5, 6.10-6.13

3.5 Approve Procurement Request:

### View Pending Procurement Request

| Name                      | Description                       | User  | Vendor                         |   |
|---------------------------|-----------------------------------|-------|--------------------------------|---|
| Procurement Request for x | A new company procurement request | Admin | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx | <button>View</button> <button>Edit</button> |
| Test                      | Testing                           | Admin | othertest                      | <button>View</button> <button>Edit</button> |
| gasg                      | fasfsafasfsf fsafsafsa            | Admin | gasgag                         | <button>View</button> <button>Edit</button> |

Table 152: View Pending Procurement Requests

[←](#) **Procurement Request For  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx**

**Please Approve or Reject the Procurement Request**

|                    |  |
|--------------------|--|
| Company Name:      | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx                   |
| Company Email:     | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx@xxxx              |
| Description:       | A new company procurement request //             |
| Procurement Quote: | <u>Test</u>                                      |
|                    | <button>Approve</button> <button>Reject</button> |

Table 153: Approve Procurement Request

**REJECTION SUCCESSFUL**

Procurement Request **test** has been **Rejected** successfully!

Table 154: Successfully Reject Request Notification

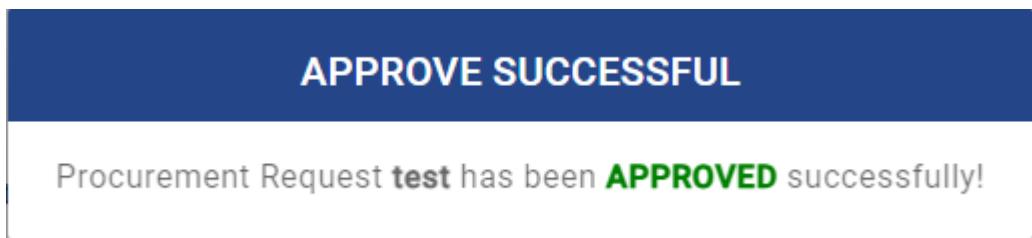


Table 155: Successfully Approve Request Notification

### 3.6 Place Procurement Details:

**Place Procurement Request**

| Name                      | Description                       | User  | Vendor                         | Action        |
|---------------------------|-----------------------------------|-------|--------------------------------|---------------|
| Procurement Request for x | A new company procurement request | Admin | xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx | <b>Create</b> |
| Test                      | Testing                           | Admin | othertest                      | <b>Create</b> |
| gasg                      | fasfsafasfsf fsafsafsa            | Admin | gasgag                         | <b>Create</b> |

Table 156: View Place Procurement Request Screen

## Place Procurement Requests Details

Please fill in the following details:

|  |  |
|--|--|
| Buyer Name:  | <input type="text" value="Buyer Name"/>                          |
| Buyer Email:   | <input type="text" value="Buyer Email"/>                         |
| Item Type:   | <input type="text" value="Consumable"/>                          |
| Select Consumable Item:  | <input type="text" value="Water"/>                               |
| Quantity:  | <input type="text" value="0"/>                                   |
| Account Code:  | <input type="text" value="No Account code exist on the system"/> |
| Payment Type:  | <input type="text" value="Credit"/>                              |
| Has Deposit?   | <input checked="" type="checkbox"/>                              |
| Full Payment Made?   | <input type="checkbox"/>   |
| Proof of payment?  | <input type="checkbox"/>   |
| Total Amount:  | <input type="text" value="0"/>                                   |
| Total Amount Due Date:   | <input type="text" value="Choose a date*"/>                      |
| Comments:  | <input type="text" value="Details Of Service Or Goods"/>         |
| <input type="button" value="Place"/> <input type="button" value="Cancel"/> |  |

Table 157: Place Procurement Details

|                                   |   |
|-----------------------------------|---|
| Description of Asset:             | Details Of Service Or Goods                               |
| Account Code:                     | No Account code exist on the system                       |
| Payment Type:                     | Credit  |
| Has Deposit?                      | <input checked="" type="checkbox"/>                       |
| Deposit Amount:                   | 0   |
| Deposit Due Date:                 | Choose a date*  |
| Full Payment Made?                | <input checked="" type="checkbox"/>                       |
| Paid On Date:                     | Choose a date*  |
| Upload Receipt:                   | <input type="button" value="Choose File"/> No file chosen |
| Proof of payment?                 | <input checked="" type="checkbox"/>                       |
| Upload Proof of payment document: | <input type="button" value="Choose File"/> No file chosen |

Table 158: Place Procurement Details alternatives

## PLACED PROCUREMENT REQUEST

Procurement details for procurement **test** has been **PLACED** successfully!

Table 159: Successfully Placed Procurement Request Notification

## 3.7 Approve Flagged Procurement Details:

| View Flagged Placed Procurement Request |            |               |               |                  |
|---|------------|---------------|---------------|------------------|
| Vendor Name                             | Created By | Mandate Limit | Request Total | Payment Due Date |
| gasgag                                  | We         | R10,000.00    | R50,000.00    | 27/07/2023       |
| othertest                               | We         | R10,000.00    | R15,000.00    | 26/08/2023       |

Table 160: View Flagged Procurement Details

[←](#) **Procurement Details for othertest**

Please Accept Or Reject the following request:

|                         |                          |
|-------------------------|--------------------------|
| Buyer Name:             | Person                   |
| Buyer Email:            | Person@gmail.com         |
| Item Type:              | Consumable               |
| Select Consumable Item: | Water                    |
| Quantity:               | 20                       |
| Account Code:           | 4000                     |
| Payment Type:           | Cash                     |
| Has Deposit?            | <input type="checkbox"/> |
| Full Payment Made?      | <input type="checkbox"/> |
| Proof of payment?       | <input type="checkbox"/> |
| Total Amount:           | 15000                    |
| Total Amount Due Date:  | 08/26/2023               |
| Comments:               | testing comments //      |

**Accept** **Reject**

Table 161: Approve Flagged Procurement Details

**APPROVE SUCCESSFUL**

Procurement Details has been **APPROVED** successfully!

Table 162: Approve Flagged Procurement Details Notification

**REJECTION SUCCESSFUL**

Procurement Details has been **Rejected** successfully!

Table 163: Reject Flagged Procurement Details Notification

### 6.5 Approve Onboard Request

#### Manage Approved or Pending Onboard Requests

| Onboard Request Number | Selected Option | Request Type      | Total Quotes | Created By | Update                  | View                  |
|------------------------|-----------------|-------------------|--------------|------------|-------------------------|-----------------------|
| 1                      | Request pending | General Suppliers | 3            | Admin      | <button>Update</button> | <button>View</button> |
| 2                      | Request pending | General Suppliers | 3            | Admin      | <button>Update</button> | <button>View</button> |
| 3                      | Request pending | Sole Supplier     | 1            | Admin      | <button>Update</button> | <button>View</button> |

Items per page: 5 | 1 – 3 of 3 | < >

Table 164:Approve Onboard Request View Screen

## Onboard Request #1

Please select one of the 3 options:

| Quotation 1  | Quotation 2                    | Quotation 3 |
|--|--------------------------------|-------------|
| Company Name:<br>CA                                      |                                |             |
| Company Email:<br>CA@C                                   |                                |             |
| Company Quote:<br><a href="#">DocExample1.txt</a>        |                                |             |
| Preferred vendor:<br><input checked="" type="checkbox"/> |                                |             |
| Confirm selected quote:<br><input type="checkbox"/>      |                                |             |
| <a href="#">Back</a>                                     | <a href="#">Reject Request</a> |             |

Table 165: Approve Vendor Screen 1

# Onboard Request #1

Please select one of the 3 options:

Quotation 1

Quotation 2

Quotation 3

Company Name:

CA

Company Email:

CA@C

Company Quote:

DocExample1.txt

Preferred vendor:



Confirm selected quote:



Continue

Back

Reject Request

Table 166: Approve Vendor Screen 2 Alternative

**REJECTION SUCCESSFUL**

Successfully **REJECTED** onboard request # 2.

Figure 107: Successfully Rejected request notification

1 DUE DILIGENCE CHECKLIST DETAILS

|  |                      |
|--|----------------------|
| Do you have a physical copy of the Due Diligence Checklist? <input type="checkbox"/> |                      |
| <a href="#">Back</a>   | <a href="#">Next</a> |

Table 167: Approve Vendor - Create screen - Step 1

2 FOUNDATIONAL DOCUMENTS

|   |   |
|---|---|
| Mutual Non-Disclosure Agreement or Confidentiality Agreement:             | <input type="checkbox"/>                                  |
| Basic Company Information:  | <input checked="" type="checkbox"/>                       |
| Ownership structure and affiliated entities:                              | <input type="checkbox"/>                                  |
| Income tax number:  | <input checked="" type="checkbox"/>                       |
| VAT number:   | <input type="checkbox"/>                                  |
| Company registration document:  | <input type="checkbox"/>                                  |
| Letters of good standing COID:  | <input type="checkbox"/>                                  |
| Has BEE?:   | <input checked="" type="checkbox"/>                       |
| BEE level:  | 1   |
| BEE Certificate:  | <input type="button" value="Choose File"/> No file chosen |
| Date of BEE validity:   | <input type="button" value="Choose a date*"/>             |
| Directors details and ID's:   | <input type="checkbox"/>                                  |
| Company Resolution Agreement:   | <input type="checkbox"/>                                  |
| <input type="button" value="Cancel"/> <input type="button" value="Next"/> |   |

Table 168: Approve Vendor - Create screen - Step 2

3 FINANCIALS

|   |                          |
|---|--------------------------|
| VAT Registration Certificate:   | <input type="checkbox"/> |
| Tax Clearance Certificate:  | <input type="checkbox"/> |
| Bank Stamped confirmation letter:                                       | <input type="checkbox"/> |
| <input type="button" value="Back"/> <input type="button" value="Next"/> |                          |

Table 169: Approve Vendor - Create screen - Step 3

4 SUB-CONTRACTING

|  |   |
|--|---|
| Name of sub-contractor:  | <input type="checkbox"/>  |
| Provide similar documents as for main supplier (In case of company):                                     | <input type="checkbox"/>  |
| Provide copy of ID, qualifications, accreditations and professional memberships (In case of individual): | <input type="checkbox"/>  |
| <a href="#" style="color: inherit; text-decoration: none;">Back</a>                                      | <a href="#" style="color: inherit; text-decoration: none;">Next</a> |

Table 170: Approve Vendor - Create screen - Step 4

5 INSURANCE

|   |   |
|---|---|
| General liability insurance:  | <input type="checkbox"/>  |
| Cyber insurance:  | <input type="checkbox"/>  |
| Professional indemnity insurance (if applicable):                   | <input type="checkbox"/>  |
| Other specific insurance required per service/industry:             | <input type="checkbox"/>  |
| <a href="#" style="color: inherit; text-decoration: none;">Back</a> | <a href="#" style="color: inherit; text-decoration: none;">Next</a> |

Table 171: Approve Vendor - Create screen - Step 5

5 INSURANCE

|   |   |
|---|---|
| General liability insurance:  | <input type="checkbox"/>  |
| Cyber insurance:  | <input type="checkbox"/>  |
| Professional indemnity insurance (if applicable):                   | <input type="checkbox"/>  |
| Other specific insurance required per service/industry:             | <input type="checkbox"/>  |
| <a href="#" style="color: inherit; text-decoration: none;">Back</a> | <a href="#" style="color: inherit; text-decoration: none;">Next</a> |

Table 172: Approve Vendor - Create screen - Step 6

### 7 INFORMATION SECURITY

|   |   |
|---|---|
| Business continuity plan:   | <input type="checkbox"/>                        |
| Disaster recovery plan:   | <input type="checkbox"/>                        |
| Protection of personal information by design:   | <input checked="" type="checkbox"/>             |
| Does the contract set out what personal data is used for what purpose?  | <input checked="" type="checkbox"/>             |
| Is the contracted partner a controller (C), joint controller (JC), processor (P) or sub-processor (SP)?                               | Controller (C) <input type="button" value="▼"/> |
| Depending on the controller/processor relationship, do you have a Data Processing Agreement or a Joint Controller Agreement in place? | <input checked="" type="checkbox"/>             |
| Does the contract highlight the importance of confidentiality?  | <input checked="" type="checkbox"/>             |
| Does the contract provide for audits and inspections?   | <input checked="" type="checkbox"/>             |
| Is it clear who is accountable and liable for different activities?   | <input checked="" type="checkbox"/>             |
| Is there a provision to cover third party processing of data?   | <input checked="" type="checkbox"/>             |
| Does a process exist for managing data when the contract ends?  | <input checked="" type="checkbox"/>             |
| Is the personal data that's being processed detailed in your and their 'Record of Processing Activities'?                             | <input checked="" type="checkbox"/>             |
| Does the supplier hold any form of certification for their processing activities?   | <input checked="" type="checkbox"/>             |
| History of data breaches and security incidents:  | <input checked="" type="checkbox"/>             |
| Site visits to assess security controls (if required):  | <input type="checkbox"/>                        |
| <input type="button" value="Back"/> <input type="button" value="Next"/>   |   |

Table 173: Approve Vendor - Create screen - Step 7

### 8 POLICY REVIEW

|   |                          |
|---|--------------------------|
| Information security or Data protection policy:                         | <input type="checkbox"/> |
| Privacy policy:   | <input type="checkbox"/> |
| Data retention and destruction policy:                                  | <input type="checkbox"/> |
| Anti-bribery and anti-corruption policy:                                | <input type="checkbox"/> |
| Ethics policy:  | <input type="checkbox"/> |
| Conflict of interest policy:  | <input type="checkbox"/> |
| Customer complaints policy:   | <input type="checkbox"/> |
| <input type="button" value="Back"/> <input type="button" value="Next"/> |                          |

Table 174: Approve Vendor - Create screen - Step 8

### 9 BUSINESS REFERENCES

|   |                          |
|---|--------------------------|
| Details of at least two business references:                            | <input type="checkbox"/> |
| <input type="button" value="Back"/> <input type="button" value="Next"/> |                          |

Table 175: Approve Vendor - Create screen - Step 9

10 Done

\*Please ensure that all details are correct\*

Back

Create

Table 176: Approve Vendor - Create screen - Step 10

## CREATE SUCCESSFUL

Successfully **ADDED** Due Diligence Checklist for **CC**.

Table 177: Created Due Diligence Checklist Success Notification

←

## Onboard Request #4

Please Approve or Reject the Request

Sole Supplier Request

|  |                          |
|--|--------------------------|
| Company Name:  | DummyVendor              |
| Company Email:   | DummyVendor@gmail.com    |
| Reason:  | Dummy date is the reason |
| Company Quote:   | No File Was Provided     |
| <a href="#" style="border: 1px solid green; padding: 5px 20px; color: green; margin-right: 10px;">Approve</a> <a href="#" style="border: 2px solid red; padding: 5px 20px; color: red;">Reject</a> |                          |

Table 178: Management Sole Supplier Approval

## REJECTION SUCCESSFUL

Successfully **REJECTED** onboard request # 1.

Table 179: Successful rejection notification

**APPROVE SUCCESSFUL**Successfully **APPROVED CA.**

Table 180: Successful Approved Onboard Request

## 6.10 Generate Due Diligence

1 / 2 | - 100% + | ☰ 🔍

### Vendor Due Diligence Checklist

| FOUNDATIONAL DOCUMENTS  |                          |
|---|--------------------------|
| Mutual Non-Disclosure Agreement or Confidentiality Agreement  | <input type="checkbox"/> |
| Basic Company Information (full legal name, address, physical location, website URL, trading name)        | <input type="checkbox"/> |
| Ownership structure and affiliated entities (Group structure)   | <input type="checkbox"/> |
| Income tax number   | <input type="checkbox"/> |
| VAT number  | <input type="checkbox"/> |
| Company registration document (CIPC)  | <input type="checkbox"/> |
| Letters of good standing COID   | <input type="checkbox"/> |
| B-BBEE certificate  | <input type="checkbox"/> |
| Directors details and ID  | <input type="checkbox"/> |
| Company resolution authorising the counter-party to enter into an agreement with Moyo and bind the entity | <input type="checkbox"/> |

| FINANCIALS                   |                          |
|------------------------------|--------------------------|
| VAT registration certificate | <input type="checkbox"/> |
| Tax clearance certificate    | <input type="checkbox"/> |

Table 181: Generated Due Diligence Checklist

## 6.11 Generate BEE Expiry Notification

**Vendor Review**

29/07/2023, 12:00 AM

Performance review for CA is needed.

Please review the request and respond accordingly.

 View

Figure 108: Sole Supplier Performance Review Notification

## 6.12 Generate Sole Supplier Performance Review Notification

**Vendor BEE renewal**

24/08/2023, 12:00 AM

3 Weeks before CA BEE expires!

Please review the request and respond accordingly.

 View

Figure 109: 3 weeks before BEE Expiry Notification

 Vendor BEE renewal

24/08/2023, 12:00 AM

**1 Week before CA BEE expires!**

Please review the request and respond accordingly.

 View

Figure 110: 1 Week Before BEE Expiry Notification

 Vendor BEE renewal

24/08/2023, 12:00 AM

**CA BEE expires has expired!**

Please review the request and respond accordingly.

 View

Figure 111: Vendor BEE Expired Notification

### 6.13 Update Approve Onboard Request

**1 DUE DILIGENCE CHECKLIST DETAILS**

Do you have a physical copy of the Due Diligence Checklist?

[Back](#) [Next](#)

Table 182: Approve Vendor - Update Screen - Step 1

**2 FOUNDATIONAL DOCUMENTS**

Mutual Non-Disclosure Agreement or Confidentiality Agreement:

Basic Company Information:

Ownership structure and affiliated entities:

Income tax number:

VAT number:

Company registration document:

Letters of good standing COID:

Has BEE?:

BEE level:

BEE Certificate:  No file chosen

Date of BEE validity:

Directors details and ID's:

Company Resolution Agreement:

[Cancel](#) [Next](#)

Table 183: Approve Vendor - Update Screen - Step 2

**3 FINANCIALS**

VAT Registration Certificate:

Tax Clearance Certificate:

Bank Stamped confirmation letter:

[Back](#) [Next](#)

Table 184: Approve Vendor - Update Screen - Step 3

**4 SUB-CONTRACTING**

|  |                          |
|--|--------------------------|
| Name of sub-contractor:  | <input type="checkbox"/> |
| Provide similar documents as for main supplier (In case of company):                                     | <input type="checkbox"/> |
| Provide copy of ID, qualifications, accreditations and professional memberships (In case of individual): | <input type="checkbox"/> |
| <b>Back</b>  | <b>Next</b>              |

Table 185: Approve Vendor - Update Screen - Step 4

**5 INSURANCE**

|   |                          |
|---|--------------------------|
| General liability insurance:                            | <input type="checkbox"/> |
| Cyber insurance:  | <input type="checkbox"/> |
| Professional indemnity insurance (if applicable):       | <input type="checkbox"/> |
| Other specific insurance required per service/industry: | <input type="checkbox"/> |
| <b>Back</b>   | <b>Next</b>              |

Table 186: Approve Vendor - Update Screen - Step 5

**6 LICENSES OR PROFESSIONAL ACCREDITATION**

|                                   |                          |
|-----------------------------------|--------------------------|
| Licenses required:                | <input type="checkbox"/> |
| Accreditation required:           | <input type="checkbox"/> |
| Professional membership required: | <input type="checkbox"/> |
| <b>Back</b>                       | <b>Next</b>              |

Table 187: Approve Vendor - Update Screen - Step 6

7 INFORMATION SECURITY

|   |   |
|---|---|
| Business continuity plan:   | <input type="checkbox"/>                        |
| Disaster recovery plan:   | <input type="checkbox"/>                        |
| Protection of personal information by design:   | <input checked="" type="checkbox"/>             |
| Does the contract set out what personal data is used for what purpose?  | <input checked="" type="checkbox"/>             |
| Is the contracted partner a controller (C), joint controller (JC), processor (P) or sub-processor (SP)?                               | Controller (C) <input type="button" value="▼"/> |
| Depending on the controller/processor relationship, do you have a Data Processing Agreement or a Joint Controller Agreement in place? | <input checked="" type="checkbox"/>             |
| Does the contract highlight the importance of confidentiality?  | <input checked="" type="checkbox"/>             |
| Does the contract provide for audits and inspections?   | <input checked="" type="checkbox"/>             |
| Is it clear who is accountable and liable for different activities?   | <input checked="" type="checkbox"/>             |
| Is there a provision to cover third party processing of data?   | <input checked="" type="checkbox"/>             |
| Does a process exist for managing data when the contract ends?  | <input checked="" type="checkbox"/>             |
| Is the personal data that's being processed detailed in your and their 'Record of Processing Activities'?                             | <input checked="" type="checkbox"/>             |
| Does the supplier hold any form of certification for their processing activities?   | <input checked="" type="checkbox"/>             |
| History of data breaches and security incidents:  | <input checked="" type="checkbox"/>             |
| Site visits to assess security controls (if required):  | <input type="checkbox"/>                        |
| <input type="button" value="Back"/> <input type="button" value="Next"/>   |   |

Table 188: Approve Vendor - Update Screen - Step 7

8 POLICY REVIEW

|   |                          |
|---|--------------------------|
| Information security or Data protection policy: | <input type="checkbox"/> |
| Privacy policy:                                 | <input type="checkbox"/> |
| Data retention and destruction policy:          | <input type="checkbox"/> |
| Anti-bribery and anti-corruption policy:        | <input type="checkbox"/> |
| Ethics policy:                                  | <input type="checkbox"/> |
| Conflict of interest policy:                    | <input type="checkbox"/> |
| Customer complaints policy:                     | <input type="checkbox"/> |

[Back](#) [Next](#)

Table 189: Approve Vendor - Update Screen - Step 8

9 BUSINESS REFERENCES

|  |                          |
|--|--------------------------|
| Details of at least two business references: | <input type="checkbox"/> |
|--|--------------------------|

[Back](#) [Next](#)

Table 190: Approve Vendor - Update Screen - Step 9

**\*Please ensure that all details are correct\***

[Back](#) [Update](#)

Table 191: Approve Vendor - Update Screen - Step 10



Table 192: Update Successful Notification

### 6.14 View Approved Or Pending Onboard Request

| Onboard Request Number | Selected Option | Request Type  | Total Quotes | Created By |
|------------------------|-----------------|---------------|--------------|------------|
| 1                      | Request pending | Sole Supplier | 1            | Admin      |

### 4.3 CONCLUSION

In conclusion this section provided an overview of how the inputs and outputs screen designs look and what principles need to be followed to achieve the desired screen designs of the Procion system.

## 5. TECHNICAL USE CASE NARRATIVES

### 5.1 INTRODUCTION

In this section we will discuss the Technical Use Case narratives for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system.

### 5.2 TECHNICAL USE CASE NARRATIVES

#### USE CASE 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36

| System Name: Procion System    |   |  |
|--------------------------------|---|--|
| Sub-System:                    |   |  |
| Author: EP Wonigkeit           | Date: 22/07/2023  | Version: 1.0.0   |
| Use case name:                 | View Profile  | USE CASE TYPE  |
| Use case id:                   | 1.5   |  |
| Priority:                      | High  | Business Requirements: <input type="checkbox"/><br>System Analysis: <input type="checkbox"/> |
| Source:                        | Moyo  | System Design: <input checked="" type="checkbox"/>   |
| Primary business actor         | User  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>This use case describes the event where the user wishes to view their profile.</p> <p>The use case begins when the user clicks on their profile picture on the top navigation bar. The admin will then select the “Profile” menu item on the dropdown menu bar which will load the “View User Profile” screen,</p> |  |

|                       |   |                        |  |
|-----------------------|---|------------------------|--|
|                       | which will allow the admin to view their information and edit their profile and information.                                  |                        |  |
| <b>Pre-condition:</b> | The admin must be logged in to the system<br><br>The top navigation bar must be loaded.<br><br>The home screen must be loaded |                        |  |
| <b>Trigger:</b>       | The admin clicks on the “Admin” option in the side navigation bar   |                        |  |
| <b>Typical course</b> | <b>Actor Action</b>   | <b>System Response</b> |  |
| <b>Of events:</b>     |   | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                       |   |                        | <p><b>Step 1:</b> The system will send a request from the Angular frontend to the Data service where the service will make a HTTP Get request to the .NET Core backend which makes use of Lamda LINQ which will create a SQL Read query to retrieve the following data:</p> <p>From the <b>Employee</b> entity:</p> <ul style="list-style-type: none"> <li>• EmployeeName</li> <li>• EmployeeSurname</li> <li>• Email</li> <li>• CellPhone_Num</li> </ul> <p>From the <b>User</b> entity using User_Id</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Profile_Picture</li> </ul> <p>From the <b>Role</b> entity using Role_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>From the <b>Branch</b> entity using Branch_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>From the <b>Department</b> entity using Department_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>From the <b>Mandate_Limit</b> entity using Mandate_ID</p> <ul style="list-style-type: none"> <li>• Amount</li> </ul> <p><b>[ALT]</b></p>   |
|  |  |  | <p><b>Step 2:</b> The system populate the Employee array with the results from the Lamda LINQ query</p>   |
|  |  |  | <p><b>Step 3:</b> The systems API (Procion) then sends the Employee array back to the Angular Front End where the system then populates the Employee array in the View employee Typescript component.</p> <p>The system then uses the Employee array to populate the table.</p>   |
|  |  |  | <p><b>Step 4:</b> The View User Profile screen loads containing the following components:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Round image element in the center of the screen.</li> <li>• Heading – containing the Employee Name and Surname.</li> <li>• Sub text – containing the employee role name.</li> <li>• Blue Button – “Edit”</li> <li>• Blue Button – “Update Password”</li> <li>• Table containing the following elements: <ul style="list-style-type: none"> <li>○ Row 1 <ul style="list-style-type: none"> <li>▪ Label – “Username:”</li> <li>▪ Label – containing the employees username</li> </ul> </li> <li>○ Row 2 <ul style="list-style-type: none"> <li>▪ Label – “Email:”</li> </ul> </li> </ul> </li> </ul> |

|                      |  |  |   |
|----------------------|--|--|---|
|                      |  |  | <ul style="list-style-type: none"> <li>▪ Label – containing the employees email</li> </ul> <ul style="list-style-type: none"> <li>○ Row 3           <ul style="list-style-type: none"> <li>▪ Label – “Phone Number:”</li> <li>▪ Label – containing the employees cellPhone_Num</li> </ul> </li> <li>○ Row 4           <ul style="list-style-type: none"> <li>▪ Label – “Mandate Limit:”</li> <li>▪ Label – containing the employees set mandate amount</li> </ul> </li> <li>○ Row 5           <ul style="list-style-type: none"> <li>▪ Label – “Department:”</li> <li>▪ Label – containing the employees department name</li> </ul> </li> <li>○ Row 6           <ul style="list-style-type: none"> <li>▪ Label – “Branch:”</li> <li>▪ Label – containing the employees branch name</li> </ul> </li> </ul> <p><b>[ALT]</b></p> |
| Alternative courses: |  |  | <p><b>ALT Step 1:</b> The system will send a request from the Angular frontend to the Data service where the service will make a HTTP Get request to the .NET Core backend which makes use of Lamda LINQ which will create a SQL Read query to retrieve the following data:</p> <p>From the <b>Admin</b> entity:</p> <ul style="list-style-type: none"> <li>• AdminName</li> <li>• AdminSurname</li> <li>• Email</li> <li>• CellPhone_Num</li> </ul>  |

|                        |  |
|------------------------|--|
|                        | <p>From the <b>User</b> entity using User_Id</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Profile_Picture</li> </ul> <p>From the <b>Role</b> entity using Role_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p><b>Proceed to Step 2.</b></p>  |
|                        | <p><b>ALT Step 4:</b> The View User Profile screen loads containing the following components:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Round image element in the center of the screen.</li> <li>• Heading – containing the Employee Name and Surname.</li> <li>• Sub text – containing the employee role name.</li> <li>• Blue Button – “Edit”</li> <li>• Blue Button – “Update Password”</li> <li>• Table containing the following elements: <ul style="list-style-type: none"> <li>○ Row 1 <ul style="list-style-type: none"> <li>▪ Label – “Username:”</li> <li>▪ Label – containing the employees username</li> </ul> </li> <li>○ Row 2 <ul style="list-style-type: none"> <li>▪ Label – “Email:”</li> <li>▪ Label – containing the employees email</li> </ul> </li> <li>○ Row 3 <ul style="list-style-type: none"> <li>▪ Label – “Phone Number:”</li> <li>▪ Label – containing the employees cellPhone_Num</li> </ul> </li> </ul> </li> </ul> <p><b>Use Case Terminates.</b></p> |
|                        | <p><b>ALT Step 5a:</b> The user clicks on the “Edit” button.</p> <p><b>Invoke Use Case 1.6 “Edit Profile”.</b></p>   |
|                        | <p><b>ALT Step 5a:</b> The admin clicks on the “Update Password” button.</p> <p><b>Invoke Use Case 1.4 “Update Password”.</b></p>  |
| <b>Conclusion:</b>     | The system displays the “View User Profile” screen   |
| <b>Post-condition:</b> | A table containing the users details can be viewed in the user details table.  |
| <b>Business rules</b>  | None   |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>The user requires a laptop or desktop.</li> <li>The user requires internet access.</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |

Table 193: 1.5 View Profile

| <b>System Name: Procion System</b>    |  |  |
|---------------------------------------|--|--|
| <b>Sub-System: Admin</b>              |  |  |
| Author: EP Wonigkeit                  | Date: 22/07/2023   | Version: 1.0.0                                 |
| <b>Use case name:</b>                 | Edit Profile   | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>                   | 1.6  | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>                      | High   | Extension: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   |  |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>This use case describes the event where the user wishes to edit their details stored on the Procion system.</p> <p>The use case begins when the user clicks on their profile picture on the top navigation bar. The admin will then select the “Profile” menu item on the dropdown menu bar which will load the “View User Profile” screen where the user will click on the “Edit” button. The “Update Profile” screen will load where the user will need to enter the following details.</p> <p>The use case begins when the admin clicks on the “Admin” option on the side navigation bar. The system will load the “View Admin” screen</p> |  |

|                                      | <p>where the admin will click on the “Edit” button. The “Edit Admin” screen where the admin will need to enter the following employee details:</p> <ul style="list-style-type: none"> <li>• Profile picture</li> <li>• Name</li> <li>• Surname</li> <li>• Email</li> <li>• Cell Phone Number</li> </ul> <p>The user will then click on the “Save” button and the use case concludes once the user details are updated in the <b>Employee (/Admin)</b> and <b>User</b> entities in the database corresponding to the role selected.</p> |   |
|--------------------------------------|--|---|
| <b>Pre-condition:</b>                | <p>The user must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The “View User Profile” screen must be loaded</p> <p>The user clicked the “Edit” button</p>   |   |
| <b>Trigger:</b>                      | <p>The user navigates to the “Edit Profile” screen by clicking on the “Edit” button</p>  |   |
| <b>Typical course<br/>Of events:</b> |  |   |
| <b>Actor Action</b>                  | <b>System Response</b>   |   |
| <b>Manual Action</b>                 | <b>Automated Action</b>  |   |
|                                      |  | <p><b>Step 1:</b> The system loads the “Edit User Profile” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Circle image container</li> <li>• Round input with Pencil icon</li> <li>• “Name” label with its corresponding input, hint sub-label, and error sub-label.</li> <li>• “Surname” label with its corresponding input, hint sub-label, and error sub-label.</li> <li>• “Email” label with its corresponding input, and error sub-label.</li> <li>• “Phone Number” label with its corresponding input, hint sub-label, and error sub-label.</li> <li>• A green “Save” button in the centre of the card.</li> </ul> |

## Iteration 5: Design &amp; Development– Team 11

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• A blue “Cancel” button in the centre of the card left of the “Save” button.</li> </ul> <p>The “Save” button is disabled until all the input fields passes validation. The error sub-labels are hidden by default and displayed when a certain validation failed. The hint sub-label is displayed by default and hidden once an error message is to be displayed.</p>  |
|  |  |  | <p><b>Step 2:</b> The system retrieves the selected users user_Id which is passed to the page through the page URL which is then sent to the Data service where it will send an http get request to the .Net Core controller which will us an Entity Framework Get method to retrieve the details from the <b>Employee</b>, <b>User</b>, and <b>Role Entities</b> where the User_Id matches the one that is retrieved.</p> <p>The retrieved information is sent back as an <b>Employee</b> object to the Data service where it is then retrieved by the Angular Front End and read into any <b>Employee</b> array.</p> <p>The information in the array is then used to pre-populate the input fields form values as follows:</p> <ul style="list-style-type: none"> <li>• “Name” input with EmployeeName</li> <li>• “Surname” input with EmployeeSurname</li> <li>• “Email” input with Email</li> <li>• “Phone Number” input with CellPhone_Num</li> </ul> <p>The system prompts the user to enter the required details.</p> <p><b>[ALT]</b></p> |

|  |   |  |  |
|--|---|--|--|
|  | <b>Step 3:</b> The user clicks on the profile picture edit button.                                  |  | <b>Step 4:</b> The system opens the users file explorer and prompts them to select a picture.  |
|  | <b>Step 5:</b> The user selects the picture that they wish to use.                                  |  | <p><b>Step 6:</b> The system captures the picture and send it to the Cropper Component.</p> <p>The system displays the cropper modal with the following components:</p> <ul style="list-style-type: none"> <li>• Image cropper element containing the image.</li> <li>• Green “Save” Button</li> <li>• Blue “Cancel” Button.</li> </ul> <p>The system prompts the user to crop the image to their liking.</p>  |
|  | <p><b>Step 7:</b> The user crops the image and clicks on the “Save” button.</p> <p><b>[ALT]</b></p> |  | <p><b>Step 8:</b> The system crops the image and sends it back to the “Edit Profile” screen as a base64 string where it is then stored in the User object.</p>   |
|  | <b>Step 9:</b> The user enters their details in their corresponding inputs.                         |  | <p><b>Step 10:</b> The system validates the inputted details from the Angular form by checking:</p> <ul style="list-style-type: none"> <li>• That the “Name” input box is not null, the entered name is between 2 to 32 characters, the name does not contain numbers.</li> <li>• That the “Surname” input box is not null, the entered surname is between 2 to 32 characters, the surname does not contain numbers.</li> <li>• That the “Email” input box is not null, the entered email is no longer than 32 characters, the email is of the correct format.</li> <li>• That the “Phone Number” input box is not null, the entered number is 12</li> </ul> |

## Iteration 5: Design &amp; Development– Team 11

|  |   |  |   |
|--|---|--|---|
|  |   |  | <p>characters long, the number does not contain any letters or characters other than a space.</p> <p>The inputs are all validated by the Angular Front End when the user clicks and leaves an input field.</p> <p>The “Save” button is only activated once all the inputs are valid.</p> <p><b>[ALT]</b></p>  |
|  | <p><b>Step 11:</b> The user clicks on the “Save” button</p> <p><b>[ALT]</b></p> |  | <p><b>Step 12:</b> The system validates the input fields as in step 5.</p> <p>The system will capture the Angular form inputs to populate the Employee, User, Role objects with the values from the inputs.</p> <p>The system then creates the username, which is done by concatenating the name, surname, and middle 3 numbers of the entered cell phone number. The profile picture file is given the path to the default picture.</p> <p>The system will send to username to the Data service where it will send an http get request to the .Net Core controller which will use an Entity Framework get method to check if the entered user details does not already exist in the database.</p> <p>The Employee and User objects, which are a combination of all the others are then sent to the Data service where it will send an http post request which will use an Entity Framework Add method to store the Admin and User details by auto generating the EmployeeID and User_Id by incrementing the last added ID.</p> |

|                           |  |  |   |
|---------------------------|--|--|---|
|                           |  |  | <p>The submitted data is stored in the following attributes in the <b>User Entity</b> as new record:</p> <ul style="list-style-type: none"> <li>• User_Id</li> <li>• Role_ID</li> <li>• Username</li> <li>• Password</li> <li>• Profile_picture</li> </ul> <p>Once the <b>User Entity</b> is added the remaining data is stored in the following attributes in the <b>Employee Entity</b> as new record:</p> <ul style="list-style-type: none"> <li>• EmployeeID</li> <li>• User_Id</li> <li>• EmployeeName</li> <li>• EmployeeSurname</li> <li>• CellPhone_Num</li> <li>• Email</li> </ul> <p>The correct Foreign Keys are retrieved from their primary table by comparing the entered name with the one stored in the database.</p> <p><b>[ALT]</b></p> |
|                           |  |  | <p><b>Step 13:</b> The system will display a success card notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “UPDATE SUCCESSFUL” header</li> <li>• “Your profile has been UPDATED successfully!” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p>   |
|                           |  |  | <p><b>Step 14:</b> The system will invoke <b>Use Case 1.5 “View Profile”</b></p>  |
| <b>Alternate courses:</b> | <p><b>ALT Step 2:</b> The system retrieves the selected users user_Id which is passed to the page through the page URL which is then sent to the Data service where it will send an http get request to the .Net Core controller which will use an Entity Framework Get method to retrieve the</p> |  |   |

details from the **Admin**, **User**, and **Role Entities** where the User\_Id matches the one that is retrieved.

The retrieved information is sent back as an **Admin** object to the Data service where it is then retrieved by the Angular Front End and read into any **Admin** array.

The information in the array is then used to pre-populate the input fields form values as follows:

- “Name” input with AdminName
- “Surname” input with AdminSurname
- “Email” input with Email
- “Phone Number” input with CellPhone\_Num

The system prompts the user to enter the required details.

### **Proceed to Step 3.**

**ALT Step 7:** The user clicks the blue “Cancel” button.

### **Proceed to Step 3**

**ALT Step 10:** The system validates the inputted information using the provided validation inside the Angular form.

The system displays the following:

- “Name is required” label underneath the Name input if the input is empty.
- “Limit exceeded” label underneath the Name input if the entered value is longer than 32 characters.
- “Name must be 3 to 32 characters long” label underneath the Name input if the entered value is shorter than 2 characters.
- “Name cannot contain numbers” label underneath the Name input if the entered value contains numbers.
- “Surname is required” label underneath the Surname input if the input is empty.
- “Limit exceeded” label underneath the Surname input if the entered value is longer than 32 characters.
- “Surname must be 3 to 32 characters long” label underneath the Surname input if the entered value is shorter than 2 characters.
- “Surname cannot contain numbers” label underneath the Surname input if the entered value contains numbers.
- “Email is required” label underneath the Email input when if the input is empty.
- “Limit exceeded” label underneath the Email input if the entered value is longer than 32 characters.
- “Enter a valid email” label underneath the Email input if the entered value is not in the correct email format.

|                        |   |
|------------------------|---|
|                        | <ul style="list-style-type: none"> <li>“Phone Number is required” label underneath the Phone Number input if the input is empty.</li> <li>“Limit exceeded” label underneath the Phone Number input if the entered value is longer than 12 characters.</li> <li>“Phone Number must be at least 12 characters including spaces” label underneath the Phone Number input if the entered value is shorter than 2 characters.</li> <li>“Please enter a valid phone number” label underneath the Phone input if the entered value contains letters or symbols other than a space</li> </ul> |
|                        | <p><b>ALT Step 11:</b> The user clicks the blue “Cancel” button.</p> <p>The system terminates the use case and invokes <b>Use Case 1.5 “View Profile”</b></p>   |
|                        | <p><b>ALT Step 12:</b> There already exists a user on the database with the same username.</p> <p>The system notifies the admin by displaying a card notification containing the following components:</p> <ul style="list-style-type: none"> <li>“ERROR: User Exists” header</li> <li>“A user with the username: (entered username) ALREADY EXISTS!” body text</li> </ul> <p>The text disappears after 1750 seconds leaving the user on the page to edit the entered data.</p> <p>Return to <b>Step 9.</b></p>   |
| <b>Post-condition:</b> | <ul style="list-style-type: none"> <li>The Employee (/Admin) and User details have been successfully updated in the <b>Employee (/Admin) and User Tables</b>.</li> <li>The database retains its data integrity after the Employee (/Admin) and User has been updated.</li> </ul>  |

Table 194: 1.6 Edit Profile

| <b>System Name: Procion System</b><br><b>Sub-System:</b> |                    |   |
|--|--------------------|---|
| Author: EP Wonigkeit                                     | Date: 10/05/2023   | Version: 1.0.0                                  |
| <b>Use case name:</b>                                    | View Notifications | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>                                      | 1.8                |   |
| <b>Priority:</b>   | High               | Business Requirements: <input type="checkbox"/> |

|                                       |  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Source:</b>                        | Moyo   | System Analysis:       | <input type="checkbox"/>   |
|                                       |  | System Design:         | <input checked="" type="checkbox"/>  |
| <b>Primary business actor</b>         | Admin  |                        |  |
| <b>Primary system actor</b>           | None   |                        |  |
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | <p>This use case describes the event where the user wishes to view their notifications on the system.</p> <p>The use case begins when the user clicks on the Bell icon option on the top navigation bar. The system will load the “View Notifications” screen. The user will be able to cycle through their notification pages and go to the relevant pages.</p> |                        |  |
| <b>Pre-condition:</b>                 | <p>The user must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The home screen must be loaded</p>  |                        |  |
| <b>Trigger:</b>                       | The user clicks on the Bell icon option on the top navigation bar.   |                        |  |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       |  |                        | <p><b>Step 1:</b> The system will send a request from the Angular frontend to the Data service where the service will make a HTTP Get request to the .NET Core backend which makes use of Entity Framework which will create a SQL Read query to retrieve the following data:</p> <p>From the <b>Notification</b> entity:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Send_Date</li> </ul> <p>From the <b>Notification_Type</b> entity using Notification_Type_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>From the <b>User</b> entity using User_ID:</p> <ul style="list-style-type: none"> <li>• Username</li> </ul> <p>From the <b>Role</b> entity using Role_ID:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul>  |
|  |  |  | <p><b>Step 2:</b> The system separates the retrieved notification by their Notification_Types (either Vendor, Inventory, or Procurement) and populates the different Notification arrays with the results from the Entity Framework query</p>  |
|  |  |  | <p><b>Step 3:</b> The systems API (Procion) then sends the Notification arrays back to the Angular Front End where the system then populates the Notification arrays in the View Notification Typescript component.</p> <p>The system then uses the Notification arrays to populate the notification tables.</p>   |
|  |  |  | <p><b>Step 4:</b> The View Notification screen loads containing the following components:</p> <ul style="list-style-type: none"> <li>• Angular Material Tabs with the following label headers:             <ul style="list-style-type: none"> <li>◦ Vendor</li> <li>◦ Inventory</li> <li>◦ Procurement</li> </ul> </li> </ul> <p>Each tab has a table with specific notifications with the following general design:</p> <ul style="list-style-type: none"> <li>• Card in the center of the screen</li> <li>• Card Title:</li> </ul> |

|                           |  |  |  |
|---------------------------|--|--|--|
|                           |  |  | <ul style="list-style-type: none"> <li>○ Unread Message icon</li> <li>○ Heading – containing the Notification_Type name</li> <li>● Card Body:           <ul style="list-style-type: none"> <li>○ Paragraph – containing the Notification name.</li> <li>○ Paragraph – containing the Notification_Type description.</li> </ul> </li> <li>● Blue “View” button in the right corner with an eye icon on the left.</li> </ul> <p><b>[ALT]</b></p> |
| <b>Alternate Courses:</b> | <p><b>ALT Step 4:</b> The user has an active delegation request and has temporary access assigned.</p> <p>The View Notification screen loads containing the following components:</p> <ul style="list-style-type: none"> <li>● Angular Material Tabs with the following label headers:           <ul style="list-style-type: none"> <li>○ My Notifications</li> <li>○ Delegating User Notifications</li> </ul> </li> <li>● Angular Material Tabs with the following label headers:           <ul style="list-style-type: none"> <li>○ Vendor</li> <li>○ Inventory</li> <li>○ Procurement</li> </ul> </li> </ul> <p>Each tab has a table with specific notifications with the following general design:</p> <ul style="list-style-type: none"> <li>● Card in the center of the screen</li> <li>● Card Title:           <ul style="list-style-type: none"> <li>○ Unread Message icon</li> <li>○ Heading – containing the Notification_Type name</li> </ul> </li> <li>● Card Body:           <ul style="list-style-type: none"> <li>○ Paragraph – containing the Notification name.</li> <li>○ Paragraph – containing the Notification_Type description</li> <li>○ Blue “View” button in the right corner with an eye icon on the left</li> </ul> </li> </ul> <p>The system splits the notifications of the user by displaying the main users notifications under the “My Notifications” tab and the delegating user’s notifications under the “Delegating User Notifications” tab.</p> |  |  |

|  |  |
|--|--|
|  | <p><b>ALT Step 5a:</b> The user clicks on the “View” button on a Vendor Notification.</p> <p><b>Invoke Use Case 6.2 “View Vendor onboard Request”.</b></p>   |
|  | <p><b>ALT Step 5b:</b> The user clicks on the “View” button on a Vendor Notification.</p> <p><b>Invoke Use Case 6.7 “View Vendor”.</b></p>                   |
|  | <p><b>ALT Step 5c:</b> The user clicks on the “View” button on an Inventory Notification.</p> <p><b>Invoke Use Case 5.2 “View Consumables”.</b></p>          |
|  | <p><b>ALT Step 5d:</b> The user clicks on the “View” button on a Procurement Notification.</p> <p><b>Invoke Use Case 3.2 “View Procurement Request”.</b></p> |
| <b>Conclusion:</b>                                   | The system displays the “View Notification” screen   |
| <b>Post-condition:</b>                               | A list of notifications is viewed in the “Notification list” card table which is retrieved from the <b>Notification</b> and <b>Notification_Type</b> tables. |
| <b>Business rules</b>                                |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The user requires a laptop or desktop.</li> <li>• The user requires internet access.</li> </ul>                     |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 195: 1.8 View Notifications

| <b>System Name: Procion System</b><br><b>Sub-System:</b> |                              |   |
|--|------------------------------|---|
| Author: EP Wonigkeit                                     | Date: 22/07/2023             | Version: 1.0.0                                  |
| <b>Use case name:</b>                                    | View Delegation of Authority | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>                                      | 2.17                         | Business Requirements: <input type="checkbox"/> |
| <b>Priority:</b>   | High                         |   |

|                                       |   |  |   |
|---------------------------------------|---|--|---|
| <b>Source:</b>                        | Moyo  | System Analysis: <input type="checkbox"/>          |   |
|                                       |   | System Design: <input checked="" type="checkbox"/> |   |
| <b>Primary business actor</b>         | Admin   |  |   |
| <b>Primary system actor</b>           | None  |  |   |
| <b>Other participating actors:</b>    | None  |  |   |
| <b>Other interested stakeholders:</b> | None  |  |   |
| <b>Description:</b>                   | <p>This use case describes the event where the admin wishes to view all the delegation of authorities on the system.</p> <p>The use case begins when the admin clicks on the “Administration” option on the top navigation bar. The admin will then select the “Delegation of Authority” menu item on the side navigation bar which will load the “View Delegation” screen, which will allow the admin to view all the delegation of authorities on the system, edit existing delegations, delete delegations from the system, search for delegations, and revoke access from specific delegations.</p> |  |   |
| <b>Pre-condition:</b>                 | <p>The admin must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The home screen must be loaded</p> <p>The View Employee page must be loaded</p> <p>The side navigation bar must be loaded</p>   |  |   |
| <b>Trigger:</b>                       | The admin clicks on the “Delegation of Authority” option in the side navigation bar   |  |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>   | <b>System Response</b>                             |   |
|                                       |   | <b>Manual Action</b>                               | <b>Automated Action</b>   |
|                                       |   |  | <b>Step 1:</b> The system will send a request from the Angular frontend to the Data service where the service will make a HTTP Get request to the .NET Core backend which makes |

|  |  |  |
|--|--|--|
|  |  | <p>use of Lamda LINQ which will create a SQL Read query to retrieve the following data:</p> <p>From the <b>Delegation_of_Authority</b> entity:</p> <ul style="list-style-type: none"> <li>• DelegatingParty</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation File</li> </ul> <p>From the <b>User</b> entity using User_Id</p> <ul style="list-style-type: none"> <li>• Username</li> </ul> <p>From the <b>Delegation_Status</b> entity using DelegationStatus_ID</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |
|  |  | <p><b>Step 2:</b> The system populate the Delegation array with the results from the Lamda LINQ query</p>  |
|  |  | <p><b>Step 3:</b> The systems API (Procion) then sends the Delegation array back to the Angular Front End where the system then populates the Deleagaiton array in the View Delegation Typescript component.</p> <p>The system then uses the Delegation array to populate the table.</p>   |
|  |  | <p><b>Step 4:</b> The View Delegations screen loads containing the following components:</p> <ul style="list-style-type: none"> <li>• Heading – “Manage Delegation of Authority”</li> <li>• Search input with <ul style="list-style-type: none"> <li>◦ Label – “Search”</li> <li>◦ Search Icon</li> </ul> </li> <li>• Loading spinner</li> <li>• Table with the following column headers: <ul style="list-style-type: none"> <li>◦ No</li> <li>◦ Delegating Party</li> </ul> </li> </ul>   |

|                    |  |  |  |
|--------------------|--|--|--|
|                    |  |  | <ul style="list-style-type: none"> <li>○ Delegate</li> <li>○ Start Date</li> <li>○ End Date</li> <li>○ Delegation Form</li> <li>○ Status</li> <li>● Three columns for buttons           <ul style="list-style-type: none"> <li>○ Green “Edit” button containing a pencil icon on the right.</li> <li>○ Light Blue “Delete” button containing a dust bin icon on the right.</li> <li>○ Light Blue “Revoke” button</li> </ul> </li> </ul> <p>The loading spinner is displayed by default when entering the page and is hidden once the data is retrieved from the API.</p> <p>The delete and revoke buttons colour turns red when hovered over.</p> <p>The icon of the buttons moves to the center of the button and the text is hidden when hovered over.</p> <p>The table is hidden by default and is displayed once the data is retrieved from the API and the loading spinner is hidden.</p> |
|                    | <p><b>ALT Step 5a:</b> The admin clicks on the “Edit” button.</p> <p><b>Invoke Use Case 2.19 “Edit Delegation of Authority”.</b></p> <p><b>ALT Step 5b:</b> The admin clicks on the “Delete” button.</p> <p><b>Invoke Use Case 2.35 “Delete Delegation of Authority”.</b></p> <p><b>ALT Step 5c:</b> The admin clicks on the “Revoke” button.</p> <p><b>Invoke Use Case 2.20 “Revoke Delegation of Authority”.</b></p> |  |  |
| <b>Conclusion:</b> | The system displays the “View Delegation” screen   |  |  |

|  |  |
|--|--|
| <b>Post-condition:</b>                               | A list of delegations is viewed in the “Delegation list” table which is retrieved from the <b>Delegation_Of_Authority, User and Delegation_Status</b> table. |
| <b>Business rules</b>                                | Only the admin can view the delegations on the system.   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• The admin requires a laptop or desktop.</li> <li>• The admin requires internet access.</li> </ul>                   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 196: 2.17 View Delegation of Authority

| <b>System Name: Procion System</b>    |  |  |
|---------------------------------------|--|--|
| <b>Sub-System: Admin</b>              |  |  |
| Author: EP Wonigkeit                  | Date: 22/07/2023   | Version: 1.0.0                                 |
| <b>Use case name:</b>                 | Assign Delegation of Authority   | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>                   | 2.18   | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>                      | High   | Extension: <input checked="" type="checkbox"/> |
| <b>Source:</b>                        | Moyo   |  |
| <b>Primary business actor</b>         | Admin  |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>This use case describes the event where the admin would like to delegate a user's authority to another user for a set period of time on the Procion system.</p> <p>The use case begins when the admin clicks on the “Administration” option on the top navigation bar. The system will load the “View</p> |  |

|                                      | <p>Employees” screen where the admin will click on the “Delegate Authority” button of the user whose authority needs to be delegated. The “Assign Delegation” screen will load where the admin will need to enter the following details:</p> <ul style="list-style-type: none"> <li>• Delegate (Name of the employee receiving the authority)</li> <li>• Start date</li> <li>• End date</li> <li>• Delegation document</li> </ul> <p>The admin will then click on the “Save” button and the use case concludes once the new delegation information is added to the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> entities in the database.</p>  |  |                        |  |  |                      |                         |  |  |  |
|--------------------------------------|--|--|------------------------|--|--|----------------------|-------------------------|--|--|--|
| <b>Pre-condition:</b>                | <p>The admin must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The “View Employees” screen must be loaded</p> <p>The admin clicked the “Delegate Authority” button</p>  |  |                        |  |  |                      |                         |  |  |  |
| <b>Trigger:</b>                      | <p>The admin navigates to the “Assign Delegation” screen by clicking on the “Delegate Authority” button</p>  |  |                        |  |  |                      |                         |  |  |  |
| <b>Typical course<br/>Of events:</b> | <table border="1"> <thead> <tr> <th><b>Actor Action</b></th><th colspan="2"><b>System Response</b></th></tr> <tr> <th></th><th><b>Manual Action</b></th><th><b>Automated Action</b></th></tr> </thead> <tbody> <tr> <td></td><td></td><td> <p><b>Step 1:</b> The system loads the “Assign Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Assign Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> </ul> </td></tr> </tbody> </table> | <b>Actor Action</b>  | <b>System Response</b> |  |  | <b>Manual Action</b> | <b>Automated Action</b> |  |  | <p><b>Step 1:</b> The system loads the “Assign Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Assign Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> </ul> |
| <b>Actor Action</b>                  | <b>System Response</b>   |  |                        |  |  |                      |                         |  |  |  |
|                                      | <b>Manual Action</b>   | <b>Automated Action</b>  |                        |  |  |                      |                         |  |  |  |
|                                      |  | <p><b>Step 1:</b> The system loads the “Assign Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Assign Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> </ul> |                        |  |  |                      |                         |  |  |  |

|  |  |   |
|--|--|---|
|  |  | <ul style="list-style-type: none"> <li>“Delegation Document” label with its corresponding file upload input.</li> <li>A green “Save” button in the centre of the card.</li> <li>A blue “Cancel” button in the centre of the card left of the “Save” button.</li> </ul> <p>The “Save” button is disabled until all the input fields passes validation. The error sub-labels are hidden by default and displayed when a certain validation failed. The hint sub-label is displayed by default and hidden once an error message is to be displayed.</p> <p>The system prompts the user to enter the required details.</p>  |
|  | <p><b>Step 2:</b> The admin enters the required details in their corresponding inputs.</p> | <p><b>Step 3:</b> The system validates the inputted details from the Angular form by checking:</p> <ul style="list-style-type: none"> <li>That the “Delegate” input box is not null, the entered name is between 2 to 32 characters. <ul style="list-style-type: none"> <li>The delegate input box makes use of autocomplete to retrieve the usernames that is similar to the information being entered.</li> </ul> </li> <li>That the “Date Range” input box is not null, start date is not invalid, end date is not invalid.</li> <li>That the “Delegation Document” input box is not null, only a pdf document is uploaded.</li> </ul> <p>The inputs are all validated by the Angular Front End when the admin clicks and leaves an input field.</p> |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <p>The “Save” button is only activated once all the inputs are valid.</p> <p><b>[ALT]</b></p>  |
|  | <p><b>Step 4:</b> The admin clicks on the “Save” button</p> <p><b>[ALT]</b></p> |  | <p><b>Step 5:</b> The system validates the input fields as in step 3.</p> <p>The system will capture the file from the File input in a file object and the name of the filename variable.</p> <p>The system send the file object and filename variable to the Data service where it will be appended into a form object which is then sent with an http post request to the .Net Core controller.</p> <p>The system will retrieve the file and file name from the Form object, where it will create a folder with the name in the file name variable which is stored in a folder named “AuthorityDelegations” in the “Files” folder.</p> <p>The system send the file path back to the frontend which is then saved in the Delegation_of_Authority object.</p> <p>The system will capture the Angular form inputs to populate the Delegation_of_Authority, User, Admin, Delegation_Status objects with the values from the inputs, the start and end dates are transformed using a DatePipe into the “MMM d, y, h:mm:ss a” format.</p> <p>The Delegation_of_Authority object, which are a combination of all the others are then sent to the Data service where it will send an http post request which will use an Entity Framework Add method to store the Delegation details by auto</p> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>generating the Delegation_ID by incrementing the last added ID.</p> <p>The submitted data is stored in the following attributes in the <b>Delegation_of_Authority Entity</b> as new record:</p> <ul style="list-style-type: none"> <li>• Delegation_ID</li> <li>• User_Id</li> <li>• Admin_ID</li> <li>• Delegation_Status_ID</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation_Document</li> <li>• DelegatingParty</li> </ul> <p>The correct Foreign Keys are retrieved from their primary table by comparing the entered name with the one stored in the database.</p> <p>The system then assigns the delegation ID in the <b>Temporary_Access</b> object in the frontend, which is then sent to the Data service where it will send an http post request which will use an Entity Framework Add method to store the Temporary Access details by auto generating the Temp_Access_ID by incrementing the last added ID.</p> <p>The submitted data is stored in the following attributes in the <b>Temporary_Access Entity</b> as new record:</p> <ul style="list-style-type: none"> <li>• Temporary_Access_ID</li> <li>• Delegation_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>The correct Foreign Keys are retrieved from their primary table by comparing the entered id with the one stored in the database.</p> |
|--|--|--|--|

|   |  |  |
|---|--|--|
|   |  | <p><b>Step 6:</b> The system initiates the recurring function in the backend which will execute every day at 00:00 and calls the CheckDelegation function in the Data service which will call the method with the same name in the Home controller that will check the from_Date and to_Date attributes and compare it to the current date and change the status accordingly.</p>  |
|   |  | <p><b>Step 7:</b> The system transforms the save button into a check mark and hides the cancel button.</p> <p>The system notifies the admin by displaying a card notification containing the following components:</p> <ul style="list-style-type: none"> <li>• “CREATE SUCCESSFUL” header</li> <li>• “The Request No (created delegation ID) has been CREATED successfully!” body text</li> </ul> <p>The text disappears after 1750 seconds leaving the user on the page to edit the entered data</p> <p>The system will invoke <b>of Use Case 2.17 “View Delegation”</b></p> |
| <b>Alternate courses:</b>                                     |  | <p><b>ALT Step 3:</b> The system validates the inputted information using the provided validation inside the Angular form.</p> <p>The system displays the following:</p> <ul style="list-style-type: none"> <li>• “Name is required” label underneath the Name input if the input is empty.</li> <li>• “Limit exceeded” label underneath the Name input if the entered value is longer than 32 characters.</li> <li>• “Name must be 2 to 32 characters long” label underneath the Name input if the entered value is shorter than 2 characters.</li> </ul>                     |
| <b>ALT Step 4:</b> The admin clicks the blue “Cancel” button. |  |  |

|                        |   |
|------------------------|---|
|                        | The system terminates the use case and invokes <b>Use Case 2.17 “View Delegation”</b>   |
|                        |   |
| <b>Post-condition:</b> | <ul style="list-style-type: none"> <li>The <b>Delegation_of_Authority</b> details has been created and added as an object inside the <b>Delegation_of_Authority entity</b> successfully.</li> <li>The <b>Temporary_Access</b> details has been created and added as an object inside the <b>Temporary_Access entity</b> successfully.</li> <li>The database retains its data integrity after the new <b>Delegation_of_Authority</b> and <b>Temporary_Access</b> has been added</li> </ul> |

Table 197: 2.18 Assign Delegation of Authority

| <b>System Name: Procion System</b><br><b>Sub-System: Admin</b> |  |  |
|--|--|--|
| Author: EP Wonigkeit   | Date: 22/07/2023   | Version: 1.0.0                                 |
| <b>Use case name:</b>  | Edit Delegation of Authority   | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>  | 2.19   | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>   | High   | Extension: <input checked="" type="checkbox"/> |
| <b>Source:</b>   | Moyo   |  |
| <b>Primary business actor</b>                                  | Admin  |  |
| <b>Primary system actor</b>                                    | None   |  |
| <b>Other participating actors:</b>                             | None   |  |
| <b>Other interested stakeholders:</b>                          | None   |  |
| <b>Description:</b>  | <p>This use case describes the event where the admin wishes to edit a Delegation entry on the Procion system.</p> <p>The use case begins when the admin clicks on the “Administration” option on the top navigation bar. The admin will then select the “Delegation of Authority” menu item on the side navigation bar which will load the “View Delegation” screen, where the admin will click on the</p> |  |

|                                      | <p>“Edit” button. The “Edit Delegation” screen will load where the admin will need to enter the following employee details:</p> <ul style="list-style-type: none"> <li>• Delegate (Name of the employee receiving the authority)</li> <li>• Start date</li> <li>• End date</li> <li>• Delegation document</li> </ul> <p>The admin will then click on the “Save” button and the use case concludes once the new delegation information is updated in the <b>Delegation_Of_Authority</b> and <b>Temporary_Access</b> entities in the database.</p>  |                         |                        |  |                      |                         |  |   |  |
|--------------------------------------|---|-------------------------|------------------------|--|----------------------|-------------------------|--|---|--|
| <b>Pre-condition:</b>                | <p>The admin must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The “View Delegations” screen must be loaded</p> <p>The admin clicked the “Edit” button</p>   |                         |                        |  |                      |                         |  |   |  |
| <b>Trigger:</b>                      | <p>The admin navigates to the “Edit Delegation” screen by clicking on the “Edit” button</p>   |                         |                        |  |                      |                         |  |   |  |
| <b>Typical course<br/>Of events:</b> | <table border="1"> <thead> <tr> <th rowspan="2"><b>Actor Action</b></th> <th colspan="2"><b>System Response</b></th> </tr> <tr> <th><b>Manual Action</b></th> <th><b>Automated Action</b></th> </tr> </thead> <tbody> <tr> <td></td><td colspan="2"> <p><b>Step 1:</b> The system loads the “Edit Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Edit Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> <li>• “Delegation Document” label with its corresponding file upload input.</li> </ul> </td></tr> </tbody> </table> | <b>Actor Action</b>     | <b>System Response</b> |  | <b>Manual Action</b> | <b>Automated Action</b> |  | <p><b>Step 1:</b> The system loads the “Edit Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Edit Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> <li>• “Delegation Document” label with its corresponding file upload input.</li> </ul> |  |
| <b>Actor Action</b>                  | <b>System Response</b>  |                         |                        |  |                      |                         |  |   |  |
|                                      | <b>Manual Action</b>  | <b>Automated Action</b> |                        |  |                      |                         |  |   |  |
|                                      | <p><b>Step 1:</b> The system loads the “Edit Delegation” screen with the following elements:</p> <ul style="list-style-type: none"> <li>• Card in the centre of the screen</li> <li>• Card Header “Edit Delegation” in the centre of the card</li> <li>• “Delegating Party” label with its corresponding input that is read only.</li> <li>• “Delegate” label with its corresponding input, hint sub-label.</li> <li>• “Date Range” label with its corresponding date picker input.</li> <li>• “Delegation Document” label with its corresponding file upload input.</li> </ul>   |                         |                        |  |                      |                         |  |   |  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• A green “Save” button in the centre of the card.</li> <li>• A blue “Cancel” button in the centre of the card left of the “Save” button.</li> </ul> <p>The “Save” button is disabled until all the input fields passes validation. The error sub-labels are hidden by default and displayed when a certain validation failed. The hint sub-label is displayed by default and hidden once an error message is to be displayed.</p>  |
|  |  |  | <p><b>Step 2:</b> The system retrieves the selected delegation’s Delegation_ID which is passed to the page through the page URL which is then sent to the Data service where it will send an http get request to the .Net Core controller which will use a Entity Framework Get method to retrieve the details from <b>Delegation_Of_Authority, User and Delegation_Status</b> where the Delegation_ID matches the one that is retrieved.</p> <p>The retrieved information is sent back as a <b>Delegation_of_Authority</b> object to the Data service where it is then retrieved by the Angular Front End and read into any <b>Delegation</b> array.</p> <p>The information in the array is then used to pre-populate the input fields form values as follows:</p> <ul style="list-style-type: none"> <li>• “Delegating Party” input with delegatingParty</li> <li>• “Delegate” input with username</li> <li>• “Date Range” input with from_Date and to_Date</li> <li>• “Delegation Document” input with Delegation_Document</li> </ul> |

|  |   |  |   |
|--|---|--|---|
|  |   |  | The system prompts the user to enter the required details.  |
|  | <p><b>Step 3:</b> The admin enters the delegations details in their corresponding inputs.</p> |  | <p><b>Step 4:</b> The system validates the inputted details from the Angular form by checking:</p> <ul style="list-style-type: none"> <li>• That the “Delegate” input box is not null, the entered name is between 2 to 32 characters.             <ul style="list-style-type: none"> <li>○ The delegate input box makes use of autocomplete to retrieve the usernames that is similar to the information being entered.</li> </ul> </li> <li>• That the “Date Range” input box is not null, start date is not invalid, end date is not invalid</li> <li>• That the “Delegation Document” input box is not null, only a pdf document is uploaded.</li> </ul> <p>The inputs are all validated by the Angular Front End when the admin clicks and leaves an input field.</p> <p>The “Save” button is only activated once all the inputs are valid.</p> <p>[ALT]</p> |
|  | <p><b>Step 5:</b> The admin clicks on the “Save” button</p> <p>[ALT]</p>                      |  | <p><b>Step 6:</b> The system validates the input fields as in step 4.</p> <p>The system checks if a new file is selected to be uploaded.</p> <p>The system will send the DelegateName and filename variables containing the current filename to the Data service where it will make use of a http delete request to send the variables to the .Net Core controller.</p>   |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>The system uses the provided DelegateName and Filename to delete the existing file from the correlating folder.</p> <p>The system will capture the file from the File input in a file object and the name of the filename variable.</p> <p>The system send the file object and filename variable to the Data service where it will be appended into a form object which is then sent with an http post request to the .Net Core controller.</p> <p>The system will retrieve the file and file name from the Form object, where it will create a folder with the name in the file name variable which is stored in a folder named “AuthorityDelegations” in the “Files” folder.</p> <p>The system send the file path back to the frontend which is then saved in the Delegation_of_Authority object.</p> <p>The system will capture the Angular form inputs to populate the Delegation_of_Authority, User, Admin, Delegation_Status objects with the values from the inputs, the start and end dates are transformed using a DatePipe into the “MMM d, y, h:mm:ss a” format.</p> <p>The Delegation_of_Authority object, which are a combination of all the others are then sent to the Data service where it will send an http post request which will use an Entity Framework Update method to store the Delegation details by auto generating the Delegation_ID by incrementing the last added ID.</p> |
|--|--|--|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>The submitted data is stored in the following attributes in the <b>Delegation_of_Authority Entity</b>:</p> <ul style="list-style-type: none"> <li>• Delegation_ID</li> <li>• User_Id</li> <li>• Admin_ID</li> <li>• Delegation_Status_ID</li> <li>• From_Date</li> <li>• To_Date</li> <li>• Delegation_Document</li> <li>• DelegatingParty</li> </ul> <p>The correct Foreign Keys are retrieved from their primary table by comparing the entered name with the one stored in the database.</p> <p>The system then assigns the delegation ID in the <b>Temporary_Access</b> object in the frontend, which is then sent to the Data service where it will send an http post request which will use an Entity Framework Update method to store the Temporary Access details.</p> <p>The submitted data is stored in the following attributes in the <b>Temporary_Access Entity</b>:</p> <ul style="list-style-type: none"> <li>• Temporary_Access_ID</li> <li>• Delegation_ID</li> <li>• Name</li> <li>• Description</li> </ul> <p>The correct Foreign Keys are retrieved from their primary table by comparing the entered id with the one stored in the database.</p> <p><b>[ALT]</b></p> |
|  |  |  | <p><b>Step 7:</b> The system calls the CheckDelegation function in the Data service which will call the method with the same name in the Home controller that will check the from_Date and to_Date attributes</p>  |

|                           |   |  |   |
|---------------------------|---|--|---|
|                           |   |  | and compare it to the current date and change the status accordingly  |
|                           |   |  | <p><b>Step 8:</b> The system will display a success card notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “EDIT SUCCESSFUL” header</li> <li>• “The Request No (given delegation ID) has been EDITED successfully!” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p> |
|                           |   |  | <p><b>Step 9:</b> The system will invoke <b>Use Case 2.17 “View Delegation”</b></p>   |
| <b>Alternate courses:</b> |   | <p><b>ALT Step 4:</b> The system validates the inputted information using the provided validation inside the Angular form.</p> <p>The system displays the following:</p> <ul style="list-style-type: none"> <li>• “Name is required” label underneath the Name input if the input is empty.</li> <li>• “Limit exceeded” label underneath the Name input if the entered value is longer than 32 characters.</li> <li>• “Name must be 2 to 32 characters long” label underneath the Name input if the entered value is shorter than 2 characters.</li> </ul> |   |
|                           |   | <p><b>ALT Step 5:</b> The admin clicks the blue “Cancel” button.</p> <p>The system terminates the use case and invokes <b>Use Case 2.17 “View Delegation”</b></p>  |   |
|                           |   | <p><b>ALT Step 6:</b> No new document is selected to be uploaded.</p> <p>The existing document is not deleted and the file path remains unchanged.</p>   |   |
| <b>Post-condition:</b>    | <ul style="list-style-type: none"> <li>• The Delegation_of_Authority details have been successfully updated in the <b>Delegation_of_Authority Tables</b>.</li> <li>• The database retains its data integrity after the Delegation_of_Authority has been updated.</li> </ul> |  |   |

Table 198: 2.19 Edit Delegation of Authority

| <b>System Name:</b> Procion System<br><b>Sub-System:</b> Admin |  |  |
|--|--|--|
| Author: EP Wonigkeit   | Date: 22/07/2023   | Version: 1.0.0                                 |
| <b>Use case name:</b>  | Revoke Access  | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>  | 2.20   | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>   | High   | Extension: <input checked="" type="checkbox"/> |
| <b>Source:</b>   | Moyo   |  |
| <b>Primary business actor</b>                                  | <ul style="list-style-type: none"> <li>• Time</li> <li>• [ALT] Admin</li> </ul>  |  |
| <b>Primary system actor</b>                                    | None   |  |
| <b>Other participating actors:</b>                             | None   |  |
| <b>Other interested stakeholders:</b>                          | None   |  |
| <b>Description:</b>  | <p>This use case describes the event where a user's temporary access is revoked after the delegation period has passed.</p> <p>The use case begins where the system checks the to_Date of a delegation request and compares it to the current date. Should the current date falls outside of the delegation period the system will change the delegation requests status to revoked and remove the temporary_access details from the <b>Temporary_Access</b> entity in the database.</p> <p>The use case concludes once the status of the delegation request has been set to "revoked" and the temporary_access details are removed in the <b>Temporary_Access</b> entities in the database.</p> |  |
| <b>Pre-condition:</b>  | <p>A delegation request must be created.</p> <p>The Procion system must be running.</p> <p>The recursive check function must be initiated.</p>   |  |
| <b>Trigger:</b>  | The system runs the recursive check function at 00:00.   |  |

| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b> | <b>System Response</b> |   |
|--------------------------------------|---------------------|------------------------|---|
|                                      |                     | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                      |                     |                        | <p><b>Step 1:</b> The system calls the CheckDelegation function in the Home controller.</p> <p>The system then calls the GetAllDelegationsAsync() function from the Delegation repository to retrieve all the delegation requests stored in the database. The system returns a Delegation_Of_Authority array of objects which is used to populate the doa array.</p> <p>The system then calls the GetAllStatusesAsync() function from the Delegation repository to retrieve all the delegation statuses stored in the database. The system returns a Delegation_Status array of objects which is used to populate the doas array.</p> |
|                                      |                     |                        | <p><b>Step 2:</b> The system loops through all the retrieved delegation requests and first checks the from_Date and to_Date attributes of all the entries with the “Inactive” status.</p> <p>The system compares the to_Date value to the current date and then calls the UpdateDelegationStatusAsync function from the Delegation repository and passed the delegation_ID and statusID.</p> <p><b>[ALT]</b></p>  |
|                                      |                     |                        | <p><b>Step 3:</b> The system uses these values to update the delegationStatus_ID attribute in the Delegation_of_Authority entity by</p>   |

|                           |  |  |  |
|---------------------------|--|--|--|
|                           |  |  | <p>making use of an Entity Framework Update method to store the Delegation details.</p> <p>The system then calls the GetTempAccAsync function which calls an Entity Framework query that will retrieve the Temporary Access with the related Delegation_ID from the database.</p> <p>The system uses Entity Framework delete function to create an SQL delete query to delete the all the related attributes from the <b>Temporary_Access table</b>.</p> |
|                           |  |  |  |
| <b>Alternate courses:</b> |  | <p><b>ALT:</b></p> <p><b>Step 1:</b> The admin clicks on the “Revoke” button.</p> <p><b>Step 2:</b> The system loads the “Revoke Delegation” card with the following elements:</p> <ul style="list-style-type: none"> <li>• Card Header “REVOKE TEMPORARY ACCESS” in the centre of the card</li> <li>• “You are about to revoke temporary access from user: (Delegate username in bold)” paragraph text</li> <li>• “Are you sure you want to continue?” paragraph text</li> <li>• A green “Yes” button in the bottom right corner of the card.</li> <li>• A blue “Cancel” button in the bottom right corner of the card left of the “Save” button.</li> </ul> <p>All controls are enabled by default.</p> <p><b>Step 3:</b> The Admin click the “Yes” button.</p> <p><b>Step 4:</b> The system uses these values to update the delegationStatus_ID attribute in the Delegation_of_Authority entity by making use of an Entity Framework Update method to store the Delegation details.</p> <p>The system then calls the GetTempAccAsync function which calls an Entity Framework query that will retrieve the Temporary Access with the related Delegation_ID from the database.</p> |  |

|                        |  |
|------------------------|--|
|                        | <p>The system uses Entity Framework delete function to create an SQL delete query to delete all the related attributes from the <b>Temporary_Access table</b>.</p> <p><b>Step 5:</b> The system will display a success card notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “REVOKE SUCCESSFUL” header</li> <li>• “Temporary Access has been successfully revoked from user (retrieved username in bold).” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p> <p><b>Step 6:</b> The system will invoke <b>Use Case 2.17 “View Delegation”</b></p> <p><b>ALT Step 2:</b> The system loops through all the retrieved delegation requests and first checks the from_Date and to_Date attributes of all the entries with the “Active” status.</p> <p>The system compares the to_Date value to the current date and then calls the UpdateDelegationStatusAsync function from the Delegation repository and passed the delegation_ID and statusID.</p> <p><b>Proceed to Step 3.</b></p> |
| <b>Post-condition:</b> | <ul style="list-style-type: none"> <li>• The DelegationStatus_ID attribute has been successfully updated in the <b>Delegation_Of_Authority entity</b>.</li> <li>• The Temporary_Access details have been successfully removed from the <b>Temporary_Access Tables</b>.</li> <li>• The database retains its data integrity after the <b>Temporary_Access</b> has been removed.</li> <li>• The database retains its data integrity after the Delegation_of_Authority has been updated</li> </ul>   |

Table 199: 2.20 Revoke Access

| <b>System Name: Procion System</b><br><b>Sub-System: Admin</b> |                                |  |
|--|--------------------------------|--|
| Author: EP Wonigkeit   | Date: 22/07/2023               | Version: 1.0.0                                 |
| <b>Use case name:</b>  | Delete Delegation of Authority | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>  | 2.35                           | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>   | High                           | Extension: <input checked="" type="checkbox"/> |
| <b>Source:</b>   | Moyo                           |  |

|                                       |   |                        |   |
|---------------------------------------|---|------------------------|---|
| <b>Primary business actor</b>         | Admin   |                        |   |
| <b>Primary system actor</b>           | None  |                        |   |
| <b>Other participating actors:</b>    | None  |                        |   |
| <b>Other interested stakeholders:</b> | None  |                        |   |
| <b>Description:</b>                   | <p>This use case describes the event where the admin wishes to delete a delegation from the Procion system.</p> <p>The use case begins when the admin clicks on the “Administration” option on the top navigation bar. The admin will then select the “Delegation of Authority” menu item on the side navigation bar which will load the “View Delegation” screen, where the admin will click on the “Delete” button. The “Delete Delegation” card will be displayed prompting the user to confirm the deletion.</p> <p>The admin will then click on the “Yes” button and the use case concludes once the employee details are removed in the <b>Delegation_of_Authority and Temporary_Access</b> entities in the database.</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The admin must be logged in to the system</p> <p>The top navigation bar must be loaded.</p> <p>The “View Delegation” screen must be loaded</p> <p>The admin clicked the “Delete” button</p>  |                        |   |
| <b>Trigger:</b>                       | The admin navigates to the “Delete Delegation” cards by clicking on the “Delete” button   |                        |   |
| <b>Typical course of events:</b>      | <b>Actor Action</b>   | <b>System Response</b> |   |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |   |                        | <b>Step 1:</b> The system loads the “Delete Delegation” card with the following elements: |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Card Header “DELETE DELEGATION REQUEST” in the centre of the card</li> <li>• “You are about to delete the request no: (Delegation ID in bold) for delegating user: (delegatingParty name in bold)” paragraph text</li> <li>• “Are you sure you want to continue?” paragraph text</li> <li>• A green “Yes” button in the bottom right corner of the card.</li> <li>• A blue “Cancel” button in the bottom right corner of the card left of the “Save” button.</li> </ul> <p>All controls are enabled by default.</p>                         |
|  | <p><b>Step 2:</b> The Admin click the “Yes” button.<br/><b>[ALT]</b></p> |  | <p><b>Step 3:</b> The system passes the Delegation_ID to the Data service which makes use of an HTTP Delete request to pass the Delegation_ID to the Delegation_of_Authority controller.</p> <p>The API then makes use of the GetTempAccAsync function which calls an Entity Framework query that will retrieve the Temporary Access with the related Delegation_ID from the database.</p> <p>The API then makes use of the GetDelegationAsync function which calls a Entity Framework query that will retrieve the Delegation with the related Delegation_ID from the database.</p> |
|  |  |  | <p><b>Step 4:</b> The system uses Entity Framework delete function to create an SQL delete query to delete all the related attributes from the <b>Temporary_Access</b> and <b>Delegation_of_Authority</b> tables.</p>  |

|                           |  |  |  |
|---------------------------|--|--|--|
|                           |  |  | <p>The system first removes the temporary access and its details from the <b>Temporary_Access</b> table.</p> <p>The system then removes the delegation and its details from the <b>Delegation_of_Authority</b> table.</p>  |
|                           |  |  | <p><b>Step 5:</b> The system will display a success card notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “Delete Successful” header</li> <li>• “The Request No (retrieved Delegation_ID) has been DELETED successfully.” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p> |
|                           |  |  | <p><b>Step 6:</b> The system will invoke <b>Use Case 2.17 “View Delegation”</b></p>  |
| <b>Alternate courses:</b> | <p><b>ALT Step 2:</b> The admin clicks the blue “Cancel” button.</p> <p>The system terminates the use case and invokes <b>Use Case 2.17 “View Delegation”</b></p>  |  |  |
| <b>Post-condition:</b>    | <ul style="list-style-type: none"> <li>• The Temporary_Access and Delegation details have been successfully removed from the <b>Temporary_Access and Delegation_of_Authority Tables</b>.</li> <li>• The database retains its data integrity after the <b>Temporary_Access and Delegation_of_Authority</b> has been removed.</li> </ul> |  |  |

Table 200: 2.35 Delete Delegation of Authority

| <b>System Name: Procion System</b><br><b>Sub-System: Admin</b> |                                  |  |
|--|----------------------------------|--|
| Author: EP Wonigkeit   | Date: 22/07/2023                 | Version: 1.0.0                                 |
| <b>Use case name:</b>  | Activate Delegation of Authority | <b>USE CASE TYPE</b>                           |
| <b>Use case id:</b>  | 2.36                             | Abstract: <input type="checkbox"/>             |
| <b>Priority:</b>   | High                             | Extension: <input checked="" type="checkbox"/> |

| <b>Source:</b>                        | Moyo   |  |                      |                         |
|---------------------------------------|--|--|----------------------|-------------------------|
| <b>Primary business actor</b>         | Time   |  |                      |                         |
| <b>Primary system actor</b>           | None   |  |                      |                         |
| <b>Other participating actors:</b>    | None   |  |                      |                         |
| <b>Other interested stakeholders:</b> | None   |  |                      |                         |
| <b>Description:</b>                   | <p>This use case describes the event where a delegation request is set to active.</p> <p>The use case begins where the system checks the from_Date of a delegation request and compares it to the current date. Should the current date falls inside of the delegation period the system will change the delegation requests status to Active.</p> <p>The use case concludes once the status of the delegation request has been set to "Active".</p> |  |                      |                         |
| <b>Pre-condition:</b>                 | <p>A delegation request must be created.</p> <p>The Procion system must be running.</p> <p>The recursive check function must be initiated.</p>   |  |                      |                         |
| <b>Trigger:</b>                       | The system runs the recursive check function at 00:00.   |  |                      |                         |
| <b>Typical course of events:</b>      | <b>Actor Action</b>  | <b>System Response</b>   |                      |                         |
|                                       |  | <table border="1"> <thead> <tr> <th><b>Manual Action</b></th> <th><b>Automated Action</b></th> </tr> </thead> <tbody> <tr> <td></td> <td> <p><b>Step 1:</b> The system calls the CheckDelegation function in the Home controller.</p> <p>The system then calls the GetAllDelegationsAsync function from the Delegation repository to retrieve all the delegation requests stored in the database. The system returns a Delegation_Of_Authority</p> </td> </tr> </tbody> </table> | <b>Manual Action</b> | <b>Automated Action</b> |
| <b>Manual Action</b>                  | <b>Automated Action</b>  |  |                      |                         |
|                                       | <p><b>Step 1:</b> The system calls the CheckDelegation function in the Home controller.</p> <p>The system then calls the GetAllDelegationsAsync function from the Delegation repository to retrieve all the delegation requests stored in the database. The system returns a Delegation_Of_Authority</p>   |  |                      |                         |

|                           |   |  |  |
|---------------------------|---|--|--|
|                           |   |  | <p>array of objects which is used to populate the doa array.</p> <p>The system then calls the GetAllStatusesAsync function from the Delegation repository to retrieve all the delegation statuses stored in the database. The system returns a Delegation_Status array of objects which is used to populate the doas array.</p>  |
|                           |   |  | <p><b>Step 2:</b> The system loops through all the retrieved delegation requests and first checks the from_Date and to_Date attributes of all the entries with the “Inactive” status.</p> <p>The system compares the from_Date value to the current date and then calls the UpdateDelegationStatusAsync function from the Delegation repository and passes the delegation_ID and statusID.</p> |
|                           |   |  | <p><b>Step 3:</b> The system uses these values to update the delegationStatus_ID attribute in the Delegation_of_Authority entity by making use of an Entity Framework Update method to store the Delegation details.</p>   |
| <b>Alternate courses:</b> |   |  |  |
| <b>Post-condition:</b>    | <ul style="list-style-type: none"> <li>• The DelegationStatus_ID attribute has been successfully updated in the <b>Delegation_Of_Authority entity</b>.</li> <li>• The database retains its data integrity after the Delegation_of_Authority has been updated</li> </ul> |  |  |

Table 201: 2.36 Activate Delegation of Authority

## USE CASE 1.7 &amp; 2.25 &amp; 2.27-2.30

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 1 User |                  |              |
| Author: Leon Combrinck                            | Date: 2023/07/10 | Version: 1.0 |

| Use case name:                 | View User Manual  |   | USE CASE TYPE                                      |
|--------------------------------|---|---|--|
| Use case id:                   | 1.7   |   | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | medium  |   | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo  |   | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | User (PBA)  |   |  |
| Primary system actor           | None  |   |  |
| Other participating actors:    | None  |   |  |
| Other interested stakeholders: | None  |   |  |
| Description:                   | In this use case the User would like to view the User Manual. The User will thus request the system to view the User Manual. The system will then open/download the User Manual file as a PDF on the Users device allowing the user to view the manual. |   |  |
| Pre-condition:                 | The User must be logged into the system.<br><br>The system has loaded the “Help” screen.  |   |  |
| Typical course Of events:      | <b>Actor Action</b><br><br><b>Manual Action</b>   | <b>System Response</b><br><br><b>Automated Action</b>   |  |
|                                |   | <b>Step 1:</b> The user clicks on the “View User Manual” Button.<br><br><b>Step 2:</b> The system will then make use of the angular frontend and make use of the Assets |  |

|  |   |   |
|--|---|---|
|  | <b>[ALT]</b>  | folder to retrieve the "UserManual.pdf" PDF. The system will then make use of the Angular frontend openUserManualPDFInNewTab() function to open the User Manual PDF in a new tab in the user's browser. |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 1:</b> The User clicks on the User Manual link on the <b>Help</b> table. The use case continues to step 3.  |   |
| <b>Conclusion:</b>                                   | The User Manual is open on the user's browser available for viewing   |   |
| <b>Post-condition:</b>                               | The users are provided with the user manual that will help explain certain parts of the system and how to work with them. |   |
| <b>Business rules</b>                                | All users of the system are allowed to View the user's manual.  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |   |

Figure 112 1.7 View User Manual

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                       |                       |   |
|-----------------------|-----------------------|---|
| <b>Use case name:</b> | Backup data in system | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>   | 2.25                  | Business Requirements: <input type="checkbox"/> |
| <b>Priority:</b>      | High                  | System Analysis: <input type="checkbox"/>       |

|                                       |   |                        |  |
|---------------------------------------|---|------------------------|--|
| <b>Source:</b>                        | Moyo  | System Design:         | <input checked="" type="checkbox"/>  |
| <b>Primary business actor</b>         | Admin (PBA)   |                        |  |
| <b>Primary system actor</b>           | None  |                        |  |
| <b>Other participating actors:</b>    | System Database (ERA)   |                        |  |
| <b>Other interested stakeholders:</b> | None  |                        |  |
| <b>Description:</b>                   | <p>In the use case the admin will request the system to make a backup of the systems data. The admin will thus request the make a backup on the system. The system will then request confirmation. The system will then make a backup of the current data on the system and save the backup file with timestamp on the admins device after the confirmation has been received. The system will then display a successful backup notification.</p> |                        |  |
| <b>Pre-condition:</b>                 | The admin must be logged into the system.   |                        |  |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>   | <b>System Response</b> |  |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       | <p><b>Step 1:</b> The Admin navigates from the home screen to the Help screen by clicking on the <b>Administration</b> button on the top <u>user</u> navbar and then in the side <u>Administration</u> navbar clicking on the <b>Settings</b> button and then a Dropdown with two options: <u>Backup</u>, <u>Restore</u>. The Admin Clicks on the <b>Backup</b> Button.</p>   |                        | <p><b>Step 2:</b> The system will generate the “<i>Backup System Data</i>” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Backup System Data</li> <li>• Label with text: “You are about to create a Backup is the systems current data. Are you sure you want to continue?”</li> <li>• Buttons: Yes, Cancel</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  | <p><b>Step 3:</b> The Admin clicks on the “Yes” button<br/>[ALT]</p> |  | <p><b>Step 4:</b> The system retrieves all the data from the SQL Database and creates a backup of that data using the Microsoft.SqlServer.Management.SMO package.</p> <p>The angular frontend will take the response received from the API and check if it was null or not. The angular frontend will then call the CreateBackup() function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the backup Object.</p> <p>The API will then receive the Backup object and create a backup of the system on the Admins Device.</p> |
|  |  |  | <p><b>Step 5:</b> The system will generate a model by making use of the .ASP.NET core API Help object and the angular for the MatDialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: “Create Backup Successful”</li> <li>• Label with text: “The Backup “[Help] has been Created successfully!”</li> </ul> <p>The dialog will pop up for 1.75 seconds. Thereafter the user will be routed to the</p>  |

|  |   |  |   |
|--|---|--|---|
|  |   |  | previous page they were on.<br><br>[ALT]  |
|  |   |  | <b>Step 6:</b> The system will navigate to the <u>Settings</u> tab on the Administration side navbar with the two dropdowns: backup, restore by making use of Angular routes with the newly created backup data added to the Admins Device. |
| <b>Alternate courses:</b>                            | [ALT] <b>Step 3:</b> The admin clicks on the “Cancel” Button. The Admin will be redirected to the screen he/she was on previously. The use case ends.<br><br><b>ALT-Step 5:</b> The angular frontend receives the object instead of the null value. A Error notification will be generated which has the following Details: <ul style="list-style-type: none"><li>• Header: ERROR: The Backup Failed</li><li>• Body: Label: The Backup FAILED and could not be created!</li></ul> Return to Step 1. |  |   |
| <b>Conclusion:</b>                                   | The system will navigate to the <u>previous</u> screen the user was on with the newly created Backup data added to the Admins Device.   |  |   |
| <b>Post-condition:</b>                               | A Backup of the system data has been created.   |  |   |
| <b>Business rules</b>                                | Only the admin can Create a Backup to the system  |  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"><li>• None</li></ul>  |  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"><li>• None</li></ul>  |  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"><li>• None</li></ul>  |  |   |

Figure 113 2.25 Backup System Data

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

| Use case name:                 | Add Help  | USE CASE TYPE                                      |
|--------------------------------|---|--|
| Use case id:                   | 2.27  | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | medium  | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | Admin (PBA)   |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | <p>In this use case the admin would like to add a help section. The admin will thus request the system to create a help section. The system will prompt the admin to enter all the help fields required. The admin will then enter all the required fields and save the entered fields.</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will capture the information and the newly created help will be added and displayed to the <b>Help</b> table in the database.</p> |  |
| Pre-condition:                 | <p>The admin must be logged into the system.</p> <p>System has loaded the “Manage Help” screen.</p>   |  |

|                                      | The admin clicked the “Add Help” button on the “Manage Help screen. |   |                         |
|--------------------------------------|---|---|-------------------------|
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   | <b>System Response</b>  |                         |
|                                      |   | <b>Manual Action</b>  | <b>Automated Action</b> |
|                                      |   | <p><b>Step 1:</b> The system navigates to the <u>Add Help</u> screen by making use of the Angular front-end routes with the following details:</p> <ul style="list-style-type: none"> <li>-Header <ul style="list-style-type: none"> <li>• Add Help</li> </ul> </li> <li>-List of labels followed by an input box <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> </ul> </li> <li>-Label called “Video:” with a corresponding input of type file with the following text (the default text of “no file chosen” called Video file)</li> <li>-Label called “User Manual:” with a corresponding input of type file with the following text (the default text of “no file chosen” called Manual file)</li> <li>-Button <ul style="list-style-type: none"> <li>• Save</li> <li>• Cancel</li> </ul> </li> </ul> <p>Top Navbar navigation buttons (From left to right)</p> |                         |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• <b>Administration</b></li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Help</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p>The system prompts the admin to enter the required fields.</p>  |
|  | <p><b>Step 2:</b> The Admin enters the required fields</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> |  | <p><b>Step 3:</b> The system will validate the input fields that was entered.</p> <ul style="list-style-type: none"> <li>-That the “Category” dropdown box is not null.</li> <li>- That the “Name” input box is not null, must be between 2 and 32 characters, cannot contain numbers.</li> <li>- That the “Description” input box is not null, must be between 2 and 50 characters, cannot contain numbers.</li> <li>- That the “Video:” must not be null and files may not contain the same name.</li> <li>- That the “User Manual:” must not be null and files may not contain the same name.</li> </ul> |

|  |  |  |
|--|--|--|
|  |  | <p><b>Step 4:</b> The system then enables the “Save” button by using Angular validation once all field requirements are met.</p> <p>[ALT]</p>  |
|  | <p><b>Step 5:</b> The admin clicks on the “<b>Save Button</b>” [ALT]</p> | <p><b>Step 6:</b> The system then calls the AddHelp() Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function will be called that validates the Help. The function will communicate with the appropriate header in the API with an HTTPGET Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>[ALT]</p> |
|  |  | <p><b>Step 7:</b> The angular frontend will take the response received from the API and check if it was null or not. The angular frontend will then call the AddHelp() function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Help Object. The angular will also then make use of a</p>  |

|  |  |  |
|--|--|--|
|  |  | <p>FileUpload() function that allows the video and user_manual files to be uploaded and saved using a file path on the system.</p> <p>The API will then receive the Help object and add the following Details in the <b>Help entity with a SQL Read Query:</b></p> <ul style="list-style-type: none"> <li>• Category in the <i>Help_Category</i> attribute</li> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> <li>• Video (file path obtain as previously mentioned) in the <i>Video</i> attribute.</li> <li>• User Manual (file path obtain as previously mentioned) in the <i>User_Manual</i> attribute.</li> <li>• [ALT]</li> </ul> |
|  |  | <p><b>Step 8:</b> The system will generate a model by making use of the .ASP NET core API Help object and the angular for the MatDialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: “Create Successful”</li> <li>• Label with text: “The Help [Help] has been</li> </ul>  |

|                           |  |  |  |
|---------------------------|--|--|--|
|                           |  |  | <p>Created successfully!"</p> <p>The dialog will pop up for 1.75 seconds.</p> <p>Thereafter the user will be routed to the "View Help" page</p> <p>[ALT]</p>   |
|                           |  |  | <p><b>Step 9:</b> The system will navigate to the <u>Manage Help</u> screen by making use of Angular routes with the newly created help data added to the <b>Help</b> table on the View Help screen.</p> |
| <b>Alternate courses:</b> |  | <p><b>[ALT] Step 4:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. Return to step 2</p>  |  |
|                           |  | <p><b>ALT-Step 5:</b> The admin clicks on the Cancel Button; the <i>use case is terminated</i>, and the system navigate back to the <u>Manage Help</u> screen with no changes made.</p>  |  |
|                           |  | <p><b>ALT-Step 6:</b> The function will retrieve the existing Help object if a Help is found. Proceed to Step 7</p>  |  |
|                           |  | <p><b>ALT-Step 8:</b> The angular frontend receives the object instead of the null value. A Error notification will be generated which has the following Details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: The Help Exists</li> <li>• Body: Label: The Help [Help] ALREADY EXISTS!.</li> <li>• The user will be Redirected to the "<u>Add Help</u>" screen.</li> </ul> <p>The [Help] Placeholder will be populated by the Help object sent+</p> <p>Return to step 4</p> |  |
| <b>Conclusion:</b>        |  | <p>The system will navigate to the <u>Manage Help</u> screen by making use of Angular routes with the newly created Help data added to the <b>Help</b> table on the View Help screen.</p>  |  |
| <b>Post-condition:</b>    |  | <p>A new record is created in the <b>Help</b> table.</p>   |  |
| <b>Business rules</b>     |  | <p>Only the admin can Create Help to the system</p>  |  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul> |

Figure 114 2.27 Add Help

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Update Help  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 2.28   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | medium   | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | Admin (PBA)  |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>In this use case the admin would like to update a Help section. The admin will thus request the system to update a Help. The system will then prompt the admin to select the specific help fields they would like to update. The admin will then enter all the required fields and save the entered fields.</p> <ul style="list-style-type: none"> <li>Category</li> <li>Name</li> <li>Description</li> </ul> |  |

|                                      | <ul style="list-style-type: none"> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will capture the updated information and the newly updated Help will be added and displayed to the <b>Help</b> table in the database.</p>   |                         |                        |  |  |                      |                         |  |  |  |
|--------------------------------------|--|-------------------------|------------------------|--|--|----------------------|-------------------------|--|--|--|
| <b>Pre-condition:</b>                | <p>The admin must be logged into the system.</p> <p>The system has loaded the “Help” screen.</p> <p>The admin clicked the “update” button on the <b>Help</b> table of the Help screen.</p>   |                         |                        |  |  |                      |                         |  |  |  |
| <b>Typical course<br/>Of events:</b> | <table border="1"> <thead> <tr> <th><b>Actor Action</b></th> <th colspan="2"><b>System Response</b></th> </tr> <tr> <th></th> <th><b>Manual Action</b></th> <th><b>Automated Action</b></th> </tr> </thead> <tbody> <tr> <td></td> <td> <p><b>Step 1:</b> The system navigates to the <u>Edit Help</u> screen by making use of the angular front-end while sending the ID of the Help to the Edit page.</p> <p>the system will capture the ID that was passed via angular front-end and call the GetHelp() and the GetHelpCategory() function that access the [HTTPGET] methods from the .ASP NET API. Core. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Category equal to Help_Category</li> <li>• Name equal to Name</li> <li>• Description equal to Description</li> <li>• Video equal to Video</li> </ul> </td> <td></td> </tr> </tbody> </table> | <b>Actor Action</b>     | <b>System Response</b> |  |  | <b>Manual Action</b> | <b>Automated Action</b> |  | <p><b>Step 1:</b> The system navigates to the <u>Edit Help</u> screen by making use of the angular front-end while sending the ID of the Help to the Edit page.</p> <p>the system will capture the ID that was passed via angular front-end and call the GetHelp() and the GetHelpCategory() function that access the [HTTPGET] methods from the .ASP NET API. Core. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Category equal to Help_Category</li> <li>• Name equal to Name</li> <li>• Description equal to Description</li> <li>• Video equal to Video</li> </ul> |  |
| <b>Actor Action</b>                  | <b>System Response</b>   |                         |                        |  |  |                      |                         |  |  |  |
|                                      | <b>Manual Action</b>   | <b>Automated Action</b> |                        |  |  |                      |                         |  |  |  |
|                                      | <p><b>Step 1:</b> The system navigates to the <u>Edit Help</u> screen by making use of the angular front-end while sending the ID of the Help to the Edit page.</p> <p>the system will capture the ID that was passed via angular front-end and call the GetHelp() and the GetHelpCategory() function that access the [HTTPGET] methods from the .ASP NET API. Core. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Category equal to Help_Category</li> <li>• Name equal to Name</li> <li>• Description equal to Description</li> <li>• Video equal to Video</li> </ul>   |                         |                        |  |  |                      |                         |  |  |  |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• User Manual equal to User_Manual</li> </ul>  |
|  |  |  | <p><b>Step 2:</b> The system will load the “Edit Help” screen by making use of the angular frontend with the following:</p> <ul style="list-style-type: none"> <li>-Label <ul style="list-style-type: none"> <li>• Update Help</li> </ul> </li> <li>-List of labels followed by an input box <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> </ul> </li> <li>- Label called “Video:” with a corresponding input of type file with the following text (the default text of “no file chosen” called Video file)</li> <li>- Label called “User Manual:” with a corresponding input of type file with the following text (the default text of “no file chosen” called Manual file)</li> <li>-Button <ul style="list-style-type: none"> <li>• Save</li> <li>• Cancel</li> </ul> </li> </ul> <p><b>Top Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• <b>Administration</b></li> </ul> |

|  |   |  |   |
|--|---|--|---|
|  |   |  | <ul style="list-style-type: none"> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Help</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p>The system then populates each of the values on the form with the values received.</p> <p>The system prompts the admin to enter the required fields.</p>   |
|  | <p><b>Step 3:</b> The Admin enters some or all of the required fields</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p><b>[ALT]</b></p> |  | <p><b>Step 4:</b> The system will validate the input fields with angular front-end that was entered.</p> <ul style="list-style-type: none"> <li>-That the “Category” dropdown box is not null.</li> <li>- That the “Name” input box is not null, must be between 2 and 32 characters, cannot contain numbers.</li> <li>- That the “Description” input box is not null, must be between 2 and 50 characters, cannot contain numbers.</li> <li>- That the “Video:” must not be null and files may not contain the same name.</li> <li>- That the “User Manual:” must not be null and files may not</li> </ul> |

|  |   |  |
|--|---|--|
|  |   | contain the same name.   |
|  |   | <p><b>Step 5:</b> The system enables the “<b>Save Button</b>” by using angular validation once all of the fields have met the requirements.</p> <p>[ALT]</p>   |
|  | <p><b>Step 6:</b> The user clicks on the <b>Save</b> button.</p> <p>[ALT]</p> | <p><b>Step 7:</b> The system then calls the EditHelp() Function in the angular frontend which will capture all of the values provided in the angular form. The system also calls the deletefile() function in order to be able to delete to video and user_manual files and then calls the uploadfile() function to then replace those files with the newly edited files.</p> <p>Inside this function will be called that validates the help. The function will communicate with the appropriate header in the API with an [HTTPGET] Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The function will check the <b>Help Entity</b> with an <u>SQL Read query</u> to ensure that there is no</p> |

|  |  |  |
|--|--|--|
|  |  | help with the corresponding category and Description and Video name equal to the Category and Description and Video name associated with it<br><b>[ALT]</b>  |
|  |  | <p><b>Step 8:</b> The angular frontend will take the response received from the API and check if it was null or not, if the response is identical to another Help or if anything has changed with the help. The angular frontend will then call the EditHelp() function which will communicate with the appropriate header in the API by means of an [HTTPPUT] request and it will send it the Help_ID and the Help Object. The angular will also then make use of a FileUpload() function that allows the video and user_manual files to be uploaded and saved using a file path on the system.</p> <p>The API will then receive the Help object and add the following Details in the <b>Help entity with a SQL Read Query</b>:</p> <ul style="list-style-type: none"> <li>• Category in the <i>Help_Category</i> attribute</li> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> <li>• Video (file path obtain as previously mentioned) in</li> </ul> |

|                           |  |   |
|---------------------------|--|---|
|                           |  | <ul style="list-style-type: none"> <li>the <i>Video</i> attribute.</li> <li>User Manual (file path obtain as previously mentioned) in the <i>Usre_Manual</i> attribute.</li> </ul>  |
|                           |  | <p><b>Step 9:</b> The system will generate a model by making use of the .ASP NET core API Help object and the angular for the MatDialog with the following details:</p> <ul style="list-style-type: none"> <li>Header: “Update Successful”</li> <li>Label with text: “The Help “[Help] has been updated successfully!”</li> </ul> <p>The dialog will pop up for 1.75 seconds. Thereafter the user will be routed to the “View all Help” page <b>[ALT]</b></p> |
|                           |  | <p><b>Step 10:</b> The system will navigate to the <u>Help</u> screen with the newly updated Help data updated on the <u>Help</u> table.</p>  |
| <b>Alternate courses:</b> |  | <p><b>[ALT] Step 3:</b> The user has not provided any details to update and the records remains the same. Proceed to step 4.</p> <p><b>[ALT] Step 5:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. Return to step 3.</p> <p><b>[ALT] Step 6:</b> The user clicks on the “Cancel” Button. The user will be redirected to the “View all Help” screen. The use case ends</p>                                  |

|  |  |
|--|--|
|  | <p><b>[ALT] Step 7:</b> The function will retrieve the existing help object if a help is found. Proceed to Step 10</p>   |
|  | <p><b>[ALT] Step 9:</b> The help received by the ASP.NET Core API is Identical to the object in the Angular frontend. The Angular frontend will make use of an angular material dialog to display the following information to the user:</p> <ul style="list-style-type: none"> <li>• Header: “Notification: Help Exists”</li> <li>• Label with text: “The Help [help] already exists”</li> </ul> <p>The dialog will pop up for 1.75 seconds. Thereafter the user will be routed to the “View All Help” page</p> |
| <b>Conclusion:</b>                                   | The system will navigate to the <b>Help</b> screen with the newly updated help data updated on the <b>Help</b> table.  |
| <b>Post-condition:</b>                               | A record is updated in the <b>Help</b> table.  |
| <b>Business rules</b>                                | Only the admin can update Help to the system.  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• No duplicate Helps allowed.</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

*Figure 115 2.28 Update Help*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                       |             |  |
|-----------------------|-------------|--|
| <b>Use case name:</b> | Delete Help | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>   | 2.29        | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>      | medium      | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>        | Moyo        | System Design: <input checked="" type="checkbox"/> |

| <b>Primary business actor</b>         | Admin (PBA)  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Primary system actor</b>           | None   |                        |  |
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | <p>In this use case the admin would like to delete a Help. The admin will thus request the system to delete a Help. The system will then prompt the admin to the specific help they would like to delete. The admin then selects the help they would like to delete. The system then prompts the admin to confirm if they really do want to delete that specific help. The admin confirms the request and the system deletes that help from the <b>Help</b> table on the system.</p> |                        |  |
| <b>Pre-condition:</b>                 | <p>The admin must be logged into the system.</p> <p>The system has loaded the “Help” screen.</p> <p>The admin clicked the “delete” button on the <b>Help</b> table of the Help screen.</p>   |                        |  |
| <b>Typical course of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       |  |                        | <p><b>Step 1:</b> The system will use the Angular frontend to navigate to the “DeleteHelp” page while sending the ID of the help to be deleted to that respective page</p>             |
|                                       |  |                        | <p><b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend and call the GetHelp() function. This function will access the appropriate</p> |

|  |  |  |
|--|--|--|
|  |  | <p>[HTTPGET] methods on the ASP.NET Core API.</p> <p>The GetHelp() function will use the ID variable and return the first object where the <b>ID</b> variable is equal to the <b>Help_ID</b> attribute in the <b>Help</b> Entity. The system will retrieve the help object with the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Category equal to Help_Category</li> <li>• Name equal to Name</li> <li>• Description equal to Description</li> <li>• Video (file path obtain as previously mentioned) in the <i>Video</i> attribute.</li> <li>• User Manual (file path obtain as previously mentioned) in the <i>User_Manual</i> attribute.</li> </ul> |
|  |  | <p><b>Step 3:</b> The system will generate the “Delete Help” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Delete Help</li> <li>• Label with text: “You are about to delete the Help: [Help]. Are you sure you</li> </ul>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>want to continue?"</p> <ul style="list-style-type: none"> <li>Buttons: Yes, Cancel</li> </ul> <p>The [Help] placeholder will be populated with the following received from the ASP.Net API in the <b>Help</b> Entity:</p> <ul style="list-style-type: none"> <li>Category</li> </ul>   |
|  | <p><b>Step 4:</b> The Admin clicks on the “Yes” button</p> <p><b>[ALT]</b></p> |  | <p><b>Step 5:</b> The system calls the OnConfirm() function which includes the DeleteHelp() function inside of it. The angular frontend will pass the ID of the Help which was captured to the appropriate header in the ASP.NET Core API by means of an [HTTPDELETE] request. The API will then remove the record from the system by means of Entity Framework and return the deleted Help to the Angular Frontend. The system will also call the angular front end DeleteFile () function that allows the video and user_manual files to be Delete of the system.</p> |
|  |  |  | <p><b>Step 6:</b> The angular frontend will then capture the response received from the API and generate the following notification:</p> <p>Header: “Delete Successful”</p>   |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <p>Label with text: “The Help [Help] has been deleted successfully!”</p> <p>The Help placeholder will be populated with the Help:</p> <p>Category</p> <p>that was captured on the Angular frontend From the ASP.Net Core API</p> |
|  |   |  | <p><b>Step 7:</b> The system routes the Admin to the <u><a href="#">View All Help</a></u> Screen</p>   |
| <b>Alternate courses:</b>                            | <p>[ALT] <b>Step 4:</b> The admin clicks on the “Cancel” Button. The user will be redirected to the “View All Help” screen. The use case ends</p> |  |  |
| <b>Conclusion:</b>                                   | <p>The system routes the user to the <u><a href="#">View All Help</a></u> Screen</p>  |  |  |
| <b>Post-condition:</b>                               | <p>The requested deleted field has been removed from the <b>Help</b>.</p>   |  |  |
| <b>Business rules</b>                                | <p>Only the admin can delete Help from the system</p>   |  |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |

*Figure 116 2.29 Delete Help*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 2 Administration |                  |              |
| Author: Leon Combrinck                                      | Date: 2023/07/10 | Version: 1.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | View All Help   |  |
| <b>Use case id:</b>                   | 2.30  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | medium  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User (PBA)  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | <p>In this use case the user would like to view all Help on the system. The user will request the system to view all Help on the system. The system will then load all the helps that are currently on the system and display them on the <b>Help</b> table with the following details:</p> <ul style="list-style-type: none"> <li>• HelpID</li> <li>• Category</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>The system will then prompt the user to enter the search field and at the search bar. The user will then enter the search field at the search bar. The system will then filter the list of Helps based on the users search criteria. The system will then display the list of the searched criteria.</p> |  |
| <b>Pre-condition:</b>                 | <p>The user must be logged into the system.</p> <p>all Users has the authority to view the “Help” screen.</p> <p>The system loaded the Administration side navbar.</p>  |  |
| <b>Trigger:</b>                       | User requests to view All Help on the system.   |  |
| <b>Typical course</b>                 | <b>Actor Action</b>   | <b>System Response</b>                             |

| Of events: |   | Manual Action | Automated Action   |
|------------|---|---------------|--|
|            | <p><b>Step 1:</b> The User navigates from the home screen to the Help screen by clicking on the <b>Help</b> button on the top <u>user</u> navbar.</p> <p><b>[ALT]</b></p> |               | <p><b>Step 2:</b> The system will call the GetHelps() and GetHelpCategory() function then make use of the Angular frontend of the Help data by making a [HTTPGET] request to the .net core API backend. The backend uses a <i>SQL read query</i> to retrieve the help data from the database.</p> <p>The system will thus retrieve from that <b>Help</b> table in the database the following information:</p> <ul style="list-style-type: none"> <li>• Help_ID</li> <li>• Name</li> <li>• Description</li> <li>• Video</li> <li>• User Manual</li> </ul> <p>From the <b>Help_Category</b> entity using Help_Category_ID</p> <ul style="list-style-type: none"> <li>• Category</li> </ul> |
|            |   |               | <p><b>Step 3:</b> The system will display the “View All Help” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Label containing text: “View All Help”</li> <li>• Button: Add Help(Only interactable by the admin)</li> <li>• Button: View User Manual</li> </ul>  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Search Input: Label Containing “Search by Name”</li> <li>• <b>Table Containing:</b></li> <li>• Column headers:           <ul style="list-style-type: none"> <li>-No,</li> <li>-Category,</li> <li>-Name,</li> <li>-Description,</li> <li>-Video</li> <li>-User Manual</li> </ul> </li> <li>• Records with the following buttons (only interactable by the admin):           <ul style="list-style-type: none"> <li>-Edit button,</li> <li>-Delete Button</li> </ul> </li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• <b>Help</b></li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p><b>Side Navbar:</b></p> <ul style="list-style-type: none"> <li>• “Employee” tab</li> <li>• “Admin” tab</li> <li>• “Roles” tab</li> <li>• “Department” tab</li> <li>• “Branch” tab</li> <li>• “Delegation of authority” tab</li> <li>• “Mandate limits” tab</li> <li>• <b>“Manage Help” tab</b></li> </ul> |
|--|--|--|--|

|  |   |  |
|--|---|--|
|  |   | <ul style="list-style-type: none"> <li>“Settings” tab</li> </ul>   |
|  |   | <p><b>Step 4:</b> The system will then populate the searched Helps array and the Helps array with the data retrieved from the database by means of the GetHelps() Function.. The angular material table will be populated with the data contained in the searchedHelps array.</p> <p>The system will prompts the user to enter search criteria.</p> <p>[ALT]</p>       |
|  | <p><b>Step 5:</b> The user provides search criteria with the following details.</p> <ul style="list-style-type: none"> <li>Name</li> </ul> <p>[ALT]</p> | <p><b>Step 6:</b> The system makes use of the LINQ LAMBDA function to populate the help array with the <b>Help</b> table information.</p> <p>The system then populates the model array by using the API to the Angular frontend that populated the Help service. This model array then corresponds with the help typescripts in the Angular frontend.</p> <p>[ALT]</p> |
|  |   | <p><b>Step 7:</b> The system then prompts the user to click on the help related video and/or user manual in the <b>help</b> table with the following details:</p>  |

|                           |  |   |  |
|---------------------------|--|---|--|
|                           |  |   | <ul style="list-style-type: none"> <li>• Video</li> <li>• User Manual</li> </ul>   |
|                           | <p><b>Step 8:</b> The user clicks on the Video link of the Help they wish to receive.</p> <p>[ALT]</p> |   | <p><b>Step 9:</b> The system makes use of an VideoDialog to create a popup screen using the chosen video path with the following information:</p> <ul style="list-style-type: none"> <li>• Iframe: Help Video</li> </ul> |
| <b>Alternate courses:</b> |  | <p><b>[ALT] Step 1a:</b> The User navigates from the home screen to the Help screen by clicking on the <b>Help</b> button on the Home screen. Continue to step 2</p> <p><b>[ALT] Step 1b:</b> The user Navigates from the home screen to the Help screen b y clicking on the “<b>Administration</b>” button on the top navbar then by clicking on the “<b>Manage Help</b>” button on the administration side navbar.(Admin Privileges only). Continue to step 2</p> <p><b>[ALT] Step 4a:</b> There were no Help entries to retrieve. The Table will not be populated and left blank. The admin will add a Help (<b>Invoke UC 2.27: Add Help</b>) and return to step 1.</p> <p><b>[ALT] Step 4b:</b> There were no Help entries to retrieve. The Table will not be populated and left blank. <b>The use case ends</b></p> <p><b>[ALT] Step 5a:</b> The user has not provided any search criteria. Proceed to step 7.</p> <p><b>[ALT] Step 5b:</b> The Admin (Only interactable by the admin) clicks on the “Add Help” button located on the Help screen. The system will Extend to the 2.27 “Add Help”. <b>The use case ends</b>.</p> <p><b>[ALT] Step 5c:</b> The Admin (Only interactable by the admin) clicks on the “Edit” button located on the Help screen. The system will Extend to the 2.28 “Update Help”. The use case ends.</p> <p><b>[ALT] Step 5d:</b> The Admin (Only interactable by the admin) clicks on the “Delete” button located on the Help screen. The system will Extend to the 2.29 “Delete Help”. The use case ends.</p> <p><b>[ALT] Step 6a:</b> No Help were found containing the search criteria. The user will add a Help (<b>Invoke UC 2.27: Add Help</b>) and return to step 1</p> <p><b>[ALT] Step 6b:</b> No Help was found containing the search criteria. The Table will not be populated and left blank. <b>The use case ends</b>.</p> |  |

|  |   |
|--|---|
|  | <p><b>[ALT] Step 8a:</b> The User clicks on the User Manual link on the <b>Help</b> table. The system will Extend to the 1.7 “View User Manual”. The use case ends.</p> <p><b>[ALT] Step 8b:</b> The User does not click on any of the two links (Video or User Manual). The use case ends.</p> |
| <b>Conclusion:</b>                                   | The system displays the Help list that matches with the users search criteria.  |
| <b>Post-condition:</b>                               | The matching <b>Help</b> table fields are displayed.  |
| <b>Business rules</b>                                | All the users can View All Help on the system.  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

Figure 117 2.30 View All Help

**USE CASE 3.8 – 3.12 & 4.1-4.6**
**System Name:** Procion System

**Sub-System:**

|                        |                    |                |
|------------------------|--------------------|----------------|
| Author: Bupe Chindongo | Date: 25 July 2023 | Version: 1.0.0 |
|------------------------|--------------------|----------------|

| Use case name:                 | USE CASE TYPE   |  |   |
|--------------------------------|---|--|---|
| Use case id:                   | 3.8   | Business Requirements: <input type="checkbox"/>    |   |
| Priority:                      | High  | System Analysis: <input type="checkbox"/>          |   |
| Source:                        | Moyo  | System Design: <input checked="" type="checkbox"/> |   |
| Primary business actor         | Requester   |  |   |
| Primary system actor           | None  |  |   |
| Other participating actors:    | None  |  |   |
| Other interested stakeholders: | None  |  |   |
| Description:                   | This use case describes the event whereby a requester on the system would like to receive procurement items. The system will prompt the requester to update the stock items which have been received, and the requester will update the stock to match with what has been received. |  |   |
| Pre-condition:                 | The requester has to be logged in to the system.<br>The main nav bar has to be loaded   |  |   |
| Trigger:                       | The requester requests to receive procurement items   |  |   |
| Typical course<br>Of events:   | <b>Actor Action</b>   | <b>System Response</b>                             |   |
|                                |   | <b>Manual Action</b>                               | <b>Automated Action</b>                         |
|                                | <b>Step 1:</b> The requester requests to receive  | <b>None</b>  | <b>Step 2:</b> The system responds by using the |

## Iteration 5: Design &amp; Development– Team 11

|  |   |  |   |
|--|---|--|---|
|  | <p>procurement items by clicking on the “<b>Receive Procurement Item</b>” button present on the “Procurement” sidenav</p> |  | <p>Angular Frontend to route to the “Receive Procurement Item” page with the following items:</p> <ul style="list-style-type: none"> <li>• Header: Receive Procurement Item</li> <li>• Label: Name</li> <li>• Read-only text field corresponding to : [Procurement Item Name], Max reorder Quantity, Min Reorder Quantity, Category</li> <li>• Number Selector: On Hand</li> <li>• Buttons: Save, Cancel</li> </ul> <p>The system will retrieve the procurement item’s name by making a request to the .Net Core API by utilizing GetConsumableForRequest() function in the service class to retrieve the details of the procurement item which has been selected. The function will make a request to the HttpGet endpoint in the .Net Core API which will return the details of the given consumable.</p> |
|--|---|--|---|

|  |   |  |   |
|--|---|--|---|
|  | <b>Step 3:</b> The requester will enter the amount received into the relevant field |  | <b>Step 4:</b> The system validates that a valid input has been filled in and is in the required format: <ul style="list-style-type: none"> <li>• On_Hand: Must contain only numbers, no negatives.</li> </ul>  |
|  |   |  | <b>Step 5:</b> The system enables the “Save Button” by using angular validation once all of the required fields have been filled in with appropriate data. <b>[ALT]</b>   |
|  | <b>Step 6:</b> The requester clicks on the “Save” Button<br><br><b>[ALT]</b>        |  | <b>Step 7:</b> The system calls the UpdateStock () function in the angular frontend which will capture the details provided in the angular form. This function communicates with the appropriate header in the API by means of an HTTPPost Request, and it will send the ID of the consumable which needs its stock updated, as well as the consumable object.<br><br>The API will then receive the consumable and add the following details in the <b>Consumable</b> entity with an SQL Insert query : |

|  |   |      |   |
|--|---|------|---|
|  |   |      | <ul style="list-style-type: none"> <li>• On_Hand</li> </ul> <p>As well as update the <b>Consumable_History</b> table to include Consumable_ID, The StockAmt and the DateCaptured.</p> |
|  |   | None | <p><b>Step 8:</b> The system will route the user to the “View Consumables” screen</p> <p>[ALT]</p>  |
| <p><b>[ALT] Step 5:</b> The required inputs do not correspond to the validation criteria. Return to Step 3.</p> <p><b>[ALT] Step 6:</b> The user clicks on the “Cancel” Button. The system routes the user to the Procurement Screen and the use case ends.</p> <p><b>[ALT] Step 8:</b> The system will display a “Update unsuccessful” notification. Which has the following details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: Unsuccessful</li> <li>• Body: Label: The receipt of the item [Item Name] Was Unsuccessful!</li> <li>• The user will be Redirected to the <u>“ProcurementRequest”</u> screen.</li> </ul> |   |      |   |
| <b>Conclusion:</b>   | The user is redirected to the “ViewConsumable” screen   |      |   |
| <b>Post-condition:</b>   | <ul style="list-style-type: none"> <li>• The quantity on hand of the consumable has been updated on the system and the database is updated</li> </ul> |      |   |
| <b>Business rules</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |      |   |
| <b>Implementation constraints and specifications</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul>  |      |   |
| <b>Assumptions:</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |      |   |
| <b>Open issues:</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |      |   |

Figure 118 3.8 Receive Procurement Item

| <b>System Name:</b> Procion System<br><b>Sub-System:</b> |                    |                |
|--|--------------------|----------------|
| Author: Bupe Chindongo                                   | Date: 25 July 2023 | Version: 1.0.0 |

| Use case name:                 | Upload tax invoice  | USE CASE TYPE                                      |
|--------------------------------|---|--|
| Use case id:                   | 3.9   | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | High  | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | User  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | The use case describes the event whereby a requester would like to upload a tax invoice for a procurement request. The requester will provide the required file to the system, and the system will save the file into its database. The system will display a notification that the file has been successfully added to the database. |  |
| Pre-condition:                 | The requester has to be logged in to the system.<br>The main nav bar has to be loaded   |  |
| Trigger:                       | The requester requests to upload a tax invoice for a procurement request  |  |
| Typical course                 | Actor Action  | System Response                                    |

| <b>Of events:</b> |   | <b>Manual Action</b> | <b>Automated Action</b>  |
|-------------------|---|----------------------|--|
|                   | <b>Step 1:</b> The requester requests to upload a tax invoice for a procurement request by clicking on the “Invoice” button on a procurement request which is awaiting budget owner requisition approval. | <b>None</b>          | <b>Step 2:</b> The system responds by utilizing the angular frontend to load a dialogue box with a file Upload button. |

|  |   |      |  |
|--|---|------|--|
|  | <b>Step 3:</b> The requester clicks on the “File Upload” button and selects the file which they wish to upload. |      | <b>Step 4:</b> The system validates that the file entered is valid by making use of angular validation<br><br>[ALT]  |
|  | <b>Step 5:</b> The requester clicks on the “Upload” button<br><br>[Alt]   | None | <b>Step 6:</b> The system captures the file by calling the AddTaxInvoice() method in the angular frontend, which will connect to the .Net Core API and communicate with the corresponding [HttpPost] method and send the following details: <ul style="list-style-type: none"> <li>• ProcurementRequestID</li> <li>• File</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>The API will receive these methods as parameters and store them in the appropriate corresponding variables, which will then be used to set the filepath and directory. The file will then be added to the system by storing the file in the relevant directory, with the appropriate path.</p>  |
|  |  |  | <p><b>Step 7:</b> The system displays a Create Notification my making use of the angular frontend. The following will be displayed:</p> <ul style="list-style-type: none"> <li>• Header: Upload successful</li> <li>• The file [Name] has been successfully saved.</li> </ul> <p>The notification will be displayed for 1.75 seconds, after which the user will be redirected to the "ViewProcurementRequest" Screen.</p> <p>[ALT]</p> |
| <p><b>[ALT] Step 4:</b> The submitted file does not match the validation criteria. Return to Step 3.</p> |  |  |  |

|  |  |
|--|--|
|  | <p><b>[ALT] Step 5:</b> The requester clicks on the “Cancel” button. The use case is terminated.</p> <p><b>[ALT] Step 7:</b> The File could not successfully be added, and thus an Unsuccessful notification is displayed by making use of the angular frontend. The following details are displayed:</p> <p><b>Header:</b> Upload unsuccessful</p> <p><b>Body:</b> The file [Name] has failed to be added to the system.</p> <p>The notification will display for 1.75 seconds. Return to Step 3.</p> |
| <b>Conclusion:</b>                                   | The requester is redirected to the “ <b>Procurement Request</b> ” screen   |
| <b>Post-condition:</b>                               | <ul style="list-style-type: none"> <li>The file is successfully added to the system.</li> </ul>  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |

Figure 119 3.9 Upload Tax Invoice

|   |                    |                |
|---|--------------------|----------------|
| <p style="text-align: center;"><b>System Name: Procion System</b></p> <p style="text-align: center;"><b>Sub-System:</b></p> |                    |                |
| Author: Bupe Chindongo  | Date: 25 July 2023 | Version: 1.0.0 |

|                       |                                   |   |
|-----------------------|-----------------------------------|---|
| <b>Use case name:</b> | Budget owner requisition approval | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>   | 3.10                              | Business Requirements: <input type="checkbox"/> |
| <b>Priority:</b>      | High                              | System Analysis: <input type="checkbox"/>       |

|                                       |   |                        |  |
|---------------------------------------|---|------------------------|--|
| <b>Source:</b>                        | Moyo  | System Design:         | <input checked="" type="checkbox"/>  |
| <b>Primary business actor</b>         | Budget Owner  |                        |  |
| <b>Primary system actor</b>           | None  |                        |  |
| <b>Other participating actors:</b>    | None  |                        |  |
| <b>Other interested stakeholders:</b> | None  |                        |  |
| <b>Description:</b>                   | <p>This use case describes the event whereby a budget owner would like to approve a requisition request. The budget owner will request to view the requisition request, after which the system will display all the details related to the relevant procurement request. The budget owner will verify that the details of the request are in order, such as the invoice has been uploaded, the total that is due, the budget line that has been selected, after which the budget owner will approve the procurement request. After the request has been approved, the finance department will be notified to continue with payment.</p> |                        |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.<br/>The main nav bar has to be loaded</p>  |                        |  |
| <b>Trigger:</b>                       | The budget owner requests to review a requisition request.  |                        |  |
| <b>Typical course of events:</b>      | <b>Actor Action</b>   | <b>System Response</b> |  |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       | <p><b>Step 1:</b> The budget owner requests to review a requisition request by clicking on the “<b>View Unapproved Requests</b>” button present on the “Procurement” sidenav</p>  | <b>None</b>            | <p><b>Step 2:</b> The system responds by using the Angular Frontend to route to the “Review requisition requests” page with the following items:</p> <ul style="list-style-type: none"> <li>• Table containing the following headers:<br/>Name, Description, Vendor, Status</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Button: View</li> </ul> <p>The system will retrieve the above details by making a request to the .Net Core API by utilizing GetUnapprovedRequests() function in the service class to retrieve the details of the procurement requests which are in need of approval. The function will make a request to the HttpGet endpoint in the .Net Core API which will return the details of the given requests.</p> |
|--|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  | <p><b>Step 3:</b> The budget owner will click on the “View” button of the requisition request which they wish to approve</p> |  | <p><b>Step 4:</b> The system responds by loading the details of the specific requisition request that the budget owner wishes to approve through the use of the Angular frontend. The following details will be displayed:</p> <ul style="list-style-type: none"> <li>• Header: Approve requisition request</li> <li>• invoice</li> <li>• Total due</li> <li>• Budget line</li> <li>• Buttons: Sign off, Cancel</li> </ul> <p>The system retrieves these details by making use of the</p> |
|--|--|--|---|

|  |   |  |   |
|--|---|--|---|
|  |   |  | GetUnapprovedReques tByID function in the service class of the Angular frontend. This function communicates with the corresponding HttpGet endpoint in the .Net Core web api to perform a SQL read and return the details of the procurement request which the budget owner clicked on by means of its ID. This SQL read query looks through the <b>Procurement_Details</b> table for the procurement details which match the ID parsed and returns the details to be used by the angular frontend. |
|  | <p><b>Step 5:</b> The user clicks on the “Sign off” button</p> <p>[ALT]</p> |  | <p><b>Step 6:</b> The system utilizes the angular frontend to call the ApproveRequisitionRequest() function from the service class, which will communicate with the relevant HttpPut method in the .Net Core backend in order to update the status of the requisition request to “approved” by means of an SQL write query.</p>   |
|  |   |  | <p><b>Step 7:</b> The system will display a Success notification with the following details:</p>  |

|   |   |  |   |
|---|---|--|---|
|   |   |  | <ul style="list-style-type: none"> <li>• Header: “Approval successful”</li> <li>• Message Body: “The requisition request has been successfully approved”</li> </ul> <p>The notification will appear for 1.75 seconds, after which the budget owner will be routed back to the <b>ViewProcurementDetails</b> screen.</p> <p><b>[ALT]</b></p> |
|   |   |  |   |
| <p><b>[ALT] Step 5:</b> The user clicks on the “Cancel” button. Terminate use case.</p> <p><b>[ALT] Step 7:</b> The system will display a “receipt unsuccessful” notification. Which has the following details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: Unsuccessful</li> <li>• Body: Label: The requisition request has failed to be approved.</li> <li>• The user will be Redirected to the <u><b>ProcurementDetails</b></u> screen.</li> </ul> |   |  |   |
| <b>Conclusion:</b>  | The user is redirected to the <b>“ViewProcurementDetails”</b> screen  |  |   |
| <b>Post-condition:</b>  | <ul style="list-style-type: none"> <li>• The status of the corresponding procurement request has been changed in the database.</li> </ul> |  |   |
| <b>Business rules</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Implementation constraints and specifications</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Assumptions:</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |

|                     |        |
|---------------------|--------|
| <b>Open issues:</b> | • None |
|---------------------|--------|

*Figure 120 3.10 Budget Owner Requisition Approval*

|  |                    |                |
|--|--------------------|----------------|
| <b>System Name:</b> Procision System<br><b>Sub-System:</b> |                    |                |
| Author: Bupe Chindongo                                     | Date: 25 July 2023 | Version: 1.0.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Upload receipt  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 3.11  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | Requester   |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | The use case describes the event whereby a requester would like to upload a receipt for a procurement request. The requester will provide the required file to the system, and the system will save the file into its database. The system will display a notification that the file has been successfully added to the database. |  |
| <b>Pre-condition:</b>                 | The requester has to be logged in to the system.<br>The main nav bar has to be loaded   |  |
| <b>Trigger:</b>                       | The requester requests to upload a receipt for a procurement request  |  |
| <b>Typical course</b>                 | <b>Actor Action</b>   | <b>System Response</b>                             |

| <b>Of events:</b> |   | <b>Manual Action</b> | <b>Automated Action</b>   |
|-------------------|---|----------------------|---|
|                   | <b>Step 1:</b> The requester requests to upload a receipt for a procurement request by clicking on the “Proof of Payment” button on a procurement request | <b>None</b>          | <b>Step 2:</b> The system responds by utilizing the angular frontend to load the procurement request's details. |

|  |  |             |  |
|--|--|-------------|--|
|  | <p><b>Step 3:</b> The requester clicks on the “File Upload” button next to the “Receipt” label and selects the file which they wish to upload.</p> |             | <p><b>Step 4:</b> The system validates that the file entered is valid by making use of angular validation</p> <p>[ALT]</p>   |
|  | <p><b>Step 5:</b> The requester clicks on the “Upload” button</p> <p>[Alt]</p>   | <p>None</p> | <p><b>Step 6:</b> The system captures the file by calling the AddReceipt() method in the angular frontend, which will connect to the .Net Core API and communicate with the corresponding [HttpPost] method and send the following details:</p> <ul style="list-style-type: none"> <li>• ProcurementRequestID</li> <li>• File</li> </ul> <p>The API will receive these methods as parameters and store them in the appropriate corresponding variables, which will</p> |

## Iteration 5: Design &amp; Development– Team 11

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>then be used to set the filepath and directory. The file will then be added to the system by storing the file in the relevant directory, with the appropriate path.</p>   |
|  |  |  | <p><b>Step 7:</b> The system displays a Create Notification my making use of the angular frontend. The following will be displayed:</p> <ul style="list-style-type: none"> <li>• Header: Upload successful</li> <li>• The file [Name] has been successfully saved.</li> </ul> <p>The notification will be displayed for 1.75 seconds, after which the user will be redirected to the Details” Screen.</p> <p>[ALT]</p> |
| <p><b>[ALT] Step 4:</b> The submitted file does not match the validation criteria. Return to Step 3.</p> <p><b>[ALT] Step 5:</b> The requester clicks on the “Cancel” button. The use case is terminated.</p> <p><b>[ALT] Step 7:</b> The File could not successfully be added, and thus an Unsuccessful notification is displayed by making use of the angular frontend. The following details are displayed:</p> |  |  |  |

|  |  |
|--|--|
|  | <p><b>Header:</b> Upload unsuccessful</p> <p><b>Body:</b> The file [Name] has failed to be added to the system.</p> <p>The notification will display for 1.75 seconds. Return to Step 3.</p> |
| <b>Conclusion:</b>                                   | The requester is redirected to the “ <b>Procurement Details</b> ” screen   |
| <b>Post-condition:</b>                               | <ul style="list-style-type: none"> <li>The file is successfully added to the system.</li> </ul>  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>None</li> </ul>   |

*Figure 121 3.11 Upload Receipt*

|   |                    |                |
|---|--------------------|----------------|
| <p style="text-align: center;"><b>System Name: Procision System</b></p> <p style="text-align: center;"><b>Sub-System:</b></p> |                    |                |
| Author: Bupe Chindongo  | Date: 25 July 2023 | Version: 1.0.0 |

|                               |                                 |  |
|-------------------------------|---------------------------------|--|
| <b>Use case name:</b>         | <b>View Procurement Details</b> | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 3.13                            | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High                            | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo                            | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User                            |  |
| <b>Primary system actor</b>   | None                            |  |

|                                       |  |                        |   |
|---------------------------------------|--|------------------------|---|
| <b>Other participating actors:</b>    | None   |                        |   |
| <b>Other interested stakeholders:</b> | None   |                        |   |
| <b>Description:</b>                   | <p>This use case describes the event whereby a user on the system would like to view a procurement request which possesses procurement details. The user will navigate to the screen using the side nav, and the system will load the details of the procurement requests which the user has made, and display these to the user. The user will be able to upload any necessary proof of payment documents such as a receipt or credit card payment, and receive a procurement item, given the procurement request is not flagged.</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.<br/>The main nav bar has to be loaded</p>   |                        |   |
| <b>Trigger:</b>                       | The user requests to view the details of the procurement requests which they have made.  |                        |   |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       | <p><b>Step 1:</b> The user requests to view their procurement requests which have been placed by clicking on the “View Procurement Details” button present on the sidenav bar present on the Procurement tab.</p>  | <b>None</b>            | <p><b>Step 2:</b> The system will make use of the GetProcurementRequestDetails function from the service class which communicates with a relevant [HttpGet] endpoint in the .Net Core web api to return the details of the procurement request which for a given user. This endpoint will return the following data from the Procurement_Details table by means of an SQL Read Query:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID</li> <li>• Employee_ID</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Sign_Off_Status_ID</li> <li>• Procurement_Payment_Status_ID</li> <li>• Account_Code</li> <li>• Procurement_Status_ID</li> <li>• Payment_Method_ID</li> <li>• Budget_ID</li> <li>• Category_ID</li> <li>• Item_Type</li> <li>• Buyer_Name</li> <li>• Buyer_Email</li> <li>• Deposit_Required</li> <li>• Full_Payment_Due_Date</li> <li>• Total_Amount</li> <li>• Payment_Made</li> <li>• Comment</li> <li>• Proof_Of_Payment_Required</li> </ul> |
|--|--|--|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p><b>Step 3:</b> The system will display the “View Procurement Details” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: View Procurement Details</li> <li>• Table rows with the following headers: Name, Description, Vendor, Status,</li> <li>• Upload Proof of Payment Button</li> <li>• Upload Invoice Button</li> </ul> |
|--|--|--|--|

|   |   |      |  |
|---|---|------|--|
|   |   |      | <ul style="list-style-type: none"> <li>• Receive Item Button (only visible if request status is not flagged)</li> </ul>  |
|   |   |      | <p><b>Step 4:</b> The system will then populate the searchedRequestsarray with the data retrieved from the database by means of the GetProcurementRequests() Function.. The angular material table will be populated with the data contained in the searchedRequestsarray array. [ALT]</p>   |
|   | <p><b>Step 5:</b> The user will provide the following criteria inside the Search input to filter through the consumables displayed on the page:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>[Alt]</p> | None | <p><b>Step 6:</b> The angular frontend will then filter through all of the data in the searchedRequestsarray array by means of a Linq Lambda function to retrieve any record which includes a letter contained in the Search input. The angular frontend will then adjust the table dynamically with each KeyUp Event and display it to the user.[ALT]</p> |
| <p><b>[ALT] Step 4:</b> There were no details entries to retrieve. The Table will not be populated and left blank. Terminate Use case.</p> <p><b>[ALT] Step 5:</b> The user has not provided any search criteria. Return to step 4. The use case ends</p> <p><b>[ALT] Step 6:</b> No details were found containing the search criteria. The Table will not be populated and left blank. The use case ends</p> |   |      |  |

|  |  |
|--|--|
| <b>Conclusion:</b>                                   | The user is redirected to the “Procurement Details” screen |
| <b>Post-condition:</b>                               | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

*Figure 122 3.13 View Procurement Details*

|  |                    |                |
|--|--------------------|----------------|
| <b>System Name:</b> Procion System<br><b>Sub-System:</b> |                    |                |
| Author: Bupe Chindongo                                   | Date: 25 July 2023 | Version: 1.0.0 |

|                               |                              |  |
|-------------------------------|------------------------------|--|
| <b>Use case name:</b>         | Finalize Procurement request | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 4.1                          | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High                         | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo                         | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | Finance user                 |  |

|                                       |  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Primary system actor</b>           | None   |                        |  |
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | <p>This use case describes the event whereby a finance user would like to finalize a procurement request. The finance user will request to view the procurement request, after which the system will display all the details related to the relevant procurement request. The finance user will verify that the details of the procurement request are in order, after which the finance user will approve the procurement request. After the request has been approved, requester will be notified that their procurement request has been finalized.</p> |                        |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.<br/>The main nav bar has to be loaded</p>   |                        |  |
| <b>Trigger:</b>                       | The finance user requests to finalize a procurement request.   |                        |  |
| <b>Typical course of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       | <p><b>Step 1:</b> The finance user requests to review the unfinalized procurement requests by clicking on the “Finalize procurement requests” button present on the “Procurement” sidenav</p>  | <p><b>None</b></p>     | <p><b>Step 2:</b> The system responds by using the Angular Frontend to route to the “Finalize Procurement Requests” page with the following items:</p> <ul style="list-style-type: none"> <li>• Table containing the following headers: Name, Description, User, Vendor, Status</li> <li>• Button: View</li> </ul> <p>The system will retrieve the above details by making a request to the .Net Core API by</p> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | utilizing<br>GetUnfinalizedRequest<br>s() function in the<br>service class to retrieve<br>the details of the<br>procurement requests<br>which are in need of<br>finalization. The<br>function will make a<br>request to the HttpGet<br>endpoint in the .Net<br>Core API which will<br>return the details of the<br>given requests. |
|--|--|--|--|

|  |   |  |   |
|--|---|--|---|
|  | <p><b>Step 3:</b> finance user will click on the “View” button of the procurement request which they wish to finalize</p> |  | <p><b>Step 4:</b> The system responds by loading the details of the specific procurement request that the finance user wishes to approve through the use of the Angular frontend. The following details will be displayed:</p> <ul style="list-style-type: none"> <li>• Header: Approve Finalize procurement request</li> <li>• Checkbox showing that the tax invoice has been uploaded</li> <li>• Clickable link to download the invoice</li> <li>• Total due</li> <li>• Budget line</li> <li>• Buttons: Approve, Cancel</li> </ul> <p>The system retrieves these details by making use of the GetUnfinalizedRequest</p> |
|--|---|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | ByID function in the service class of the Angular frontend. This function communicates with the corresponding HttpGet endpoint in the .Net Core web api to perform a SQL read and return the details of the procurement request which the budget owner clicked on by means of its ID. This SQL read query looks through the <b>Procurement_Details</b> table for the procurement details which match the ID parsed and returns the details to be used by the angular frontend. |
|  | <p><b>Step 5:</b> The user clicks on the “Approve” button</p> <p>[ALT]</p> |  | <p><b>Step 6:</b> The system utilizes the angular frontend to call the FinalizeProcurementRequest() function from the service class, which will communicate with the relevant HttpPut method in the .Net Core backend in order to update the status of the procurement request to “fianlized” by means of an SQL write query.</p>  |
|  |  |  | <p><b>Step 7:</b> The system will display a Success notification with the following details:</p>   |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <ul style="list-style-type: none"> <li>• Header: “Finalization successful”</li> <li>• Message Body: “The procurement request has been successfully finalized”</li> </ul> <p>The notification will appear for 1.75 seconds, after which the budget owner will be routed back to the ViewProcurementRequests screen.</p> <p><b>[ALT]</b></p> |
|  |   |  |  |
| <p><b>[ALT] Step 5:</b> The user clicks on the “Cancel” button. Terminate use case.</p> <p><b>[ALT] Step 7:</b> The system will display a “finalization unsuccessful” notification. Which has the following details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: Unsuccessful</li> <li>• Body: Label: The procurement request has failed to be finalized.</li> <li>• The user will be Redirected to the <u>“ProcurementRequest”</u> screen.</li> </ul> |   |  |  |
| <b>Conclusion:</b>   | The user is redirected to the “ViewProcurementRequest” screen   |  |  |
| <b>Post-condition:</b>   | <ul style="list-style-type: none"> <li>• The requester is notified that the procurement request has successfully been finalized.</li> </ul> |  |  |
| <b>Business rules</b>  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |
| <b>Implementation constraints and specifications</b>   | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |

|                     |  |
|---------------------|--|
| <b>Assumptions:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Figure 123 4.1 Finalize Procurement Request

|  |                   |                |
|--|-------------------|----------------|
| <b>System Name:</b> Procion System<br><b>Sub-System:</b> 4 |                   |                |
| Author: Bupe Chindongo                                     | Date: 16 May 2023 | Version: 1.0.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Create Budget Allocation  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 4.3   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | Budget Owner  |  |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to create a new budget allocation. The user will click on the <b>Create Allocation</b> button on the <u><a href="#">View Budget Allocations</a></u> Screen. The user will input all of the relevant details that they wish for the budget allocation, the system will validate these details and the user will click on the <b>Save</b> Button. The system will save the new budget allocation and the use case concludes.</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.   |                        |  |
|--------------------------------------|---|------------------------|--|
| <b>Trigger:</b>                      | The user requests to add a budget category to the system  |                        |  |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   | <b>System Response</b> |  |
|                                      |   | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                      |   |                        | Step 1: The system invokes Use Case 4.4 View Budget Allocations  |
|                                      | Step 2: The user requests to add a budget allocation to the system by clicking on the “Create Allocation” Button located on the “View Budget Allocation” Screen | <b>None</b>            | <p>Step 3: The system makes use of the angular frontend and routes the user to the “Create Budget allocation” page with the following Items:</p> <ul style="list-style-type: none"> <li>● Header: Create Budget Allocation</li> <li>● Department Label with a corresponding dropdown input</li> <li>● Date Label with a corresponding date picker input</li> <li>● Total Label with a corresponding input</li> <li>● Year label with a corresponding input</li> <li>● Buttons: Save, Cancel</li> </ul> <p>Top Navbar navigation buttons (From left to right)</p> <ul style="list-style-type: none"> <li>● Moyo Logo (Just a logo)</li> <li>● Home Icon</li> <li>● Administration</li> <li>● Inventory</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul><br><ul style="list-style-type: none"> <li>• The system then prompts the user to enter values into the required fields.</li> </ul>                                   |
|  | <p><b>Step 4:</b> The user will enter the following details into the related inputs:</p> <ul style="list-style-type: none"> <li>• Date</li> <li>• Year</li> <li>• Total</li> <li>• Department</li> </ul> |  | <p><b>Step 5:</b> The system validates that all of the fields have been filled in and is in the required format:</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 Characters and cannot contain numbers.</li> <li>• Total: must be in numbers only and must be positive.</li> <li>• Year: Must be a valid 4 number date.</li> </ul> |
|  |  |  | <p><b>Step 6:</b> The system enables the “Save Button” by using angular validation once all of the fields have met the requirements [ALT]</p>  |
|  | <p><b>Step 7:</b> The user clicks on the “Save Button” [ALT]</p>   |  | <p><b>Step 8:</b> The system then calls the AddBugetAllocation() Function in the angular frontend which will</p>   |

|  |  |   |
|--|--|---|
|  |  | <p>capture all of the values provided in the angular form.</p> <p>Inside this function function will be called that validates the budget allocation. The function will communicate with the appropriate header in the API with an HTTPGET Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Total</li> <li>• Year</li> <li>• Department</li> </ul> <p>The function will check the <b>Budget_Allocation Entity</b> with an <u><a href="#">SQL Read query</a></u> to ensure that there is no budget allocation with the corresponding total, year and department associated with it</p> <p>The function will return null if no budget allocation is found [ALT]</p> |
|  |  | <p><b>Step 9:</b> The angular frontend will take the response received from the API and check if it was null or not. The angular frontend will then call the AddBudgetAllocation() function which will</p>  |

|                           |  |   |   |
|---------------------------|--|---|---|
|                           |  |   | <p>communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Budget Allocation Object.</p> <p>The API will then receive the Budget Allocation object and add the following Details in the <b>Budget_Allocation entity with a SQL Insert Query:</b></p> <ul style="list-style-type: none"> <li>• Date in the <i>Date_Created</i> attribute</li> <li>• Total in the <i>Total</i> attribute</li> <li>• Year in the <i>Year</i> attribute</li> </ul> <p><b>[ALT]</b></p> |
|                           |  |   | <p><b>Step 10:</b> The system routes the user to the <u><i>ViewBudgetAllocation</i></u> Screen</p>  |
| <b>Alternate courses:</b> |  | <p><b>[ALT] Step 6:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. Return to step 4</p>   |   |
|                           |  | <p><b>[ALT] Step 7:</b> The user clicks on the “Cancel” Button. The system routes the user to the <u><i>ViewBudgetAllocation</i></u> Screen and the use case ends</p>   |   |
|                           |  | <p><b>[ALT] Step 8:</b> The function will retrieve the existing budget allocation object if a budget allocation is found. Proceed to Step 9</p>   |   |
|                           |  | <p><b>[ALT] Step 9:</b> The angular frontend receives the object instead of the null value. A Error notification will be generated which has the following Details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: The budget allocation Exists</li> </ul> |   |

|  |  |
|--|--|
|  | <ul style="list-style-type: none"> <li>• Body: Label: The allocation [Total] for year [Year] and [Department] ALREADY EXISTS!.</li> <li>• The user will be Redirected to the <u><a href="#">AddBudgetAllocation</a></u> screen.</li> </ul> <p>The Placeholders will be populated by the budget allocation object sent.</p> <p>Return to step 4</p> |
| <b>Conclusion:</b>                                   | The system routes the user to the <u><a href="#">ViewBudgetCategory</a></u> Screen   |
| <b>Post-condition:</b>                               | A new budget allocation is added to the system and the database is updated   |
| <b>Business rules</b>                                | A budget allocation cannot be duplicated in the same category  |
| <b>Implementation constraints and specifications</b> | None   |
| <b>Assumptions:</b>                                  | None   |
| <b>Open issues:</b>                                  | None   |

*Figure 124 4.3 Create Budget Allocation*

| Use case name:                 | View Budget Allocations   | USE CASE TYPE                                      |
|--------------------------------|---|--|
| Use case id:                   | 4.4   | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | High  | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | User  |  |
| Primary system actor           | None  |  |
| Other participating actors:    | None  |  |
| Other interested stakeholders: | None  |  |
| Description:                   | This use case describes the event where the user would like to view all of the budget allocations saved on the system. The use case begins when the admin navigates to the Finance tab by clicking on the Finance |  |

|                                      | <p>button on the top navbar, and then clicks on the “Budget Allocations” button on the side Administration navigation. The system will then load the “ViewBudgetAllocation” screen along with all of the budget allocations saved on the system. The user will provide their search criteria and the table contained on the “ViewBudgetAllocation” screen will dynamically be adjusted to reflect the filtered results.</p> |                      |  |
|--------------------------------------|---|----------------------|--|
| <b>Pre-condition:</b>                | The user has to be logged in to the system.   |                      |  |
| <b>Trigger:</b>                      | The user requests to view the Budget Allocations saved on the system.   |                      |  |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   |                      | <b>System Response</b>   |
|                                      |   | <b>Manual Action</b> | <b>Automated Action</b>  |
|                                      | <p><b>Step 1:</b> The user navigates from the home screen to the Budget Allocations screen by clicking on the <b>Finance</b> button on the top <u>user</u> navbar and then the <b>Budget Allocations</b> button on the side <u>Finance</u> navbar</p>   | <p><b>None</b></p>   | <p><b>Step 2:</b> The system calls the <u>GetBudgetAllocations()</u> function in the angular frontend that communicates to the appropriate header in the ASP.NET core API by means of an HTTPGET request. This function will get the following data from the <b>Budget Allocation</b> entity by means of an <u>SQL Read Query</u>:</p> <ul style="list-style-type: none"> <li>• Budget_ID</li> <li>• Department_ID</li> <li>• Date_Created</li> <li>• Year</li> <li>• Total</li> </ul> <p>The system will also include the following details from the <b>Department</b> Entity where the Department_ID in the <b>Department</b> entity is equal to the <b>Department ID</b> in the</p> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <b>Budget_Allocation</b><br>entity:<br><ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> |
|--|--|--|--|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <p><b>Step 3:</b> The system will display the “View Budget Allocations” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Label containing text: “Manage Budget Allocations”</li> <li>• Button: Create Allocation</li> <li>• Search Input: Label Containing “Search by Department”, Placeholder</li> <li>• <b>Table Containing:</b></li> <li>• Column headers: No. , Department, Date, Year, Total</li> <li>• Records with the following buttons: Budget Lines Button, Edit button, Delete Button</li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> </ul> |
|--|--|------|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>Notification Bell Icon</li> <li>Profile Icon Dropdown</li> </ul> <p><b>Side Navbar:</b></p> <ul style="list-style-type: none"> <li>“Budget Categories” tab</li> <li>“Budget Allocations” tab</li> </ul>   |
|  |  |  | <p><b>Step 4:</b> The system will then populate the searchedBudgetAllocations array and the BudgetAllocations array with the data retrieved from the database by means of the GetBudgetAllocations() Function.. The angular material table will be populated with the data contained in the searchedBudgetAllocations array. [ALT]</p>                         |
|  | <p><b>Step 5:</b> The user will provide the following criteria inside the Search input to filter through the budget allocations displayed on the page:</p> <ul style="list-style-type: none"> <li>Year</li> </ul> <p>[ALT]</p> |  | <p><b>Step 6:</b> The angular frontend will then filter through all of the data in the searchedBudgetAllocations array by means of a Linq Lambda function to retrieve any record which includes a letter contained in the Search input. The angular frontend will then adjust the table dynamically with each KeyUp Event and display it to the user.[ALT]</p> |

|  |  |
|--|--|
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4a:</b> There were no budget allocation entries to retrieve. The Table will not be populated and left blank. The user will create a budget allocation (<b>Invoke UC 4.3 Create Budget Allocation</b>) and return to step 1</p> <p><b>[ALT] Step 4b:</b> There were no Budget Allocation entries to retrieve. The Table will not be populated and left blank. The use case ends</p> <p><b>[ALT] Step 5:</b> The user has not provided any search criteria. Return to step 4. The use case ends</p> |
|  | <p><b>[ALT] Step 6a:</b> No Budget Allocations were found containing the search criteria.</p> <p>The user will create a Budget Allocation (<b>Invoke UC 4.7: Create Budget Allocation</b>) and return to step 1</p> <p><b>[ALT] Step 6b:</b> No budget allocations were found containing the search criteria. The Table will not be populated and left blank. The use case ends</p>  |
| <b>Conclusion:</b>                                   | The system displays the filtered data to the user  |
| <b>Post-condition:</b>                               | The matching budget allocation records containing the search criteria is displayed to the user.  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Figure 125 4.4 View Budget Allocation

|   |                   |                |
|---|-------------------|----------------|
| <p style="text-align: center;"><b>System Name: Procion System</b></p> <p style="text-align: center;"><b>Sub-System: 4</b></p> |                   |                |
| Author: Bupe Chindongo  | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |  |                        |   |
|---------------------------------------|--|------------------------|---|
| <b>Use case name:</b>                 | Update Budget Allocation   |                        | <b>USE CASE TYPE</b>  |
| <b>Use case id:</b>                   | 4.5  |                        | Business Requirements: <input type="checkbox"/>                         |
| <b>Priority:</b>                      | High   |                        | System Analysis: <input type="checkbox"/>                               |
| <b>Source:</b>                        | Moyo   |                        | System Design: <input checked="" type="checkbox"/>                      |
| <b>Primary business actor</b>         | User   |                        |   |
| <b>Primary system actor</b>           | None   |                        |   |
| <b>Other participating actors:</b>    | None   |                        |   |
| <b>Other interested stakeholders:</b> | None   |                        |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to update an existing budget allocation. The use case starts when the user clicks on the edit button next to the budget allocation that they wish to update. The system will load the "<i>EditBudgetAllocation</i>" screen with the details of the budget allocation that they wish to update. The user will provide the details to edit and the system will continuously validate these details. The user will click on the <b>Save</b> button. The system will validate the details and save the budget allocation. The use case ends when the user is rerouted to the "<i>ViewBudgetAllocation</i>" screen</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>There must be a budget allocation stored on the system</p>   |                        |   |
| <b>Trigger:</b>                       | The user requests to update a budget allocation.   |                        |   |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |  |                        | <b>Step 1:</b> The system invokes use case 4.4 "View Budget Allocation" |

|  |  |      |   |
|--|--|------|---|
|  | <b>Step 2:</b> The user requests to update a budget allocation by clicking on the “Edit” Button located on the “View Budget Allocation” Screen | None | <b>Step 3:</b> The system will use the Angular frontend to navigate to the “EditBudgetAllocation” page while sending the ID of the budget allocation to be edited to that respective page |
|--|--|------|---|

|  |  |      |  |
|--|--|------|--|
|  |  | None | <p><b>Step 4:</b> The system will capture the ID that was passed via the header of the angular frontend and call the GetBudgetAllocationByID() function along with the GetDepartments() function. These functions will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The GetBudgetAllocationByID() function will use the ID variable and return the first object where the <b>ID variable</b> is equal to the <b>Budget_ID attribute</b> in the <b>Budget Allocation</b> Entity. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Budget_ID</li> <li>• Department_ID</li> <li>• Date_Created</li> <li>• Year</li> </ul> |
|--|--|------|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Total</li> </ul> <p>The GetDepartments() method will get the departments in the <b>Department</b> entity and retrieve the following by making use on an SQL read:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>This will be populated in the BudgetAllocationArray</p>  |
|  |  |  | <p><b>Step 5:</b> The system will load the “<i>EditBudgetAllocation</i>” screen by making use of the angular frontend with the following:</p> <ul style="list-style-type: none"> <li>• Header label: Edit Budget Allocation</li> <li>• Date Label with the corresponding datepicker input</li> <li>• Total label with the corresponding input</li> <li>• Year label with the corresponding input</li> <li>• Department label with the corresponding dropdown</li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p>The system then populates the abovementioned fields with the data retrieved from the ASP.NET Core API, with the total, year, department and date fields being populated values retrieved from the SQL read.</p>  |
|  | <p><b>Step 6:</b> The user provides some or all of the following details they wish to modify:</p> <ul style="list-style-type: none"> <li>• Date</li> <li>• Total</li> <li>• Year</li> <li>• Department</li> </ul> <p>[ALT]</p> |  | <p><b>Step 7:</b> The angular frontend will validate each of the input fields (Which are all required unless stated otherwise) against the following criteria:</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 Characters and cannot contain numbers.</li> <li>• Total: must be in numbers only and must be positive.</li> <li>• Year: Must be a valid 4 number date.</li> </ul> |
|  |  |  | <p><b>Step 8:</b> The system enables the “<b>Save Button</b>” by using angular validation once all of the fields have met the requirements</p> <p>[ALT]</p>  |

|  |   |  |   |
|--|---|--|---|
|  | <p><b>Step 9:</b> The user clicks on the <b>Save</b> button [ALT]</p> |  | <p><b>Step 10:</b> The system then calls the <code>onSubmit()</code> Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function function will be called that validates the budget allocation. The function will communicate with the appropriate header in the API with an <code>HTTPGET</code> Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Total</li> <li>• Year</li> <li>• Department Name</li> </ul> <p>The function will check the <b>Budget_Allocation Entity</b> with an <u><a href="#">SQL Read query</a></u> to ensure that there is no budget allocation with the corresponding total, year, and department name.</p> <p>The function will retrieve null if no budget allocation is found [ALT]</p> |
|  |   |  | <p><b>Step 11:</b> The angular frontend will take the response received from the API and check if it</p>  |

|                           |  |  |  |
|---------------------------|--|--|--|
|                           |  |  | <p>was null or not, if the response is identical to another budget allocation or if anything has changed with the budget category.. The angular frontend will then call the <code>UpdateBudgetAllocation()</code> function which will communicate with the appropriate header in the API by means of an <code>HTTPPUT</code> request and it will send it the <code>Budget_ID</code> and the <code>BudgetAllocation Object</code>.</p> <p>The API will then receive the <code>BudgetAllocation object</code> and add the following Details in the <b><code>Budget_Allocation entity with a SQL Read Query:</code></b></p> <ul style="list-style-type: none"> <li>● Date in the <code>Date_Created</code> attribute</li> <li>● Year in the <code>Year</code> attribute</li> <li>● Total in the <code>Total</code> attribute</li> </ul> |
|                           |  |  | <p><b>Step 12:</b> The system routes the user to the <u><code>ViewBudgetAllocation</code></u> Screen</p>   |
| <b>Alternate courses:</b> | <p><b>[ALT] Step 6:</b> The user has not provided any details to update and the records remains the same. Proceed to step 7.</p> |  |  |

|  |  |
|--|--|
|  | <b>[ALT] Step 8:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. Return to step 6.              |
|  | <b>[ALT] Step 9:</b> The user clicks on the “Cancel” Button. The user will be redirected to the “ViewBudgetAllocation” screen. The use case ends |
|  | <b>[ALT] Step 10:</b> The function will retrieve the existing budget allocation object if a budget allocation is found. Proceed to Step 11       |
| <b>Conclusion:</b>                                   | The system routes the user to the <u>ViewBudgetAllocation</u> Screen   |
| <b>Post-condition:</b>                               | An existing budget allocation is updated on the system   |
| <b>Business rules</b>                                | A budget allocation cannot be duplicated   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Figure 126 4.5 Update Budget Allocation

| <b>System Name: Procion System</b><br><b>Sub-System: 4</b> |                   |                |
|--|-------------------|----------------|
| Author: Bupe Chindongo                                     | Date: 13 May 2023 | Version: 1.0.0 |

|                               |                                 |  |
|-------------------------------|---------------------------------|--|
| <b>Use case name:</b>         | <b>Delete Budget Allocation</b> | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 4.6                             | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High                            | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo                            | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User                            |  |
| <b>Primary system actor</b>   | None                            |  |

|                                       |  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | <p>The user requests to delete the budget allocation. The use case starts when the user clicks on the <b>delete</b> button next to the budget allocation item that they wish to delete. The system will load the “DeleteBudgetAllocation” screen which contains an angular material dialog. The material dialog will ask for confirmation from the user to ask if they are sure that they wish to delete the specified budget allocation. The user will click on the <b>Yes</b> button, the request will be validated and the user will be redirected to the “ViewBudgetAllocation” screen</p> |                        |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>There must be atleast one budget allocation stored on the system</p>   |                        |  |
| <b>Trigger:</b>                       | The user requests to delete a budget allocation on the system  |                        |  |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       |  |                        | <b>Step 1:</b> Invoke UC 4.5 “View Budget Allocation”  |
|                                       | <b>Step 2:</b> The user requests to delete a budget allocation on the system by clicking on the “Delete” Button located on the “Budget Allocation” Screen  | <b>None</b>            | <ul style="list-style-type: none"> <li><b>Step 3:</b> The system will use the Angular frontend to navigate to the “DeleteBudgetAllocation” page while sending the ID of the budget allocation to be deleted to that respective page</li> </ul> |

|  |  |      |   |
|--|--|------|---|
|  |  | None | <b>Step 4:</b> The system will capture the ID that was passed via the header of the angular frontend and call the GetBudgetAllocationBy |
|--|--|------|---|

|  |  |  |
|--|--|--|
|  |  | <p>ID() function. This function will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The GetBudgetAllocationBy ID() function will use the ID variable and return the first object where the <b>ID variable</b> is equal to the <b>Budget_ID attribute</b> in the <b>Budget Allocation</b> Entity. The system will retrieve the budget allocation object with the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Date equal to <i>Date_Created</i> attribute</li> <li>• Total equal to <i>Total</i> attribute</li> <li>• Year equal to <i>Year</i> attribute</li> <li>• DepartmentID equal to <i>Department_ID</i></li> </ul> |
|  |  | <p><b>Step 5:</b> The system will generate the “DeleteBudgetAllocation” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Delete Budget Category</li> <li>• Label with text: “You are about to delete the budget allocation with total: [Total]. Are</li> </ul>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>you sure you want to continue?"</p> <ul style="list-style-type: none"> <li>• Buttons: Yes, Cancel</li> </ul> <p>The [Total] placeholder will be populated with the following received from the ASP.Net API in the <b>Budget Allocation</b> Entity:</p> <ul style="list-style-type: none"> <li>• Total</li> </ul>   |
|  | <p><b>Step 6:</b> The user clicks on the “Yes” button</p> <p>[ALT]</p> |  | <p><b>Step 7:</b> The system calls the OnConfirm() function which includes the DeleteBudgetAllocation () function inside of it. The angular frontend will pass the ID of the budget allocation which was captured to the appropriate header in the ASP.NET Core API by means of an HTTPDELETE request. The API will then remove the record from the system by means of Entity Framework and return the deleted allocation to the Angular Frontend</p> |
|  |  |  | <p><b>Step 8:</b> The angular frontend will then capture the response received from the API and generate the following notification:</p> <p>Header: “Delete Successfull”</p> <p>Label with text: “The category [Total] has</p>  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | been deleted successfully!"  |
|  |  |  | The Total placeholder will be populated with the Total that was captured on the Angular frontend From the ASP.Net Core API |
|  |  |  | <b>Step 9:</b> The system routes the user to the <u><a href="#">ViewBudgetAllocation</a></u> Screen                        |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 6:</b> The user clicks on the “Cancel” Button. The user will be redirected to the “ViewBudgetAllocation” screen. The use case ends |  |  |
| <b>Conclusion:</b>                                   | The system routes the user to the <u><a href="#">ViewBudgetAllocation</a></u> Screen   |  |  |
| <b>Post-condition:</b>                               | An existing budget allocation is removed from the system and the database is updated   |  |  |
| <b>Business rules</b>                                | A budget allocation cannot be duplicated   |  |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |  |

Figure 127 4.6 Delete Budget Allocation

USE CASE 5.1-5.8 &amp; 5.9 , 5.10 &amp; 1.1-1.4 &amp; 3.1-3.4

System Name: Procion System

Sub-System 5: Inventory

|                             |                   |                |
|-----------------------------|-------------------|----------------|
| Author: Jason van der Merwe | Date: 13 May 2023 | Version: 1.0.0 |
|-----------------------------|-------------------|----------------|

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | <b>Create Consumable</b>   |  |
| <b>Use case id:</b>                   | 5.1  | Business Requirements: <input type="checkbox"/>  |
| <b>Priority:</b>                      | High   | System Analysis: <input type="checkbox"/>  |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/>   |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to create a new consumable item. The user will input all of the relevant details that they wish for the consumable, the system will validate these details and the user will on the <b>Save</b> Button. The system will save the new consumable and redirect the user to the <u><a href="#">View Consumables</a></u> screen. The use case concludes</p> |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>The user has loaded the “View Consumables” screen</p> <p>The user has clicked on the <b>Create Consumable</b> button</p>   |  |
| <b>Trigger:</b>                       | The user requests to add a consumable item to the system   |  |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b>   |
|                                       |  | <b>Manual Action</b> <b>Automated Action</b>   |
|                                       | <b>None</b>  | <b>Step 1:</b> The system makes use of the angular frontend and routes the user to the “Create Consumable” |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>screen with the following Items:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Create Consumable</li> <li>• <b>Name Label</b> with a corresponding input</li> <li>• <b>Description</b> Label with a corresponding input</li> <li>• <b>Category Label</b> with a corresponding dropdown input</li> <li>• <b>Minimum Reorder Quantity label</b> with a corresponding input</li> <li>• <b>Maximum Reorder Quantity label</b> with a corresponding input</li> <li>• <b>On Hand</b> label with a corresponding input</li> <li>• Buttons: <b>Save, Cancel</b></li> </ul> <p>Top Navbar navigation buttons (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>The system will populate the category dropdown box by using the <b>GetCategories()</b> function in the angular frontend which will communicate with the appropriate header in the API ConsumableController. The system will then retrieve all of the categories stored in the <b><u>CONSUMABLE_CATEGORY ENTITY</u></b> by means of an <b>HTTPGET</b> request and an <b><u>SQL Read query</u></b>. The system will populate the angular dropdown with the following values from the <b><u>CONSUMABLE_CATEGORY ENTITY</u></b>:</p> <ul style="list-style-type: none"> <li>• Name.</li> </ul> |
|--|--|--|---|

|  |   |      |   |
|--|---|------|---|
|  | <p><b>Step 2:</b> The user will enter the following details into the related inputs:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On-Hand amount</li> <li>• Minimum Reorder Quantity</li> <li>• Maximum-Reorder quantity</li> <li>• Consumable Category</li> </ul> | None | <p><b>Step 3:</b> The system validates that all of the fields have been filled in and is in the required format:</p> <ul style="list-style-type: none"> <li>• <b>Name:</b> Must be between 3 and 32 Characters and can not contain numbers.</li> <li>• <b>Description:</b> Must be between 3 and 50 Characters, cannot contain symbols</li> </ul> |
|--|---|------|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• <b>Minimum Reorder Quantity:</b> Cannot contain negative numbers</li> <li>• <b>Maximum Reorder Quantity:</b> Cannot contain negative numbers</li> <li>• <b>On Hand:</b> Cannot contain negative numbers</li> </ul>  |
|  |  |  | <p><b>Step 4:</b> The system enables the “<b>Save</b>” Button by using angular validation once all of the fields have met the requirements <b>[ALT]</b></p>  |
|  | <p><b>Step 5:</b> The user clicks on the “<b>Save</b>” Button”<b>[ALT]</b></p> |  | <p><b>Step 6:</b> The system then calls the <b>AddConsumable()</b> Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function a function will be called that validates the consumable. The function will communicate with the appropriate header in the API with an HTTPGET Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Consumable category Name</li> </ul> <p>The function will check the <u>Consumable Entity</u> with an <u>SQL Read query</u> to ensure that there is no consumable with the corresponding name and category name equal to the Name and category name stored in the entity by retrieving the Consumable as well as including the Consumable_Category where the name is equal to the name variable passed by means of Linq Lambda.</p> <p>The function will retrieve null if no consumable is found [ALT]</p> |
|  |  |  | <p><b>Step 7:</b> The angular frontend will take the response received from the API and check if it was null or not. The angular frontend will then call the <b>AddConsumables()</b> function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the <b>Consumable Object</b>.</p> <p>The API will then receive the</p>  |

|  |  |   |
|--|--|---|
|  |  | <p><b>Consumable</b> object and add the following Details in the <b>Consumable</b> entity by means of entity framework:</p> <ul style="list-style-type: none"> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> <li>• On_Hand in the <i>On Hand</i> attribute</li> <li>• Minimum_Reorder_Quantity in the <i>Minimum_Reorder_Quantity</i> attribute</li> <li>• Maximum_Reorder_Quantity in the <i>Maximum_Reorder_Quantity</i> attribute</li> <li>• Consumable_Catetory_ID in the <i>Consumable_Catetory_ID</i> attribute</li> </ul> <p>[ALT]</p> |
|  |  | <p><b>Step 8:</b> The system generates the following success notification:</p> <ul style="list-style-type: none"> <li>• Header: Create Successful!</li> <li>• Body: Label: The consumable [Consumable] Has been added successfully!</li> </ul> <p>The notification closes after 1.75 seconds and the system invokes <b>UC</b></p>   |

|  |   |  |                              |
|--|---|--|------------------------------|
|  |   |  | <b>5.2: View Consumables</b> |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 4:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. The user will be notified with the appropriate labels below the inputs where the error occurs. These issues will need to be fixed before the user continues.</p> <p><b>[ALT] Step 5:</b> The user clicks on the “Cancel” Button. The system routes the user to the <u><a href="#">ViewConsumable</a></u> Screen and the use case ends</p> <p><b>[ALT] Step 6:</b> The function will retrieve the existing consumable object if a consumable is found. Proceed to Step 7</p>                          |  |                              |
|  | <p><b>[ALT] Step 7:</b> The angular frontend receives the object instead of the null value. A Error notification will be generated which has the following Details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: The consumable Exists</li> <li>• Body: Label: The consumable [Consumable] ALREADY EXISTS!.</li> <li>• The user will be Redirected to the <u><a href="#">AddConsumable</a></u> screen.</li> </ul> <p>The [Consumable] Placeholder will be populated by the consumable object sent</p> <p>The notification will be displayed for 1.75 seconds. After which the notification will close.</p> |  |                              |
| <b>Conclusion:</b>                                   | The system routes the user to the <u><a href="#">ViewConsumable</a></u> Screen  |  |                              |
| <b>Post-condition:</b>                               | A new consumable is added to the system and the database is updated   |  |                              |
| <b>Business rules</b>                                | A consumable cannot be duplicated in the same category  |  |                              |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |                              |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |                              |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |                              |

Figure 128 5.1 Create Consumable

**System Name: Procion System**
**Sub-System 5: Inventory**

|                             |                   |                |
|-----------------------------|-------------------|----------------|
| Author: Jason van der Merwe | Date: 13 May 2023 | Version: 1.0.0 |
|-----------------------------|-------------------|----------------|

|                                       |   |   |
|---------------------------------------|---|---|
| <b>Use case name:</b>                 | <b>View Consumables</b>   |   |
| <b>Use case id:</b>                   | 5.2   | Business Requirements: <input type="checkbox"/>   |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>   |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/>  |
| <b>Primary business actor</b>         | User  |   |
| <b>Primary system actor</b>           | None  |   |
| <b>Other participating actors:</b>    | None  |   |
| <b>Other interested stakeholders:</b> | None  |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to view all of the Consumables saved on the system. The Use case begins when the user clicks on the Inventory tab on the main top navbar. The system loads the “ViewConsumables” screen along with all of the consumables saved on the system. The user will provide their search criteria and the table contained on the “ViewConsumables” screen will dynamically be adjusted to reflect the filtered results.</p> |   |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.   |   |
| <b>Trigger:</b>                       | The user requests to view all of the Consumables saved on the system.   |   |
| <b>Typical course of events:</b>      | <b>Actor Action</b>   | <b>System Response</b>  |
|                                       |   | <b>Manual Action</b> <b>Automated Action</b>  |
|                                       | <b>Step 1:</b> The user requests to view all of the Consumables saved on the system. The user clicks on the   | <b>None</b><br><b>Step 2:</b> The system calls the function <u>GetConsumables()</u> in the angular frontend that communicates to the appropriate header |

|  |   |  |
|--|---|--|
|  | <p>Inventory tab on the main top navbar</p> | <p>in the ASP.NET core API by means of an HTTPGET request. This function will get the following data from the <b><u>Consumable</u></b> entity by means of an <b><u>SQL</u></b> Read Query:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity</li> <li>• Consumable_Category_ID</li> </ul> <p>The system will also INCLUDE the following details from the <b><u>CONSUMABLE_CATEGORY</u></b> Entity where the <b><u>Consumable_Category_ID</u></b> in the <b><u>Consumable</u></b> entity is equal to the <b><u>Consumable_Category_ID</u></b> in the <b><u>CONSUMABLE_CATEGORY</u></b> Entity by means of Linq Lamda:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The system will then populate the <b><u>searchedConsumables</u></b> array and the <b><u>Consumables</u></b> array with the data retrieved from the database by means of the <b><u>GetConsumables()</u></b> Function.. The angular material table will be</p> |
|--|---|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | populated with the data contained in the <b>searchedConsumables</b> array. |
|--|--|--|--|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <p><b>Step 3:</b> The system will display the “View Consumables” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Label containing text: “View Consumables”</li> <li>• Button: Create Consumable</li> <li>• Search Input: Label Containing “Search by Name”, Placeholder containing: “e.g. Water”</li> <li>• <b>Table Containing:</b></li> <li>• Column headers: Name, Description On Hand, Minimum Reorder Quantity, Maximum Reorder Quantity, Category</li> <li>• Records with the following buttons: <b>Edit</b> button, <b>Delete</b> Button</li> <li>• Paginator</li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> </ul> |
|--|--|------|---|

|   |   |  |   |
|---|---|--|---|
|   |   |  | <ul style="list-style-type: none"> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p><b>Side Navbar:</b></p> <ul style="list-style-type: none"> <li>• “View Consumables” tab</li> <li>• “View Consumable Categories” tab</li> <li>• “View assets” tab</li> </ul> |
|   | <p><b>Step 4:</b> The user will provide the following criteria inside the Search input to filter through the consumables displayed on the page:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>[ALT]</p> |  | <p><b>Step 5:</b> The angular frontend will then filter through all of the data in the <b>searchedConsumables</b> array by means of a Linq Lambda function to retrieve any record which includes a letter contained in the Search input. The angular frontend will then adjust the table dynamically with each KeyUp Event and display it to the user.</p>  |
| <p>[ALT] <b>Step 4a:</b> User clicks the “Update” Button – <b>Extend to use case 5.3 Update Consumable</b></p> <p>[ALT] <b>Step 4b:</b> User clicks the “Delete” button – <b>Extend to use case 5.4 Delete Consumable</b></p> <p>[ALT] <b>Step 4c:</b> User clicks the “Create Consumable” button – <b>Extend to use case 5.1 Create Consumable</b></p> <p>[ALT] <b>Step 4d:</b> User clicks the “Export Details” button – <b>Extend to use case 5.9 Export Inventory Details</b></p> |   |  |   |

|  |  |
|--|--|
|  | <b>[ALT] Step 4e:</b> User clicks the “Export Details” button – <b>Extend to use case 5.10 Complete Stock Take</b> |
| <b>Conclusion:</b>                                   | The system displays the filtered data to the user  |
| <b>Post-condition:</b>                               | The matching consumable records containing the search criteria is displayed to the user.                           |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Figure 129 5.2 View Consumables

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                               |                   |  |
|-------------------------------|-------------------|--|
| <b>Use case name:</b>         | Update Consumable | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 5.3               | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High              | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo              | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User              |  |
| <b>Primary system actor</b>   | None              |  |

|                                       |  |                        |   |
|---------------------------------------|--|------------------------|---|
| <b>Other participating actors:</b>    | None   |                        |   |
| <b>Other interested stakeholders:</b> | None   |                        |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to update an existing consumable item. The user will provide the details to edit and the system will continuously validate these details. The user will click on the <b>Save</b> button. The system will validate the details and save the consumable. The use case ends when the user is rerouted to the "ViewConsumable" screen</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>There must be at least one consumable stored on the system</p> <p>The user has loaded the "View Consumables" screen</p> <p>The user has clicked on the <b>Edit</b> button</p>  |                        |   |
| <b>Trigger:</b>                       | The user requests to update a consumable item.   |                        |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |  | <b>None</b>            | <p><b>Step 1:</b> The system will use the Angular frontend to navigate to the "EditConsumable" page while sending the ID of the consumable to be edited to that respective page</p> |

|  |  |      |  |
|--|--|------|--|
|  |  | None | <p><b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend and call the <b>GetConsumableByID()</b> function along with the <b>GetCategories()</b> function. These functions will access the appropriate</p> |
|--|--|------|--|

|  |  |   |
|--|--|---|
|  |  | <p>HTTPGET methods on the ASP.NET Core API.</p> <p>The <code>GetConsumableByID()</code> function will use the ID variable and return the first object where the <code>ID</code> variable is equal to the <code>Consumable_ID</code> attribute in the <u>Consumable</u> Entity.</p> <p>The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"><li>• Name equal to Name</li><li>• Description equal to Description</li><li>• On_Hand equal to On_Hand</li><li>• Minimum_Reorder_Quantity equal to Minimum_Reorder_Quantity</li><li>• Maximum_Reorder_Quantity equal to Maximum_Reorder_Quantity</li><li>• Consumable_Category_ID equal to Consumable_Category_ID</li></ul> <p>The <code>GetCategories()</code> function will get all of the categories in the <u>Consumable Category</u> entity and retrieve the following by making use of a <u>Sql Read</u>:</p> |
|--|--|---|

## Iteration 5: Design &amp; Development– Team 11

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>This will be populated in the <b>ConsumableCategory Array</b></p>   |
|  |  |  | <p><b>Step 3:</b> The system will load the “<i>EditConsumable</i>” screen by making use of the angular frontend with the following:</p> <ul style="list-style-type: none"> <li>• Header label: Edit Consumable</li> <li>• Name Label with the corresponding input</li> <li>• Description label with the corresponding input</li> <li>• Category label with the corresponding input</li> <li>• Minimum Reorder Quantity: label with the corresponding input</li> <li>• Maximum Reorder Quantity label with the corresponding input</li> <li>• On Hand label with the corresponding input</li> <li>• Buttons: Save, Cancel</li> </ul> |

|  |   |   |
|--|---|---|
|  |   | <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul>   |
|  |   | <p><b>Step 4:</b> The system then populates each of the values on the form with the values received.</p> <p>The angular frontend will populate the category value with the category that is equal to the category referenced in the <b><u>Consumable Entity</u></b> where the <b><u>Consumable_Category_ID</u></b> is equal to the <b><u>Consumable_Category_ID</u></b> in the <b><u>Consumable Category entity</u></b>.</p> <p>The rest of the dropdown will be populated by the <b>ConsumableCategory Array</b></p> |
|  | <p><b>Step 5:</b> The user provides some or all of the following details they wish to modify:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> | <p><b>Step 6:</b> The angular frontend will validate each of the input fields (Which are all required unless stated)</p>  |

|  |   |  |  |
|--|---|--|--|
|  | <ul style="list-style-type: none"> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity</li> <li>• Consumable_Category Name</li> </ul> |  | <p>otherwise) against the following criteria:</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 Characters and can not contain numbers.</li> <li>• Description: Must be between 3 and 50 Characters, cannot contain symbols</li> <li>• Minimum Reorder Quantity: Cannot contain negative numbers</li> <li>• Maximum Reorder Quantity: Cannot contain negative numbers</li> <li>• On Hand: Cannot contain negative numbers</li> </ul> |
|  |   |  | <p><b>Step 7:</b> The system enables the “<b>Save Button</b>” by using angular validation once all of the fields have met the requirements [ALT]</p>   |
|  | <p><b>Step 8:</b> The user clicks on the <b>Save</b> button [ALT]</p>   |  | <p><b>Step 9:</b> The system then calls the <b>updateConsumable()</b> Function in the angular frontend which will capture all of the values provided in the angular form.</p>  |

|  |  |   |
|--|--|---|
|  |  | <p>Inside this function a validation function will be called that validates the consumable. The function will communicate with the appropriate header in the API with an HTTPGET Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Consumable category Name</li> </ul> <p>The function will check the <b><u>Consumable</u></b> Entity with an <u>SQL Read query</u> to ensure that there is no consumable with the corresponding name and category name equal to the Name and category name associated with it</p> <p>The function will retrieve null if no consumable is found [ALT]</p> |
|  |  | <p><b>Step 10:</b> The angular frontend will take the response received from the API and check if it was null or not, if the response is identical to another Consumable or if anything has changed with the consumable.. The angular frontend will then call the <b>UpdateConsumable()</b> function which will</p>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the Consumable_ID and the <b>Consumable</b> Object.</p> <p>The API will then receive the <b>Consumable</b> object and add the following Details in the <b>Consumable entity with a SQL Read Query</b>:</p> <ul style="list-style-type: none"> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> <li>• On_Hand in the <i>On Hand</i> attribute</li> <li>• Minimum_Reorder_Quantity in the <i>Minimum_Reorder_Quantity</i> attribute</li> <li>• Maximum_Reorder_Quantity in the <i>Maximum_Reorder_Quantity</i> attribute</li> <li>• Consumable_Catégorie_ID in the <i>Consumable_Catégorie_ID</i> attribute</li> </ul> <p>[ALT]</p> |
|  |  |  | <p><b>Step 11:</b> The system, generates the following success notification:</p>  |

|                        |  |  |  |
|------------------------|--|--|--|
|                        |  |  | <ul style="list-style-type: none"> <li>• Header: Update Successful!</li> <li>• Body: Label: The consumable [Consumable] Has been Updated successfully!</li> </ul> <p>The notification closes after 1.75 seconds and the system invokes <b>UC 5.2: View Consumables</b> to refresh the <i>ViewConsumables</i> screen</p>  |
|                        |  |  | <p><b>[ALT] Step 7:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. The user will be notified with the appropriate labels below the inputs where the error occurs. These issues will need to be fixed before the user continues.</p>  |
|                        |  |  | <p><b>[ALT] Step 8:</b> The user clicks on the “Cancel” Button. The system invokes <b>UC 5.2: View Consumables</b>. The use case ends</p>  |
|                        |  |  | <p><b>[ALT] Step 9:</b> The function will retrieve the existing consumable object if a consumable is found. Proceed to Step 10</p>   |
|                        |  |  | <p><b>[ALT] Step 10a:</b> The consumable received by the ASP.NET Core API is identical to the object in the Angular frontend. The Angular frontend will make use of an angular material dialog to display the following information to the user:</p> <ul style="list-style-type: none"> <li>• Header: “Notification: No changes made”</li> <li>• Label with text: “No changes made to the consumable [Consumable]”</li> </ul> <p>The dialog will pop up for 1.75 seconds. The system invokes <b>UC 5.2: View Consumables</b></p> |
| <b>Conclusion:</b>     |  |  | <p>The system routes the user to the <u>ViewConsumable</u> Screen</p>  |
| <b>Post-condition:</b> |  |  | <p>An existing consumable is updated on the system</p>   |
| <b>Business rules</b>  |  |  | <p>A consumable cannot be duplicated in the same category</p>  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Figure 130 Update Consumable

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | <b>Delete Consumable</b>   | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 5.4  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High   | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | The user requests to delete the consumable. The system will load the “DeleteConsumable” screen which contains a angular material dialog. The material dialog will ask for confirmation from the user to ask if they are sure that they wish to delete the specified consumable. The user will click on the <b>Yes</b> button, the request will be validated and the user will be redirected to the “ViewConsumable” screen |  |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.<br><br>There must be at least one consumable stored on the system  |  |

|                                      | The user has loaded the “View Consumables” screen<br><br>The user has clicked on the <b>Delete</b> button |                        |   |
|--------------------------------------|---|------------------------|---|
| <b>Trigger:</b>                      | The user requests to delete a consumable on the system  |                        |   |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   | <b>System Response</b> |   |
|                                      |   | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                      |   | <b>None</b>            | <ul style="list-style-type: none"> <li>• <b>Step 1:</b> The system will use the Angular frontend to navigate to the “DeleteConsumable” page while sending the ID of the consumable to be deleted to that respective page</li> </ul> |

|  |  |      |  |
|--|--|------|--|
|  |  | None | <p><b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend and call the <b>GetConsumableByID()</b> function. This function will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The <b>GetConsumableByID()</b> function in the API will use the ID variable and return the first object where the <b>ID variable</b> is equal to the <b>Consumable_ID attribute</b> in the <b>Consumable Entity</b>.</p> |
|--|--|------|--|

|  |  |   |
|--|--|---|
|  |  | <p>The system will retrieve the <b>consumable</b> object with the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Name equal to Name</li> <li>• Description equal to Description</li> <li>• On_Hand equal to On_Hand</li> <li>• Minimum_Reorder_Quantity equal to Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity equal to Maximum_Reorder_Quantity</li> <li>• Consumable_Category_ID equal to Consumable_Category_ID</li> </ul> |
|  |  | <p><b>Step 3:</b> The system will generate the “DeleteConsumable” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Delete Consumable</li> <li>• Label with text: “You are about to delete the consumable: <b>[Consumable]</b>. Are you sure you want to continue?”</li> </ul>   |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <ul style="list-style-type: none"> <li>• Buttons: <b>Yes</b>, <b>Cancel</b></li> <li>• Paginator</li> </ul> <p>The [Consumable] placeholder will be populated with the following received from the ASP.Net API in the <b><u>Consumable</u></b> Entity:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul>  |
|  | <p><b>Step 4:</b> The user clicks on the “<b>Yes</b>” button</p> <p>[ALT]</p> |  | <p><b>Step 5:</b> The system calls the <b>OnConfirm()</b> function which includes the <b>DeleteConsumable()</b> function inside of it. The angular frontend will pass the ID of the consumable which was captured to the appropriate header in the ASP.NET Core API by means of an <b>HTTPDELETE</b> request. The API will then remove the record from the system by means of Entity Framework and return the deleted Consumable to the Angular Frontend</p> |
|  |   |  | <p><b>Step 6:</b> The angular frontend will then capture the response received from the API and generate the following notification within the mat-dialog:</p> <p>Header: “Delete Successfull”</p> <p>Label with text: “The consumable [Consumable] has</p>  |

|  |   |  |   |
|--|---|--|---|
|  |   |  | been deleted successfully!"   |
|  |   |  | The consumable placeholder will be populated with the Consumable Name that was captured on the Angular frontend From the ASP.Net Core API |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 4:</b> The user clicks on the “Cancel” Button. The dialog closes. The use case ends |  |   |
| <b>Conclusion:</b>                                   | The system routes the user to the <u><a href="#">ViewConsumable</a></u> Screen                    |  |   |
| <b>Post-condition:</b>                               | An existing consumable is removed from the system and the database is updated                     |  |   |
| <b>Business rules</b>                                | None  |  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |

Figure 131 5.4 Delete Consumable

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Create Consumable Category  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 5.5   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | The use case describes the event where the user will want to create a new consumable category.. The user will input all of the relevant details that they wish for the consumable category, the system will validate these details and the user will click the <u>Save</u> Button. The system |  |

|                                  | The system will save the new consumable category and redirect the user to the <u><a href="#">ViewConsumableCategories</a></u> screen. The use case concludes   |                      |   |
|----------------------------------|--|----------------------|---|
| <b>Pre-condition:</b>            | The user has to be logged in to the system.  |                      |   |
| <b>Trigger:</b>                  | The user requests to add a Consumable Category item to the system<br>The user has loaded the “ <u><a href="#">ViewConsumableCategories</a></u> ” screen<br>The user has clicked on the <b>Create Category</b> button |                      |   |
| <b>Typical course Of events:</b> |  | <b>Actor Action</b>  | <b>System Response</b>  |
|                                  |  | <b>Manual Action</b> | <b>Automated Action</b>   |
|                                  |  | <b>None</b>          | <p>Step 1: The system makes use of the angular frontend and routes the user to the “<i>CreateCategory</i>” page with the following items:</p> <ul style="list-style-type: none"> <li>• Header: Create Consumable Category</li> <li>• Name Label with a corresponding input</li> <li>• Description Label with a corresponding input</li> <li>• Buttons: <b>Save</b>, <b>Cancel</b></li> </ul> <p>Top Navbar navigation buttons (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> </ul> |

|  |  |      |   |
|--|--|------|---|
|  | <p><b>Step 2:</b> The user will enter the following details into the related inputs:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> | None | <p><b>Step 3:</b> The system validates that all of the fields have been filled in and is in the required format: (All fields are required unless stated otherwise)</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 Characters and can not contain numbers.</li> <li>• Description: Must be between 3 and 50 Characters, cannot contain symbols</li> </ul> |
|  |  |      | <p><b>Step 4:</b> The system enables the “<b>Save Button</b>” by using angular validation once all of the fields have met the requirements [ALT]</p>  |
|  | <p><b>Step 5:</b> The user clicks on the “<b>Save Button</b>” [ALT]</p>  |      | <p><b>Step 6:</b> The system then calls the <b>AddCategory()</b> Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function the <b>CategoryValidation()</b> function will be called that validates the consumable category. The function will communicate with the appropriate header in</p>                     |

|  |  |  |
|--|--|--|
|  |  | <p>the API with an HTTPGET Request and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The function will check the <b>Consumable_Category Entity</b> with an <u>SQL Read query</u> to ensure that there is no category with the corresponding name and Description equal to the Name and Description associated with it</p> <p>The function will retrieve null if no consumable is found [ALT]</p> |
|  |  | <p><b>Step 7:</b> The angular frontend will take the response received from the API and check if it was null or not. The angular frontend will then call the <b>CreateCategory()</b> function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the <b>Consumable_Category Object</b>.</p> <p>The API will then receive the</p>  |

|                           |  |   |  |
|---------------------------|--|---|--|
|                           |  |   | <p><b>Consumable_Categor</b>y object and add the following Details in the <b>Consumable_Categor</b>y entity by means of entity framework:</p> <ul style="list-style-type: none"> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> </ul> <p>[ALT]</p>  |
|                           |  |   | <p><b>Step 8:</b></p> <p>The system generates the following success notification:</p> <ul style="list-style-type: none"> <li>• Header: Create Successful!: Body: Label: The Category [Category] Has been added Successfully</li> </ul> <p>The system <b>invokes UC 5.6 View Consumable categories</b> to refresh the View Consumable Categories screen</p> |
| <b>Alternate courses:</b> |  | <p><b>[ALT] Step 4:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. The user will be notified with the appropriate labels below the inputs where the error occurs. These issues will need to be fixed before the user continues.</p> <p><b>[ALT] Step 5:</b> The user clicks on the “Cancel” Button. The system <b>invokes UC 5.6 View Consumable categories</b> and the use case ends</p> <p><b>[ALT] Step 6:</b> The function will retrieve the existing <i>Consumable_Category</i> object if a Category is found. Proceed to Step 7</p> |  |

|  |   |
|--|---|
|  | <p><b>[ALT] Step 7:</b> The angular frontend receives the object instead of the null value. A Error notification will be generated which has the following Details:</p> <ul style="list-style-type: none"> <li>• Header: ERROR: Category Exists</li> <li>• Body: Label: The consumable category: [Category] ALREADY EXISTS!.</li> <li>• The system <b>invokes UC 5.6 View Consumable categories.</b></li> </ul> <p>The [Category] Placeholder will be populated by the Consumable object sent</p> <p>Return to step 2</p> |
| <b>Conclusion:</b>                                   | The system routes the user to the <i>ViewConsumableCategories</i> Screen  |
| <b>Post-condition:</b>                               | A new consumable category is added to the system and the database is updated  |
| <b>Business rules</b>                                | A consumable category cannot be duplicated.   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

*Figure 132 5.5 Create Consumable Category*

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                       |                                 |  |
|-----------------------|---------------------------------|--|
| <b>Use case name:</b> | <b>View Consumable Category</b> | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>   | 5.6                             | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>      | High                            | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>        | Moyo                            | System Design: <input checked="" type="checkbox"/> |

|                                       |  |                        |   |
|---------------------------------------|--|------------------------|---|
| <b>Primary business actor</b>         | User   |                        |   |
| <b>Primary system actor</b>           | None   |                        |   |
| <b>Other participating actors:</b>    | None   |                        |   |
| <b>Other interested stakeholders:</b> | None   |                        |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to view all of the Consumable Categories saved on the system. The Use case begins when the user clicks on the <b>View Consumable Categories</b> tab on the side nav bar. The system loads the “ViewConsumableCategories” screen along with all of the Consumable Categories saved on the system. The user will provide their search criteria and the table contained on the “ViewConsumableCategories” screen will dynamically be adjusted to reflect the filtered results.</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>The system Displays the Inventory Side Nav Bar</p>   |                        |   |
| <b>Trigger:</b>                       | The user requests to view all of the Consumable categories saved on the system.  |                        |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       | <p><b>Step 1:</b> The user navigates to the Side Nav bar and clicks on the <b>View Consumable Categories</b> tab</p>   | <b>None</b>            | <p><b>Step 2:</b> The system calls the function <b>GetCategories()</b> in the angular frontend that communicates to the appropriate header in the ASP.NET core API by means of an HTTPGET request. This function will get the following data from the <b>Consumable Category</b> entity by means of an SQL Read Query:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Description</li> </ul> <p>This function will populate the <b>ConsumableCategories</b> and <b>SearchedConsumableCategories</b> arrays.</p> |
|--|--|--|--|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <p><b>Step 3:</b> The system will display the “ViewConsumableCategories” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Label containing text: “View Consumable Categories”</li> <li>• Button: Create Category</li> <li>• Search Input: Label Containing “Search by Name”, Placeholder containing: “e.g. Kitchen”</li> <li>• <b>Table Containing:</b></li> <li>• Column headers: Name, Description</li> <li>• Records with the following buttons: Edit button, Delete Button</li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> </ul> |
|--|--|------|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p><b>Side Navbar:</b></p> <ul style="list-style-type: none"> <li>• “View Consumables” tab</li> <li>• “View Consumable Categories” tab</li> <li>• “View assets” tab</li> </ul> <p>The system will then populate the <b>Searchedconsumable categories</b> array and the <b>consumablecategories</b> array with the data retrieved from the database by means of the <b>GetCategories()</b> Function.. The angular material table will be populated with the data contained in the <b>Searchedconsumable categories</b> array.</p> |
|  | <p><b>Step 4:</b> The user will provide the following criteria inside the Search input to filter through the consumables categories displayed on the page:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p>[ALT]</p> |  | <p><b>Step 5:</b> The angular frontend will then filter through all of the data in the <b>Searchedconsumable categories</b> array by means of a Linq Lambda function to retrieve any record which includes a letter contained in the Search input. The angular frontend will then adjust the table dynamically with each KeyUp Event and display it to the user.</p>   |

|  |  |
|--|--|
|  | <p><b>[ALT] Step 4a:</b> User clicks the “Update” Button – <b>Extend to use case 5.7 Update Consumable Category</b></p> <p><b>[ALT] Step 4b:</b> User clicks the “Delete” button – <b>Extend to use case 5.8 Delete Consumable Category</b></p> <p><b>[ALT] Step 4c:</b> Admin clicks the “Create Consumable” – <b>Extend to use case 5.5 Create Consumable Category</b></p> |
| <b>Conclusion:</b>                                   | The system displays the filtered data to the user  |
| <b>Post-condition:</b>                               | The matching consumable category records containing the search criteria is displayed to the user.  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

*Figure 133 5.6 View Consumable Category*

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                               |                            |  |
|-------------------------------|----------------------------|--|
| <b>Use case name:</b>         | Update Consumable Category | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 5.7                        | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High                       | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo                       | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User                       |  |

|                                       |  |                        |   |
|---------------------------------------|--|------------------------|---|
| <b>Primary system actor</b>           | None   |                        |   |
| <b>Other participating actors:</b>    | None   |                        |   |
| <b>Other interested stakeholders:</b> | None   |                        |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to update an existing consumable category. The system will load the “<i>EditConsumableCategory</i>” screen with the details of the consumable category that they wish to update. The user will provide the details to edit and the system will continuously validate these details. The user will click on the <b>Save</b> button. The system will validate the details and save the consumable category. The use case ends when the user is rerouted to the “<i>ViewConsumableCategories</i>” screen</p> |                        |   |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>There must be a consumable category stored on the system</p> <p>The user has loaded the “<i>ViewConsumableCategories</i>” screen</p> <p>The user has clicked on the <b>Edit</b> button</p>   |                        |   |
| <b>Trigger:</b>                       | The user requests to update a consumable category.   |                        |   |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |  | <b>None</b>            | <p><b>Step 1:</b> The system will use the Angular frontend to navigate to the “<i>EditConsumableCategory</i>” page while sending the ID of the consumable category to be edited to that respective page</p> |

|  |  |      |  |
|--|--|------|--|
|  |  | None | <b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend |
|--|--|------|--|

|  |  |   |
|--|--|---|
|  |  | <p>and call the <code>GetCategorybyID()</code> function. These functions will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The <code>GetCategorybyID()</code> function will use the <code>ID</code> variable and return the first object where the <code>ID</code> variable is equal to the <code>Consumable_Category_ID</code> attribute in the <code>Consumable_Categor y Entity</code>. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Name equal to Name</li> <li>• Description equal to Description</li> </ul> <p>This will be populated in the <code>CategoryToEdit</code> Object</p> |
|  |  | <p><b>Step 3:</b> The system will load the “<code>EditConsumableCateg ory</code>” screen by making use of the angular frontend with the following:</p> <ul style="list-style-type: none"> <li>• Header label: Edit Consumable Category</li> <li>• Name Label with the</li> </ul>  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>corresponding input</p> <ul style="list-style-type: none"> <li>• Description label with the corresponding input</li> </ul> <p><b>Main Navbar:</b> (From left to right)</p> <ul style="list-style-type: none"> <li>• Moyo Logo (Just a logo)</li> <li>• Home Icon</li> <li>• Administration</li> <li>• Inventory</li> <li>• Finance</li> <li>• Procurement</li> <li>• Vendor</li> <li>• Reports</li> <li>• Notification Bell Icon</li> <li>• Profile Icon Dropdown</li> </ul> <p>The system then populates each of the values on the form with the values received from the ASP.NET Core API Backend</p> |
|  | <p><b>Step 4:</b> The user provides some or all of the following details they wish to modify:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>[ALT]</p> |  | <p><b>Step 5:</b> The angular frontend will validate each of the input fields (Which are all required unless stated otherwise) against the following criteria:</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 Characters and can not contain numbers.</li> <li>• Description: Must be between 3 and 50 Characters, cannot contain symbols</li> </ul>  |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <b>Step 6:</b> The system enables the “ <b>Save Button</b> ” by using angular validation once all of the fields have met the requirements [ALT]   |
|  | <b>Step 7:</b> The user clicks on the <b>Save button</b> [ALT] |  | <p><b>Step 8:</b> The system then calls the <b>UpdateCategory()</b> Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function a validation function will be called that validates the consumable category. The function will communicate with the appropriate header in the API with an <b>HTTPGET Request</b> and send the following attributes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The function will check the <b>Consumable Category Entity</b> with an <b>SQL Read query</b> to ensure that there is no consumable with the corresponding name and category name equal to the Name and Description.</p> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | The function will retrieve null if no consumable_category is found [ALT]  |
|  |  |  | <p><b>Step 9:</b> The angular frontend will take the response received from the API and check if it was null or not, if the response is identical to another Consumable category or if anything has changed with the consumable category. The angular frontend will then call the <b>updateCategory()</b> function which will communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the Consumable_ID and the <b>Consumable Object</b>.</p> <p>The API will then receive the <b>ConsumableCategory ToEdit</b> object and add the following Details in the <b>Consumable_Categor y</b> entity with a SQL Read Query:</p> <ul style="list-style-type: none"> <li>• Name in the <i>Name</i> attribute</li> <li>• Description in the <i>Description</i> attribute</li> </ul> <p>[ALT]</p> |

|                        |  |  |   |
|------------------------|--|--|---|
|                        |  |  | <p><b>Step 10:</b> The system, generates the following success notification:</p> <ul style="list-style-type: none"> <li>• Header: Update Successful!</li> <li>• Body: Label: The consumable category [Category] Has been Updated successfully!</li> </ul> <p>The system <b>invokes UC 5.6 View Consumable categories</b> to refresh the View Consumable Categories screen</p> |
|                        | <p><b>[ALT] Step 5:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. The user will be notified with the appropriate labels below the inputs where the error occurs. These issues will need to be fixed before the user continues.</p>  |  |   |
|                        | <p><b>[ALT] Step 7:</b> The user clicks on the “Cancel” Button. The system <b>invokes UC 5.6 View Consumable categories</b>. The use case ends</p>   |  |   |
|                        | <p><b>[ALT] Step 8:</b> The function will retrieve the existing consumablecategorytoEdit object if a category is found. Proceed to Step 9</p>  |  |   |
|                        | <p><b>[ALT] Step 9a:</b> The consumablecategorytoEdit object received by the ASP.NET Core API is Identical to the object in the Angular frontend. The Angular frontend will make use of an angular material dialog to display the following information to the user:</p> <ul style="list-style-type: none"> <li>• Header: “Notification: No changes made”</li> <li>• Label with text: “No changes made to the category [Category]”</li> </ul> <p>The dialog will pop up for 1.75 seconds. The system <b>invokes UC 5.6 View Consumable categories</b>.</p> |  |   |
| <b>Conclusion:</b>     | The system routes the user to the <i>ViewConsumableCategories</i> Screen   |  |   |
| <b>Post-condition:</b> | An existing category is updated on the system  |  |   |

|  |  |
|--|--|
| <b>Business rules</b>                                | A category cannot be duplicated                          |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Figure 134 5.7 Update Consumable Category

| <b>System Name: Procion System</b><br><b>Sub-System 5: Inventory</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe  | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | <b>Delete Consumable Category</b>  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 5.8  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High   | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The user requests to delete the consumable category. The use case starts when the user clicks on the <b>delete</b> button next to the consumable category item that they wish to delete. The system will load the “DeleteConsumableCategory” screen which contains a angular material dialog. The material dialog will ask for confirmation from the user to ask if they are sure that they wish to delete the specified consumable. The user will click on the <b>Yes</b> button, the request will be validated and the user will be redirected to the “ViewConsumablecategory” screen</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.<br>There must be at least one consumable category stored on the system<br>The user has loaded the " <u>ViewConsumableCategories</u> " screen<br>The user has clicked on the <b>Delete</b> button |                        |   |
|--------------------------------------|--|------------------------|---|
| <b>Trigger:</b>                      | The user requests to delete a consumable on the system   |                        |   |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                      |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                      |  | <b>None</b>            | <p><b>Step 1:</b> The system will use the Angular frontend to navigate to the "<i>DeleteConsumableCategory</i>" page while sending the ID of the consumable to be deleted to that respective page</p> |

|  |  |      |  |
|--|--|------|--|
|  |  | None | <p><b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend and call the <b>CategoryValidation()</b> function which will loop through all of the records in the <b>Consumable Entity</b> and match the <b>Consumable_Category_ID</b> to the <b>Consumable_Category_ID</b> in the <b>Consumable_Categor y entity</b> and count all of the instances where they are equal. The</p> |
|--|--|------|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>function will return 0 if no instances are found.</p> <p>The <b>GetCategoryByID()</b> function. This function will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The <b>GetCategoryByID ()</b> function will use the ID variable and return the first object where the <b>ID variable</b> is equal to the <b>Consumable_Category_ID</b> attribute in the <b>Consumable_Categor y</b> Entity. The system will retrieve the consumable category object with the following by making use of a <b>SQL Read</b>:</p> <ul style="list-style-type: none"> <li>• Name equal to Name</li> <li>• Description equal to Description</li> </ul> |
|  |  |  | <p><b>Step 3:</b> The system will generate the “<i>DeleteConsumableCategory</i>” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Delete Category</li> <li>• Label with text: “You are about to delete the category: <b>[Category]</b>”</li> </ul>   |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <p><b>ory]. Are you sure you want to continue?"</b></p> <ul style="list-style-type: none"> <li>• Buttons: Yes, Cancel</li> </ul> <p>The <b>[Category]</b> placeholder will be populated with the following received from the ASP.Net API in the <b>Consumable Category</b> Entity:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> <p><b>[ALT]</b></p>  |
|  | <p><b>Step 4:</b> The user clicks on the "Yes" button</p> <p><b>[ALT]</b></p> |  | <p><b>Step 5:</b> The system calls the <b>OnConfirm()</b> function which includes the <b>DeleteConsumableCategory()</b> function inside of it. The angular frontend will pass the ID of the consumable category which was captured to the appropriate header in the ASP.NET Core API by means of an <b>HTTPDELETE</b> request. The API will then remove the record from the system by means of Entity Framework and return the deleted Consumable Category to the Angular Frontend</p> |
|  |   |  | <p><b>Step 6:</b> The angular frontend will then capture the response received from the API and generate the following notification:</p>   |

|                           |   |  |  |
|---------------------------|---|--|--|
|                           |   |  | <p>Header: “Delete Successfull”</p> <p>Label with text: “The category [Category] has been deleted successfully!”</p> <p>The Category placeholder will be populated with the Consumable Category Name that was captured on the Angular frontend From the ASP.Net Core API</p> |
|                           |   |  | <p><b>Step 7:</b> The system invokes UC 5.6 View Consumable categories to refresh the View Consumable Categories screen</p>  |
| <b>Alternate courses:</b> | <p><b>[ALT] Step 3:</b> The validation has retrieved a value greater than 0 meaning there are Consumables associated with the category to be deleted. The angular frontend will generate the following notification in the form of a angular material dialog:</p> <p>Header: “Error: Category in use”</p> <p>Label with text: “The category is associated with a consumable! Please remove the category from the consumable to continue with deletion”</p> <p>This notification will be displayed for a total of 4 seconds and afterwards the the modal will close. The use case ends</p> |  |  |
|                           | <p><b>[ALT] Step 4:</b> The user clicks on the “Cancel” Button. The modal closes.. The use case ends</p>  |  |  |
| <b>Conclusion:</b>        | The system routes the user to the <i>ViewConsumableCategories</i> Screen  |  |  |
| <b>Post-condition:</b>    | An existing consumable category is removed from the system and the database is updated  |  |  |
| <b>Business rules</b>     | A category cannot be deleted if it is associated with another record  |  |  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |

*Figure 135 5.8 Delete Consumable Category*

|   |                   |                |
|---|-------------------|----------------|
| <p style="text-align: center;"><b>System Name: Procion System</b></p> <p style="text-align: center;"><b>Sub-System:</b></p> |                   |                |
| Author: Jason van der Merwe   | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Export Inventory Details  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 5.9   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | <p>The use case describes the event where a user will want to generate a report which contains all of the details for consumable items stored on the system. The user will click on the Export Details button located on the “View Consumables” screen. The system will get all of the consumables and consumable categories in the database. The pdf will be generated and displayed to a user in a new tab. The use case ends</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.   |                        |  |
|--------------------------------------|---|------------------------|--|
| <b>Trigger:</b>                      | The user requests to view all of the Consumable items stored on the system in a report  |                        |  |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   | <b>System Response</b> |  |
| Manual Action                        | Automated Action  |                        |  |
|                                      | <b>Step 1:</b> The user requests to view all of the Consumable items stored on the system in a report. The User clicks on the Export Details button | <p><b>None</b></p>     | <p><b>Step 2:</b> The system calls the function <u>GetConsumables()</u> in the angular frontend that communicates to the appropriate header in the ASP.NET core API by means of an HTTPGET request. This function will get the following data from the <u>Consumable</u> entity by means of an <u>SQL Read Query</u>:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity</li> <li>• Consumable_Category_ID</li> </ul> <p>The system will also INCLUDE the following details from the <u>CONSUMABLE_CATEGORY Entity</u> where the <u>Consumable_Category_ID</u> in the <u>Consumable</u> entity is equal to the <u>Consumable_Category_ID</u> in the</p> |

|                    |      |      |   |
|--------------------|------|------|---|
|                    |      |      | <p><b><u>CONSUMABLE CATE<br/>GORY Entity:</u></b></p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The Angular frontend will receive this data in the following array</p> <ul style="list-style-type: none"> <li>• Consumables[]</li> </ul>   |
|                    |      | None | <p><b>Step 3:</b> The angular frontend will generate a pdf using PDFMAKE as follows:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Consumable Inventory details Report</li> <li>• <b>Generated on:</b> [Date]</li> <li>• <b>Label:</b> “Details per consumable item”</li> <li>• <b>Table</b> with headers: Name, category. Minimum reorder quantity</li> <li>• Maximum reorder quantity</li> <li>• On hand</li> </ul> <p>The Rows of the table will be populated using the data contained in the Consumables[] array.</p> |
|                    |      |      | <p><b>Step 4:</b> The PDF is displayed to the user in a new tab</p>   |
| Alternate courses: | None |      |   |

|  |  |
|--|--|
| <b>Conclusion:</b>                                   | The system displays the Inventory details report to the user |
| <b>Post-condition:</b>                               | The report is generated containing the inventory details     |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>     |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>     |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>     |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>     |

*Figure 136 5.9 Export Inventory Details*

| <b>System Name: Procion System</b><br><b>Sub-System:</b> |                   |                |
|--|-------------------|----------------|
| Author: Jason van der Merwe                              | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |                     |  |
|---------------------------------------|---------------------|--|
| <b>Use case name:</b>                 | Complete Stock Take | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 5.10                | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High                | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo                | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User                |  |
| <b>Primary system actor</b>           | None                |  |
| <b>Other participating actors:</b>    | None                |  |
| <b>Other interested stakeholders:</b> | None                |  |

| <b>Description:</b>          | The use case describes the event where the user will want to update the consumable stock of an existing consumable item. The system will load the "UpdateConsumableStock" screen A input field to capture the new stock amount. The user will provide the details. The user will click on the <b>UpdateStock</b> button. The use case ends when the user is rerouted to the "ViewConsumable" screen and the on hand and stock amount is updated on the system |                 |  |
|------------------------------|---|-----------------|--|
| <b>Pre-condition:</b>        | <p>The user has to be logged in to the system.</p> <p>There must be at least one consumable stored on the system</p>  |                 |  |
| <b>Trigger:</b>              | The user requests to update a consumable item.  |                 |  |
| Typical course<br>Of events: | Actor Action  | System Response |  |
|                              |   | Manual Action   | Automated Action   |
|                              |   | <b>None</b>     | <p><b>Step 1 :</b>The system will capture the consumable Name and ID sent through the Header route in the following variables:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• ID</li> </ul> <p>The system will load the "UpdateConsumableStock" screen by making use of the angular frontend with the following:</p> <ul style="list-style-type: none"> <li>• <b>Tabs:</b> Update Stock, View Predictions</li> <li>• <b>Header label:</b> Update [ConsumableName]</li> <li>• <b>Label:</b> "Please enter the stock level for the relevant consumable <b>[Name]:</b>" with its</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>corresponding input</p> <ul style="list-style-type: none"> <li><b>Buttons:</b> Update Stock, Cancel</li> </ul> |
|--|--|--|---|

|  |  |      |  |
|--|--|------|--|
|  | <p><b>Step 2:</b> The user provides the new stock amount</p>                         | None | <p><b>Step 3:</b> The angular frontend will validate each of the input fields (<b>Which are all required unless stated otherwise</b>) against the following criteria:</p> <ul style="list-style-type: none"> <li>Stock Level Input: Cannot contain negative numbers</li> </ul>                                       |
|  |  |      | <p><b>Step 4:</b> The system enables the “<b>Update Stock Button</b>” by using angular validation once all of the fields have met the requirements <b>[ALT]</b></p>  |
|  | <p><b>Step 5:</b> The user clicks on the <b>Update Stock</b> button <b>[ALT]</b></p> |      | <p><b>Step 6:</b> The system then calls the <b>UpdateStock()</b> Function in the angular frontend which will capture the value provided in the angular form.</p> <p>The function will then call the <b>GetConsumableByID()</b> function which will communicate with the ASP.NET Core API through the appropriate</p> |

|  |  |   |
|--|--|---|
|  |  | <p>headers by means of a HTTPGET request.</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• ID</li> </ul> <p>The API will retrieve the following from the <b>Consumable Entity</b> where the Consumable_ID attribute is equal to the ID variable passed by means of a SQL READ query:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> <li>• On_Hand</li> <li>• Minimum_Reorder_Quantity</li> <li>• Maximum_Reorder_Quantity</li> <li>• Consumable_Category_ID</li> </ul> <p>The Result will be returned as a Consumable object in the Angular frontend</p> |
|  |  | <p><b>Step 7:</b> The angular Frontend will call the <b>UpdateConsumableStock()</b> function which will communicate with the ASP.NET Core API through the appropriate headers by meand of a HTTPPOST Request.</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Stock Ammount</li> </ul>  |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Date</li> </ul> <p>The API will receive the data and create a new entry in the <b>Consumable_History</b> entity which contains the following details</p> <ul style="list-style-type: none"> <li>• History_ID</li> <li>• Consumable_ID</li> <li>• StockAmt</li> <li>• DateCaptured</li> </ul> <p>The API will update the On-Hand attribute in the <b>Consumable</b> entity to the value of the StockAmt attribute in the <b>Consumable_History</b> entity</p> <p>The API will return the Updated <b>Consumable_History</b> as a object.</p> |
|  |  |  | <p><b>Step 8:</b> The system will add the information for the notification that will be generated in the Notification Hub for the relevant user to approve the procurement request.</p> <p>The angular frontend will call the <b>AddStock tNotification()</b> function which will call the appropriate function in the .NET CORE API through the appropriate</p>  |

|  |  |   |
|--|--|---|
|  |  | <p>headers by means of a HTTPPOST request.</p> <p>The Angular frontend will send the following data:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Send_Date</li> </ul> <p>The following details will be added:</p> <p>In the <b>Notification</b> entity:</p> <ul style="list-style-type: none"> <li>• NotificationID: Auto Incremented</li> <li>• Notification_Type_ID: Foreign key associated with the <b>Notification_Type</b> entity</li> <li>• Name: Name Received</li> <li>• Send_Date: Send date received</li> </ul> <p>In the <b>Notification_Type</b> Entity:</p> <ul style="list-style-type: none"> <li>• Notification_Type_ID: 18</li> <li>• Name: Prepopulated Name</li> <li>• Description: Prepopulated Description</li> </ul> <p>The added notification will return the Added notification to the Angular frontend</p> |
|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <b>Step 9:</b> The system invokes <b>UC 5.2: View Consumables</b> |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 5</b> The user clicks on the “Cancel” Button. The system invokes <b>UC 5.2: View Consumables</b> |  |   |
| <b>Conclusion:</b>                                   | The system routes the user to the <u><a href="#">ViewConsumable</a></u> Screen                                 |  |   |
| <b>Post-condition:</b>                               | The stock amount and on-hand amount is updated for an existing consumable                                      |  |   |
| <b>Business rules</b>                                | None   |  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |

Figure 137 5.10.Complete stock take

|  |                   |                |
|--|-------------------|----------------|
| <b>System Name:</b> Procion System<br><b>Sub-System:</b> |                   |                |
| Author: Jason van der Merwe                              | Date: 7 July 2023 | Version: 1.0.0 |

|                               |       |  |
|-------------------------------|-------|--|
| <b>Use case name:</b>         | Login | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 1.1   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User  |  |

|                                       |  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Primary system actor</b>           | None   |                        |  |
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | The use case describes the event a user wants to log in to the system. The user will be prompted to enter their username/ email and password on the login screen. The system will check the systems database for any temporary access levels for the user. The user's credentials will be validated in the system's database. The use case concludes once the system has generated a token for the user and the view page is displayed |                        |  |
| <b>Pre-condition:</b>                 | The user has loaded the “Login” screen   |                        |  |
| <b>Trigger:</b>                       | The user Requests to Login to the system.  |                        |  |
| <b>Typical course of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       |  | <b>None</b>            | <p><b>Step 1:</b> The system makes use of the angular frontend and routes the user to the “Login” screen with the following Items:</p> <ul style="list-style-type: none"> <li>• <b>Left Side Container:</b></li> <li>• Header: “Login”</li> <li>• Username or Email input</li> <li>• Password input</li> <li>• Buttons: Login</li> <li>• <b>Right Side Container:</b></li> <li>• Header: “Forgot Password?”</li> <li>• Buttons: Forgot Password</li> </ul> |

|  |  |      |   |
|--|--|------|---|
|  | <b>Step 2:</b> The user will enter the following | None | <b>Step 3:</b> The system validates that all of the |
|--|--|------|---|

|  |  |  |  |
|--|--|--|--|
|  | <p>details into the related inputs:</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Password</li> </ul> <p><b>[ALT]</b></p> |  | <p>fields are in the required format (all fields are required unless stated otherwise):</p> <ul style="list-style-type: none"> <li>• <b>Username or Email:</b> This field is required and cannot be left empty</li> <li>• <b>Password:</b> This field is required and cannot be left empty</li> </ul>  |
|  |  |  | <p><b>Step 4:</b> The system enables the “Login” Button by using angular material validation once all of the fields have met the requirements</p> <p><b>[ALT]</b></p>  |
|  | <p><b>Step 5:</b> The user clicks on the “Login” Button</p>  |  | <p><b>Step 6:</b> The system then calls the <b>LoginUser()</b> Function in the angular frontend which will capture all of the values provided in the angular form.</p> <p>Inside this function a function will be called that gets all of the temporary access levels inside of the database. The function will communicate with the <b>.NET Core API</b> through the appropriate Header by means of an <b>HTTPGET</b> request.</p> <p>The user’s credentials will be retrieved from</p> |

|  |  |   |
|--|--|---|
|  |  | <p>the systems database by means of a <i>LinqLambda</i> function which will generate a SQLREAD query that will retrieve all of the users details contained in the <b>User</b> entity where the Username provided is equal to the <b>Username</b> attribute.</p> <p>The Password provided by the user will be hashed in the API and compared to the users hashed password in the Password attribute of the User entity.</p> <p><b>The User entity contains the following details</b></p> <ul style="list-style-type: none"> <li>• User_ID</li> <li>• Role_ID</li> <li>• Username</li> <li>• Password</li> <li>• Profile_Picture</li> <li>• No_Notifications</li> </ul> |
|  |  | <p><b>Step 7:</b> The users details have been validated.</p> <p>The .Net Core API will use the details retrieved for the user and get the temporary access details from the information contained in the <b>Delegation_Of_Autho</b></p>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>rity entity. This is done through the foreign key of <b>User_ID</b> in the <b>Delegation_Of_Authority</b> entity which is linked to the <b>User_ID</b> in the <b>User</b> entity [ALT]</p>   |
|  |  |  | <p><b>Step 8:</b> The system retrieves the User and Delegation details.</p> <p>The system generates a token which stores the user's Username, Role and temporary access level. This token is used to manage the user's access and session timer.</p> <p>The token will be sent to the Angular frontend and saved in SessionStorage [ALT]</p>      |
|  |  |  | <p><b>Step 9:</b> The user is directed to the Home Page which contains the following details:</p> <p><b>Header:</b> Welcome Label</p> <p><b>Subheader:</b> Explore</p> <p><b>Tabs</b> with relevant Images:</p> <ul style="list-style-type: none"> <li>• Inventory</li> <li>• Administration</li> <li>• Procurement</li> <li>• Finance</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Vendor</li> <li>• Calender</li> <li>• Reports</li> <li>• Help</li> </ul> |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 2:</b> The user has forgot his password. The user clicks on the forgot Password Button. The system invokes <b>UC: 1.3 Forgot Password</b></p> <p><b>[ALT] Step 4:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. Return to step 2</p> <p><b>[ALT] Step 7:</b> The user's credentials are invalid. A error notification will be generated by means of Angular Material which includes the following details:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Login Failed</li> <li>• <b>Body Text:</b> Your Username/ Password is incorrect!</li> </ul> <p><b>Return to step 2</b></p> <p><b>[ALT] Step 8:</b> No Temporary access was found for the relevant user. The system generates token containing the user's Username and role. The token is sent to the angular frontend which is saved in session storage. Proceed to Step 9</p> |  |   |
| <b>Conclusion:</b>                                   | The system generates a token and routes the user to the “ <b>Home</b> ” screen   |  |   |
| <b>Post-condition:</b>                               | A token is generated for the user as they are logged in to the system.   |  |   |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |   |

Figure 138 1.1 Login

|  |                    |                |
|--|--------------------|----------------|
| <b>System Name:</b> Procion System<br><b>Sub-System:</b> |                    |                |
| Author: Jason van der Merwe                              | Date: 24 July 2023 | Version: 1.0.0 |

| Use case name:                 | Logout   | USE CASE TYPE   |
|--------------------------------|--|---|
| Use case id:                   | 1.2  | Business Requirements: <input type="checkbox"/>       |
| Priority:                      | High   | System Analysis: <input type="checkbox"/>             |
| Source:                        | Moyo   | System Design: <input checked="" type="checkbox"/>    |
| Primary business actor         | User   |   |
| Primary system actor           | None   |   |
| Other participating actors:    | None   |   |
| Other interested stakeholders: | None   |   |
| Description:                   | The use case describes the event where a user will want to log out of the system. The system captures the Logout request of the user. The user's token is destroyed and the user is redirected to the “Login” Screen |   |
| Pre-condition:                 | The user has to be logged in to the system.<br>The main nav bar has to be loaded   |   |
| Trigger:                       | The user requests to view all of the Consumables saved on the system.  |   |
| Typical course of events:      | <b>Actor Action</b><br><br><b>Manual Action</b>  | <b>System Response</b><br><br><b>Automated Action</b> |

|  |   |      |   |
|--|---|------|---|
|  | <p><b>Step 1:</b> The user requests to log out of the system by navigating to the profile icon on the main nav bar and clicks on the “Logout” dropdown</p> <p>[ALT]</p> | None | <p><b>Step 2:</b> The system calls the <b>Logout()</b> Function.</p> <p>The system will destroy the token and SessionTimer which is stored in the system’s SessionStorage</p> |
|--|---|------|---|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <b>Step 3:</b> The system will redirect the user to the <b>Login</b> Screen |
|  | <p><b>[ALT] Step 1:</b> The token is no longer valid. The token and the session timer will both be destroyed in the session storage and the user will be redirected to the “Login” screen</p>                      |      |   |
| <b>Conclusion:</b>                                   | The user is redirected to the “Login” screen   |      |   |
| <b>Post-condition:</b>                               | <ul style="list-style-type: none"> <li>• The User is Logged out of the system</li> <li>• The token in the session storage is destroyed</li> <li>• The session timer in the session storage is destroyed</li> </ul> |      |   |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>   |      |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |      |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |      |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |      |   |

Figure 139 1.2 Logout

|  |                    |                |
|--|--------------------|----------------|
| <b>System Name:</b> Procion System<br><b>Sub-System:</b> |                    |                |
| Author: Jason van der Merwe                              | Date: 24 July 2023 | Version: 1.0.0 |

| Use case name:                 | <b>USE CASE TYPE</b>   |   |  |
|--------------------------------|--|---|--|
| Use case id:                   | 1.3  | Business Requirements: <input type="checkbox"/>       |  |
| Priority:                      | High   | System Analysis: <input type="checkbox"/>             |  |
| Source:                        | Moyo   | System Design: <input checked="" type="checkbox"/>    |  |
| Primary business actor         | User   |   |  |
| Primary system actor           | None   |   |  |
| Other participating actors:    | None   |   |  |
| Other interested stakeholders: | None   |   |  |
| Description:                   | The use case describes the event where a user has forgot their password. The user will provide their email and click on the forgot password button. The system will validate the existence of the email and generate a new password. The new password will be emailed to the user. The use case ends once the email is sent to the user. |   |  |
| Pre-condition:                 | The Login Screen has to be loaded.   |   |  |
| Trigger:                       | The user clicks on the forgot password button.   |   |  |
| Typical course Of events:      | <b>Actor Action</b><br><br><b>Manual Action</b>  | <b>System Response</b><br><br><b>Automated Action</b> |  |
|                                |  |   |  |

|  |   |      |  |
|--|---|------|--|
|  | <b>Step 1:</b> The user clicks on the <b>Forgot Password</b> button located on the “Login” Screen | None | <b>Step 2:</b> The system generates the forgot password section on the Login screen which contains the following attributes: <ul style="list-style-type: none"> <li>• <b>Header:</b>Forgot Password?</li> <li>• <b>Email Input</b></li> <li>• <b>Buttons:</b> Forgot Password</li> </ul> |
|--|---|------|--|

|  |   |      |  |
|--|---|------|--|
|  | <b>Step 3:</b> The user provides their email                        | None | <b>Step 4:</b> The system will validate the users input which contains the following: <ul style="list-style-type: none"> <li>• <b>Email:</b> The system will validate this field as it can not be blank.</li> <li>• The email must be in email format</li> </ul>   |
|  |   |      | <b>Step 5:</b> The system enables the <b>Forgot Password</b> button. <span style="color: red;">[ALT]</span>  |
|  | <b>Step 6:</b> The user clicks on the <b>Forgot Password</b> button |      | <b>Step 7:</b> The system will call the <code>getEmployeebyEmail()</code> function in the angular frontend which will communicate with the API through the appropriate headers by means of a HTTPGET function and sends the following information <ul style="list-style-type: none"> <li>• <b>Email</b></li> </ul> |

|  |  |  |
|--|--|--|
|  |  | <p>The API will retrieve the following information from the <b>Employee</b> entity where the <u><b>Email</b></u> attribute in the <b>Employee</b> entity is equal to the Email variable passed through by means of a LinqLambda function which generates the appropriate SQLREAD query:</p> <ul style="list-style-type: none"> <li>• <b>PK:</b> <b>EmployeeID</b></li> <li>• <b>FK UserID</b></li> <li>• <b>FK:</b> <b>Department_ID</b></li> <li>• <b>FK: Branch_ID</b></li> <li>• <b>FK:</b> <b>Mandate_ID</b></li> <li>• <b>EmployeeName</b></li> <li>• <b>EmployeeSurname</b></li> <li>• <b>Cellphone_Number</b></li> <li>• <b>Email</b></li> </ul> <p>User_ID, Department_ID, Branch_ID and Mandate_ID will be populated by means of a LinqLambda Query as follows:</p> <ul style="list-style-type: none"> <li>• User_ID attribute in the Employee entity equal to the User_ID attribute in the User entity</li> <li>• Department_ID attribute in the Employee entity equal to the Department_ID</li> </ul> |
|--|--|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>attribute in the Department entity</p> <ul style="list-style-type: none"> <li>• Branch_ID attribute in the Employee entity equal to the Branch_ID attribute in the Branch entity</li> <li>• Mandate_ID attribute in the Employee entity equal to the Mandate_ID attribute in the Mandate_Limit entity</li> </ul> <p>The Api will send the Employee found by email as a object to the frontend.[ALT]</p> |
|  |  |  | <p><b>Step 8:</b> The system will generate the new randomized password in the Angular frontend and capture the following variables to be used in the Mailkit Tempelate,</p> <ul style="list-style-type: none"> <li>• Username (Received from the Employee object)</li> <li>• EmployeeName</li> <li>• Password</li> <li>• Email</li> </ul>  |
|  |  |  | <p><b>Step 9:</b> The system calls the UpdatePassword() function in the Angular fronted which will communicate with the Api through the</p>  |

|  |  |  |
|--|--|--|
|  |  | <p>appropriate Headers. The password will be received and Hashed and the Password attribute in the <b>User</b> Entity will be updated for the specific user by means of a HTTPPUT request. The following information will be sent:</p> <ul style="list-style-type: none"> <li>• <b>User_ID</b></li> <li>• <b>NewPassword</b></li> </ul>  |
|  |  | <p><b>Step 10:</b> The system calls the SendPasswordMail() function. This function will use the variables captured in step 7 and send the mail object to the .Net Core API backend through the appropriate headers by means of a HTTPPOST request.</p> <p>The .NetCore backend will build the email by means of a .csHtml Tempelate which contains the following</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Hi [EmployeeName]</li> <li>• ... Your password has been reset!</li> <li>• <b>Body Text:</b></li> <li>• “Please use the details below to log in to the system:</li> <li>• Username:[User name],</li> </ul> |

|  |   |  |   |
|--|---|--|---|
|  |   |  | <ul style="list-style-type: none"> <li>• Password: [Password]</li> <li>• Best Regards,</li> <li>• Moyo Support Team”</li> <li>• <b>Footer Text:</b> General disclaimer text</li> </ul>  |
|  |   |  | <p><b>Step 11: The system Sends the email</b> and notifies the user of the successful password update by means of an angular material dialog which contains the following elements:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Password Update Successful</li> <li>• <b>Body Text:</b> Your password has been updated successfully</li> </ul> <p>The notification will display for 1.75 Seconds</p> <p>The user will be redirected to the “<b>Login</b>” Screen</p> <p>[ALT]</p> |
|  | <p><b>[ALT] Step 6a:</b> No Employee was found. The system calls the GetAdminByEmail() Function which will communicate with the .Net Core Api through the appropriate header by means of a HTTPGET Function. The system will retrieve the Admin object to the Angular Frontend by means of a SQL READ query where the Email attribute in the Admin entity is equal to the email provided. Continue with <b>Step 7</b></p> |  |   |

|  |  |
|--|--|
|  | <p><b>[ALT] Step 6b:</b> No admin or Employee has been found. The system generates a unsuccessful notification by means of an Angular material dialog which contains the following attributes:</p> <p><b>Header:</b> “Reset Unsuccessful”</p> <p><b>Body text:</b> “User Does not exist, please try again”</p> <p>The use case concludes</p> |
| <b>Conclusion:</b>                                   | The system routes the user to the “ <u>Login</u> ” Screen  |
| <b>Post-condition:</b>                               | The user’s password is updated on the system   |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Figure 140 1.3 Forgot Password

| <p style="text-align: center;"><b>System Name: Procion System</b></p> <p style="text-align: center;"><b>Sub-System:</b></p> |                    |                |
|---|--------------------|----------------|
| Author: Jason van der Merwe   | Date: 24 July 2023 | Version: 1.0.0 |

|                               |                 |  |
|-------------------------------|-----------------|--|
| <b>Use case name:</b>         | Update Password | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 1.4             | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | High            | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo            | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User            |  |

|                                       |  |                        |  |
|---------------------------------------|--|------------------------|--|
| <b>Primary system actor</b>           | None   |                        |  |
| <b>Other participating actors:</b>    | None   |                        |  |
| <b>Other interested stakeholders:</b> | None   |                        |  |
| <b>Description:</b>                   | The use case describes the event where a user would like to update their password. The user will navigate to the View Profile Page on the Main Nav bar and click on the profile button. The user will click on the update password button. The user will be prompted to provide their old password and their new password. The user will be notified of their successful update and they will be logged out of the system and redirected to the Login page |                        |  |
| <b>Pre-condition:</b>                 | <p>The user has to be logged in to the system.</p> <p>The main navbar has to be displayed</p> <p>The ViewProfile Screen has to be loaded</p>   |                        |  |
| <b>Trigger:</b>                       | The user requests to update their password and clicks on the profile button  |                        |  |
| <b>Typical course of events:</b>      | <b>Actor Action</b>  | <b>System Response</b> |  |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>  |
|                                       |  | <b>None</b>            | <p><b>Step 1:</b> The system will send a request from the Angular frontend to the Data service where the service will make a HTTP Get request to the .NET Core backend which makes use of Lamda LINQ which will create a SQL Read query to retrieve the following data:</p> <p>From the <b>User</b> entity using User_Id:</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Profile_Picture</li> </ul> <p>The system populates the ProfilePicture variable in the angular</p> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | frontend with the<br><b>Profile_Picture</b><br>attribute received from<br>the API request |
|--|--|--|---|

|  |   |  |
|--|---|--|
|  | None  | <p><b>Step 2:</b> The system loads the Update Password screen by making use of the angular frontend containing the following details:</p> <ul style="list-style-type: none"> <li>• <b>Picture:</b> User Profile Picture</li> <li>• <b>Header:</b> Change your password!</li> <li>• <b>Subtitle:</b> “Please enter your current and new password in order to update it”</li> <li>• <b>Label:</b> “Current Password:” with its corresponding password input</li> <li>• <b>Label:</b> “New Password” with its corresponding password input</li> </ul> <p><b>Buttons:</b> Submit button, Cancel Button</p> |
|  | <p><b>Step 3:</b> The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Current Password</li> <li>• New Password</li> </ul> | <p><b>Step 4:</b> The system will validate the users input which contains the following:</p> <ul style="list-style-type: none"> <li>• <b>Current Password:</b> The Field is required</li> </ul>  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>and cannot be empty</p> <ul style="list-style-type: none"> <li>• <b>New Password:</b> The Field is required and cannot be empty</li> </ul> <p>[ALT]</p>   |
|  | <p><b>Step 5:</b> The user clicks on the Submit button [ALT]</p> |  | <p><b>Step 6:</b> The system captures the users input. The system captures the users username by retrieving it from the SessionStorage and calls the <b>VerifyCredentials()</b> method in the Angular frontend. This will communicate with the ASP.NET Core API through the appropriate headers using a <b>HTTPGET</b> request. The Angular frontend will send the following variables:</p> <ul style="list-style-type: none"> <li>• Username</li> <li>• Password</li> </ul> <p>The API will hash the password and compare the password sent to the <b>Password</b> variable in the <b>User</b> entity by means of a LinqLambda statement. The system returns the user object as a result</p> <p>[ALT]</p> |
|  |  |  | <p><b>Step 7:</b> The system calls the <b>UpdatePassword()</b> method in the Angular</p>   |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>frontend which will communicate with the ASP.NET Core Api through the appropriate headers by means of a <b>HTTPPUT</b> request. The function will send the following information:</p> <ul style="list-style-type: none"> <li>• UserID</li> <li>• NewPassword</li> </ul> <p>The API will update the <b>Password</b> variable in the <b>User</b> entity with the new hashed password where the <b>User_Id</b> variable in the <b>User</b> entity is equal to the UserID passed.</p> |
|  |  |  | <p><b>Step 8:</b> The system calls the <b>Logout()</b> function which destroys the token and the SessionTimer stored in Session storage.</p>   |
|  |  |  | <p><b>Step 9:</b> The system will display a successful update notification by making use of a Angular Material Dialog which will display the following:</p> <p><b>Header:</b> Update Successful</p> <p><b>Body text:</b> The password for [Username] has been updated successfully.</p>  |

|  |   |  |   |
|--|---|--|---|
|  |   |  | The notification will display for 1.75 seconds.<br><br>The user is redirected to the “Login” Screen |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 5:</b> The requirements for the fields have not been met. The user is notified using the appropriate error message under the respected input which is not valid. Return to step 4</p>  |  |   |
|  | <p><b>[ALT] Step 6:</b> The user clicks on the cancel button. Invoke <b>UC: 1.5 View Profile</b></p>  |  |   |
|  | <p><b>[ALT] Step 7:</b> The system has returned null indicating the password provided is incorrect. The system generates a unsuccessful update notification by using a Angular Material Dialog which contains the following details:</p> <p><b>Header:</b> Update Unsuccessful<br/><br/> <b>Body Text:</b> The password for [Username] does not match!.<br/><br/> The notification will be displayed for 1.75 seconds<br/><br/> Return to Step 3.</p> |  |   |
| <b>Conclusion:</b>                                   | The system routes the user to the <u>Login</u> Screen   |  |   |
| <b>Post-condition:</b>                               | The user's password is updated in the User entity   |  |   |
| <b>Business rules</b>                                | None  |  |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |   |

Figure 141 1.4 Update Password

|   |                   |                |
|---|-------------------|----------------|
| <p style="text-align: center;"><b>System Name:</b> Procion System</p> <p style="text-align: center;"><b>Sub-System:</b></p> |                   |                |
| Author: Jason van der Merwe   | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Create Procurement Request   | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 3.1  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High   | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The use case describes the event where a user would like to add a procurement request to the system. The user will click on the create request button located on the view procurement request screen. The screen will be populated and the user will provide their input which includes</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor Name</li> <li>• Procurement Quote</li> </ul> <p>The user will click on the save button. The use case concludes when the procurement request and the procurement quote is added to the systems database, the files are uploaded to the system and the success notification is displayed</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.<br>The user has loaded the " <u>ViewProcurementRequests</u> " screen<br>The user has clicked on the <b>Create Request</b> button |                        |   |
|--------------------------------------|--|------------------------|---|
| <b>Trigger:</b>                      | The user requests to add a Consumable Category item to the system  |                        |   |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>  | <b>System Response</b> |   |
|                                      |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                      |  | <b>None</b>            | <p><b>Step 1:</b> The system sets the frontend <b>VendorType</b> variable to approved which will do the following:</p> <p>The system loads the Approved vendor data by accessing the <b>GetVendors()</b> method in the Angular frontend which will communicate with the .NetCore API through the appropriate headers by means of a HTTPGET Function.</p> <p>The following data will be retrieved:</p> <p>From the <b>Vendor</b> Entity</p> <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Vendor_Status_ID</li> <li>• Name</li> <li>• Email</li> <li>• Number_Of_Times_Used</li> <li>• SoleSupplier_Provided</li> </ul> <p>From the <b>Vendor_Status</b> Entity</p> <ul style="list-style-type: none"> <li>• Vendor_Status_ID</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The system will return the vendors based on the condition that the Name attribute for the specific vendor is set to "Approved". The function will return the vendors as an array where the Vendor_Status_ID attribute in the Vendor entity is equal to the Vendor_Status_ID attribute in the Vendor_Status entity where the Name attribute in the Vendor_Status Entity is set to "Approved" by means of a LinqLambda statement.</p> <p>The vendors will be populated in the Angular frontend in the ApprovedVendors array</p> |
|--|--|--|--|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <p><b>Step 2:</b> The system loads the <b>CreateProcurementRequest</b> screen which contains the following:</p> <p><b>Header:</b> "Create Procurement Request" text</p> <p>Radio buttons:</p> <ul style="list-style-type: none"> <li>• Approved vendor</li> </ul> |
|--|--|------|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Other</li> </ul> <p>Inputs:</p> <ul style="list-style-type: none"> <li>• Request Name Label with the corresponding input</li> <li>• Request Description label with the corresponding input</li> <li>• Vendor Label with its corresponding Dropdown which will be populated with the data stored in the ApprovedVendors array</li> <li>• The procurement quote label with its corresponding file upload input field</li> <li>• Buttons</li> <li>• Save, Cancel</li> </ul> |
|  | <p><b>Step 3:</b> The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor</li> <li>• Procurement Quote</li> </ul> <p>[ALT]</p> |  |   |
|  |  |  | <p><b>Step 4:</b> The system validates the users input by using Angular Material Errors which is set to the following</p>   |

|  |   |  |   |
|--|---|--|---|
|  |   |  | <p>conditions (All fields are required unless stated otherwise):</p> <ul style="list-style-type: none"> <li>• Name: Must be between 3 and 32 characters</li> <li>• Description: Must be between 3 and 50 characters</li> <li>• Vendor: Cannot be left Unselected</li> <li>• Quote: Cannot be left empty</li> </ul> <p>[ALT]</p> |
|  |   |  | <p><b>Step 5:</b> The system enables the save Button</p>  |
|  | <p><b>Step 6:</b> The user clicks on the save button</p> <p>[ALT]</p> |  |   |
|  |   |  | <p><b>Step 7:</b> The system captures the data that the user has provided along with the Username which is stored in the token contained in Session Storage.</p> <p>The system will populate the ProcurementRequest</p>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>object with following values:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• VendorName</li> <li>• UserName</li> </ul>  |
|  |  |  | <p><b>Step 8 :</b> The system calls the <b>AddProcurementRequest()</b> function in the Angular frontend which will communicate with the .NetCore API through the appropriate headers by means of a HTTPPOST method</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• The procurement Request object captured object</li> </ul> <p>The API will retrieve the details from the <b>User</b>, <b>Vendor</b> and <b>Requisition_status</b> entities as follows:</p> <ul style="list-style-type: none"> <li>• The API will find the user by username through a SQLREAD query that is generated by a LinqLambda function which will return the User where the Username attribute in the <b>User</b> entity is equal to the</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>UserName variable sent.</p> <ul style="list-style-type: none"> <li>The API will find the Vendor by name through a SQLREAD query that is generated by a LinqLambda function which will return the Vendor where the Name attribute in the <b>Vendor</b> entity is equal to the Name variable sent.</li> <li>The API will find the Requisition Status by name through a SQLREAD query that is generated by a LinqLambda function which will return the Requisition Status where the Name attribute in the Requisition_Status entity is equal to the Name variable sent.</li> </ul> <p>The API will add the following to the database by means of a SQLCREATE query:</p> |
|  |  |  | <p><b>Step 9:</b> The API will use this data in order to create a new record for a procurement request.</p> <p>The following details will be added to the</p>   |

|  |  |  |
|--|--|--|
|  |  | <p>database by means of a LinqLambda function which generates a SQLCREATE query</p> <p>In the <b>Procurement_Request Entity</b>:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID: Incremented ID</li> <li>• Vendor_ID: Primary key in <b>Vendor Entity</b></li> <li>• Requisition_Status_ID: Primary key in <b>Requisition_Status Entity</b></li> <li>• User_ID: Primary key in <b>User Entity</b></li> <li>• Name: Name provided</li> <li>• Description: Description Provided</li> </ul> <p>The API will return the added Procurement Request as a result</p> |
|  |  | <p><b>Step 10:</b> The system captures the file uploaded in a File variable and calls the <b>AddFile()</b> method in the Angular frontend.</p> <p>The angular frontend will communicate with the .NET Core API through the appropriate headers by means of a HTTPPOST method.</p>  |

|  |  |  |
|--|--|--|
|  |  | <p>The following variables will be sent:</p> <ul style="list-style-type: none"> <li>• VendorName</li> <li>• File</li> </ul> <p>The API will receive the file and the Vendorname which will be stored in respective variables.</p> <p>A directory will be created based on the VendorName variable.</p> <p>A path will be generated which contains the following details:</p> <ul style="list-style-type: none"> <li>• Set-Directory: Files</li> <li>• Sub-Directory: Vendorname</li> <li>• Filepath: FileName</li> </ul> <p>The file will be added to the system and the path saved will be returned as a variable</p> |
|  |  | <p><b>Step 11:</b> The system will capture the following details in the Procurement_Quote Object in the Angular frontend:</p> <ul style="list-style-type: none"> <li>• The added procurement request</li> <li>• Path: File Path Recieved</li> <li>• PrefferedQuote: true</li> </ul>  |

|  |  |   |
|--|--|---|
|  |  | <p>The system will communicate with the .NET Core API through the appropriate headers by means of a <b>HTTPPOST</b> request.</p> <p>The API will create the new procurement quote as follows by means of a SQLCREATE query:</p> <p>In the <b>Procurement_Request_Quote</b> entity:</p> <ul style="list-style-type: none"> <li>• Quote_ID: Incremented new ID</li> <li>• Procurement_Request_ID: Procurement_Request_ID Received</li> <li>• Path: Path variable received</li> <li>• Upload_Date: Generated date of Today()</li> <li>• Preferred_Quote : Preferred_Quote variable received</li> </ul> <p>The system returns the added Procurement_Request_Quote</p> |
|  |  | <p><b>Step 12:</b> The system will add the information for the notification that will be generated in the Notification Hub for the relevant user to</p>   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>approve the procurement request.</p> <p>The angular frontend will call the <b>AddProcurementRequest()</b> function which will call the appropriate function in the .NET CORE API through the appropriate headers by means of a HTTPPOST request.</p> <p>The Angular frontend will send the following data:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Send_Date</li> </ul> <p>The following details will be added:</p> <p>In the <b>Notification</b> entity:</p> <ul style="list-style-type: none"> <li>• NotificationID: Auto Incremented</li> <li>• Notification_Type_ID: Foreign key associated with the <b>Notification_Type</b> entity</li> <li>• Name: Name Received</li> <li>• Send_Date: Send date received</li> </ul> <p>In the <b>Notification_Type</b> Entity:</p> <ul style="list-style-type: none"> <li>• Notification_Type_ID: 18</li> </ul> |
|--|--|--|---|

|                    |  |  |   |
|--------------------|--|--|---|
|                    |  |  | <ul style="list-style-type: none"> <li>• Name: Prepopulated Name</li> <li>• Description: Prepopulated Description</li> </ul> <p>The added notification will return the Added notification to the Angular frontend</p>   |
|                    |  |  | <p><b>Step 13:</b> The system will display a successful Create notification by making use of a Angular Material Dialog which will display the following:</p> <p><b>Header:</b> Create Successful</p> <p><b>Body text:</b> The procurement request for [Name] has been added successfully.</p> <p>The notification will display for 1.75 seconds.</p> <p>The system Invokes UC 3.2</p> <p><b>ViewProcurementRequest</b></p> <p>[ALT]</p> |
| Alternate courses: | <p><b>[ALT] Step 3:</b> The user clicks on the Other Radio button.</p> <p>The system loads the CreateProcurementRequest screen which contains the following:</p> <p><b>Header:</b> “Create Procurement Request” text</p> <p>Radio buttons:</p> |  |   |

|                    |   |
|--------------------|---|
|                    | <ul style="list-style-type: none"> <li>• Approved vendor</li> <li>• Other</li> </ul> <p>Inputs:</p> <ul style="list-style-type: none"> <li>• Request Name Label with the corresponding input</li> <li>• Request Description label with the corresponding input</li> <li>• Vendor Label with its corresponding input text field</li> <li>• The Preferred quote label with its corresponding file upload input field</li> <li>• The Comparative quote 1 label with its corresponding file upload input field</li> <li>• The Comparative quote 2 label with its corresponding file upload input field</li> <li>• Buttons: Save, Cancel</li> </ul> <p>The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor Name</li> <li>• Preferred quote</li> <li>• Comparative Quote 1</li> <li>• Comparative Quote 2</li> </ul> <p>Proceed to Step 4</p> |
|                    | <p><b>[ALT] Step 4:</b> The requirements for the fields have not been met. The user is notified using the appropriate error message under the respected input which is not valid. Return to step 3</p>  |
|                    | <p><b>[ALT] Step 6:</b> The user clicks on the cancel button. Invoke <b>UC: 3.2 View Procurement Request</b></p>  |
|                    | <p><b>[ALT] Step 13:</b> A error has occurred in the create procurement process</p> <p>The system will display a Unsuccess Create notification by making use of a Angular Material Dialog which will display the following:</p> <p><b>Header:</b> Create Unsuccessful</p> <p><b>Body text:</b> The procurement request for [Name] has Failed</p> <p>The notification will display for 1.75 seconds.</p> <p>Return to Step 3</p>   |
| <b>Conclusion:</b> | The system routes the user to the <i>ViewProcurementRequest</i> Screen  |

|  |   |
|--|---|
| <b>Post-condition:</b>                               | A new procurement request is added, The notification is added to the systems database, files are added and a new procurement request quote is added to the system |
| <b>Business rules</b>                                | A procurement request does not allow for the same file to be uploaded twice in one procurement request  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

*Figure 142 3.1 Create Procurement Request*

|  |                   |                |
|--|-------------------|----------------|
| <b>System Name: Procion System</b><br><b>Sub-System:</b> |                   |                |
| Author: Jason van der Merwe                              | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |                                 |  |
|---------------------------------------|---------------------------------|--|
| <b>Use case name:</b>                 | <b>View Procurement Request</b> | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 3.2                             | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High                            | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo                            | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User                            |  |
| <b>Primary system actor</b>           | None                            |  |
| <b>Other participating actors:</b>    | None                            |  |
| <b>Other interested stakeholders:</b> | None                            |  |

| <b>Description:</b>   | The use case describes the event where a user would like to view the procurement requests on the system. The use case begins where the user clicks on the Procurement Tab located on the main navigation bar. The system will load the “View Procurement Requests” screen that contains all of the procurement requests stored on the system.. They are then able to Search for Procurement Requests, Edit procurement requests and subsequently delete a procurement requests |                      |  |
|---|--|----------------------|--|
| <b>Pre-condition:</b>   | <p>The user has to be logged in to the system.</p> <p>The top navigation bar must be loaded</p>  |                      |  |
| <b>Trigger:</b>   | The user requests to view all of the Consumable categories saved on the system.  |                      |  |
| <b>Typical course<br/>Of events:</b>  |  | <b>Actor Action</b>  | <b>System Response</b>   |
|   |  | <b>Manual Action</b> | <b>Automated Action</b>  |
| <p><b>Step 1:</b> The user clicks on the <b>Procurement Tab</b> located on the main nav bar</p> |  | <b>None</b>          | <p><b>Step 2:</b> The system calls the functions <b>GetProcurementRequests()</b> and <b>GetProcurementRequestQuotes()</b></p> <p>This will communicate with the .Net Core API through the appropriate headers by means of a HTTPGET Response.</p> <p>The API will Return the following by means of a SQLREAD Query:</p> <p>From the <b>Procurement_Request Entity</b>:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Procurement_Request</li> <li>• Description</li> </ul> |

|  |  |  |
|--|--|--|
|  |  | <p>From the <b>Procurement_Request_Quote</b> entity:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>The data from the <b>Procurement_Request_Quote</b> entity will be retrieved where the Procurement_Request_ID attribute in the <b>Procurement_Request_Quote</b> entity is equal to the Procurement_Request_ID in the <b>Procurement_Request</b> Entity</p> <p>The data will be returned to the angular frontend and the following arrays will be populated:</p> <ul style="list-style-type: none"> <li>• ProcurementRequests[]: Populated with the Procurement_Request Data received from the API</li> <li>• ProcurementRequestQuotes[]: Populated with data received from the Procurement_Request_Quotes</li> </ul> |
|--|--|--|

|  |  |      |   |
|--|--|------|---|
|  |  |      | data received from the API  |
|  |  | None | <p><b>Step 3:</b> The system will display the “ViewProcurementRequest” screen with the following details:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Label containing text: “View Procurement Requests”</li> <li>• <b>Button:</b> Create Request</li> <li>• <b>Search Input:</b> Label Containing “Search by Name”, Placeholder containing: “e.g. Kitchen”</li> <li>• <b>Table Containing:</b> Column headers: Name, Description, User, Vendor, Status</li> <li>• Records with the following buttons: Edit button, Delete Button</li> <li>• Angular Material Paginator</li> </ul> <p><b>Side Navbar:</b></p> <ul style="list-style-type: none"> <li>• “View Procurement Requests” tab</li> <li>• “View Pending Procurement Requests” tab</li> <li>• “Place Procurement Request” tab</li> </ul> |

|   |  |  |   |
|---|--|--|---|
|   |  |  | <ul style="list-style-type: none"> <li>“View Flagged Procurement Requests” tab</li> </ul> <p>The system will then populate the table with the data contained in the <u>SearchedProcurementRequests</u> array</p>  |
|   | <p><b>Step 4:</b> The user will provide the following criteria inside the Search input to filter through the Procurement Requests displayed on the page:</p> <ul style="list-style-type: none"> <li>Name</li> </ul> <p>[ALT]</p> |  | <p><b>Step 5:</b> The angular frontend will then filter through all of the data in the <u>SearchedProcurementRequests</u> array by means of a LinqLambda function to retrieve any record which includes a letter contained in the Search input. The angular frontend will then adjust the table dynamically with each KeyUp Event and display it to the user.</p> |
| <p><b>ALT] Step 4a:</b> User clicks the “Update” Button – <b>Extend to use case 3.3 Update Procurement Request</b></p> <p><b>ALT] Step 4b:</b> User clicks the “Delete” button – <b>Extend to use case 3.4 Delete Procurement Request</b></p> <p><b>ALT] Step 4c:</b> Admin clicks the “Create Consumable” – <b>Extend to use case 3.1 Create Procurement Request</b></p> |  |  |   |
| <b>Conclusion:</b>  | The system displays the filtered data to the user  |  |   |
| <b>Post-condition:</b>  | The matching Procurement Request records containing the search criteria is displayed to the user.  |  |   |
| <b>Business rules</b>   | <ul style="list-style-type: none"> <li>None</li> </ul>   |  |   |
| <b>Implementation constraints and specifications</b>  | <ul style="list-style-type: none"> <li>None</li> </ul>   |  |   |

|                     |  |
|---------------------|--|
| <b>Assumptions:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Figure 143 3.2 View Procurement Request

|   |                   |                |
|---|-------------------|----------------|
| <p style="text-align: center;"><b>System Name: Procion System</b></p> <p style="text-align: center;"><b>Sub-System:</b></p> |                   |                |
| Author: Jason van der Merwe   | Date: 13 May 2023 | Version: 1.0.0 |

|                                       |   |  |
|---------------------------------------|---|--|
| <b>Use case name:</b>                 | Update Procurement Request  | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 3.3   | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo  | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User  |  |
| <b>Primary system actor</b>           | None  |  |
| <b>Other participating actors:</b>    | None  |  |
| <b>Other interested stakeholders:</b> | None  |  |
| <b>Description:</b>                   | <p>The use case describes the event where a user would like to update a procurement request. The system load the “EditProcurementRequest” screen and populate the inputs with the following details:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Vendor</li> <li>• Quote</li> <li>• Table containing the quotes related to the request</li> </ul> <p>The user will provide the details that they will like to change and click on the save button. The use case ends as the Procurement_Request is updated in the database, new replacement quotes are added and they are populated in the system’s database and the successful update notification is displayed and the user is redirected to the “ViewProcurementRequest” Page.</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.<br>There must be a Procurement request stored on the system<br>The user has loaded the “ViewProcurementRequest” screen<br>The user has clicked on the <b>Edit</b> button |  |  |
|--------------------------------------|--|--|--|
| <b>Trigger:</b>                      | The user requests to update a procurementRequest   |  |  |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>  | <b>System Response</b>   |  |
| <b>Manual Action</b>                 | <b>Automated Action</b>  |  |  |
|                                      | <b>None</b>  | <p><b>Step 1:</b> The System will receive the <b>ProcurementRequestID</b> and <b>VendorTypeName</b> variable sent through the route and store it in the ID and Name variable Respectively.</p> <p>The Angular frontend will call the <b>GetProcurementRequestsbyID()</b> method and the <b>GetProcurementRequestQuotesbyID()</b> method which will communicate with the ASP.NET Core API through the appropriate headers by means of a HTTPGET Request. The following data will be sent</p> <ul style="list-style-type: none"> <li>• ID</li> </ul> <p>The API get the following details from the database by making use of a SQLREAD query</p> |  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>generated by a LinqLambda function.</p> <p>The following details will be retrieved from the database:</p> <p>From the <b>Procurement_Request Entity</b>:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Procurement_Request</li> <li>• Description</li> </ul> <p>From the <b>Procurement_Request_Quote entity</b>:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>The data from the <b>Procurement_Request Entity</b> will be retrieved based on where the <b>Procurement_Request_ID</b> in the <b>Procurement_Request Entity</b> is equal to the ID variable sent through.</p> |
|--|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>The data from the <b>Procurement_Request_Quote</b> Entity will be retrieved based on the <i>Procurement_Request_ID</i> in the <b>Procurement_Request</b> Entity which is equal to the <i>Procurement_Request_ID</i> in the <b>Procurement_Request_Quote</b> Entity</p> <p>The <b>Procurement_Request</b> and <b>Procurement_Request_Quote</b> object will be returned.</p> |
|--|--|--|---|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <p><b>Step 3:</b> The Angular frontend will populate the following:</p> <p><b>Procurement_Request object:</b> This will be populated with the <b>Procurement_Request</b> result returned from the API</p> <p><b>Procurement_Request_Quote Array:</b> This will be populated with the <b>Procurement_Request_Quote</b> results returned from the API</p> |
|  |  |      | <p><b>Step 4:</b> The Angular frontend will generate the “<i>EditProcurementReq</i></p>   |

|  |  |   |
|--|--|---|
|  |  | <p>est" screen which contains the following details:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> "Edit Procurement Request"</li> <li>• <b>Label:</b> "Vendor Type: [Name]" (Populated with the Name variable received in step 1)</li> <li>• <b>"Request Name" Label</b> with its corresponding input</li> <li>• <b>"Request Description" Label</b> with its corresponding input</li> <li>• <b>"Vendor" Label</b> with its corresponding input</li> <li>• <b>"Request Quote" Label</b> with its corresponding file input</li> <li>• <b>Table</b> Containing headers: "Your Quotes", "Quote"</li> <li>• <b>Buttons:</b> Save, Cancel</li> </ul> <p>The Request Name, Request Description and Vendor inputs will be populated with the respective data stored in the <b>Procurement_Request</b> Object</p> |
|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | The table rows will be populated with the data stored in the <b>Procurement_Requests_Quote</b> array   |
|  | <p><b>Step 5:</b> The user provides some or all of the following details they wish to modify:</p> <ul style="list-style-type: none"> <li>• Request Name</li> <li>• Request Description</li> <li>• Replacement Quote</li> </ul> |  | <p><b>Step 6:</b> The angular frontend will validate each of the input fields (<b>Which are all required unless stated otherwise</b>) against the following criteria:</p> <ul style="list-style-type: none"> <li>• <b>Request Name:</b> Must be between 3 and 32 Characters and can not contain numbers.</li> <li>• <b>Request Description:</b> Must be between 3 and 50 Characters, cannot contain symbols</li> </ul> |
|  |  |  | <p><b>Step 7:</b> The system enables the “<b>Save Button</b>” by using angular validation once all of the fields have met the requirements [ALT]</p>   |
|  | <p><b>Step 8:</b> The user clicks on the <b>Save</b> button [ALT]</p>  |  | <p><b>Step 9:</b> The system then calls the <b>EditProcurementRequest()</b> Function in the angular frontend which will capture all of the values provided in the angular form. The form</p>   |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>Values will be captured as follows::</p> <ul style="list-style-type: none"> <li>• The updated procurement request is to be stored in the <b>UpdatedProcurementRequest Object</b></li> <li>• The updated files are to be stored in the <b>Files[] Array</b></li> </ul> <p>The function will communicate with the ASP.Net Core API through the appropriate headers by means of a HTTPPUT request. The following data will be sent:</p> <ul style="list-style-type: none"> <li>• <b>Procurement_Request_ID</b></li> <li>• <b>UpdatedProcurementRequest Object</b></li> </ul> <p>The api will retrieve the Procurement request to update based on the <b>Procurement_Request_ID</b> variable contained in the <b>Procurement_Request</b> object passed through by means of a <b>SQLREAD Query</b>.</p> <p>The following will be retrieved:</p> <p>From the <b>Procurement_Request Entity</b>:</p> <ul style="list-style-type: none"> <li>• <b>Procurement_Request_ID</b></li> </ul> |
|--|--|--|--|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Procurement_Request</li> <li>• Description</li> </ul> <p>The API will Update the <b>Procurement_Request Entity</b> record with the details contained in the <b>UpdatedProcurementRequest Object</b> passed through:</p> <p>The API will return the Updated <b>Procurement_Request Record</b> as a object</p>   |
|  |  |  | <p><b>Step 10:</b> The Angular frontend will call the <b>DeleteProcurementRequestFiles()</b> function which will loop through all of the file paths contained in the <b>Procurement_Quotes[]</b> array and it will capture the following:</p> <ul style="list-style-type: none"> <li>• <b>Vendorname</b></li> <li>• <b>FileName</b></li> </ul> <p>In each iteration of the loop the Angular frontend will communicate with the ASP.NET Core API through the appropriate headers by means of a <b>HTTPDELETE</b> request.</p> |

|  |  |  |
|--|--|--|
|  |  | <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• <b>Vendorname</b></li> <li>• <b>FileName</b></li> </ul> <p>The API Will search through the directories contained in the <b>Files</b> directory and delete the records containing the above mentioned details.</p> <p>The API will Return 'OK' on successful deletion</p>  |
|  |  | <p><b>Step 11:</b> The Angular frontend will call the <b>AddNewProcurementRequestFiles()</b> function which will loop through all of the files contained in the <b>Files[]</b> array and it will capture the following:</p> <ul style="list-style-type: none"> <li>• <b>File</b></li> </ul> <p>In each iteration of the loop the Angular frontend will communicate with the ASP.NET Core API through the appropriate headers by means of a HTTPPOST request.</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• <b>Vendorname</b></li> <li>• <b>File</b></li> </ul> <p>The API will receive the file and the Vendorname which will</p> |

|  |  |  |
|--|--|--|
|  |  | <p>be stored in respective variables.</p> <p>A directory will be created based on the VendorName variable.</p> <p>A path will be generated which contains the following details:</p> <ul style="list-style-type: none"> <li>• Set-Directory: Files</li> <li>• Sub-Directory: Vendorname</li> <li>• Filepath: FileName</li> </ul> <p>The file will be added to the system and the path saved will be returned as a variable</p>                           |
|  |  | <p><b>Step 12:</b> The Angular frontend will call the <b>GetProcurementRequestsbyID()</b> which will communicate with the ASP.NET Core API through the appropriate headers by means of a HTTPGET Request. The following data will be sent</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> </ul> <p>The API get the following details from the database by making use of a SQLREAD query generated by a LinqLambda function.</p> |

|  |  |   |
|--|--|---|
|  |  | <p>The following details will be retrieved from the database:</p> <p>From the <b>Procurement_Request_Quote</b> entity:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>The data from the <b>Procurement_Request_Quote</b> Entity will be retrieved based on the <i>Procurement_Request_ID</i> in the <b>Procurement_Request</b> Entity which is equal to the <i>Procurement_Request_ID</i> in the <b>Procurement_Request_Quote</b> Entity</p> <p>The <i>Procurement_Request_Quote</i> array will be returned.</p> <p>The angular frontend will receive this array and populate the <b>NewQuotes[]</b> array</p> |
|  |  | <p><b>Step 13:</b> The Angular frontend will call the <b>UpdateProcurementQuotes()</b> function which will loop through the <b>NewQuotes[]</b> Array and for each of the the <b>Procurement_Request</b></p>   |

|  |  |   |
|--|--|---|
|  |  | <p>_Quote objects contained in it will call the ASP.NET Core API through the appropriate headers by means of a HTTPPUT request.</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_Quote object</li> </ul> <p>The API will receive this object and update the Procurement_Request_Quote Entity with the changed details as follows:</p> <p>In the <b>Procurement_Request_Quote</b> entity based on the Quote_ID variable received:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>The API will return the updated object if successful</p> |
|  |  | <p><b>Step 14:</b> The system will display a successful Update notification by making use of a Angular Material Dialog which will display the following:</p>  |

|                        |   |  |  |
|------------------------|---|--|--|
|                        |   |  | <p><b>Header:</b> Update Successful</p> <p><b>Body text:</b> The procurement request for [Name] has been UPDATED successfully.</p> <p>The notification will display for 1.75 seconds.</p> <p>The system Invokes UC 3.2</p> <p><b>ViewProcurementRequest</b></p> <p>[ALT]</p> |
|                        | <p><b>[ALT] Step 7:</b> The requirements for the fields have not been met. The user is notified using the appropriate error message under the respected input which is not valid. Return to step 5</p>  |  |  |
|                        | <p><b>[ALT] Step 8:</b> The user clicks on the cancel button. Invoke <b>UC: 3.2 View Procurement Request</b></p>  |  |  |
|                        | <p><b>[ALT] Step 14:</b> A error has occurred in the Update procurement process</p> <p>The system will display a Unsuccessful Update notification by making use of a Angular Material Dialog which will display the following:</p> <p><b>Header:</b> Update Unsuccessful</p> <p><b>Body text:</b> The Update for [Name] has Failed</p> <p>The notification will display for 1.75 seconds.</p> <p>Return to Step 5</p> |  |  |
| <b>Conclusion:</b>     | The system routes the user to the <b>ViewProcurementRequest</b> Screen  |  |  |
| <b>Post-condition:</b> | An existing Procurement Request and Procurement Quote/s is updated on the system  |  |  |
| <b>Business rules</b>  | The vendor Name in the procurement request cannot be changed  |  |  |

|  |  |
|--|--|
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul> |

Figure 144 3.3 Update Procurement Request

|  |                    |                |
|--|--------------------|----------------|
| <b>System Name: Procion System</b><br><b>Sub-System:</b> |                    |                |
| Author: Jason van der Merwe                              | Date: 25 July 2023 | Version: 1.0.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | Delete Procurement Request   | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                   | 3.4  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                      | High   | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | User   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The user requests to delete a procurement request. The system will load the “DeleteProcurementRequest” screen which contains a angular material dialog. The material dialog will ask for confirmation from the user to ask if they are sure that they wish to delete the specified Procurement Request. The user will click on the <b>Yes</b> button which will cause the system to delete all the quotes related to a procurement request and subsequently delete the procurement request. The user will be redirected to the <b>ViewProcurementRequest</b> screen</p> |  |

| <b>Pre-condition:</b>                | The user has to be logged in to the system.<br>There must be at least one procurement request stored on the system<br>The user has loaded the “ <b>ViewProcurementRequest</b> ” screen<br>The user has clicked on the <b>Delete</b> button  |                        |  |
|--------------------------------------|---|------------------------|--|
| <b>Trigger:</b>                      | The user requests to delete a consumable on the system  |                        |  |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>   | <b>System Response</b> |  |
| <b>Manual Action</b>                 | <b>Automated Action</b>   |                        |  |
| <b>None</b>                          | <p><b>Step 2:</b> The system will use the Angular frontend to navigate to the “<i>DeleteProcurementRequest</i>” page while sending the ID of the Procurement request to be deleted to that respective page</p> <p>The Angular frontend will call the <b>GetProcurementRequestsbyID()</b> method and the <b>GetProcurementRequestQuotesbyID()</b> which will communicate with the ASP.NET Core API through the appropriate headers by means of a HTTPGET Request. The following data will be sent</p> <ul style="list-style-type: none"> <li>• ID</li> </ul> <p>The API get the following details from the database by making use of a SQLREAD query</p> |                        |  |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>generated by a LinqLambda function.</p> <p>The following details will be retrieved from the database:</p> <p>From the <b>Procurement_Request Entity</b>:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Procurement_Request</li> <li>• Description</li> </ul> <p>From the <b>Procurement_Request_Quote entity</b>:</p> <ul style="list-style-type: none"> <li>• Quote_ID</li> <li>• Procurement_Request_ID</li> <li>• Path</li> <li>• Upload_Date</li> <li>• Preferred_Quote</li> </ul> <p>The data from the <b>Procurement_Request Entity</b> will be retrieved based on where the <b>Procurement_Request_ID</b> in the <b>Procurement_Request Entity</b> is equal to the ID variable sent through.</p> |
|--|--|--|--|

|  |  |   |
|--|--|---|
|  |  | <p>The data from the <b>Procurement_Request_Quote</b> Entity will be retrieved based on the <b>Procurement_Request_ID</b> in the <b>Procurement_Request</b> Entity which is equal to the <b>Procurement_Request_ID</b> in the <b>Procurement_Request_Quote</b> Entity.</p> <p>The <b>Procurement_Request</b> and <b>Procurement_Request_Quote</b> array will be returned.</p> <p>The Angular frontend will capture the response as follows:</p> <ul style="list-style-type: none"> <li>• In the <b>Procurement_Request</b> Object: The <b>Procurement_Request_Received</b></li> <li>• In the <b>Procurement_Quotes[]</b> array: The <b>Procurement_Request_Quote</b> received.</li> </ul> |
|--|--|---|

|  |  |      |   |
|--|--|------|---|
|  |  | None | <b>Step 3:</b> The system will generate the |
|--|--|------|---|

|  |   |   |
|--|---|---|
|  |   | <p>“DeleteProcurementRequest” screen with the angular frontend that contains an angular material dialog with the following details:</p> <ul style="list-style-type: none"> <li>• <b>Header:</b> Delete Procurement Request</li> <li>• <b>Label</b> with text: “You are about to delete the request for: <b>[Request Name]</b>. Are you sure you want to continue?”</li> <li>• <b>Buttons:</b> Yes, Cancel</li> </ul> <p>The <b>[Request Name]</b> placeholder will be populated with the following received from the ASP.Net API in the <b>Procurement Request</b> Entity:</p> <ul style="list-style-type: none"> <li>• Name</li> </ul> |
|  | <p><b>Step 4:</b> The user clicks on the “Yes” button</p> <p><b>[ALT]</b></p> |   |
|  |   | <p><b>Step 6:</b> The system calls the <b>OnConfirm()</b> function which includes the <b>DeleteProcurementRequest()</b> function inside of it. The angular frontend will pass the following:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> </ul> <p>The Angular frontend will communicate with</p>   |

|  |  |  |
|--|--|--|
|  |  | <p>the API through the appropriate header by means of an HTTPDELETE request. The API will then remove the record from the system by means of Entity Framework which contains the following details:</p> <p><b>Procurement_Request</b> Entity where the Procurement_Request_ID attribute is equal to the Procurement_Request_ID variable sent:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Vendor_ID</li> <li>• Requisition_Status_ID</li> <li>• User_ID</li> <li>• Name</li> <li>• Procurement_Request</li> <li>• Description</li> </ul> <p>The API will return the deleted Procurement request to the Angular Frontend</p> |
|  |  | <p><b>Step 7:</b> The Angular frontend will call the <b>DeleteProcurementRequestFiles()</b> function which will loop through all of the files contained in the <b>Procurement_Quotes[</b></p>  |

|  |  |   |
|--|--|---|
|  |  | <p>] array and it will capture the following:</p> <ul style="list-style-type: none"> <li>• <b>Vendorname</b></li> <li>• <b>FileName</b></li> </ul> <p>In each iteration of the loop the Angular frontend will communicate with the ASP.NET Core API through the appropriate headers by means of a HTTPDELETE request.</p> <p>The following data will be sent:</p> <ul style="list-style-type: none"> <li>• <b>Vendorname</b></li> <li>• <b>FileName</b></li> </ul> <p>The API Will search through the directories contained in the <u>Files</u> directory and delete the records containing the above mentioned details.</p> <p>The API will Return 'OK' on successful deletion</p> |
|  |  | <p><b>Step 8:</b> The angular frontend will then capture the response received from the API and generate the following notification:</p> <p>Header: "Delete Successfull"</p> <p>Label with text: "The request [Request] has been deleted successfully!"</p>   |

|  |   |  |  |
|--|---|--|--|
|  |   |  | The Category placeholder will be populated with the Procurement request Name that was captured on the Angular frontend From the ASP.Net Core API |
|  |   |  | <b>Step 8:</b> The system invokes UC 3.2 View Procurement Request  |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 4:</b> The user clicks on the cancel button. Invoke <b>UC: 3.2 View Procurement Request</b> |  |  |
| <b>Conclusion:</b>                                   | The system routes the user to the <b>ViewProcurementRequest</b> Screen                                    |  |  |
| <b>Post-condition:</b>                               | An existing Procurement Request along with all associated quotes are removed from the system              |  |  |
| <b>Business rules</b>                                | None  |  |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |  |  |

Figure 145 3.4 Delete Procurement Request

## USE CASE 3.5-3.7,6.5 &amp; 6.10-6.14

|  |                  |              |
|--|------------------|--------------|
| System Name: Procision System<br>Sub-System: 3 Procurement |                  |              |
| Author: Werner Schutte                                     | Date: 2023/07/23 | Version: 1.0 |

| Use case name:                 | View Pending Procurement Request   |                 | USE CASE TYPE                                      |
|--------------------------------|--|-----------------|--|
| Use case id:                   | 3.5  |                 | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | high   |                 | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo   |                 | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | User (PBA)   |                 |  |
| Primary system actor           | None   |                 |  |
| Other participating actors:    | None   |                 |  |
| Other interested stakeholders: | None   |                 |  |
| Description:                   | The use case describes the event where the user will want to approve or reject a procurement that has been created. The user will click on the <b>View</b> button on the <u>View Pending Procurement Request</u> Screen. The system will load the <u>View Procurement Request Approval</u> Screen. The user will then provide the Approval or rejection details of the procurement. Finally the use case conclude when a notification is displayed based on provided details and user is routed back to previous screen. |                 |  |
| Pre-condition:                 | The requestor has to be logged in to the system.   |                 |  |
| Trigger:                       | The user requests to view all of the vendors saved on the system.  |                 |  |
| Typical course of events:      | Actor Action   | System Response |  |
|                                |  | Manual Action   | Automated Action                                   |

|  |  |  |
|--|--|--|
|  |  | <p><b>Step 1:</b> The system makes use of the angular frontend and routes the requestor to the “View Pending Procurement Request Approval” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “View Pending Procurement Request”</li> <li>• Search Input: Label Containing “Search by Name”</li> </ul> <p>Angular Mat table containing the following Column headers:</p> <ul style="list-style-type: none"> <li>• Procurement Name</li> <li>• Procurement Description</li> <li>• User</li> <li>• Vendor name</li> <li>• Button: View</li> </ul> |
|  |  | <p><b>Step 2:</b> The system will then populate table by calling the function <b>GetProcurementRequests()</b>.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a Procurement_Request object following Details in the</p>   |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <b>Procurement_Request</b> entity with a SQL read Query.   |
|  | <b>Step 3:</b> The user will click the “View” Button |  | <p><b>Step 4:</b> The system will display the “View Procurement Request Approval” screen with the following details:</p> <ul style="list-style-type: none"> <li>• Header: Label containing text: “Procurement Request For ‘Vendor name’”</li> <li>• Mat icon “arrow_back” button</li> <li>• Label: “Please Approve or Reject the Procurement Request”</li> <li>• “Company Name” label containing a corresponding input (set to read only)</li> <li>• “Company Email” label containing a corresponding input (set to read only)</li> <li>• “Company Name” label containing a corresponding input (set to read only)</li> <li>• “Description” label containing a corresponding input (set to read only)</li> </ul> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• “Preferred Quote” label containing a corresponding input (set to read only)</li> <li>• “Description” label containing a corresponding input (set to read only)</li> <li>• “Comparative Quote 1” label containing a corresponding input (set to read only) display if vendor is of type other</li> <li>• “Comparative Quote 2” label containing a corresponding input (set to read only) display if vendor is of type other</li> <li>• Buttons: Accept Request and Reject Request</li> </ul> |
|  |  |  | <p><b>Step 5:</b> The system will prepopulate the controls by calling on the function GetProcurementRequestByID().</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and receive Procurement_Request_ID.</p>  |

|  |  |  |
|--|--|--|
|  |  | <p>The API will then retrieve a Procurement_Request object following Details in the <b>Procurement_Request</b> entity with a SQL read Query.</p> <p>The system will then call on the function GetProcurementRequestQuoteByID ().</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and receive Procurement_Request_ID.</p> <p>The API will then retrieve Procurement_Request_Quotes following Details in the <b>Procurement_Request_Quote</b> entity with a SQL read Query.</p> <p>The system will then prompt the user to provide accept or rejection details.</p> |
|--|--|--|

|                           |  |  |   |
|---------------------------|--|--|---|
|                           | <p><b>Step 6:</b> The user will then provide accept or rejection details.</p> <p>[ALT]</p>   |  | <p><b>Step 7:</b> The system will call the function UpdateProcurementRequestStatus().</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUT request and receive Procurement_Request object and StatusID.</p> <p>The API will then update the following Details in the <b>Procurement_Request</b> entity with a SQL Update Query.</p> <ul style="list-style-type: none"> <li>• Procurement_Status_ID</li> </ul> |
|                           |  |  | <p><b>Step 8:</b> The system will display a Approve success notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “APPROVAL SUCCESSFUL” header</li> <li>• “The Procurement Request ‘Number’ has been successfully approved!” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p> <p>[ALT]</p>   |
| <b>Alternate courses:</b> | <p>[ALT] <b>Step 8:</b> The user does not wish to continue and clicks the mat icon “arrow_back” button. <b>Use Case Terminates</b></p> |  |   |

|  |   |
|--|---|
|  | <p><b>[ALT] Step 10:</b> The system will display a Reject success notification with the following elements:</p> <ul style="list-style-type: none"> <li>• “Rejection SUCCESSFUL” header</li> <li>• “The Procurement Request ‘Number’ has been successfully Rejected!” card body</li> </ul> <p>The notification will disappear after 1750 seconds</p> |
| <b>Conclusion:</b>                                   | The system updates the procurement request status based on provided details from user.  |
| <b>Post-condition:</b>                               | The correct notification is displayed to the user.  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

Table 202: 3.5 View Pending Procurement Request

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 3 Procurement |                  |              |
| Author: Werner Schutte                                   | Date: 2023/07/23 | Version: 1.0 |

|                               |                           |  |
|-------------------------------|---------------------------|--|
| <b>Use case name:</b>         | Place procurement request | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>           | 3.6                       | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>              | high                      | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                | Moyo                      | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b> | User (PBA)                |  |
| <b>Primary system actor</b>   | None                      |  |

|                                       |   |  |  |
|---------------------------------------|---|--|--|
| <b>Other participating actors:</b>    | None  |  |  |
| <b>Other interested stakeholders:</b> | None  |  |  |
| <b>Description:</b>                   | <p>The use case describes the event where the user wants to place a procurement for an approve procurement request. The user will click on the <u>View</u> button on the <u>Place Procurement Request</u> Screen. The system will load the <u>Place Procurement Request Create Details</u> Screen. The user will then provide the following details:</p> <ul style="list-style-type: none"> <li>• Buyer Name</li> <li>• Buyer Email</li> <li>• Item Type</li> <li>• Select Consumable Item (if Item Type = Consumables)</li> <li>• Quantity (if Item Type = Consumables)</li> <li>• Asset Name (if Item Type = Assets)</li> <li>• Description of Asset (if Item Type = Assets)</li> <li>• Account Code</li> <li>• Payment Type</li> <li>• Deposit (True or False)</li> <li>• Deposit Amount (Deposit = true)</li> <li>• Deposit Due Date (Deposit = true)</li> <li>• Full Payment Made (True or False)</li> <li>• Paid On Date (Full Payment Made = true)</li> <li>• Upload Receipt (Full Payment Made = true)</li> <li>• Proof of payment (True or False)</li> <li>• Proof of payment document (Proof of payment = true)</li> <li>• Total Amount</li> <li>• Total Amount Due Date</li> <li>• Comments</li> </ul> <p>Finally, the use case conclude when a notification is display based on provided details and user a routed back to previous screen.</p> |  |  |
| <b>Pre-condition:</b>                 | There must be procurement request that was approved   |  |  |
| <b>Trigger:</b>                       | The user wants to place a procurement for an approve procurement request.   |  |  |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b><br><b>Manual Action</b><br>   | <b>System Response</b><br><b>Automated Action</b><br><b>Step 1:</b> The system makes use of the angular frontend and routes the requestor to |  |

|  |  |  |
|--|--|--|
|  |  | <p>the “Place Procurement Request” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Place Procurement Request”</li> </ul> <p>Mat table containing the following details:</p> <ul style="list-style-type: none"> <li>• Procurement Name</li> <li>• Procurement Description</li> <li>• User</li> <li>• Vendor Name</li> <li>• Button: View</li> </ul>  |
|  |  | <p><b>Step 2:</b> The system will then populate table by calling the function <b>GetProcurementRequests ()</b>.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a <b>Procurement_Request</b> object following Details in the <b>Procurement_Request</b> entity with a SQL read Query.</p> <p>This object will then be filter to only those that have been approved</p> |

|  |  |  |   |
|--|--|--|---|
|  | <p><b>Step 3:</b> The user will click the “View” button.</p> |  | <p><b>Step 4:</b> The system makes use of the angular frontend and routes the requestor to the “Place Procurement Request Create Details” page with the procurement request ID as a parameter and the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Place Procurement Requests Details”</li> <li>• “Please fill in the following details:” label</li> <li>• “Buyer Name:” label with its corresponding input.</li> <li>• “Buyer Name:” label with its corresponding input.</li> <li>• “Buyer Email:” label with its corresponding input.</li> <li>• “Item type:” label with its corresponding input. (Default set to Consumable)</li> </ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"> <li>• “Select Consumable Item:” label with its corresponding mat select. (Displayed if Item type = ‘Consumable’)</li> </ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>“Quantity:” label with its corresponding input of type number.<br/>(Displayed if Item type = ‘Consumable’)</li> </ul> <p>Mat Expansion panel end here</p> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"> <li>“Asset Name:” label with its corresponding input. (Displayed if Item type = ‘Assets’)</li> <li>“Description of Asset:” label with its corresponding input. (Displayed if Item type = ‘Assets’)</li> </ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"> <li>“Account Code:” label with its corresponding mat select.</li> <li>“Payment Type” label with its corresponding mat select (value either Credit Payments, Cash Payments, EFT)</li> <li>“Has Deposit?” label with its corresponding checkbox</li> </ul> <p>Mat Expansion Panel:</p> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• “Deposit Amount:” label with its corresponding input of type number.</li><li>• “Deposit Due Date:” label with its corresponding date picker input input.</li></ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"><li>• “Full Payment Made?” label with its corresponding checkbox</li></ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"><li>• “Paid On Date:” label with its corresponding date picker input.</li><li>• “Upload Receipt:” label with its corresponding input of type file.</li></ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"><li>• “Proof of payment?” label with its corresponding checkbox</li></ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"><li>• “Upload Proof of payment document:” label with its</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>corresponding input of type file.</p> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"> <li>• “Total Amount:” label with its corresponding input of type number.</li> <li>• “Total Amount Due Date:” label with its corresponding date picker input.</li> <li>• “Comments:” label with its corresponding textarea.</li> <li>• Button:Place, Cancel</li> </ul>   |
|  |  |  | <p><b>Step 5:</b> The system then calls on the following functions:</p> <ul style="list-style-type: none"> <li>• GetConsumables()</li> <li>• GetBudgetLines()</li> <li>• GetEmployeeByUsername()</li> <li>• GetProcurementRequestByID()</li> </ul> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a 4 different objects from the following entities:</p> |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• <b>Procurement_Request</b></li> <li>• <b>Budget_Line</b></li> <li>• <b>Employee</b></li> <li>• <b>Consumables</b></li> </ul> <p>with a SQL read Query.</p> <p>To populate the required inputs.</p>  |
|  |  |  | <p><b>Step 6:</b> The system will prompt the user to provide the required details.</p>   |
|  | <p><b>Step 7:</b> The user provides the following details:</p> <ul style="list-style-type: none"> <li>• Buyer Name</li> <li>• Buyer Email</li> <li>• Item Type</li> <li>• Select Consumable Item (if Item Type = Consumables)</li> <li>• Quantity (if Item Type = Consumables)</li> <li>• Asset Name (if Item Type = Assets)</li> <li>• Description of Asset (if Item Type = Assets)</li> <li>• Account Code</li> <li>• Payment Type</li> <li>• Deposit (True or False)</li> <li>• Deposit Amount (Deposit = true)</li> <li>• Deposit Due Date (Deposit = true)</li> </ul> |  | <p><b>Step 8:</b> The system will capture the provided details and validate it as follows:</p> <ul style="list-style-type: none"> <li>• Buyer Name: Must not contain numbers and a max length of 32 characters and is required.</li> <li>• Buyer Email: Must not exceed length of 32 characters and must be an email and is required.</li> <li>• Item Type: not required as default is set to consumable.</li> <li>• Select Consumable Item: Must be provided and is disabled if Item Type = "Assets".</li> <li>• Quantity: Must be a number and is required and is disabled if</li> </ul> |

|  |   |   |
|--|---|---|
|  | <ul style="list-style-type: none"> <li>• Full Payment Made (True or False)</li> <li>• Paid On Date (Full Payment Made = true)</li> <li>• Upload Receipt (Full Payment Made = true)</li> <li>• Proof of payment (True or False)</li> <li>• Proof of payment document (Proof of payment = true)</li> <li>• Total Amount</li> <li>• Total Amount Due Date</li> <li>• Comments</li> </ul> | <ul style="list-style-type: none"> <li>• Item Type = “Assets”.</li> <li>• Asset Name: Must not contain numbers and a max length of 32 characters and is required and is disabled if Item Type = “Consumable”.</li> <li>• Description of Asset: Has max length of 50 characters and is required and is required and is disabled if Item Type = “Consumable”.</li> <li>• Account Code: Is required.</li> <li>• Payment Type: Is required</li> <li>• Has Deposit?: Optional</li> <li>• Deposit Amount: Required (if “Has Deposit = true) and must not contain letters.</li> <li>• Deposit Due Date: must not be null, Must not be earlier than current date then the current date (if “Has Deposit = true)</li> <li>• Full Payment Made?: Optional</li> <li>• Paid On Date: must not be null, (if “Full Payment Made? = true)</li> <li>• Upload Receipt: must not be null</li> </ul> |
|--|---|---|

|  |  |   |
|--|--|---|
|  |  | <ul style="list-style-type: none"> <li>(if “Full Payment Made? = true)</li> <li>• Proof of payment?: Optional</li> <li>• Upload Proof of payment document: Required (If Proof of payment? = true)</li> <li>• Total Amount: must only contain numbers.</li> <li>• Total Amount Due Date: must not be null</li> <li>• Comments: Optional and must not be more than 50 characters.</li> </ul> <p>[ALT]</p> |
|  |  | <p><b>Step 9:</b> The system will then call on the function AddProcurementDetails()</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUSH request and receive Procurement_Details object.</p> <p>The API will add the following details into the Procurement_Details entity using SQL Insert Query:</p>         |

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"><li>• Employee_ID (based on the user placing the procurement request)</li><li>• Procurement_Status_ID (Either set to 1 or 3 based on whether procurement gets flagged)</li><li>• Sign_Off_Status (set as default to 1)</li><li>• Procurement_Payment_Status_ID (based on whether payment has been partially, fully or still needs payment)</li><li>• Account_Code (based on account code selected by user)</li><li>• Payment_Method_ID (based on payment method selected by user)</li><li>• Item_Type (either assets or consumables)</li><li>• Buyer_Name</li><li>• Buyer_Email</li><li>• Deposit_Required (Boolean value)</li><li>• Full_Payment_Due_Date (Boolean value)</li><li>• Total_Amount</li><li>• Payment_Made</li><li>• Comment</li><li>• Proof_Of_Payment_Required (Boolean value)</li></ul> |
|--|--|--|--|

|  |  |   |
|--|--|---|
|  |  | <p>Afterwards It will return the newly added procurement_details object and that will be used to call the following function AddDeposit() if “Has Deposit?” = true</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUSH request and receive Deposit object.</p> <p>The API will add the following details into the Deposit entity using SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Deposit_Due_Date</li> <li>• Deposit_Amount</li> <li>• Amount_Outstanding (based on total amount subtracted by deposit amount)</li> </ul> <p>Same will occur for Proof of payment and Payment made calling these functions:</p> <ul style="list-style-type: none"> <li>• AddProofOfPayment()</li> <li>• AddPaymentMade()</li> </ul> <p>In both instance the uploadProcureFile() function is called where the angular frontend will then call the function which will communicate with the appropriate header in the API by means of an</p> |
|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>HTTP PUSH request and receive FormData. That will go on to add the file in API side and return the file path to that file.</p> <p>After returning the file path for both previous functions the data is stored in database in the following tables with attributes such as:</p> <p><b>PAYMENT_MADE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Paid_On_Date</li> <li>• Receipt_Upload</li> </ul> <p><b>PROOF_OF_PAYMENT</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Request_ID</li> <li>• Proof_Of_Payment_Doc</li> </ul> <p>Using SQL Insert query.</p> <p>If Item type = “Consumable”<br/>The Next the function AddProcurementConsumable() is called to add the link between procurement_details and consumable with the following data:</p> <p><b>PROCUREMENT_CONSUMABLE</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID (PK,FK)</li> <li>• Consumable_ID (PK,FK)</li> <li>• Quantity</li> </ul> |
|--|--|---|

|  |  |  |
|--|--|--|
|  |  | <p>Afterwards a GetVendorConsumable() function is called to get the right consumable ID to be used in the next function AddVendorConsumable() which adds a link between consumable and vendor entities:</p> <p><b>VENDOR_CONSUMABLE</b> table:</p> <ul style="list-style-type: none"> <li>• Consumable_ID</li> <li>• Vendor_ID</li> <li>• Vendor_Consumable_ID</li> </ul> <p>Otherwise if Item Type = “Assets”</p> <p>The next function AddAsset() is called.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUSH request and receive Asset object.</p> <p>The API will add the following details into the ASSET entity using SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Description</li> </ul> <p>The Next the function AddProcurementAsset() is called to add the link between procurement_details and Assets with the following data:</p> |
|--|--|--|

|  |  |  |
|--|--|--|
|  |  | <p><b>PROCUREMENT_ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Procurement_Details_ID</li> <li>• Asset_ID</li> </ul> <p>Afterwards a GetVendorConsumable() function is called to get the right consumable ID to be used in the next function AddVendorConsumable() which adds a link between assets and vendor entities:</p> <p><b>VENDOR_ASSET</b> table:</p> <ul style="list-style-type: none"> <li>• Asset_ID</li> <li>• Vendor_ID</li> <li>• Vendor_Asset_ID</li> </ul> |
|  |  | <p><b>Step 10:</b> The Angular frontend will make use of an angular material dialog to display the following information to the user:</p> <ul style="list-style-type: none"> <li>• Header: “Procurement Placed!”</li> <li>• Label with text: “The Procurement request has been Placed successfully!”</li> </ul>  |

|  |  |  |
|--|--|--|
|  |  | The dialog will pop up for 1.75 seconds. |
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 2:</b> The user does not want to proceed with use case. <b>Use Case Terminates.</b></p> <p><b>[ALT] Step 8:</b> Not all fields are filled in or a field/fields do not correspond to the validation criteria. <b>Return to step 7.</b></p>   |  |
|  | <p>The system has added all the details in following tables where needed:</p> <ul style="list-style-type: none"> <li>• <b>PROCUREMENT_DETAILS</b></li> <li>• <b>DEPOSIT</b></li> <li>• <b>PAYMENT_MADE</b></li> <li>• <b>PROOF_OF_PAYMENT</b></li> <li>• <b>PROCUREMENT_CONSUMABLE</b></li> <li>• <b>VENDOR_CONSUMABLE</b></li> <li>• <b>PROCUREMENT_ASSET</b></li> <li>• <b>ASSET</b></li> <li>• <b>VENDOR_ASSET</b></li> </ul> |  |
| <b>Post-condition:</b>                               | Success notification is displayed and user is routed back to previous page.  |  |
| <b>Business rules</b>                                | None   |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |  |

Table 203: 3.6 Place Procurement Request

|  |                  |              |
|--|------------------|--------------|
| System Name: Procion System<br>Sub-System: 3 Procurement |                  |              |
| Author: Werner Schutte                                   | Date: 2023/07/23 | Version: 1.0 |

|                                       |  |  |
|---------------------------------------|--|--|
| <b>Use case name:</b>                 | View Flagged Procurement Request   |  |
| <b>Use case id:</b>                   | 3.7  | Business Requirements: <input type="checkbox"/>  |
| <b>Priority:</b>                      | high   | System Analysis: <input type="checkbox"/>  |
| <b>Source:</b>                        | Moyo   | System Design: <input checked="" type="checkbox"/>   |
| <b>Primary business actor</b>         | User (PBA)   |  |
| <b>Primary system actor</b>           | None   |  |
| <b>Other participating actors:</b>    | None   |  |
| <b>Other interested stakeholders:</b> | None   |  |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to approve or reject a procurement that has been placed but is flagged due to user that created it mandate limit being to low compare to request total amount. The user will click on the <u>View</u> button on the <u>View Flagged Procurement Request</u> Screen. The system will load the <u>View Flagged Procurement Details</u> Screen. The user will then provide the Approval or rejection details of the procurement details. Finally the use case conclude when a notification is display based on provided details and user a routed back to previous screen.</p> |  |
| <b>Pre-condition:</b>                 | The requestor has to be logged in to the system.   |  |
| <b>Trigger:</b>                       | The requestor requests to delete a Vendor and its details on the system  |  |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>  | <b>System Response</b>   |
|                                       |  | <b>Manual Action</b>   |
|                                       |  | <b>Step 1:</b> The system makes use of the angular frontend to display the “View Flagged Procurement |

|  |  |  |
|--|--|--|
|  |  | <p>Request" page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: "View Flagged Placed Procurement Request"</li> <li>• Search input with following label "Search by Name"</li> </ul> <p>Mat table containing the following details:</p> <ul style="list-style-type: none"> <li>• Vendor Name</li> <li>• Created By</li> <li>• Mandate Limit</li> <li>• Request Total</li> <li>• Payment Due</li> <li>• Button: View</li> </ul>   |
|  |  | <p><b>Step 2:</b> The system will then populate the table by calling the function <b>GetProcurementRequestDetails ()</b>.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a <b>Procurement_Details</b> object following Details in the <b>Procurement_Details</b> entity with a SQL read Query.</p> <p>This object will then be filter to only those that have been Flagged.</p> |

|  |  |  |  |
|--|--|--|--|
|  | <p><b>Step 3:</b> The user clicks on the “View” button</p> |  | <p><b>Step 4:</b> The system makes use of the angular frontend and routes the user to the “View Flagged Procurement Details” page with the procurement request ID as a parameter and the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Place Procurement Requests Details”</li> <li>• Mat Icon “arrow_back” button</li> <li>• “Please fill in the following details:” label</li> <li>• “Buyer Name:” label with its corresponding input.</li> <li>• “Buyer Name:” label with its corresponding input.</li> <li>• “Buyer Email:” label with its corresponding input.</li> <li>• “Item type:” label with its corresponding input. (Default set to Consumable)</li> </ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"> <li>• “Select Consumable Item:” label with its corresponding mat select. (Displayed if</li> </ul> |
|--|--|--|--|

|  |  |  |
|--|--|--|
|  |  | <p>Item type = ‘Consumable’)</p> <ul style="list-style-type: none"> <li>“Quantity:” label with its corresponding input of type number.<br/>(Displayed if Item type = ‘Consumable’)</li> </ul> <p>Mat Expansion panel end here</p> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"> <li>“Asset Name:” label with its corresponding input. (Displayed if Item type = ‘Assets’)</li> <li>“Description of Asset:” label with its corresponding input. (Displayed if Item type = ‘Assets’)</li> </ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"> <li>“Account Code:” label with its corresponding mat select.</li> <li>“Payment Type” label with its corresponding mat select (value either Credit Payments, Cash Payments, EFT)</li> <li>“Has Deposit?” label with its corresponding checkbox</li> </ul> |
|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"><li>• “Deposit Amount:” label with its corresponding input of type number.</li><li>• “Deposit Due Date:” label with its corresponding date picker input input.</li></ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"><li>• “Full Payment Made?” label with its corresponding checkbox</li></ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"><li>• “Paid On Date.” label with its corresponding date picker input.</li><li>• “Upload Receipt:” label with its corresponding input of type file.</li></ul> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"><li>• “Proof of payment?” label with its corresponding checkbox</li></ul> <p>Mat Expansion Panel:</p> <ul style="list-style-type: none"><li>• “Upload Proof of payment document:” label with its</li></ul> |
|--|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>corresponding input of type file.</p> <p>Mat Expansion panel end here</p> <ul style="list-style-type: none"> <li>• “Total Amount:” label with its corresponding input of type number.</li> <li>• “Total Amount Due Date:” label with its corresponding date picker input.</li> <li>• “Comments:” label with its corresponding textarea.</li> <li>• Button:Accept, Reject</li> </ul>  |
|  |  | <p><b>Step 6:</b> The system will then populate the controls by calling the function <b>GetProcurementDetailsByID ()</b>.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a <b>Procurement_Details</b> object following Details in the <b>Procurement_Details</b> entity with a SQL read Query.</p> |

|  |  |   |
|--|--|---|
|  |  | <p>Based on the returned Procurement_Details the following functions are also called:</p> <ul style="list-style-type: none"> <li>• GetDepositByID()</li> <li>• GetProofOfPaymentByID()</li> <li>• GetFullPaymentMadeByID()</li> </ul> <p>The angular frontend will then call the functions which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve the following objects:</p> <ul style="list-style-type: none"> <li>• Deposit</li> <li>• PaymentMade</li> <li>• ProofOfPayment</li> </ul> <p>From their related entities:</p> <ul style="list-style-type: none"> <li>• Deposit</li> <li>• Payment_Made</li> <li>• Proof_Of_Payment</li> </ul> <p>with a SQL read Query.</p> <p>Then based on Item Type the next function that can be called for consumable:</p> <ul style="list-style-type: none"> <li>• GetProcurementConsumable()</li> </ul> <p>The angular frontend will then call the function which will communicate with the</p> |
|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a <b>Procurement_Consumable</b> object following Details in the <b>Procurement_Consumable</b> entity with a SQL read Query.</p> <p>If the Item Type is for asset the following functions are called:</p> <ul style="list-style-type: none"> <li>• GetProcurementAsset()</li> <li>• GetAssetByID()</li> </ul> <p>The angular frontend will then call the functions which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve the following objects:</p> <ul style="list-style-type: none"> <li>• Procurement_Asset</li> <li>• Asset</li> </ul> <p>From their related entities:</p> <ul style="list-style-type: none"> <li>• Procurement_Asset</li> <li>• Asset</li> </ul> <p>with a SQL read Query.</p> |
|--|--|---|

|  |  |   |   |
|--|--|---|---|
|  |  |   | Finally prompting the user to provide accept or reject details                                    |
|  | <p><b>Step 7:</b> The user clicks on the “Approve” Button</p> <p>[ALT]</p> | <p><b>Step 8:</b> The UpdateProcurementDetailsStatus() function is called</p> <p>The angular frontend will then call the functions which will communicate with the appropriate header in the API by means of an HTTPPUT request and receive the Procurement_status_ID and Procurement_Details object</p> <p>The angular frontend will then capture the response received from the API and generate the following notification:</p> <p>Header: “APPROVE SUCCESSFUL”</p> <p>Label with text: “The Procurement details has been Approved successfully!”</p> <p>[ALT]</p> |   |
|  |  |   | <p><b>Step 9:</b> The system routes the user to the “View Flagged Procurement Request” Screen</p> |

|  |   |
|--|---|
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 7a:</b> The user clicks on the mat icon “arrow_back” Button. The user will be redirected to the “View Flagged Procurement Request” screen. <b>Use Case Terminates</b></p> <p><b>[ALT] Step 7b:</b> The user clicks on the “Reject” Button. <b>Proceed to step 8.</b></p> |
|  | <p><b>Step 8:</b> The angular frontend will then capture the response received from the API and generate the following notification:</p> <p>Header: “REJECTION SUCCESSFUL”</p> <p>Label with text: “The Procurement details has been rejected successfully!”</p>                          |
| <b>Conclusion:</b>                                   | The system routes the user to the <i>VendorView</i> Screen  |
| <b>Post-condition:</b>                               | An existing vendor and its details are removed from the system and the database is updated  |
| <b>Business rules</b>                                | none  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>  |

Table 204: View Flagged Procurement Request

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                       |                         |   |
|-----------------------|-------------------------|---|
| <b>Use case name:</b> | Approve onboard request | <b>USE CASE TYPE</b>                            |
| <b>Use case id:</b>   | 6.5                     | Business Requirements: <input type="checkbox"/> |

|                                       |  |                  |                                     |
|---------------------------------------|--|------------------|-------------------------------------|
| <b>Priority:</b>                      | High   | System Analysis: | <input type="checkbox"/>            |
| <b>Source:</b>                        | Moyo   | System Design:   | <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>         | GRC Officer (PBA)  |                  |                                     |
| <b>Primary system actor</b>           | None   |                  |                                     |
| <b>Other participating actors:</b>    | None   |                  |                                     |
| <b>Other interested stakeholders:</b> | None   |                  |                                     |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to approve or reject a vendor from an onboard request. The user will click on the <u>View</u> button on the <u>Vendor Unofficial Vendorlist</u> Screen. The system will load the <u>Approve Vendor</u> Screen. The user will then provide the selection details of the available vendors that they wish to approve for the onboard request, the system will then request the user to provide confirmation details. The user does so by then clicking on the “Continue” button. The system responds by loading a <u>Vendor Approve Add Details</u> Screen and prompts the user to add the following details:</p> <ul style="list-style-type: none"> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> <li>• POPIA_Declaration</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Direcor_Details_Provided</li> <li>• Company_Resolution_Agreement_Provided</li> <li>• Subcontractor_Name_Provided</li> <li>• Company_Details_Provided</li> <li>• Individual_Details_Provided</li> <li>• General_Liability_Insurance_Present</li> <li>• Cyber_Insurance_Present</li> <li>• Proffesional_Indemnity_Insurance_Present</li> </ul> |                  |                                     |

- Other\_Insurance\_Required
- Licenses\_Required
- Accreditation\_Required
- Professional\_Membership\_Required
- Business\_Continuity\_Present
- Disaster\_Recovery\_Plan\_Present
- POPI\_Present
- Data\_Security\_Breaches\_Present
- Site\_Visits\_Present
- Information\_Security\_Policy\_Present
- Privacy\_Policy\_Present\_Present
- Data\_Retention\_Destruction\_Policy\_Present
- Anti\_Bribery\_Corruption\_Policy\_Present
- Ethics\_Policy\_Present
- Conflict\_Of\_Interest\_Policy\_Present
- Customer\_Complaints\_Policy\_Present
- Business\_References\_Present
- Contracted\_Partner\_Type\_ID (FK)
- Personal\_Data\_Purpose
- DataProcessing\_JointController\_Agreement
- Confidentiality\_Importance\_Highlighted
- Contract\_Audits\_Provisions\_Provided
- Activity\_Liability\_Present
- Third\_Party\_Data\_Processing\_Provisioned
- Contract\_End\_Data\_Management\_Provided
- Personal\_Data\_Processing\_Details\_Present
- Processing\_Activities\_Certification\_Held
- General\_Liability\_Insurance\_Present\_Document
- Cyber\_Insurance\_Present\_Document
- Professional\_Indemnity\_Insurance\_Present\_Document
- Other\_Insurance\_Required\_Document

The user will then click on the “Create” button and the use case conclude after saving the details into the following tables:

- Vendor\_BEE
- Vendor\_POPIA
- DUE\_DILIGENCE
- POPI
- Insurnace

And updating the following status in the **Vendor** table and **Onboard\_Request** table.

Finally Displaying a success notification to the user.

|                                      |   |   |                         |
|--------------------------------------|---|---|-------------------------|
| <b>Pre-condition:</b>                | The requestor must be logged into the system. |   |                         |
| <b>Typical course<br/>Of events:</b> | <b>Actor Action</b>                           | <b>System Response</b>  |                         |
|                                      |   | <b>Manual Action</b>  | <b>Automated Action</b> |
|                                      |   | <p><b>Step 1:</b> The system makes use of the angular frontend and routes the requestor to the “Vendor Approved” page with the following items:</p> <ul style="list-style-type: none"> <li>• Header: “Onboard Request # ‘Onboard Request Number’ ”</li> <li>• Label: “Please select one of the ‘Total vendors’ options”</li> <li>• Tab group containing at least 3 tabs where each having the following contents:</li> <li>• Label with a corresponding text box called company name (set to read only)</li> <li>• Label with a corresponding text box called company email (set to read only)</li> <li>• Label with a corresponding file name with a download or open in new tab function when clicked on</li> </ul> |                         |

|  |   |  |  |
|--|---|--|--|
|  |   |  | <ul style="list-style-type: none"> <li>• Label with a corresponding checkbox (set to disabled) for displaying if vendor is preferred or not.</li> <li>• Label with a corresponding checkbox called “Confirm selected Quote”</li> <li>• Buttons:           <ul style="list-style-type: none"> <li>◦ Continue (if “Confirm selected Quote” is selected as true)</li> <li>◦ Back</li> <li>◦ Reject Request</li> </ul> </li> </ul> <p>[ALT]</p>  |
|  | <p><b>Step 2:</b> The user will select the one the:</p> <ul style="list-style-type: none"> <li>• “Confirm selected Quote” checkbox”</li> </ul> <p>Then click on the “Continue” button.</p> <p>[ALT]</p> |  | <p><b>Step 3:</b> The system makes use of the angular frontend and routes the requestor to the “Vendor Approved add details” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Due Diligence Checklist”</li> <li>• Label: “Please fill in the following details:”</li> <li>• Angular material stepper:</li> </ul> <p>First step labelled “DUE DILIGENCE CHECKLIST” containing the following:</p> <ul style="list-style-type: none"> <li>• Due Diligence physical copy confirmation</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>Label with a corresponding checkbox.</p> <ul style="list-style-type: none"><li>• Buttons: <b>Next, Cancel</b></li></ul> <p>second step labelled “FOUNDATIONAL DOCUMENTS” containing the following:</p> <ul style="list-style-type: none"><li>• “Mutual Non-Disclosure Agreement or Confidentiality Agreement” Label with a corresponding checkbox</li><li>• “Basic Company Information” Label with a corresponding checkbox</li><li>• “Ownership structure and affiliated entities” Label with a corresponding checkbox</li><li>• “Income tax number” Label with a corresponding checkbox</li><li>• “VAT number” label with a corresponding checkbox</li><li>• “Company registration document” label with a corresponding check box</li><li>• “Letters of good standing COID” Label with a</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• corresponding checkbox</li><li>• “Has BEE?” Label with a corresponding checkbox</li><li>• Mat expansion panel (set to default to hide) displayed when “Has BEE?” checkbox has been set to true displaying the following:</li><li>• “BEE level” Label with a corresponding input of type number set to have a default value of 1 and a max range between 1 to 5.</li><li>• “BEE Certificate” label with a corresponding input for type file</li><li>• Date of BEE validity label with its corresponding date picker input</li><li>• “Directors details and ID's” Label with a corresponding Checkbox</li><li>• “Company Resolution Agreement” Label with a corresponding input</li><li>• Contact Person Email Label with a corresponding input</li></ul> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>Buttons: <b>Back, Next</b></li> </ul> <p>Third step labelled “FINANCIALS” containing the following:</p> <ul style="list-style-type: none"> <li>“VAT Registration Certificate” Label with a corresponding Checkbox</li> <li>“Tax Clearance Certificate” Label with a corresponding Checkbox</li> <li>“Bank Stamped Confirmation Letter” Label with a corresponding Checkbox</li> <li>Buttons: <b>Next, Back</b></li> </ul> <p>Forth step labelled “SUB-CONTRACTING” containing the following:</p> <ul style="list-style-type: none"> <li>“Name of sub-contractor” label with a corresponding check box</li> <li>“Provide similar documents as for main supplier (In case of company)” label with a corresponding Checkbox</li> <li>“Provide copy of ID, qualifications, accreditations and professional</li> </ul> |
|--|--|--|--|

|  |  |   |
|--|--|---|
|  |  | <p>memberships (In case of individual)" Label with a corresponding Checkbox</p> <ul style="list-style-type: none"> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Fifth step labelled "INSURANCE" containing the following:</p> <ul style="list-style-type: none"> <li>• "General liability insurance" Label with a corresponding check box with a input of type file (if checked true)</li> <li>• "Cyber insurance " Label with a corresponding check box with a input of type file (if checked true)</li> <li>• "Professional indemnity insurance (if applicable)" Label with a corresponding check box with a input of type file (if checked true)</li> <li>• "Other specific insurance required per service/industry" Label with a corresponding check box with an input (if checked true)</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Sixth step labelled "LICENSES OR</p> |
|--|--|---|

|  |  |  |
|--|--|--|
|  |  | <p>PROFESSIONAL ACCREDITATION" containing the following:</p> <ul style="list-style-type: none"> <li>• "Licenses required" Label with a corresponding Checkbox</li> <li>• "Accreditation required" Label with a corresponding checkbox</li> <li>• "Professional membership required" Label with a corresponding Checkbox</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Seventh step labelled "INFORMATION SECURITY" containing the following:</p> <ul style="list-style-type: none"> <li>• "Business continuity plan" with a corresponding Checkbox</li> <li>• "Disaster recovery plan" label with a corresponding Checkbox</li> <li>• "Protection of personal information by design" Label with a corresponding Checkbox</li> <li>• Mat expansion panel (default set to hide)</li> </ul> |
|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>display when “Protection of personal information by design” checkbox is checked as true displaying the following:</p> <ul style="list-style-type: none"><li>• “Does the contract set out what personal data is used for what purpose?” label with a corresponding checklist.</li><li>• “Is the contracted partner a controller (C), joint controller (JC), processor (P) or sub-processor (SP)?” label with a corresponding mat select (option being either: C, JC, P or SP)</li><li>• “Depending on the controller/processor relationship, do you have a Data Processing Agreement or a Joint Controller Agreement in place?” label with corresponding checkbox</li><li>• “Does the contract highlight the importance of confidentiality?” label with its</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>corresponding checkbox</li><li>• “Does the contract provide for audits and inspections?” label with its corresponding checkbox</li><li>• “Is it clear who is accountable and liable for different activities?” label with its corresponding checkbox</li><li>• “Is there a provision to cover third party processing of data?” label with its corresponding checkbox</li><li>• “Does a process exist for managing data when the contract ends?” label with its corresponding checkbox</li><li>• “Is the personal data that's being processed detailed in your and their 'Record of Processing Activities'?” label with its corresponding checkbox</li><li>• “Does the supplier hold any form of certification for their processing activities?” label</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>with it corresponding checkbox</p> <ul style="list-style-type: none"> <li>• “History of data breaches and security incidents:” Label with a corresponding Checkbox</li> <li>• “Site visits to assess security controls (if required)” with a corresponding checkbox</li> <li>• Buttons: <b>Next</b>, <b>Back</b></li> </ul> <p>Eighth step labelled “POLICY REVIEW” containing the following:</p> <ul style="list-style-type: none"> <li>• “Information security or Data protection policy” with a corresponding checkbox</li> <li>• “Privacy policy” with a corresponding checkbox</li> <li>• “Data retention and destruction policy” with a corresponding checkbox</li> <li>• “Anti-bribery and anti-corruption policy” with a corresponding checkbox</li> <li>• “Ethics policy” with a corresponding checkbox</li> <li>• “Conflict of interest policy”</li> </ul> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  | <ul style="list-style-type: none"> <li>with a corresponding checkbox</li> <li>• “Customer complaints policy” with a corresponding checkbox</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Ninth step labelled “BUSINESS REFERENCES” containing the following:</p> <ul style="list-style-type: none"> <li>• “Details of at least two business references” with a corresponding checkbox</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Tenth step labelled “Done” containing the following:</p> <ul style="list-style-type: none"> <li>• Label: “*Please ensure that all details are correct*”</li> <li>• Buttons: <b>Create, Back</b></li> </ul> |  |
|  | <p><b>Step 4:</b> The user will then provide the following details as applicable for each step:</p> <ul style="list-style-type: none"> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> <li>• POPIA_Declaration</li> <li>• Due_Diligence_Doc</li> </ul> | <p><b>Step 5:</b> The system validates that all of the fields have been filled in and is in the required format for each step: (Most fields are optional except for the following)</p> <ul style="list-style-type: none"> <li>• Basic Company Information: Must be set to true.</li> </ul>   |  |

|  |   |   |
|--|---|---|
|  | <ul style="list-style-type: none"> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Direcor_Details_Provided</li> <li>• Company_Rsolution_Agreement_Provided</li> <li>• Subcontractor_Name_Provided</li> <li>• Company_Details_Provided</li> <li>• Individual_Details_Provided</li> <li>• General_Liability_Insurance_Present</li> <li>• Cyber_Insurance_Present</li> <li>• Professional_Indemnity_Insurance_Present</li> <li>• Other_Insurance_Required</li> <li>• Licenses_Required</li> <li>• Accreditation_Required</li> </ul> | <ul style="list-style-type: none"> <li>• Income tax number: Must be set to true.</li> </ul> <p>If “Has BEE?” has been checked as true:</p> <ul style="list-style-type: none"> <li>• BEE level must contain a number between 1 and 5.</li> <li>• BEE Certificate must not be null</li> <li>• Date of BEE validity: must not be null, Must not be earlier than a year then the current date.</li> </ul> <p>If “Protection of personal information by design” checkbox is set to true:</p> <ul style="list-style-type: none"> <li>• Personal Data Purpose: Must be set to true.</li> <li>• Contracted Partner Type: must be set to either controller (C), joint controller (JC), processor (P) or sub-processor (SP)</li> <li>• Data Processing Joint Controller Agreement: Must be set to true.</li> <li>• Confidentiality Importance Highlighted: Must be set to true.</li> <li>• Contract Audits Provisions Provided: Must be set to true.</li> </ul> |
|--|---|---|

|  |   |  |   |
|--|---|--|---|
|  | <ul style="list-style-type: none"> <li>• Professional_Membership_Required</li> <li>• Business_Continuity_Present</li> <li>• Disaster_Recovery_Plan_Present</li> <li>• POPI_Present</li> <li>• Data_Security_Breaches_Present</li> <li>• Site_Visits_Present</li> <li>• Information_Security_Policy_Present</li> <li>• Privacy_Policy_Present_Present</li> <li>• Data_Retention_Destruction_Policy_Present</li> <li>• Anti_Bribery_Corruption_Policy_Present</li> <li>• Ethics_Policy_Present</li> <li>• Conflict_Of_Interest_Policy_Present</li> <li>• Customer_Complaints_Policy_Present</li> <li>• Business_References_Present</li> <li>• Contracted_Partner_Type_ID (FK)</li> <li>• Personal_Data_Purpose</li> <li>• DataProcessing_JointController_Agreement</li> <li>• Confidentiality_Importance_Highlighted</li> </ul> |  | <ul style="list-style-type: none"> <li>• Activity_Liability_Present: Must be set to true.</li> <li>• Third_Party_Data_Processing_Provisioned: Must be set to true.</li> <li>• Contract_End_Data_Management_Provided: Must be set to true.</li> <li>• Contract_End_Data_Management_Provided: Must be set to true.</li> <li>• Personal_Data_Processing_Details_Present: Must be set to true.</li> <li>• Processing_Activities_Certification_Held: Must be set to true.</li> </ul> |
|--|---|--|---|

|  |  |  |  |
|--|--|--|--|
|  | <ul style="list-style-type: none"> <li>• Contract_Audits_Provisions_Provided</li> <li>• Activity_Liability_Present</li> <li>• Third_Party_Data_Processing_Proposed</li> <li>• Contract_End_Data_Management_Provided</li> <li>• Personal_Data_Processing_Details_Present</li> <li>• Processing_Activities_Certification_Held</li> <li>• General_Liability_Insurance_Present_Document</li> <li>• Cyber_Insurance_Present_Document</li> <li>• Professional_Indemnity_Insurance_Present_Document</li> <li>• Other_Insurance_Required_Document</li> </ul> |  |  |
|  | <p><b>Step 6:</b> The user clicks on the “Create“ Button” [ALT]</p>  | <p><b>Step 7:</b> The system then calls the VendorFileAdd() Function in the angular frontend which will capture the bee certificate document provided in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an</p> |  |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>HTTPPOST request and it will send a formData object used to insert the file at a specific path location on the API side.</p> <p>Afterwards a part of the filepath to file is send back where the system then calls the AddBEEDetails () Function in the angular frontend which will capture all of the related values for BEE provided in the angular form.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Vendor_BEE Object.</p> <p>The API will then receive the Vendor_BEE object and add the following Details in the <b>Vendor_BEE</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"><li>• BEE_ID (PK)</li><li>• Vendor_ID (FK)</li><li>• BEE_Level</li><li>• BEE_Certificate</li><li>• Date</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>Inside of that angular function the newly created Vendor_BEE object is returned and the AddBEEDelayJob() function.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and it will send The Vendor_ID and Date based from the newly created Vendor_BEE object. This will then create a “DelayJob” which will call on a HTTPPOST request. The API will then receive the VendorID, Date and NotifyWhenNumber and add the following Details in the <b>Notification</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Notification_Type_ID (set to 2)</li> <li>• User_ID</li> <li>• Name</li> <li>• Send_Date</li> </ul> <p>The system then calls the AddDueDiligence () Function in the angular frontend which will capture all of the values provided in the angular form</p> <p>The angular frontend will then call the</p> |
|--|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Due_Dilligence Object.</p> <p>The API will then receive the Due_Dilligence object and add the following Details in the <b>DUE_DILIGENCE</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Due_Diligence_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_ Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Direcor_Details_Provided</li> </ul> |
|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Company_Resolution_Agreement_Provided</li><li>• Subcontractor_Name_Provided</li><li>• Company_Details_Provided</li><li>• Individual_Details_Provided</li><li>• General_Liability_Insurance_Present</li><li>• Cyber_Insurance_Present</li><li>• Professional_Indemnity_Insurance_Present</li><li>• Other_Insurance_Required</li><li>• Licenses_Required</li><li>• Accreditation_Required</li><li>• Professional_Membership_Required</li><li>• Business_Continuity_Present</li><li>• Disaster_Recovery_Plan_Present</li><li>• POPI_Present</li><li>• Data_Security_Breach_Present</li><li>• Site_Visits_Present</li><li>• Information_Security_Policy_Present</li><li>• Privacy_Policy_Present_Present</li><li>• Data_Retention_Destruction_Policy_Present</li><li>• Anti_Bribery_Corruption_Policy_Present</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Ethics_Policy_Present</li> <li>• Conflict_Of_Interest_Policy_Present</li> <li>• Customer_Complaints_Policy_Present</li> <li>• Business_References_Present</li> </ul> <p>The request brings back the newly added Due_diligence object that is used by system should the user have provided popi details it then calls the AddPOPI() Function in the angular frontend which will capture all of the values provided details related to popi in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the POPI Object.</p> <p>The API will then receive the POPI object and add the following Details in the <b>POPI</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• POPI_ID (PK)</li> <li>• Contracted_Partner_Type_ID (FK)</li> </ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Due_Diligence_I D (FK)</li><li>• Personal_Data_ Purpose</li><li>• DataProcessing _JointController _Agreement</li><li>• Confidentiality_I mportance_High lighted</li><li>• Contract_Audits _Provisions_Pro vided</li><li>• Activity_Liability _Present</li><li>• Third_Party_Dat a_Processing_P rovisioned</li><li>• Contract_End_D ata_Managemen t_Provided</li><li>• Personal_Data_ Processing_Det ails_Present</li><li>• Processing_Acti vities_Certificati on_Held</li></ul> <p>The system then calls the VendorFileAdd() Function in the angular frontend which will capture the types of insurance document provided in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send a formData object used to insert the file at a</p> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  | <p>specific path location on the API side.</p> <p>Afterwards a part of the filepath to file is send back where the system then calls the AddInsurance () Function in the angular frontend which will capture all of the related values for Insurance details provided in the angular form.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Insurance Object.</p> | <p>The API will then receive the Vendor_Insurance object and add the following Details in the <b>Vendor_ Insurance</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Confirmation_Doc</li> </ul> |
|  |  |  | <p><b>Step 8:</b> The system transforms the Create button into a check mark and hides the cancel button.</p>   |

|                           |   |  |  |
|---------------------------|---|--|--|
|                           |   |  | <p>The system notifies the user by displaying a notification containing the following components:</p> <ul style="list-style-type: none"> <li>• “CREATE SUCCESSFUL” header</li> <li>• “The Due Diligence has been CREATED successfully!” body text</li> </ul> <p>The text disappears after 1750 seconds routing the user back to “Vendor Unofficial Vendorlist” screen.</p> |
| <b>Alternate courses:</b> | <p><b>[ALT] Step 2:</b> The user is a general manager and the onboard request is sole supplier requiring management approval.</p> <p>The system will respond by making use of the angular frontend with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Onboard Request # ‘Onboard Request Number’ ”</li> <li>• Label: “Please Approve or Reject the Request”</li> <li>• Tab group containing 1 tab with the following contents:</li> <li>• Label with a corresponding text box called company name (set to read only)</li> <li>• Label with a corresponding text box called company email (set to read only)</li> <li>• Label with a corresponding text box called Reason (set to read only)</li> <li>• Label with a corresponding file name with a download or open in new tab function when clicked on</li> <li>• Label with a corresponding checkbox called “Confirm Sole Supplier approval”.</li> <li>• Buttons: <ul style="list-style-type: none"> <li>◦ Continue (if “Confirm Sole Supplier approval” is selected as true)</li> <li>◦ Back</li> <li>◦ Reject Request</li> </ul> </li> </ul> <p><b>[ALT] Step 3a:</b> User does not want to continue and click the “Back” Button. Use Case Terminates.</p> |  |  |

|  |  |
|--|--|
|  | <p><b>[ALT] Step 3b:</b> The user clicks on the “Reject Request” button and the onboard request and vendor status are updated and set to rejected.</p> <p><b>[ALT] Step 6:</b> The user clicks on the “Back” Button. The system routes the user to the <u>Vendor Approve Screen</u>. <b>Return to step 3</b></p> |
| <b>Conclusion:</b>                                   | The system routes the user to the “Vendor Unofficial Vendorlist” Screen  |
| <b>Post-condition:</b>                               | New data is added or updated to the following tables: <ul style="list-style-type: none"> <li>• DUE_DILIGENCE</li> <li>• Vendor_BEE</li> <li>• Vendor</li> <li>• Onboard_Request</li> <li>• POPI</li> <li>• Insurnace</li> </ul>  |
| <b>Business rules</b>                                | Only one quote may be approved per onboard request   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>   |

Table 205: 6.5 Approve Onboard Request

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                       |                                       |  |
|-----------------------|---------------------------------------|--|
| <b>Use case name:</b> | Export due diligence vendor checklist | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>   | 6.10                                  | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>      | high                                  | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>        | Moyo                                  | System Design: <input checked="" type="checkbox"/> |

|                                       |   |                        |   |
|---------------------------------------|---|------------------------|---|
| <b>Primary business actor</b>         | User (PBA)  |                        |   |
| <b>Primary system actor</b>           | None  |                        |   |
| <b>Other participating actors:</b>    | None  |                        |   |
| <b>Other interested stakeholders:</b> | None  |                        |   |
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to generate a due diligence checklist an approved vendor from an onboard request. The user will click on the <u>View</u> button on the <u>Vendor Unofficial Vendorlist</u> Screen for an approved vendor. The system will load the <u>Approve Vendor</u> Screen. The user will then click on the black color text called “Generate Checklist”. The system will make use of pdfmake and generate a Due Diligence checklist based on data from the <b>Due_Diligence</b> and <b>POPI</b> table and display it for the user in other tab</p> |                        |   |
| <b>Pre-condition:</b>                 | The user must be logged into the system.  |                        |   |
| <b>Trigger:</b>                       | Requestor requests to view all of the onboard request saved on the system.  |                        |   |
| <b>Typical course Of events:</b>      | <b>Actor Action</b>   | <b>System Response</b> |   |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |   |                        | <p><b>Step 1:</b> The system makes use of the angular frontend and routes the user to the “Vendor Approved” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Onboard Request # ‘Onboard Request Number’”</li> <li>• Button with a mat icon “arrow_back” displayed inline with the header</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Tab group having the following contents:</li> <li>• Label with a corresponding text box called company name (set to read only)</li> <li>• Label with a corresponding text box called company email (set to read only)</li> <li>• Label with a corresponding text box called Reason (set to read only and only displayed if approved vendor is a sole supplier)</li> <li>• Label with a corresponding file name with a download or open in new tab function when clicked on</li> <li>• Label called “Export Due Diligence” with a corresponding text called “Generate Checklist” with a download or open in new tab function when clicked on.</li> <li>• Buttons             <ul style="list-style-type: none"> <li>○ Reselect Quote</li> <li>○ Reject Request</li> </ul> </li> </ul> |
|  |  |  | <b>Step 2:</b> The system will capture the ID that was passed via the header of the angular frontend  |

|  |  |   |
|--|--|---|
|  |  | <p>and call the GetRequestByID() function. This function will access the appropriate HTTPGET methods on the ASP.NET Core API.</p> <p>The GetRequestByID() function will use the ID variable and return the first object where the ID variable is equal to the Onboard_Request_Id attribute in the <b><u>Onboard Request</u></b> Entity. The system will retrieve the following by making use of a SQL Read:</p> <ul style="list-style-type: none"> <li>• Name equal to Company Name</li> <li>• Email equal to Company email</li> <li>• Quote equal to Company quote</li> <li>• PreferredVendor equal to Preferred vendor</li> </ul> |
|  | <p><b>Step 3:</b> The user will click on the “Generate Checklist” text.</p> <p>[ALT]</p> | <p><b>Step 4:</b> The system will use the angular frontend to call the function called GenerateList() where inside the function GetDueDiligence() is called.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and it will send it the VendorID.</p>   |

|  |  |   |
|--|--|---|
|  |  | <p>The API will then obtain the Due_Diligence object details from the <b>Due_Diligence</b> entity with a SQL Read Query</p> <p>Afterwards the system checks whether popi has been added to the object if it has it will call on the function GetPOPI()</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and it will send it the DueDiligenceID.</p> <p>The API will then obtain the POPI object details from the <b>POPI</b> entity with a SQL Read Query.</p> <p>In both cases whether it has POPI or not the system will make use of pdfMake to format and generate a pdf file contain the following details:</p> <p>Due Diligence:</p> <ul style="list-style-type: none"><li>• Due_Diligence_Doc</li><li>• Mutual_Nda_Signed</li><li>• Basic_Company_Info_Provided</li></ul> |
|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Group_Structure_Provided</li><li>• Income_Tax_Number_Provided</li><li>• Tax_Clearance_Certificate_Provided</li><li>• Vat_Number_Provided</li><li>• Vat_Reg_Certificate_Provided</li><li>• Company_Reg_Doc_Provided</li><li>• Letter_Of_Good_Standing_Provided</li><li>• B_BBEE_Certificate_Provided</li><li>• Direcor_Details_Provided</li><li>• Company_Resolution_Agreement_Provided</li><li>• Subcontractor_Name_Provided</li><li>• Company_Details_Provided</li><li>• Individual_Details_Provided</li><li>• General_Liability_Insurance_Present</li><li>• Cyber_Insurance_Present</li><li>• Professional_Indemnity_Insurance_Present</li><li>• Other_Insurance_Required</li><li>• Licenses_Required</li><li>• Accreditation_Required</li><li>• Professional_Membership_Required</li><li>• Business_Continuity_Present</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Disaster_Recovery_Plan_Present</li><li>• POPI_Present</li><li>• Data_Security_Breach_Present</li><li>• Site_Visits_Present</li><li>• Information_Security_Policy_Present</li><li>• Privacy_Policy_Present_Present</li><li>• Data_Retention_Destruction_Policy_Present</li><li>• Anti_Bribery_Corruption_Policy_Present</li><li>• Ethics_Policy_Present</li><li>• Conflict_Of_Interest_Policy_Present</li><li>• Customer_Complaints_Policy_Present</li><li>• Business_Referrals_Present</li><li>• Contracted_Partner_Type_ID_(FK)</li><li>• Personal_Data_Purpose</li><li>• DataProcessing_JointController_Agreement</li><li>• Confidentiality_Importance_Highlighted</li><li>• Contract_Audits_Provisions_Provided</li><li>• Activity_Liability_Present</li></ul> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>• Third_Party_Data_Processing_Provisioned</li> <li>• Contract_End_Data_Management_Provided</li> <li>• Personal_Data_Processing_Details_Present</li> <li>• Processing_Activities_Certification_Held</li> </ul> <p>POPI (if include):</p> <ul style="list-style-type: none"> <li>• Contracted_Partner_Type</li> <li>• Personal_Data_Purpose</li> <li>• DataProcessing_JointController_Agreement</li> <li>• Confidentiality_Importance_Highlighted</li> <li>• Contract_Audits_Provisions_Provided</li> <li>• Activity_Liability_Present</li> <li>• Third_Party_Data_Processing_Provisioned</li> <li>• Contract_End_Data_Management_Provided</li> <li>• Personal_Data_Processing_Details_Present</li> <li>• Processing_Activities_Certification_Held</li> </ul> <p>Opening the pdf into a new tab for the user to view or download.</p> |
|--|--|--|--|

**Alternate courses:**

**[ALT] Step 3a:** The user clicks on “Reselect Quote” Button.

The system will use the angular frontend to call the function called UpdateOnboardRequestStatus () where inside validation is done to check if user may reselect a the onboard request quote if successful the function ChangeVendorStatus () and ChangeOnboardStatus() is called.

The angular frontend will then call the functions which will communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the Status\_ID and onboard\_Request\_Id or vendor\_ID.

The API will then receive the Status\_ID, onboard\_Request\_Id and update the following Details in the **onboard\_Request** entity with a SQL update Query:

- Status\_ID

The API will then receive the Status\_ID, vendor\_ID and update the following Details in the **Vendor** entity with a SQL update Query:

- Vendor\_Status\_ID

Finally invoking use case 6.5

**[ALT] Step 3b:** User clicks on the “Rekect Request” Button.

The system will use the angular frontend to call the function called RejectRequest() where inside validation is done to check if user may reselect a the onboard request quote if successful the function ChangeVendorStatus() and ChangeOnboardStatus() is called.

The angular frontend will then call the functions which will communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the Status\_ID and onboard\_Request\_Id or vendor\_ID.

The API will then receive the Status\_ID, onboard\_Request\_Id and update the following Details in the **onboard\_Request** entity with a SQL update Query:

- Status\_ID

The API will then receive the Status\_ID, vendor\_ID and update the following Details in the **Vendor** entity with a SQL update Query:

- Vendor\_Status\_ID

Finally I notification is display that request has been rejected and route the user back to “Vendor Unofficial Vendorlist” screen.

**[ALT] Step 3c:** The user click on the “arrow\_back” mat icon button. **use case Terminates.**

|  |   |
|--|---|
| <b>Conclusion:</b>                                   | A new tab is open on the user web browser contain the generate Due diligence checklist. |
| <b>Post-condition:</b>                               | The user can now view or download the generate due diligence checklist.                 |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>                                |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>                                |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                                |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                                |

Table 206: 6.10 Export Due Diligence Vendor Checklist

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                                    |  |  |
|------------------------------------|--|--|
| <b>Use case name:</b>              | Generate Sole Supplier Review Notification | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                | 6.11                                       | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                   | high                                       | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                     | Moyo                                       | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>      | Time (PBA)                                 |  |
| <b>Primary system actor</b>        | None                                       |  |
| <b>Other participating actors:</b> | None                                       |  |

|                                       |  |  |   |
|---------------------------------------|--|--|---|
| <b>Other interested stakeholders:</b> | None   |  |   |
| <b>Description:</b>                   | <p>This use case describes the event where a monthly notification is send to the GRC Officer to review sole supplier performance.</p> <p>The use case begins where the system checks whether the date is the beginning of a new month. If it is the system obtain all details from Vendor and Sole_Supplier table using sql read query.</p> <p>The use case will then conclude when it add the notification to the notification table.</p> |  |   |
| <b>Pre-condition:</b>                 | The must be sole supplier that exist on the system.  |  |   |
| <b>Trigger:</b>                       | It becomes the first day of the month in the first hour of the day and in the first minute of the hour.  |  |   |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>  | <b>System Response</b>   |   |
|                                       |  | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |  | <p><b>Step 1:</b> The system calls the function SoleSupplierPerformanceNotification() from VendorRepository to retrieve all the Vendor and Sole_Supplier details stored in the database using a sql read query.</p> <p>It is then put into 2 separate arrays of objects</p> <p>[ALT]</p> | <p><b>Step 2:</b> The system then loops through the Vendor object to find only vendors that are sole suppliers and uses the Sole_Supplier object to filter it to those that are</p> |

|  |  |  |
|--|--|--|
|  |  | approved by management.  |
|  |  | <p><b>Step 3:</b> The system then calls AddSoleSupplierNotification() function and sends a Description and date.</p> <p>The system will then add the new details to Notification table using an sql Insert query:</p> <ul style="list-style-type: none"> <li>• User_ID</li> <li>• Name</li> <li>• Send_Date</li> </ul> |
| <b>Alternate courses:</b>                            | <b>[ALT] Step 1:</b> The system found no sole supplier. <b>Use Case Terminates</b> |  |
| <b>Conclusion:</b>                                   | The system has added the new notification to Notification table                    |  |
| <b>Post-condition:</b>                               | The new notification is display to the GRC Officer                                 |  |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>                           |  |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>                           |  |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                           |  |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                           |  |

Table 207: 6.11 Generate Sole Supplier Review Notification

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

|                                       |   |                        |   |
|---------------------------------------|---|------------------------|---|
| <b>Use case name:</b>                 | Generate BEE Certificate Expiry Notification  |                        | <b>USE CASE TYPE</b>  |
| <b>Use case id:</b>                   | 6.12  |                        | Business Requirements: <input type="checkbox"/>   |
| <b>Priority:</b>                      | high  |                        | System Analysis: <input type="checkbox"/>   |
| <b>Source:</b>                        | Moyo  |                        | System Design: <input checked="" type="checkbox"/>  |
| <b>Primary business actor</b>         | Time (PBA)  |                        |   |
| <b>Primary system actor</b>           | None  |                        |   |
| <b>Other participating actors:</b>    | None  |                        |   |
| <b>Other interested stakeholders:</b> | None  |                        |   |
| <b>Description:</b>                   | <p>This use case describes the event where a notification is send to the GRC Officer 3 weeks, 1 week and on the day of a vendor BEE expiry date.</p> <p>The use case begins where the system checks whether the date is the either 3 weeks before, 1 week before or the day of vendor BEE expiry date. The use case will then conclude when it adds the notification to the notification table.</p> |                        |   |
| <b>Pre-condition:</b>                 | The requestor has to be logged in to the system.  |                        |   |
| <b>Trigger:</b>                       | The requestor requests to add a vendor's detail to the system   |                        |   |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>   | <b>System Response</b> |   |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b>   |
|                                       |   |                        | <b>Step 1:</b> The system checks that it either 3 weeks before, 1 week before or the day of a |

|  |   |   |
|--|---|---|
|  |   | vendor BEE expiry date.   |
|  |   | <p><b>Step 2:</b> if it is the system calls the function AddVendorBEENotificationAsync () from VendorRepository and receives a date, vendorID and Description and adds into the notification table using sql insert query:</p> <ul style="list-style-type: none"> <li>• User_ID</li> <li>• Name</li> <li>• Send_Date</li> </ul> |
|  |   | <p><b>Step 3:</b> The system then displays the notification in the angular front end to the GRC Officer.</p>  |
| <b>Alternate courses:</b>                            |   |   |
| <b>Conclusion:</b>                                   | The system has added the notification into the notification table |   |
| <b>Post-condition:</b>                               | The notification is display to the GRC Officer                    |   |
| <b>Business rules</b>                                | <ul style="list-style-type: none"> <li>• None</li> </ul>          |   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>          |   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>          |   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>          |   |

Table 208: 6.12 Generate BEE Certificate Expiry Notification

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/23 | Version: 1.0 |

| Use case name:                 | Update Approved onboard request   |                 | USE CASE TYPE                                      |
|--------------------------------|---|-----------------|--|
| Use case id:                   | 6.13  |                 | Business Requirements: <input type="checkbox"/>    |
| Priority:                      | high  |                 | System Analysis: <input type="checkbox"/>          |
| Source:                        | Moyo  |                 | System Design: <input checked="" type="checkbox"/> |
| Primary business actor         | GRC Officer (PBA)   |                 |  |
| Primary system actor           | None  |                 |  |
| Other participating actors:    | None  |                 |  |
| Other interested stakeholders: | None  |                 |  |
| Description:                   | <p>The use case describes the event where the user will want to edit the due diligence checklist, BEE or POPI details of a approved vendor from an onboard request. The user will click on the <u>Update</u> button on the <u>Vendor Unofficial Vendorlist</u> Screen. The system will load the <u>Vendor Approval Edit</u> Screen. The user will then provide the updated details of the available vendors that they wish to approve for the due diligence checklist, BEE or POPI, the system will then request the user to provide confirmation details. The user does so by then clicking on the “Update” button.</p> <p>Finally the system Displays a success notification to the user.</p> |                 |  |
| Pre-condition:                 | <p>The requestor has to be logged in to the system.</p> <p>There must be an onboard request stored on the system that has yet been approved or reject.</p>  |                 |  |
| Typical course                 | Actor Action  | System Response |  |

| Of events: |  | Manual Action | Automated Action  |
|------------|--|---------------|---|
|            |  |               | <p><b>Step 1:</b> The system makes use of the angular frontend and routes the requestor to the “Vendor Approved Edit” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Due Diligence Checklist”</li> <li>• Label: “Please fill in the following details:”</li> <li>• jAngular material stepper:</li> </ul> <p>First step labelled “DUE DILIGENCE CHECKLIST” containing the following:</p> <ul style="list-style-type: none"> <li>• Due Diligence physical copy confirmation Label with a corresponding checkbox.</li> <li>• Buttons: <b>Next, Cancel</b></li> </ul> <p>second step labelled “FOUNDATIONAL DOCUMENTS” containing the following:</p> <ul style="list-style-type: none"> <li>• “Mutual Non-Disclosure Agreement or Confidentiality Agreement” Label with a corresponding checkbox</li> </ul> |

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• “Basic Company Information” Label with a corresponding checkbox</li><li>• “Ownership structure and affiliated entities” Label with a corresponding checkbox</li><li>• “Income tax number” Label with a corresponding checkbox</li><li>• “VAT number” label with a corresponding checkbox</li><li>• “Company registration document” label with a corresponding check box</li><li>• “Letters of good standing COID” Label with a corresponding checkbox</li><li>• “Has BEE?” Label with a corresponding checkbox</li><li>• Mat expansion panel (set to default to hide) displayed when “Has BEE?” checkbox has been set to true displaying the following:<ul style="list-style-type: none"><li>• “BEE level” Label with a corresponding input of type number set to</li></ul></li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <p>have a default value of 1 and a max range between 1 to 5.</p> <ul style="list-style-type: none"> <li>• “BEE Certificate” label with a corresponding input for type file</li> <li>• Date of BEE validity label with its corresponding date picker input</li> <li>• “Directors details and ID's” Label with a corresponding Checkbox</li> <li>• “Company Resolution Agreement” Label with a corresponding input</li> <li>• Contact Person Email Label with a corresponding input</li> <li>• Buttons: <b>Back</b>, <b>Next</b></li> </ul> <p>Third step labelled “FINANCIALS” containing the following:</p> <ul style="list-style-type: none"> <li>• “VAT Registration Certificate” Label with a corresponding Checkbox</li> <li>• “Tax Clearance Certificate” Label with a corresponding Checkbox</li> <li>• “Bank Stamped Confirmation Letter” Label</li> </ul> |
|--|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>with a corresponding Checkbox</p> <ul style="list-style-type: none"> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Forth step labelled “SUB-CONTRACTING” containing the following:</p> <ul style="list-style-type: none"> <li>• “Name of sub-contractor” label with a corresponding check box</li> <li>• “Provide similar documents as for main supplier (In case of company)” label with a corresponding Checkbox</li> <li>• “Provide copy of ID, qualifications, accreditations and professional memberships (In case of individual)” Label with a corresponding Checkbox</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Fifth step labelled “INSURANCE” containing the following:</p> <ul style="list-style-type: none"> <li>• “General liability insurance” Label with a corresponding check box with a file link (if checked true)</li> </ul> |
|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"><li>• “Add new General liability Insurance Document?” Label with a corresponding input of type file.</li><li>• “Cyber insurance “ Label with a corresponding check box with a file link(if checked true)</li><li>• “Add a new Cyber Insurance Document?” label with a corresponding input of type file</li><li>• “Professional indemnity insurance (if applicable)” Label with a corresponding check box with a file link (if checked true)</li><li>• “Add a new Professional Indemnity Insurance Document?” label with corresponding input of type file</li><li>• “Other specific insurance required per service/industry” Label with a corresponding check box with an file link (if checked true)</li><li>• “Add new Other specific insurance</li></ul> |
|--|--|--|--|

|  |  |  |
|--|--|--|
|  |  | <p>Document?” label with its corresponding input of type file.</p> <ul style="list-style-type: none"> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Sixth step labelled “LICENSES OR PROFESSIONAL ACCREDITATION” containing the following:</p> <ul style="list-style-type: none"> <li>• “Licenses required” Label with a corresponding Checkbox</li> <li>• “Accreditation required” Label with a corresponding checkbox</li> <li>• “Professional membership required” Label with a corresponding Checkbox</li> <li>• Buttons: <b>Next, Back</b></li> </ul> <p>Seventh step labelled “INFORMATION SECURITY” containing the following:</p> <ul style="list-style-type: none"> <li>• “Business continuity plan” with a corresponding Checkbox</li> <li>• “Disaster recovery plan” label with a corresponding Checkbox</li> </ul> |
|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• “Protection of personal information by design” Label with a corresponding Checkbox</li></ul> <p>Mat expansion panel (default set to hide) display when “Protection of personal information by design” checkbox is checked as true displaying the following:</p> <ul style="list-style-type: none"><li>• “Does the contract set out what personal data is used for what purpose?” label with a corresponding checklist.</li><li>• “Is the contracted partner a controller (C), joint controller (JC), processor (P) or sub-processor (SP)?” label with a corresponding mat select (option being either: C, JC, P or SP)</li><li>• “Depending on the controller/processor relationship, do you have a Data Processing Agreement or a Joint Controller Agreement in place?” label with</li></ul> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"><li>corresponding checkbox</li><li>• “Does the contract highlight the importance of confidentiality?” label with its corresponding checkbox</li><li>• “Does the contract provide for audits and inspections?” with its corresponding checkbox</li><li>• “Is it clear who is accountable and liable for different activities?” label with its corresponding checkbox</li><li>• “Is there a provision to cover third party processing of data?” label with its corresponding checkbox</li><li>• “Does a process exist for managing data when the contract ends?” label with its corresponding checkbox</li><li>• “Is the personal data that's being processed detailed in your and their 'Record of Processing Activities'?” label with its corresponding checkbox</li></ul> |
|--|--|--|--|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>corresponding checkbox</li><li>• “Does the supplier hold any form of certification for their processing activities?” label with it corresponding checkbox</li></ul> <p>After the expansion panel:</p> <ul style="list-style-type: none"><li>• “History of data breaches and security incidents:” Label with a corresponding Checkbox</li><li>• “Site visits to assess security controls (if required)” with a corresponding checkbox</li><li>• Buttons: <b>Next, Back</b></li></ul> <p>Eighth step labelled “POLICY REVIEW” containing the following:</p> <ul style="list-style-type: none"><li>• “Information security or Data protection policy” with a corresponding checkbox</li><li>• “Privacy policy” with a corresponding checkbox</li><li>• “Data retention and destruction policy” with a corresponding checkbox</li></ul> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <ul style="list-style-type: none"> <li>“Anti-bribery and anti-corruption policy” with a corresponding checkbox</li> <li>“Ethics policy” with a corresponding checkbox</li> <li>“Conflict of interest policy” with a corresponding checkbox</li> <li>“Customer complaints policy” with a corresponding checkbox</li> <li>Buttons: <b>Next, Back</b></li> </ul> <p>Ninth step labelled “BUSINESS REFERENCES” containing the following:</p> <ul style="list-style-type: none"> <li>“Details of at least two business references” with a corresponding checkbox</li> <li>Buttons: <b>Next, Back</b></li> </ul> <p>Tenth step labelled “Done” containing the following:</p> <ul style="list-style-type: none"> <li>Label: “*Please ensure that all details are correct*”</li> <li>Buttons: <b>Update, Back</b></li> </ul> |
|  |  |  | <b>Step 2:</b> The system will use the Angular frontend to navigate to   |

|  |  |  |   |
|--|--|--|---|
|  |  |  | the “Vendor Approve Edit” page while sending the ID of the Vendor ID to be updated to that respective page  |
|  |  |  | <p><b>Step 3:</b> The system will use the angular frontend to call the function <code>GetDueDiligence()</code>. The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an <code>HTTPGET</code> request and it will send it the <code>VendorID</code> obtained through the parameter send through the route.</p> <p>The API will then obtain the <code>Due_Diligence</code> object details from the <b>Due_Diligence</b> entity with a SQL Read Query</p> <p>Afterwards the system checks whether <code>popi</code> has been added to the object if it has it will call on the function <code>GetPOPI()</code></p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an <code>HTTPGET</code> request and</p> |

|  |  |  |
|--|--|--|
|  |  | <p>it will send it the DueDiligenceID.</p> <p>The API will then obtain the POPI object details from the <b>POPI</b> entity with a SQL Read Query.</p> <p>The system also checks whether Vendor_BEE has been previously added if it has it will call on the function GetBEEDetails()</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and it will send it the VendorID.</p> <p>The API will then obtain the POPI object details from the <b>Vendor_BEE</b> entity with a SQL Read Query.</p> <p>Next The system checks whether any insurance are linked to the vendor if it does it will call on the function GetInsuranceByID()</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and</p> |
|--|--|--|

|  |  |   |
|--|--|---|
|  |  | <p>it will send it the VendorID.</p> <p>The API will then obtain the Insurance object details from the <b>Vendor_Insurance</b> entity with a SQL Read Query.</p> <p>Finally the obtained details will prepopulate the different inputs based on the following details:</p> <ul style="list-style-type: none"> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> <li>• POPIA_Declaration</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Direcor_Details_Provided</li> </ul> |
|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Company_Resolution_Agreement_Provided</li><li>• Subcontractor_Name_Provided</li><li>• Company_Details_Provided</li><li>• Individual_Details_Provided</li><li>• General_Liability_Insurance_Present</li><li>• Cyber_Insurance_Present</li><li>• Professional_Indemnity_Insurance_Present</li><li>• Other_Insurance_Required</li><li>• Licenses_Required</li><li>• Accreditation_Required</li><li>• Professional_Membership_Required</li><li>• Business_Continuity_Present</li><li>• Disaster_Recovery_Plan_Present</li><li>• POPI_Present</li><li>• Data_Security_Breach_Present</li><li>• Site_Visits_Present</li><li>• Information_Security_Policy_Present</li><li>• Privacy_Policy_Present_Present</li><li>• Data_Retention_Destruction_Policy_Present</li><li>• Anti_Bribery_Corruption_Policy_Present</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Ethics_Policy_Present</li><li>• Conflict_Of_Interest_Policy_Present</li><li>• Customer_Complaints_Policy_Present</li><li>• Business_Referrals_Present</li><li>• Contracted_Partner_Type_ID(FK)</li><li>• Personal_Data_Purpose</li><li>• DataProcessing_JointController_Agreement</li><li>• Confidentiality_Importance_Highlighted</li><li>• Contract_Audits_Provisions_Provided</li><li>• Activity_Liability_Present</li><li>• Third_Party_Data_Processing_Provisioned</li><li>• Contract_End_Data_Management_Provided</li><li>• Personal_Data_Processing_Details_Present</li><li>• Processing_Activities_Certification_Held</li><li>• General_Liability_Insurance_Present_Document</li><li>• Cyber_Insurance_Present_Document</li><li>• Professional_Indemnity_Insurance_Present_Document</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Other_Insurance _Required_Docu ment</li> </ul> |
|  | <p><b>Step 4:</b> The user provides the following updated details as they wish:</p> <ul style="list-style-type: none"> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> <li>• POPIA_Declarat ion</li> <li>• Due_Diligence_ Doc</li> <li>• Mutual_Nda_Si gned</li> <li>• Basic_Company _Info_Provided</li> <li>• Group_Structur e_Provided</li> <li>• Income_Tax_Nu mber_Provided</li> <li>• Tax_Clearance_ Certificate_Provi ded</li> <li>• Vat_Number_Pr ovided</li> <li>• Vat_Reg_Certifi cate_Provided</li> <li>• Company_Reg_ Doc_Provided</li> <li>• Letter_Of_Good _Standing_Prov ided</li> <li>• B_BBEE_Certifi cate_Provided</li> <li>• Direcor_Details _Provided</li> <li>• Company_Reso lution_Agreeme nt_Provided</li> <li>• Subcontractor_ Name_Provided</li> </ul> | <p><b>Step 5:</b> The system validates that all of the fields have been filled in and is in the required format for each step: (Most fields are optional except for the following)</p> <ul style="list-style-type: none"> <li>• Basic Company Information: Must be set to true.</li> <li>• Income tax number: Must be set to true.</li> </ul> <p>If “Has BEE?” has been check as true:</p> <ul style="list-style-type: none"> <li>• BEE level must contain a number between 1 and 5.</li> <li>• BEE Certificate must not be null</li> <li>• Date of BEE validity: must not be null, Must not be earlier than a year then the current date.</li> </ul> <p>If “Protection of personal information by design” checkbox is set to true:</p> <ul style="list-style-type: none"> <li>• Personal Data Purpose: Must be set to true.</li> <li>• Contracted Partner Type: must be set to</li> </ul> |   |

|  |  |   |
|--|--|---|
|  | <ul style="list-style-type: none"> <li>• Company_Detai ls_Provided</li> <li>• Individual_Detai ls_Provided</li> <li>• General_Liabilit y_Insurance_Pres ent</li> <li>• Cyber_Insuranc e_Present</li> <li>• Proffesional_Ind emnity_Insuranc e_Present</li> <li>• Other_Insuranc e_Required</li> <li>• Licenses_Requi red</li> <li>• Accreditation_R equired</li> <li>• Proffesional_Me mbership_Requi red</li> <li>• Business_Conti nuity_Present</li> <li>• Disaster_Recov ery_Plan_Pres ent</li> <li>• POPI_Present</li> <li>• Data_Security_ Breaches_Pres ent</li> <li>• Site_Visits_Pres ent</li> <li>• Information_Sec urity_Policy_Pres ent</li> <li>• Privacy_Policy_ Present_Presen t</li> <li>• Data_Retention _Destruction_P olicy_Present</li> <li>• Anti_Bribery_Co rruption_Policy_P resent</li> <li>• Ethics_Policy_P resent</li> </ul> | <ul style="list-style-type: none"> <li>either controller (C), joint controller (JC), processor (P) or sub-processor (SP)</li> <li>• Data Processing Joint Controller Agreement: Must be set to true.</li> <li>• Confidentiality Importance Highlighted: Must be set to true.</li> <li>• Contract Audits Provisions Provided: Must be set to true.</li> <li>• Activity Liability Present: Must be set to true.</li> <li>• Third Party Data Processing Provisioned: Must be set to true.</li> <li>• Contract End Data Management Provided: Must be set to true.</li> <li>• Contract End Data Management Provided: Must be set to true.</li> <li>• Personal Data Processing Details Present: Must be set to true.</li> <li>• Processing Activities Certification Held: Must be set to true.</li> </ul> |
|--|--|---|

|  |   |  |  |
|--|---|--|--|
|  | <ul style="list-style-type: none"><li>• Conflict_Of_Inte rest_Policy_Pres ent</li><li>• Customer_Com plaints_Policy_Pres ent</li><li>• Business_Refer ences_Present</li><li>• Contracted_Part ner_Type_ID (FK)</li><li>• Personal_Data_ Purpose</li><li>• DataProcessing _JointController _Agreement</li><li>• Confidentiality_I mportance_High lighted</li><li>• Contract_Audits _Provisions_Pro vided</li><li>• Activity_Liability _Present</li><li>• Third_Party_Dat a_Processing_P rovisioned</li><li>• Contract_End_ Data_Managem ent_Provided</li><li>• Personal_Data_ Processing_Det ails_Present</li><li>• Processing_Acti vities_Certificati on_Held</li><li>• General_Liabilit y_Insurance_Pres ent_Documen t</li><li>• Cyber_Insuranc e_Present_Doc ument</li><li>• Professional_Ind emnity_Insuranc e_Present_Doc ument</li></ul> |  |  |
|--|---|--|--|

|  |  |  |  |
|--|--|--|--|
|  | <ul style="list-style-type: none"> <li>• Other_Insurance_Required_Document</li> </ul> <p>[ALT]</p> |  |  |
|  | <p><b>Step 6:</b> The user clicks on the “<b>Update“ Button”</b> [ALT]</p>                         | <p><b>Step 7:</b> The system validates whether BEE has been updated if the file has been updated the DeleteVendorFile() function is called</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPDELETE request and it will send a formData object used to remove the file at a specific path location on the API side.</p> <p>the VendorFileAdd() Function in the angular frontend is called next which will capture the bee certificate document provided in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send a formData object used to insert the file at a specific path location on the API side.</p> |  |

|  |  |   |
|--|--|---|
|  |  | <p>Afterwards a part of the filepath to file is send back where the system then calls the UpdateBEEDetails () Function in the angular frontend which will capture all of the related values for BEE provided in the angular form.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPOST request and it will send it the Vendor_BEE Object.</p> <p>The API will then receive the Vendor_BEE object and update the following Details in the <b>Vendor_BEE</b> entity with a SQL Update Query:</p> <ul style="list-style-type: none"> <li>• BEE_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• BEE_Level</li> <li>• BEE_Certificate</li> <li>• Date</li> </ul> <p>If changes are made to the date the scheduled jobs are then removed and replace as follow: the newly created Vendor_BEE object is returned and the</p> |
|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>AddBEEDelayJob() function is called.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request and it will send The Vendor_ID and Date based from the newly created Vendor_BEE object. This will then create a “DelayJob” which will call on a HTTPPOST request. The API will then receive the VendorID, Date and NotifyWhenNumber and add the following Details in the <b>Notification</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Notification_Type_ID (set to 2)</li> <li>• User_ID</li> <li>• Name</li> <li>• Send_Date</li> </ul> <p>The system then calls the UpdateDueDiligence () Function in the angular frontend which will capture all of the values provided in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in</p> |
|--|--|---|

|  |  |   |
|--|--|---|
|  |  | <p>the API by means of an HTTPPUT request and it will send it the Due_Diligence Object with the VendorID.</p> <p>The API will then receive the Due_Diligence object and update the following Details in the <b>DUE_DILIGENCE</b> entity with a SQL Update Query:</p> <ul style="list-style-type: none"> <li>• Due_Diligence_ID (PK)</li> <li>• Vendor_ID (FK)</li> <li>• Due_Diligence_Doc</li> <li>• Mutual_Nda_Signed</li> <li>• Basic_Company_Info_Provided</li> <li>• Group_Structure_Provided</li> <li>• Income_Tax_Number_Provided</li> <li>• Tax_Clearance_Certificate_Provided</li> <li>• Vat_Number_Provided</li> <li>• Vat_Reg_Certificate_Provided</li> <li>• Company_Reg_Doc_Provided</li> <li>• Letter_Of_Good_ Standing_Provided</li> <li>• B_BBEE_Certificate_Provided</li> <li>• Director_Details_Provided</li> </ul> |
|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"><li>• Company_Resolution_Agreement_Provided</li><li>• Subcontractor_Name_Provided</li><li>• Company_Details_Provided</li><li>• Individual_Details_Provided</li><li>• General_Liability_Insurance_Present</li><li>• Cyber_Insurance_Present</li><li>• Professional_Indemnity_Insurance_Present</li><li>• Other_Insurance_Required</li><li>• Licenses_Required</li><li>• Accreditation_Required</li><li>• Professional_Membership_Required</li><li>• Business_Continuity_Present</li><li>• Disaster_Recovery_Plan_Present</li><li>• POPI_Present</li><li>• Data_Security_Breach_Present</li><li>• Site_Visits_Present</li><li>• Information_Security_Policy_Present</li><li>• Privacy_Policy_Present_Present</li><li>• Data_Retention_Destruction_Policy_Present</li><li>• Anti_Bribery_Corruption_Policy_Present</li></ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Ethics_Policy_Present</li> <li>• Conflict_Of_Interest_Policy_Present</li> <li>• Customer_Complaints_Policy_Present</li> <li>• Business_References_Present</li> </ul> <p>The request brings back the newly updated Due_diligence object that is used by system should the user have provided popi details it then calls the UpdatePOPI () Function in the angular frontend which will capture all of the values provided details related to popi in the angular form</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the POPI Object and due_Diligence_ID.</p> <p>The API will then receive the POPI object and update the following Details in the <b>POPI</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• POPI_ID (PK)</li> </ul> |
|--|--|--|---|

|  |  |  |   |
|--|--|--|---|
|  |  |  | <ul style="list-style-type: none"> <li>• Contracted_Partner_Type_ID (FK)</li> <li>• Due_Diligence_ID (FK)</li> <li>• Personal_Data_Purpose</li> <li>• DataProcessing_JointController_Agreement</li> <li>• Confidentiality_Importance_Highlighted</li> <li>• Contract_Audits_Provisions_Provided</li> <li>• Activity_Liability_Present</li> <li>• Third_Party_Data_Processing_Provisioned</li> <li>• Contract_End_Data_Management_Provided</li> <li>• Personal_Data_Processing_Details_Present</li> <li>• Processing_Activities_Certification_Held</li> </ul> <p>The system then calls DeleteVendorFile() function and then the VendorFileAdd() Function in the angular frontend which will capture the types of insurance document provided in the angular form</p> <p>The angular frontend will then call the function which will communicate with the</p> |
|--|--|--|---|

|  |  |  |  |
|--|--|--|--|
|  |  |  | <p>appropriate header in the API by means of an HTTPPOST request and it will send a formData object used to insert the file at a specific path location on the API side.</p> <p>Afterwards a part of the filepath to file is send back where the system then calls the UpdateInsurance () Function in the angular frontend which will capture all of the related values for Insurance details provided in the angular form.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPPUT request and it will send it the Insurance Object.</p> <p>The API will then receive the Vendor_Insurance object and add the following Details in the <b>Vendor_ Insurance</b> entity with a SQL Insert Query:</p> <ul style="list-style-type: none"> <li>• Vendor_ID</li> <li>• Confirmation_Do c</li> </ul> |
|--|--|--|--|

|  |  |  |
|--|--|--|
|  |  | <p>In the case where any of the above except due diligence has been indicated as unchecked would result in system calling a HTTPDELETE request for deleting the details or files related to the unchecked details.</p>   |
|  |  | <p><b>Step 8:</b> The system transforms the Create button into a check mark and hides the cancel button.</p> <p>The system notifies the user by displaying a notification containing the following components:</p> <ul style="list-style-type: none"> <li>• “UPDATE SUCCESSFUL” header</li> <li>• “The Due Diligence has been Update successfully!” body text</li> </ul> <p>The text disappears after 1750 seconds routing the user back to “Vendor Unofficial Vendorlist” screen.</p> |
| <b>Alternate courses:</b>  |  | <p><b>[ALT] Step 4a:</b> User does not want to continue and click the “Cancel” Button. <b>Use Case Terminates.</b></p>   |
| <b>[ALT] Step 6:</b> The user clicks on the “Back” Button. <b>Return to step 4</b> |  |  |

|  |   |
|--|---|
| <b>Conclusion:</b>                                   | The system routes the user to the "Vendor Unofficial Vendorlist" Screen |
| <b>Post-condition:</b>                               | Must be an approved vendor that has a due diligence checklist           |
| <b>Business rules</b>                                | None.   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>• None</li> </ul>                |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>• None</li> </ul>                |

*Table 209: 6.13 Update Approved Onboard Request*

|   |                  |              |
|---|------------------|--------------|
| System Name: Procion System<br>Sub-System: 6 Vendor |                  |              |
| Author: Werner Schutte                              | Date: 2023/07/24 | Version: 1.0 |

|                                    |   |  |
|------------------------------------|---|--|
| <b>Use case name:</b>              | View Approved Or Pending onboard request. | <b>USE CASE TYPE</b>                               |
| <b>Use case id:</b>                | 6.14                                      | Business Requirements: <input type="checkbox"/>    |
| <b>Priority:</b>                   | high                                      | System Analysis: <input type="checkbox"/>          |
| <b>Source:</b>                     | Moyo                                      | System Design: <input checked="" type="checkbox"/> |
| <b>Primary business actor</b>      | User (PBA)                                |  |
| <b>Primary system actor</b>        | None                                      |  |
| <b>Other participating actors:</b> | None                                      |  |

| <b>Other interested stakeholders:</b> | None  |  |                         |
|---------------------------------------|---|--|-------------------------|
| <b>Description:</b>                   | <p>The use case describes the event where the user will want to view pending or approved onboard request. The user will click on the “Approve Vendor” side navbar header link. The system will load the <u>View Approved or Pending Onboard Requests Screen</u>. The system then loads the required details, which allows the user to view, add or update onboard request related details by clicking the appropriate button for the onboard request.</p> |  |                         |
| <b>Pre-condition:</b>                 | The user has to be logged in to the system.   |  |                         |
| <b>Trigger:</b>                       | The user request to view approved or pending onboard request  |  |                         |
| <b>Typical course<br/>Of events:</b>  | <b>Actor Action</b>   | <b>System Response</b>   |                         |
|                                       |   | <b>Manual Action</b>   | <b>Automated Action</b> |
|                                       |   | <p><b>Step 1:</b> The system makes use of the angular frontend to display the “View Approved or Pending Onboard Requests” page with the following Items:</p> <ul style="list-style-type: none"> <li>• Header: “Manage Approved or Pending Onboard Requests”</li> <li>• Mat Select with the following options: “All”, “Approved Vendor”, “Pending Vendor”, “Management Approval”</li> <li>• Search input with following label “Search”</li> </ul> |                         |

|  |   |   |
|--|---|---|
|  |   | <p>Mat table containing the following details:</p> <ul style="list-style-type: none"> <li>• Onboard Request Number</li> <li>• Selected Option</li> <li>• Request Type</li> <li>• Total Quotes</li> <li>• Created By</li> <li>• Button: Update, View, add</li> </ul>   |
|  |   | <p><b>Step 2:</b> The system will then populate the table by calling the function <b>GetAllOnboardRequest ()</b>.</p> <p>The angular frontend will then call the function which will communicate with the appropriate header in the API by means of an HTTPGET request.</p> <p>The API will then retrieve a Onboard_Request object following Details in the <b>Onboard_Request</b> entity with a SQL read Query.</p> <p>This object will then be filter to only those that has not been rejected and based on the mat select filter option.</p> |
|  | <p><b>Step 3:</b> The user clicks on the “View” button</p> <p>[ALT]</p> | <p><b>Step 4:</b> The system responds by routing the user to the “Approve Vendor” Screen</p>  |

|  |  |
|--|--|
| <b>Alternate courses:</b>                            | <p><b>[ALT] Step 3a:</b> The user clicks on the “Update” Button. The user will be redirected to the “<u>Vendor Approval Edit Details</u>” screen. <b>Use Case Terminates</b></p> <p><b>[ALT] Step 3b:</b> The user clicks on the “Add” Button. The user will be redirected to the “<u>Vendor Approve add details</u>” screen. <b>Use Case Terminates</b></p> |
| <b>Conclusion:</b>                                   | The system routes the user to the correct screen   |
| <b>Post-condition:</b>                               | Table with the required details is display and the user clicks on one of the buttons.  |
| <b>Business rules</b>                                | None   |
| <b>Implementation constraints and specifications</b> | <ul style="list-style-type: none"> <li>● None</li> </ul>   |
| <b>Assumptions:</b>                                  | <ul style="list-style-type: none"> <li>● None</li> </ul>   |
| <b>Open issues:</b>                                  | <ul style="list-style-type: none"> <li>● None</li> </ul>   |

Table 199: View Approved Or Pending Onboard Request

### 5.3 CONCLUSION

This concludes the Logical Use case Narratives section of the Procion system.

## 6. TECHNICAL CONTEXT DIAGRAM

### 6.1 INTRODUCTION

In this section we will display the visual representation of the Technical Context diagrams for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system.

### 6.2 TECHNICAL CONTEXT DIAGRAM

#### USE CASE 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36

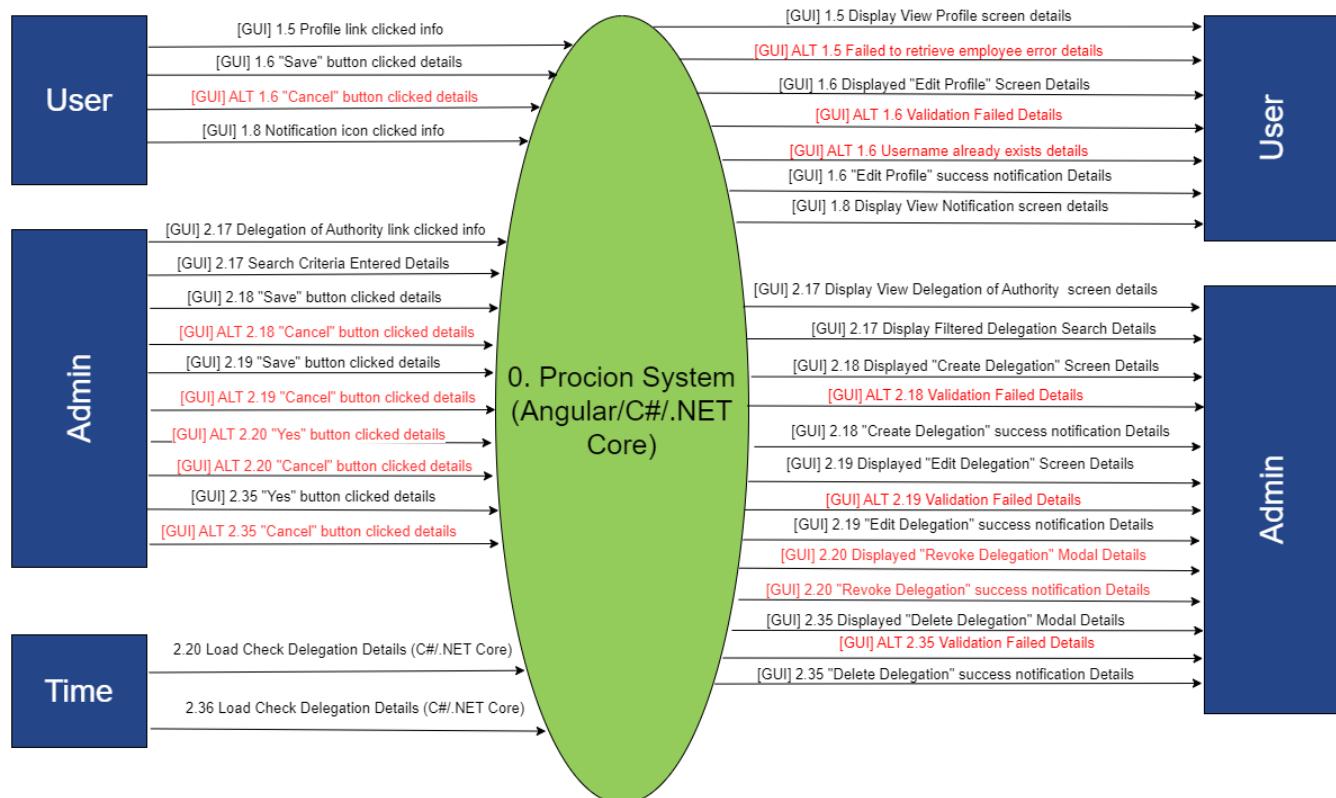


Figure 146 Use Case 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36 Tech Context

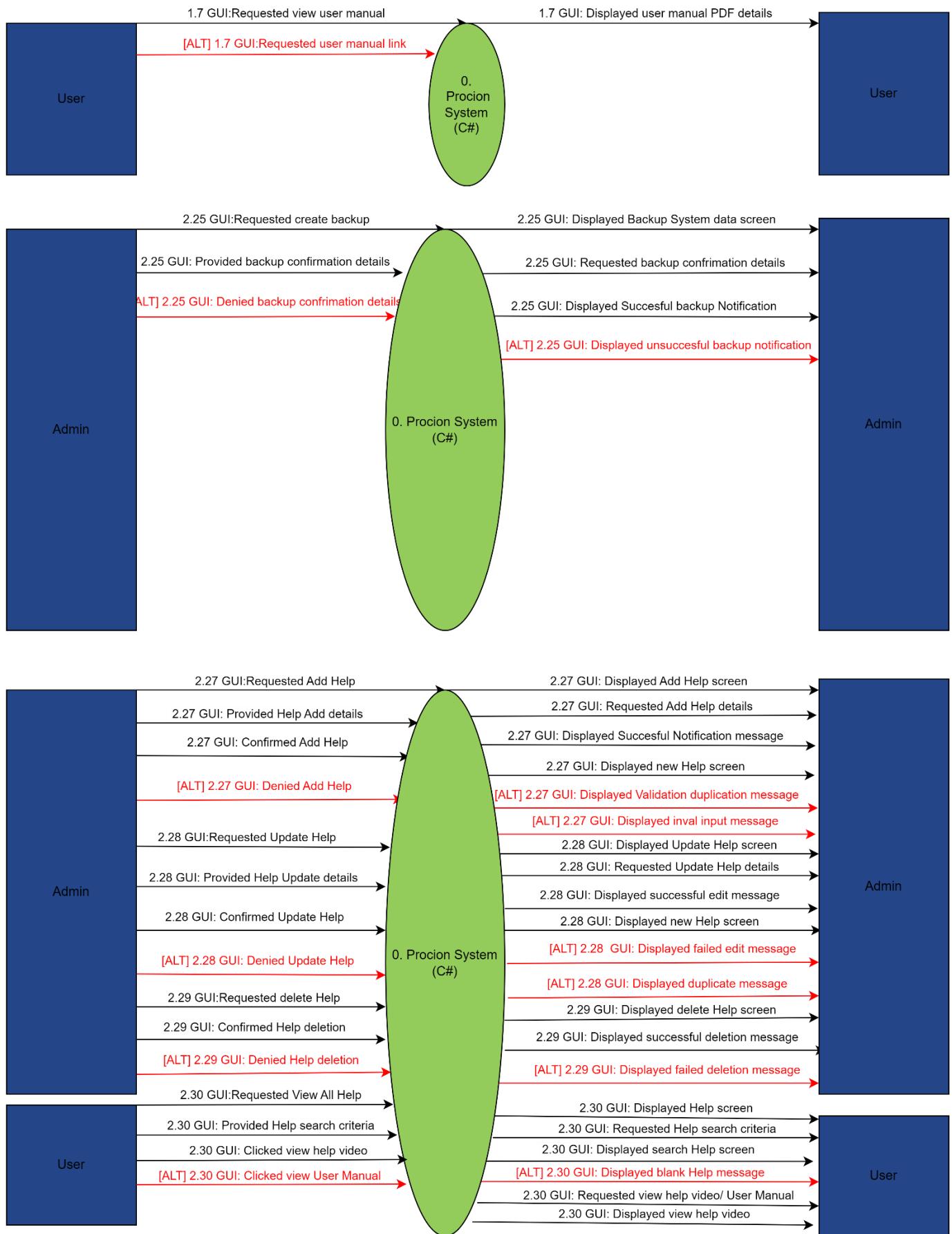
**USE CASE 1.7 & 2.25 & 2.27-2.30**


Figure 147 Use Case 1.7-1.8 &amp; 2.25-2.26 &amp; 2.27-2.30 Tech Context

### USE CASE 3.8-4.1 & 4.3-4.10

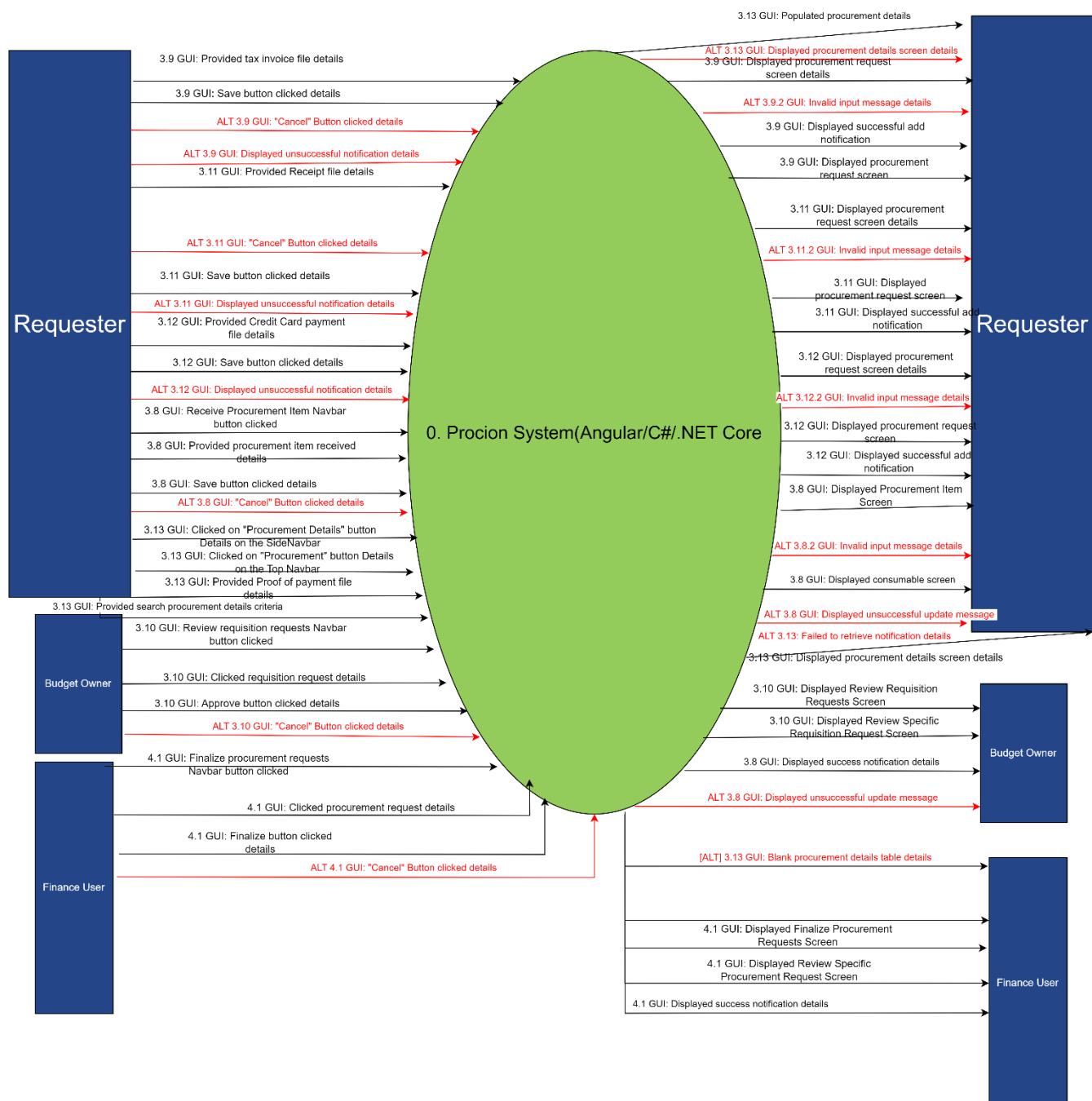


Figure 148 3.8-4.1 Tech Context

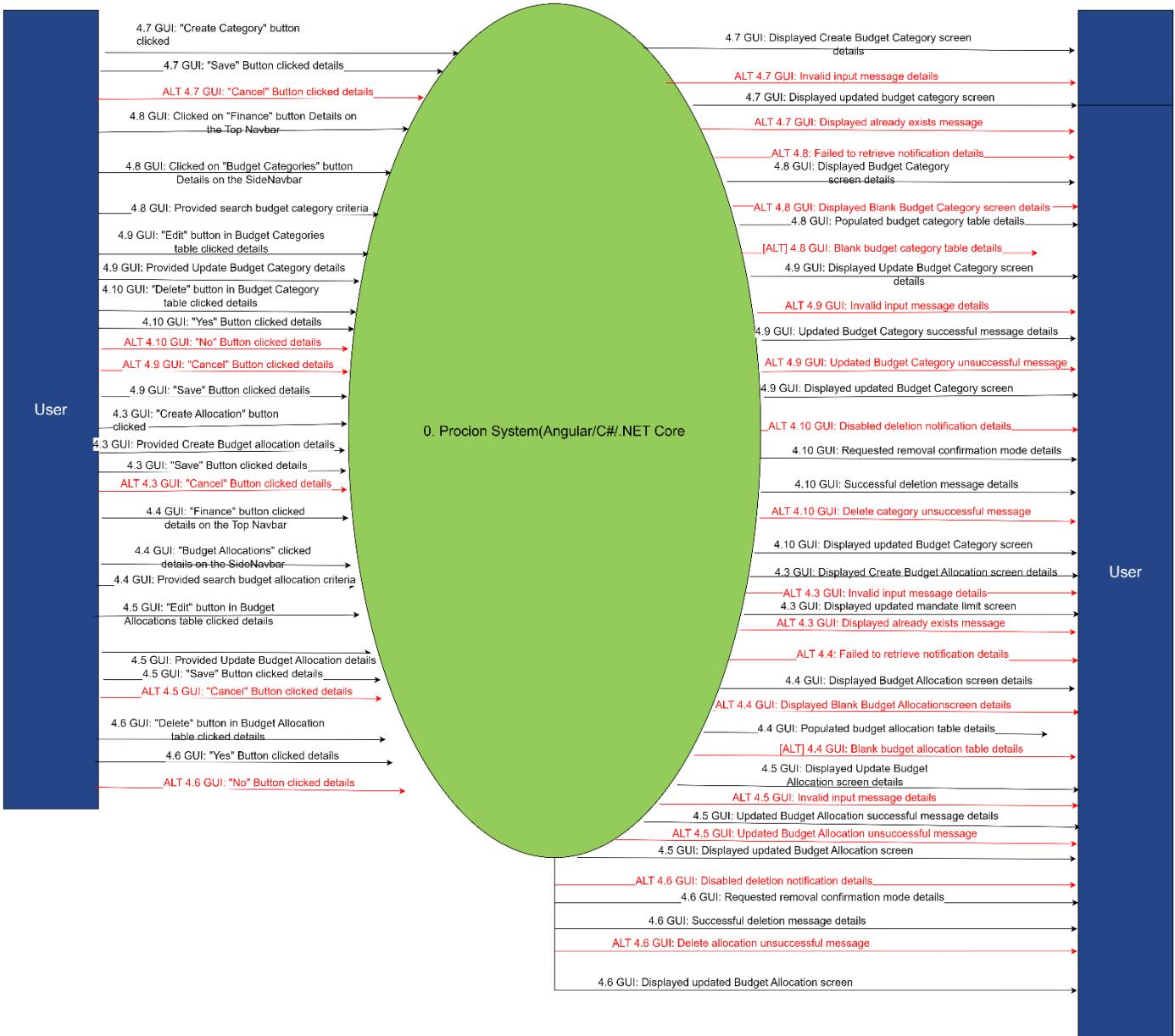
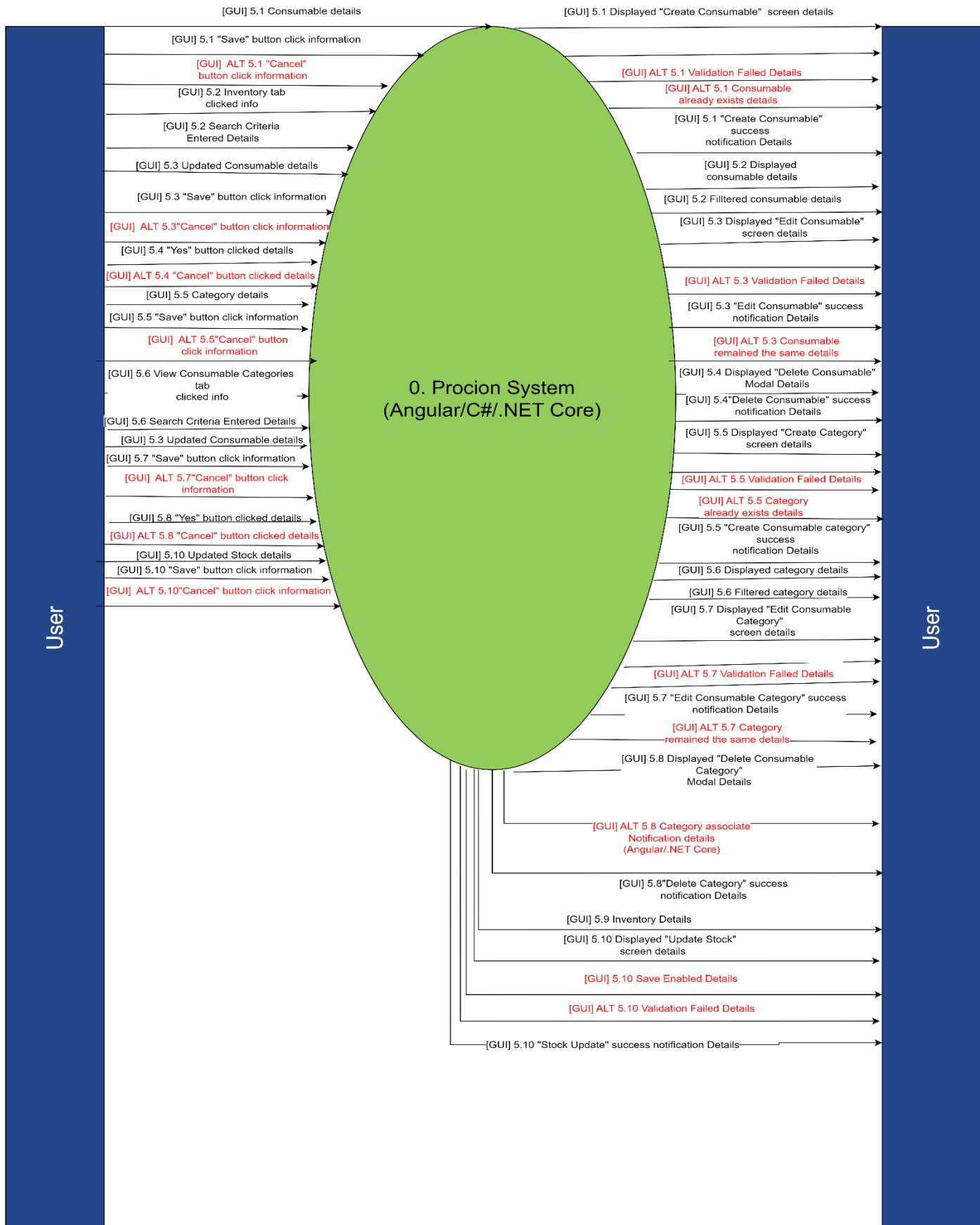


Figure 149 4.3-4.10 Tech Context

**USE CASE 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4**

**Figure 150 Use Case 5.1 - 5.8 & 5.9, 5.10 Tech Context**

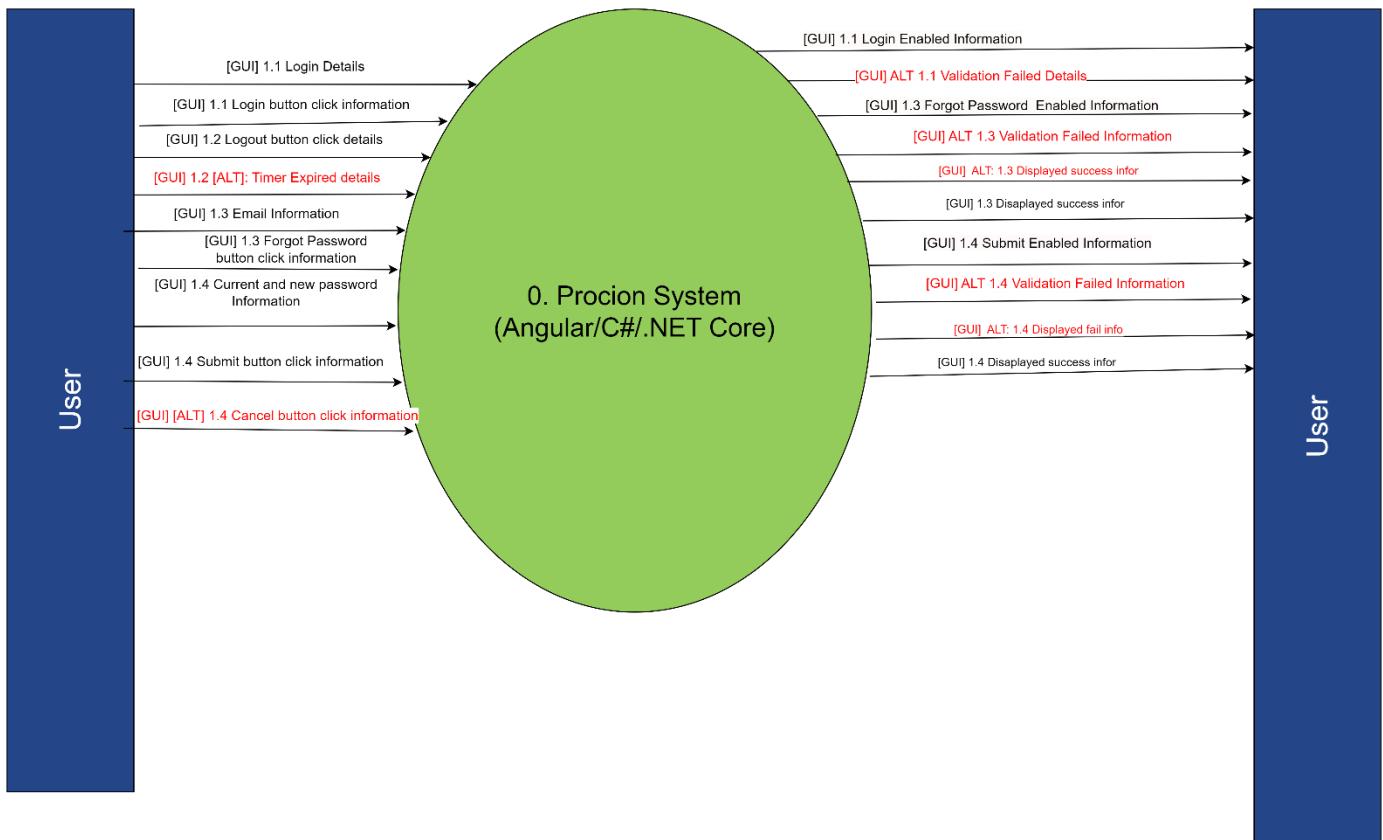


Figure 151 Use Case 1.1-1.4 Tech Context

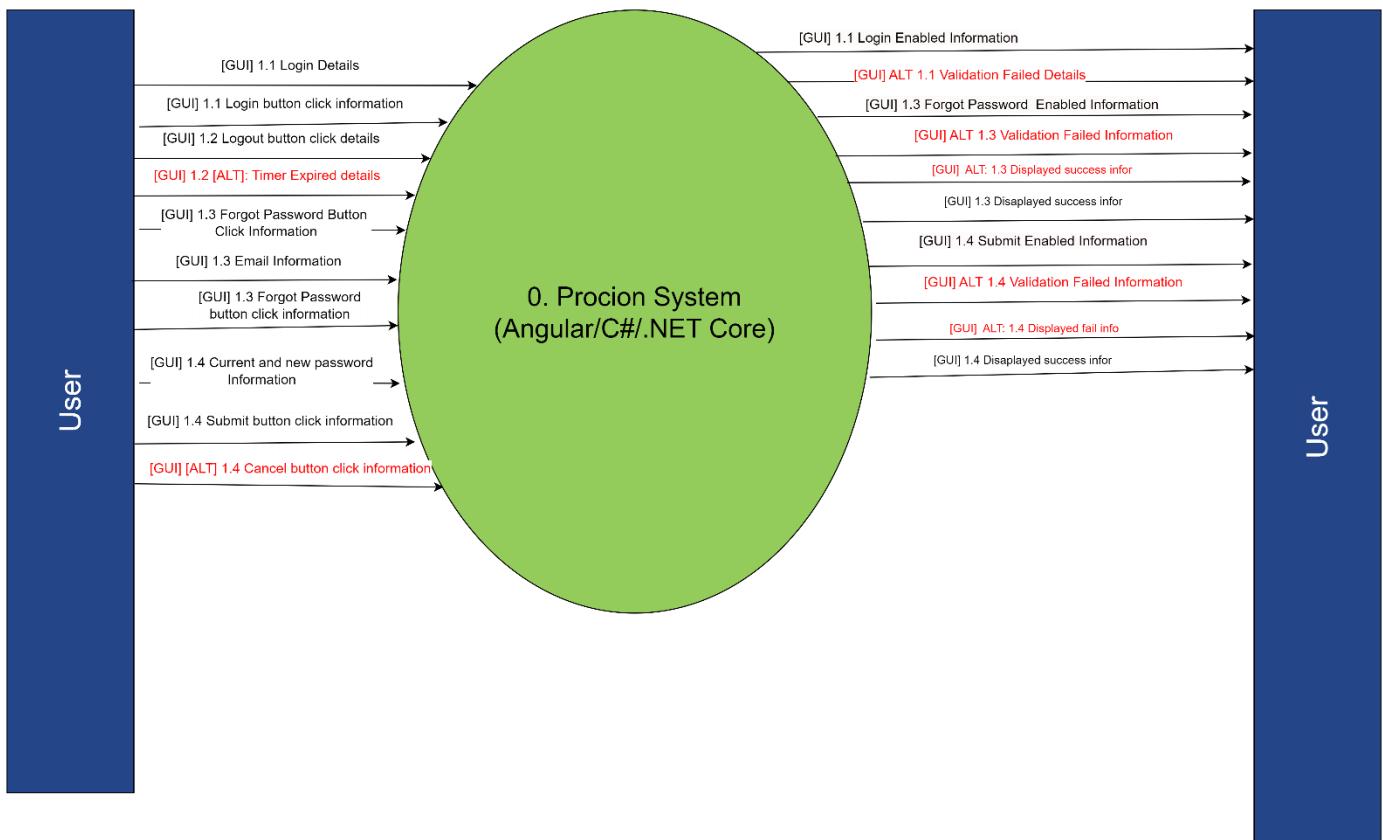


Figure 152 Use Case 3.1-3.4 Tech Context

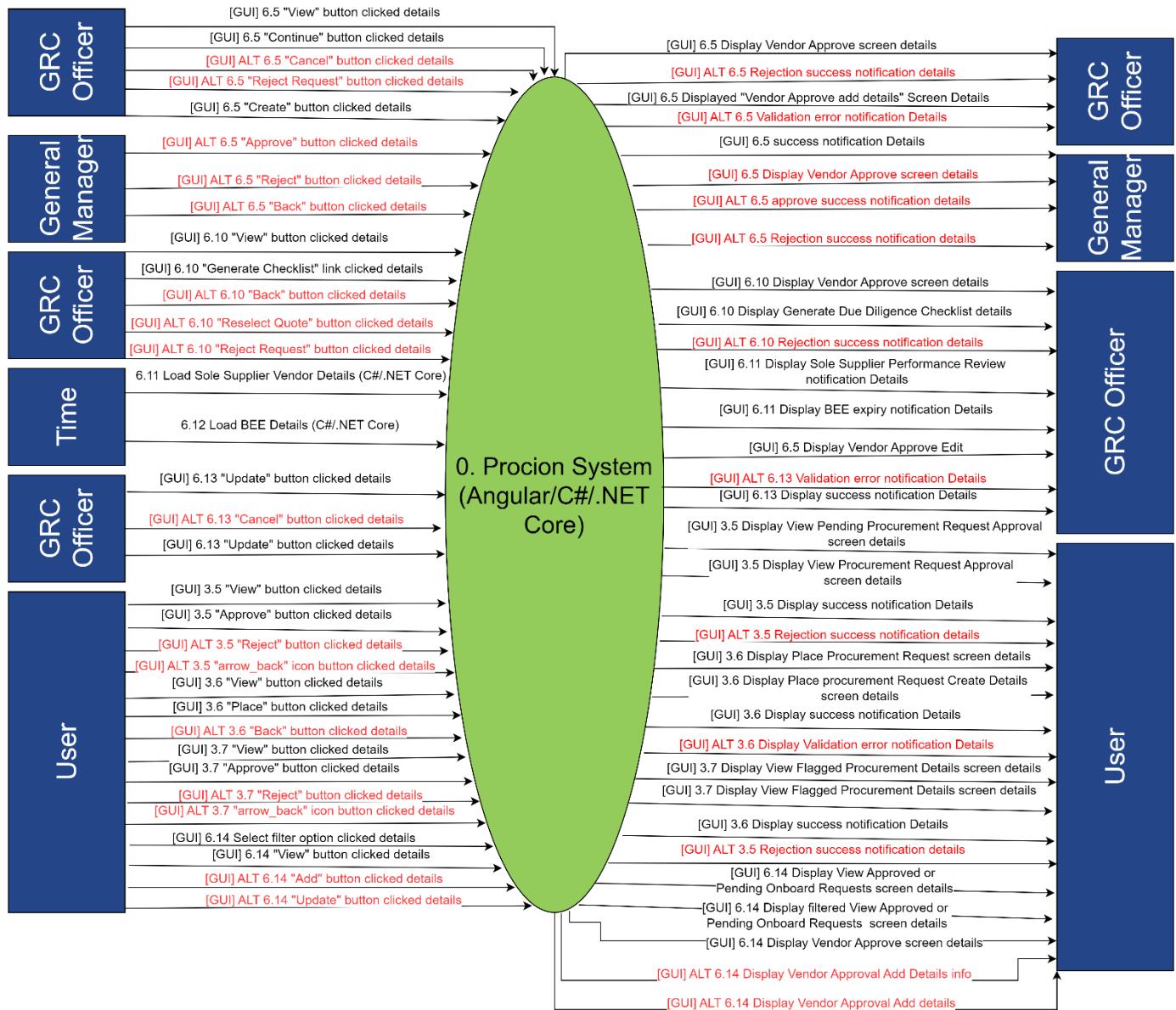
**USE CASE 3.5-3.7 & 6.5 & 6.10-6.14**


Table 210: Use case 3.5-3.7, 6.5, 6.10-6.14 Technical Context

### 6.3 CONCLUSION

This Concludes the visual representation of the Technical Context diagrams for the Procion system.

## 7. TECHNICAL PRIMITIVE LEVEL DIAGRAM

### 7.1 INTRODUCTION

In this section we will display the visual representation of the Technical Primitive level diagrams for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) respectively, of the Procion system.

### 7.2 TECHNICAL PRIMITIVE LEVEL DIAGRAM

#### USE CASE 1.5, 1.6, 1.8 & 2.17-2.20 & 2.35-2.36

##### 1.5 VIEW PROFILE

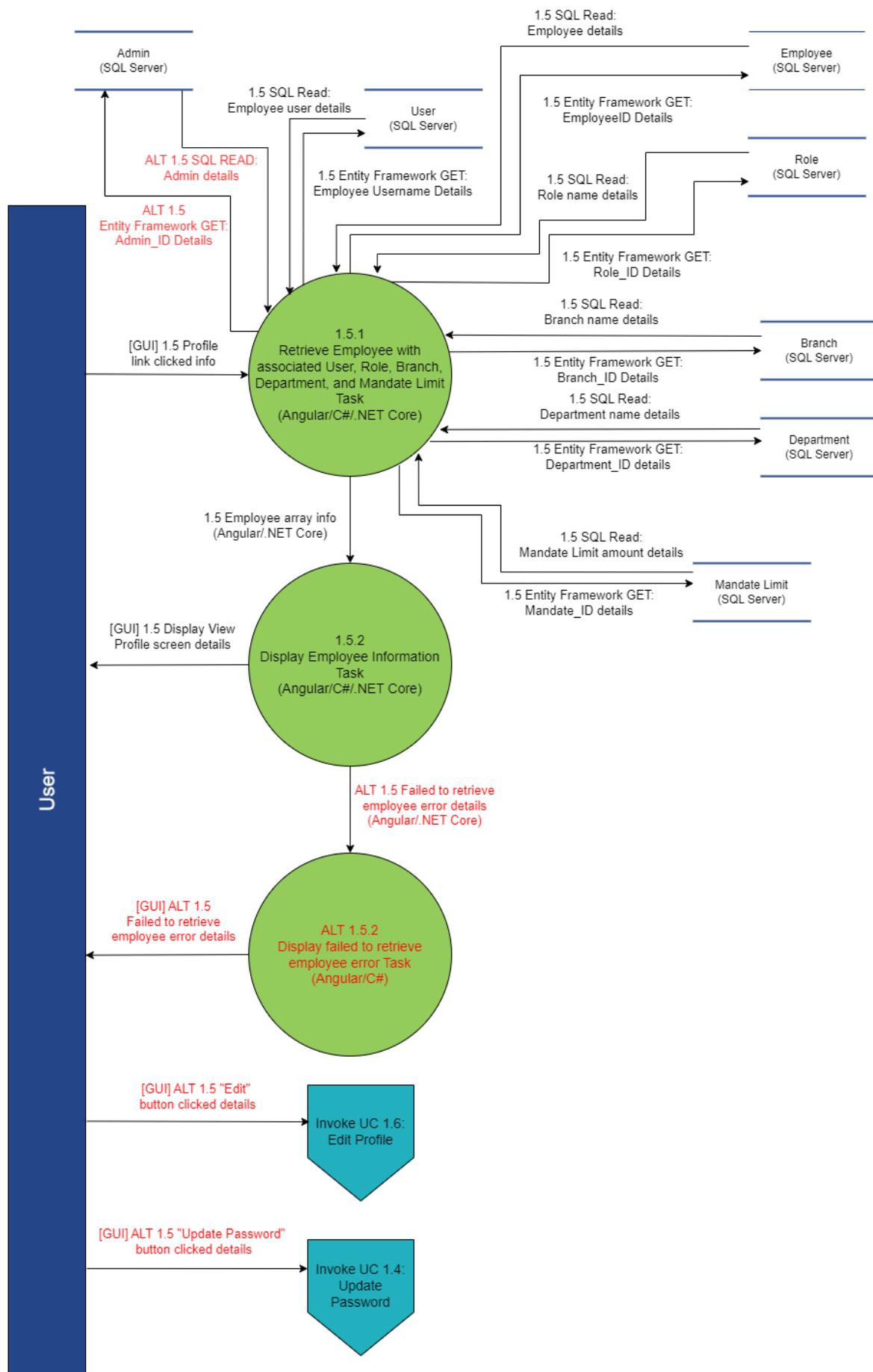


Figure 153 1.5 View Profile Tech Prim

### 1.6 EDIT PROFILE

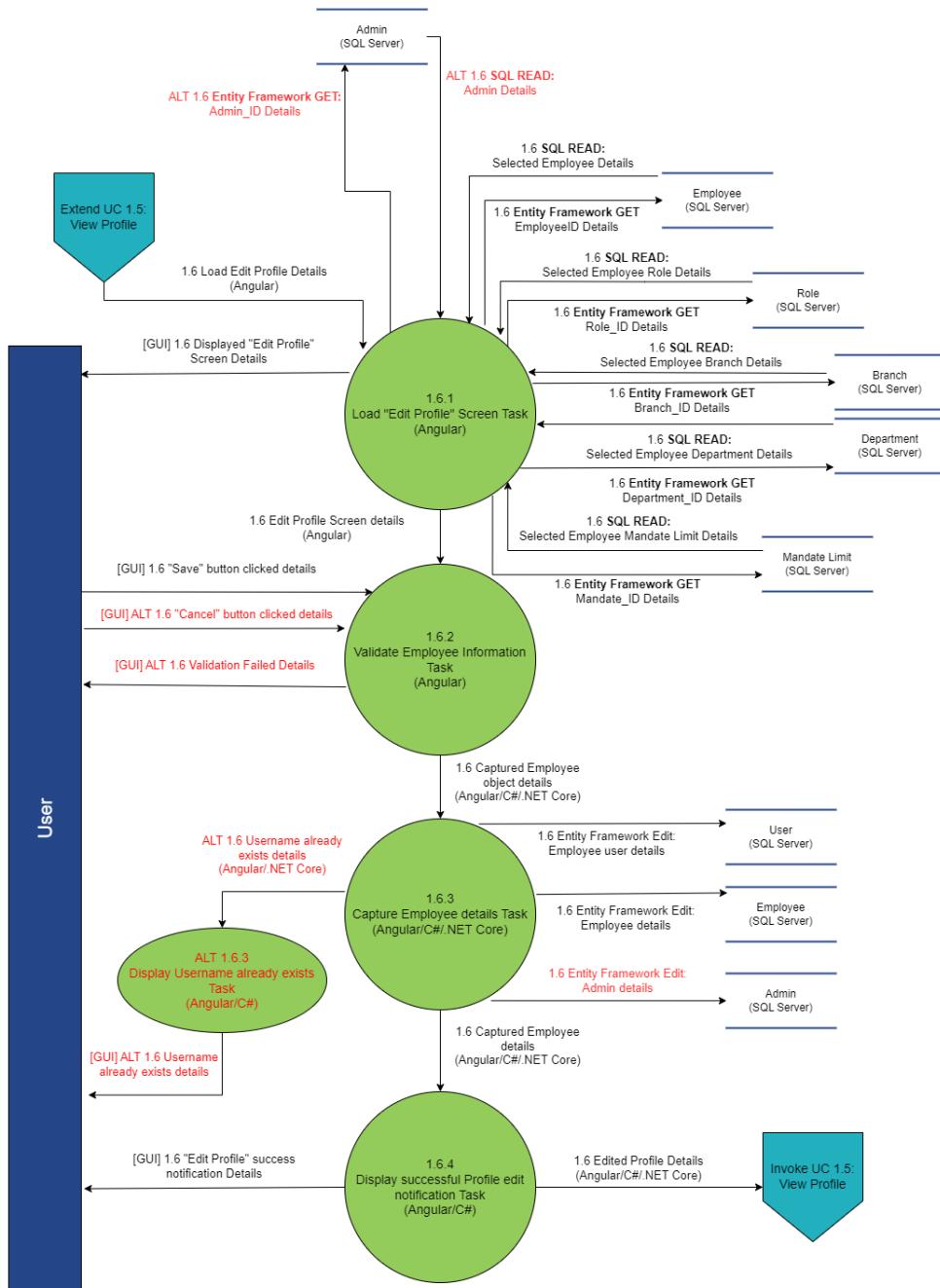


Figure 154 1.6 Edit Profile Tech Prim

## 1.8 VIEW NOTIFICATION

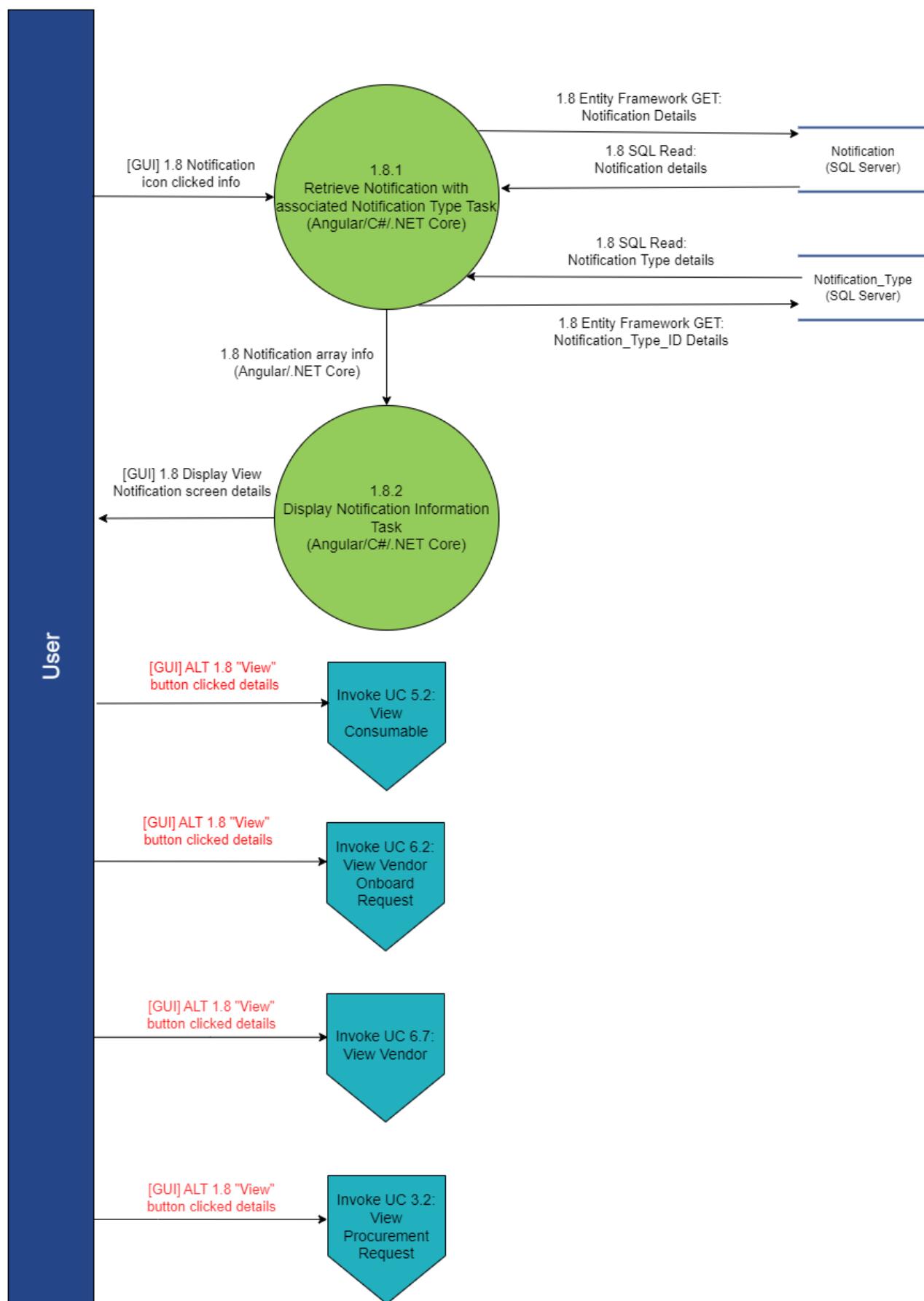


Figure 155 1.8 View Notifications Tech Prim

## 2.17 VIEW DELEGATION OF AUTHORITY

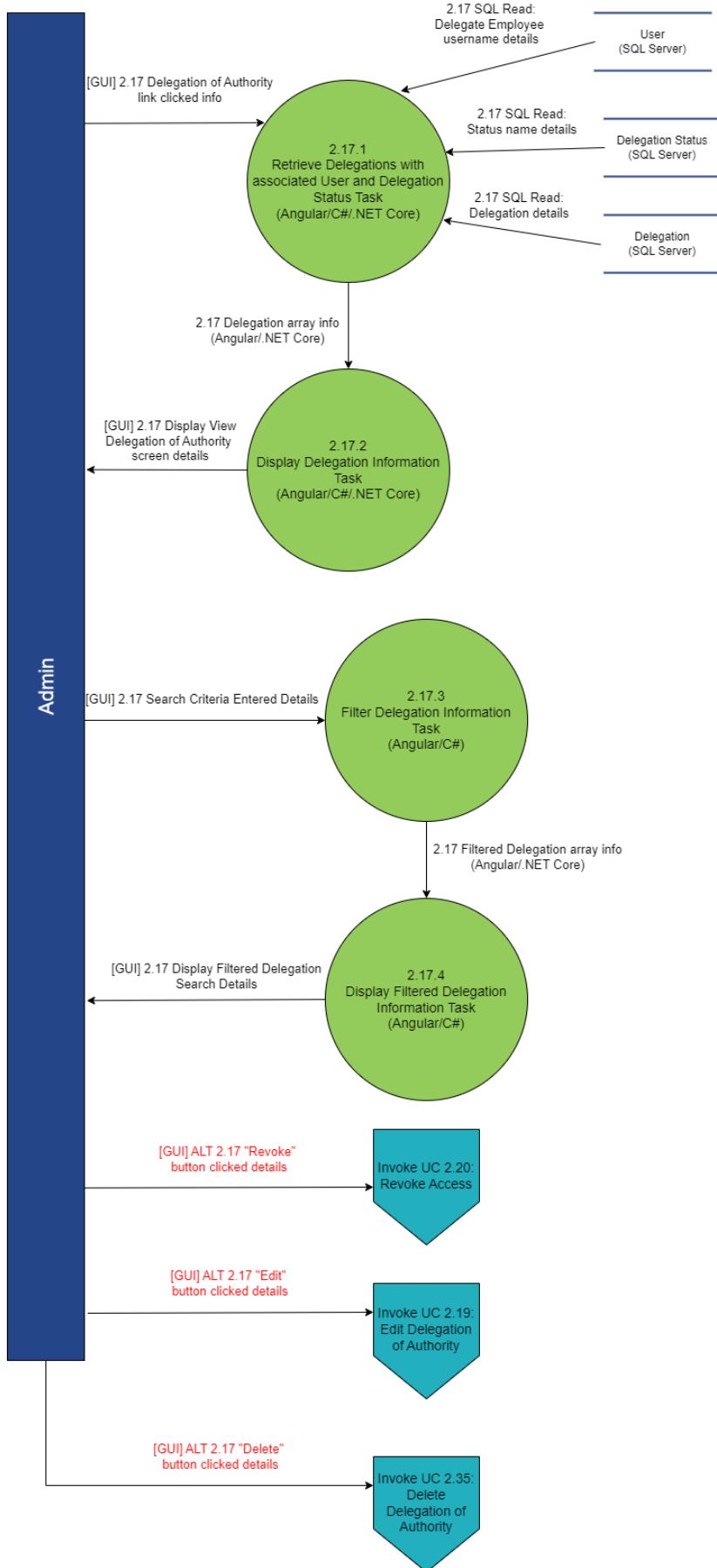
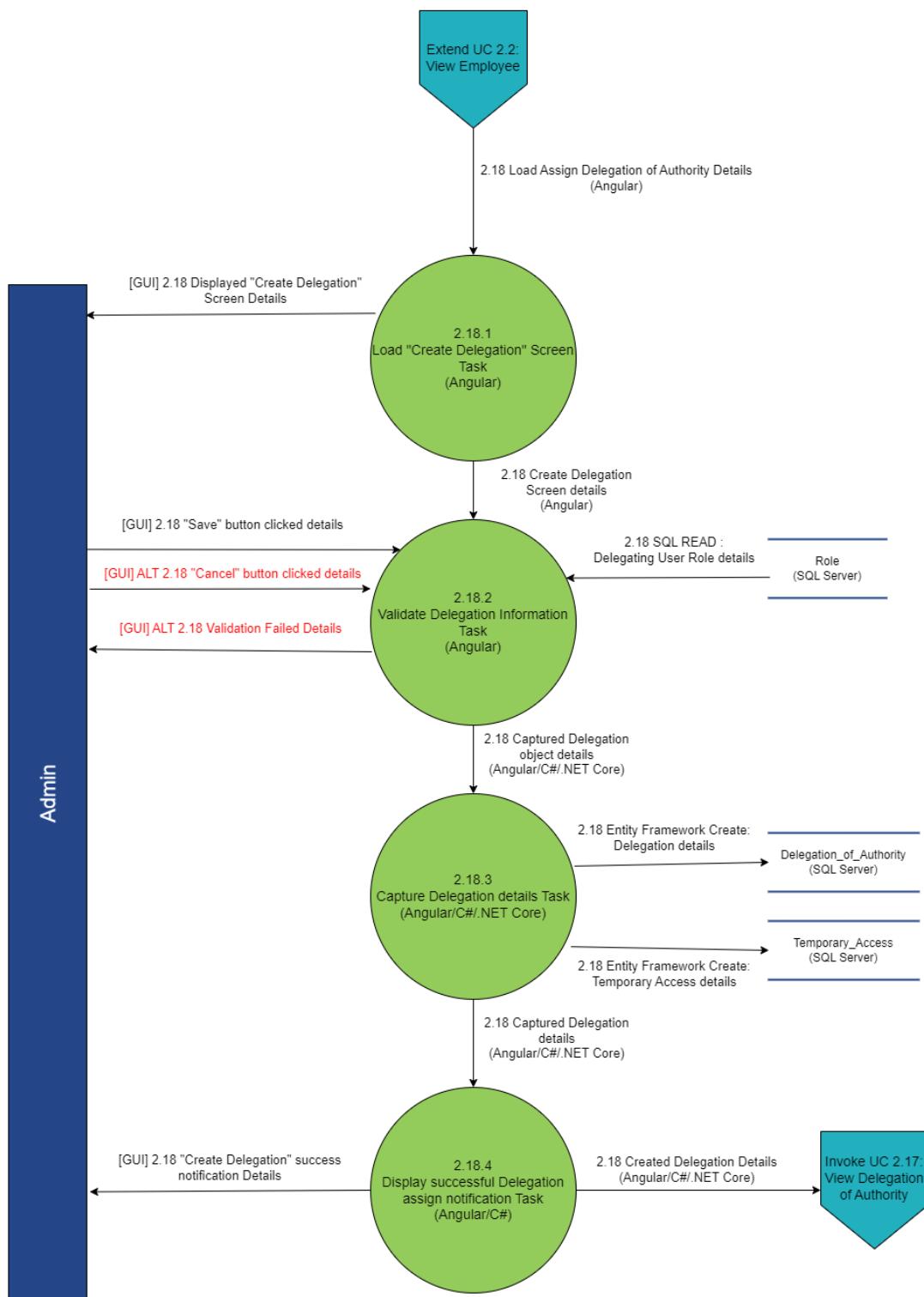


Figure 156 2.17 View Delegation of Authority Tech Prim

**2.18 ASSIGN DELEGATION OF AUTHORITY**

*Figure 157 2.18 Assign Delegation of Authority Tech Prim*

## 2.19 EDIT DELEGATION OF AUTHORITY

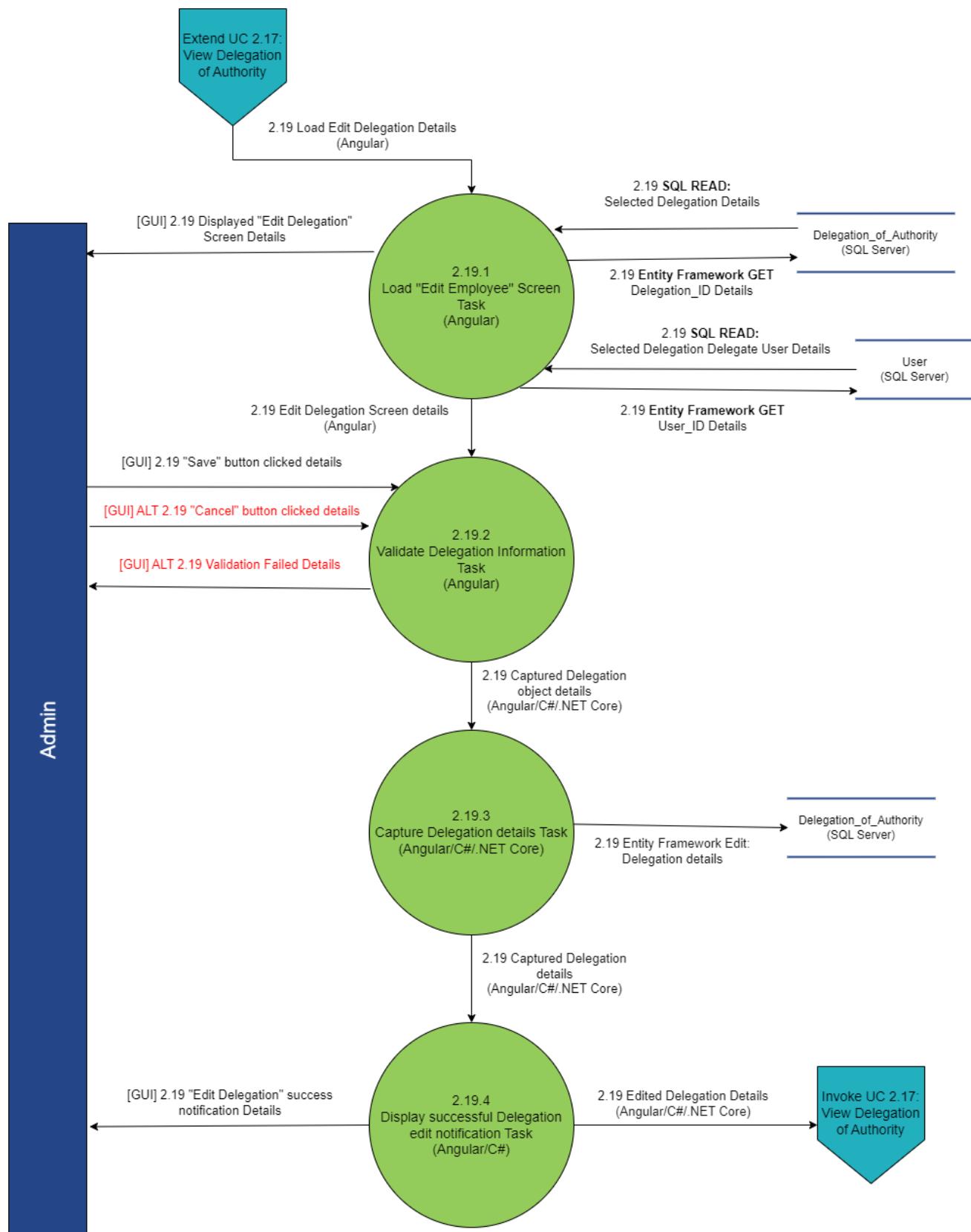
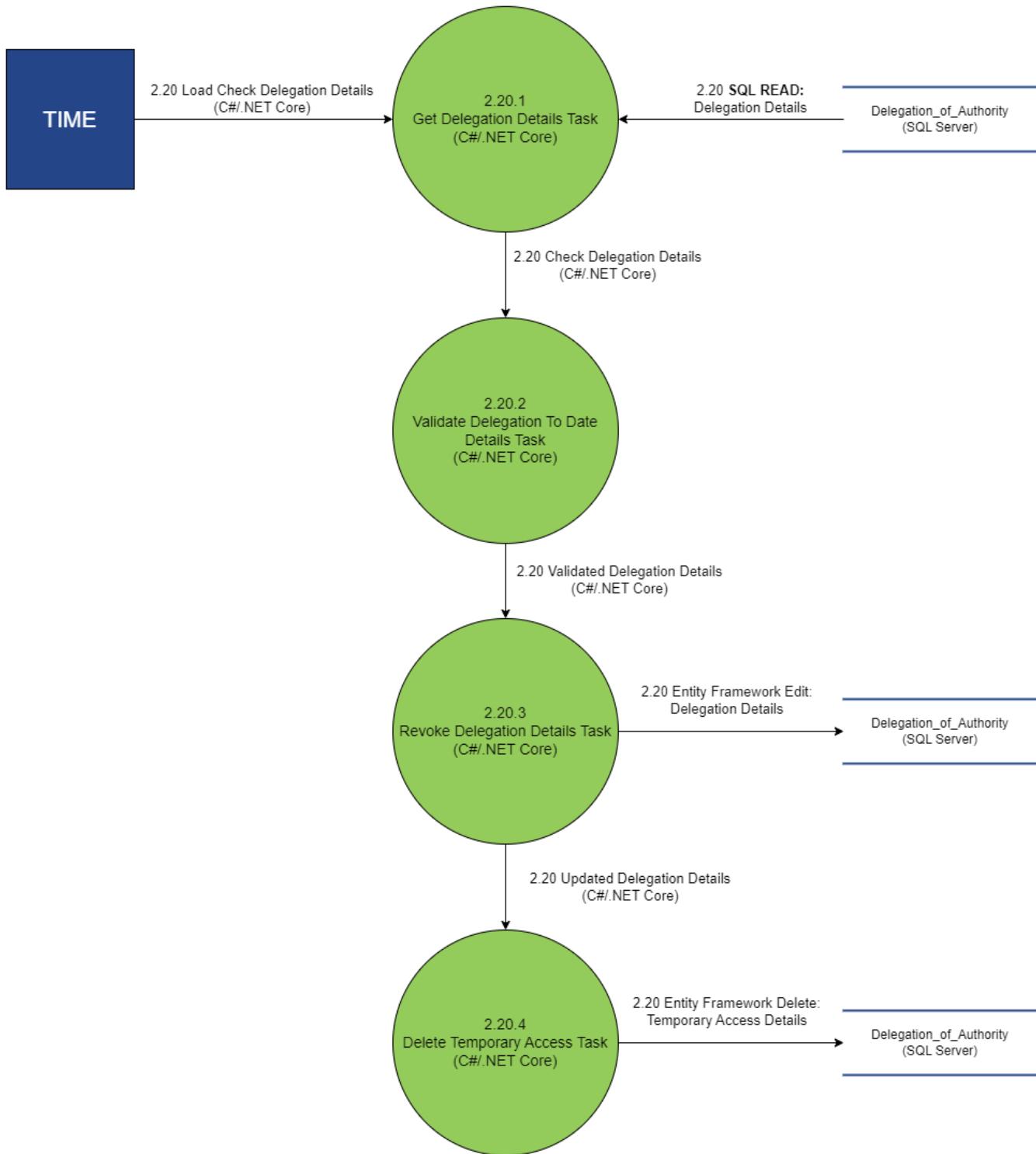
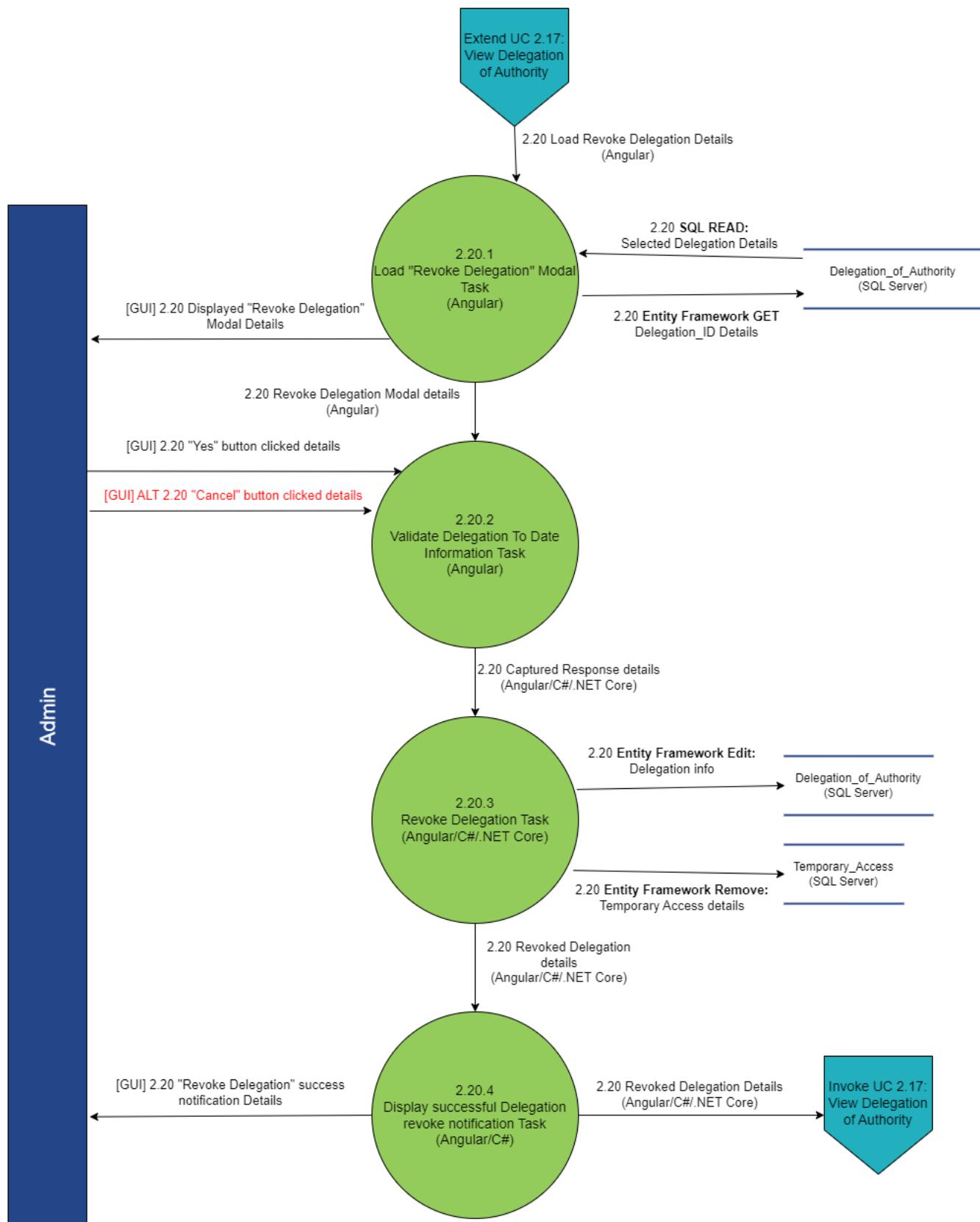


Figure 158 2.19 Edit Delegation of Authority Tech Prim

**2.20 REVOKE ACCESS**

*Figure 159 2.20 Revoke Access Tech Prim*

**2.20 REVOKE ACCESS ALT**

**Figure 160 2.20 Revoke Access Tech Prim**

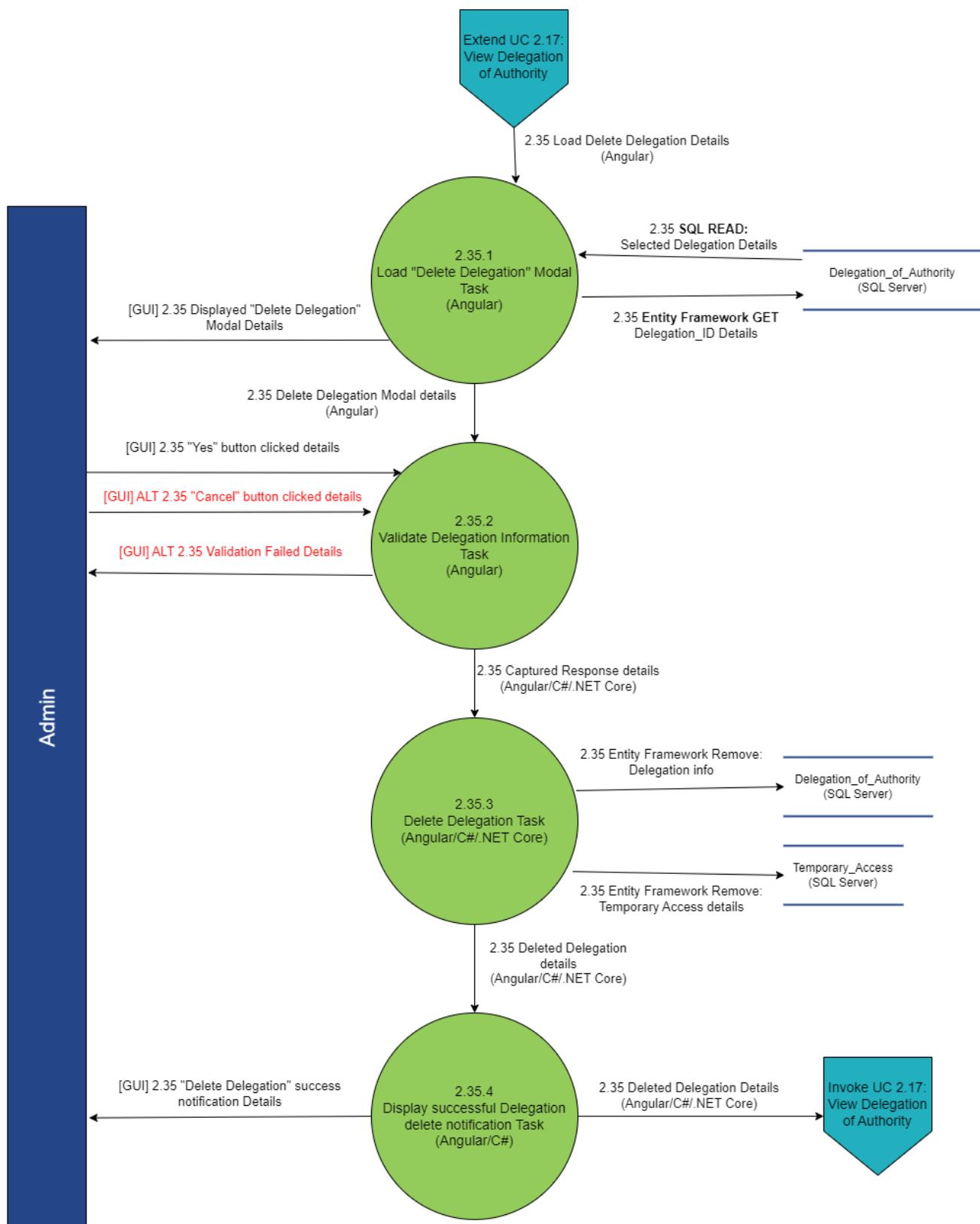
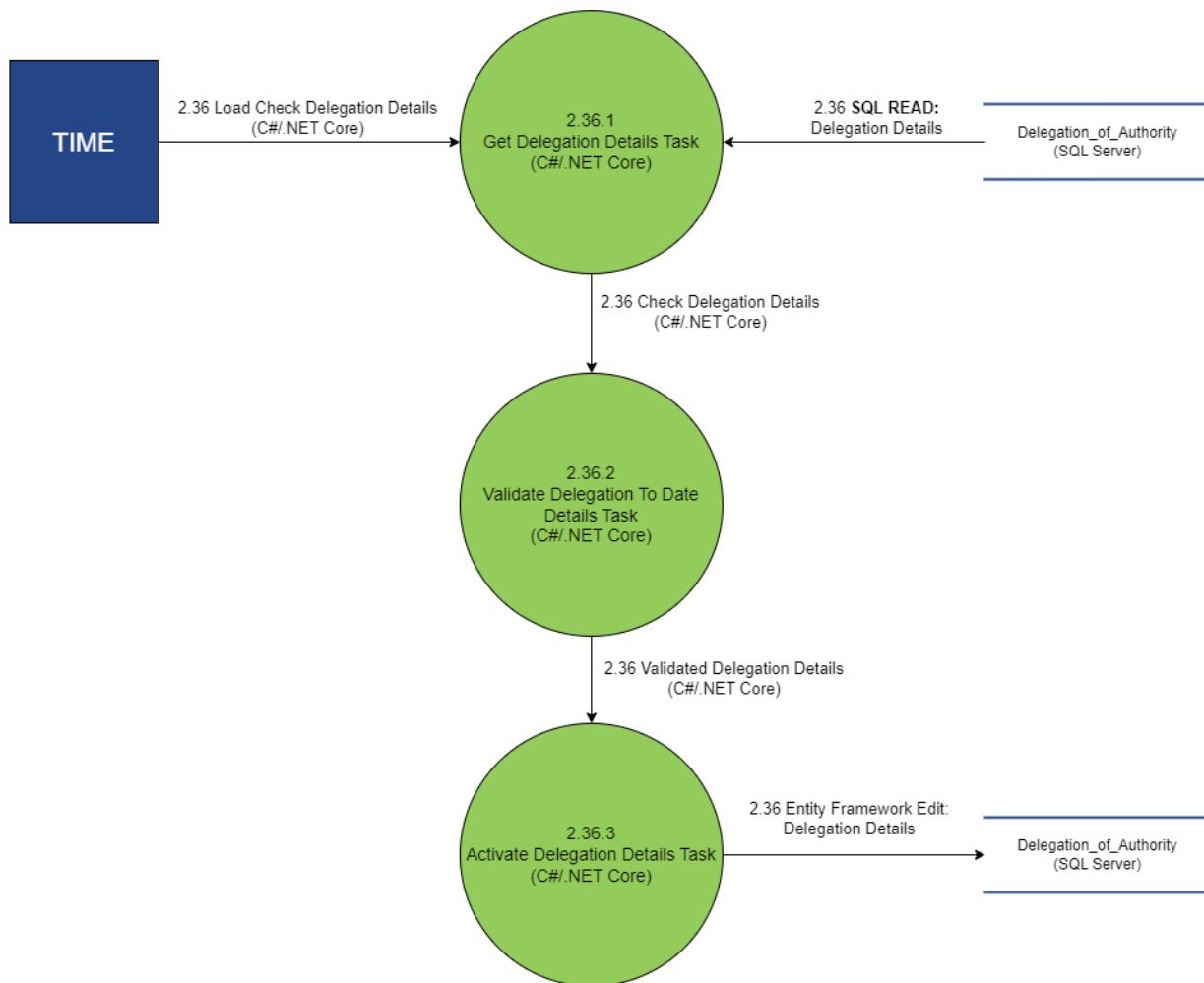
**2.35 DELETE DELEGATION OF AUTHORITY**


Figure 161 2.35 Delete Delegation of Authority Tech Prim

**2.36 ACTIVATE DELEGATION OF AUTHORITY**

*Figure 162 2.36 Activate Delegation of Authority Tech Prim*

### USE CASE 1.7 & 2.25 & 2.27-2.30

#### 1.7 VIEW USER MANUAL

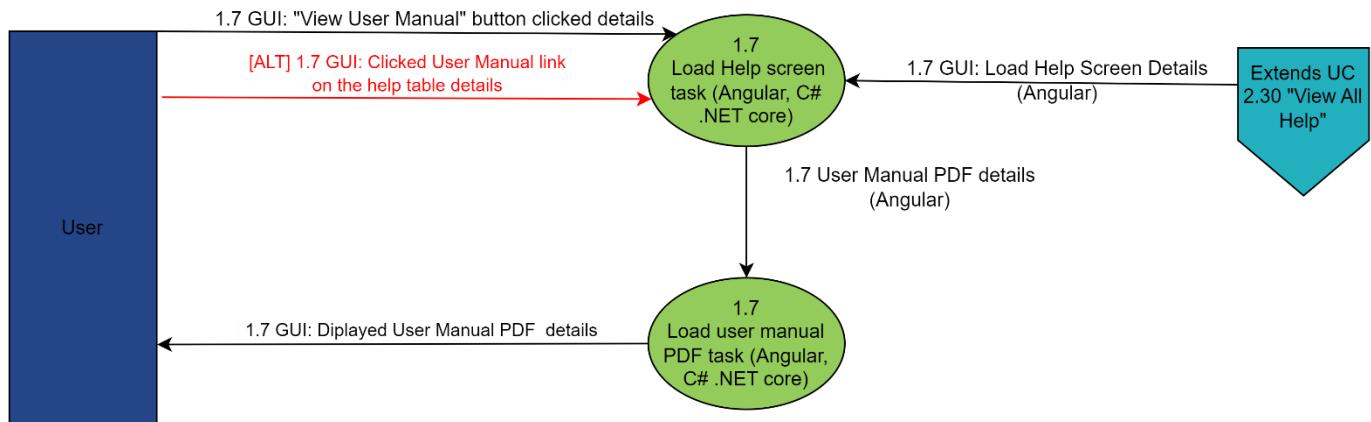


Figure 163 1.7 View User Manual Tech Prim

#### 2.25 BACKUP SYSTEM DATA

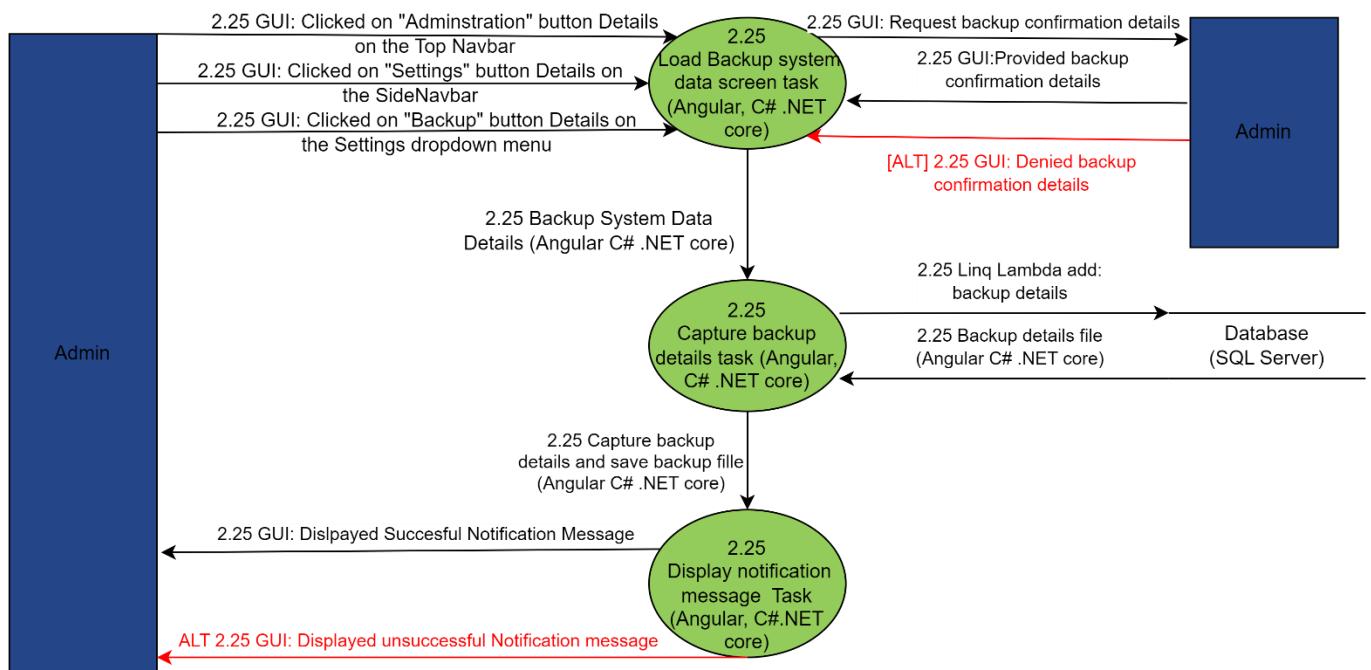
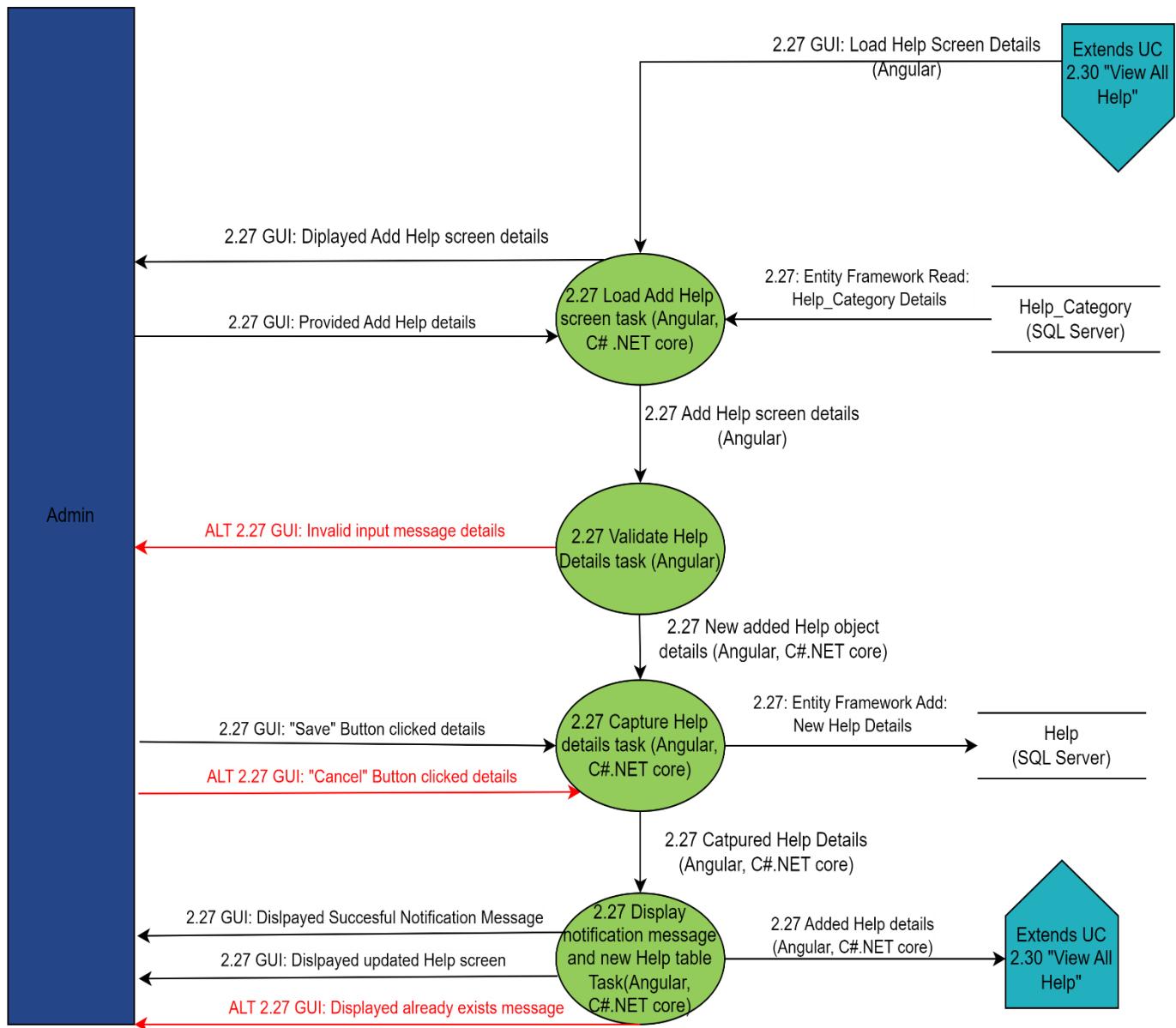
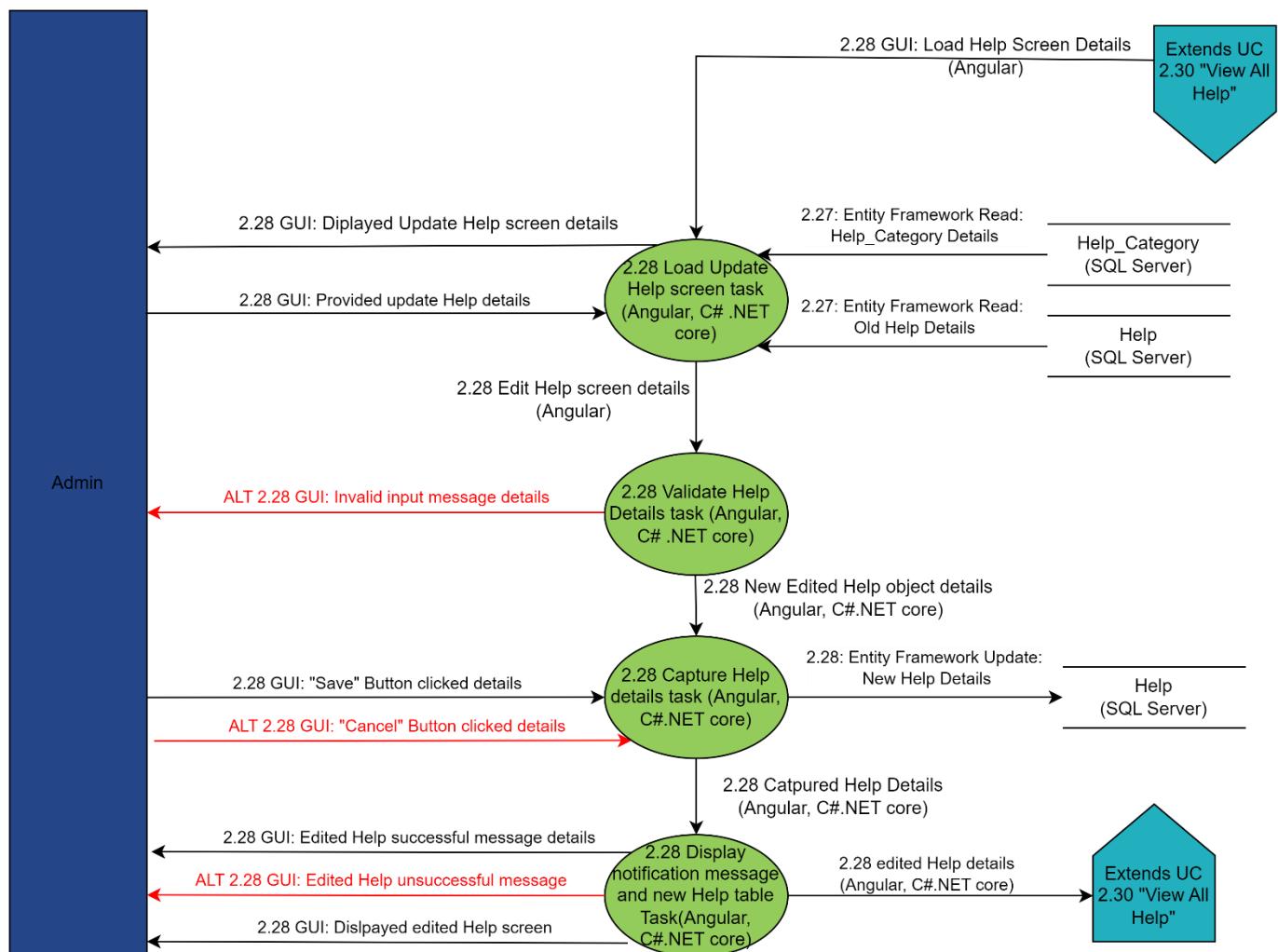
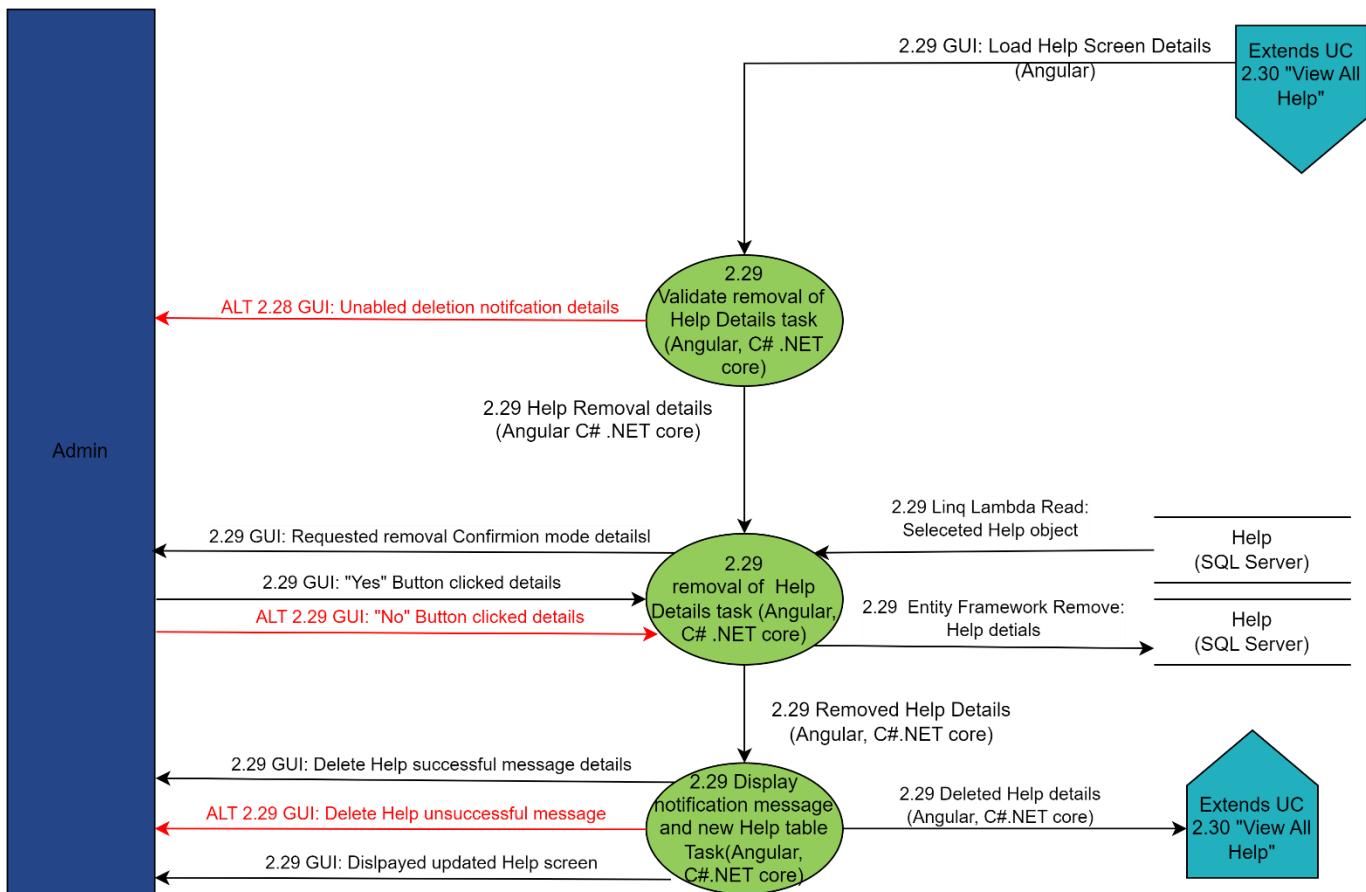


Figure 164 2.25 Backup System Data Tech Prim

**2.27 ADD HELP**

*Figure 165 2.27 Add Help Tech Prim*

**2.28 UPDATE HELP**

*Figure 166 2.28 Update Help Tech Prim*

**2.29 DELETE HELP**

*Figure 167 2.29 Delete Help Tech Prim*

## 2.30 VIEW ALL HELP

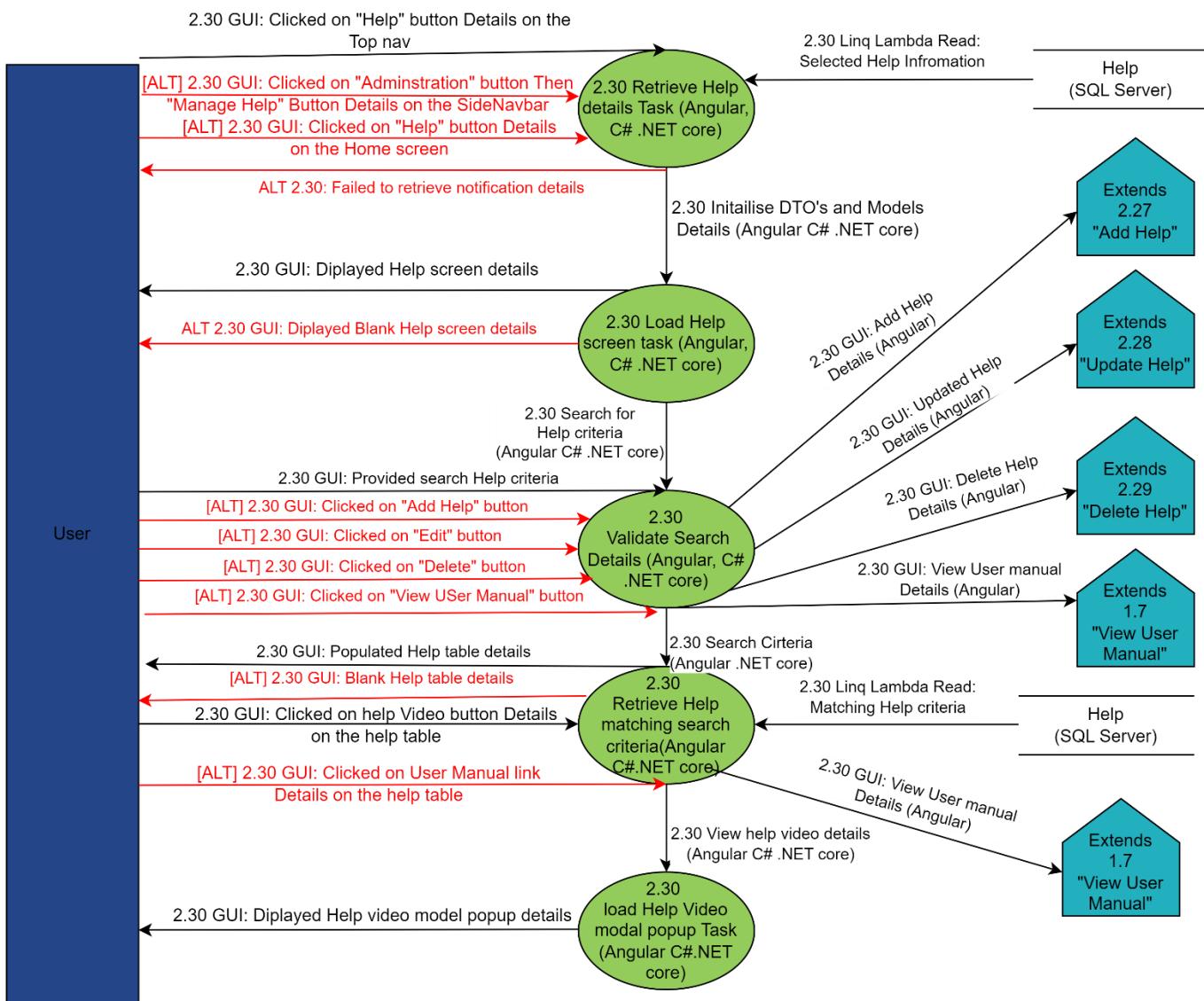


Figure 168 2.30 View All Help Tech Prim

## USE CASE 3.8-3.11 & 4.1-4.6

### 3.8 RECEIVE PROCUREMENT ITEM

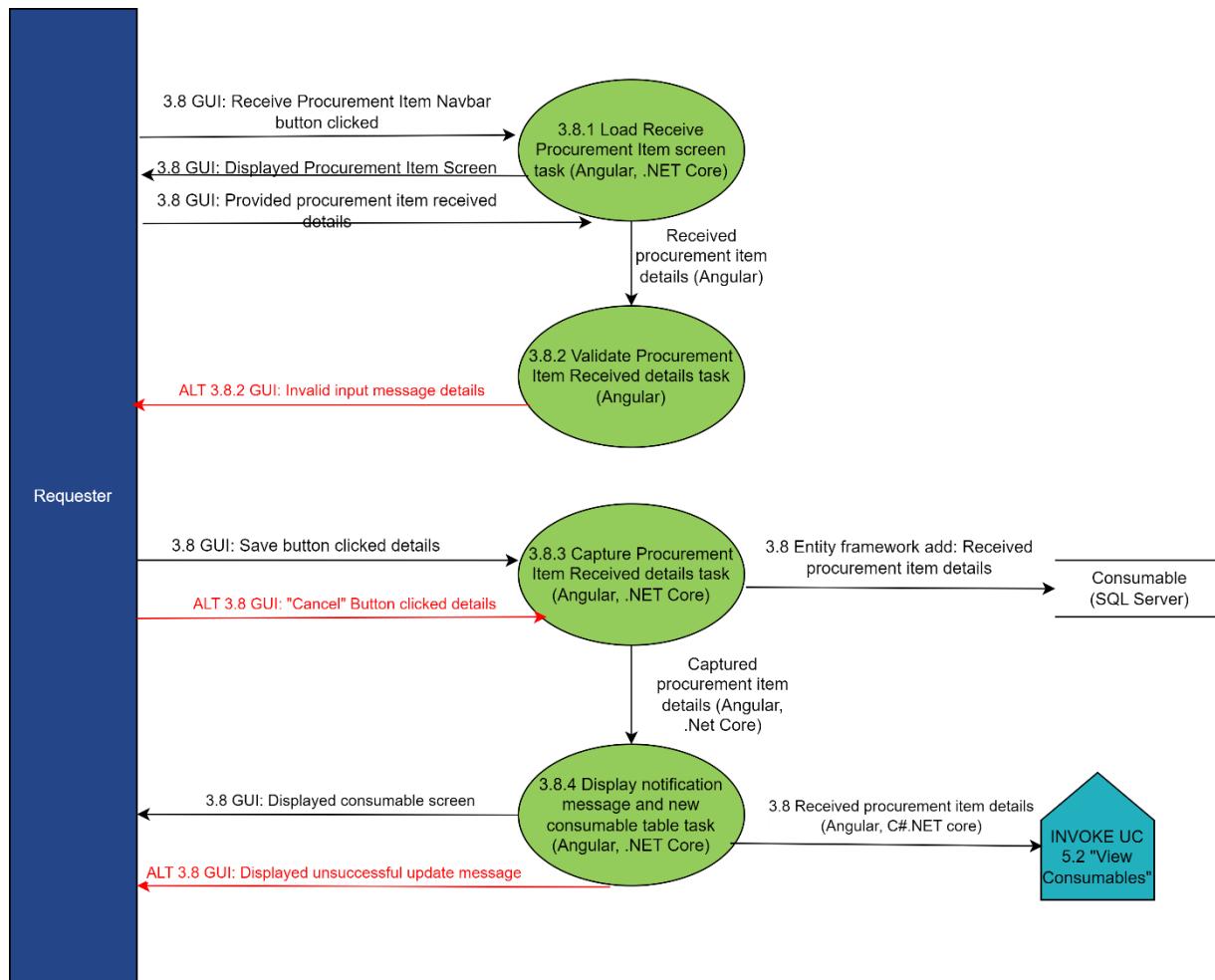


Figure 169 3.8 Receive Procurement Item Tech Prim

### 3.9 UPLOAD TAX INVOICE

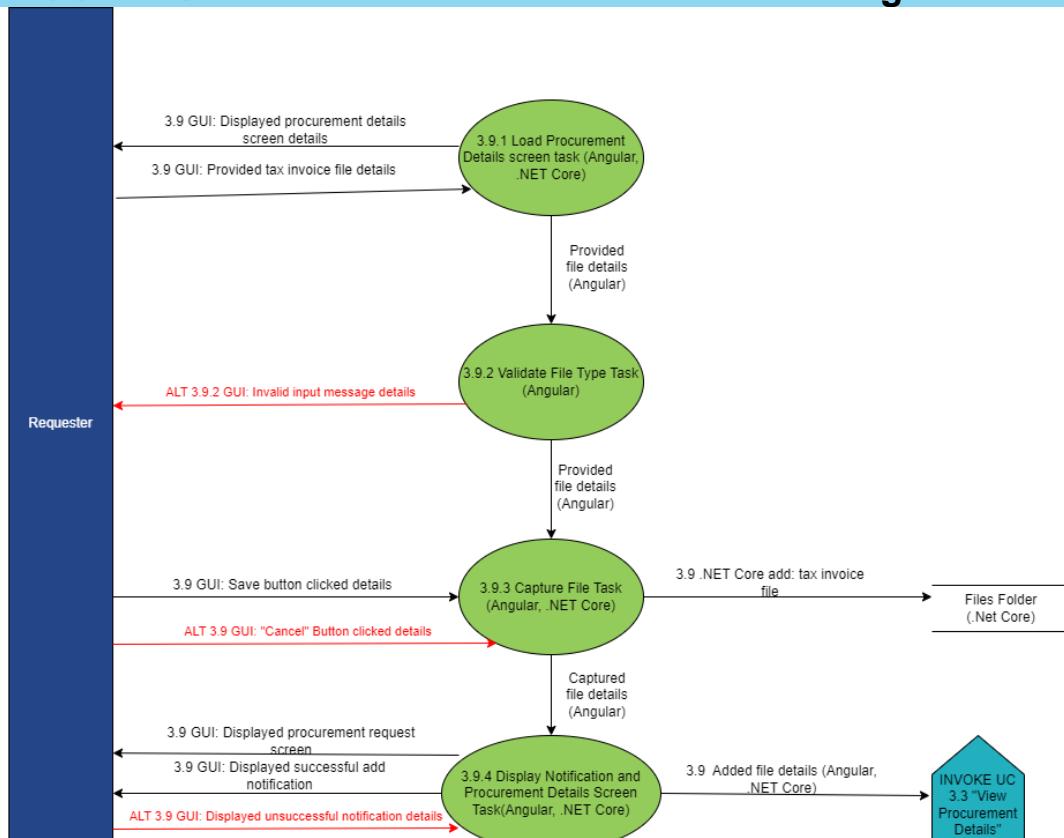


Figure 170 3.9 Upload Tax Invoice Tech Prim

### 3.10 BUDGET OWNER REQUISITION APPROVAL

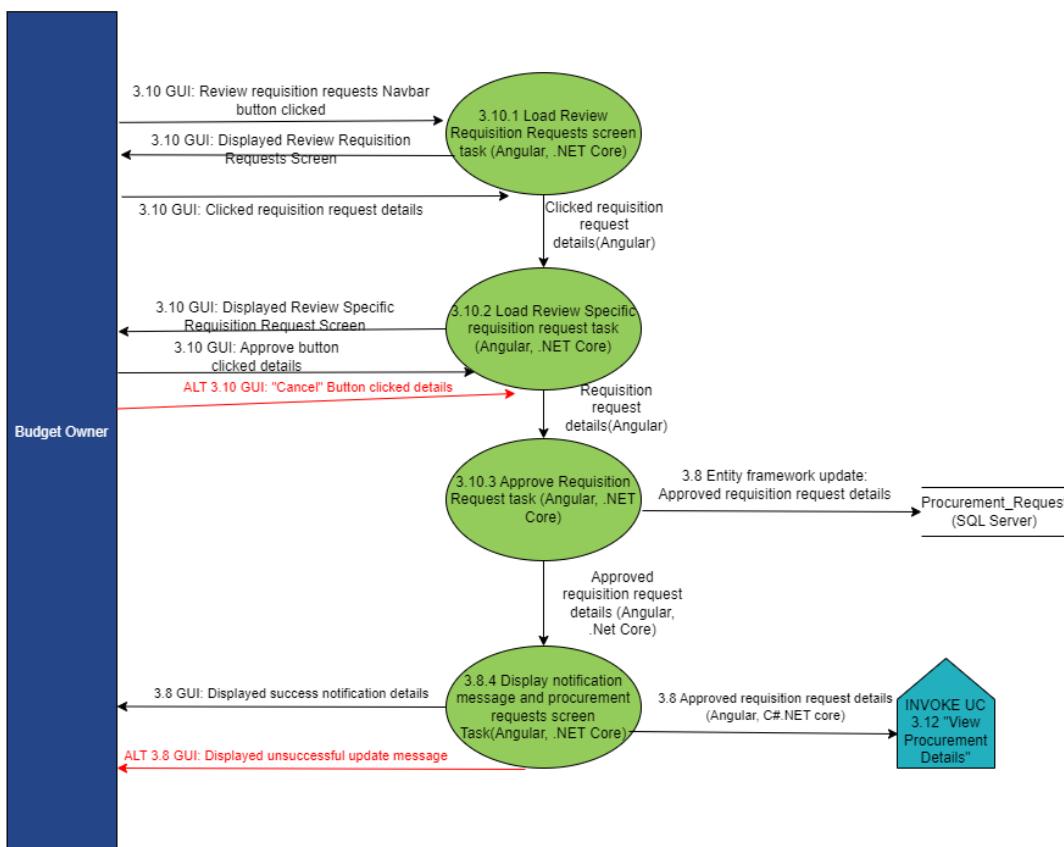


Figure 171 3.10 Budget Owner Requisition Approval Tech Prim

### 3.11 UPLOAD RECEIPT

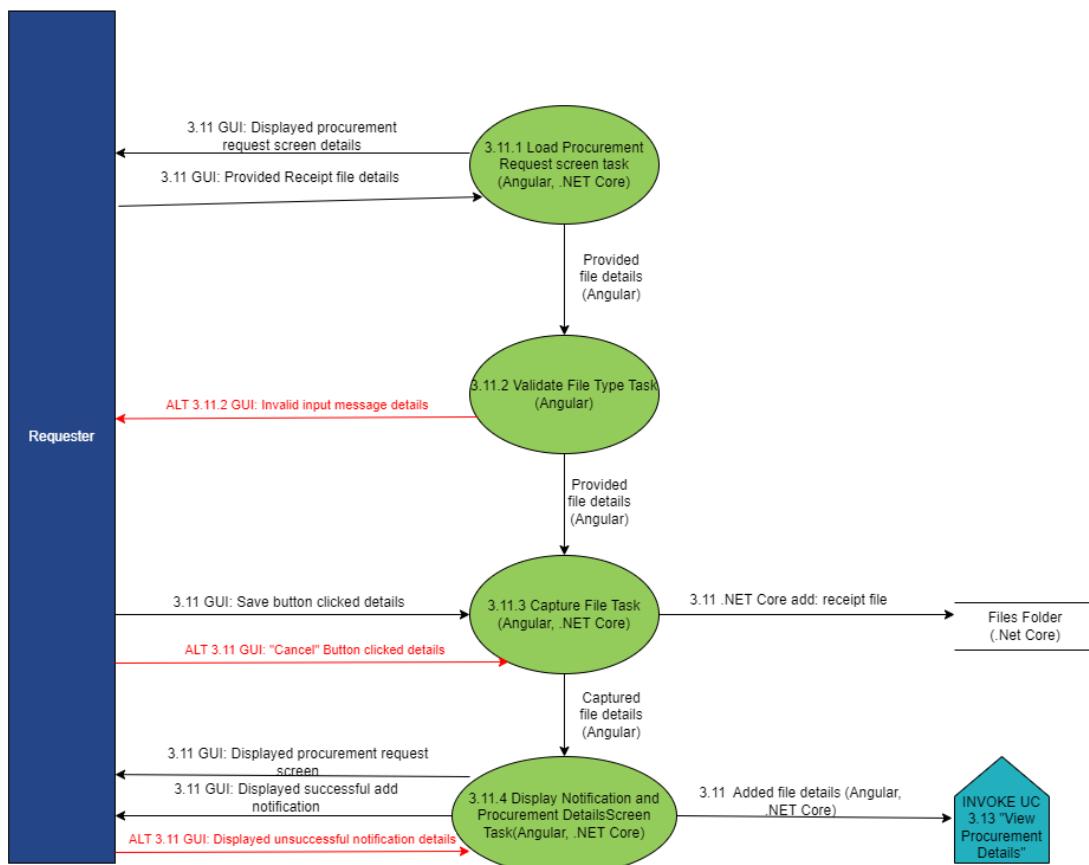


Figure 172 3.11 Upload Receipt Tech Prim

### 3.12 UPLOAD CREDIT CARD PAYMENT

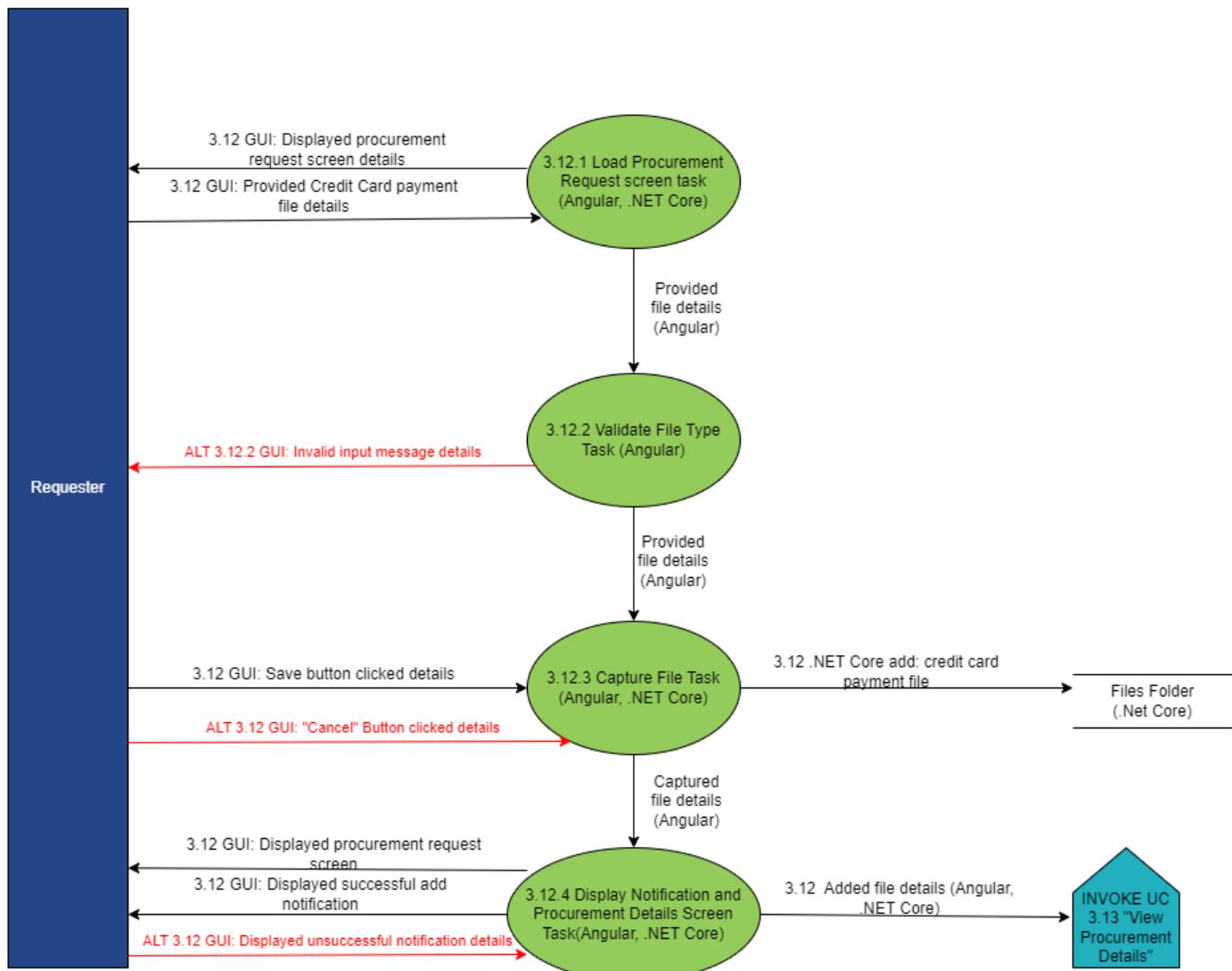


Figure 173: 3.12 Upload Credit Card Payment

### 3.13 VIEW PROCUREMENT DETAILS

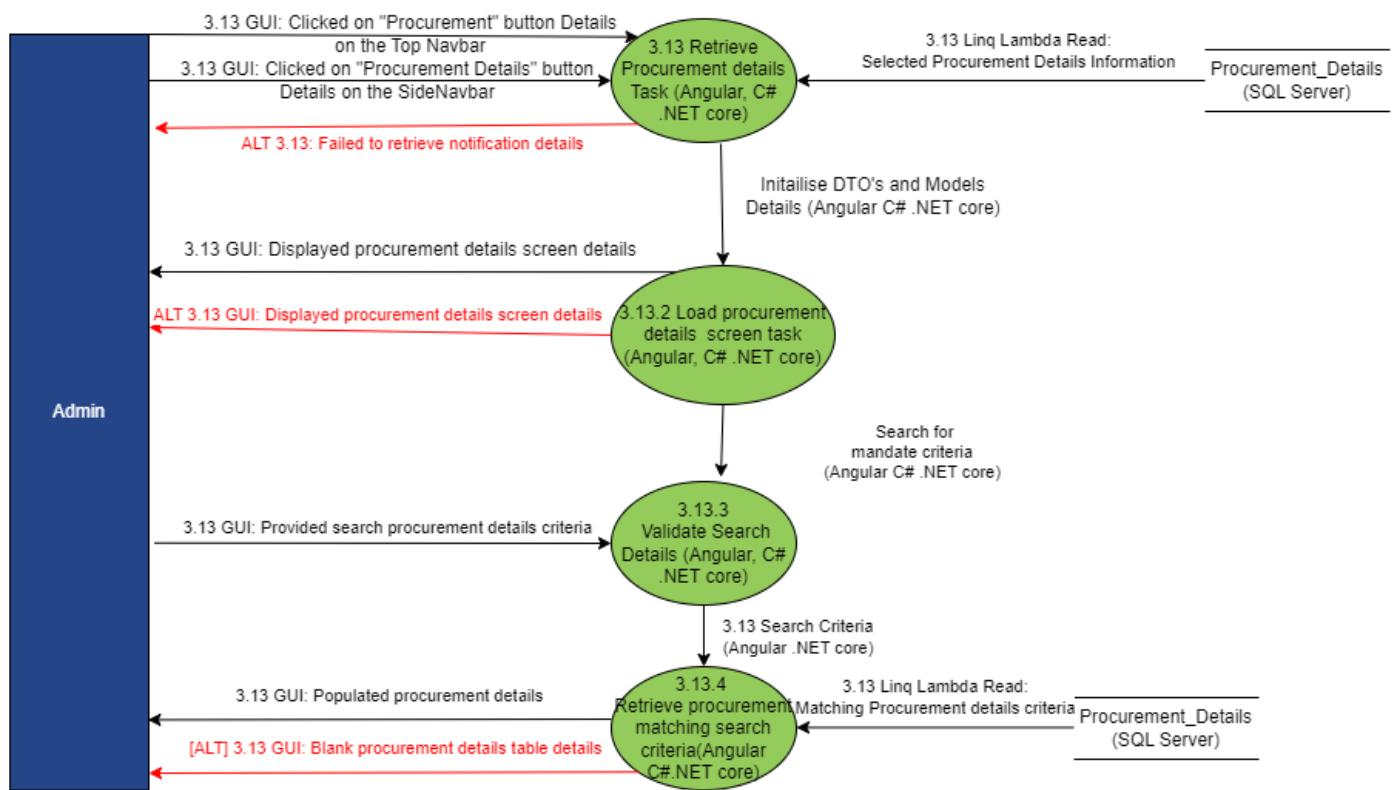


Figure 174: 3.13 View Procurement Details

### 4.1 FINALIZE PROCUREMENT REQUEST

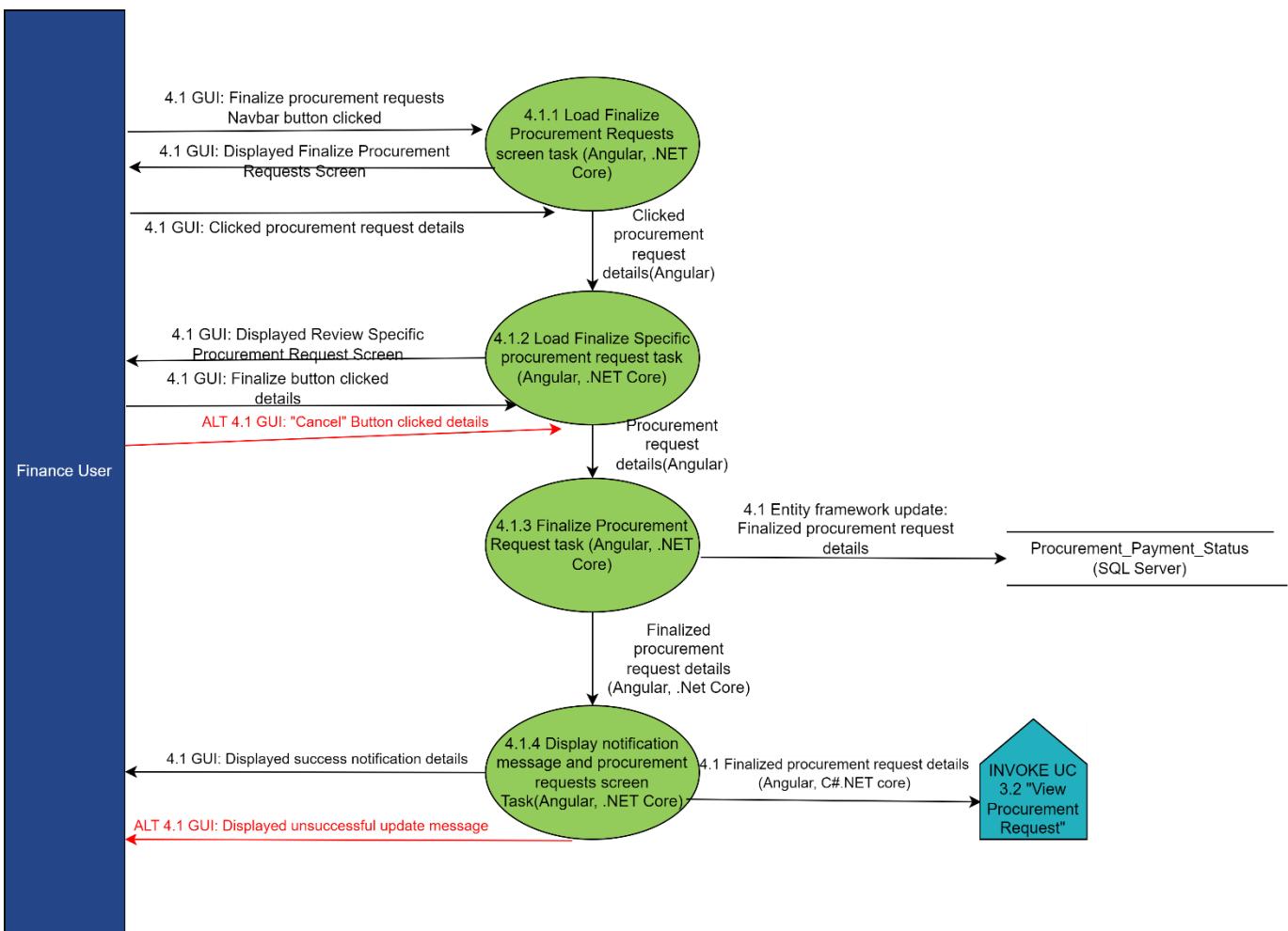
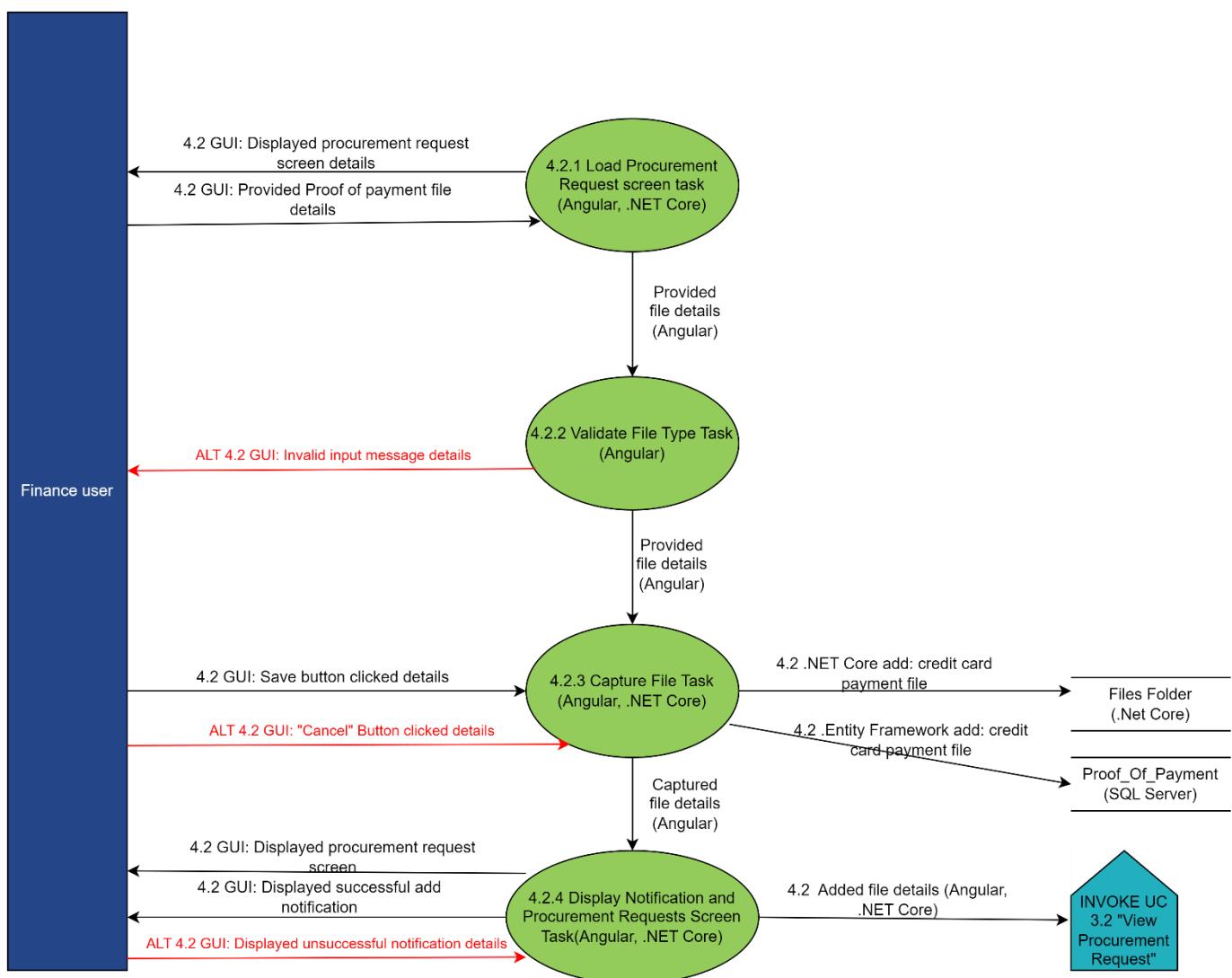


Table 211: 4.1 Finalize Procurement Request

## 4.2 UPLOAD PROOF OF PAYMENT


**Table 212: 4.2 Upload Proof of Payment**

### 4.3 CREATE BUDGET ALLOCATION

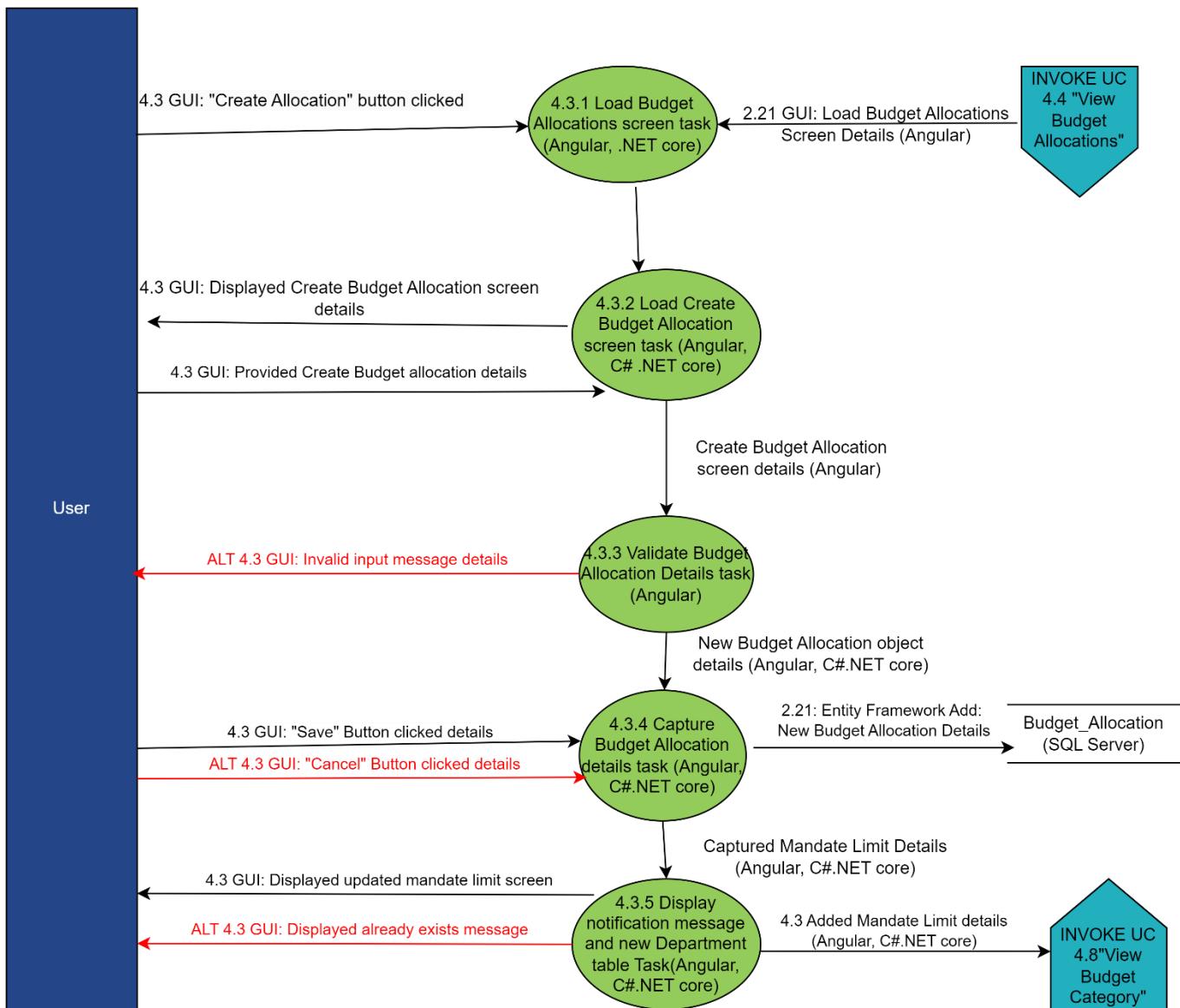


Figure 175 4.3 Create Budget Allocation Tech Prim

## 4.4 VIEW BUDGET ALLOCATION

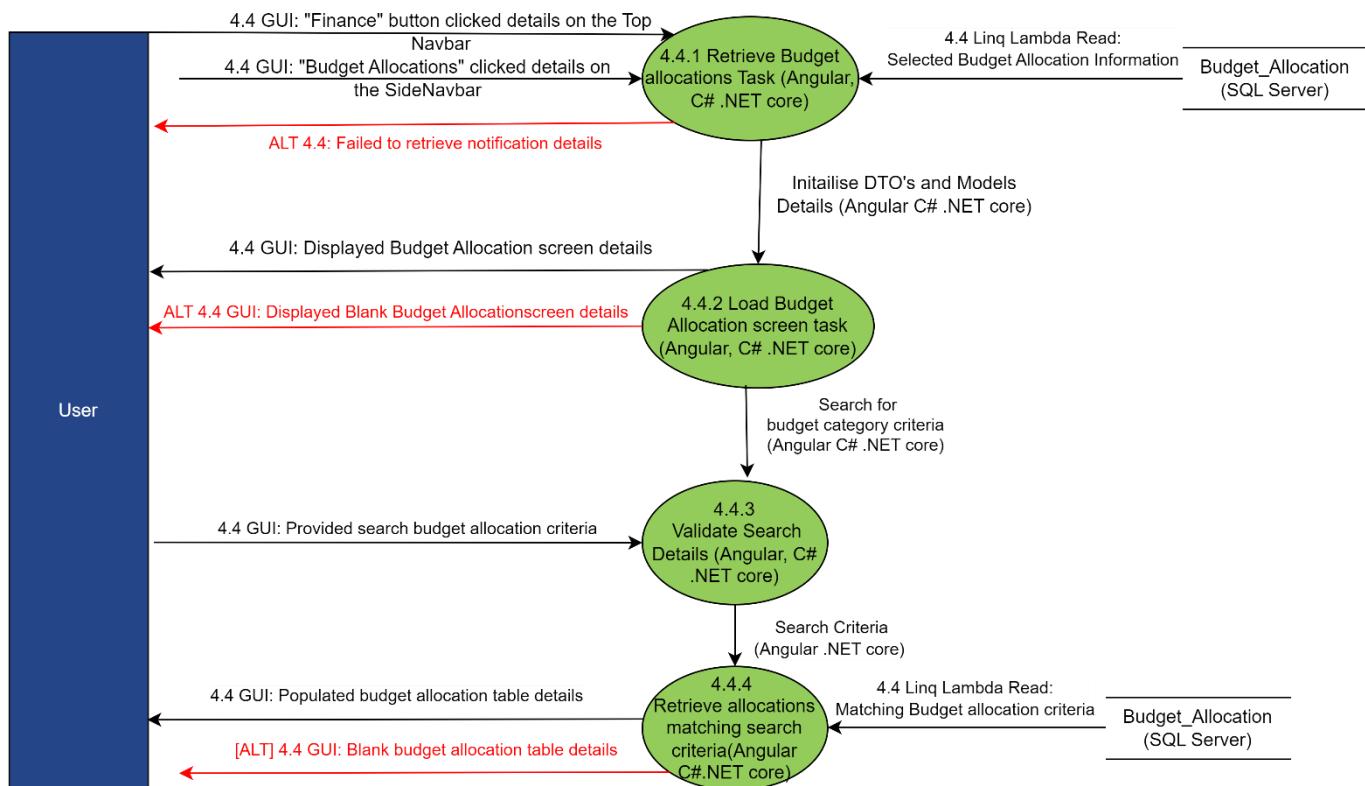


Figure 176 4.4 View Budget Allocation Tech Prim

## 4.5 UPDATE BUDGET ALLOCATION

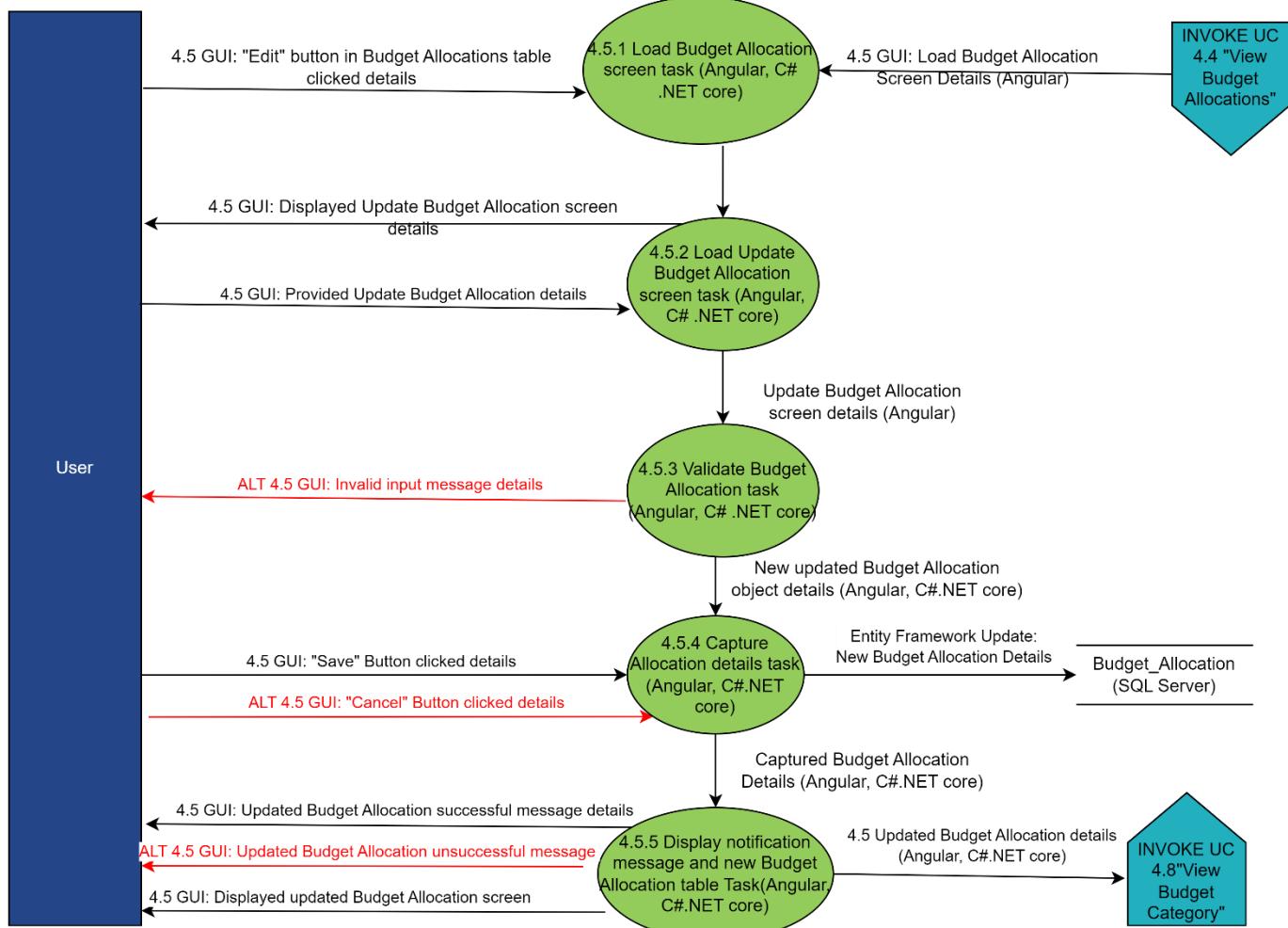


Figure 177 4.5 Update Budget Allocation Tech Prim

## 4.6 DELETE BUDGET ALLOCATION

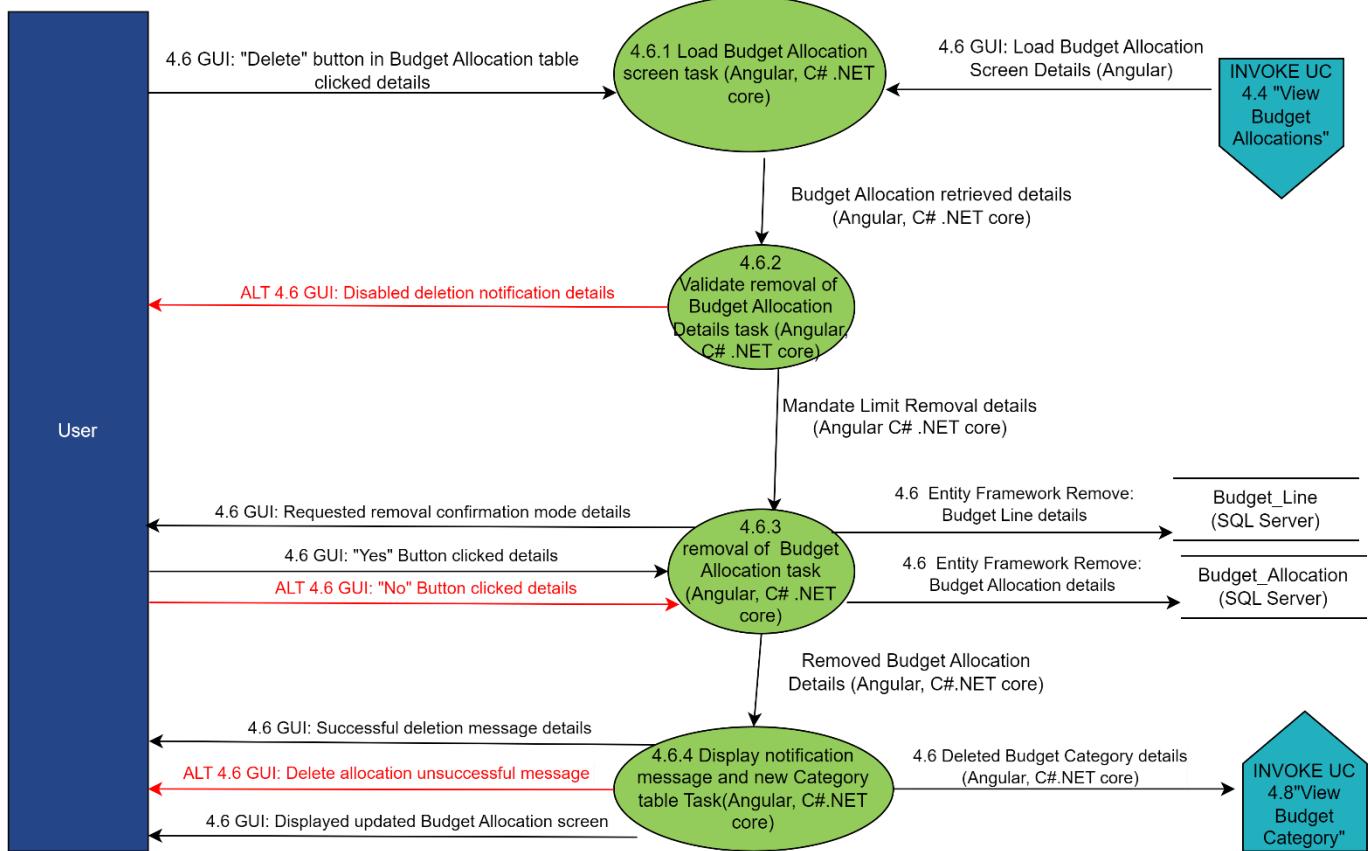
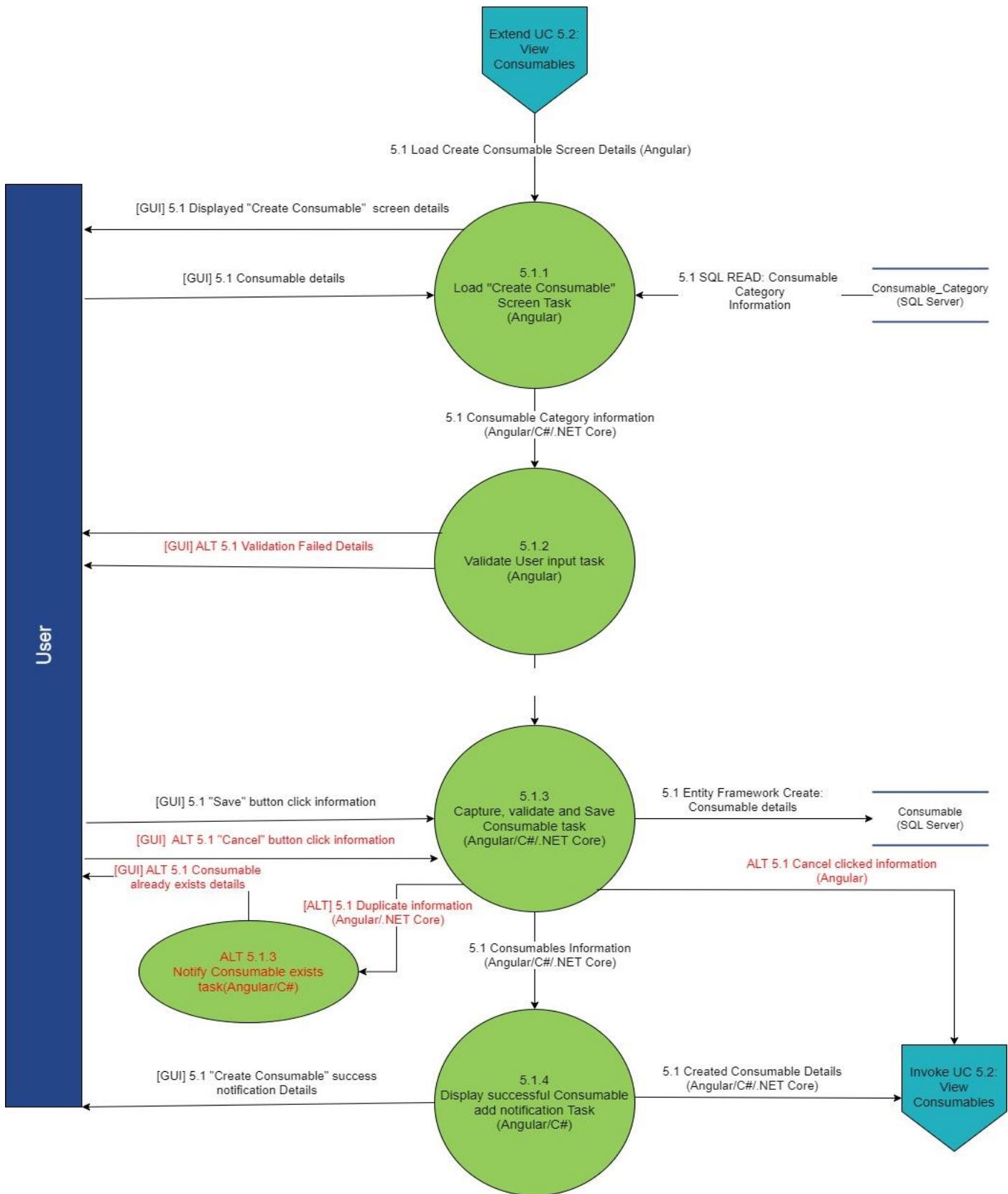


Figure 178 4.6 Delete Budget Allocation Tech Prim

**USE CASE 5.1-5.8 & 5.9 , 5.10 & 1.1-1.4 & 3.1-3.4**

## 5.1 CREATE CONSUMABLE


**Figure 179 5.1 Create Consumable Tech Prim**

### 5.2 VIEW CONSUMABLES

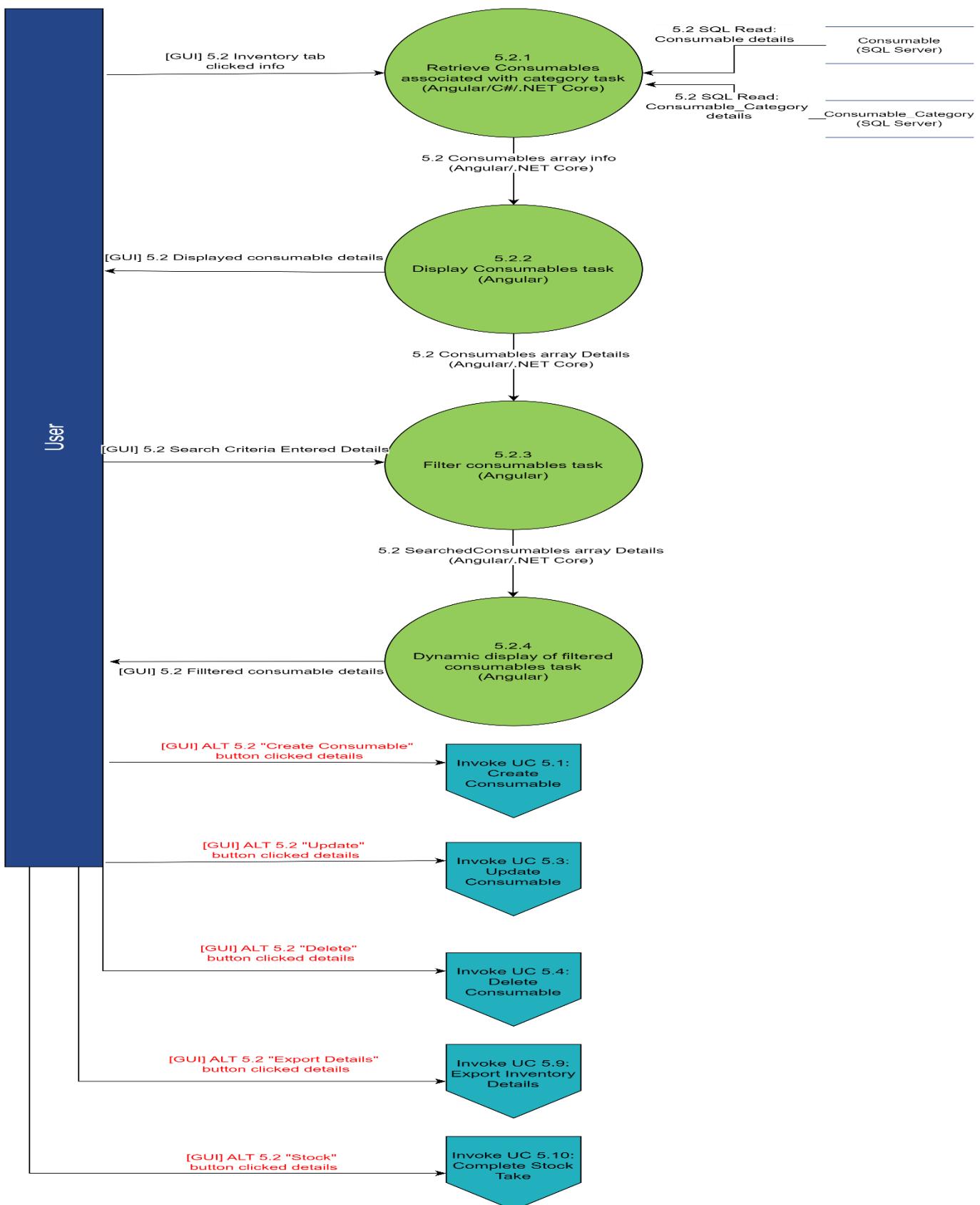


Figure 180 5.2 View Consumables Tech Prim

### 5.3 UPDATE CONSUMABLE

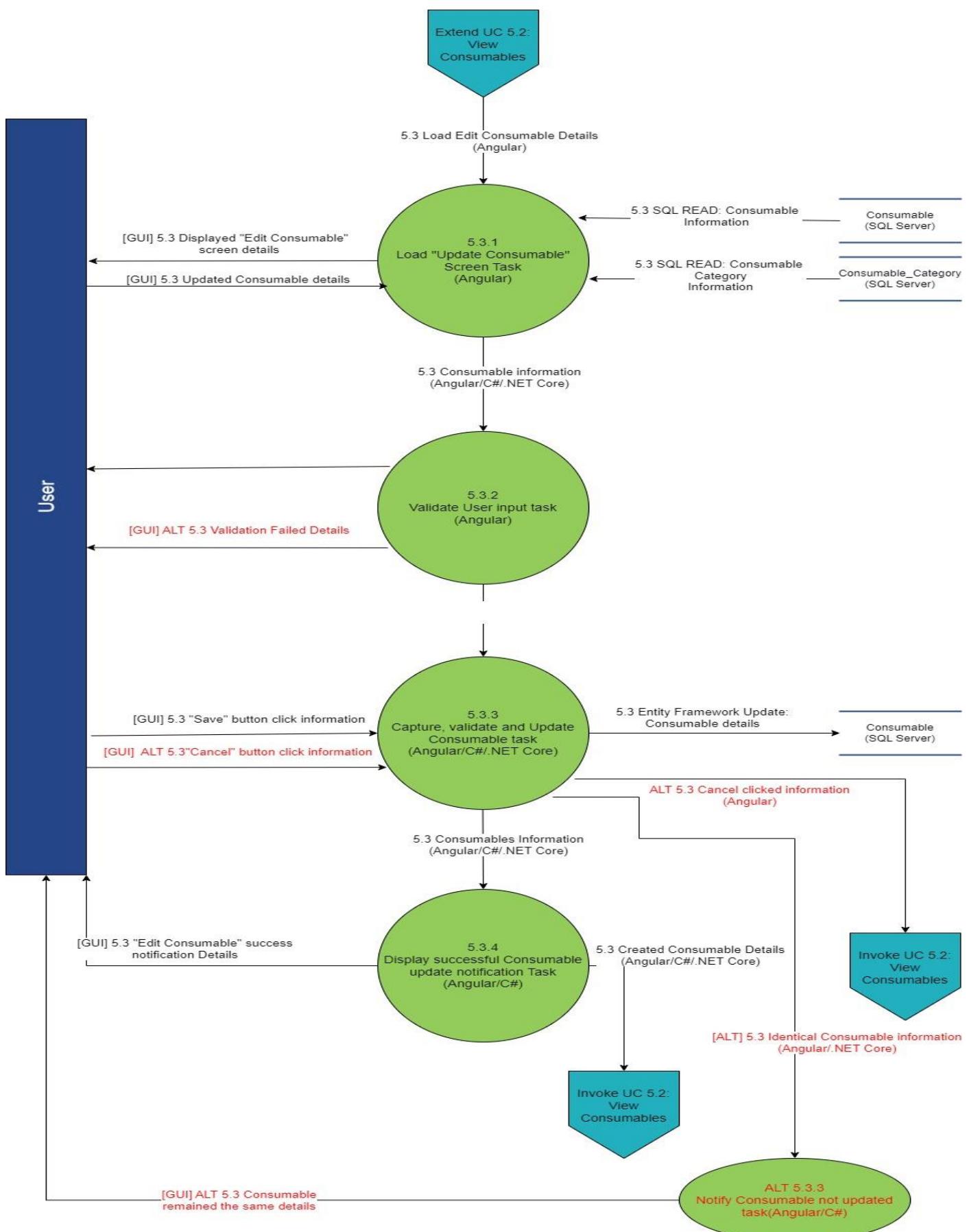


Figure 181 5.3 Update Consumable Tech Prim

### 5.4 DELETE CONSUMABLE

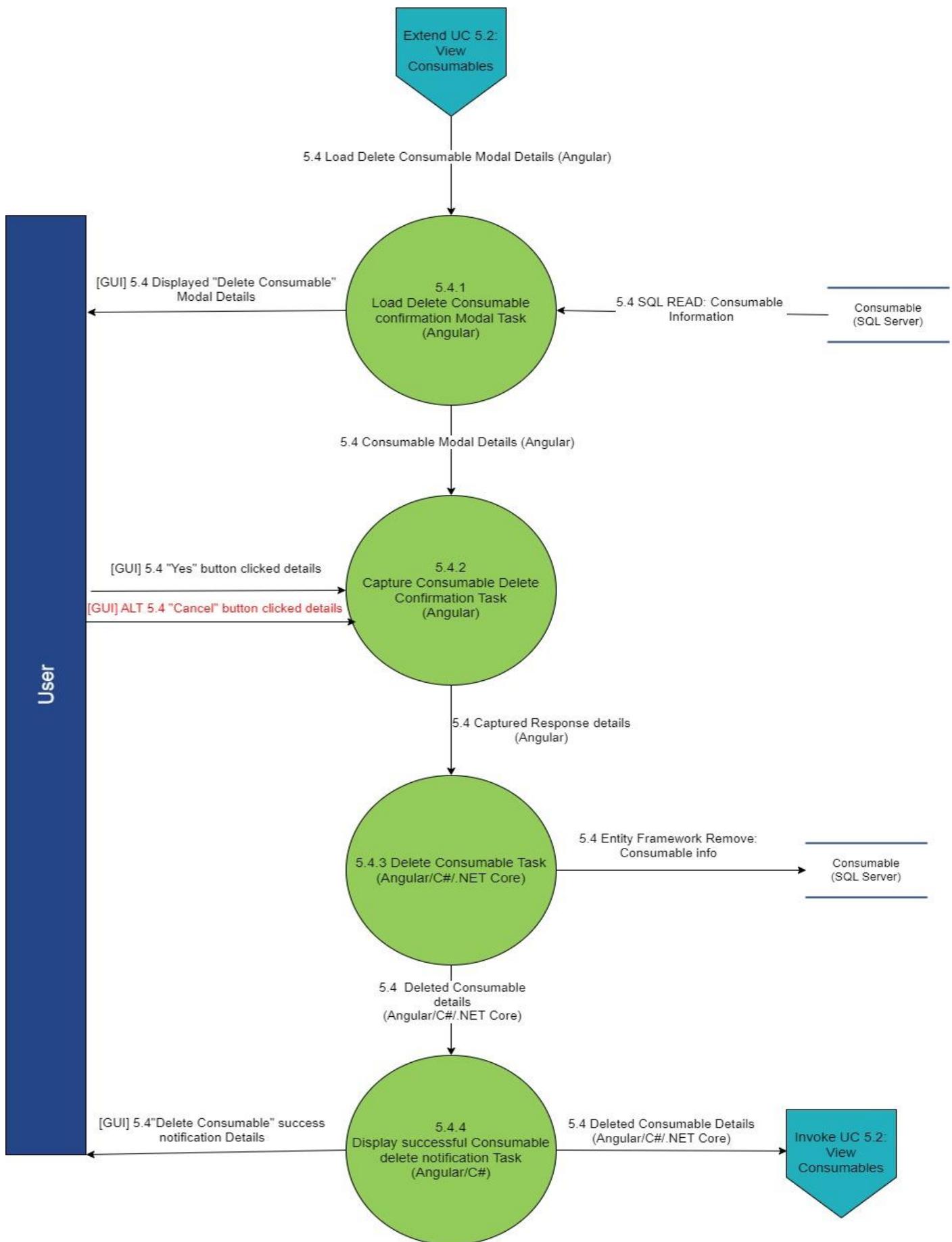


Figure 182 5.4 Delete Consumable Tech Prim

## 5.5 CREATE CONSUMABLE CATEGORY

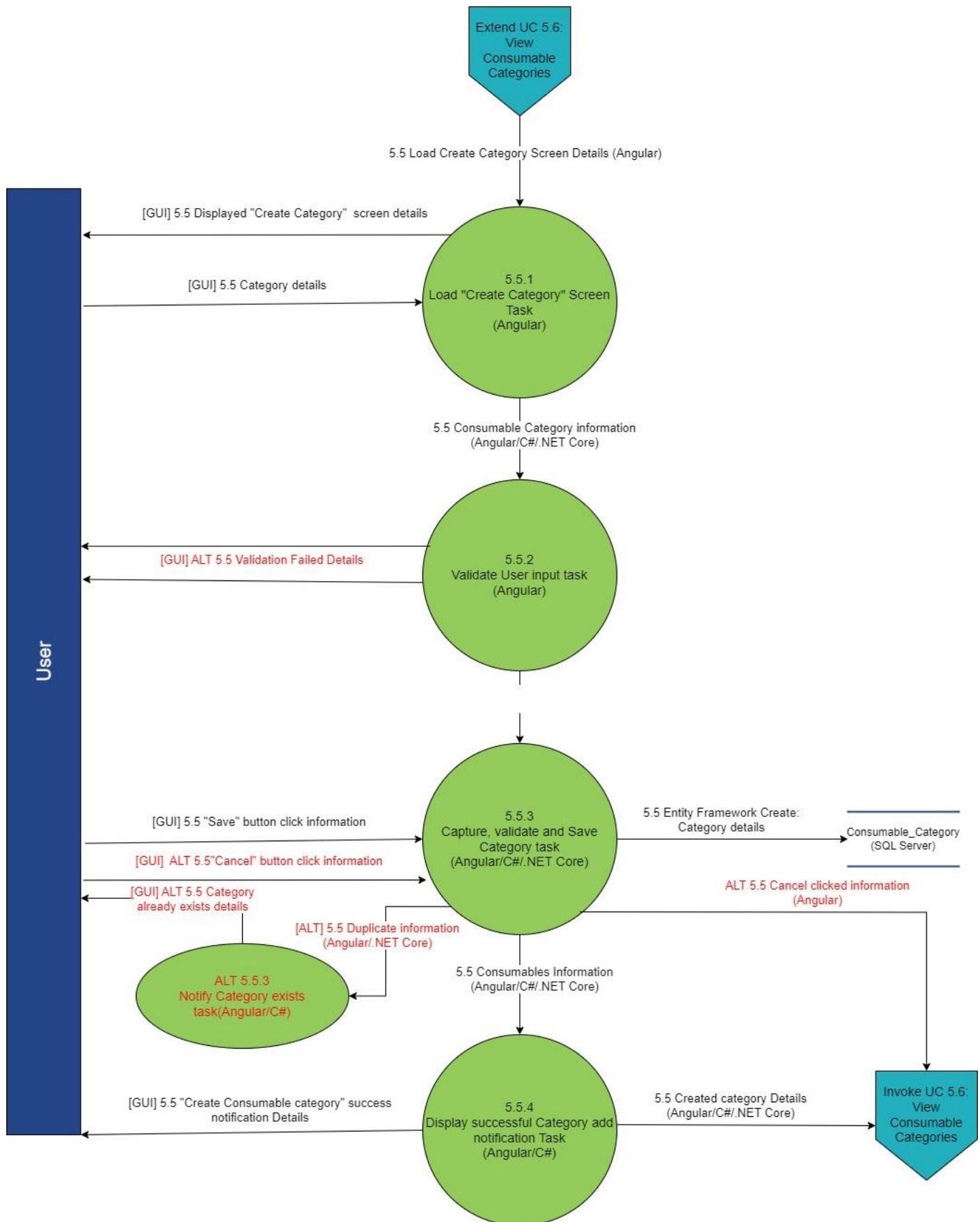


Figure 183 5.5 Create Consumable Category Tech Prim

### 5.6 VIEW CONSUMABLE CATEGORIES

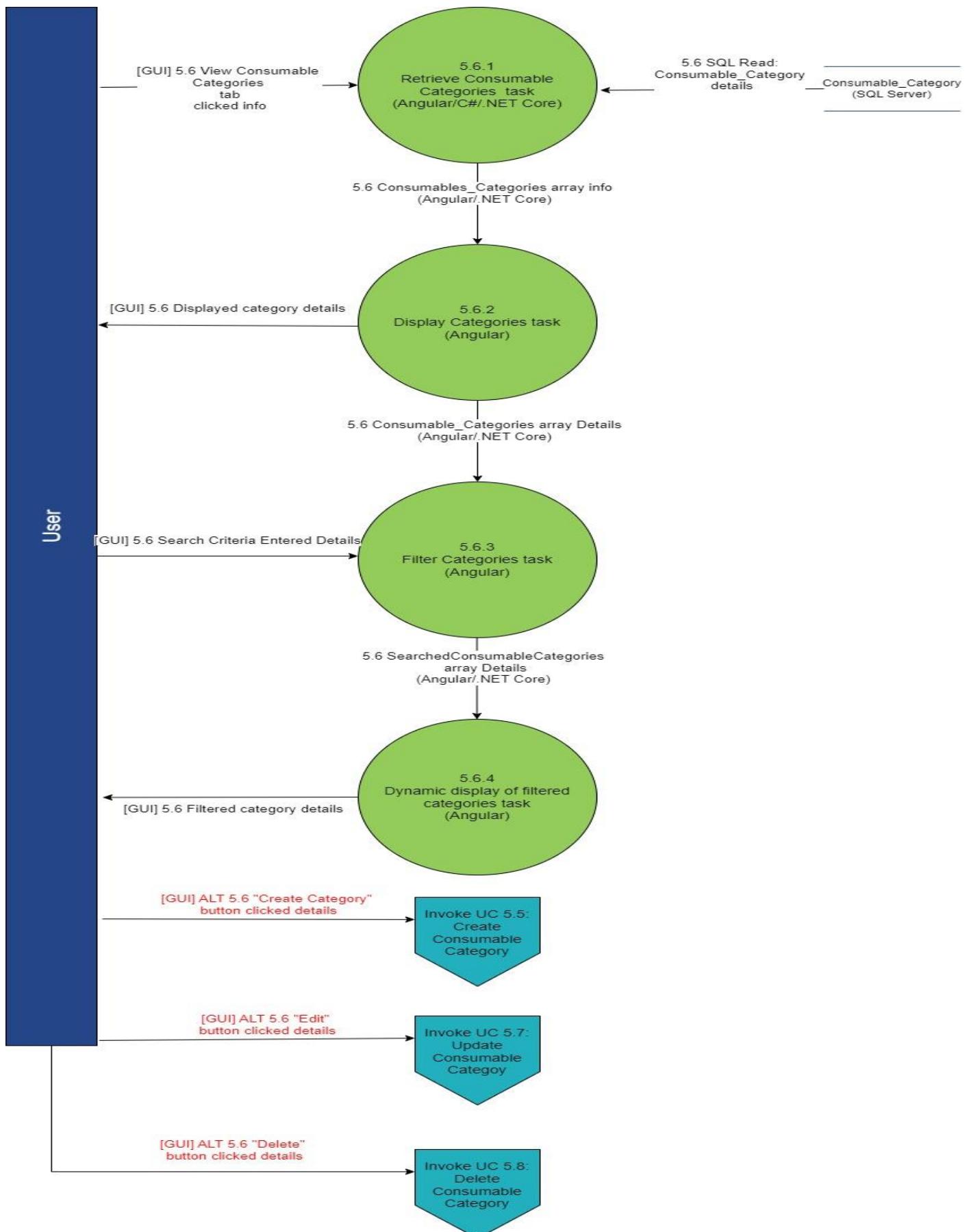


Figure 184 5.6 View Consumable Categories Tech Prim

### 5.7 UPDATE CONSUMABLE CATEGORY

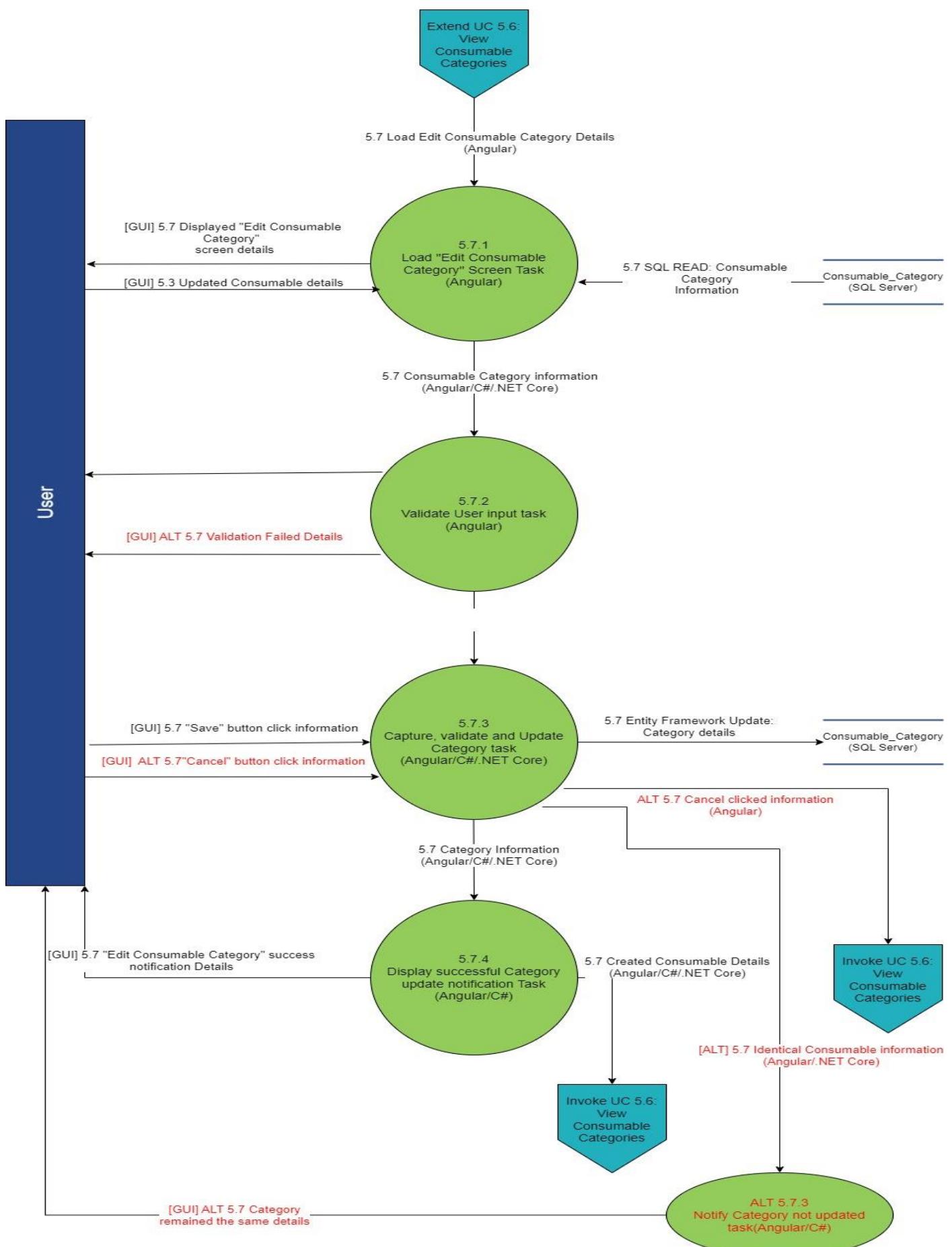


Figure 185 5.7 Update Consumable Category Tech Prim

### 5.8 DELETE CONSUMABLE CATEGORY

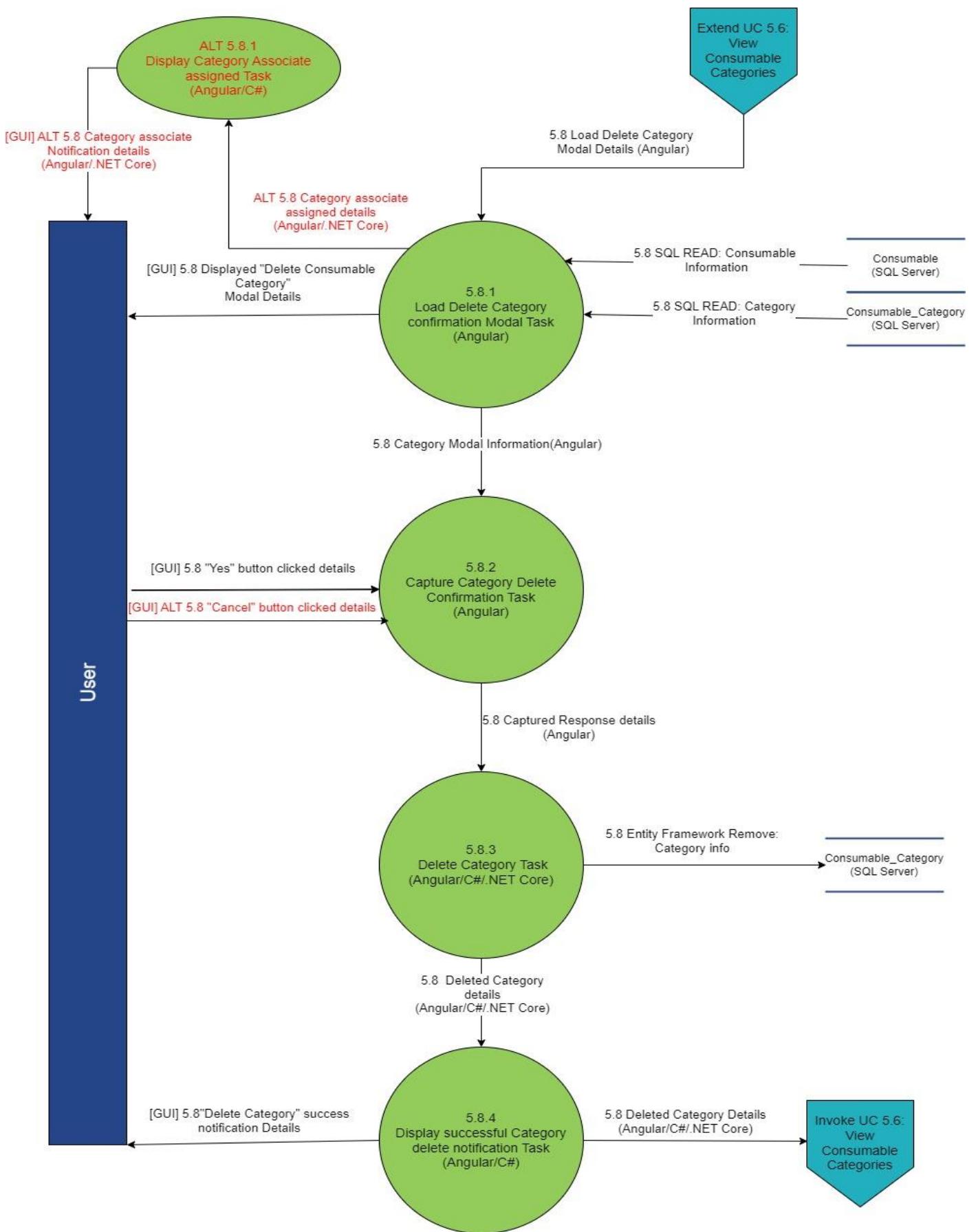


Figure 186 5.8 Delete Consumable Category Tech Prim

## 5.9 EXPORT INVENTORY DETAILS

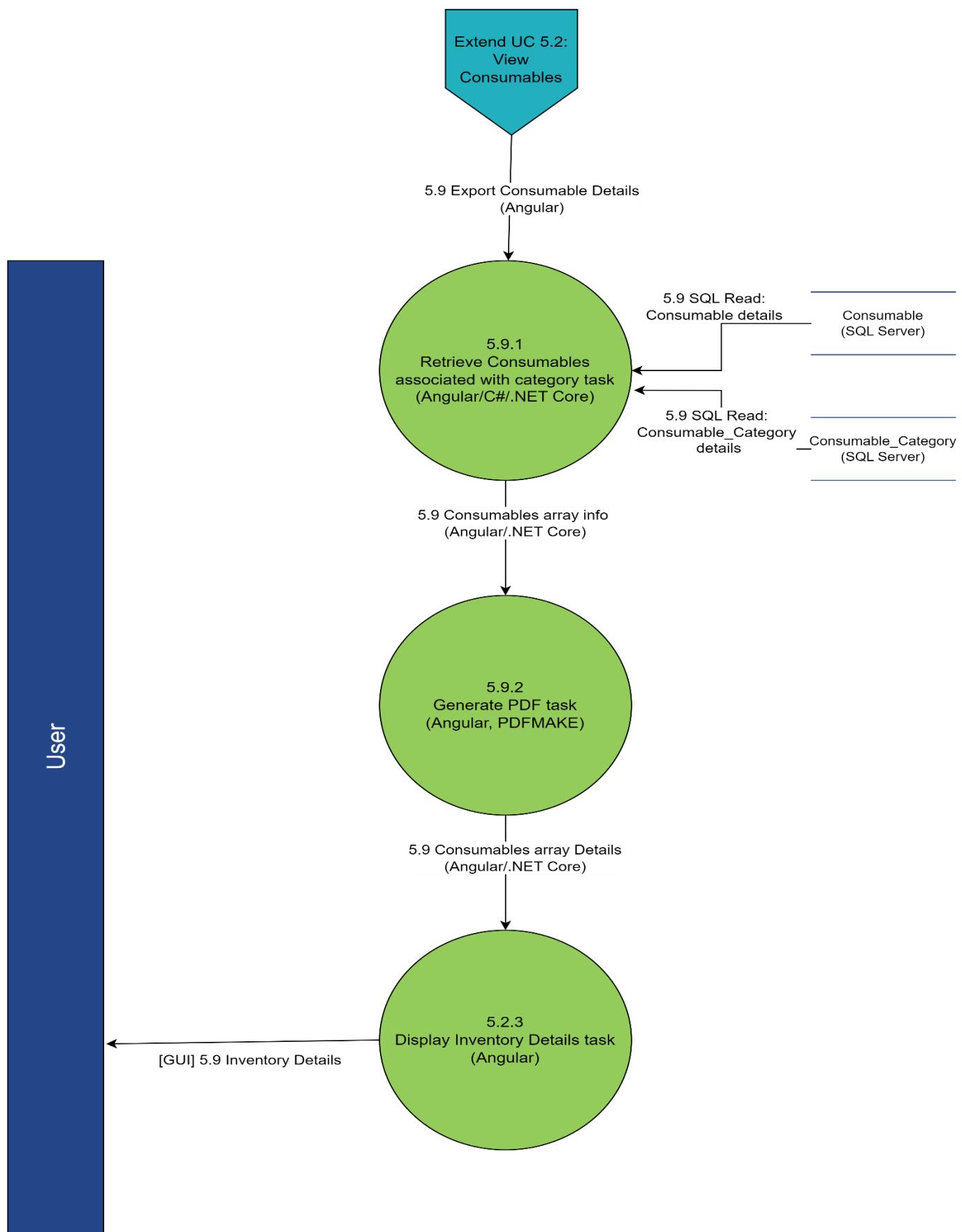


Figure 187 5.9 Export Inventory Details Tech Prim

## 5.10 UPDATE STOCK

sequence

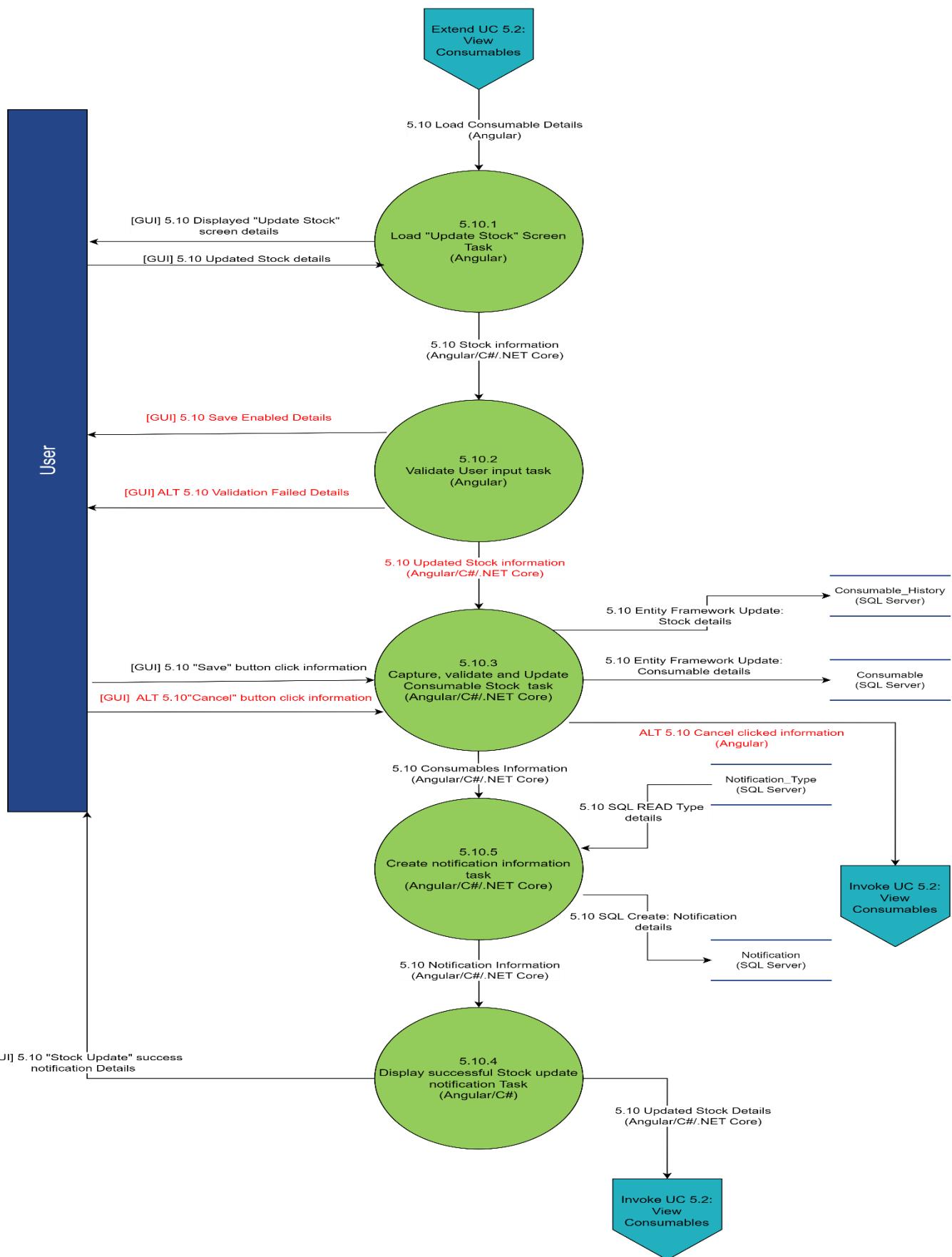


Figure 188 5.10 Update Stock Tech Prim

## 1.1 LOGIN

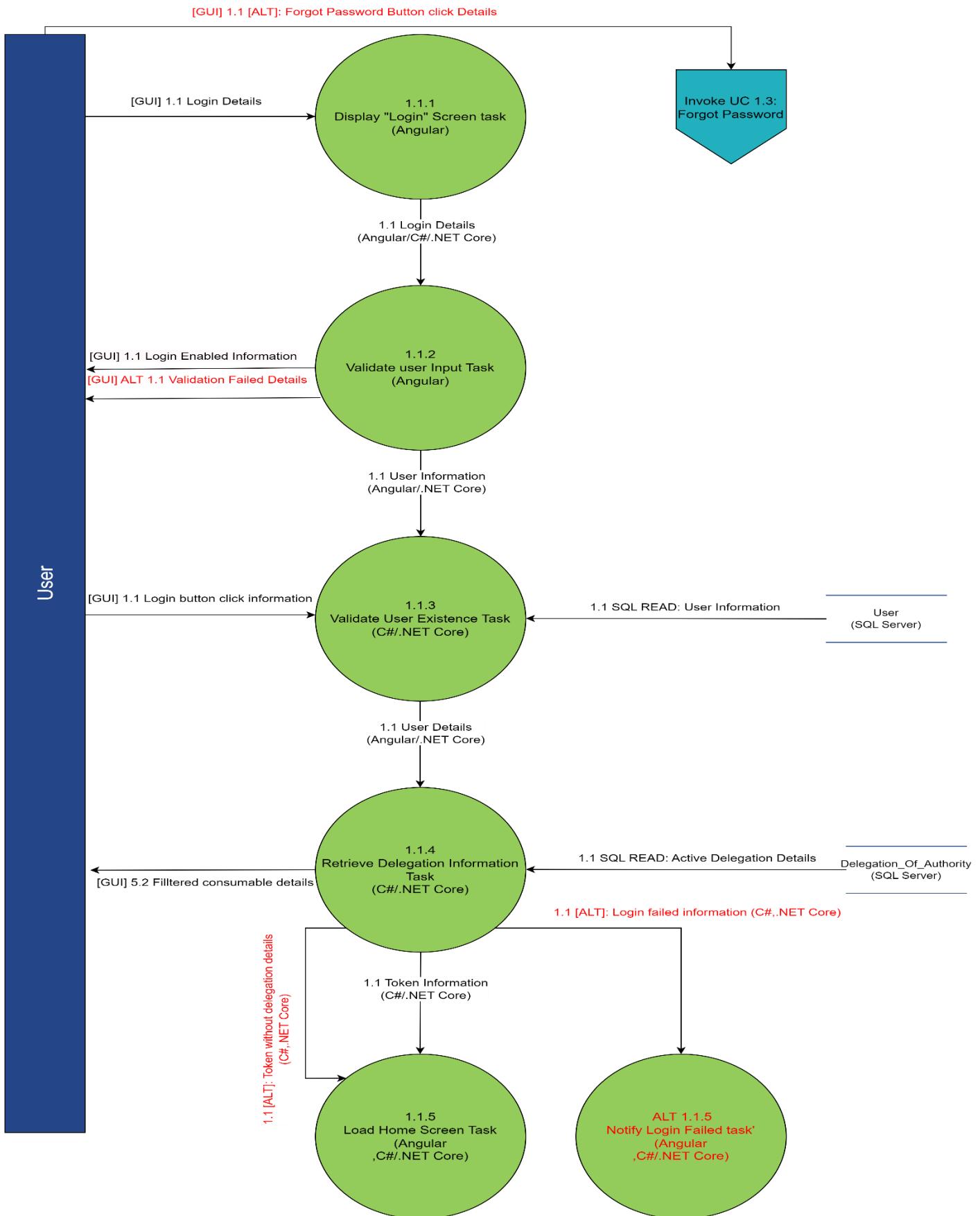


Figure 189 1.1 Login Tech Prim

## 1.2 LOGOUT

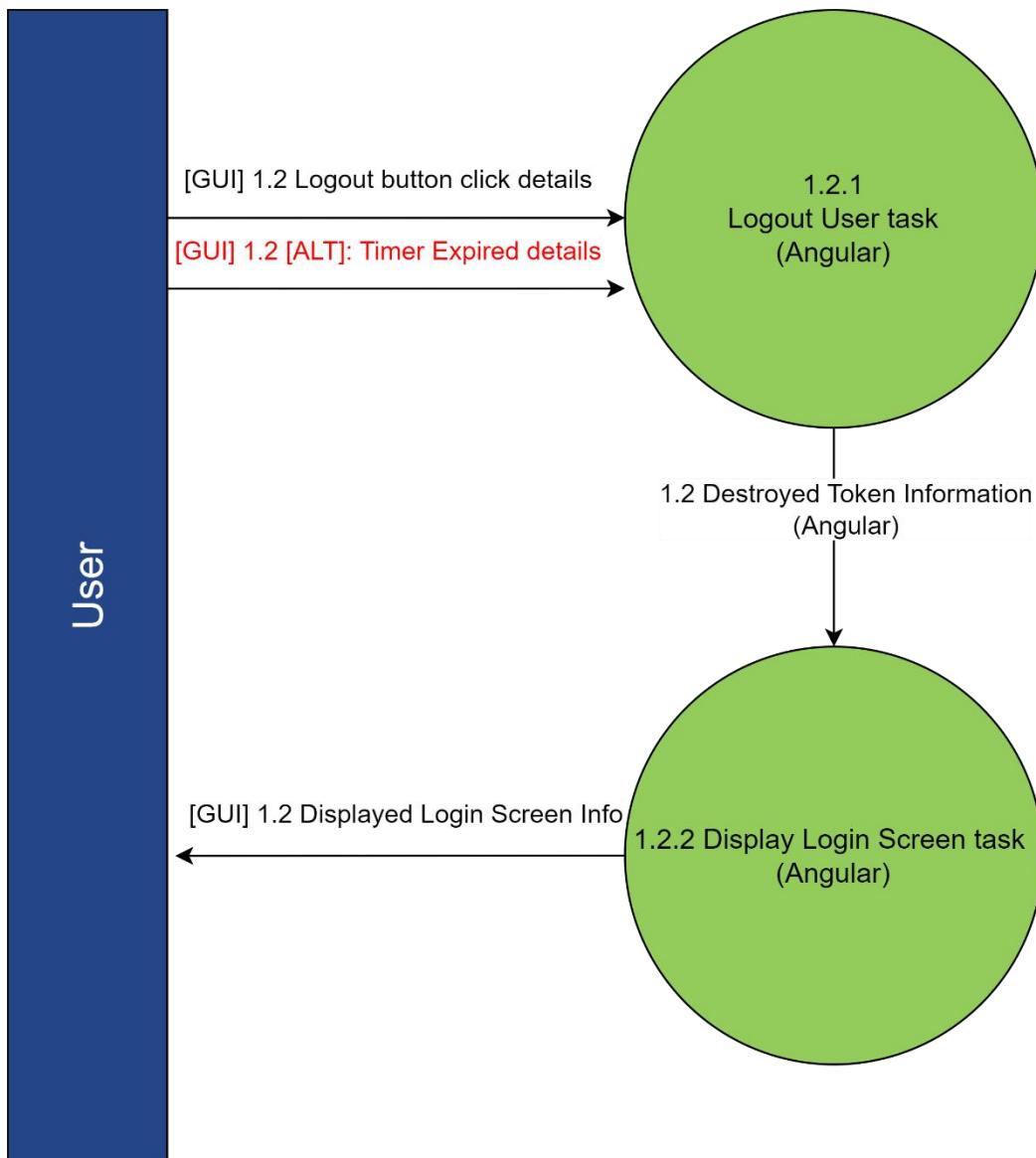


Figure 190 1.2 Logout Tech Prim

### 1.3 FORGOT PASSWORD

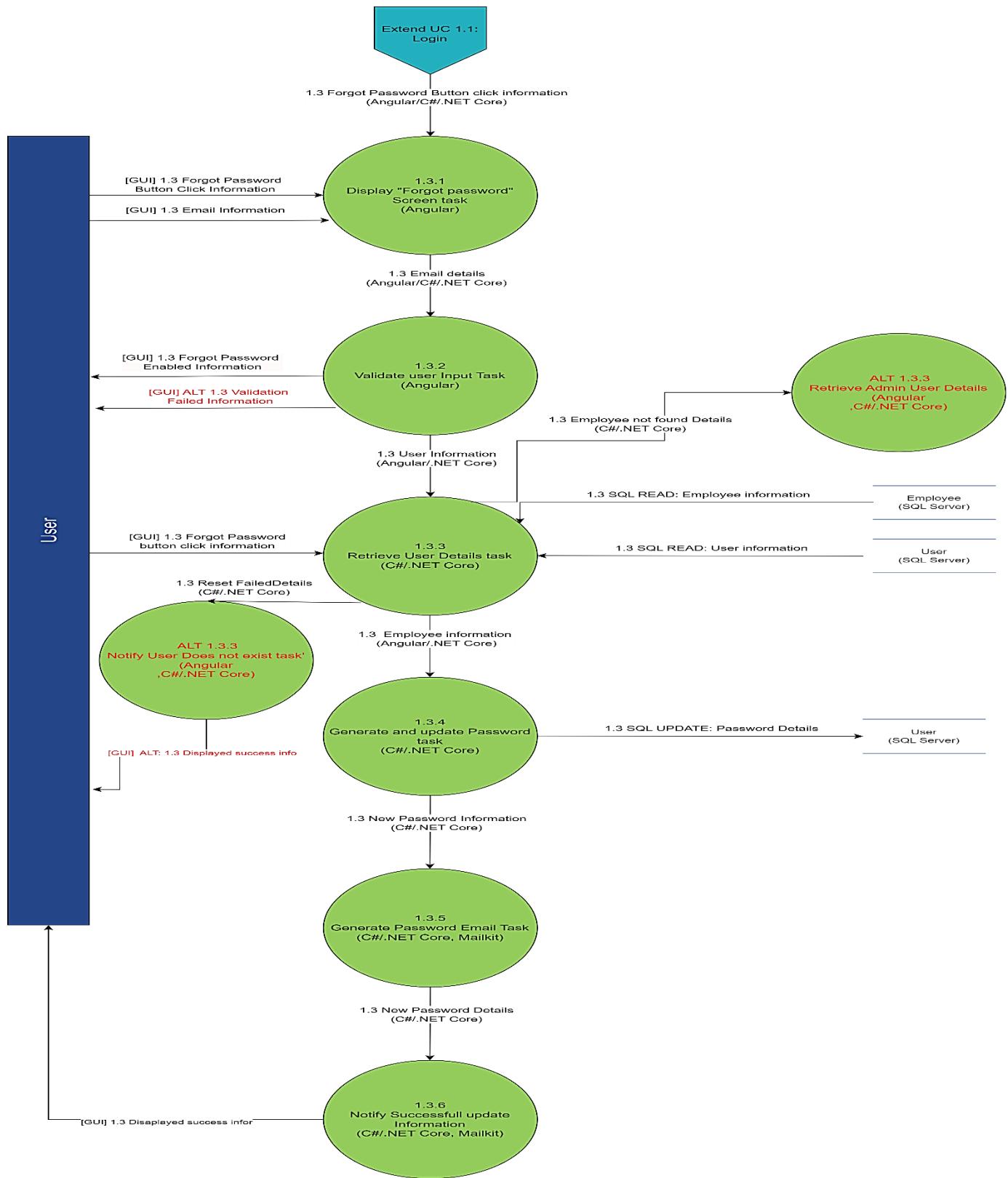


Figure 191 1.3 Forgot Password Tech Prim

## 1.4 UPDATE PASSWORD

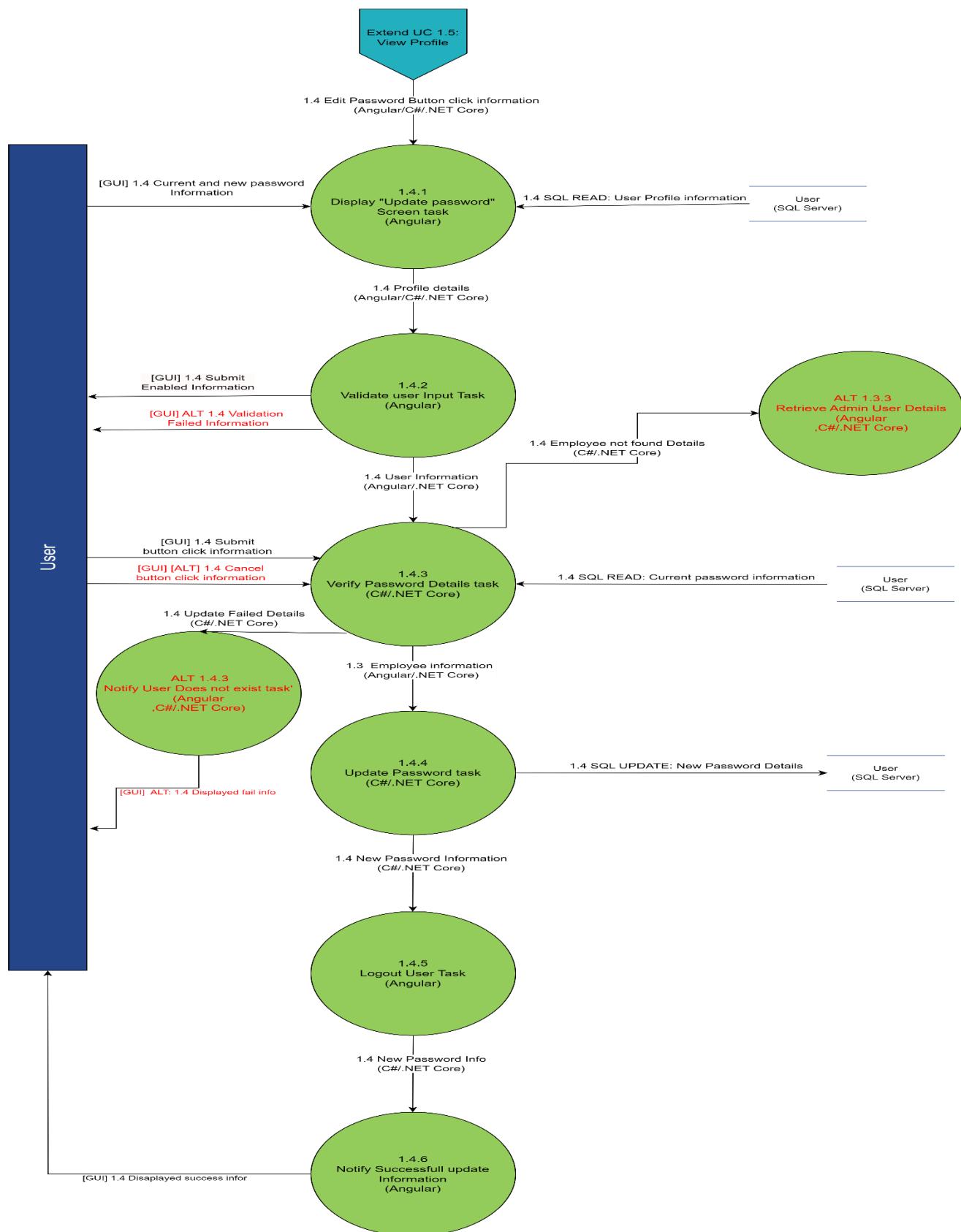


Figure 192 1.4 Update Password Tech Prim

### 3.1 CREATE PROCUREMENT REQUEST

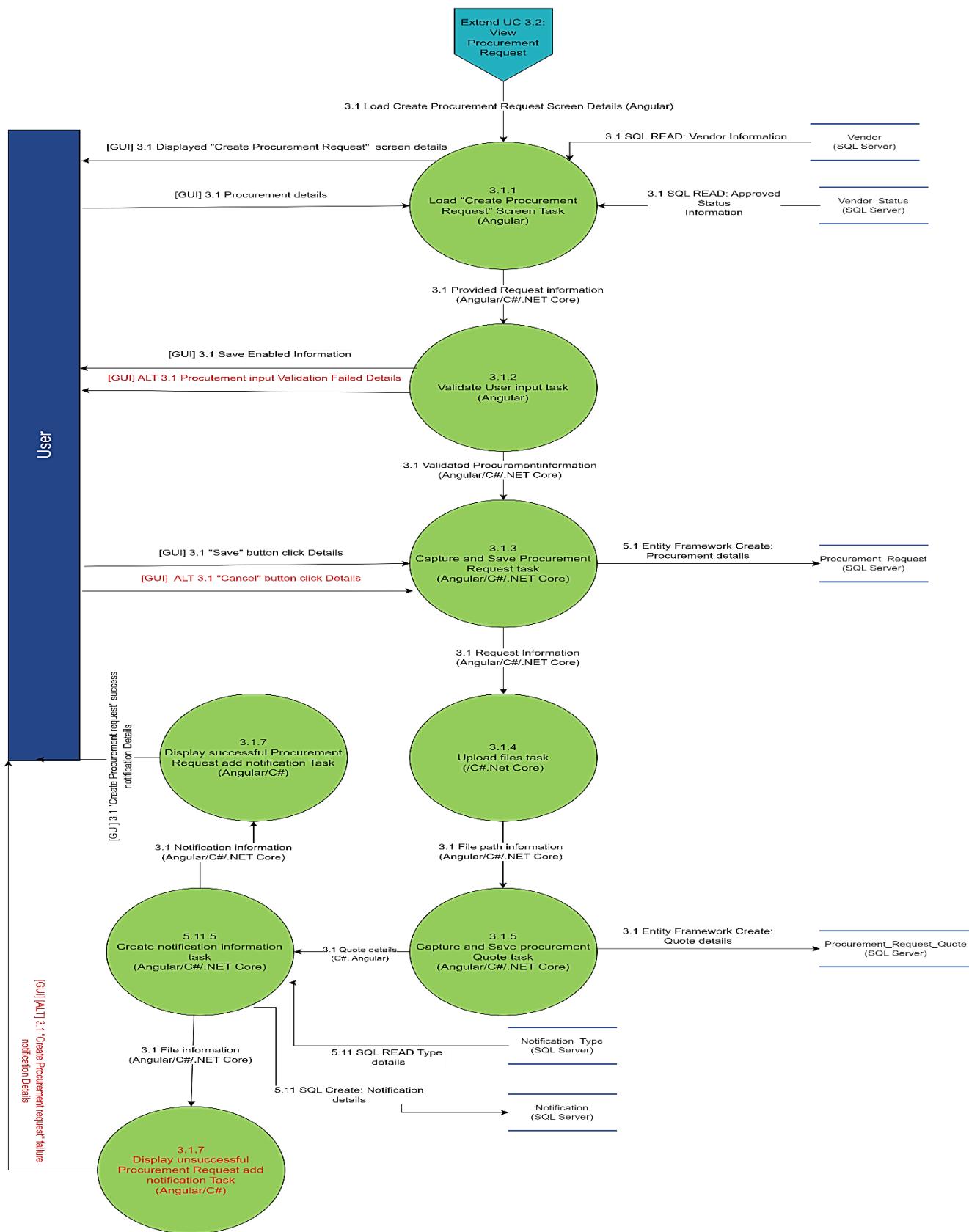


Figure 193 3.1 Create Procurement Request Tech Prim

### 3.2 VIEW PROCUREMENT REQUEST

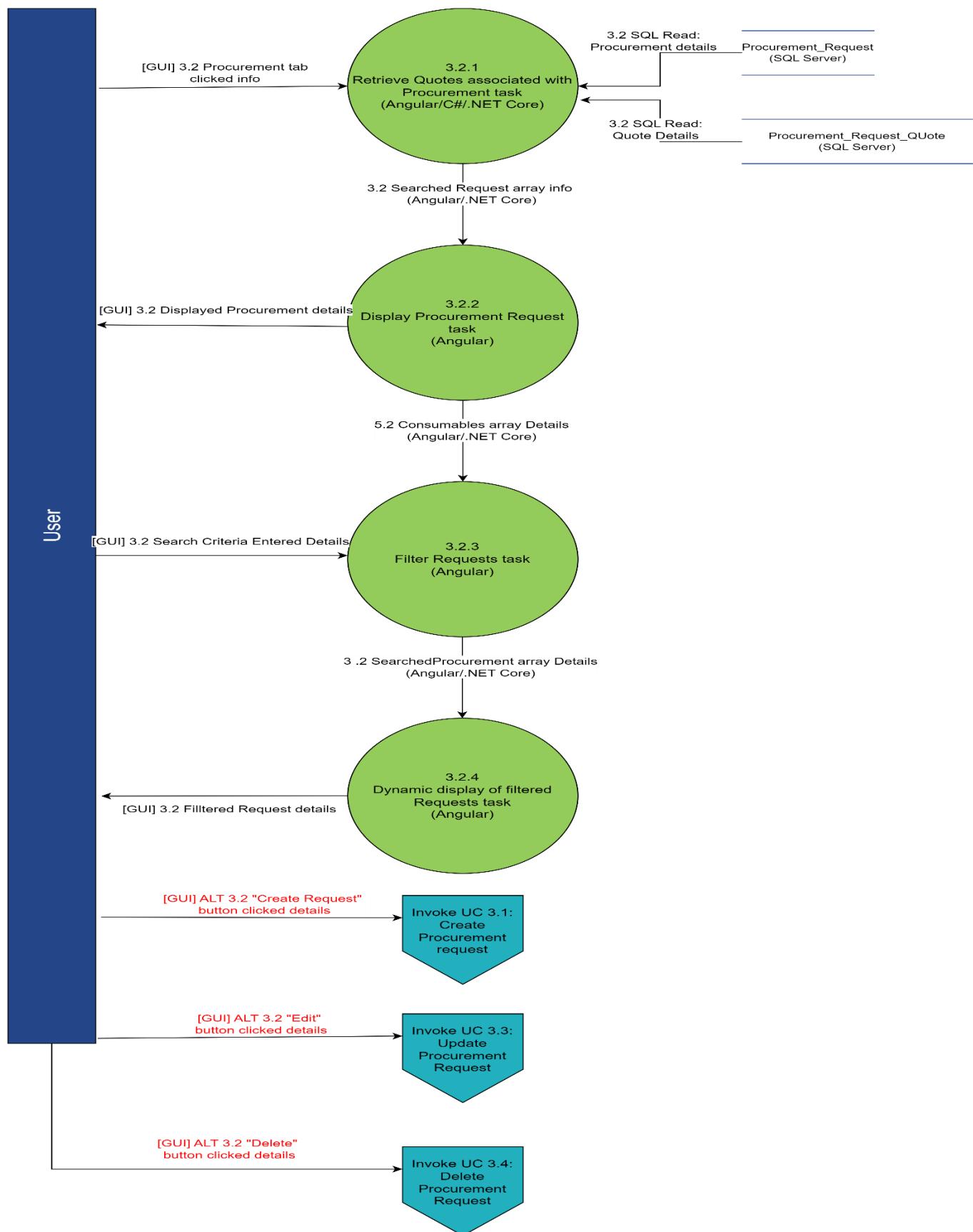


Figure 194 3.2 View Procurement Request Tech Prim

### 3.3 UPDATE PROCUREMENT REQUEST

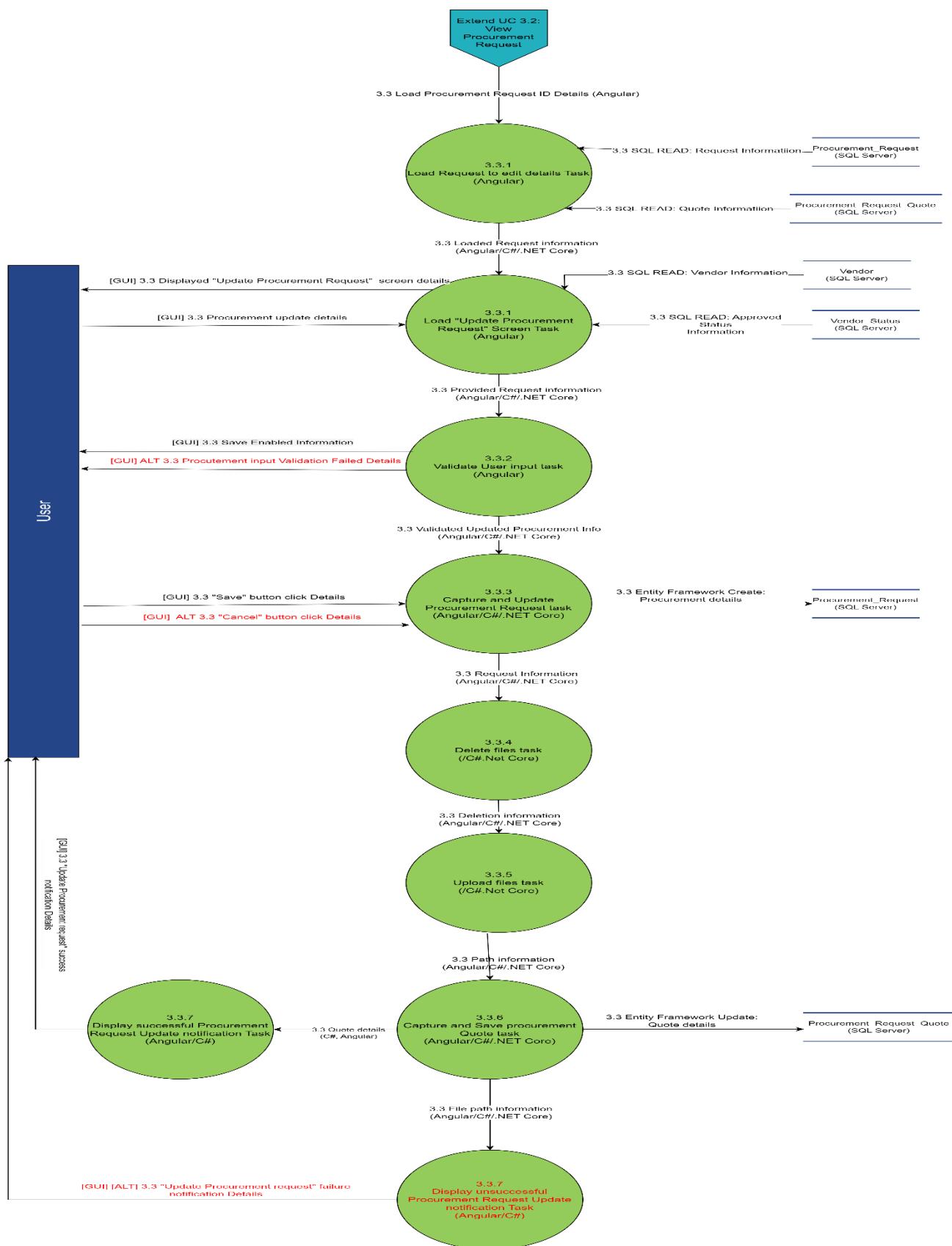


Figure 195 3.3 Update Procurement Request Tech Prim

### 3.4 DELETE PROCUREMENT REQUEST

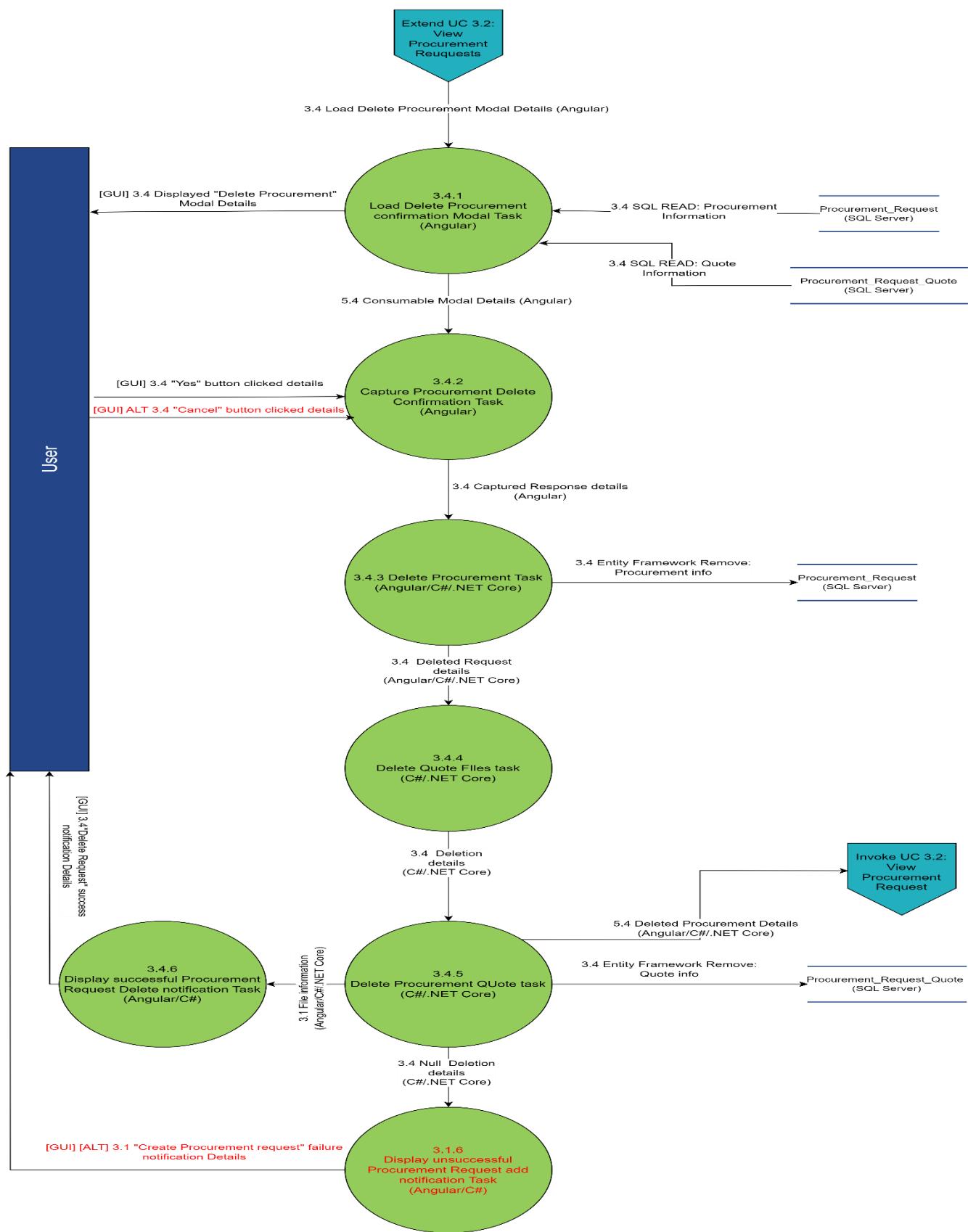


Figure 196 3.4 Delete Procurement Request Tech Prim

## USE CASE 3.5-3.7, 6.5, 6.10-6.14

## 3.5 APPROVE PROCUREMENT REQUEST

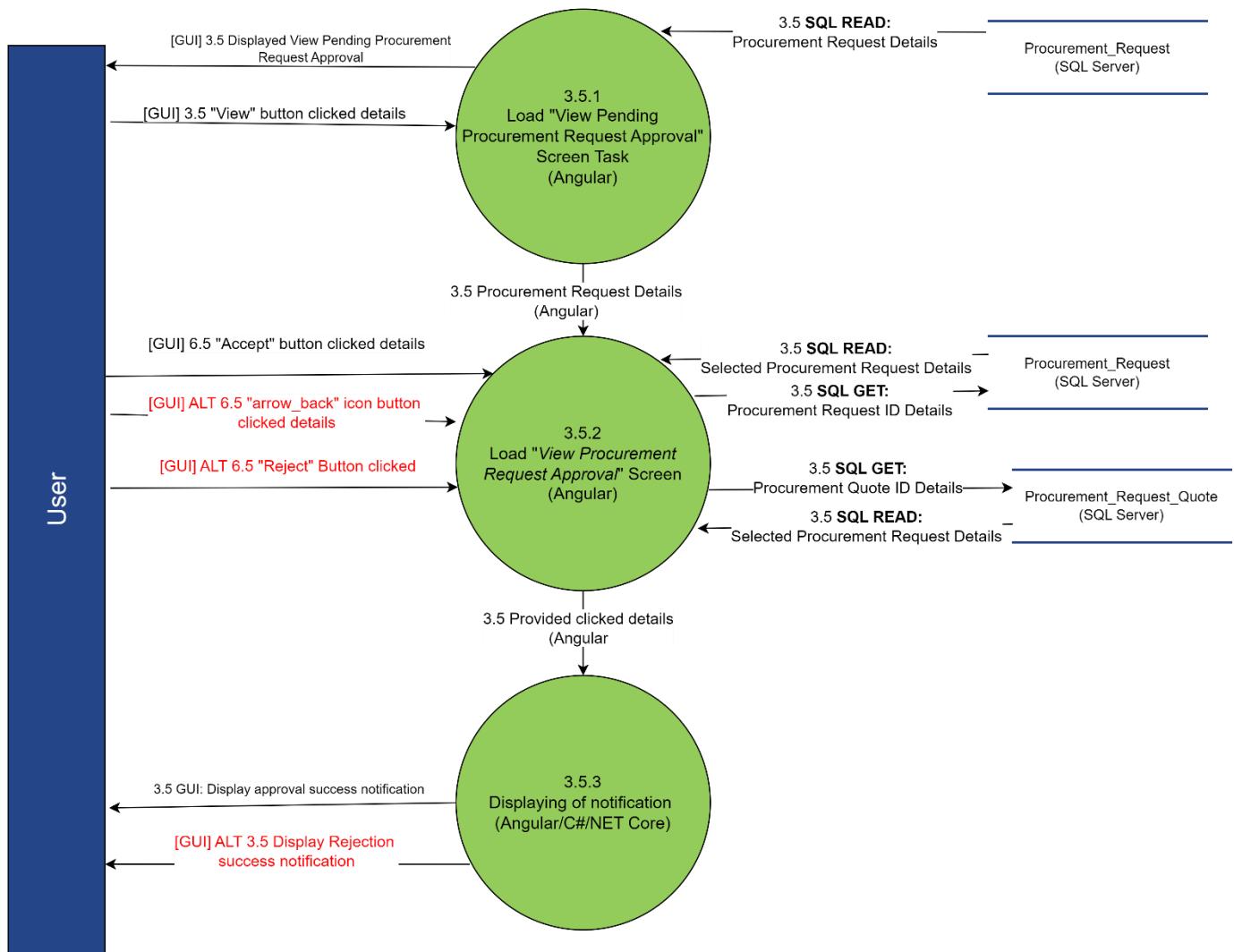


Table 213: 3.5 Approve Procurement Request

### 3.6 PLACE PROCUREMENT DETAILS

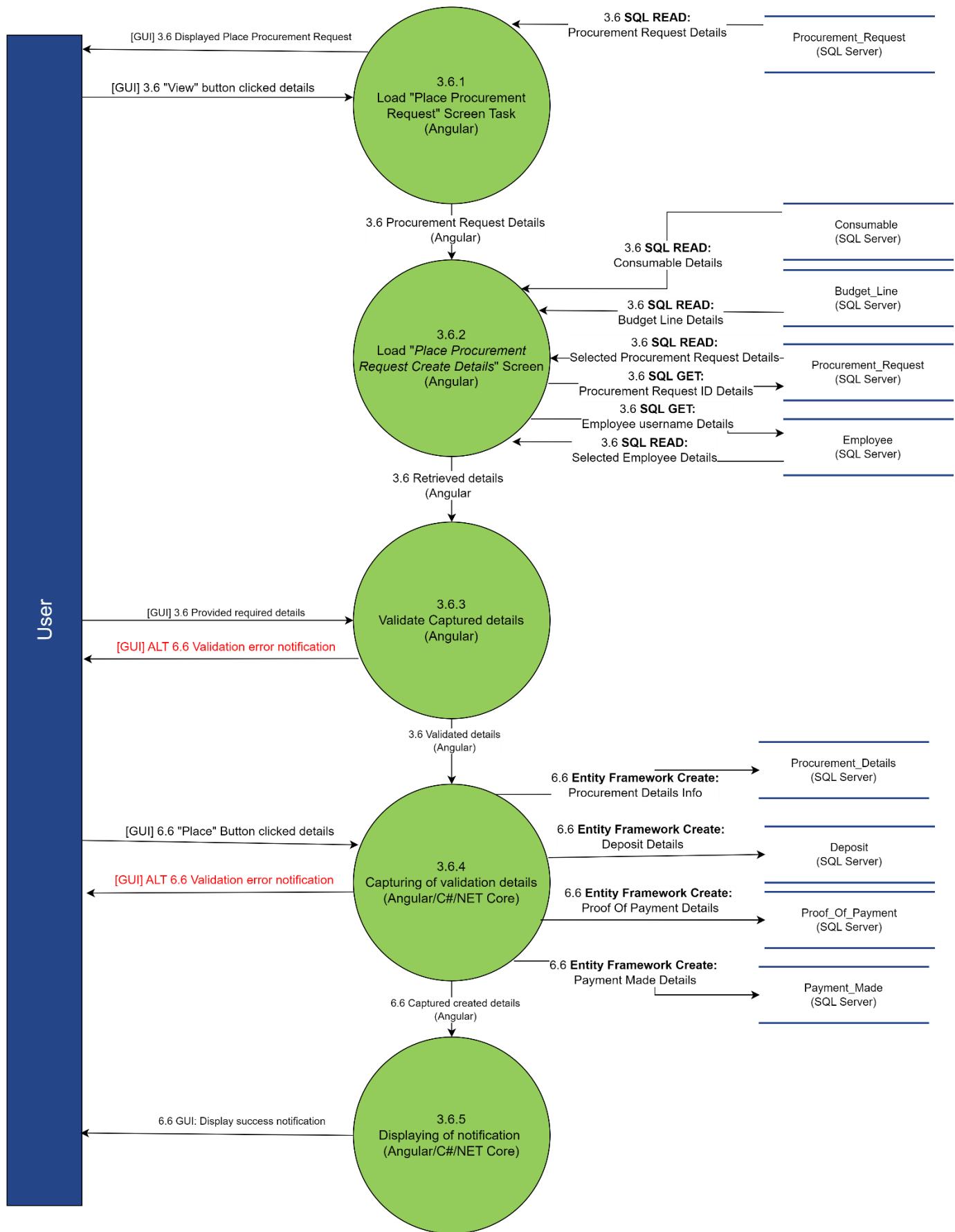


Table 214: 3.6 Place Procurement Details

### 3.7 VIEW FLAGGED PROCUREMENT DETAILS

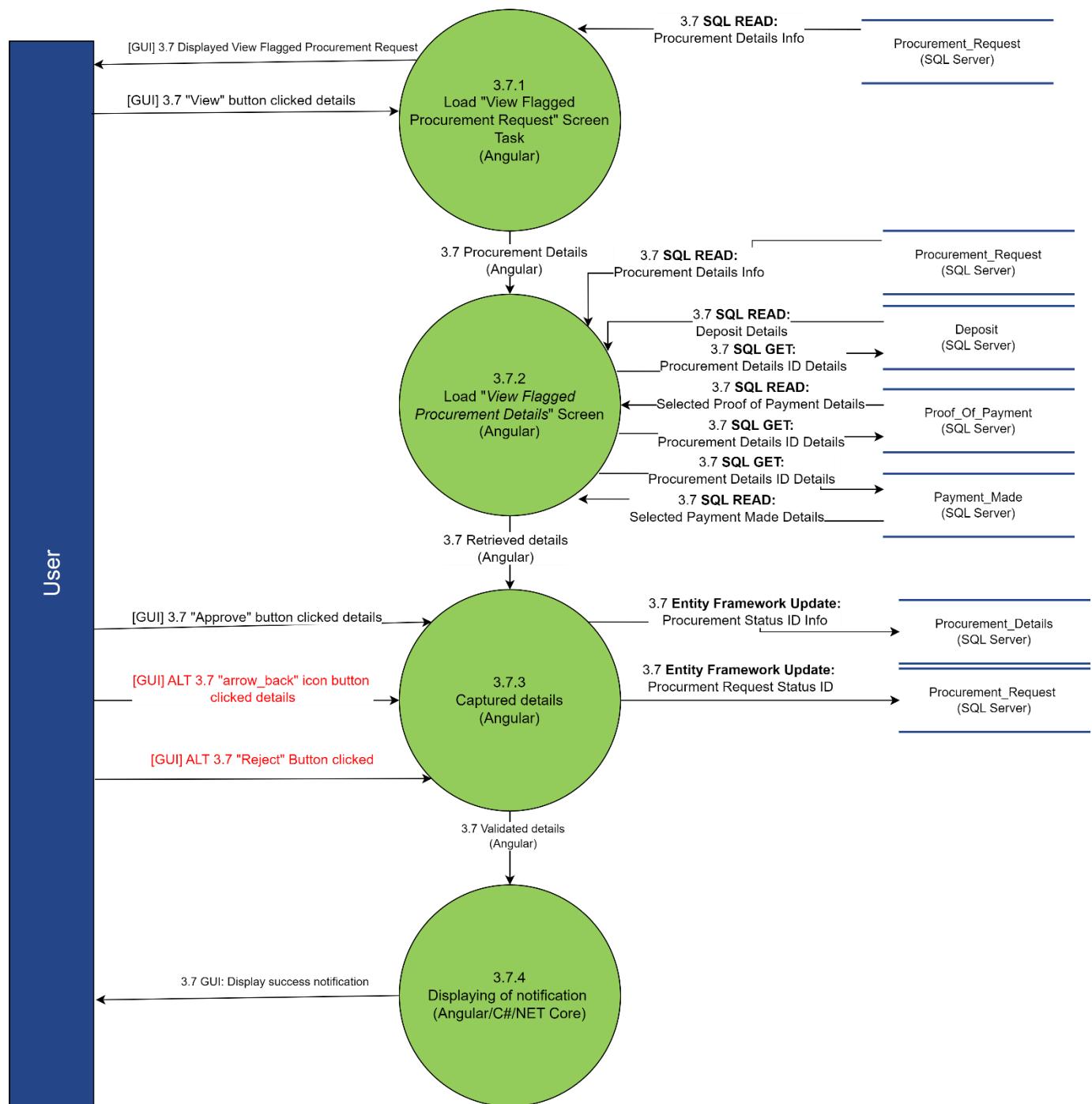


Table 215: 3.7 View Flagged Procurement Details

## 6.5 APPROVE ONBOARD REQUEST

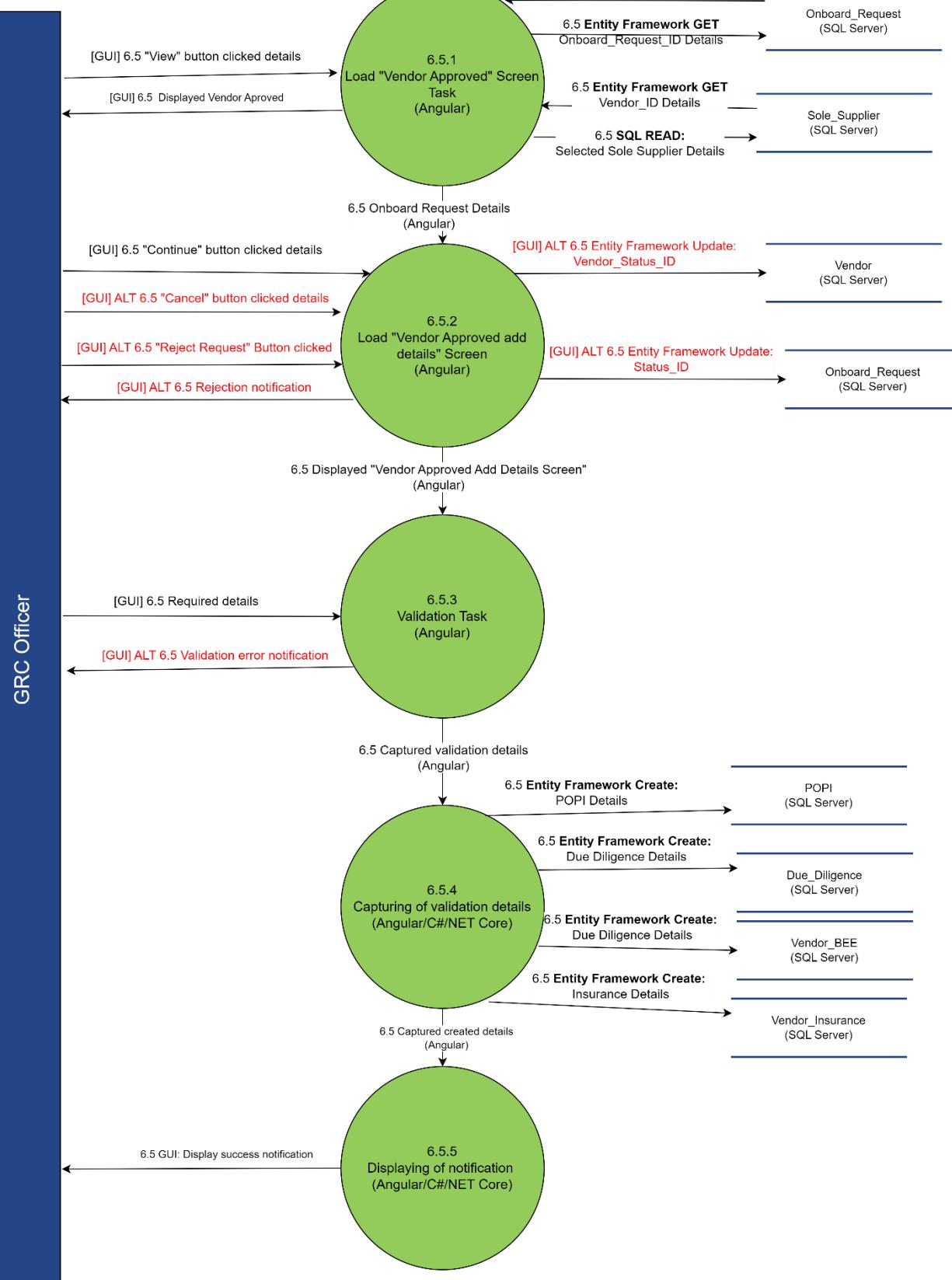


Table 216: 6.5 Approve Onboard Request

## 6.10 GENERATE DUE DILIGENCE CHECKLIST

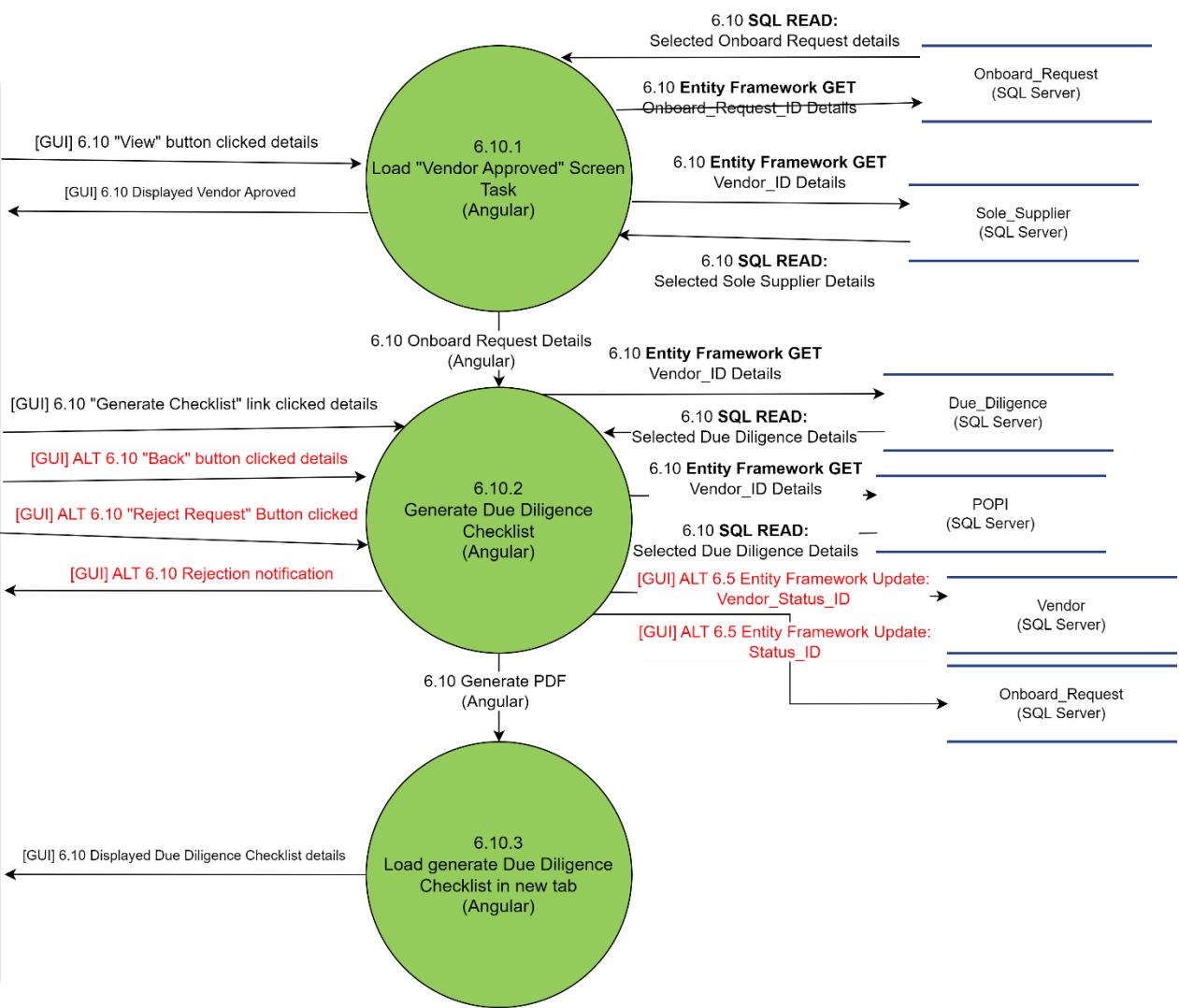


Table 217: 6.10 Generate Due Diligence Checklist

## 6.11 GENERATE SOLE SUPPLIER PERFORMANCE REVIEW NOTIFICATION

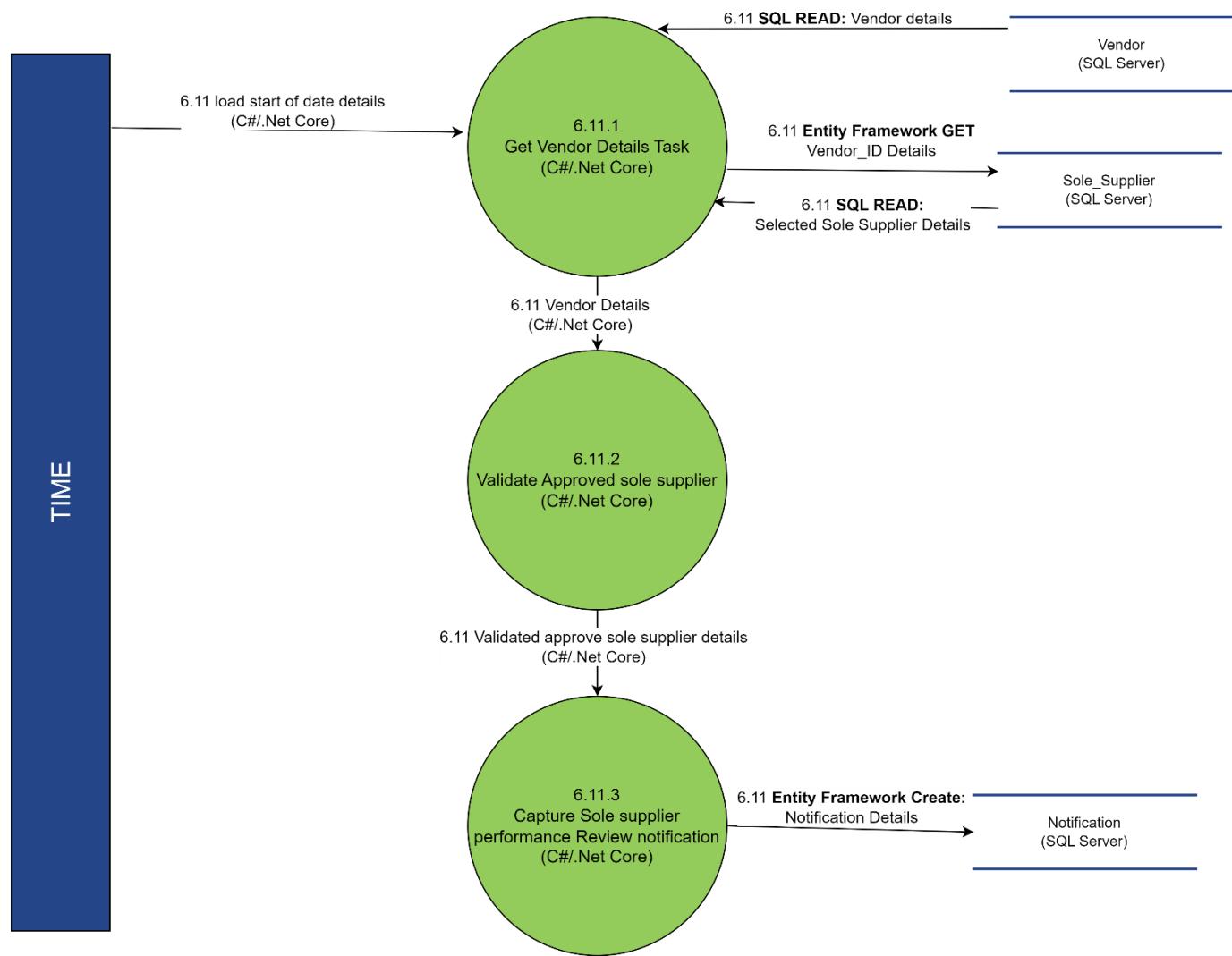


Table 218: 6.11 Generate Sole Supplier Performance Review Notification

### 6.13 GENERATE BEE EXPIRY NOTIFICATION

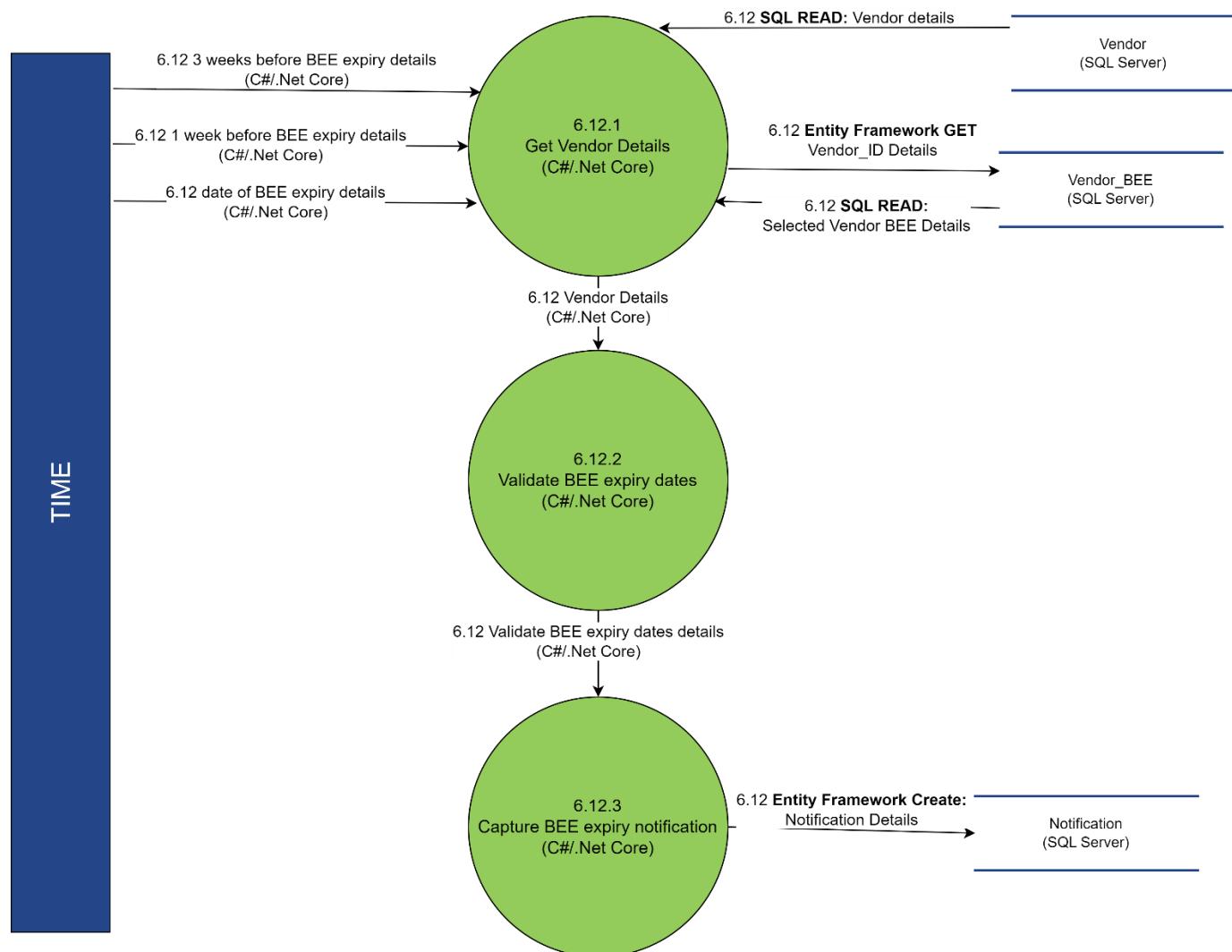


Table 219: 6.13 Generate BEE Expiry Notification

### 3.13 UPDATE APPROVE VENDOR ONBOARD REQUEST

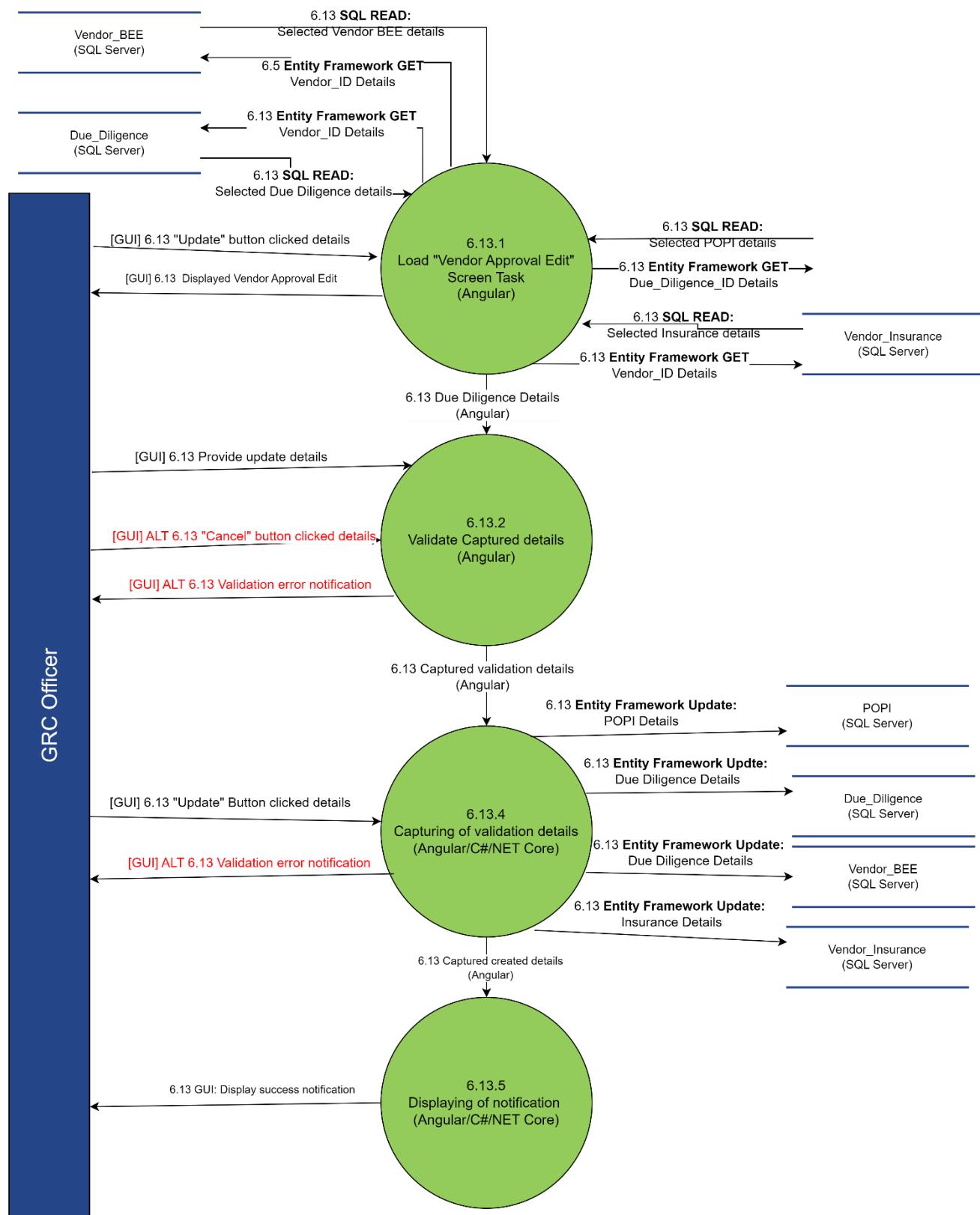


Table 220: 3.13 Update Approve Vendor Onboard Request

### 3.14 VIEW APPROVED OR PENDING ONBOARD REQUEST



Figure 197: 6.14 View Approved Or Pending Onboard Request

---

## 6.9 DELETE VENDOR

*Figure 198 6.9 Delete Vendor Tech Prim*

## 7.3 CONCLUSION

This Concludes the visual representation of the Technical Primitive level diagrams for the Procion system.

## 8. ACTIVITY DIAGRAM

### 8.1 INTRODUCTION

In this section we will display the visual representation of the Activity diagrams for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) respectively, of the Procion system.

### 8.2 ACTIVITY DIAGRAM

---

USE CASE 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35-2.36

---

#### 1.5 VIEW PROFILE

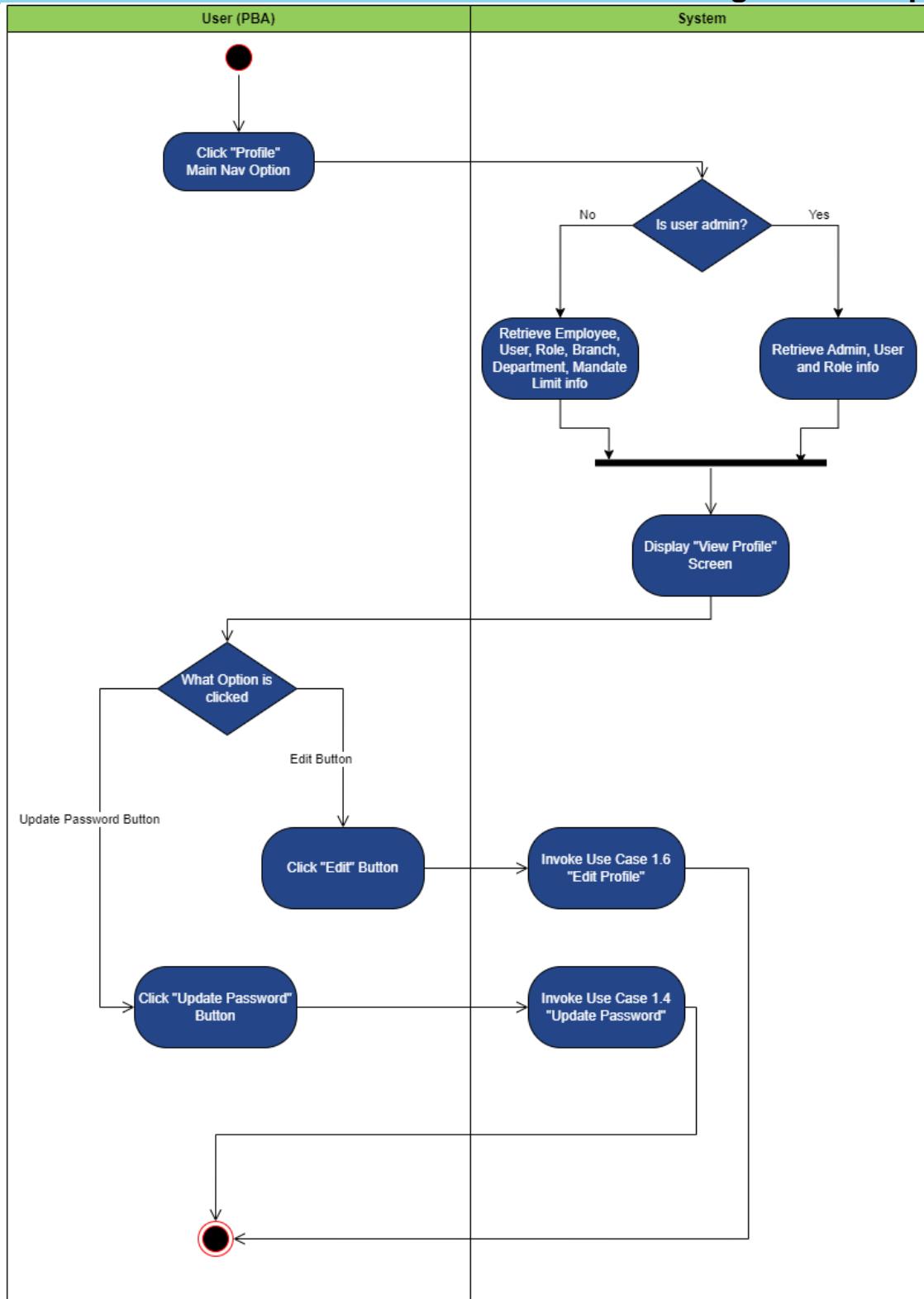


Figure 199 1.5 View Profile Activity Diagram

### 1.6 EDIT PROFILE

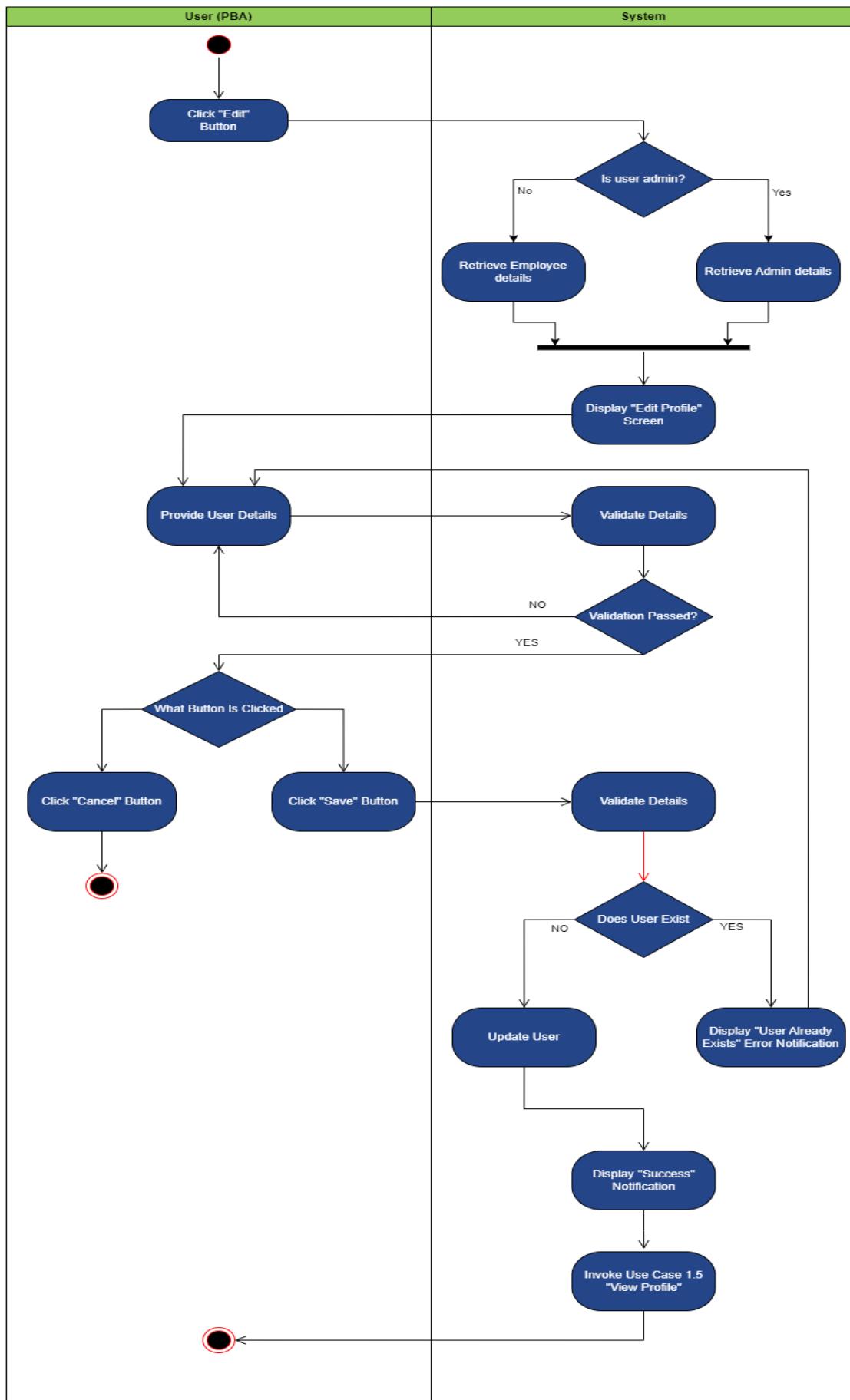


Figure 200 1.6 Edit Profile Activity Diagram

## 1.8 VIEW NOTIFICATIONS

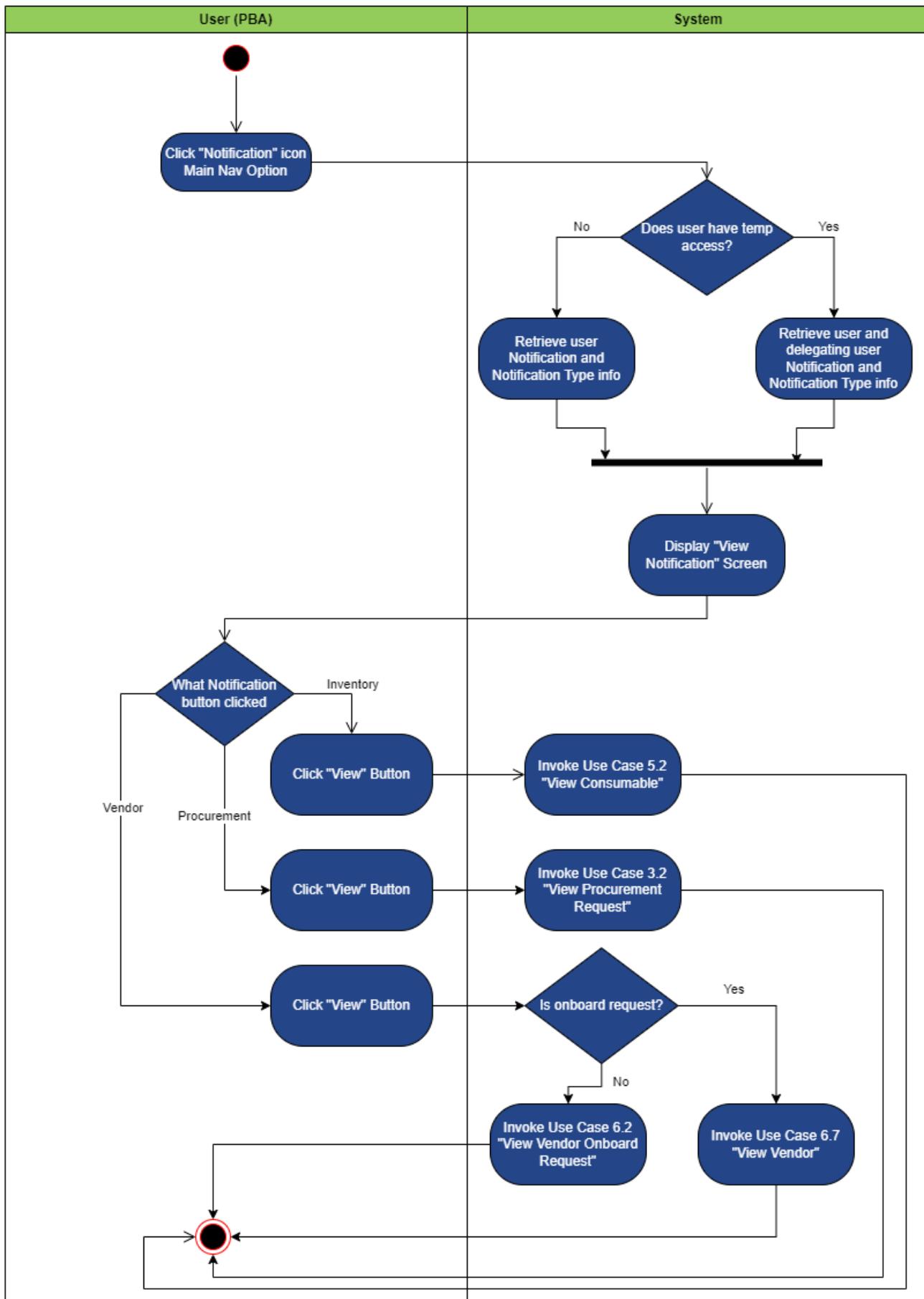


Figure 201 1.8 View Notifications Activity Diagram

## 2.17 VIEW DELEGATION OF AUTHORITY

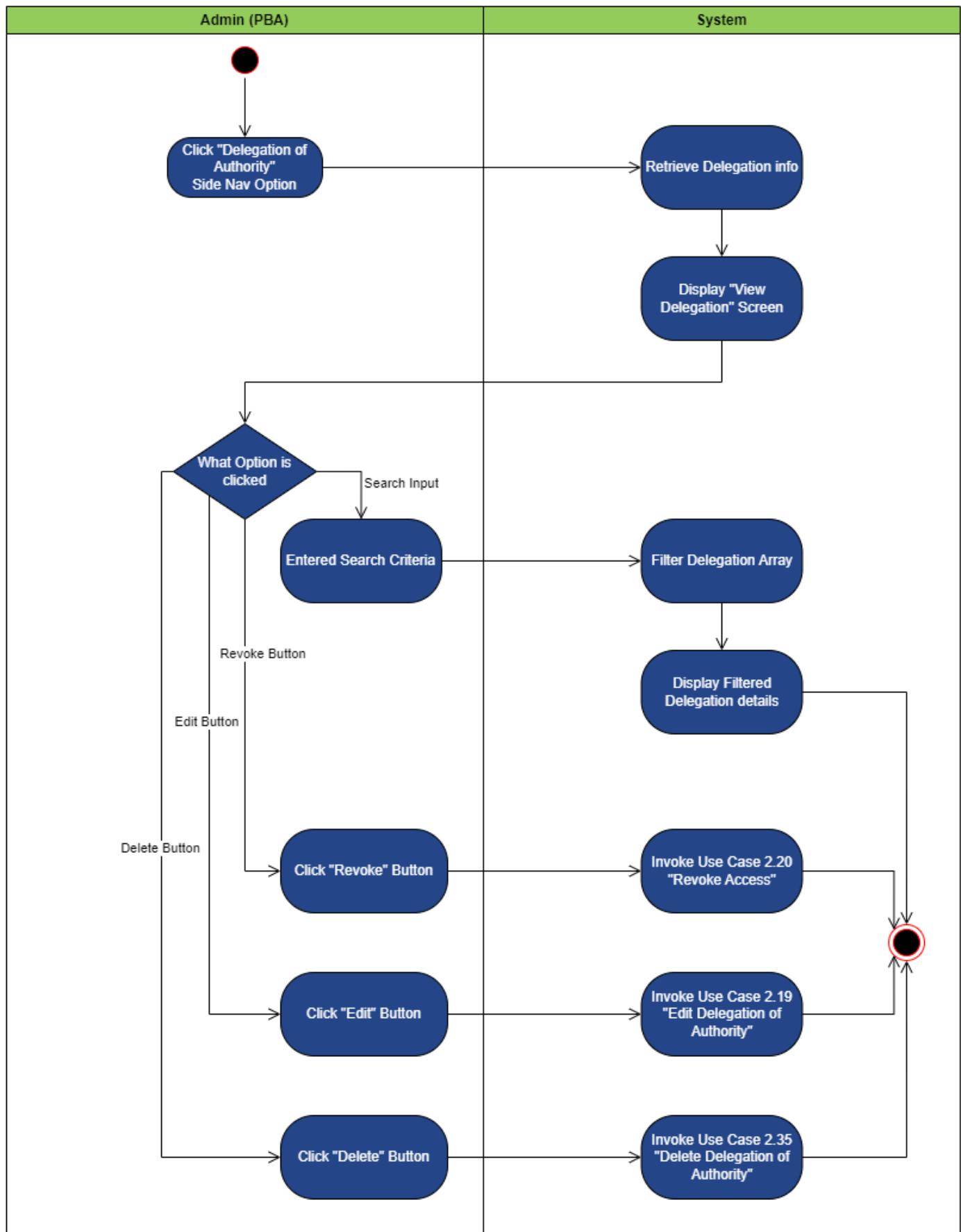


Figure 202 2.17 View Delegation of Authority Activity Diagram

## 2.18 ASSIGN DELEGATION OF AUTHORITY

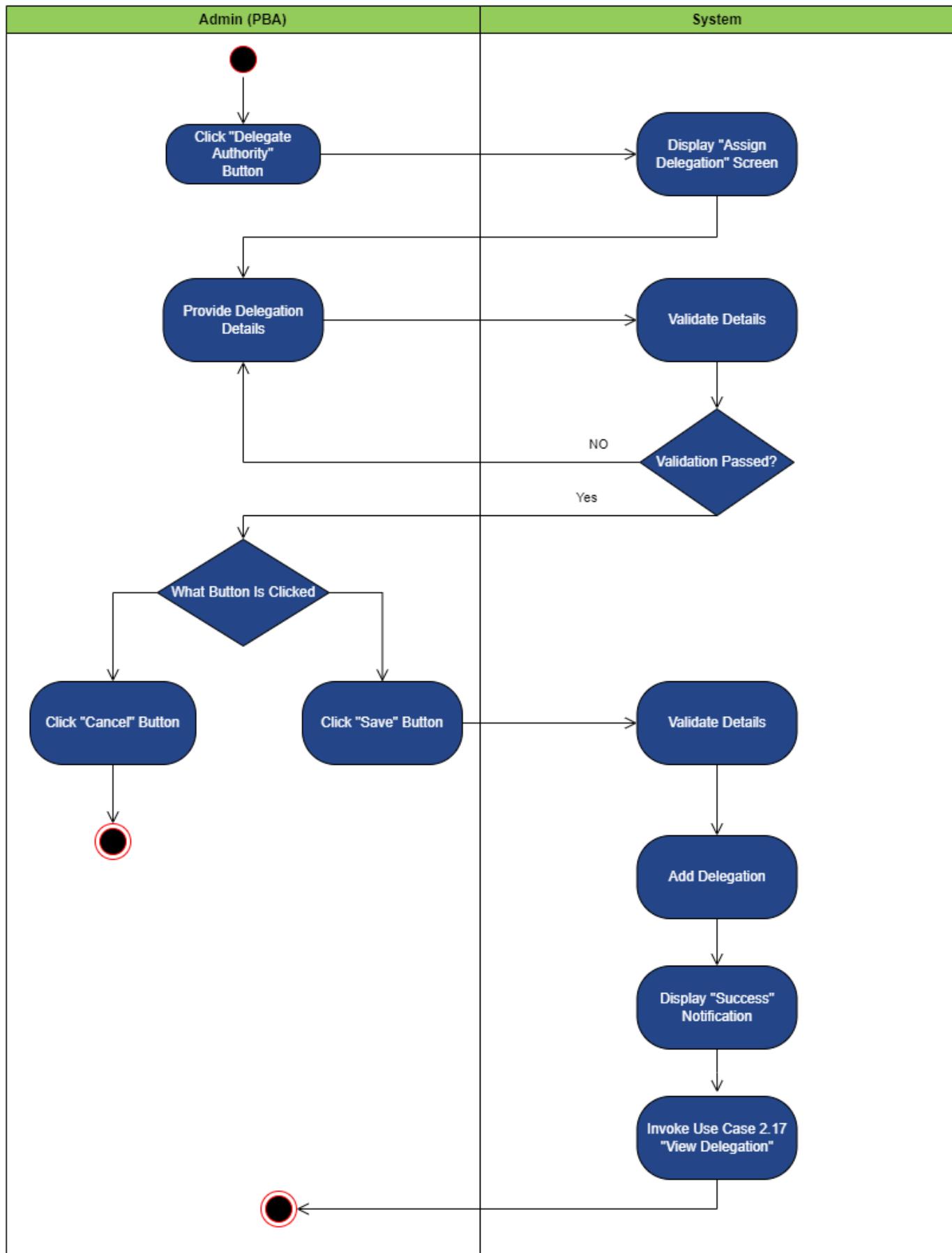


Figure 203 2.18 Assign Delegation of Authority Activity Diagram

## 2.19 EDIT DELEGATION OF AUTHORITY

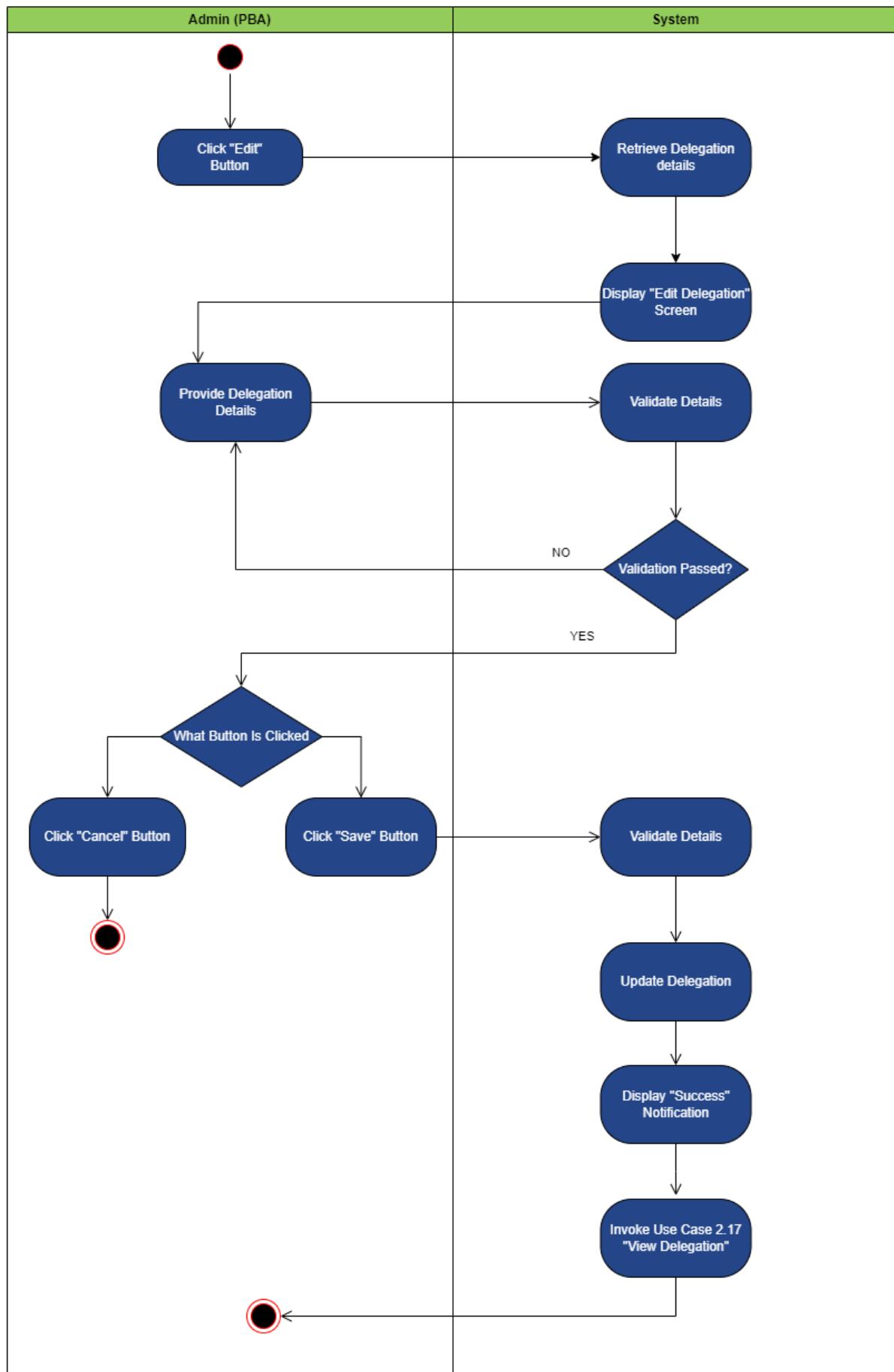


Figure 204 2.19 Edit Delegation of Authority Activity Diagram

## 2.20 REVOKE ACCESS

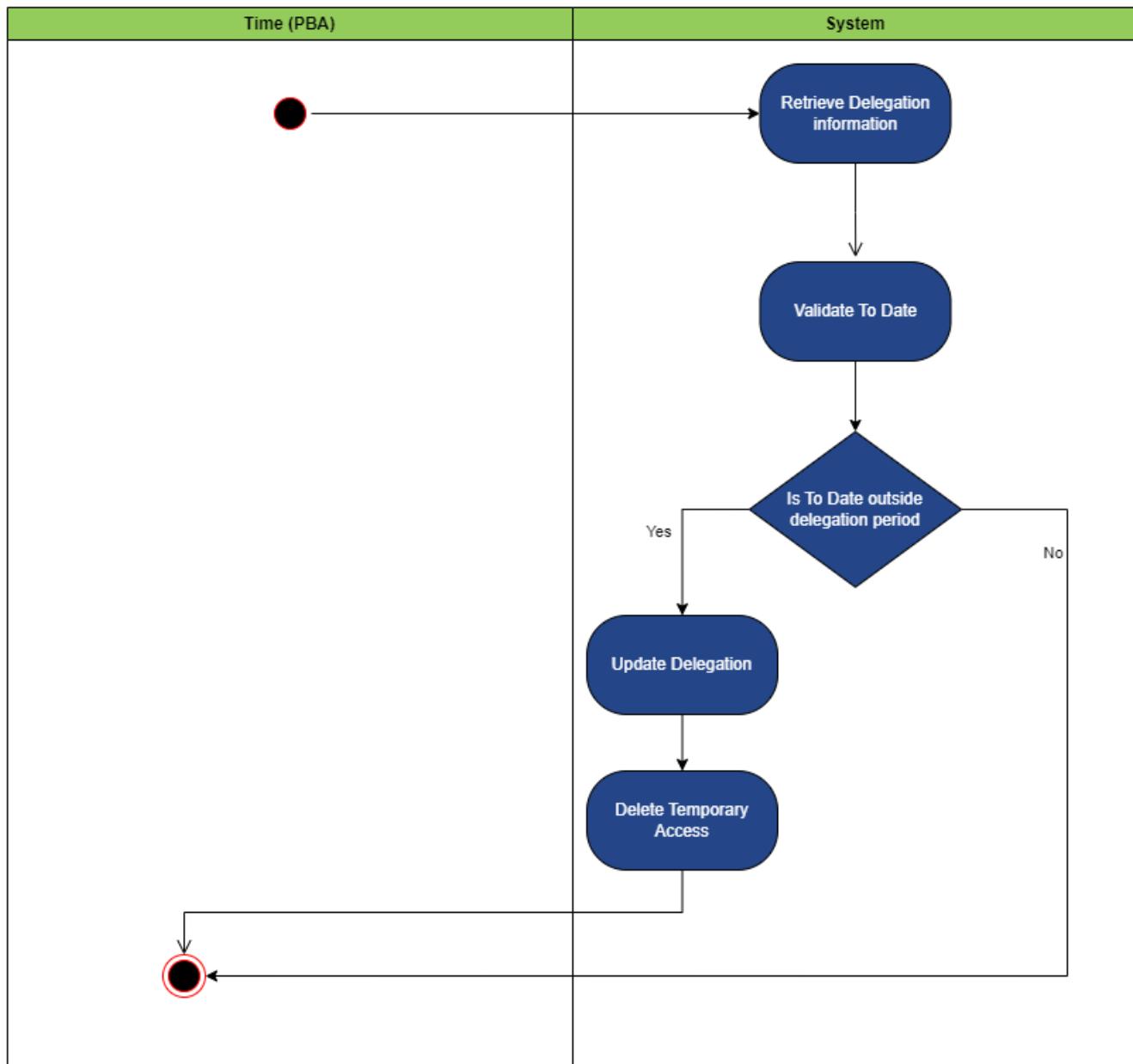


Figure 205 2.20 Revoke Access Activity Diagram

## 2.35 DELETE DELEGATION OF AUTHORITY

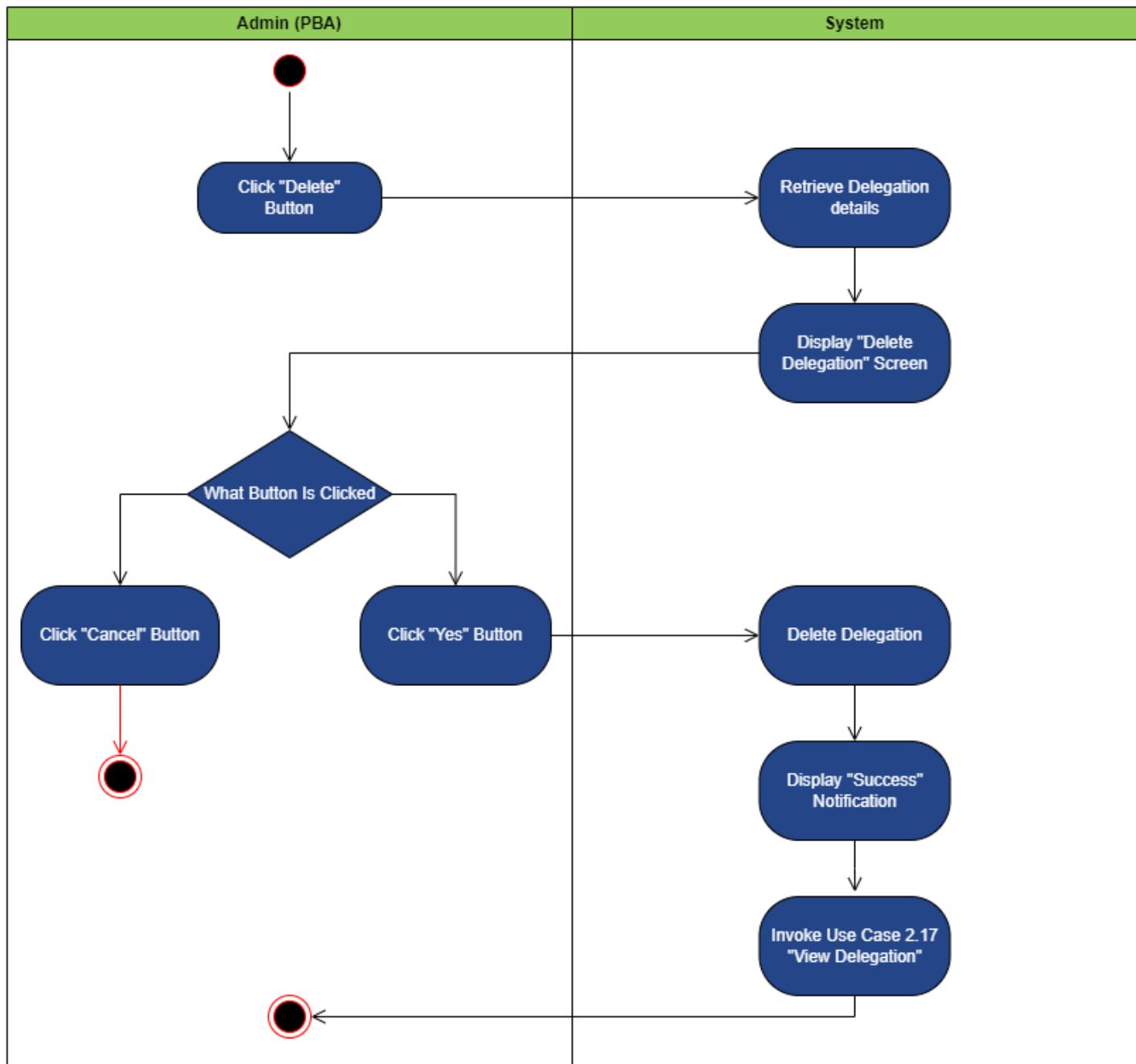
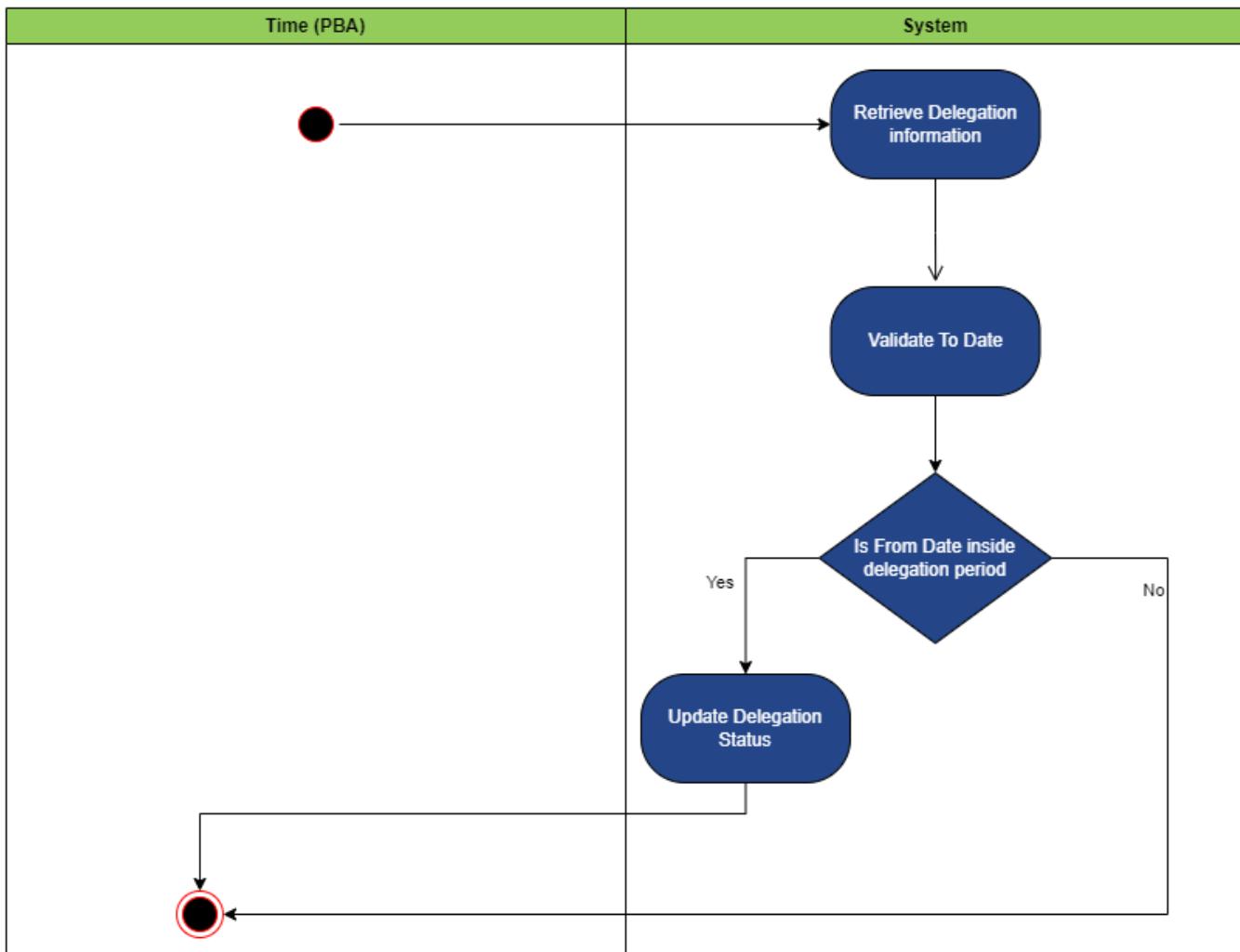


Figure 206 2.35 Delete Delegation of Authority Activity Diagram

**2.36 ACTIVATE DELEGATION OF AUTHORITY**

*Figure 207 2.36 Activate Delegation of Authority Activity Diagram*

USE CASE 1.7 &amp; 2.25 &amp; 2.27-2.30

## 1.7 VIEW USER MANUAL

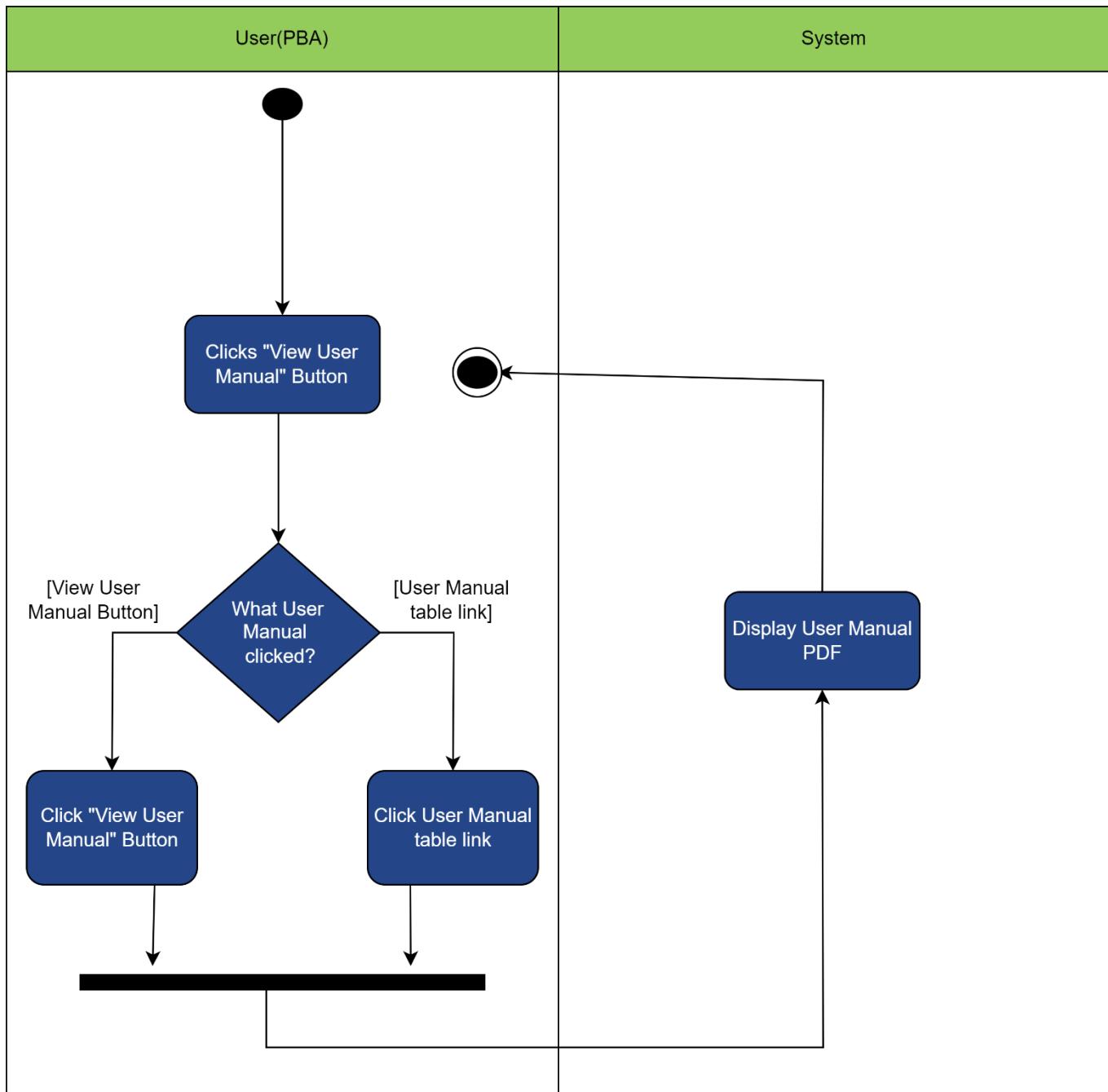


Figure 208 1.7 View User Manual Activity Diagram

## 2.25 BACKUP SYSTEM DATA

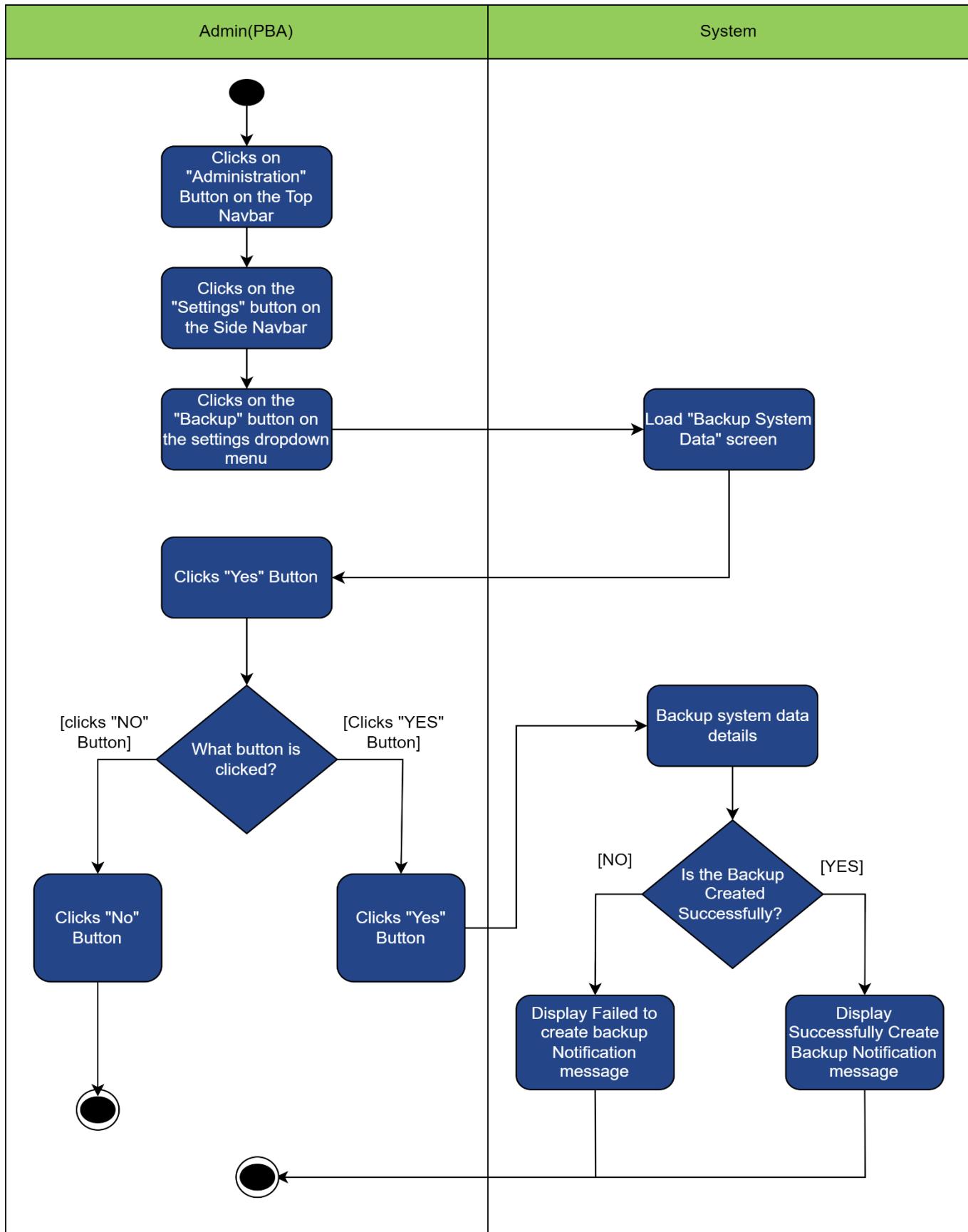


Figure 209 2.25 Backup System Data Activity Diagram

## 2.27 ADD HELP

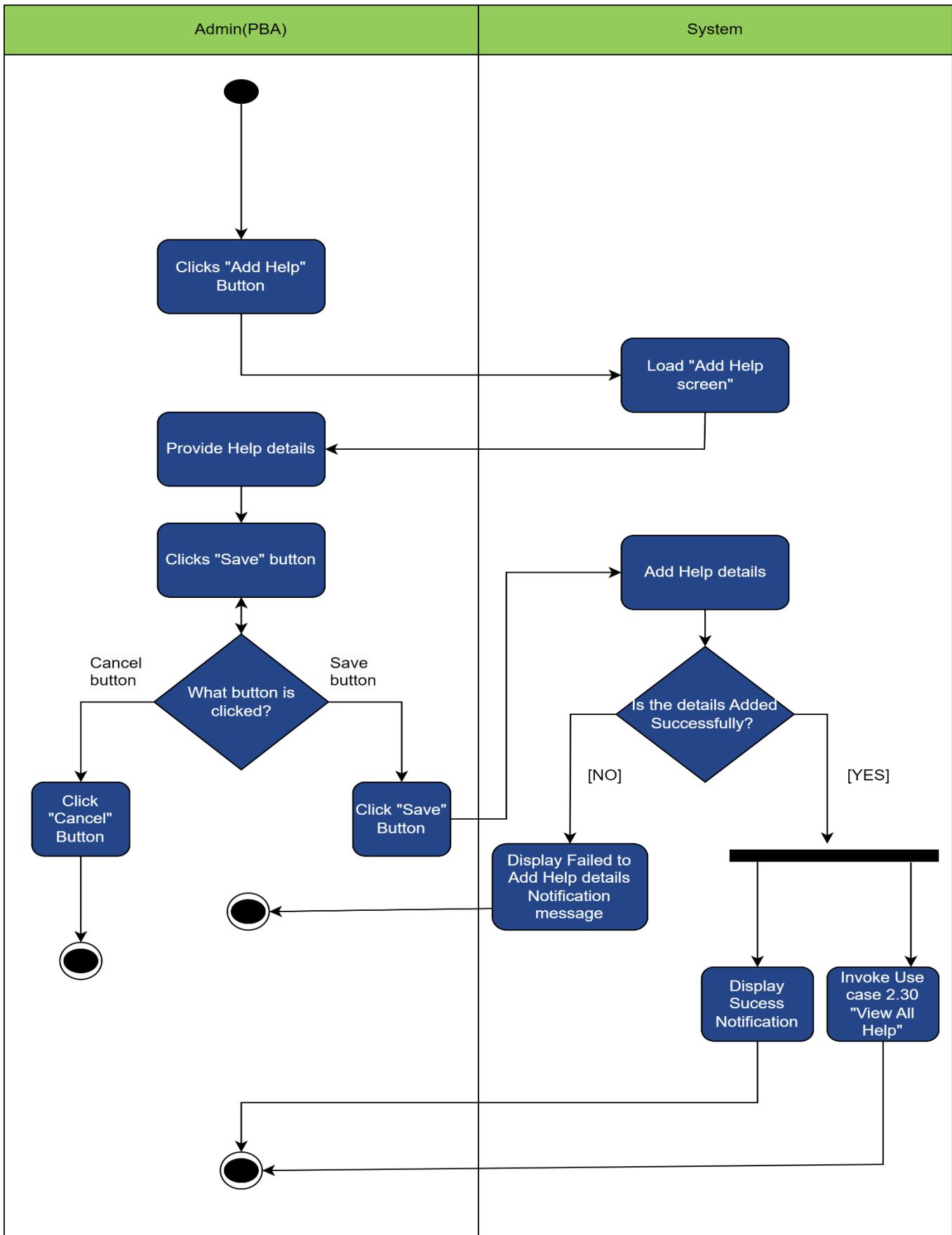


Figure 210 2.27 Add Help Activity Diagram

## 2.28 UPDATE HELP

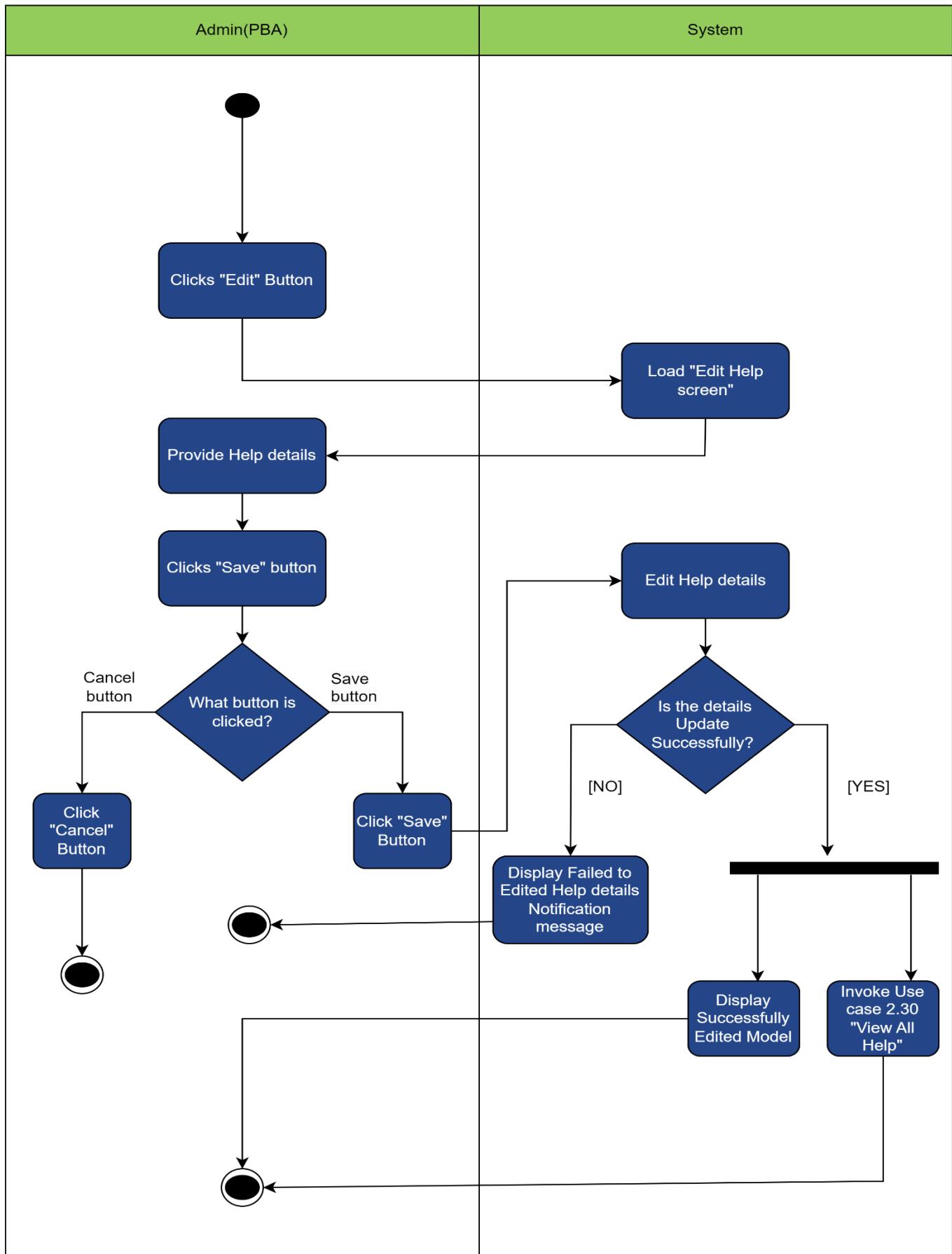


Figure 211 2.28 Update Help Activity Diagram

## 2.29 DELETE HELP

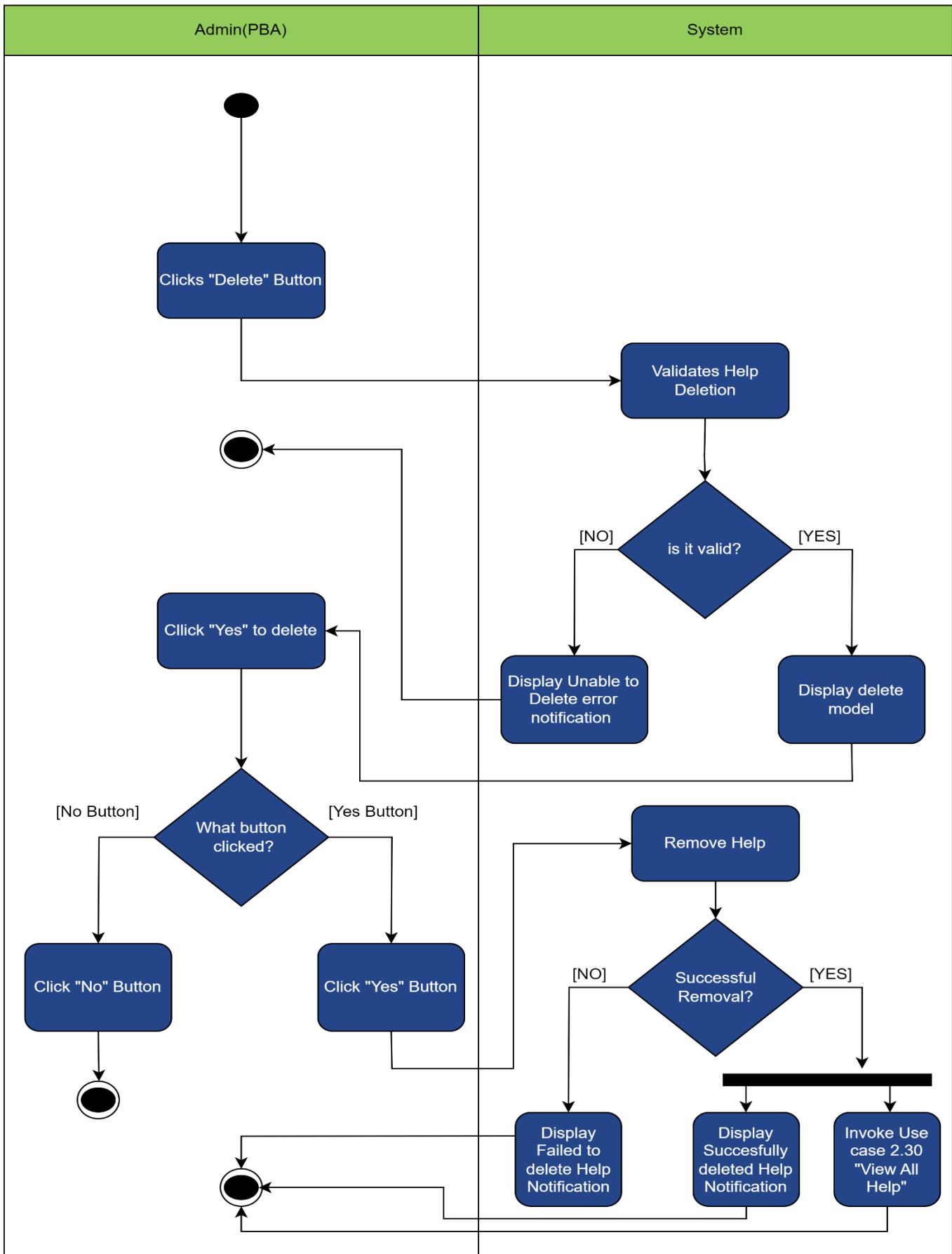


Figure 212 2.29 Delete Help Activity Diagram

## 2.30 VIEW ALL HELP

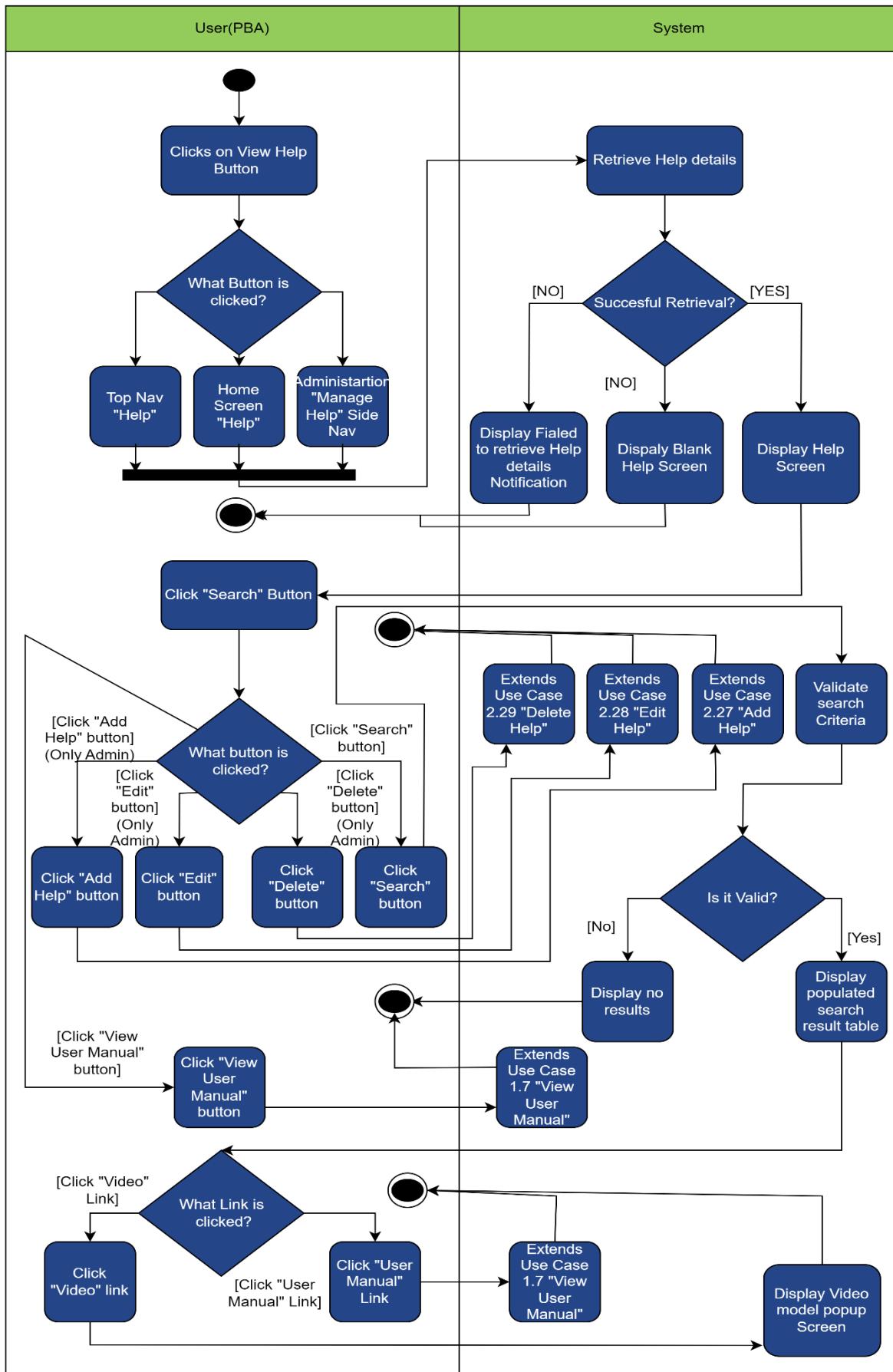


Figure 213 2.30 View All Help Activity Diagram

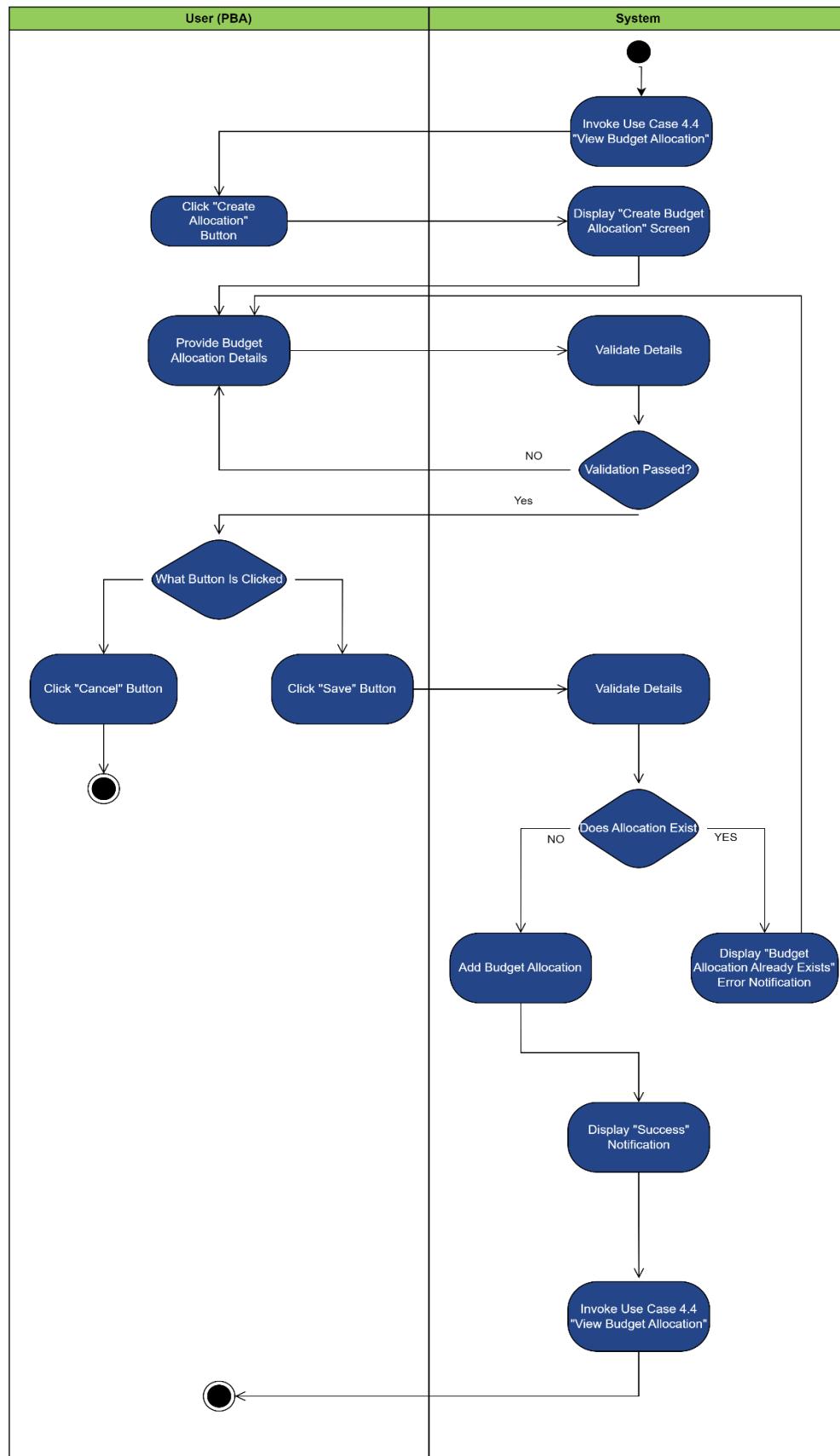
**USE CASE 4.3-4.6 & 4.1-4.2 & 3.8-3.12**
**4.3 CREATE BUDGET ALLOCATION**


Figure 214 4.3 Create Budget allocation Activity Diagram

#### 4.4 VIEW BUDGET ALLOCATION

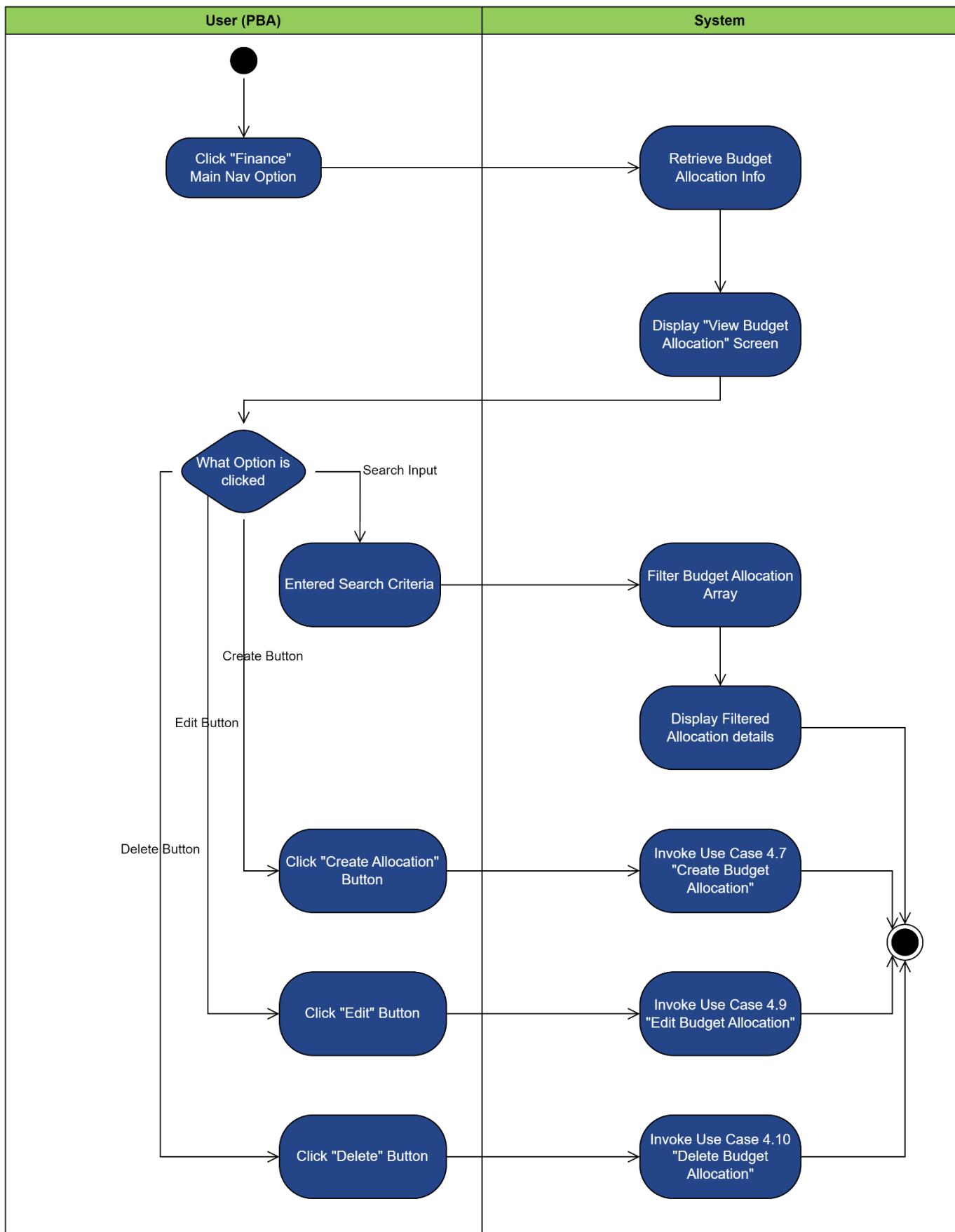


Figure 215 4.4 View Budget Allocation Activity Diagram

## 4.5 UPDATE BUDGET ALLOCATION

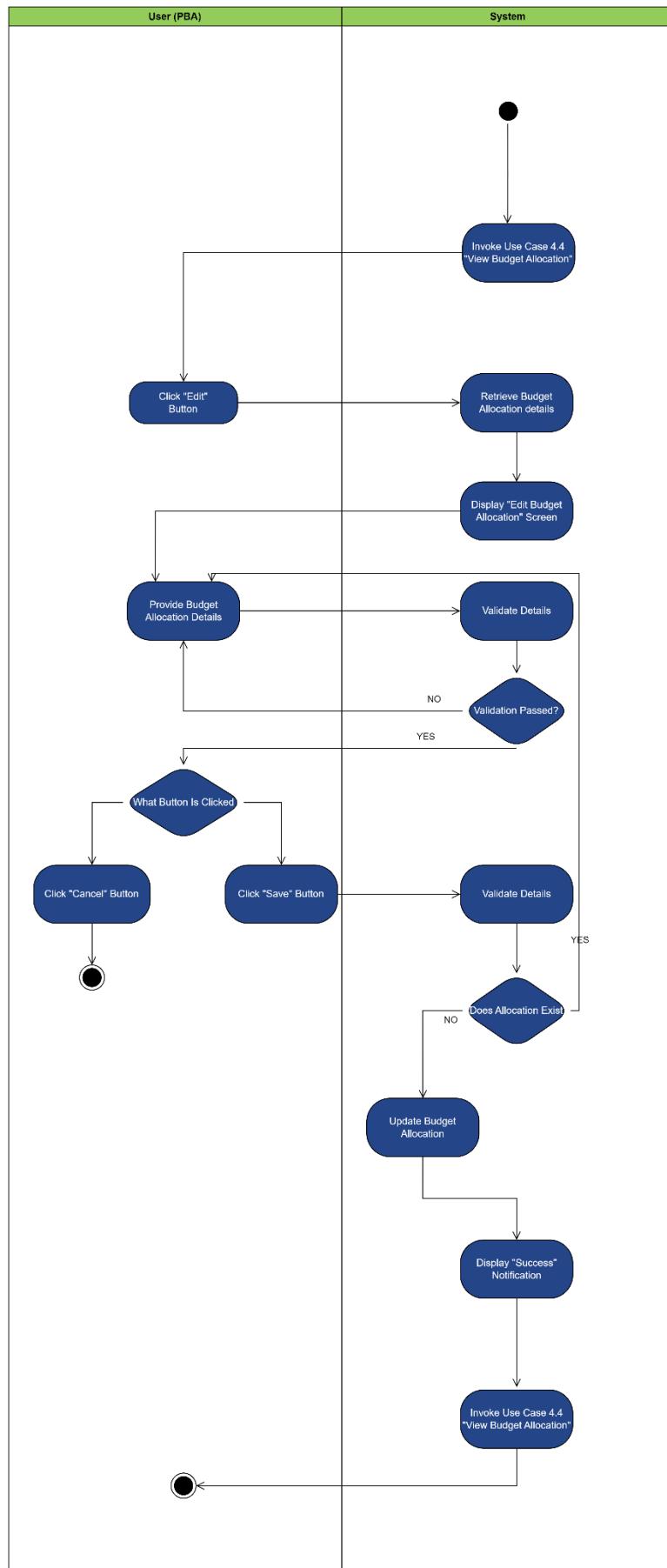


Figure 216 4.5 Update Budget Allocation Activity Diagram

## 4.6 DELETE BUDGET ALLOCATION

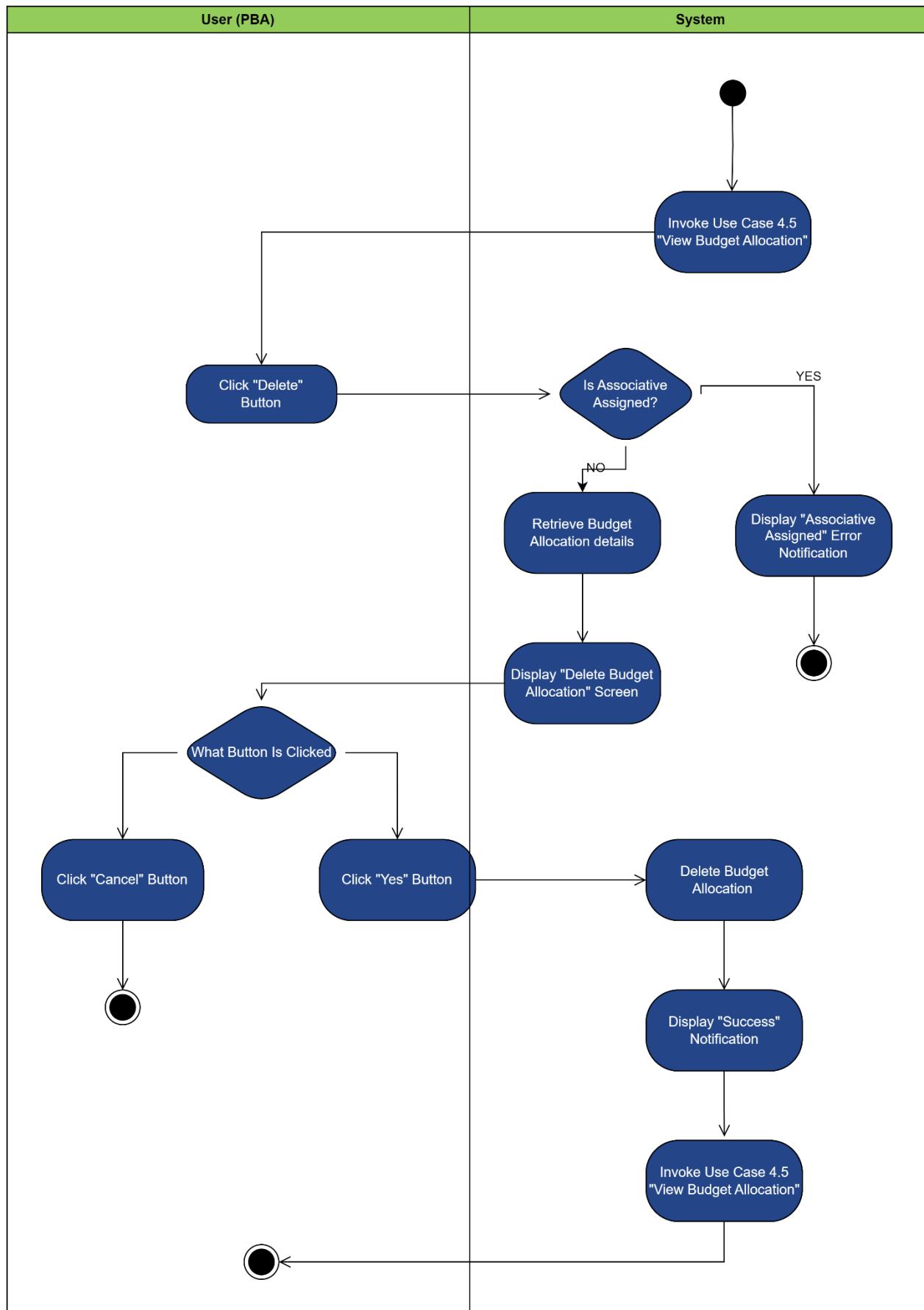
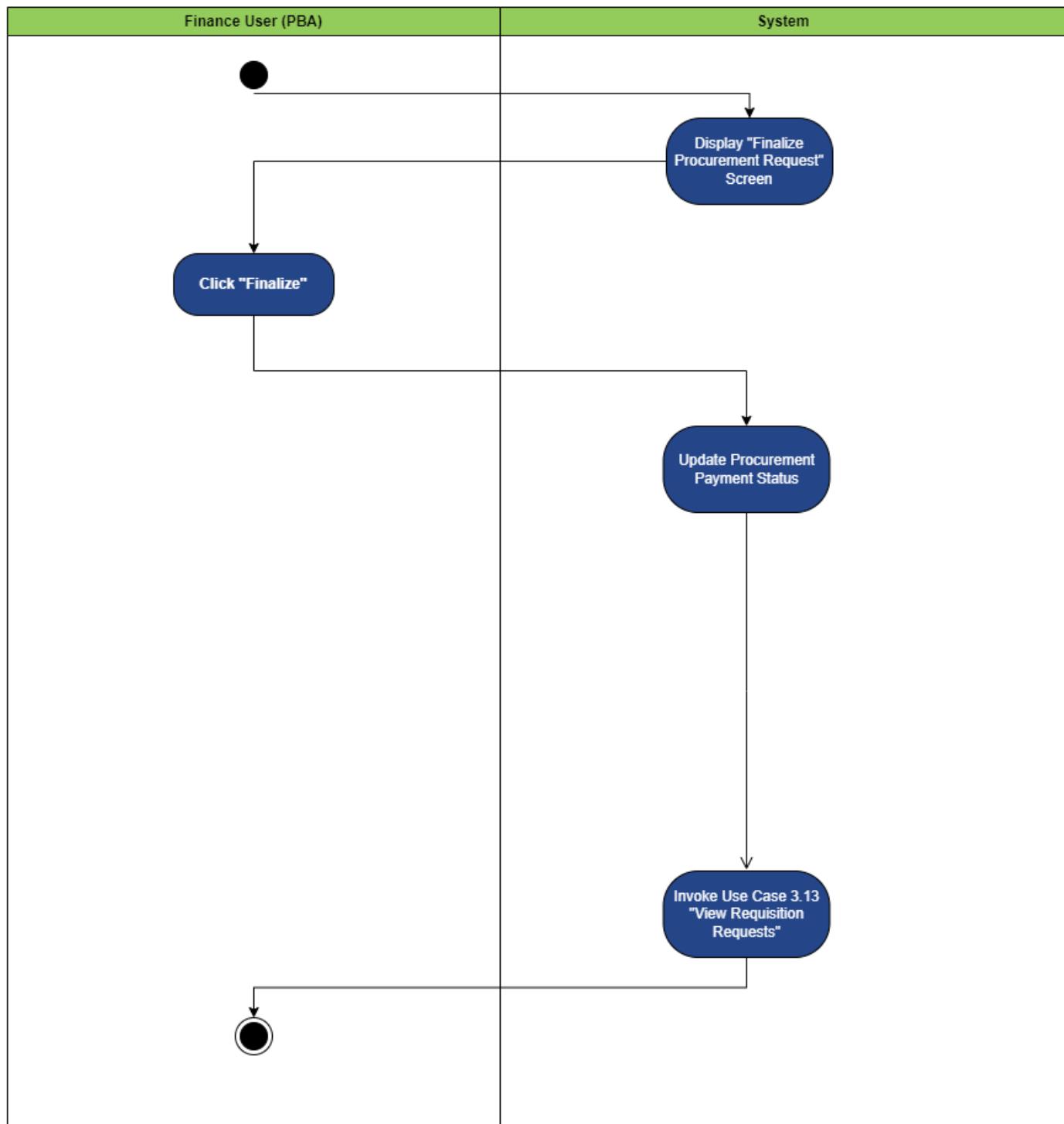


Figure 217 4.6 Delete Budget Allocation Activity Diagram

**4.1 FINALIZE PROCUREMENT REQUEST**

*Figure 218 4.1 Finalize Procurement Request Activity Diagram*

## 4.2 UPLOAD PROOF OF PAYMENT

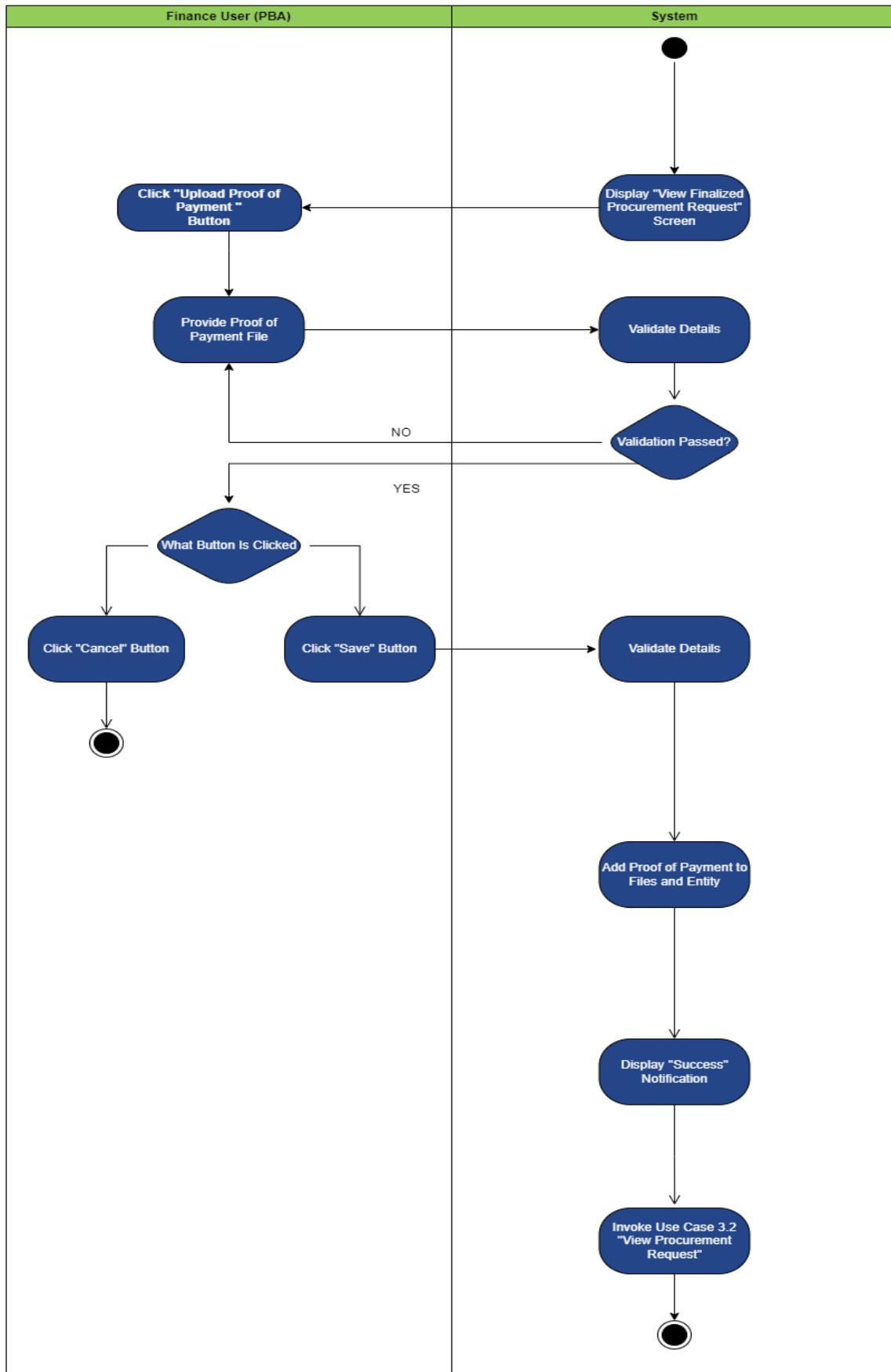


Figure 219 4.2 Upload Proof Of Payment Activity Diagram

### 3.8 RECEIVE PROCUREMENT ITEM

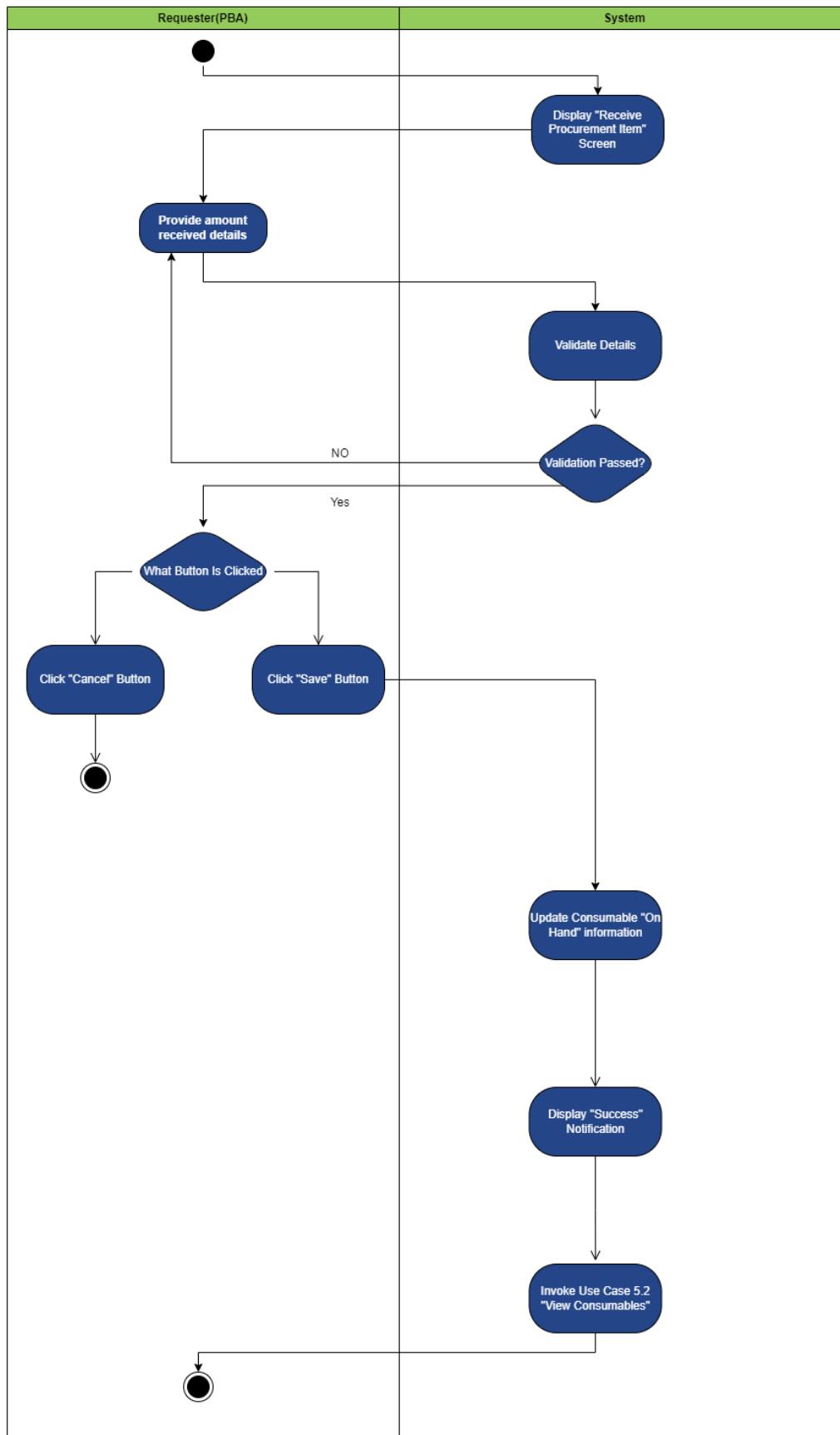


Figure 220 3.8 Receive Procurement Item Activity Diagram

### 3.9 UPLOAD TAX INVOICE

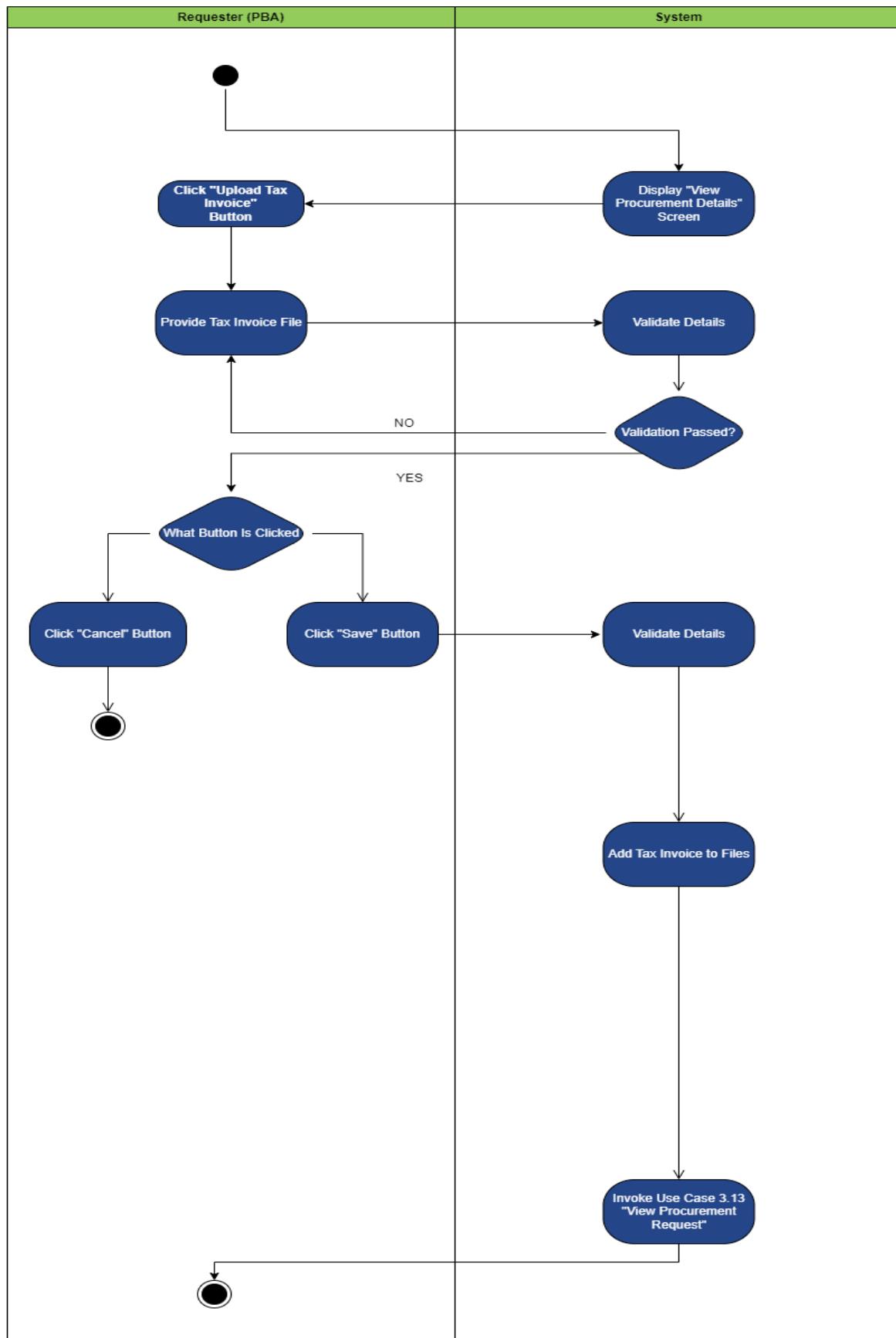


Figure 221 3.9 Upload Tax Invoice Activity Diagram

### **3.10 BUDGET OWNER REQUISITION APPROVAL**

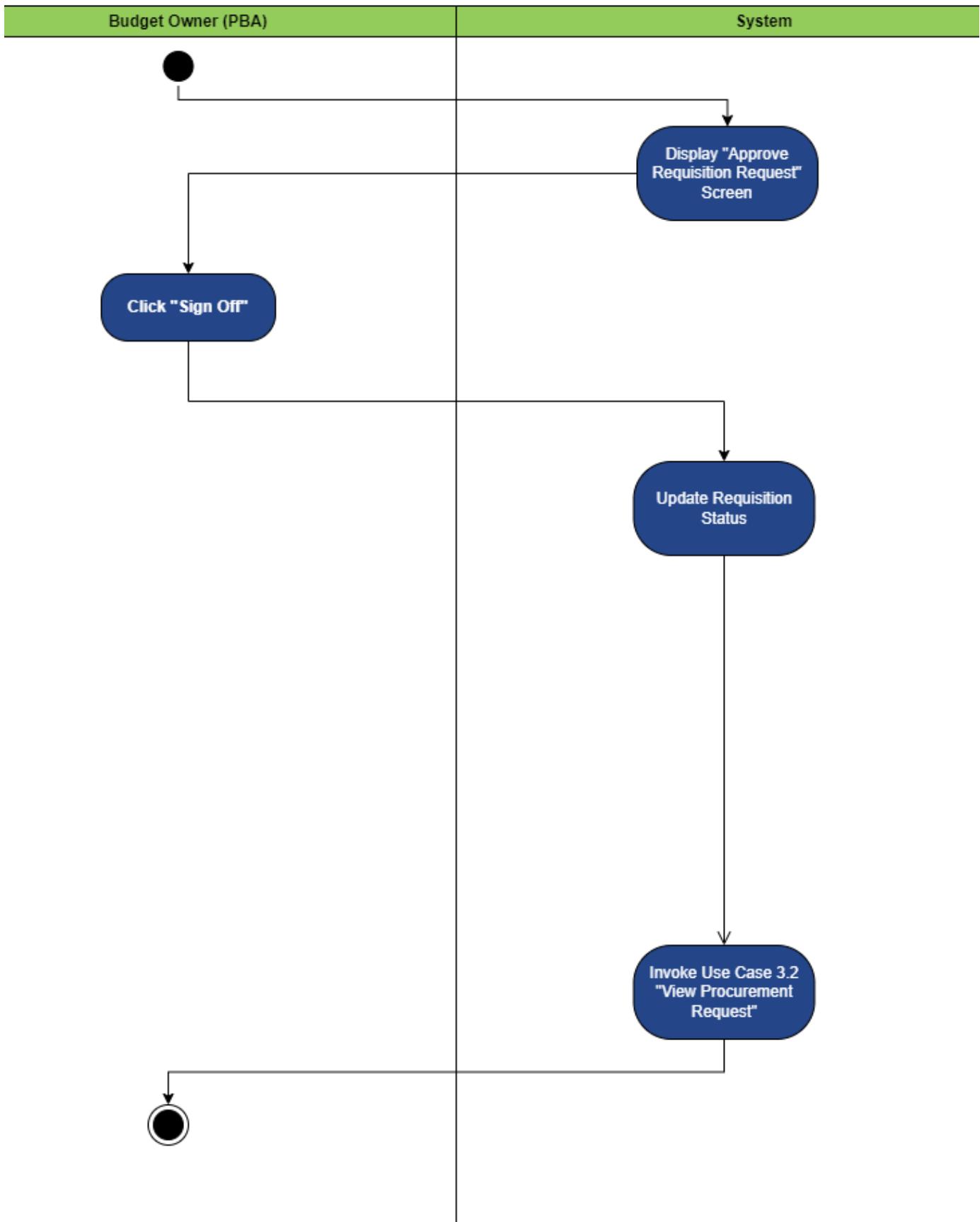


Figure 222 3.10 budget Owner Requisition Approval Activity Diagram

### 3.11 UPLOAD RECEIPT

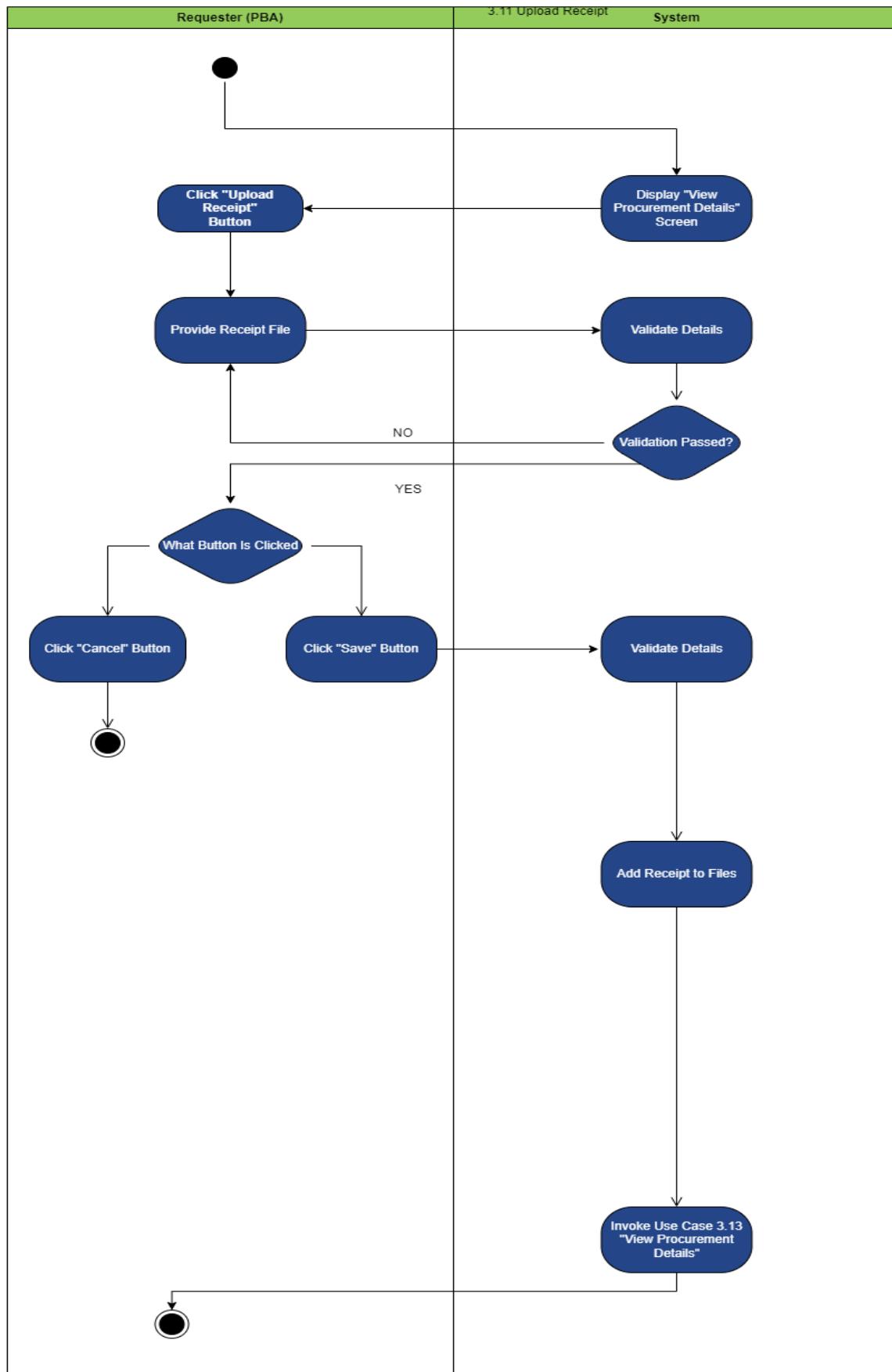


Figure 223 3.11 Upload receipt Activity Diagram

### 3.12 UPLOAD CREDIT CARD PAYMENT

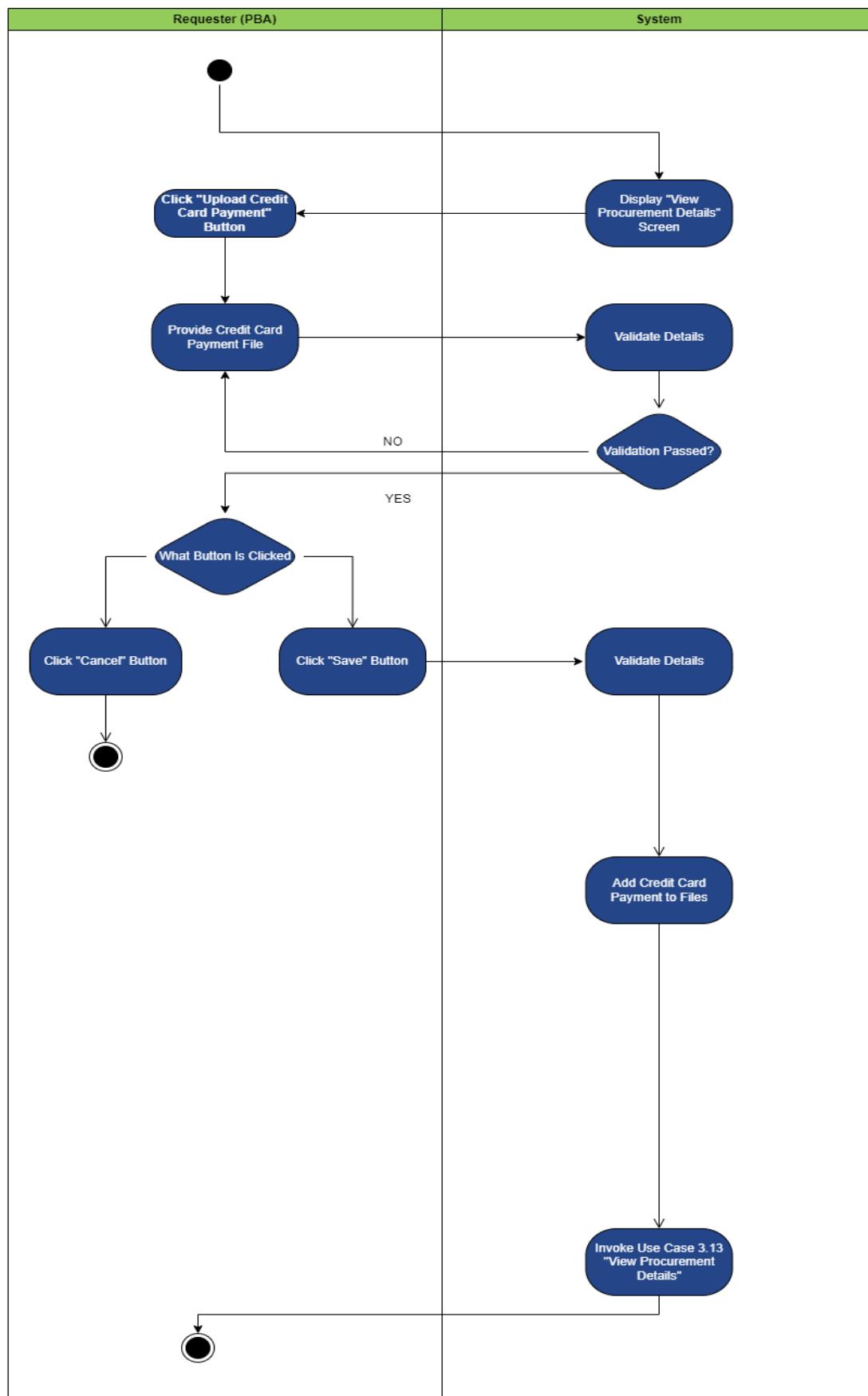


Figure 224 3.12 Upload Credit Card payment Diagram

USE CASE 5.1-5.8 &amp; 5.9 , 5.10 &amp; 1.1-1.4 &amp; 3.1-3.4

## 5.1 CREATE CONSUMABLE

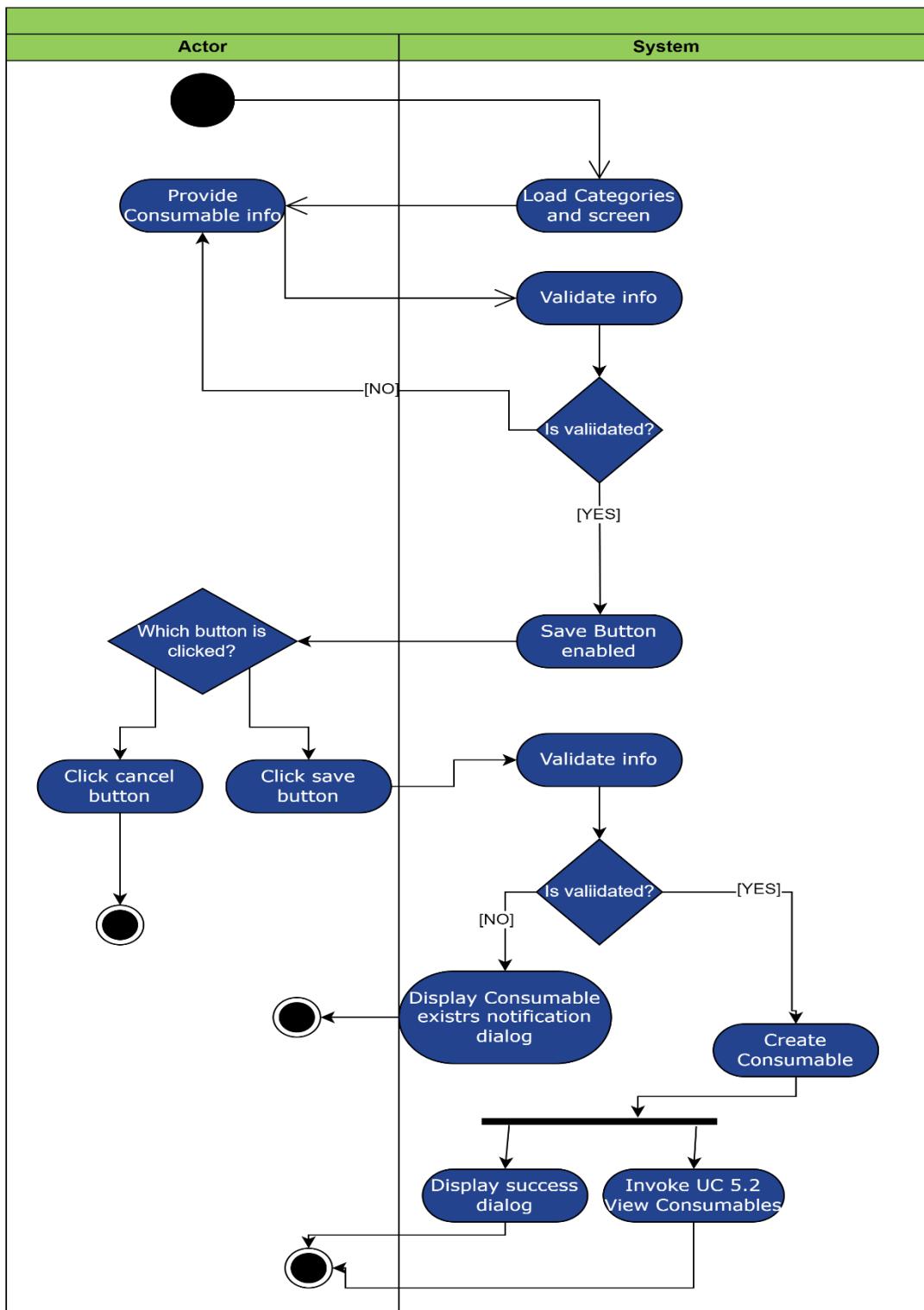


Figure 225 5.1 Create Consumable Activity Diagram

## 5.2 VIEW CONSUMABLES

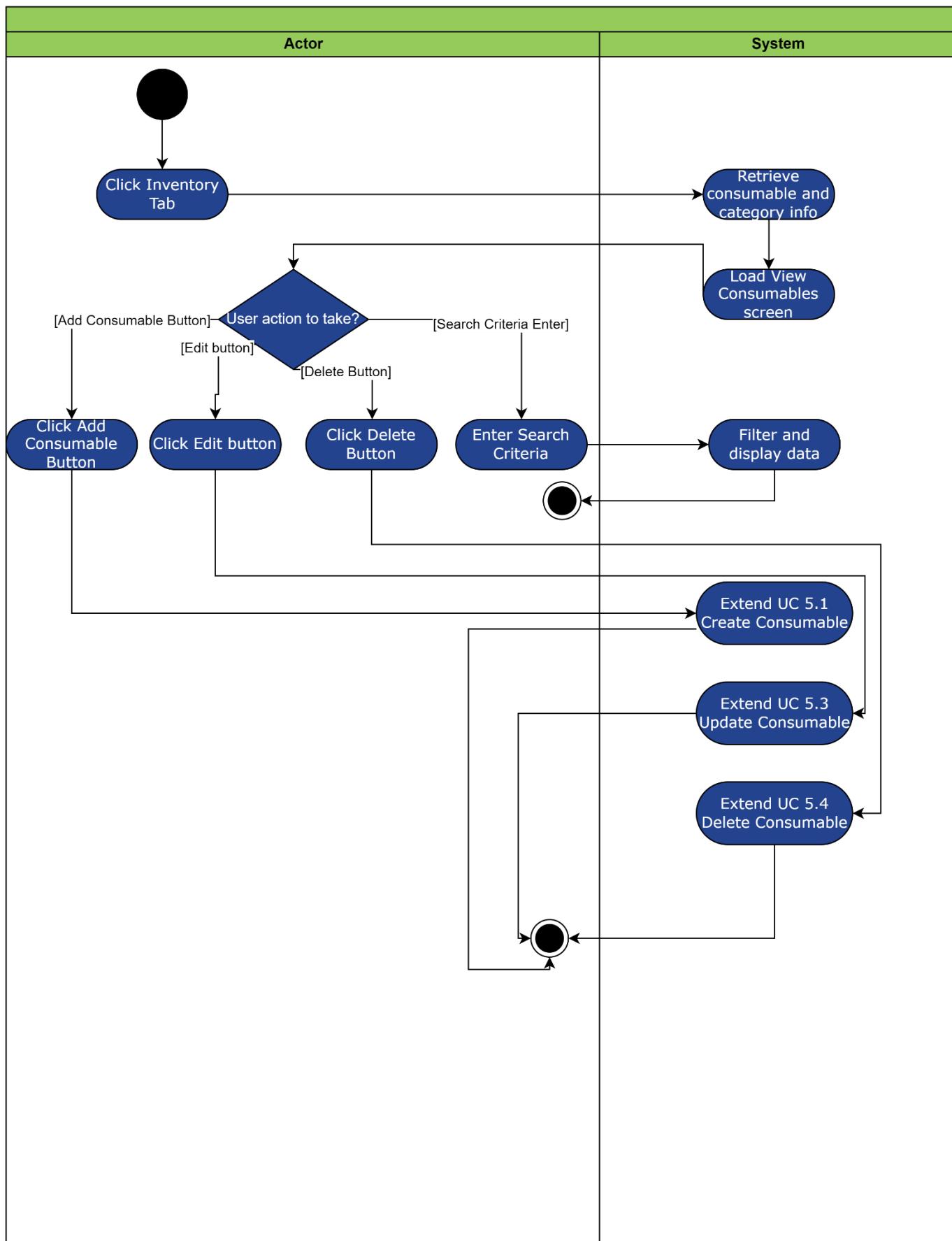


Figure 226 5.2 View Consumable Activity Diagram

### 5.3 UPDATE CONSUMABLE

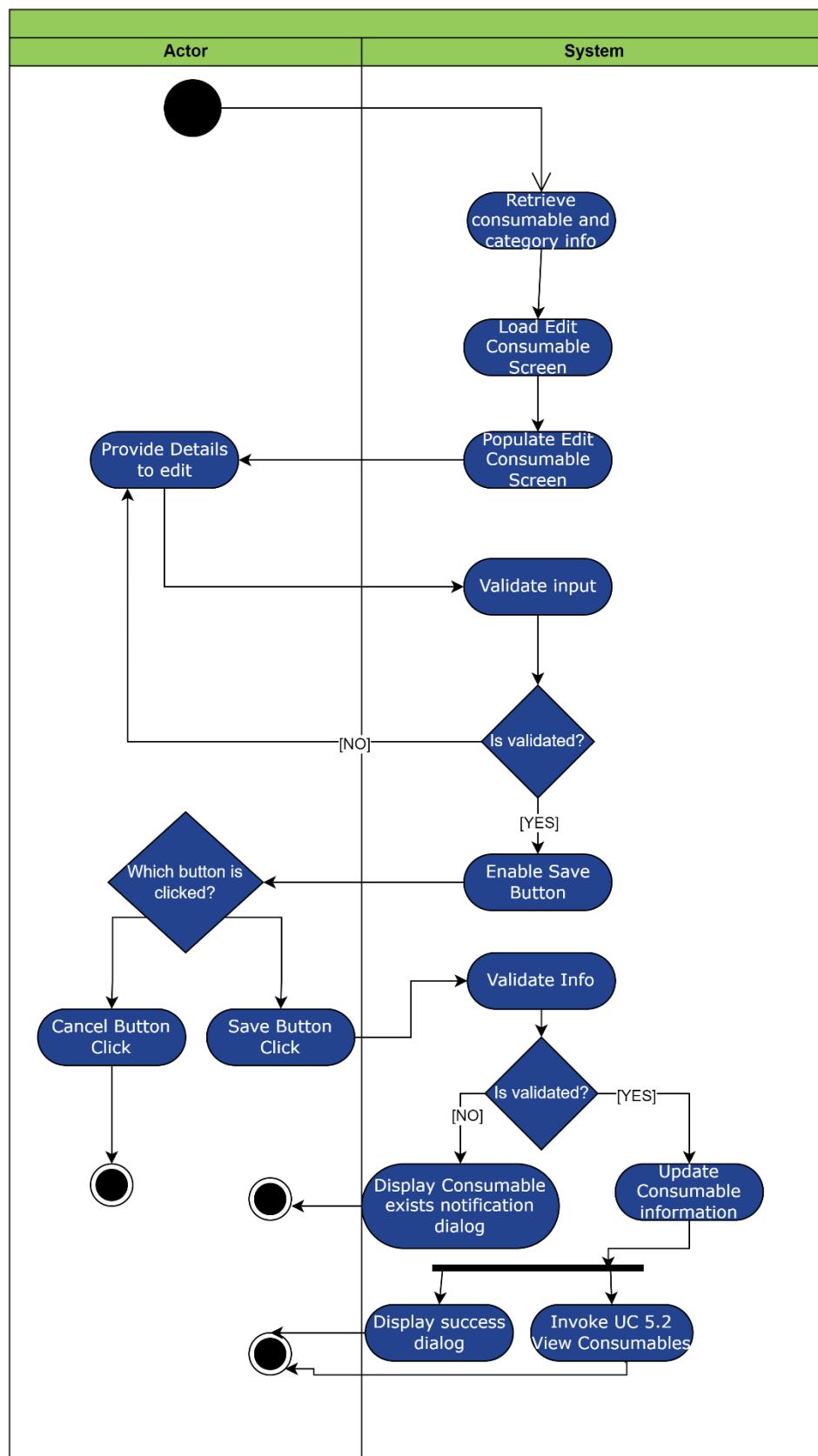


Figure 227 5.3 Update Consumable Activity Diagram

## 5.4 DELETE CONSUMABLE

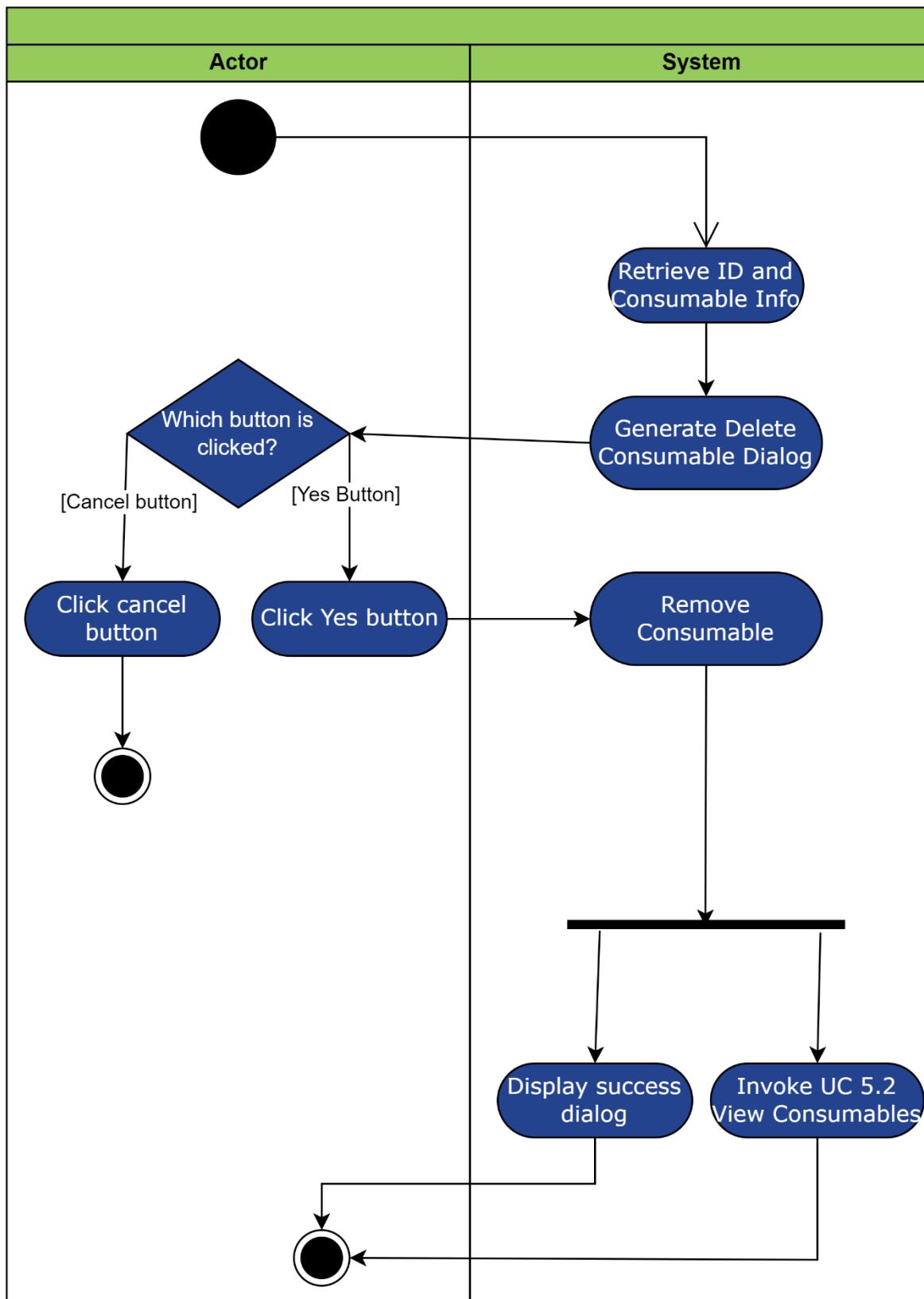


Figure 228 5.4 Delete Consumable Activity Diagram

## 5.5 CREATE CONSUMABLE CATEGORY

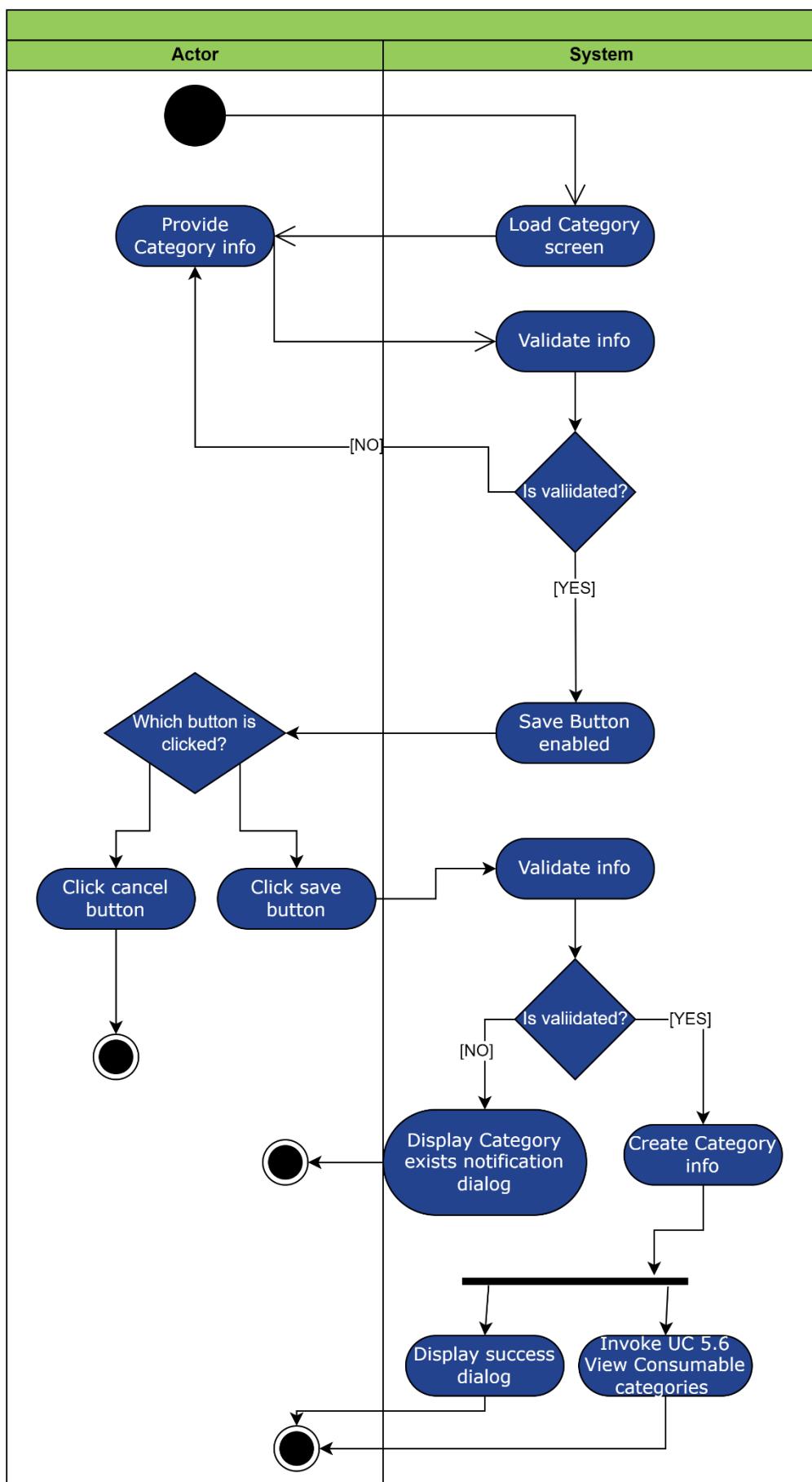


Figure 229 5.5 Create Consumable category Activity Diagram

## 5.6 VIEW CONSUMABLE CATEGORIES

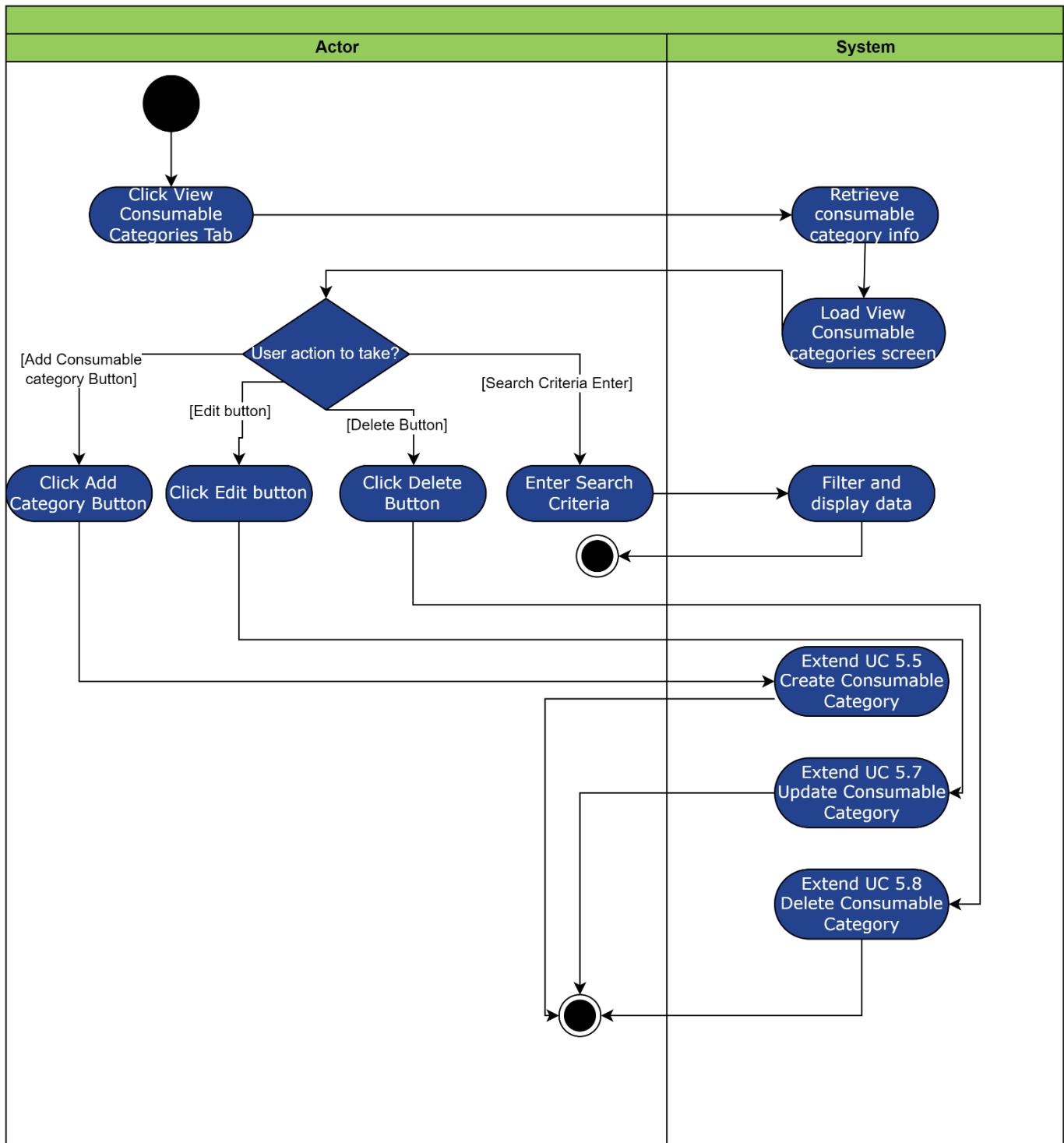


Figure 230 5.6 View Consumable category Activity Diagram

## 5.7 UPDATE CONSUMABLE CATEGORY

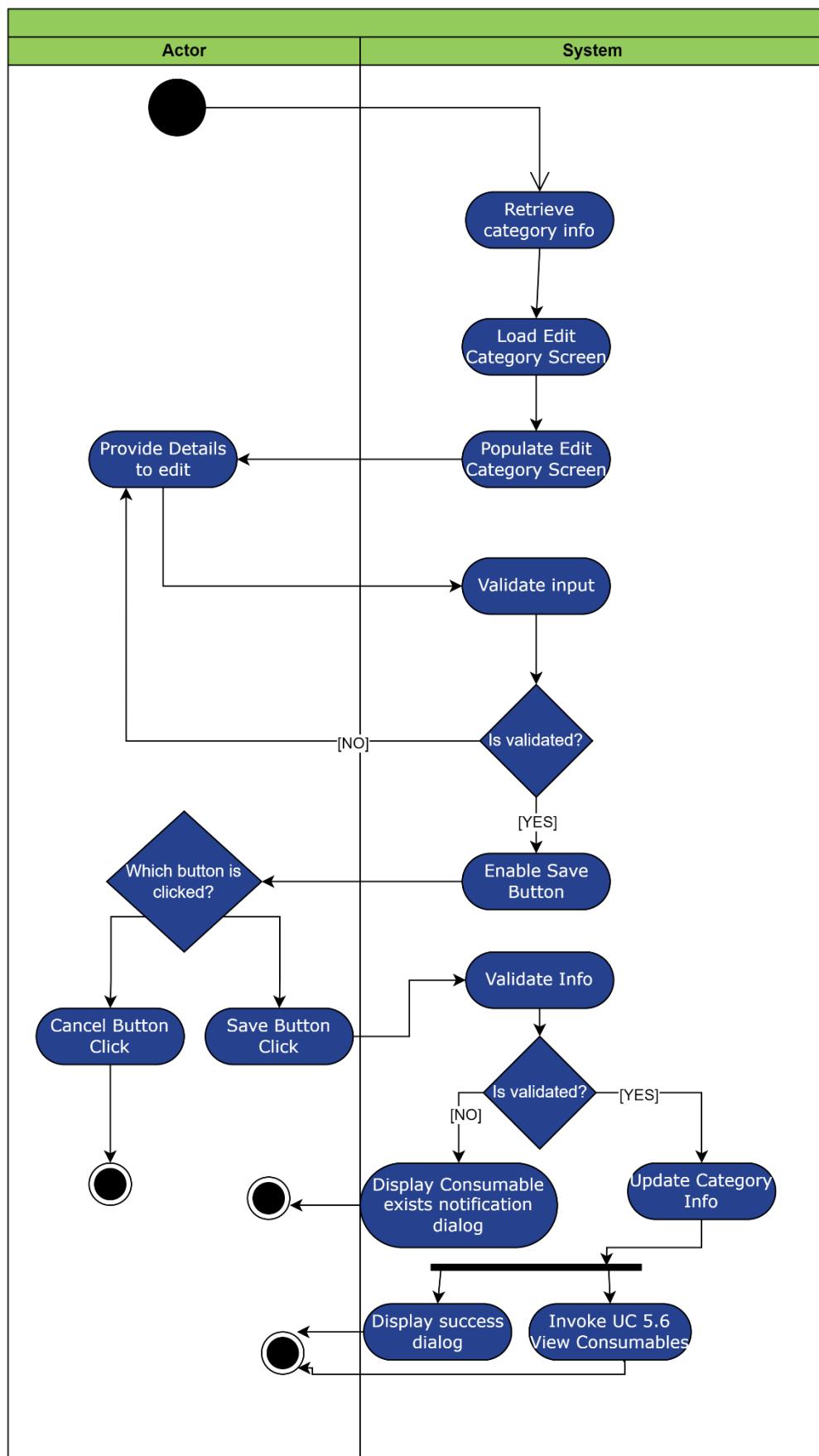


Figure 231 5.7 Update Consumable Category Activity Diagram

## 5.8 DELETE CONSUMABLE CATEGORY

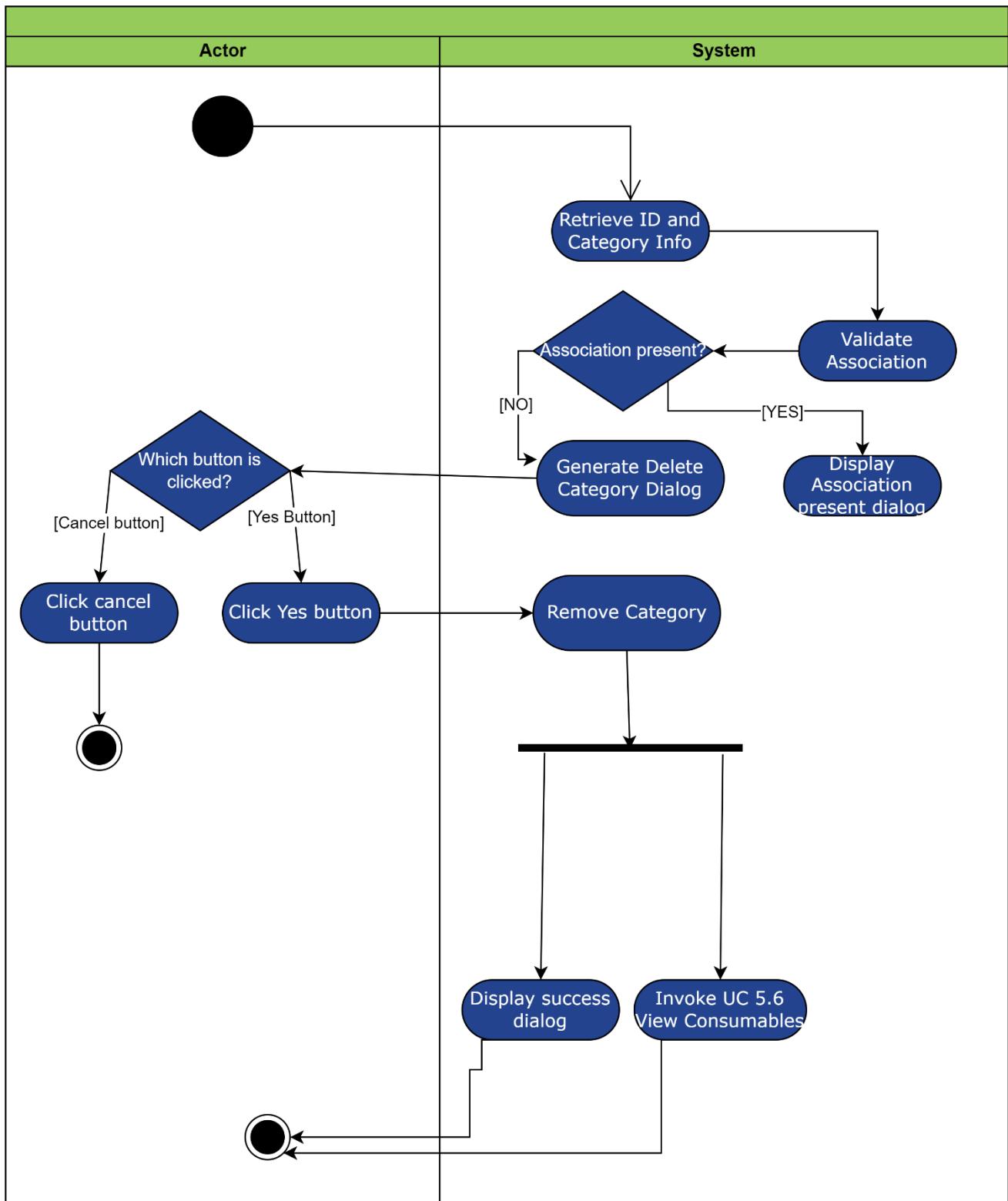


Figure 232 5.8 Delete Consumable Category Activity Diagram

## 5.9 EXPORT INVENTORY DETAILS

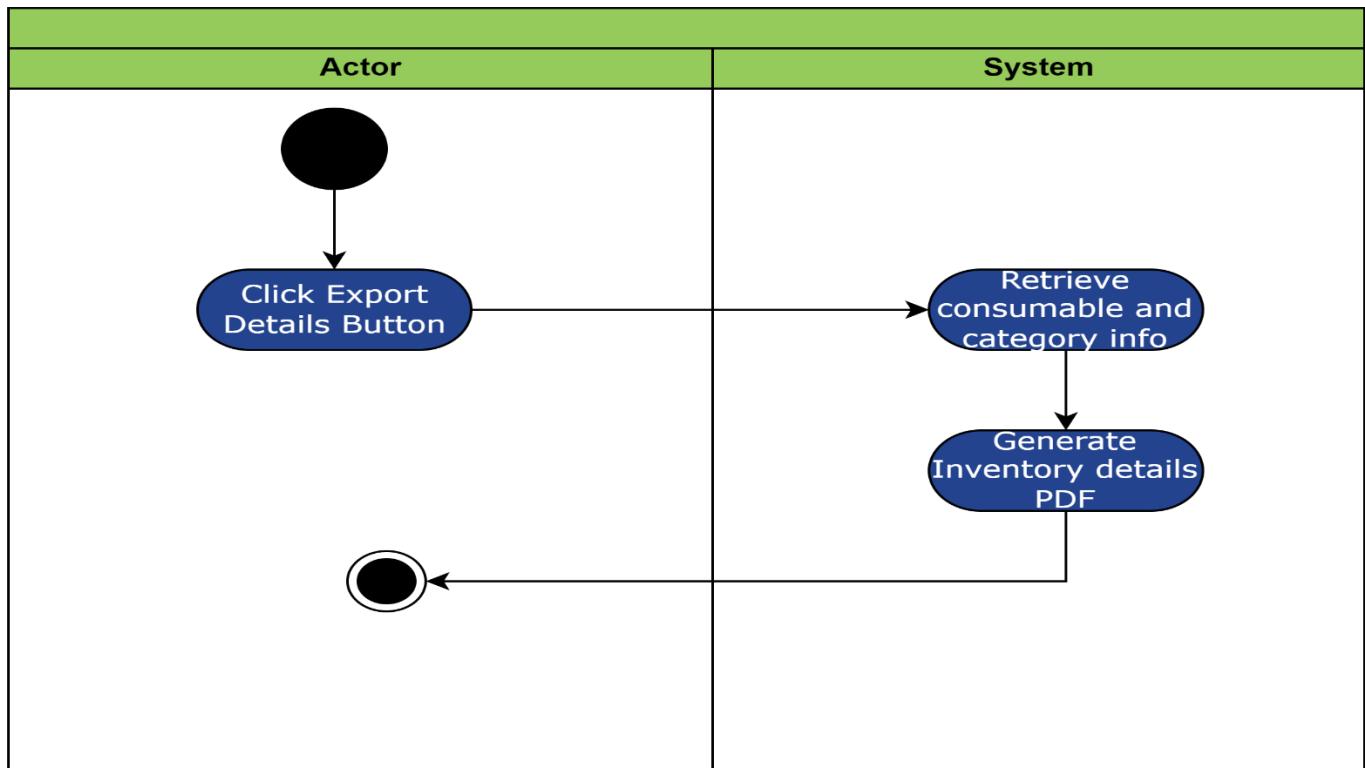


Figure 233 5.9. Export Inventory Details Activity Diagram

## 5.11 UPDATE STOCK

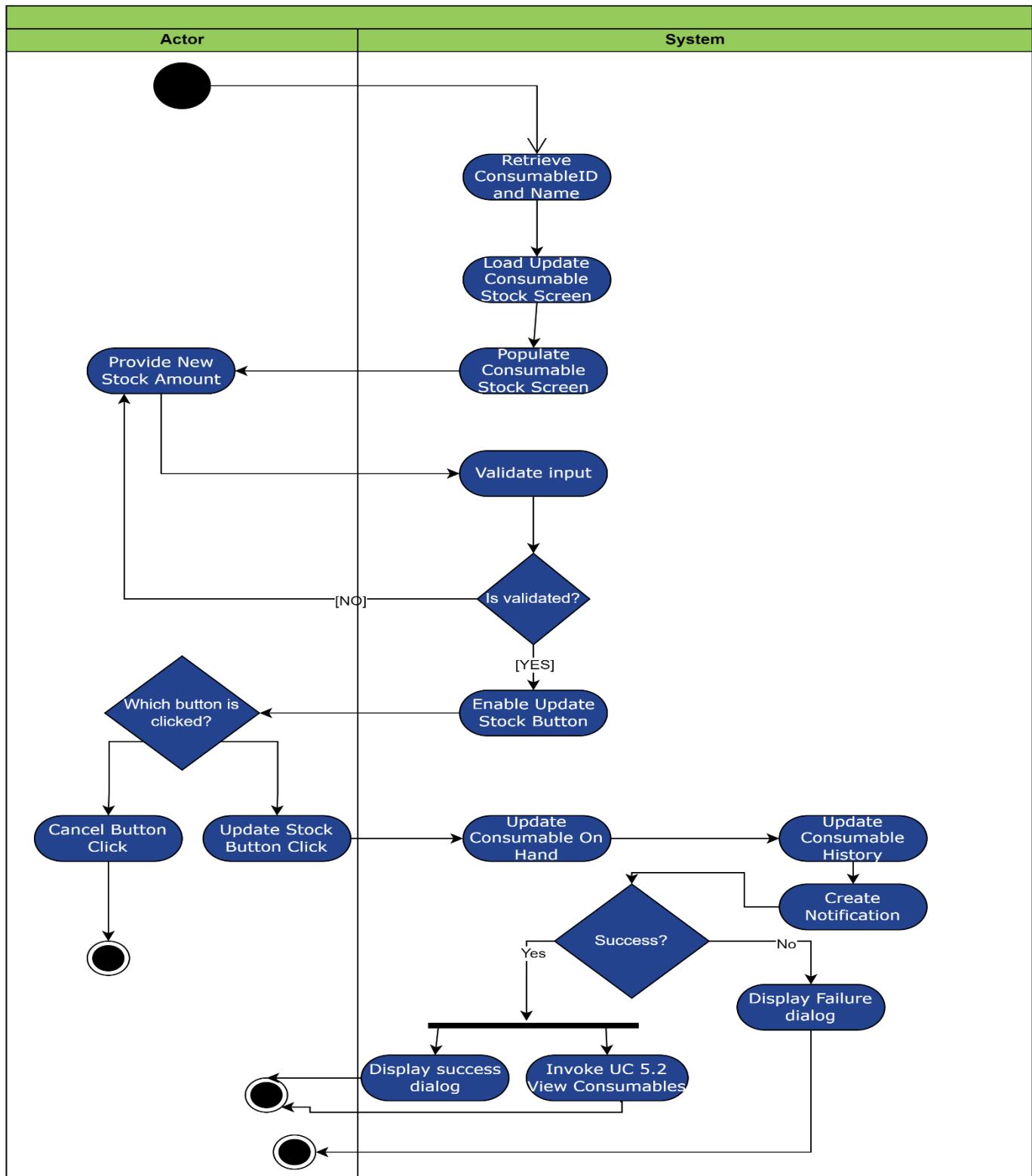


Figure 234 5.10. Update Stock Activity Diagram

## 1.1 LOGIN

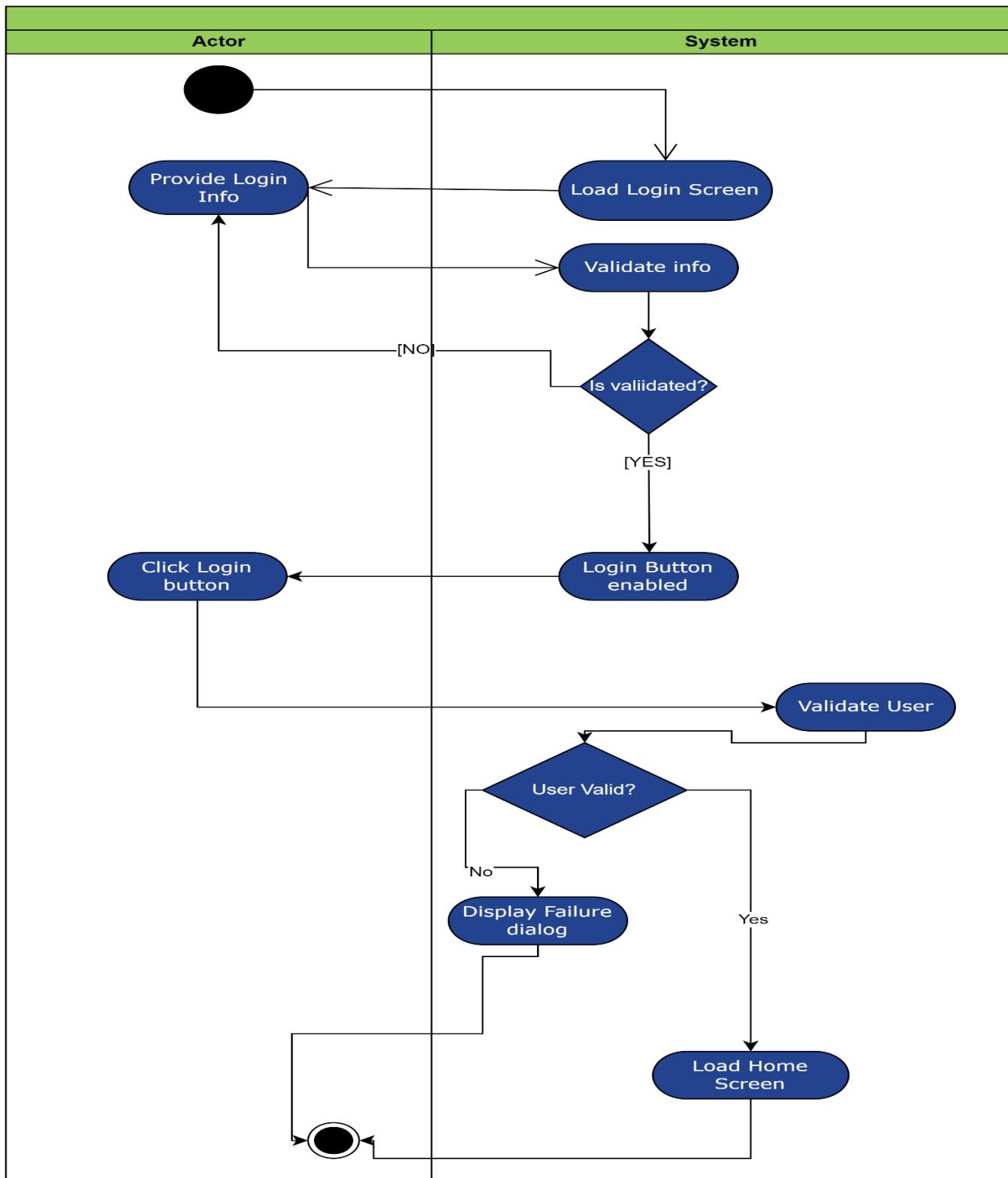


Figure 235 1.1. Login Activity Diagram

## 1.2 LOGOUT

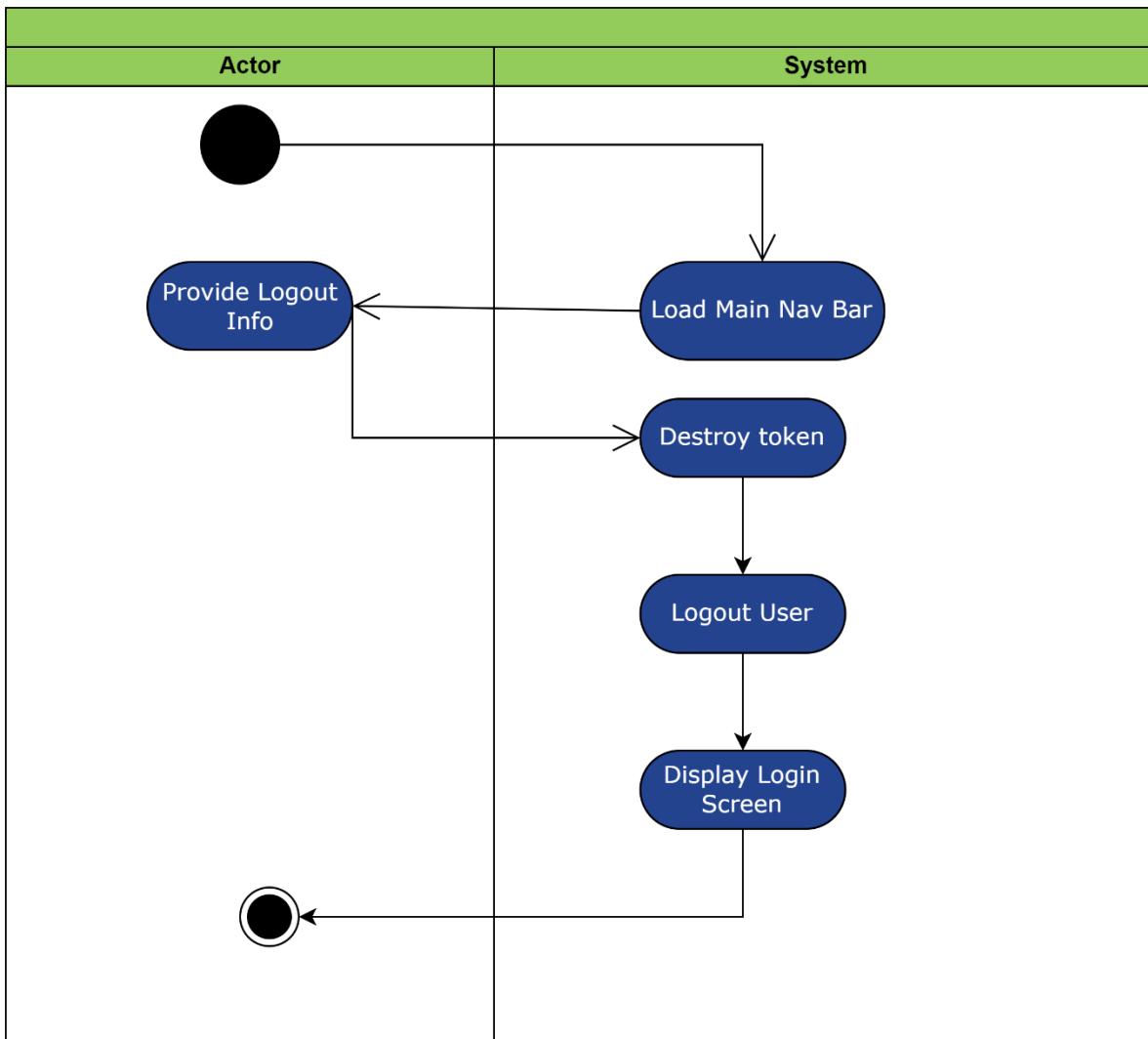


Figure 236 1.2. Logout Activity Diagram

### 1.3 FORGOT PASSWORD

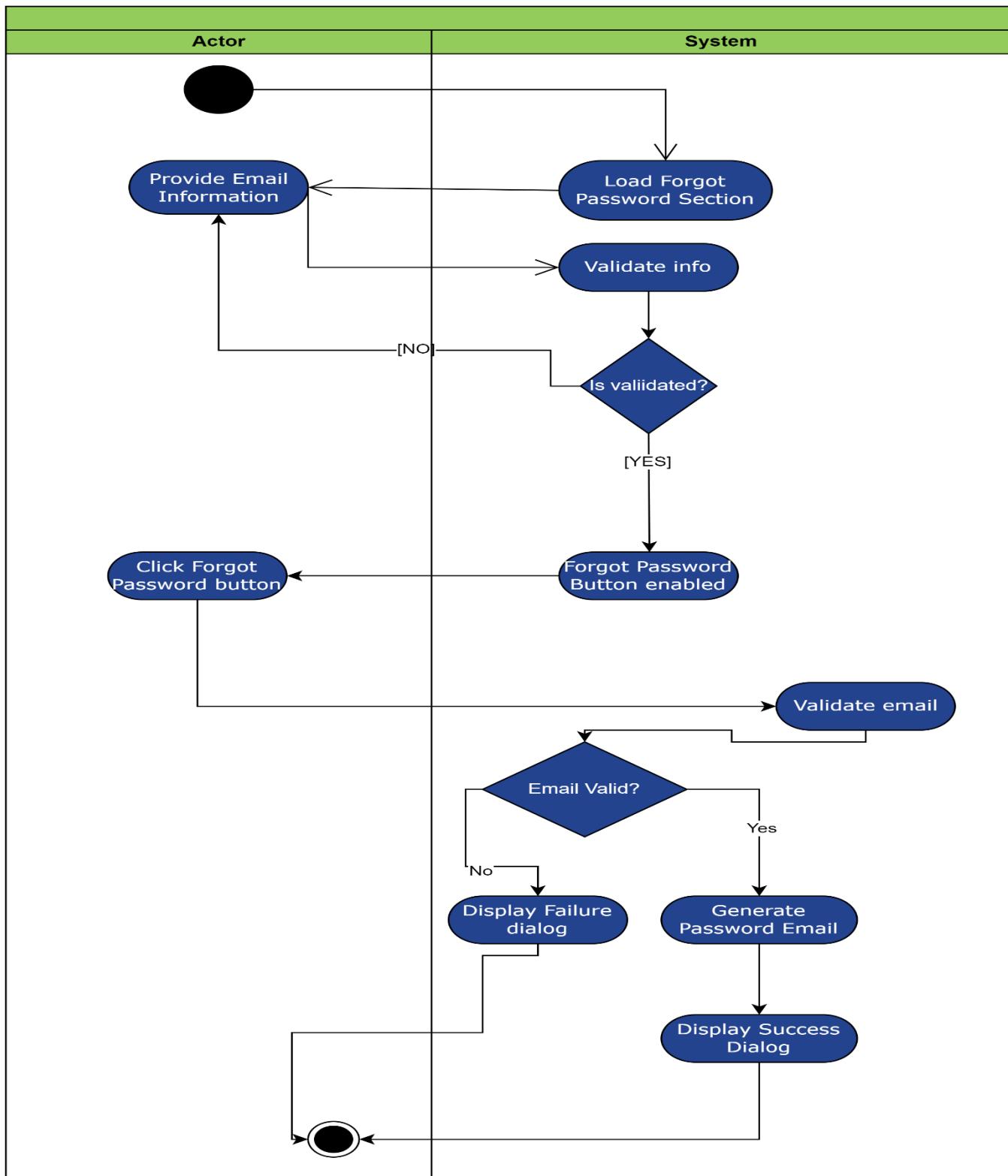


Figure 237 1.3. Forgot Password Activity Diagram

## 1.4 UPDATE PASSWORD

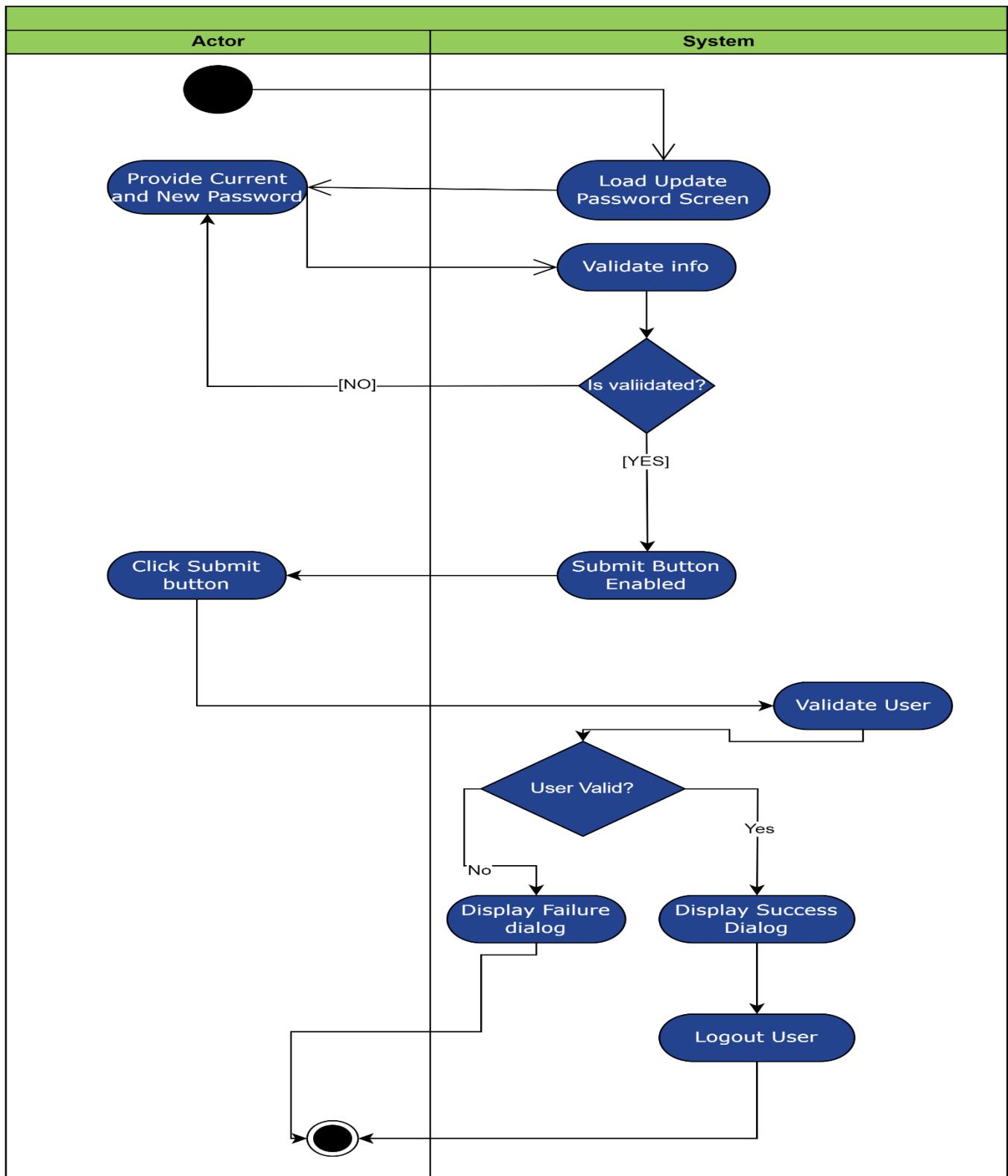


Figure 238 1.4. Update Password Activity Diagram

### 3.1 CREATE PROCUREMENT REQUEST

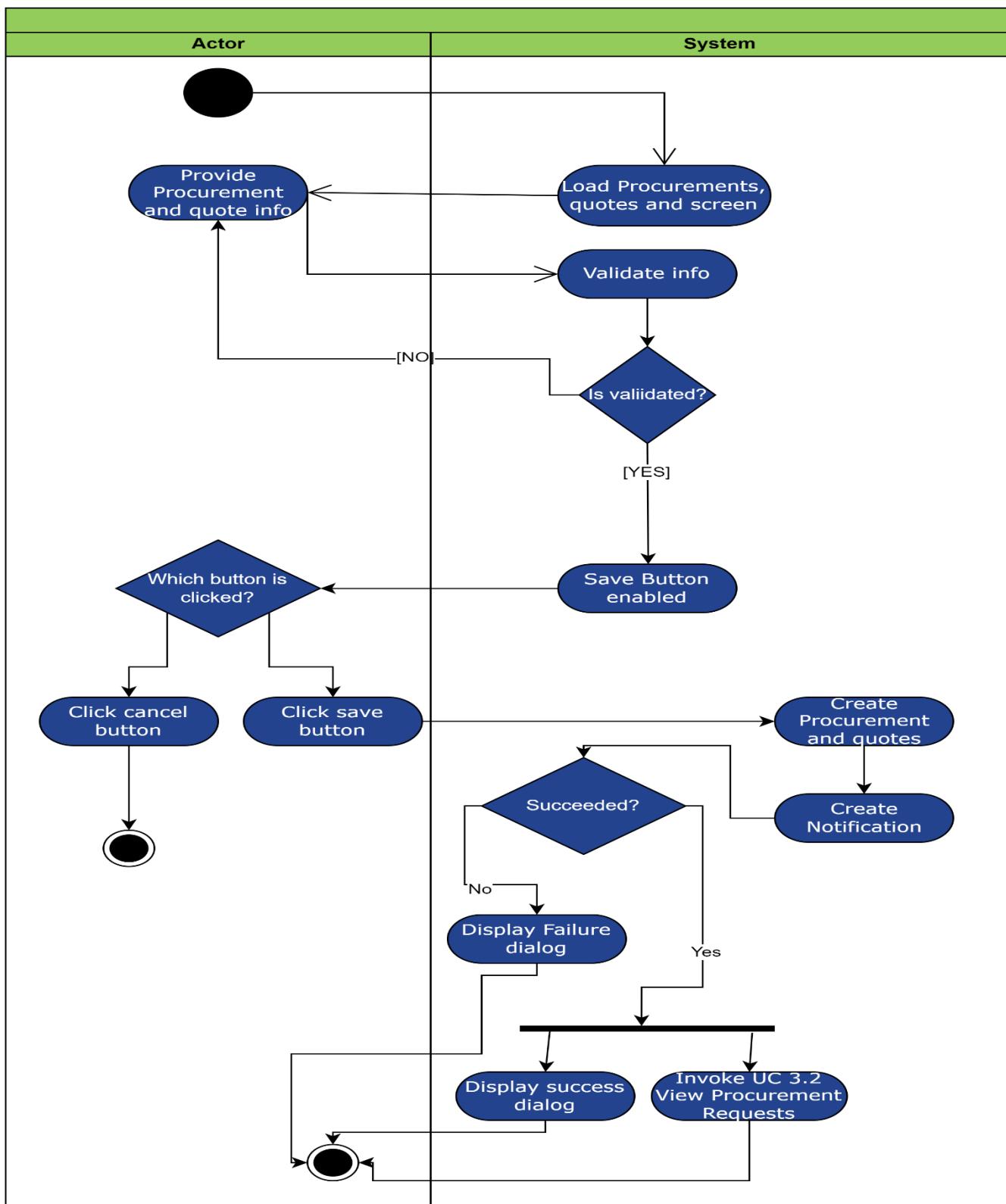


Figure 239 3.1. Create Procurement Request Activity Diagram

### 3.2 VIEW PROCUREMENT REQUEST

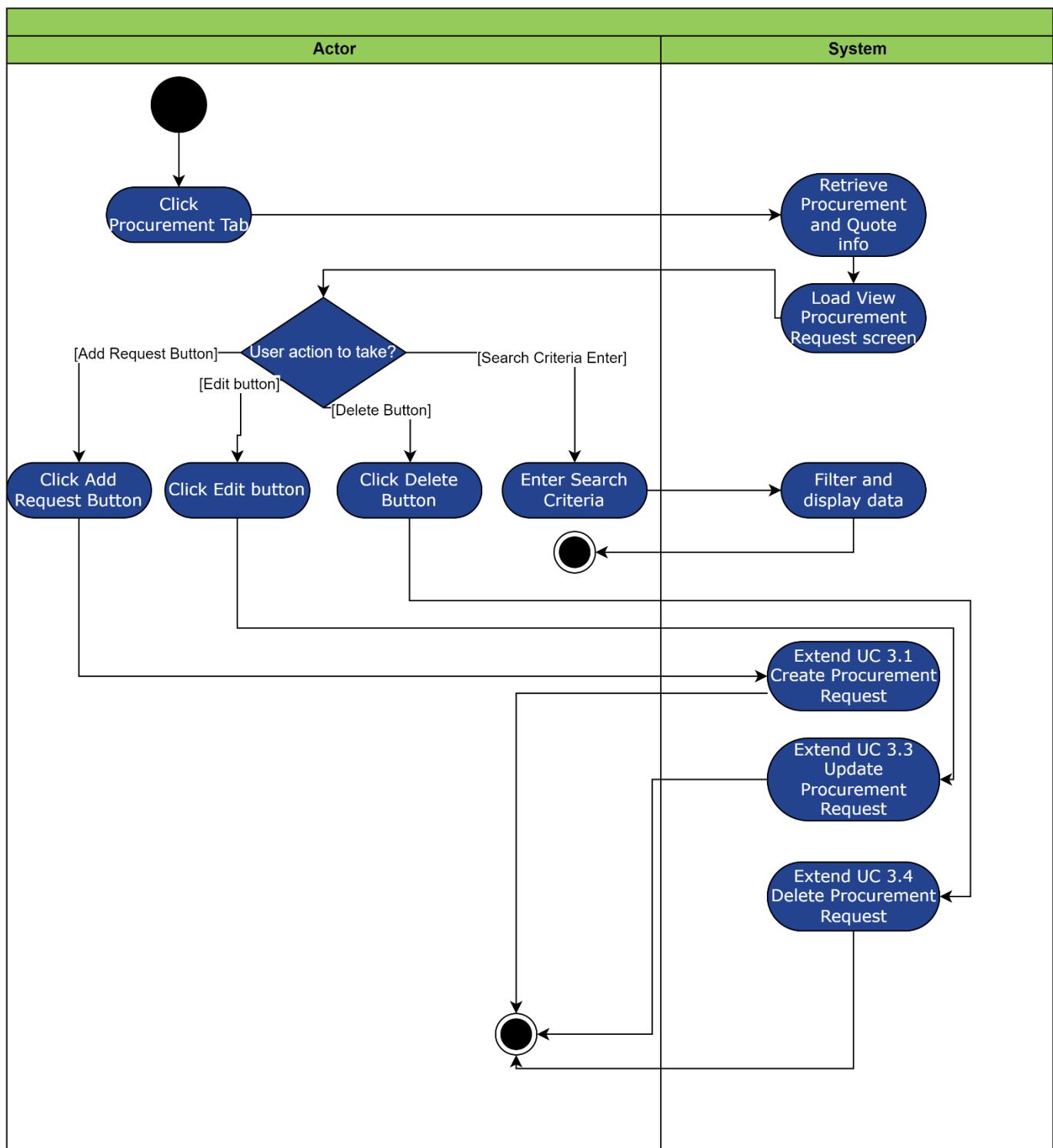


Figure 240 3.2. View Procurement Request Activity Diagram

### 3.3 UPDATE PROCUREMENT REQUEST

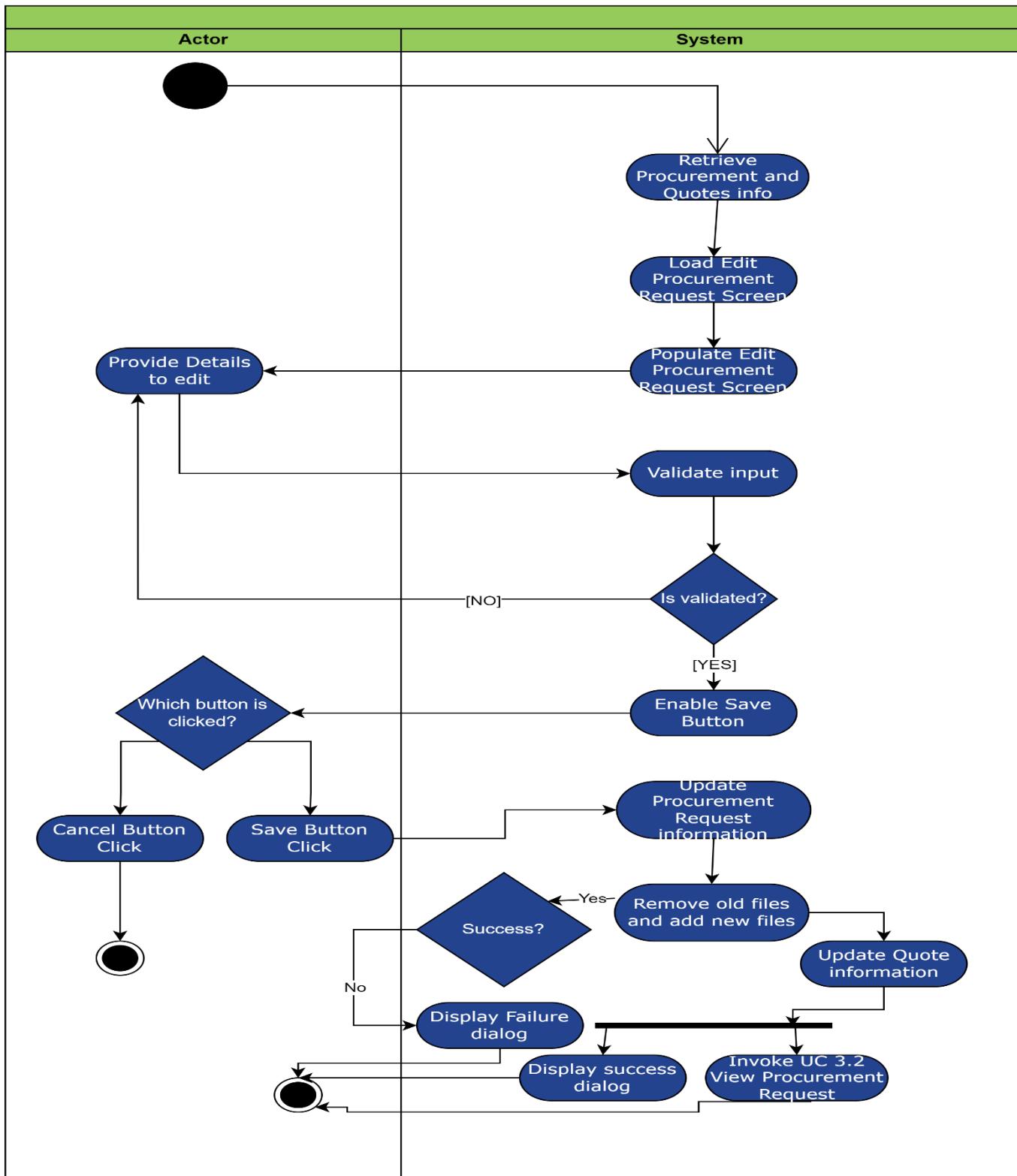


Figure 241 3.3. Update Procurement Request Activity Diagram

### 3.4 DELETE PROCUREMENT REQUEST

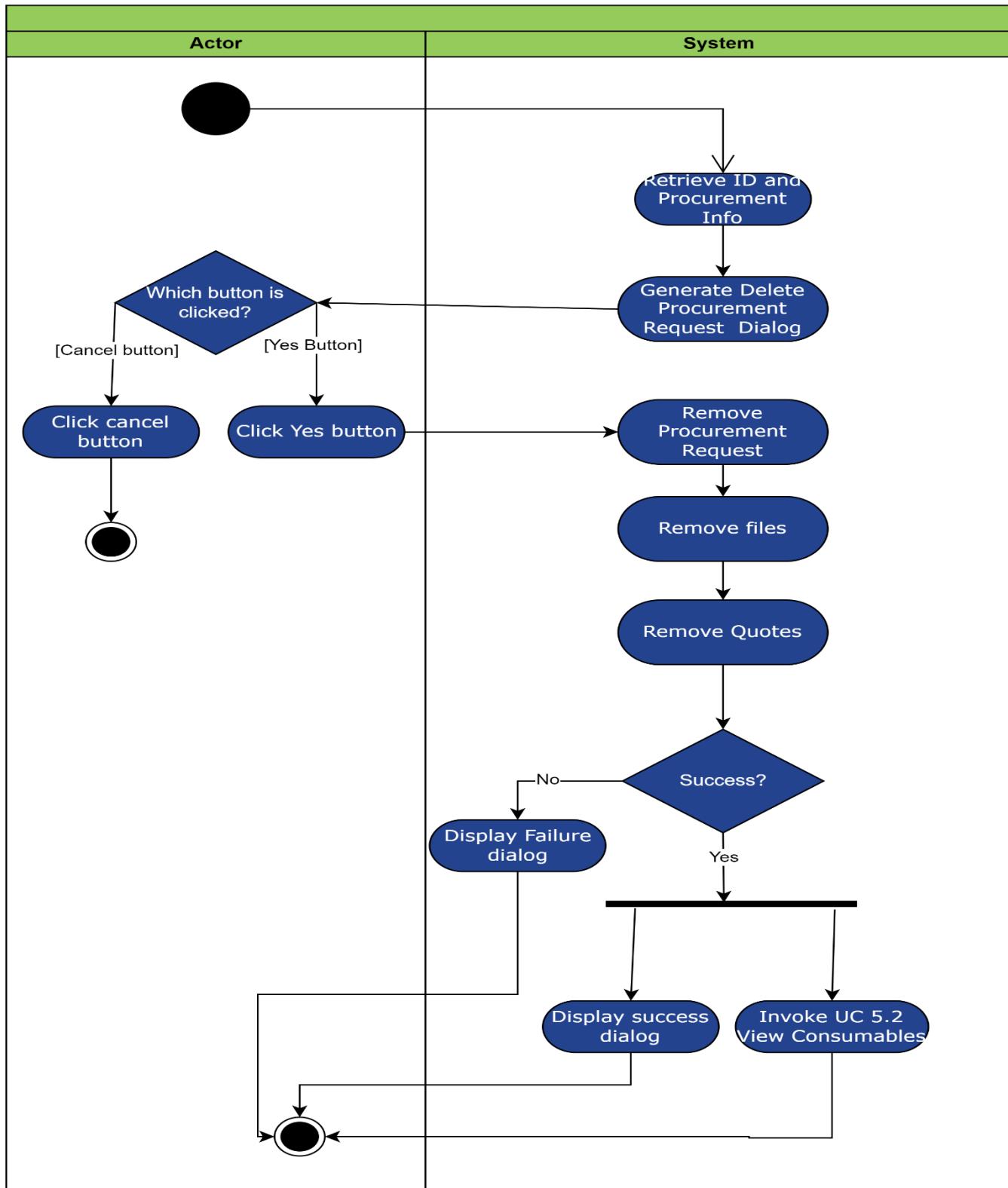


Figure 242 3.4. Delete Procurement Request Activity Diagram

USE CASE 3.5-3.7,6.5,6.10-6.14

## 3.5 APPROVE PROCUREMENT REQUEST

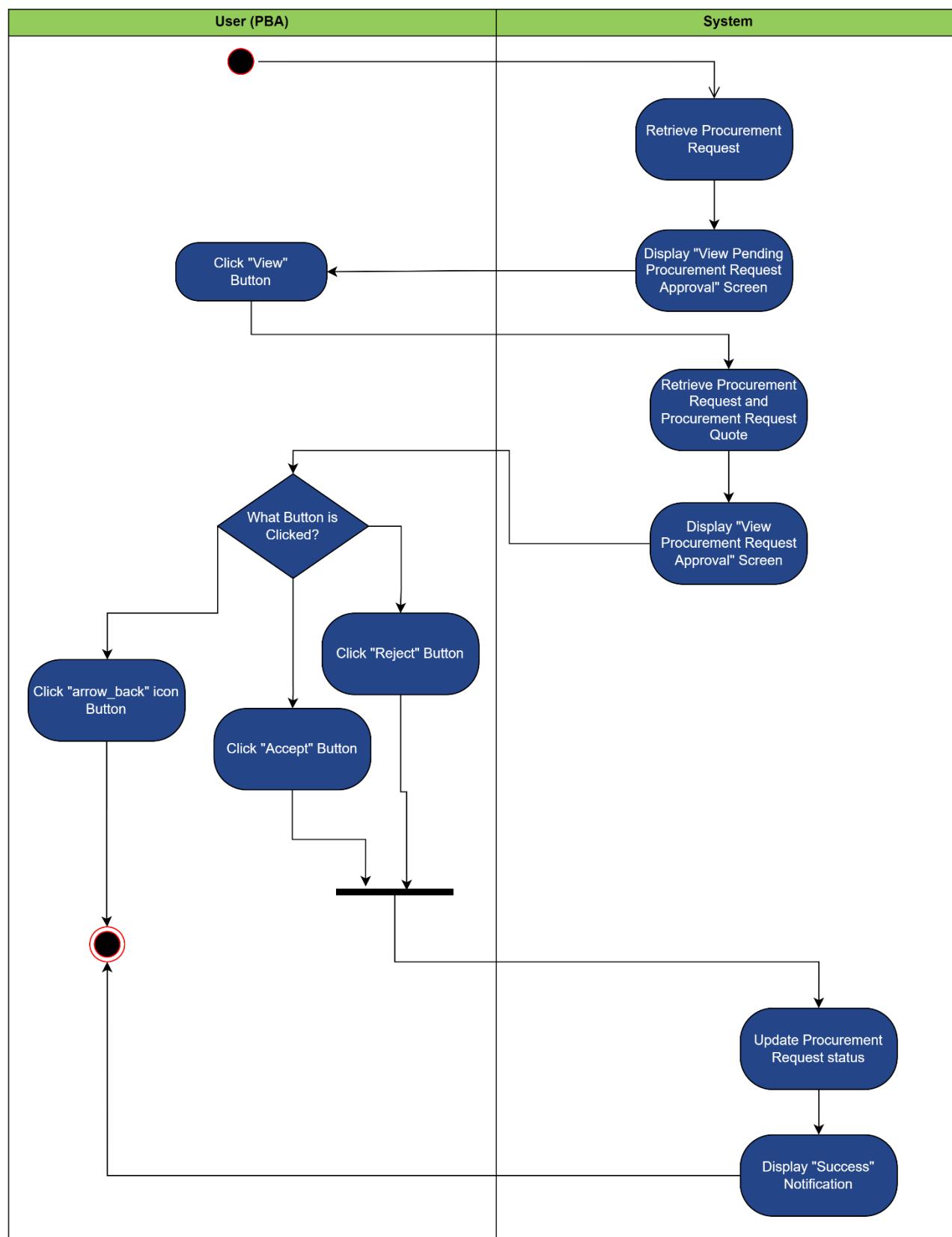


Figure 243: 3.5 Approve Procurement Request

### 3.6 PLACE PROCUREMENT DETAILS

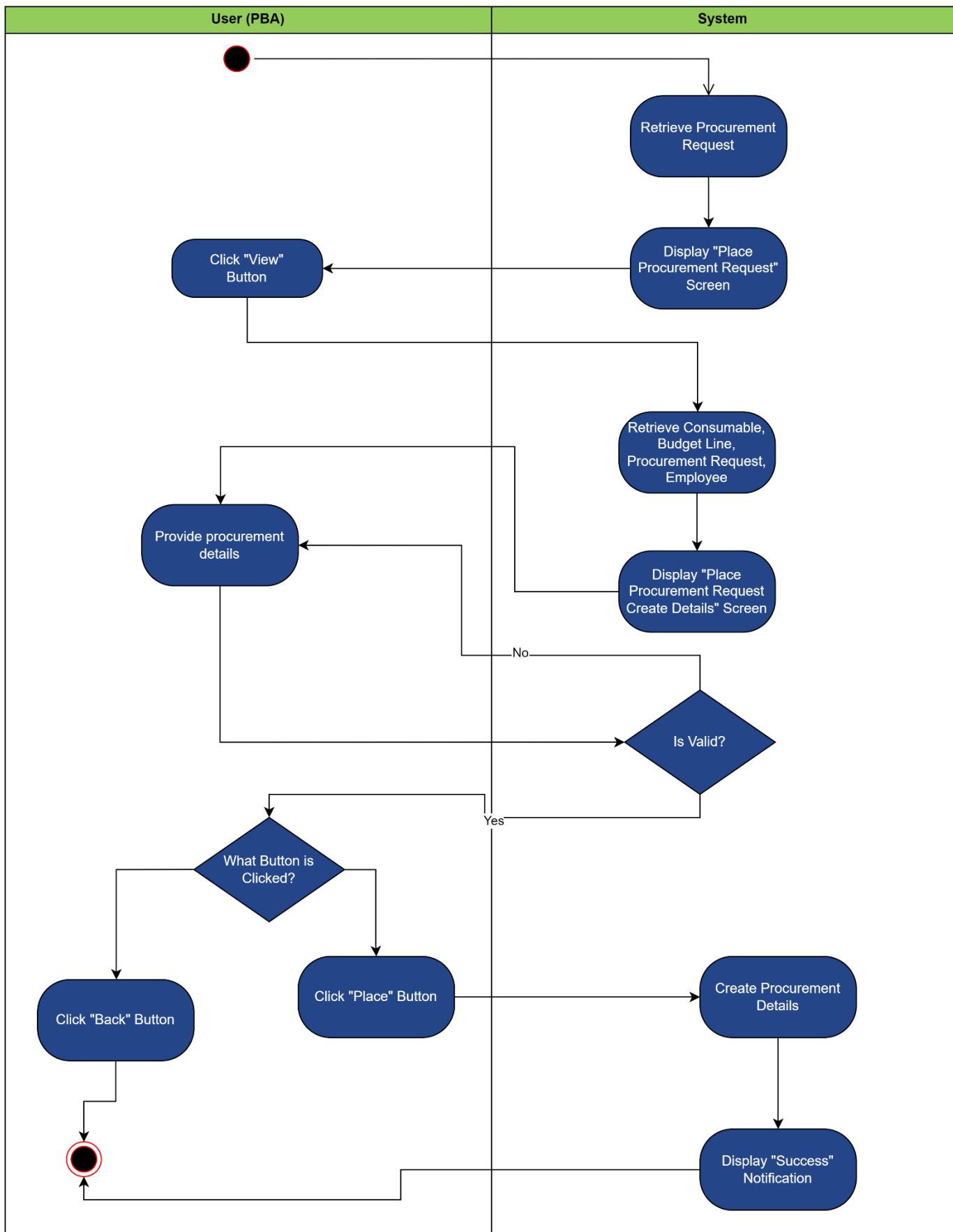


Figure 244: 3.6 Place Procurement Details

### 3.7 APPROVE FLAGGED PROCUREMENT DETAILS

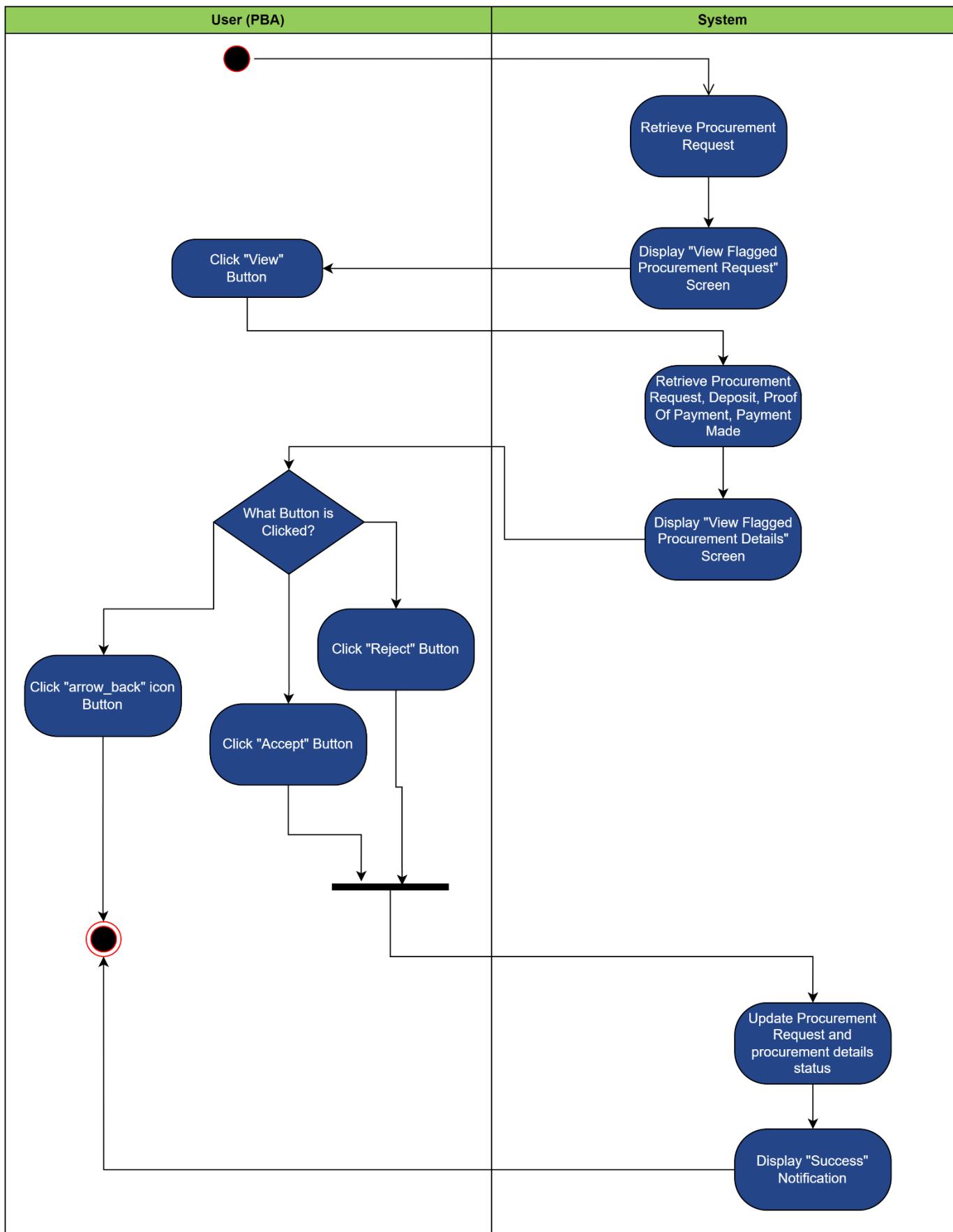


Figure 245: Approve Flagged Procurement Details

## 6.5 APPROVE ONBOARD REQUEST

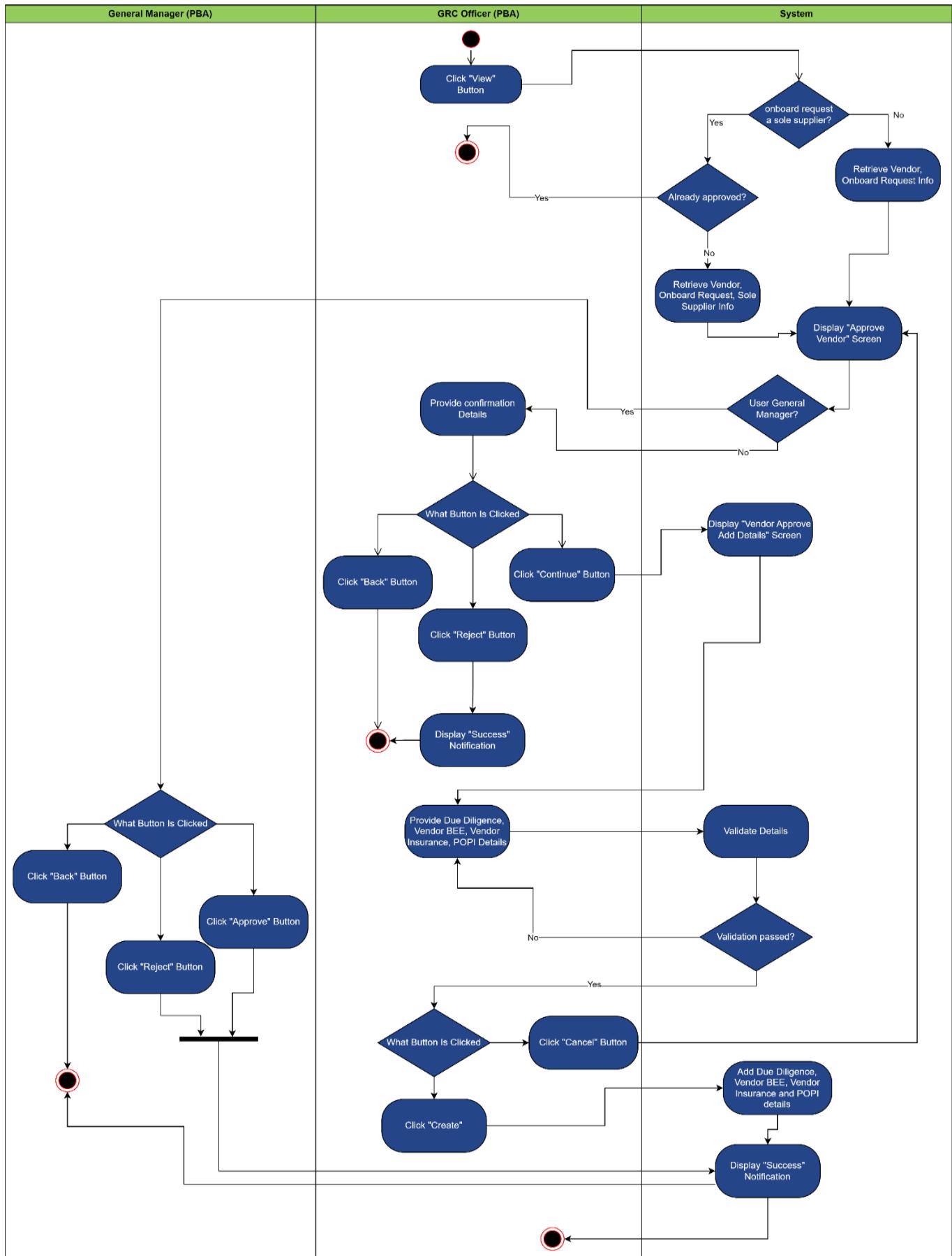
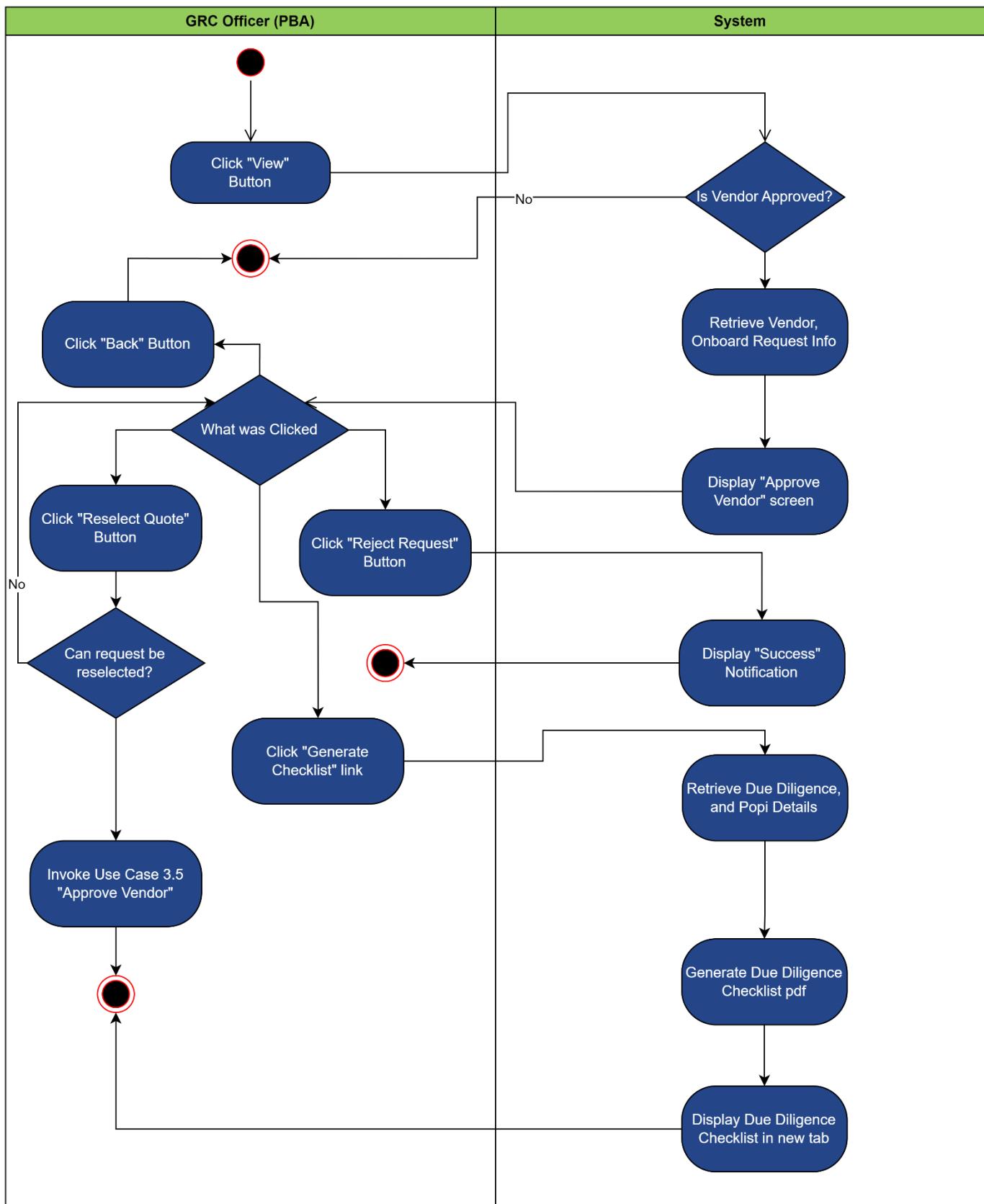


Figure 246: 6.5 Approve Vendor

**6.10 GENERATE DUE DILIGENCE CHECKLIST**

*Figure 247: 6.10 Generate Due Diligence Checklist*

### 6.11 GENERATE SOLE SUPPLIER PERFORMANCE REVIEW NOTIFICATION

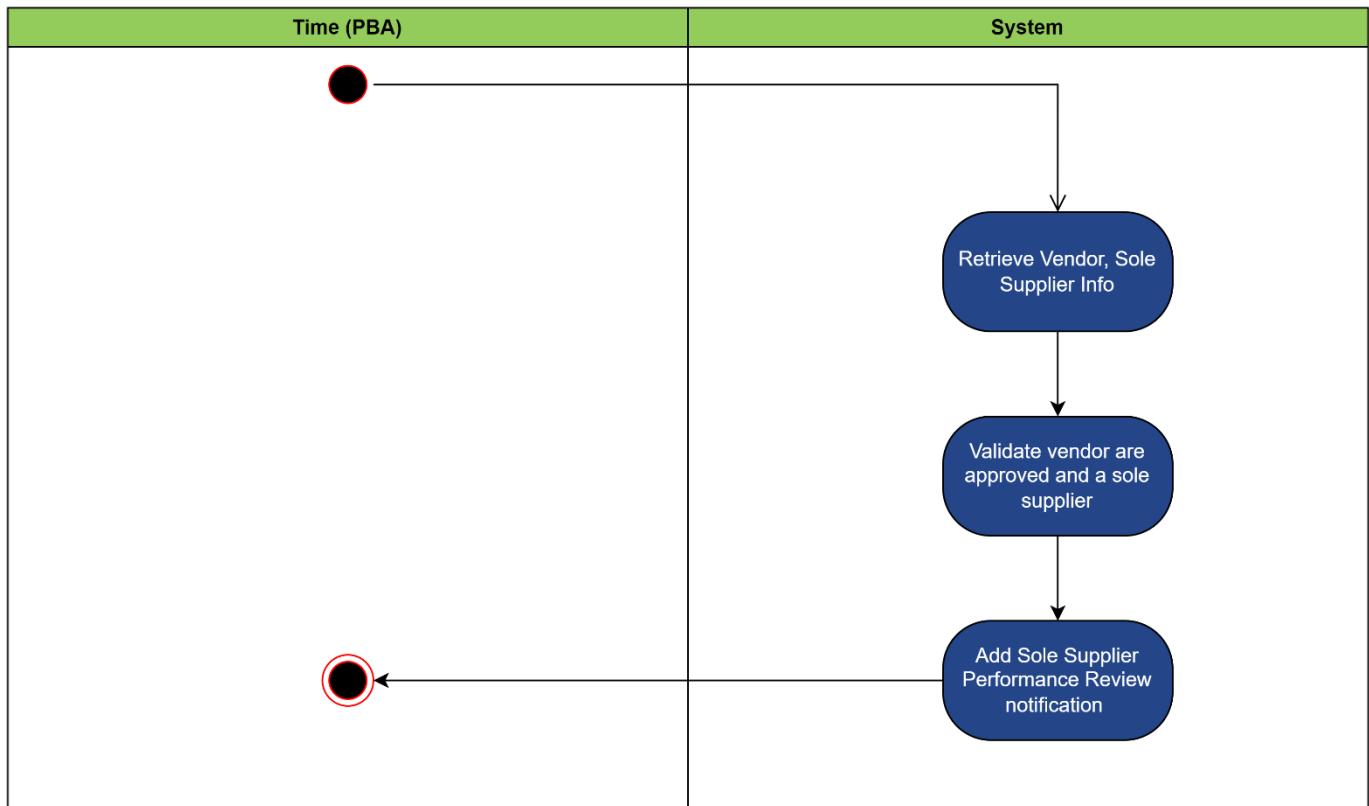


Table 221: 6.11 Generate Sole Supplier Performance Review Notification

## 6.12 GENERATE BEE EXPIRY NOTIFICATION

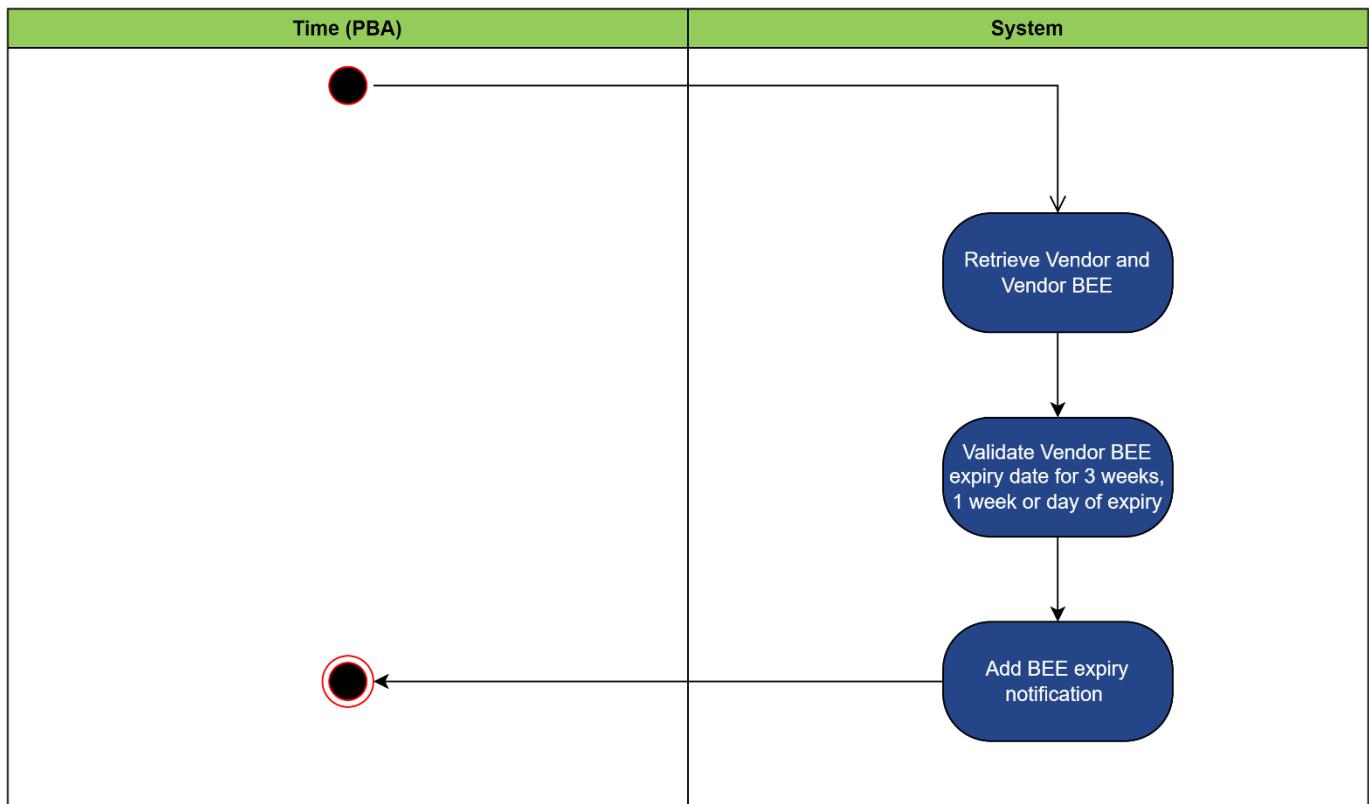


Table 222: 6.12 Generate BEE Expiry notification

### 6.13 UPDATE APPROVE ONBOARD REQUEST

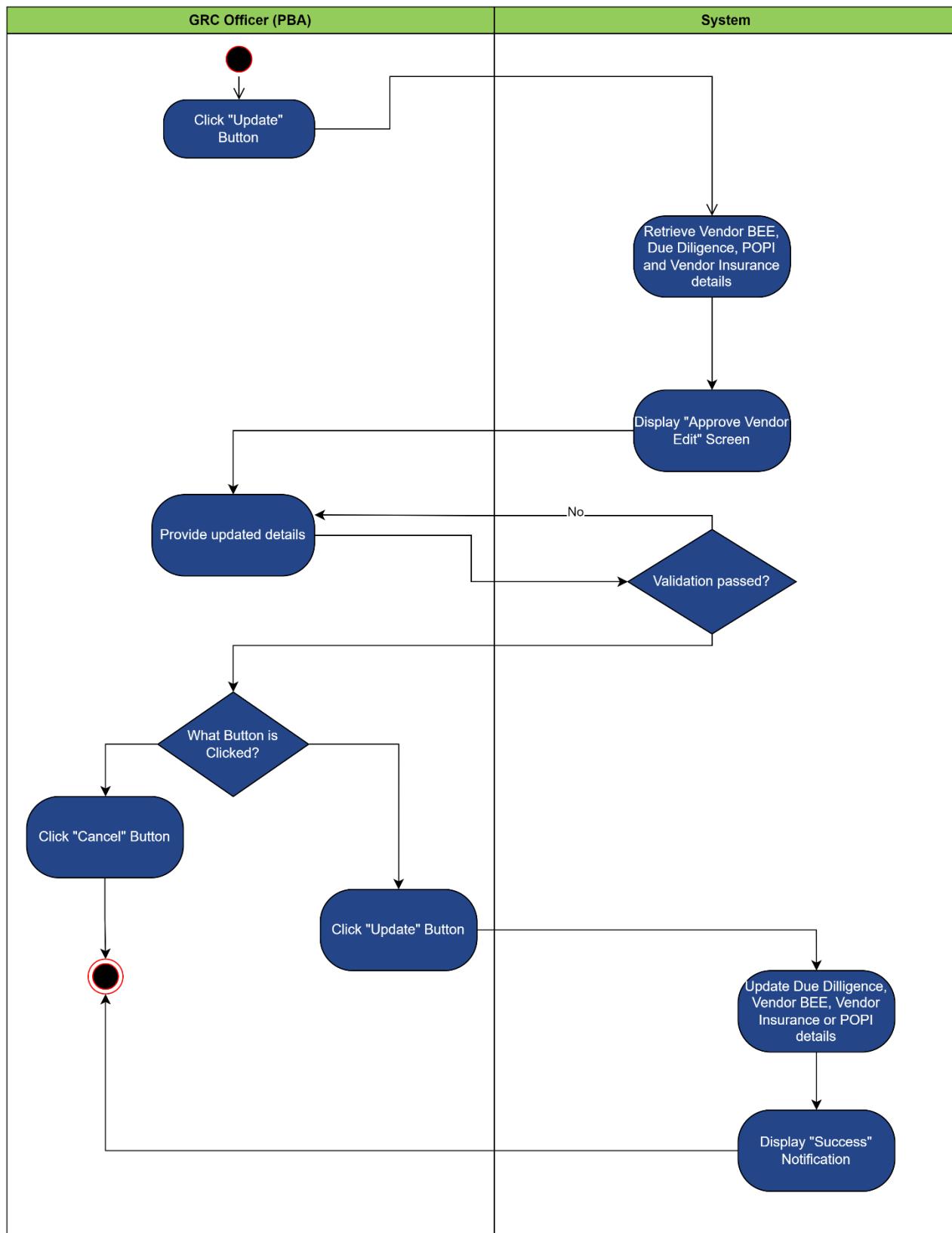


Table 223: 6.13 Update Approve Onboard Request

## 6.14 VIEW PENDING OR APPROVED ONBOARD REQUEST

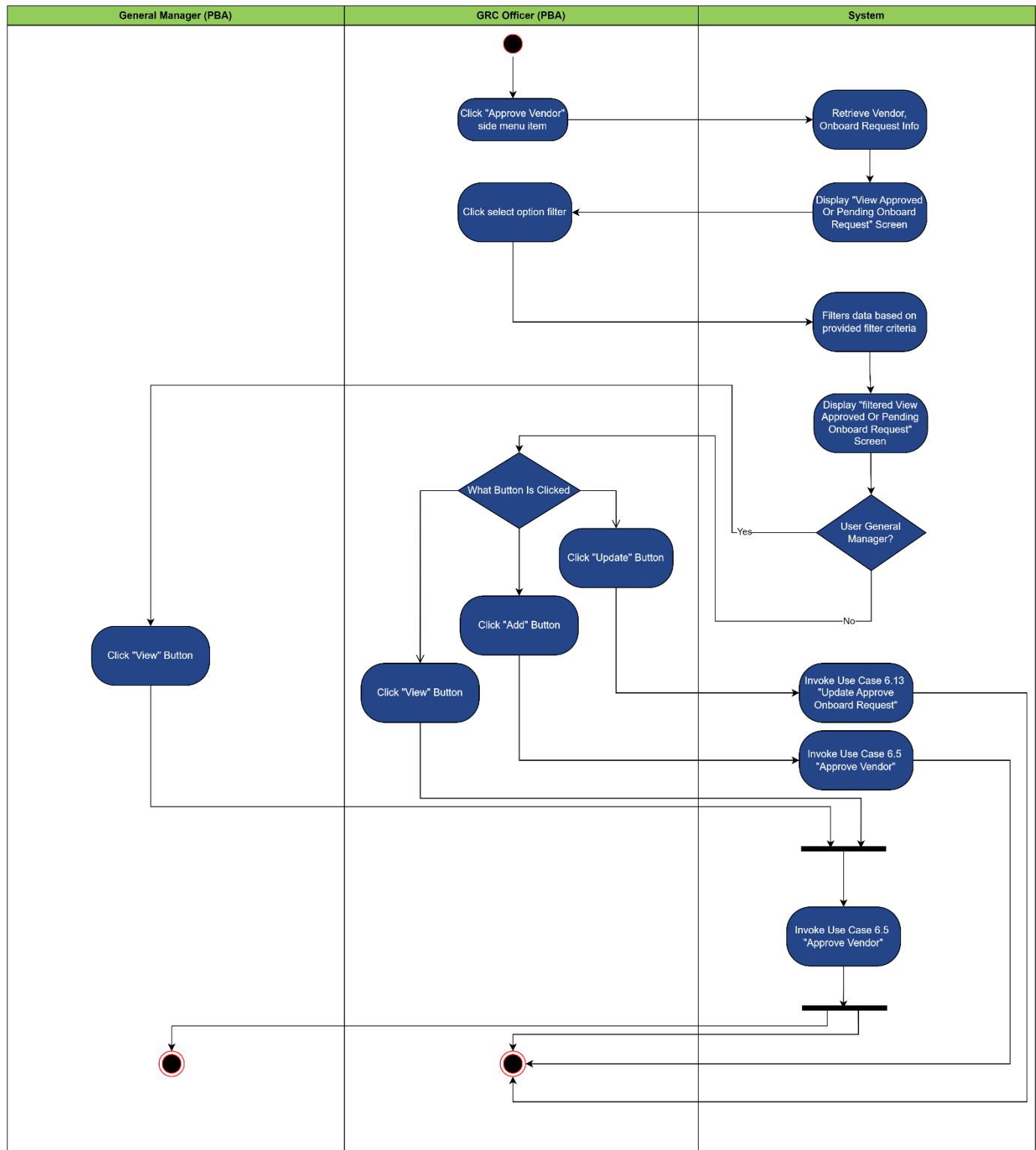


Figure 248: View Pending Or Approved Onboard Request

## 8.3 CONCLUSION

This Conclude the visual representation of the Activity diagram with for the respective use case diagrams for the Procion system.

## 9. SEQUENCE DIAGRAM

## 9.1 INTRODUCTION

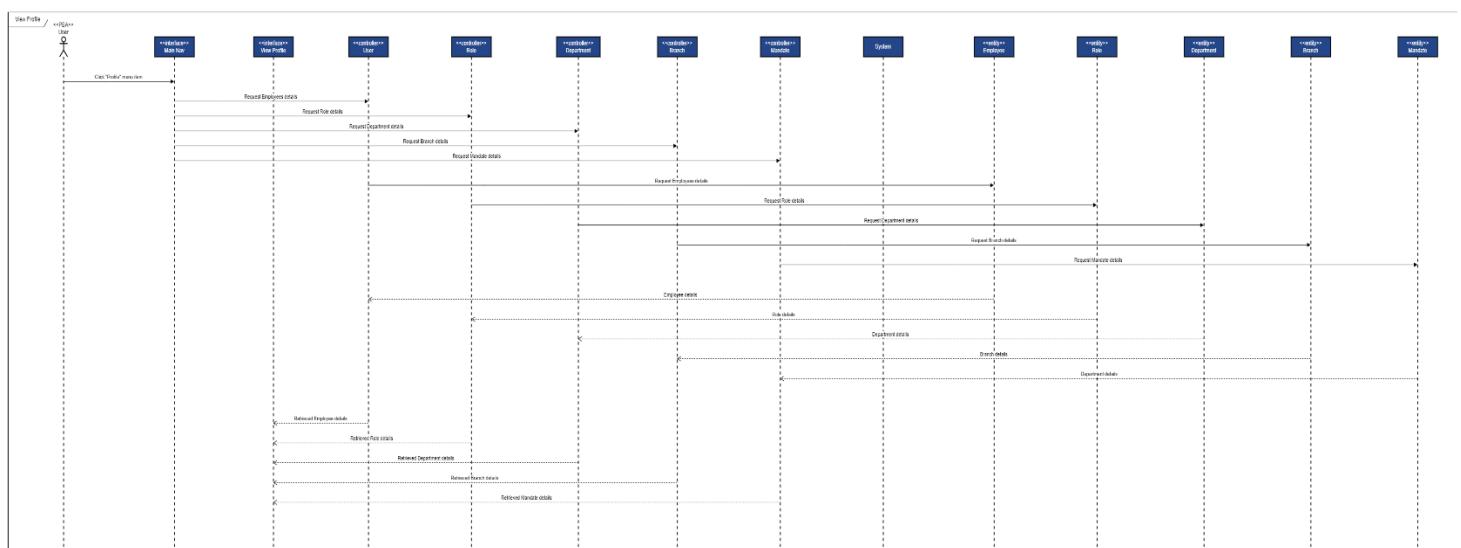
In this section we will display the visual representation of the Sequence diagrams for the following use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) respectively, of the Procion system.

## 9.2 SEQUENCE DIAGRAM

---

USE CASE 1.5, 1.6, 1.9 & 2.17-2.20 & 2.35-2.36

## 1.5 VIEW PROFILE



*Figure 249 1.5 View Profile Sequence Diagram*

### 1.6 EDIT PROFILE

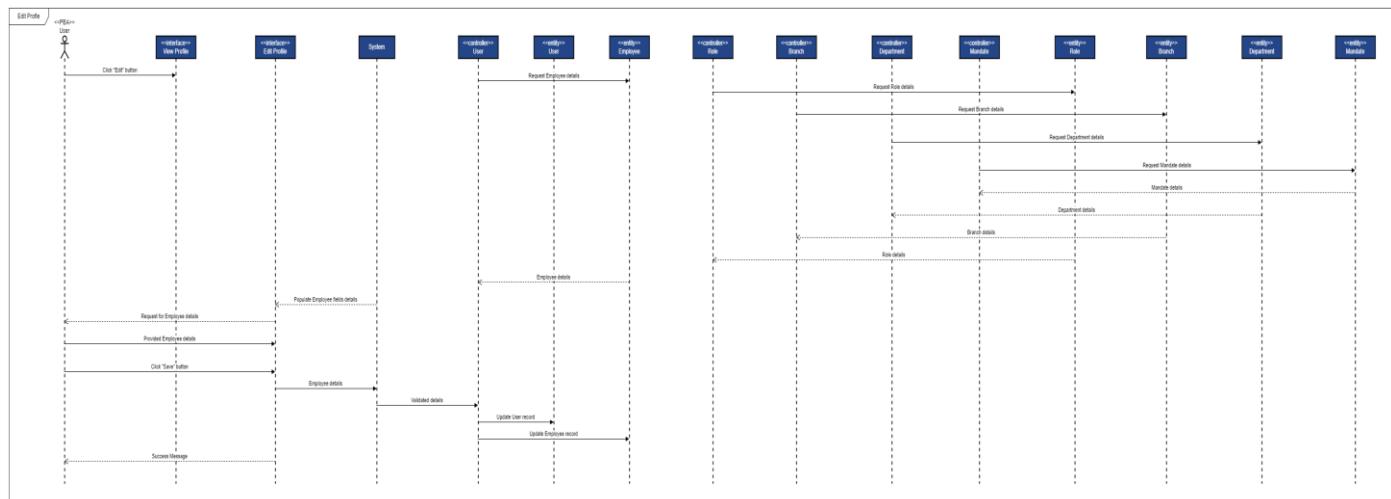


Figure 250 1.6 Edit Profile Sequence Diagram

### 1.8 VIEW NOTIFICATIONS

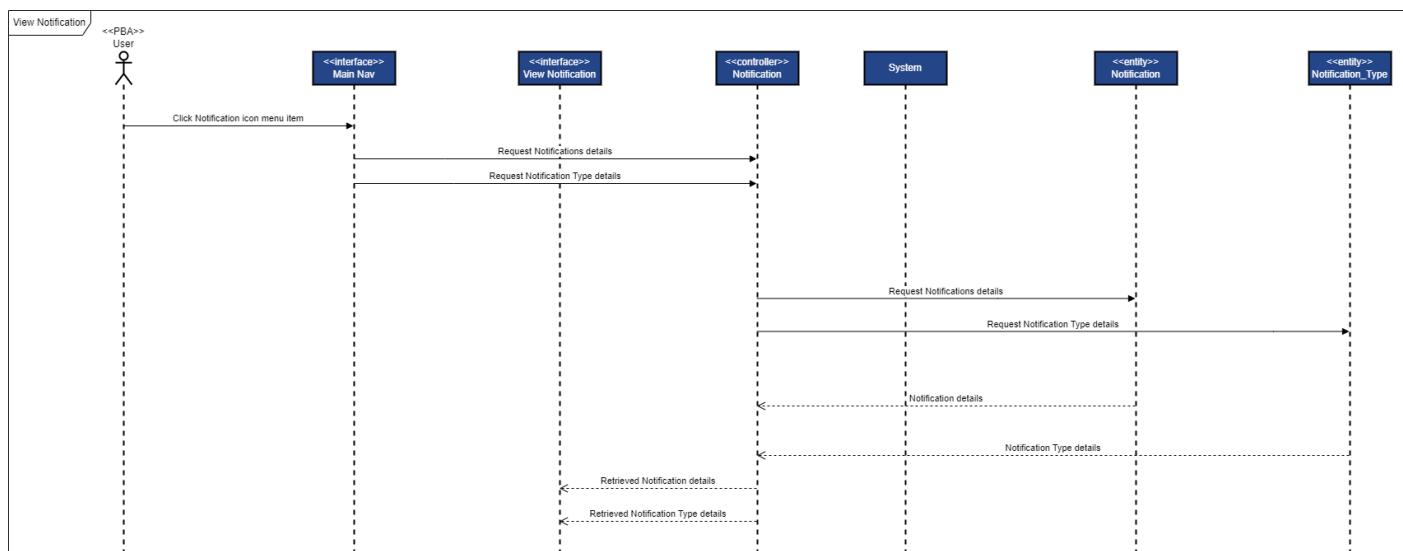


Figure 251 1.8 View Notifications Sequence Diagram

## 2.17 VIEW DELEGATION OF AUTHORITY

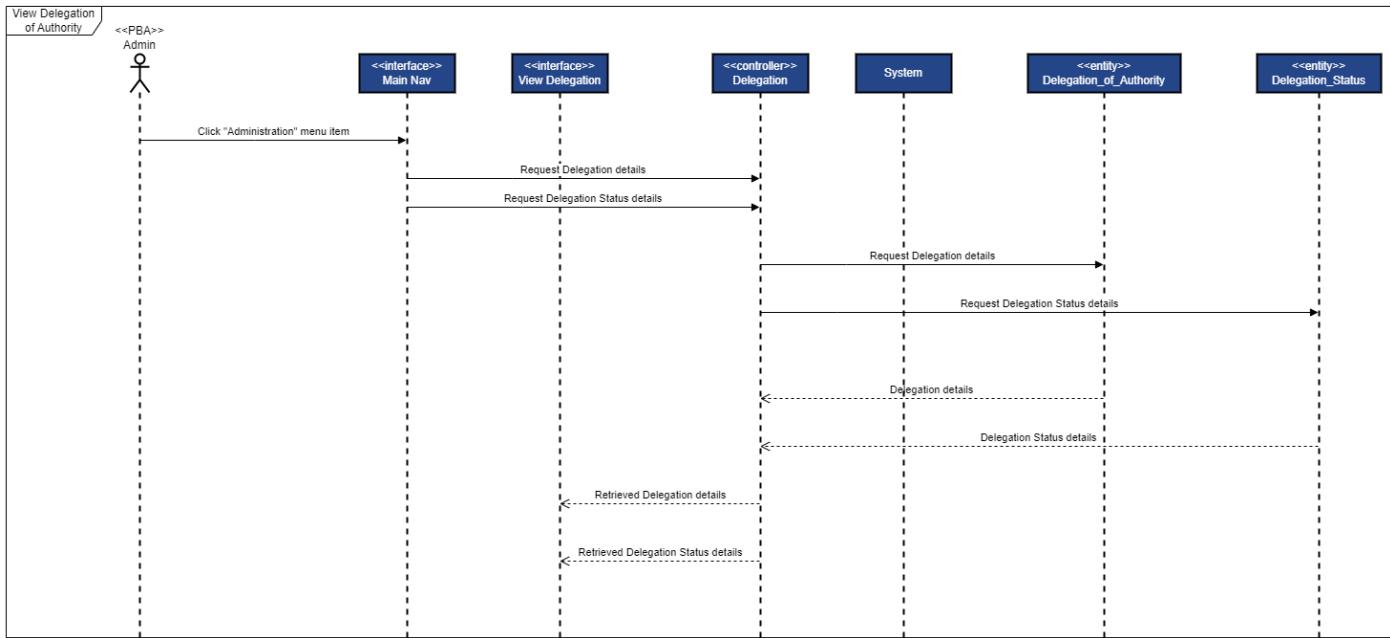


Figure 252 2.17 View Delegation of Authority Sequence Diagram

## 2.18 ASSIGN DELEGATION OF AUTHORITY

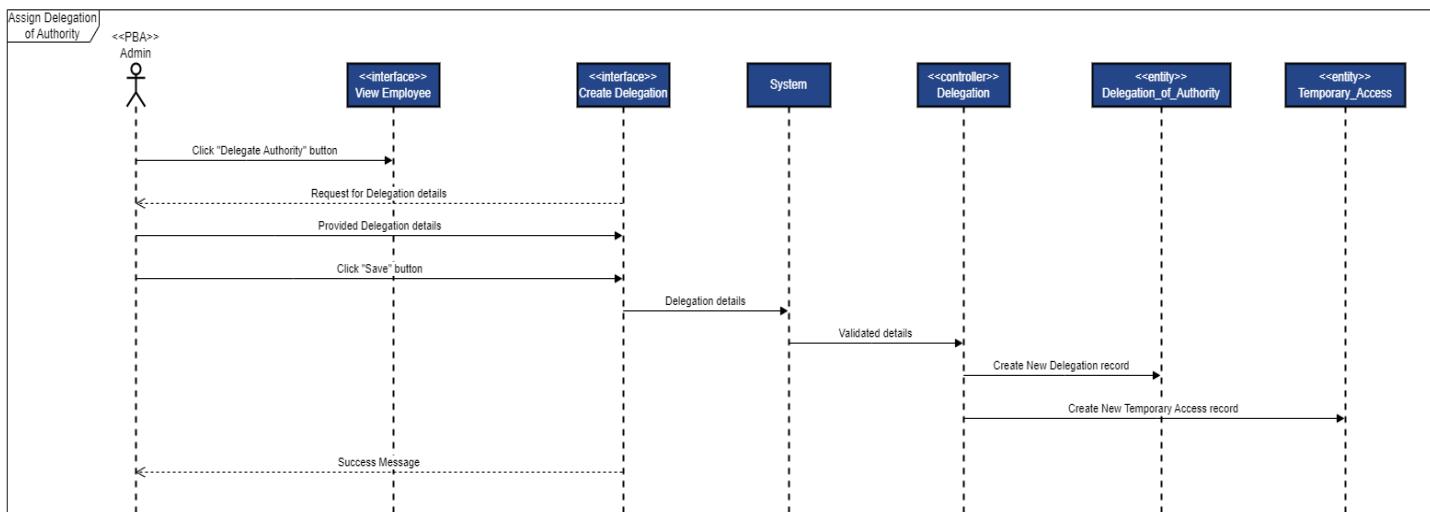


Figure 253 2.18 Assign Delegation of Authority Sequence Diagram

## 2.19 EDIT DELEGATION OF AUTHORITY

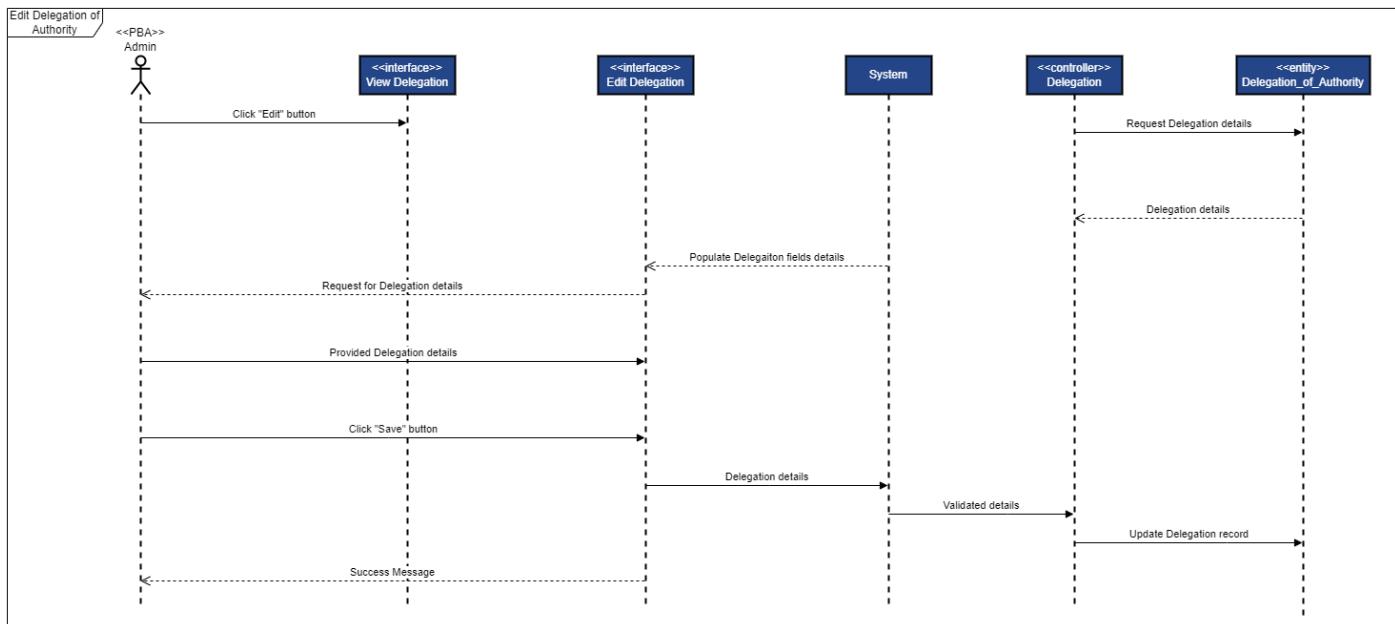


Figure 254 2.19 Edit Delegation of Authority Sequence Diagram

## 2.20 REVOKE ACCESS

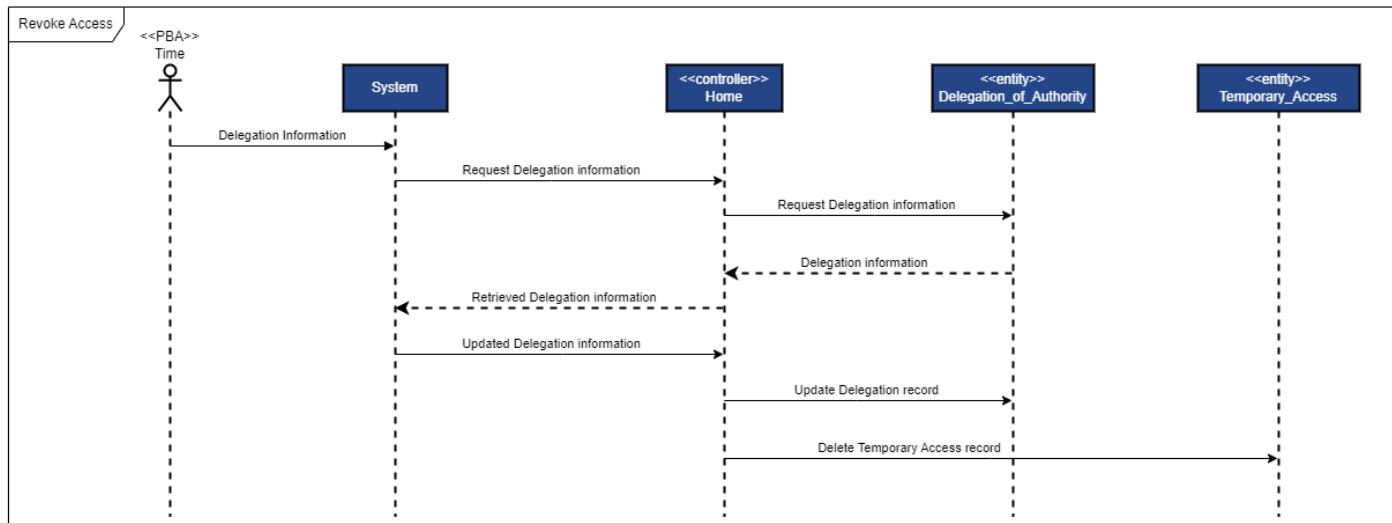


Figure 255 2.20 Revoke Access Sequence Diagram

### 2.35 DELETE DELEGATION OF AUTHORITY

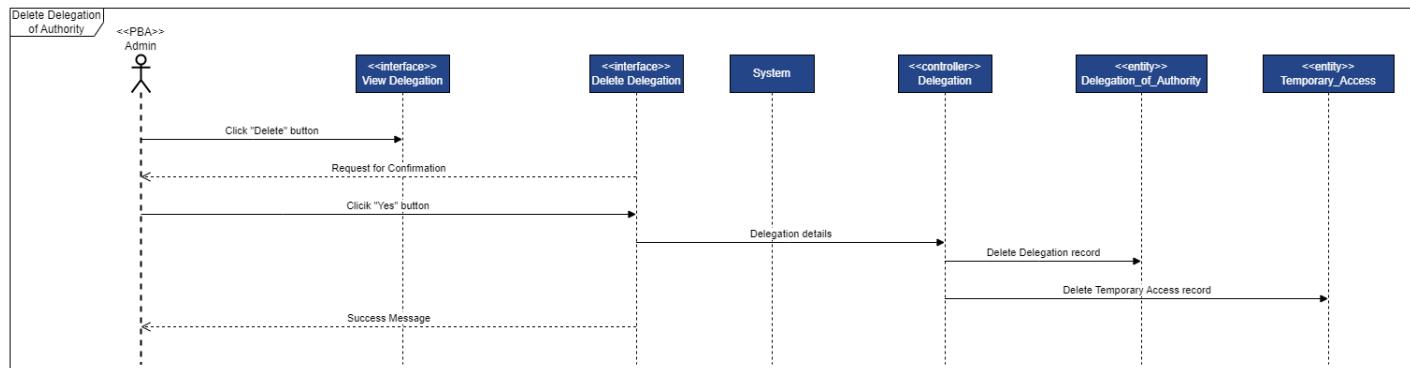


Figure 256 2.35 Delete Delegation of Authority Sequence Diagram

### 2.36 ACTIVATE DELEGATION OF AUTHORITY

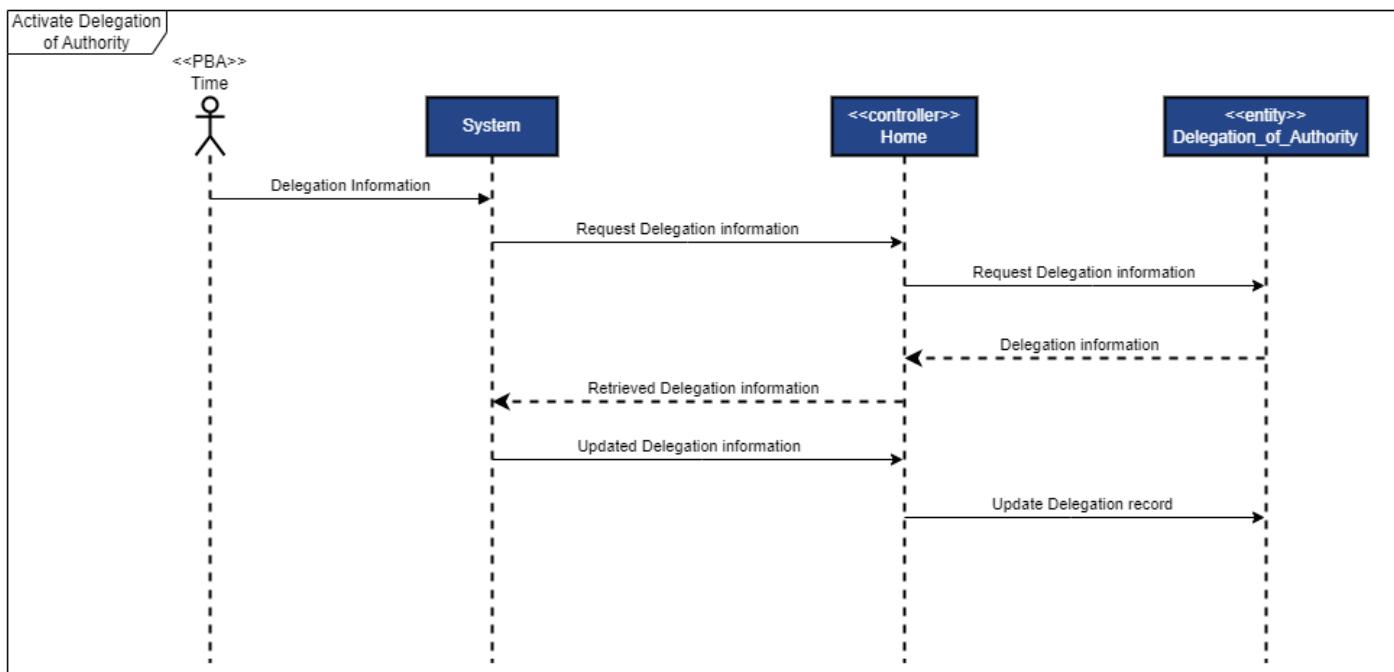


Figure 257 2.36 Activate Delegation of Authority Sequence Diagram

USE CASE 1.7 & 2.25 & 2.27-2.30

### 1.7 VIEW USER MANUAL

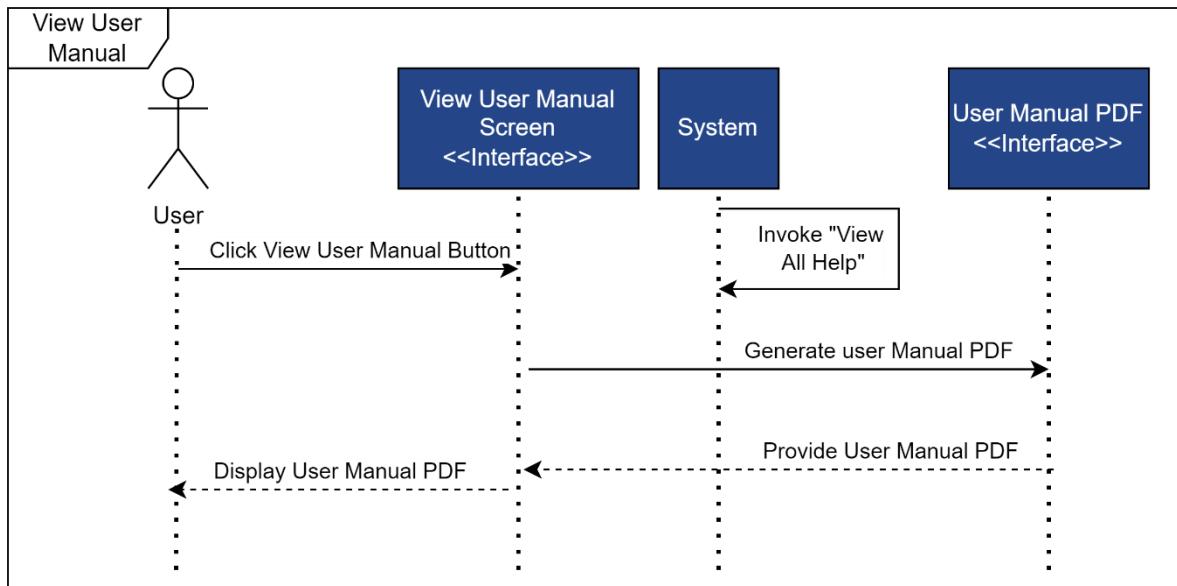


Figure 258 1.7 View User Manual Sequence Diagram

### 2.25 BACKUP SYSTEM DATA

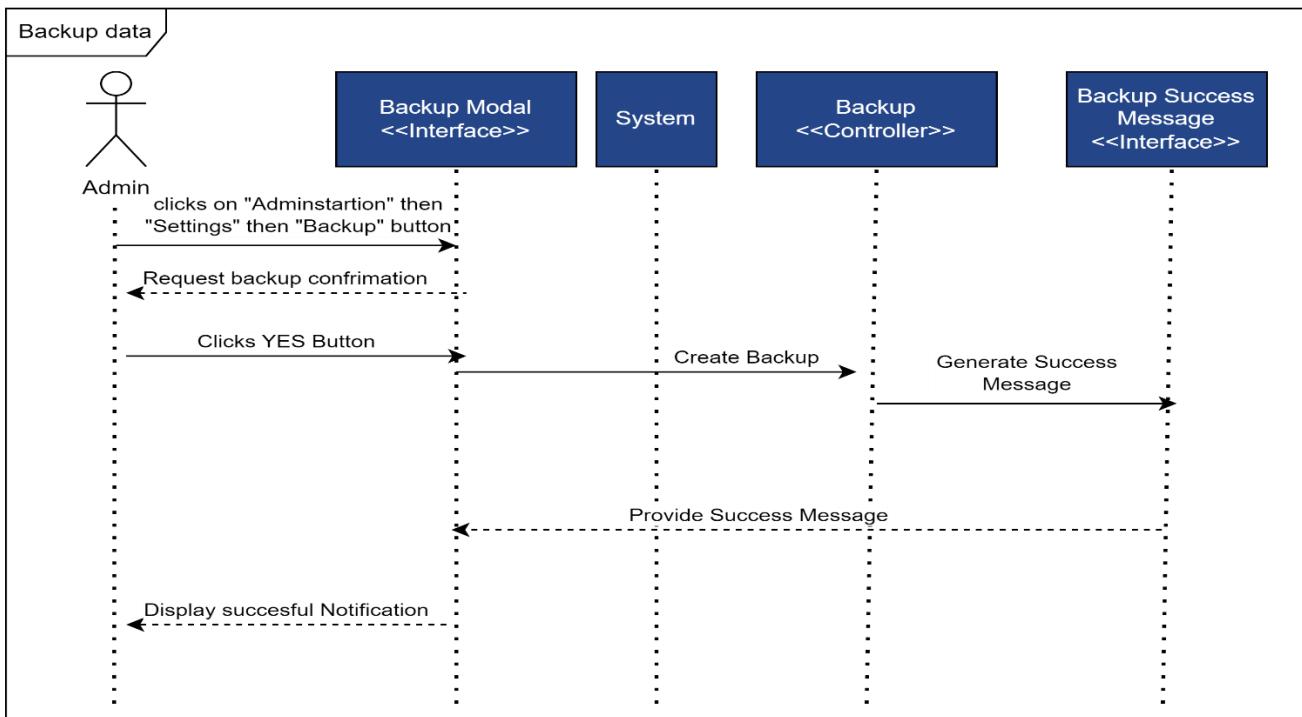


Figure 259 2.25 Backup System data Sequence Diagram

### 2.27 ADD HELP

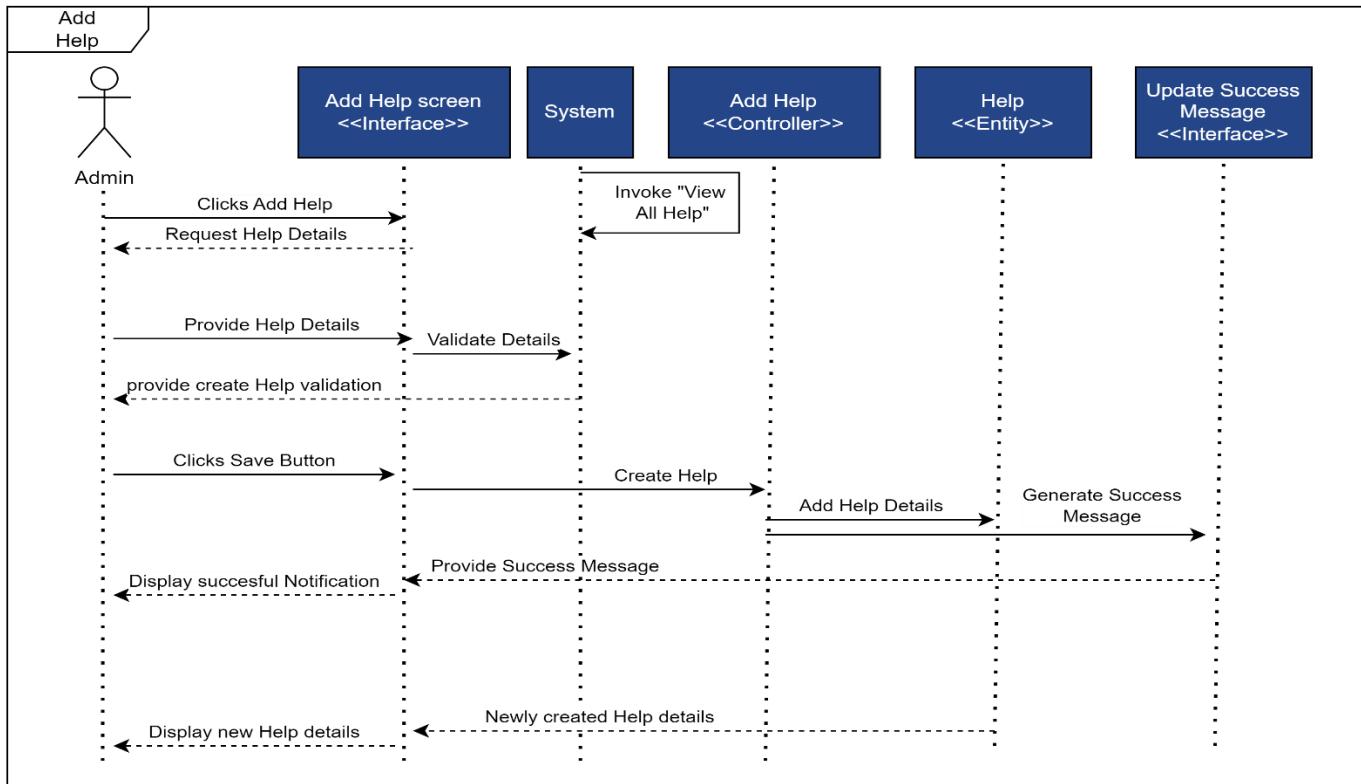


Figure 260 2.27 Add Help Sequence Diagram

## 2.28 UPDATE HELP

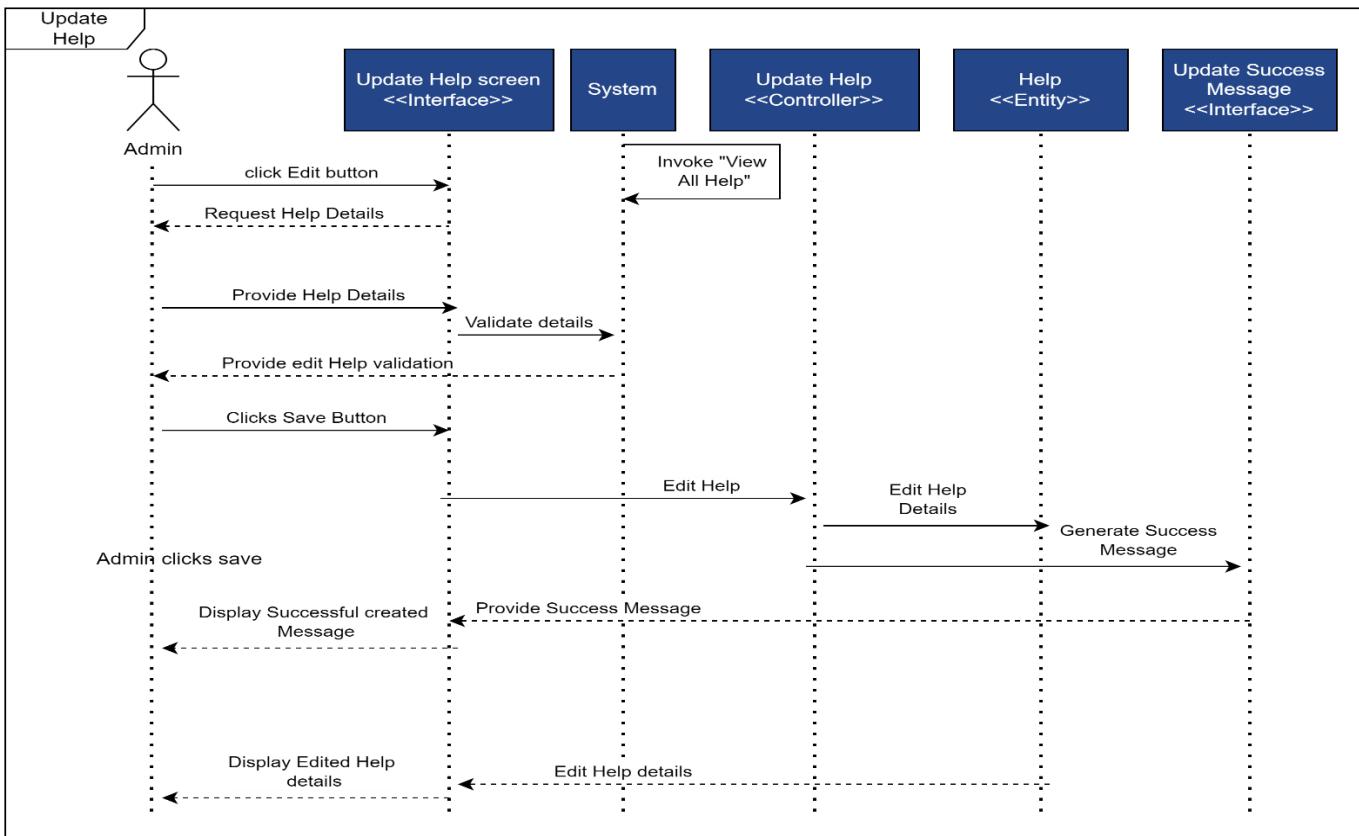


Figure 261 2.28 Update Help Sequence Diagram

## 2.29 DELETE HELP

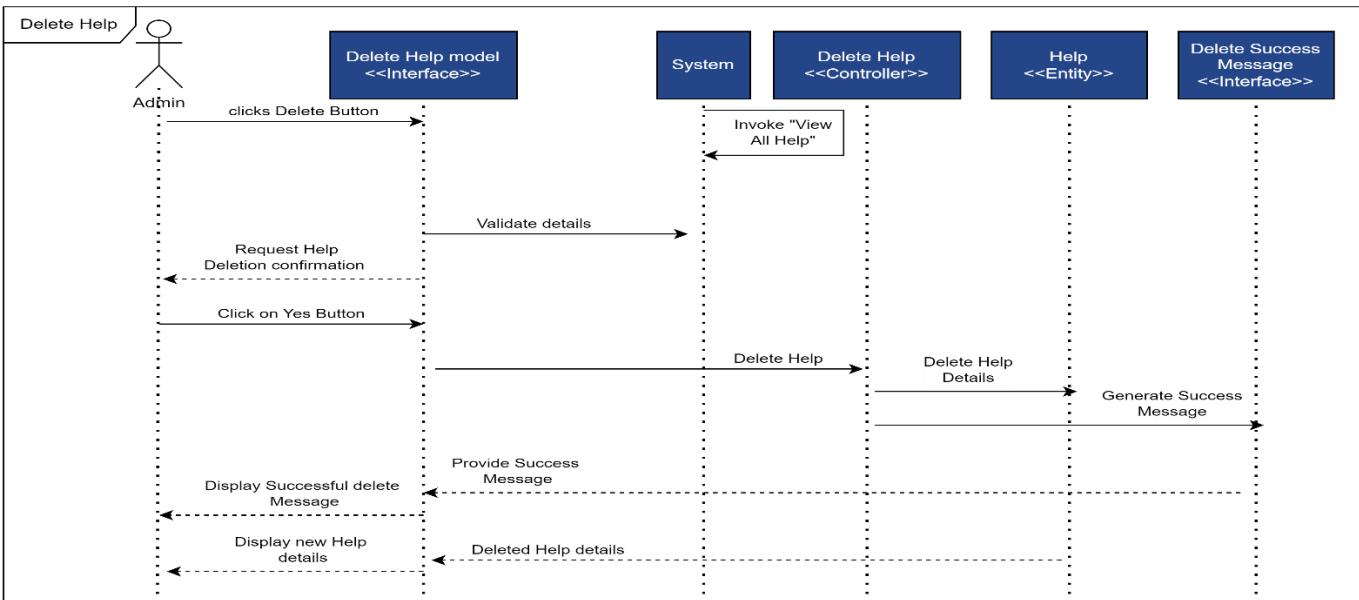


Figure 262 2.29 Delete Help Sequence Diagram

## 2.30 VIEW ALL HELP

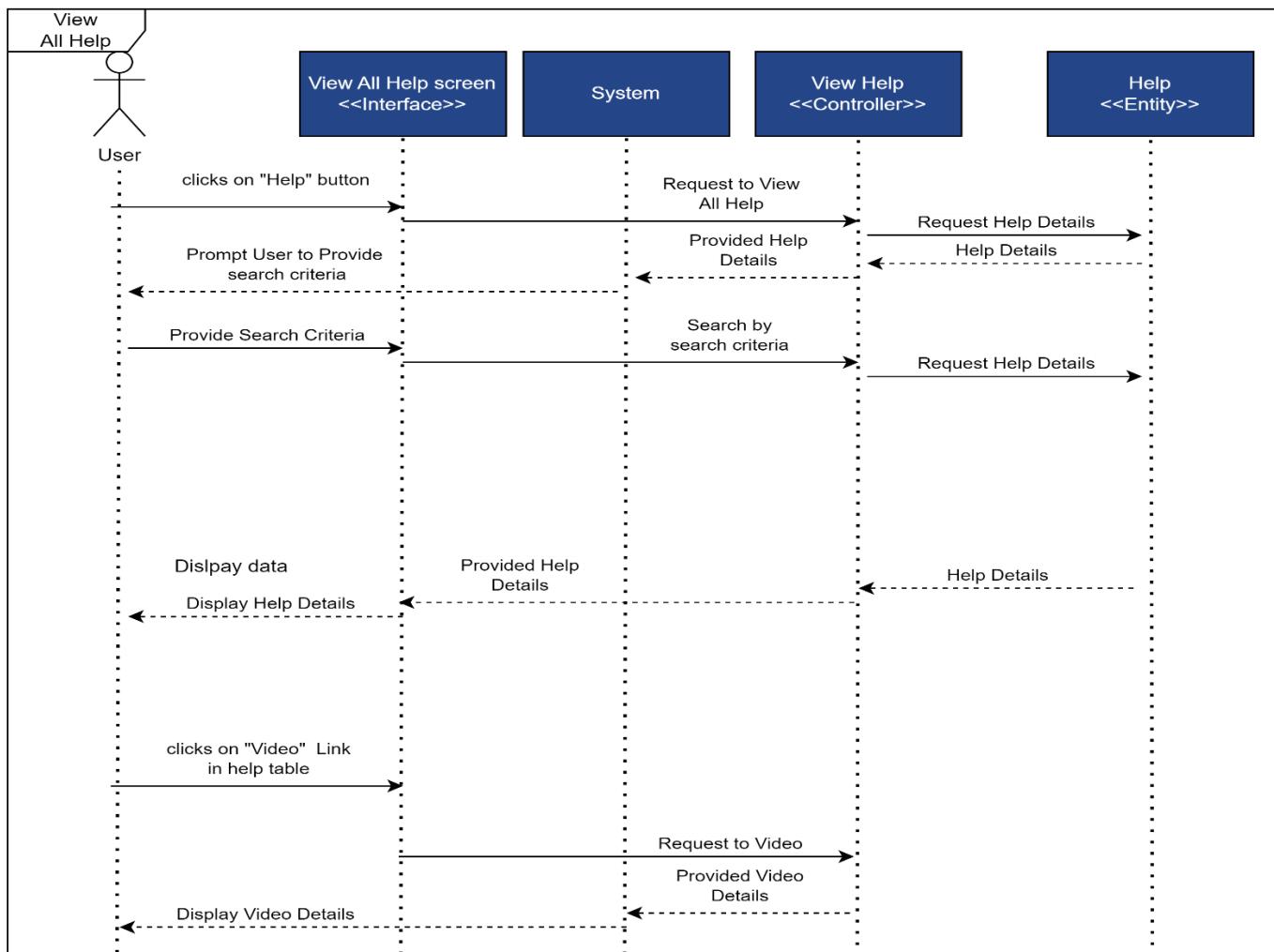


Figure 263 2.30 View All Help Sequence Diagram

### USE CASE 3.8 – 3.12 & 4.1-4.6

#### 3.8 RECEIVE PROCUREMENT ITEM

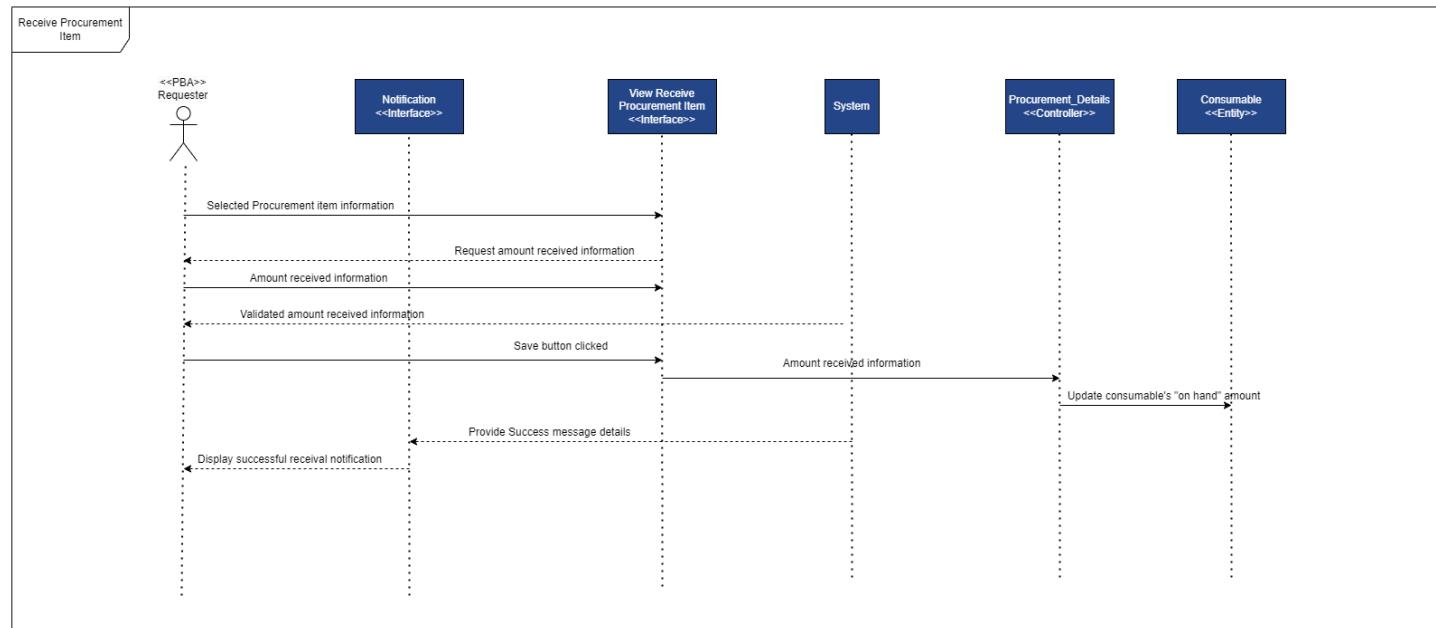


Figure 264 3.8 Receive Procurement Item Sequence Diagram

### 3.9 UPLOAD TAX INVOICE

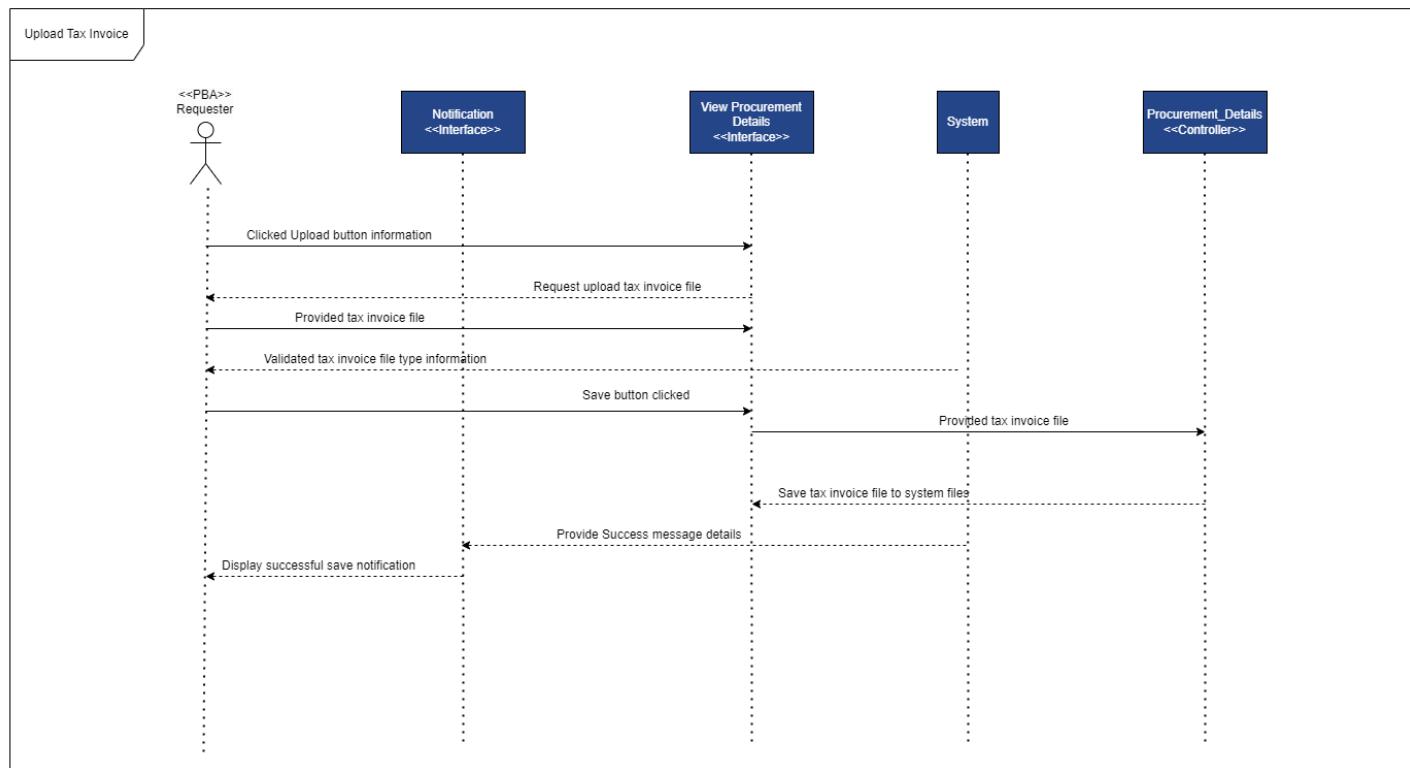


Figure 265 3.9 Upload Tax Invoice Sequence Diagram

### 3.10 BUDGET OWNER REQUISITION APPROVAL

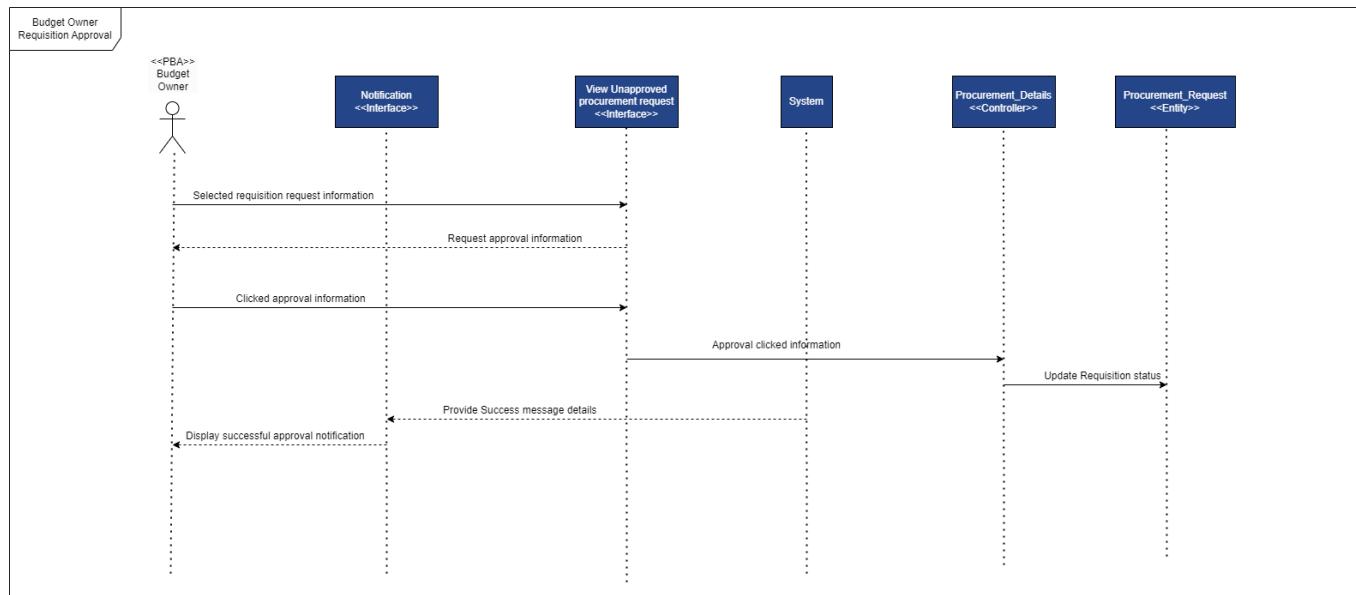


Figure 266 3.10 Budget Owner Requisition Approval Sequence Diagram

### 3.11 UPLOAD RECEIPT

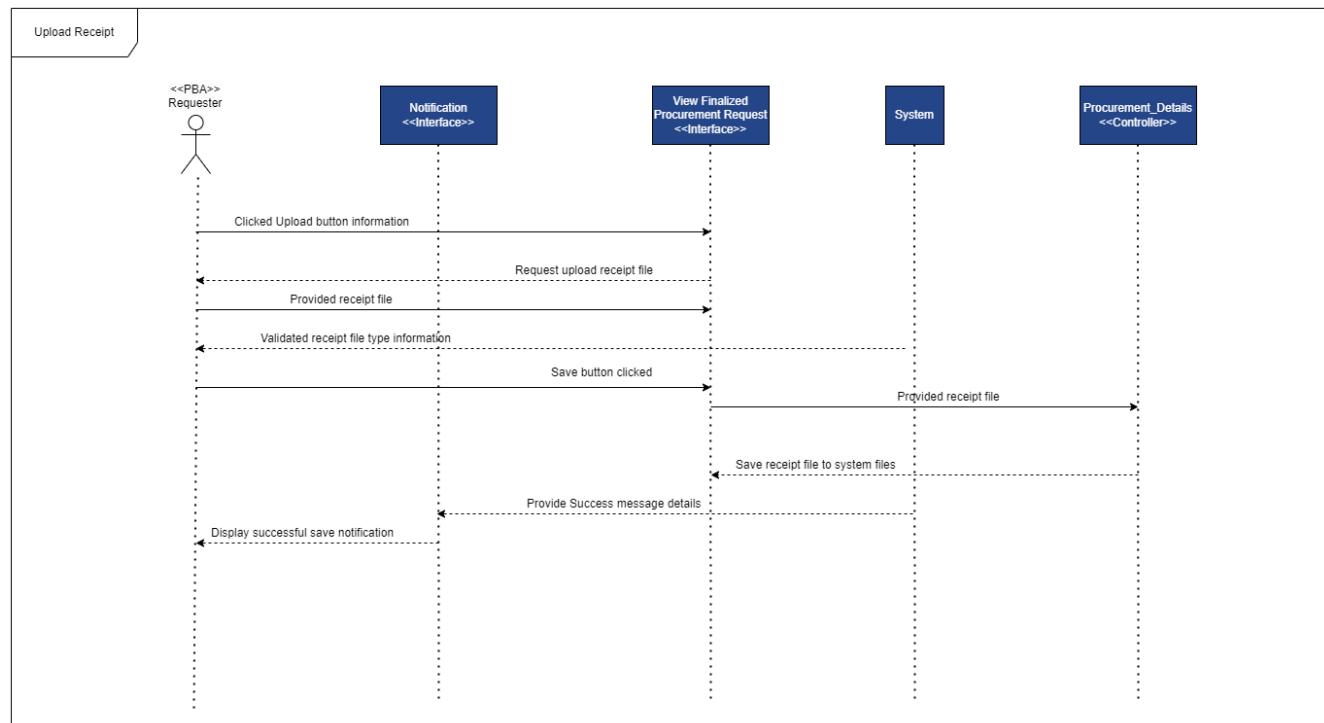


Figure 267 3.11 Upload Receipt Sequence Diagram

### 3.12 UPLOAD CREDIT CARD PAYMENT

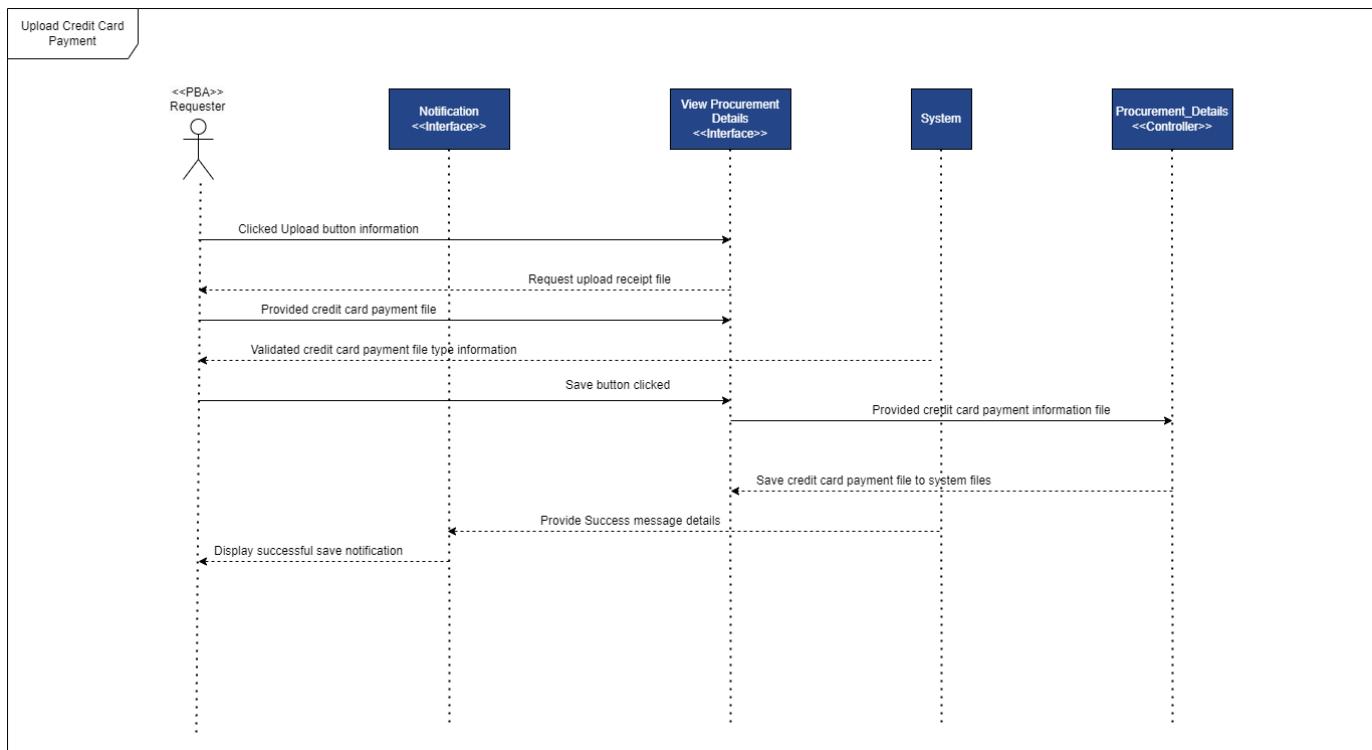


Figure 268 3.12 Upload Credit Card Payment Sequence Diagram

### 3.13 VIEW PROCUREMENT DETAILS

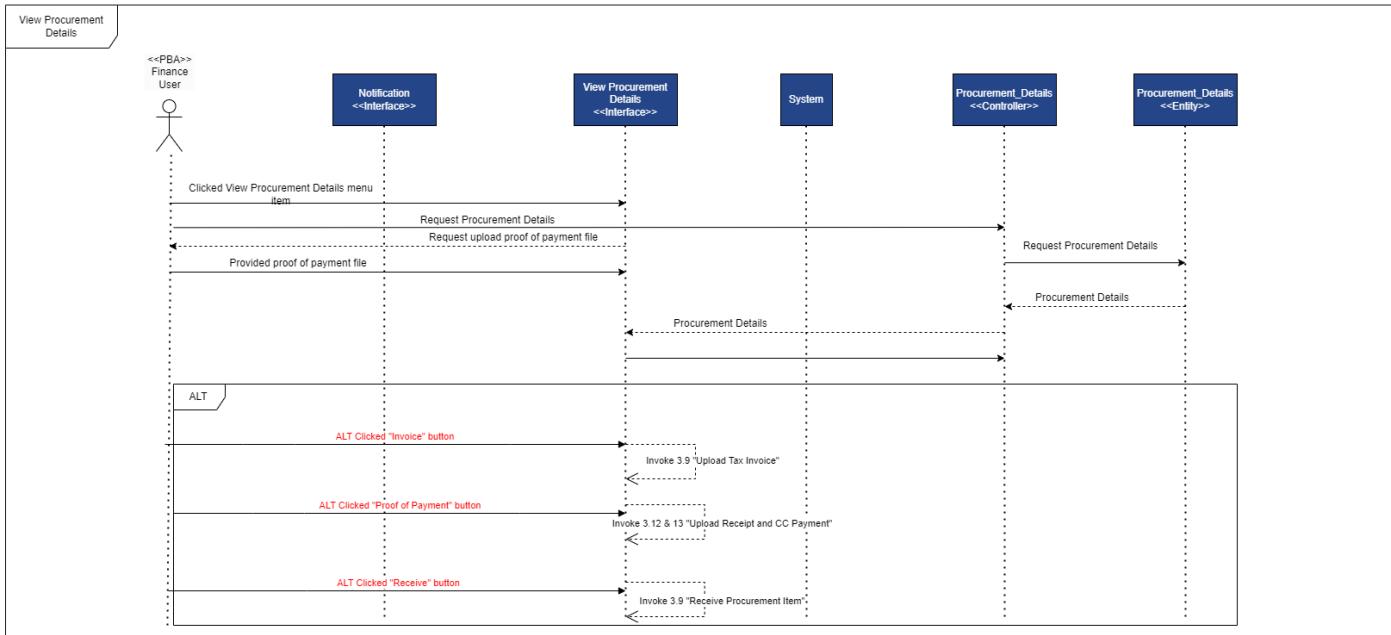


Figure 269: 3.13 View Procurement Details

### 4.1 FINALIZE PROCUREMENT REQUEST

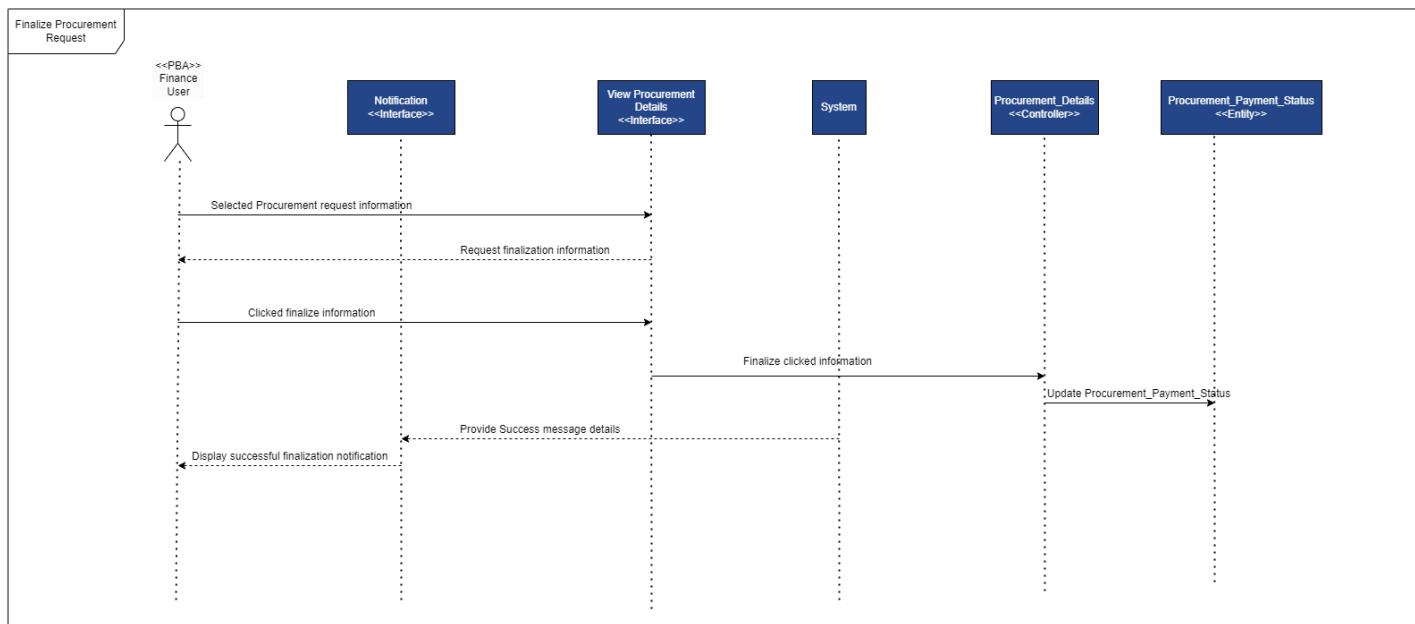


Figure 270 4.1 Finalize Procurement Request Sequence Diagram

### 4.2 UPLOAD PROOF OF PAYMENT

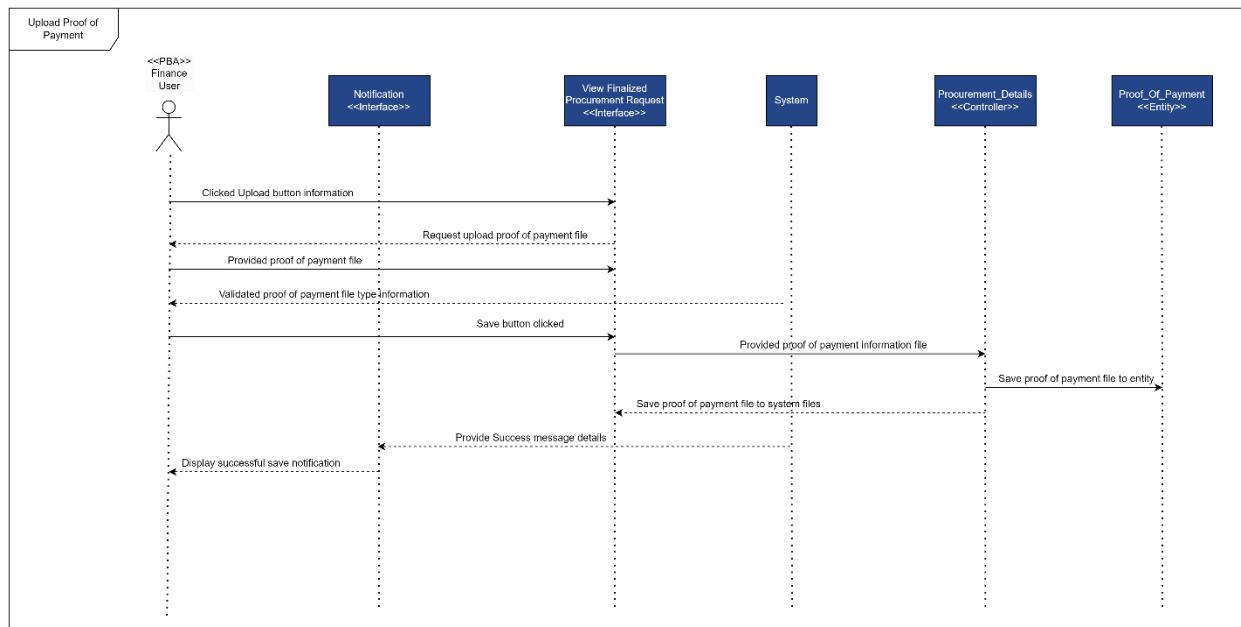


Figure 271 4.2 Upload Proof Of Payment Sequence Diagram

### 4.3 CREATE BUDGET ALLOCATION

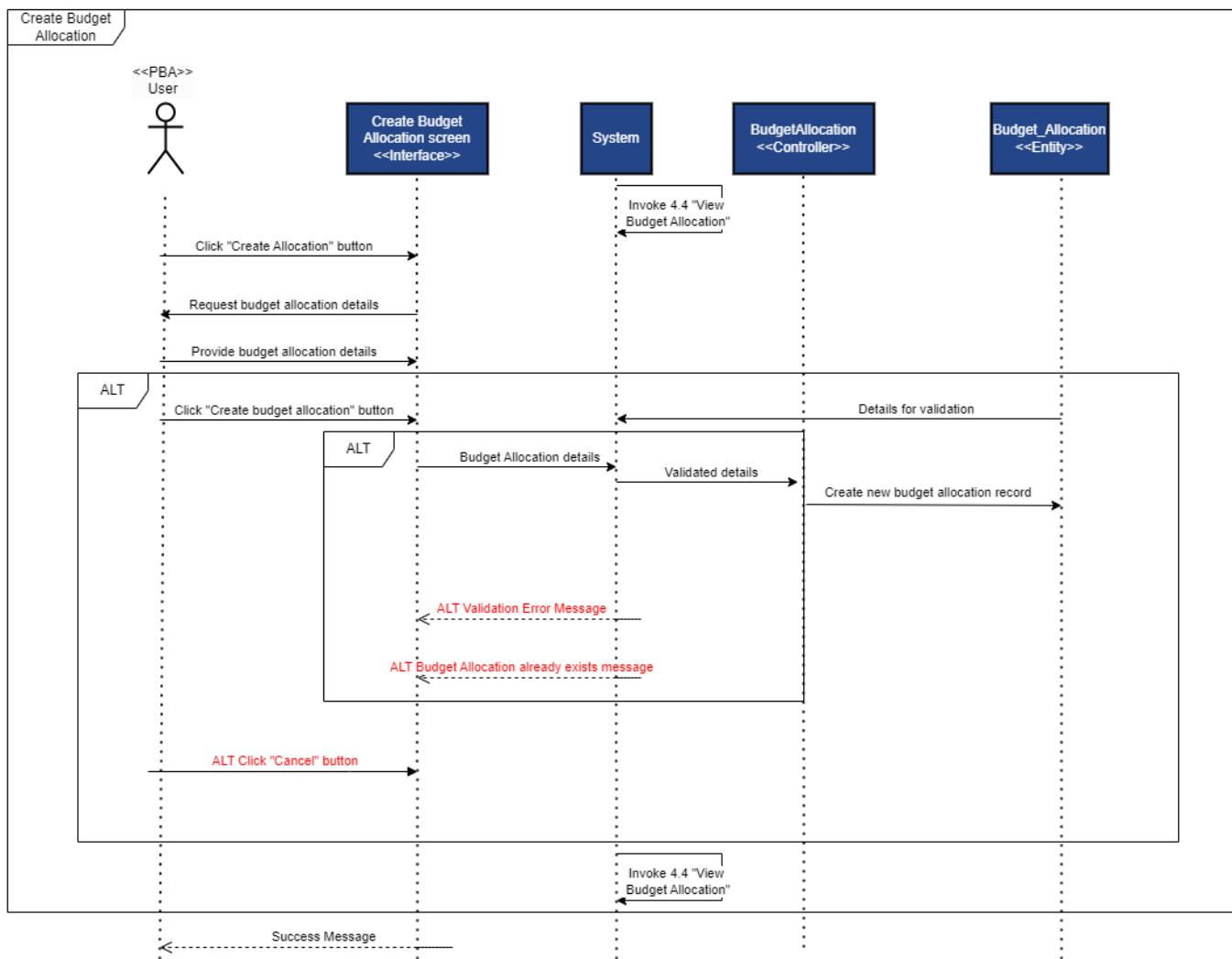


Figure 272 4.3 Create Budget Allocation Sequence Diagram

### 4.4 VIEW BUDGET ALLOCATION

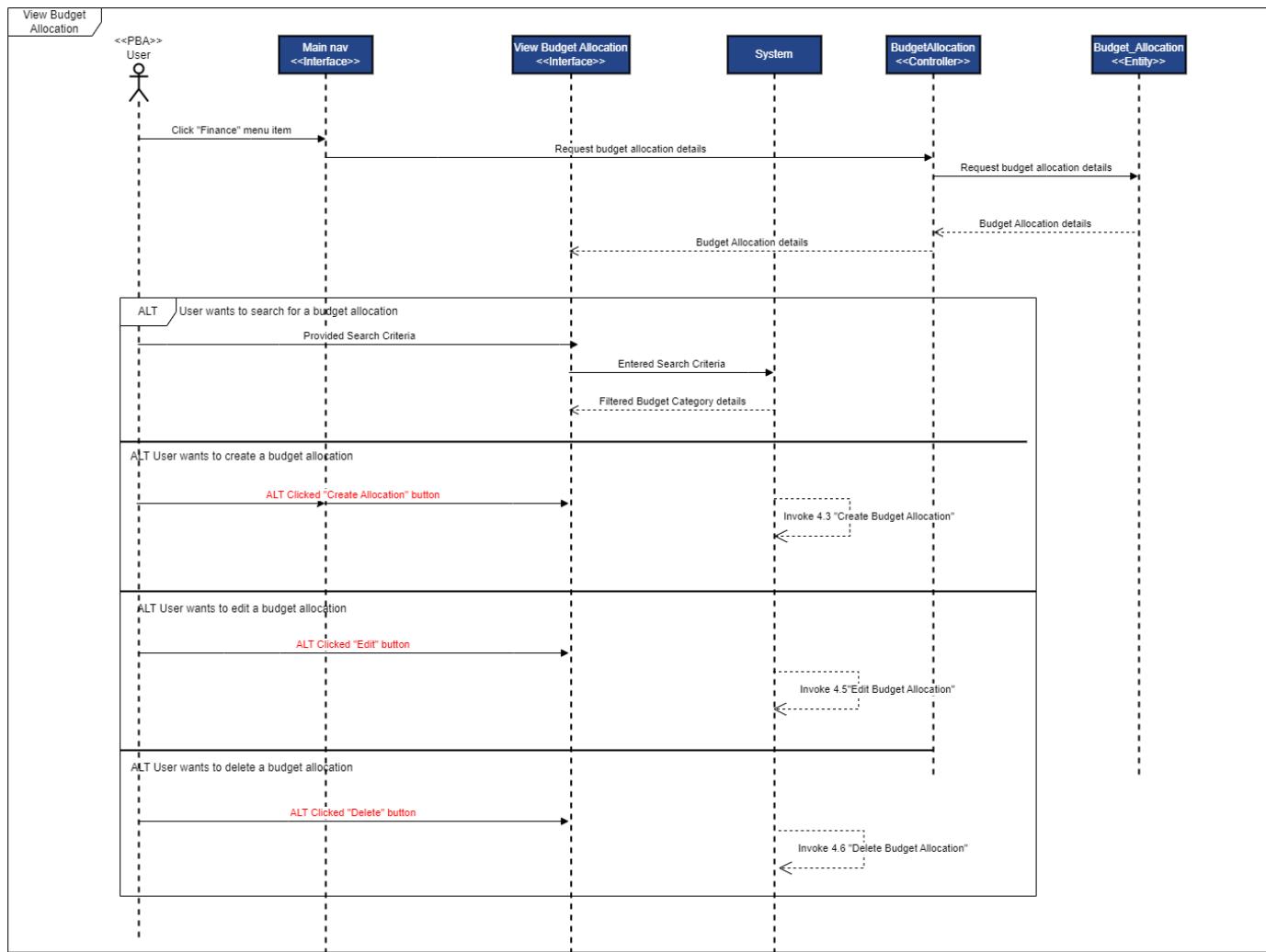


Figure 273 4.4 View Budget Allocation Sequence Diagram

### 4.5 UPDATE BUDGET ALLOCATION

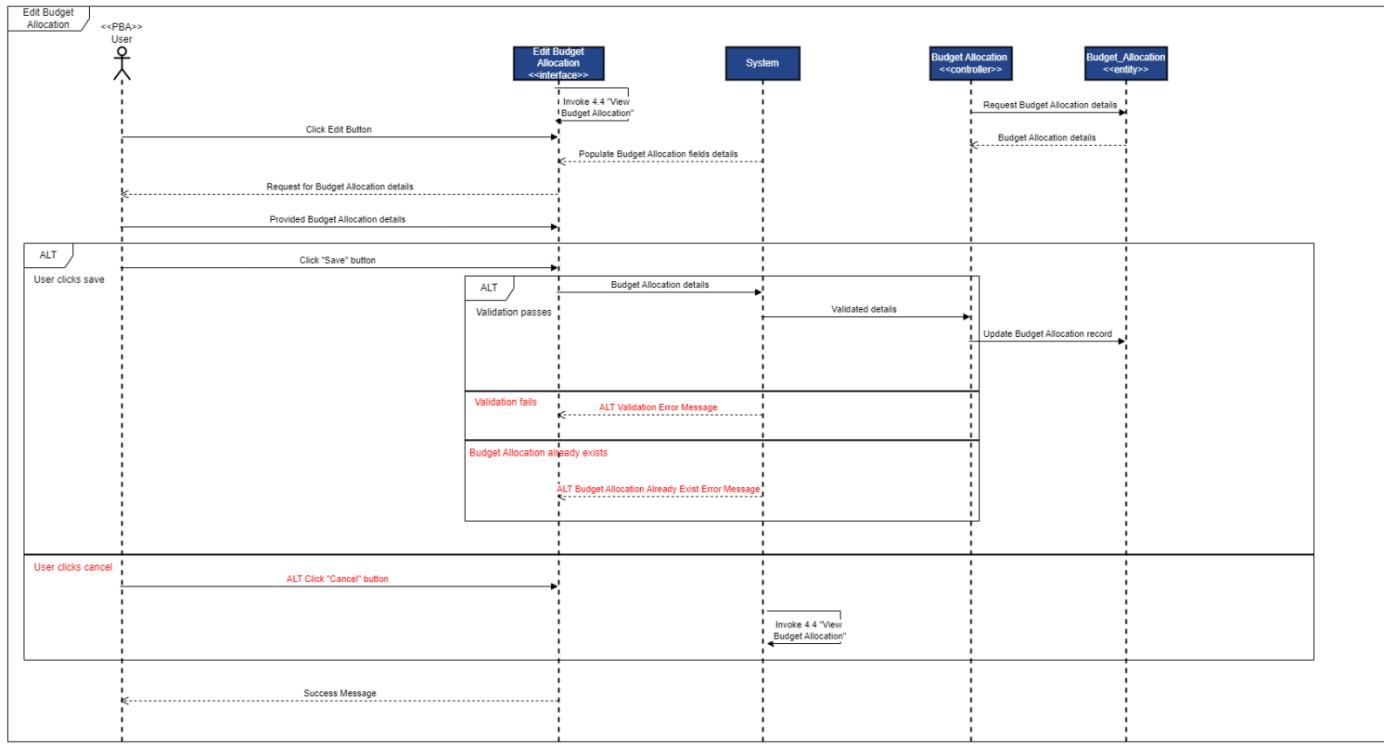


Figure 274 4.5 Update Budget Allocation Sequence Diagram

### 4.6 DELETE BUDGET ALLOCATION

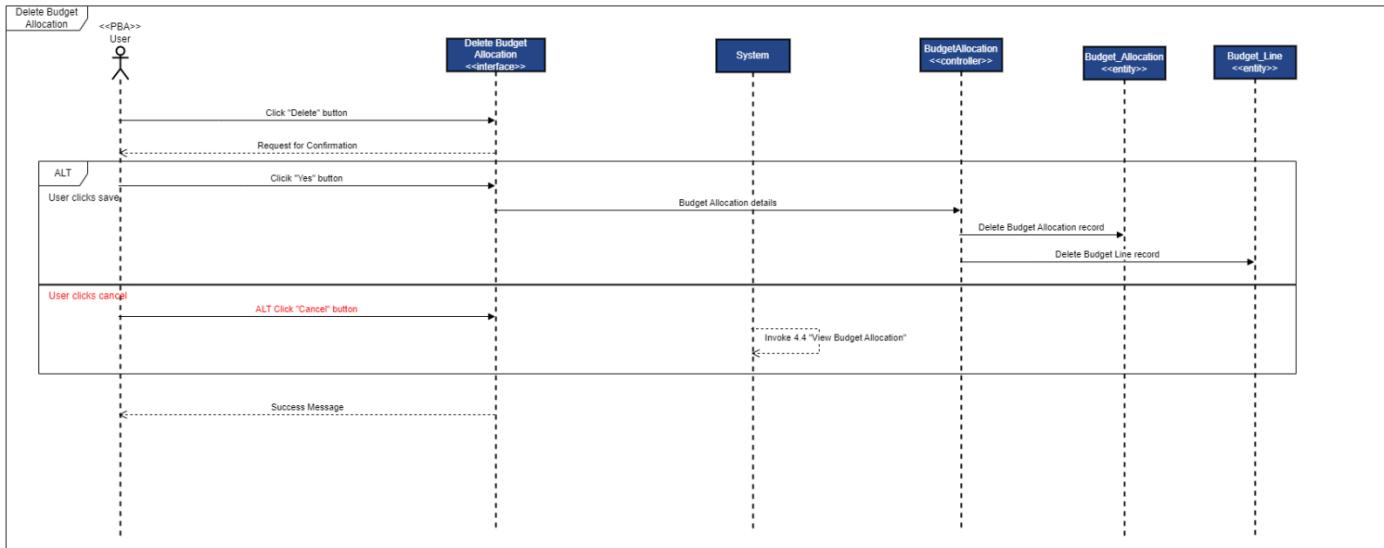


Figure 275 4.6 Delete Budget Allocation Sequence Diagram

## USE CASE 5.1-5.8 & 5.9, 5.10 & 1.1-1.4 & 3.1-3.4

### 5.1 CREATE CONSUMABLE

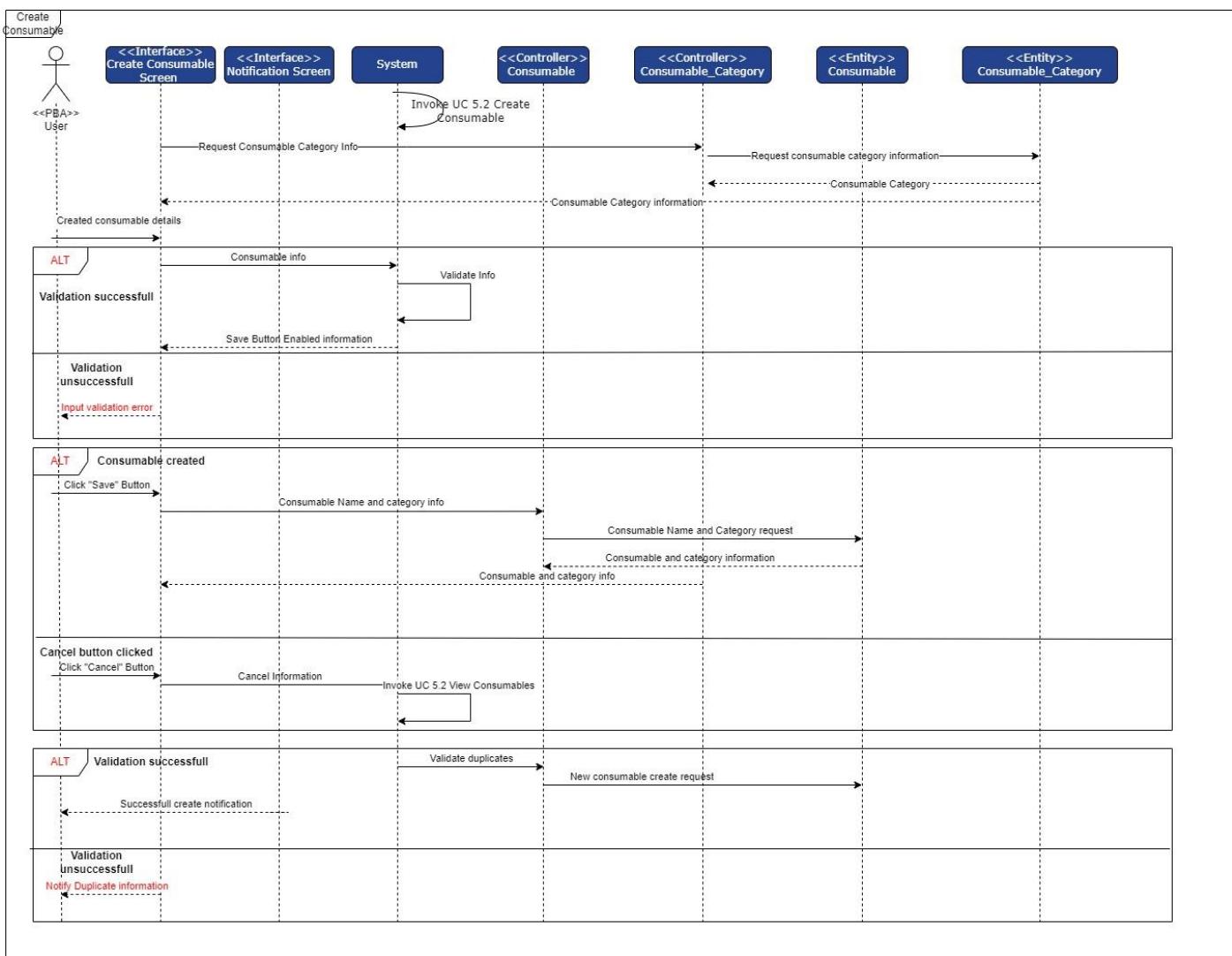


Figure 276 5.1 Create Consumable Sequence Diagram

## 5.2 VIEW CONSUMABLES

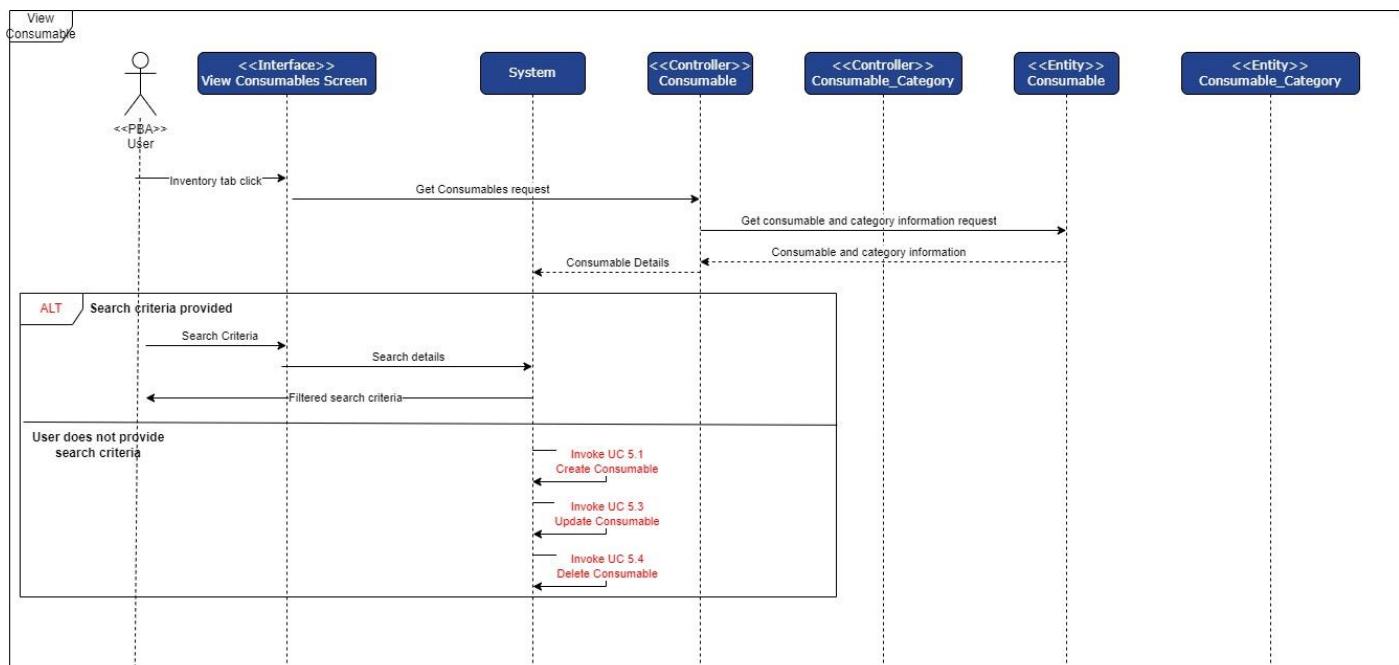


Figure 277 5.2 View Consumable Sequence Diagram

### 5.3 UPDATE CONSUMABLE

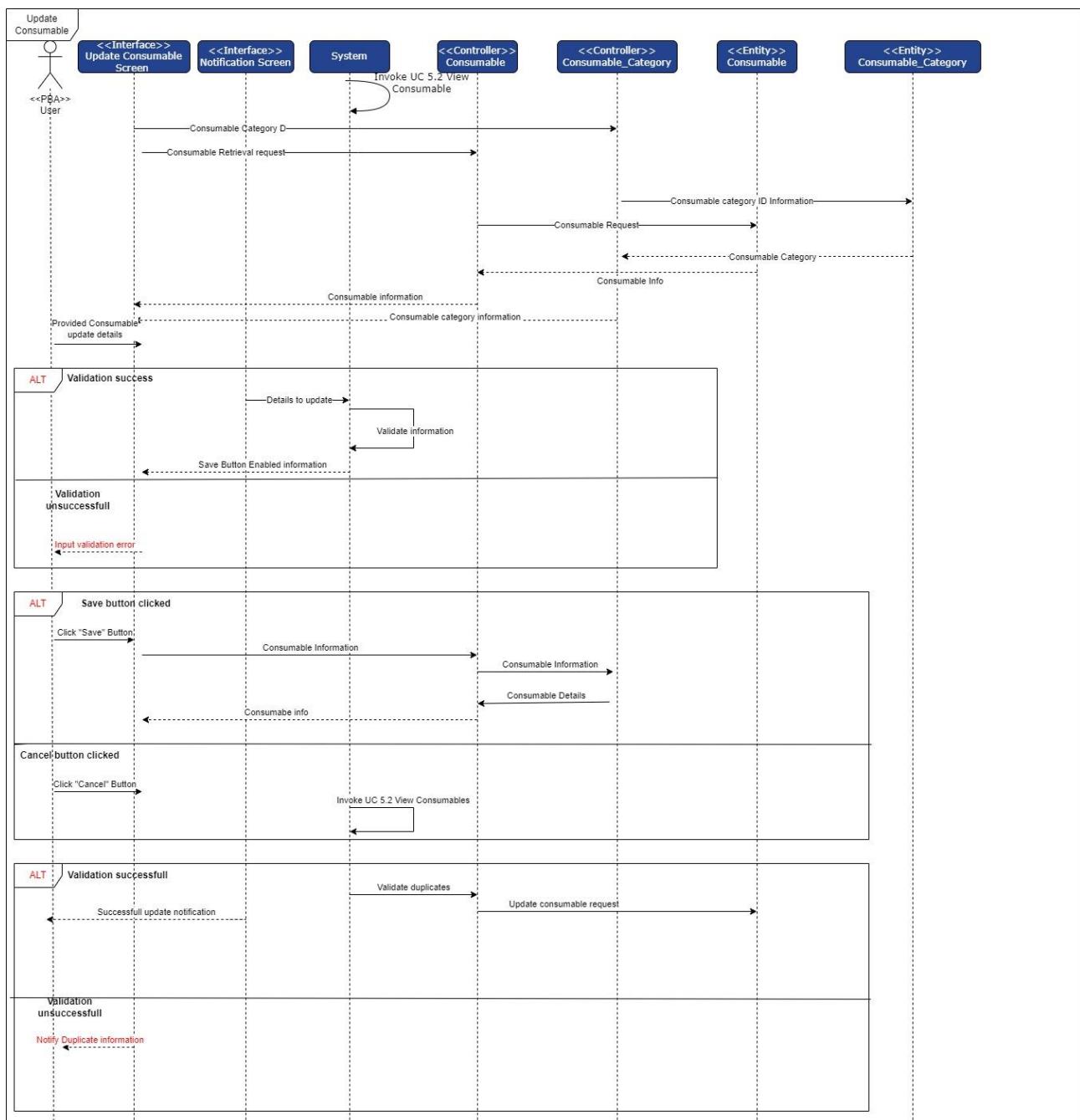


Figure 278 5.3 Update Consumable Sequence Diagram

### 5.4 DELETE CONSUMABLE

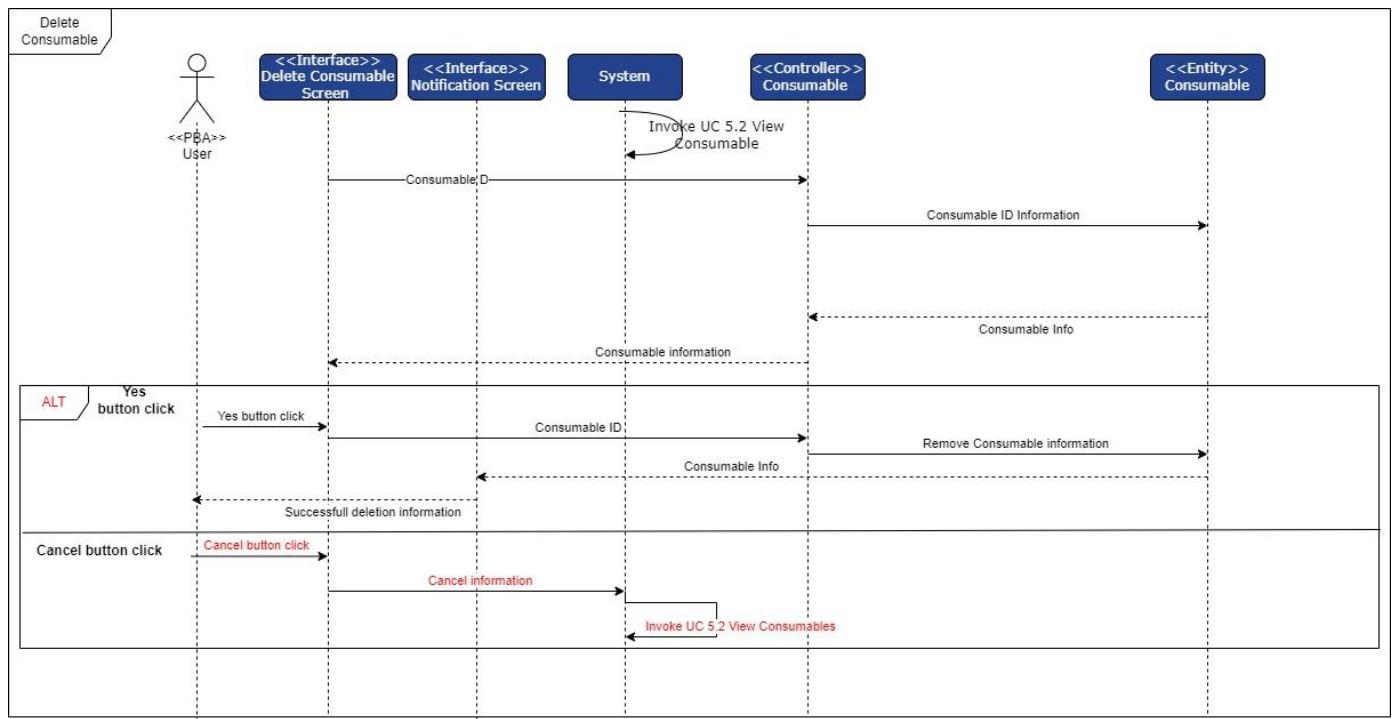


Figure 279 5.4 Delete Consumable Sequence Diagram

### 5.5 CREATE CONSUMABLE CATEGORY

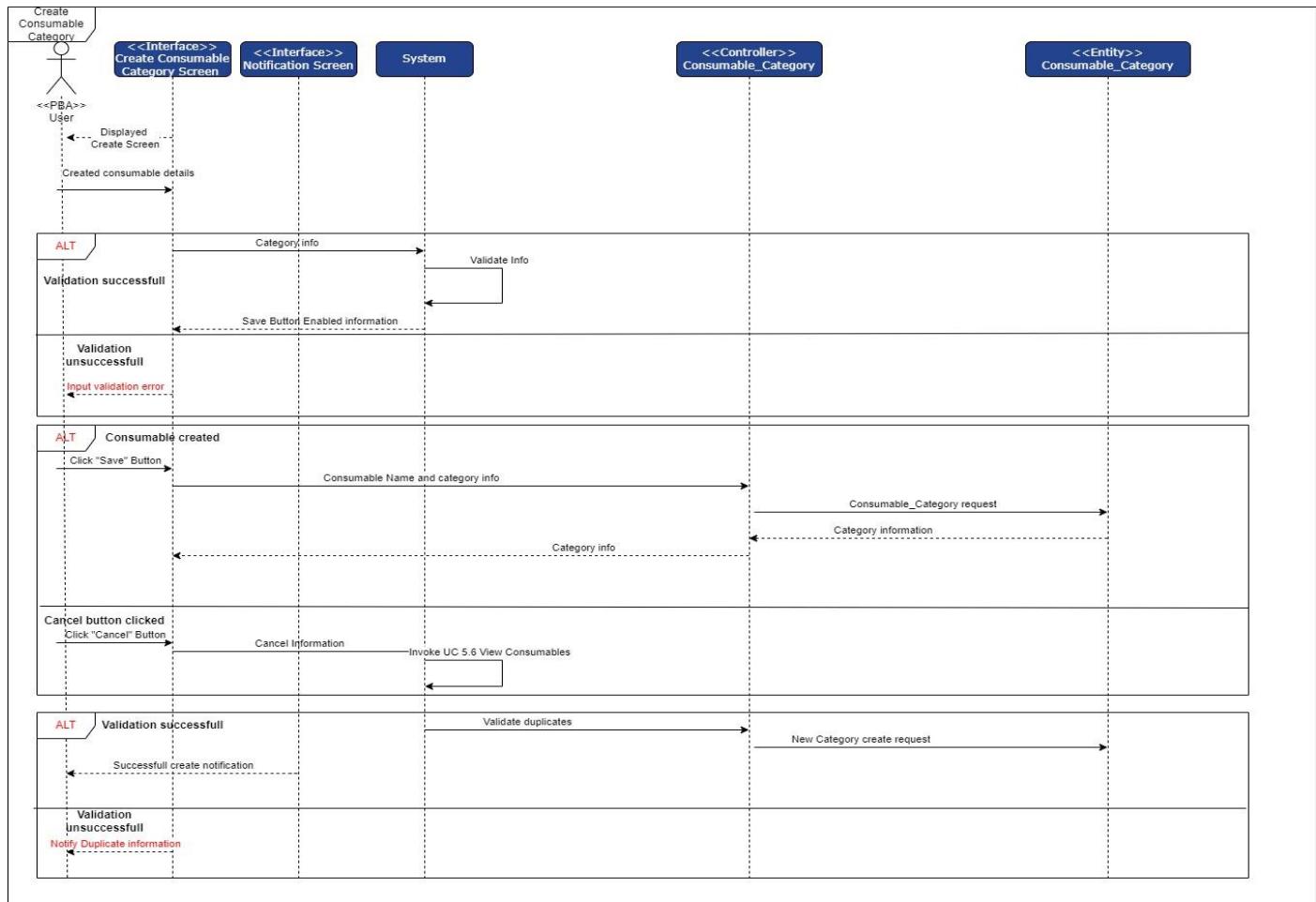


Figure 280 5.5 Create Consumable Category Sequence Diagram

### 5.6 VIEW CONSUMABLE CATEGORIES

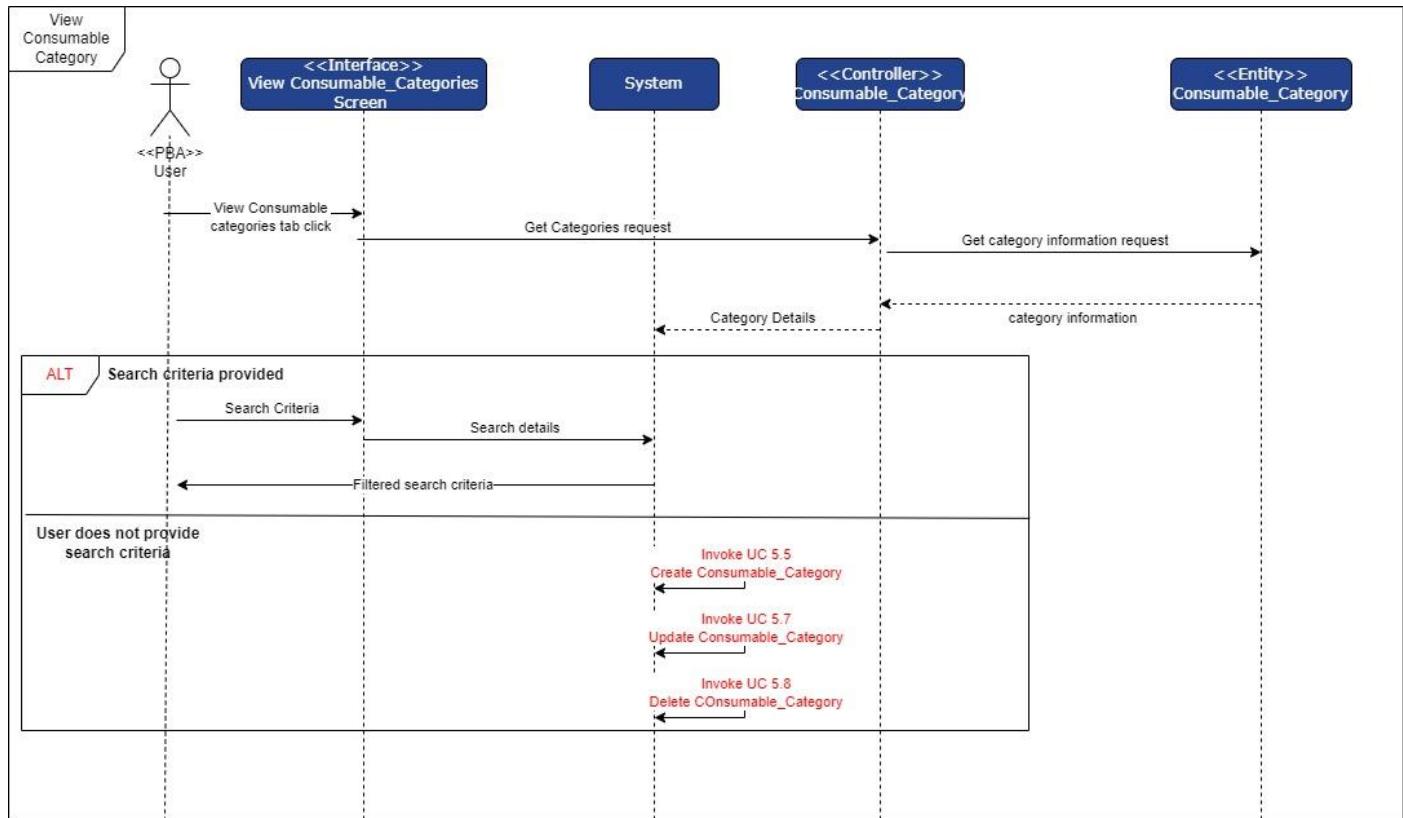


Figure 281 5.6 View Consumable Category Sequence Diagram

### 5.7 UPDATE CONSUMABLE CATEGORY

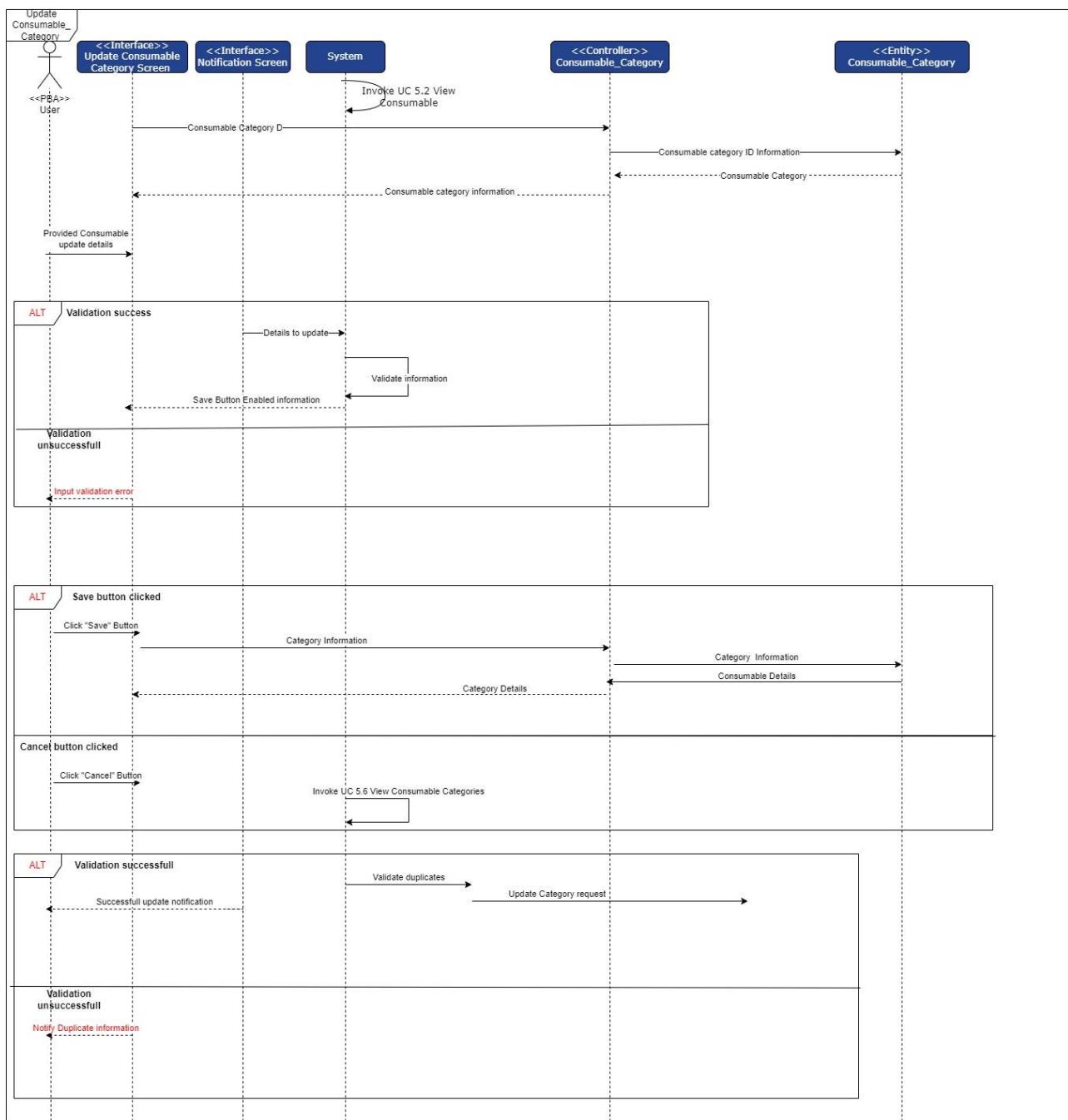


Figure 282 5.7 Update Consumable Category Sequence Diagram

### 5.8 DELETE CONSUMABLE CATEGORY

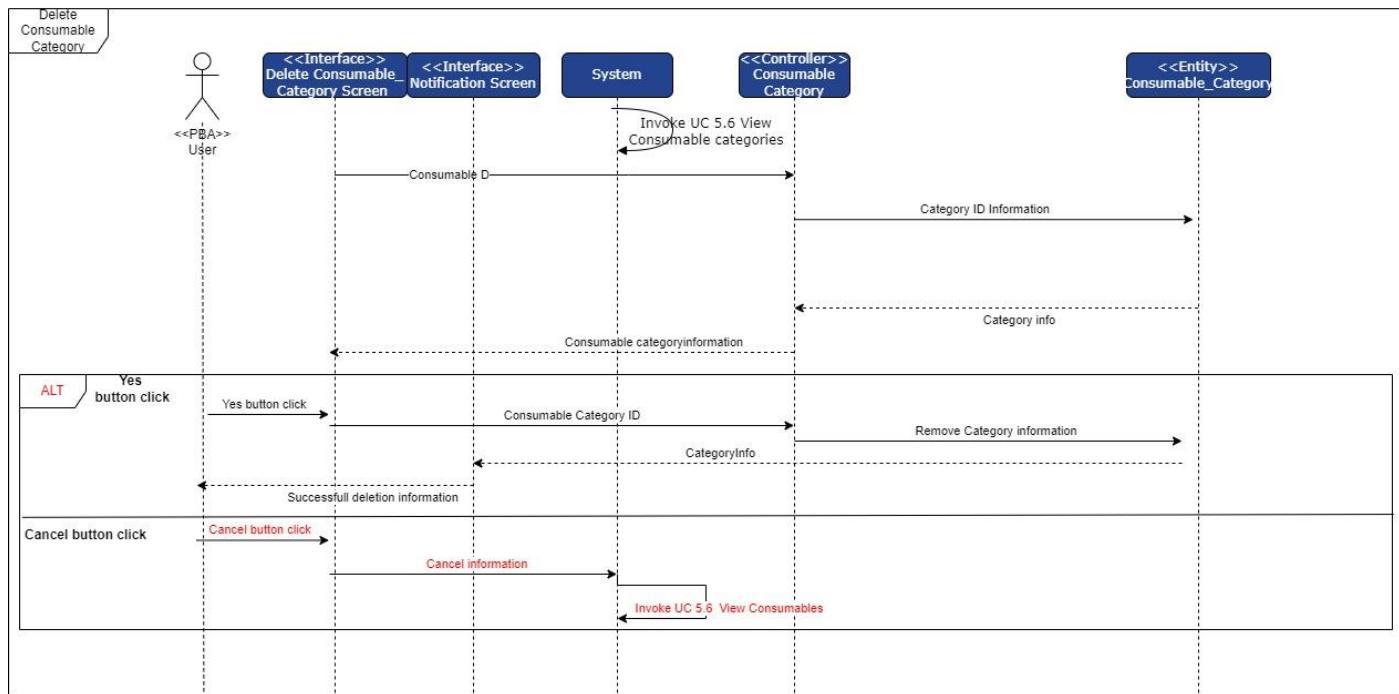


Figure 283 5.8 Delete Consumable Category Sequence Diagram

### 5.9 EXPORT INVENTORY DETAILS

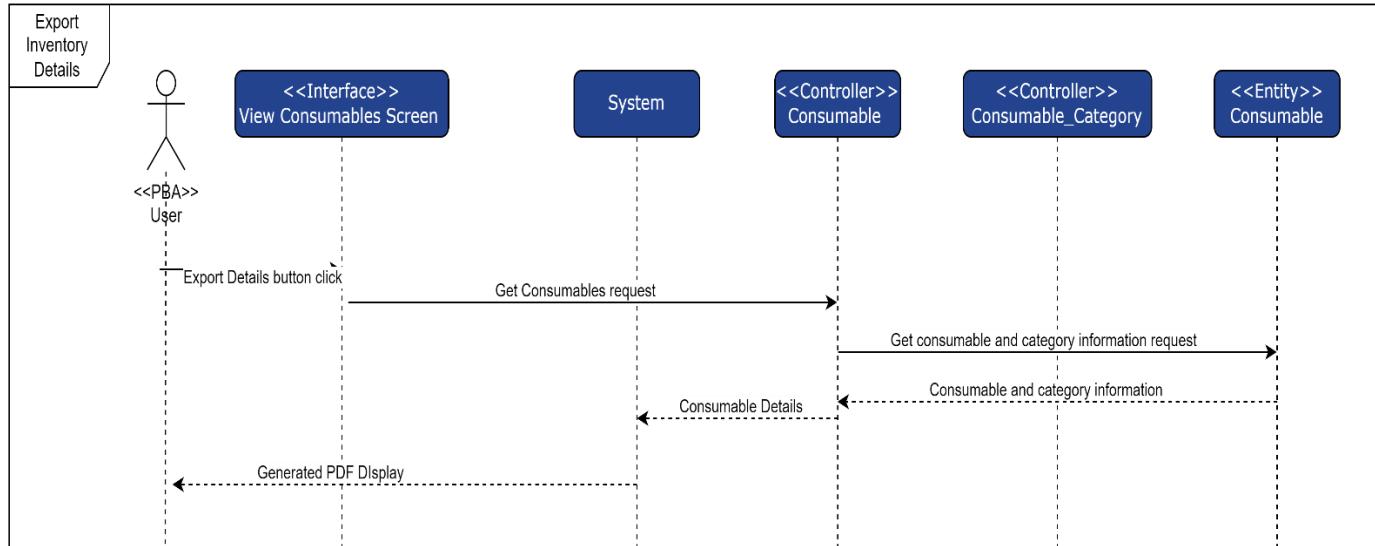


Figure 284 5.9.Export Inventory Details Sequence Diagram

## 5.10 UPDATE STOCK DETAILS

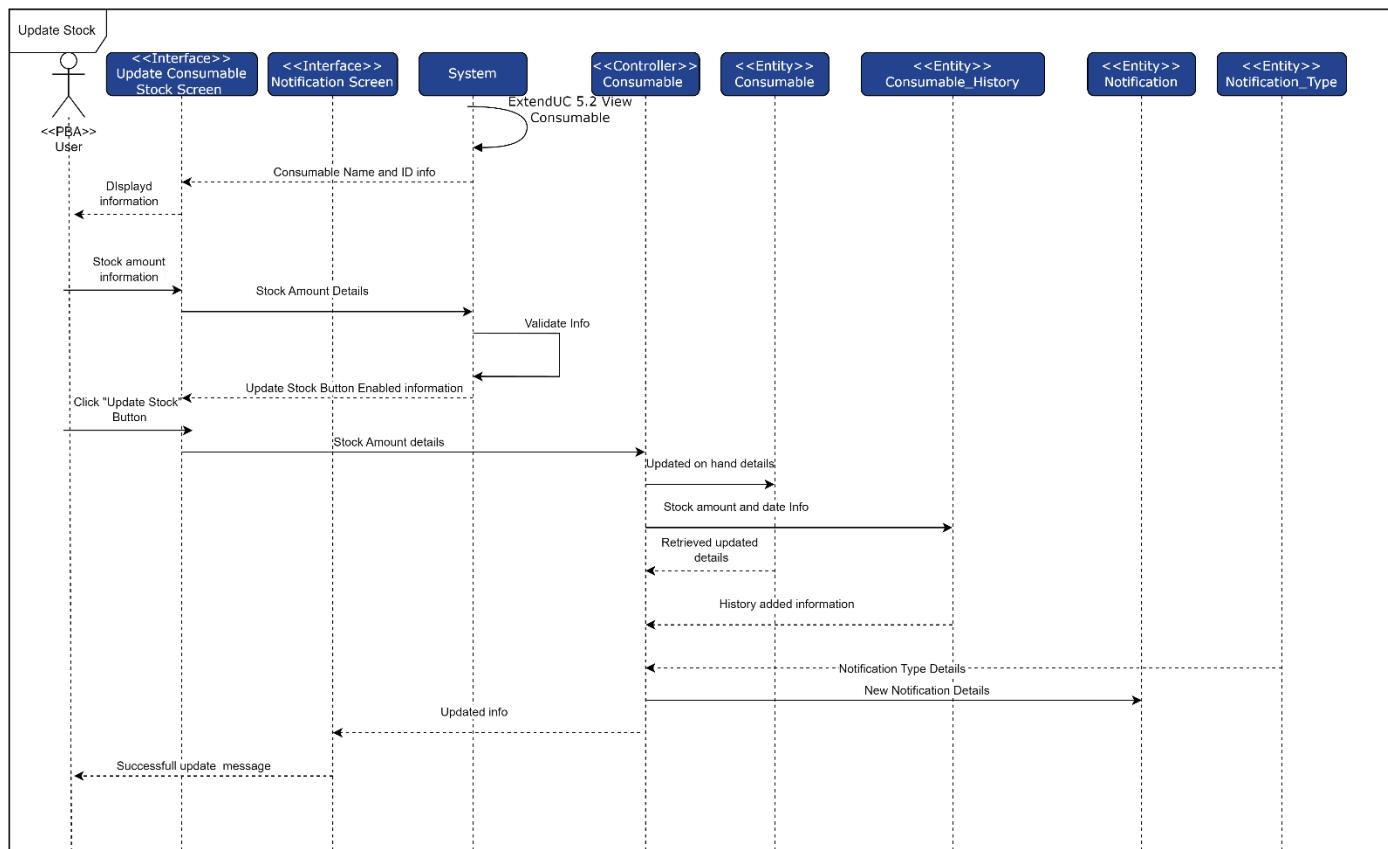


Figure 285 5.10 Update Stock Sequence Diagram

## 1.1 LOGIN

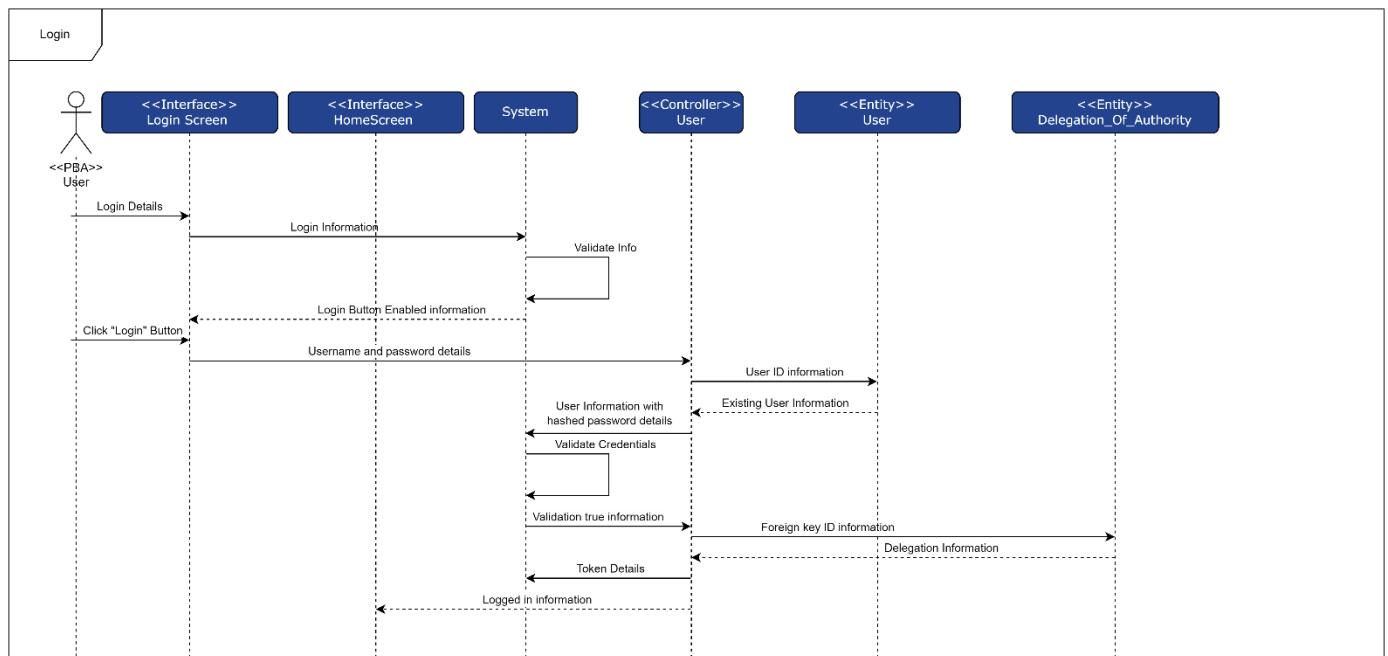


Figure 286 1.1 Login Sequence Diagram

## 1.2 LOGOUT

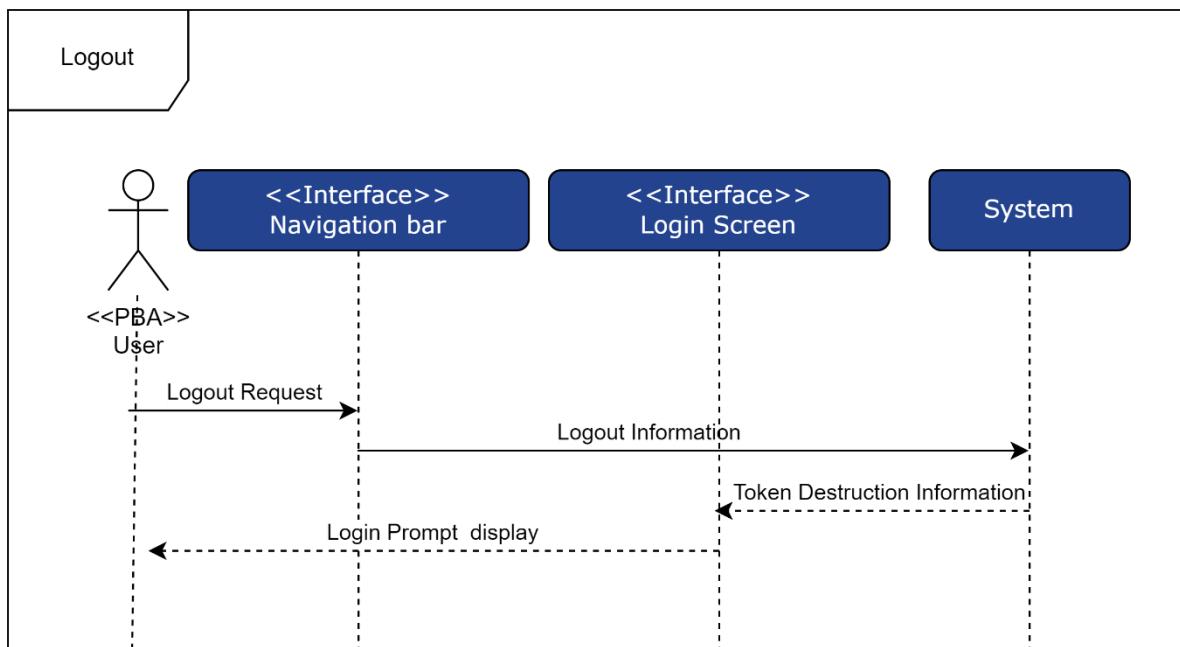


Figure 287 1.2 Logout Sequence Diagram

### 1.3 FORGOT PASSWORD

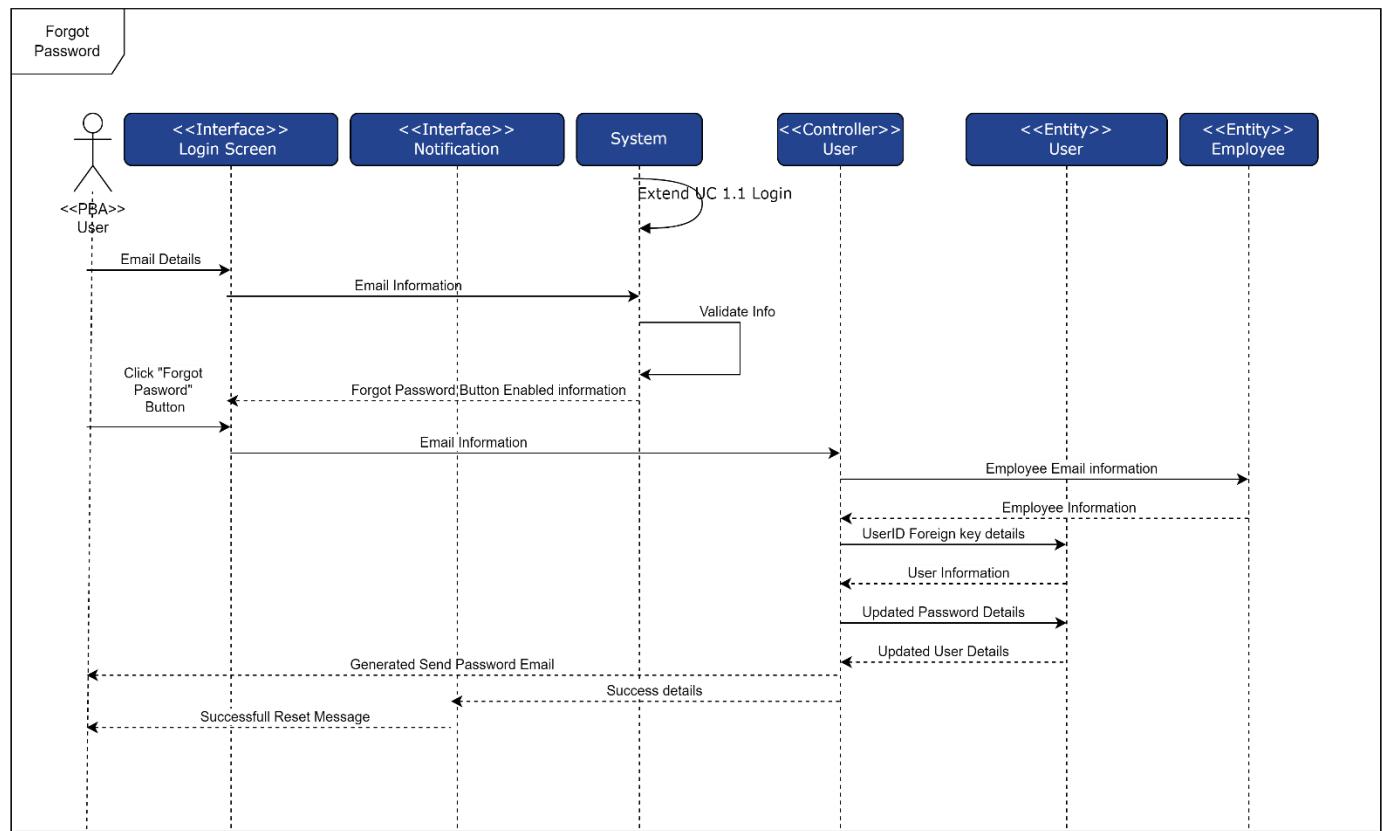


Figure 288 1.3 Forgot Password Sequence Diagram

### 1.4 UPDATE PASSWORD

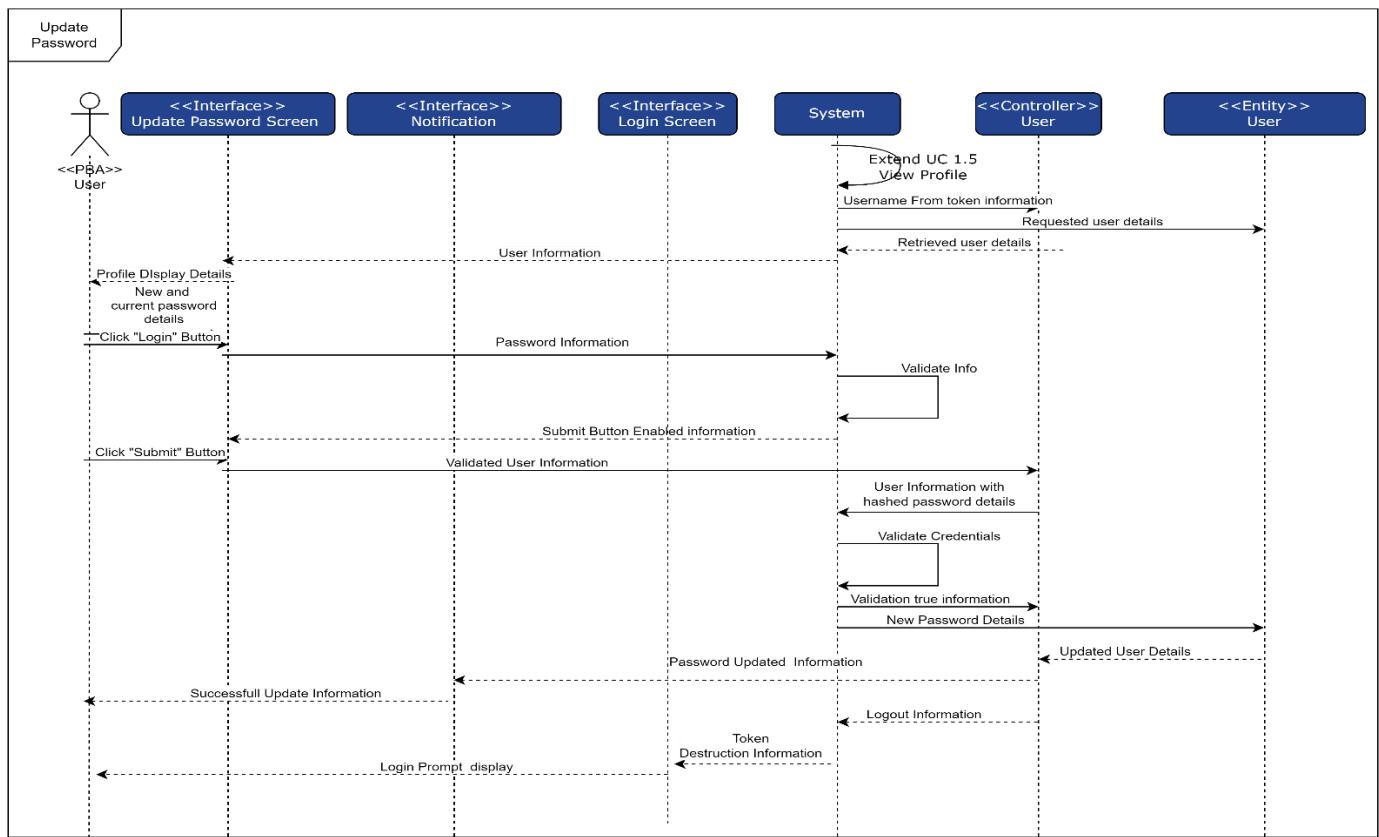


Figure 289 1.4 Update Password Sequence Diagram

### 3.1 CREATE PROCUREMENT REQUEST

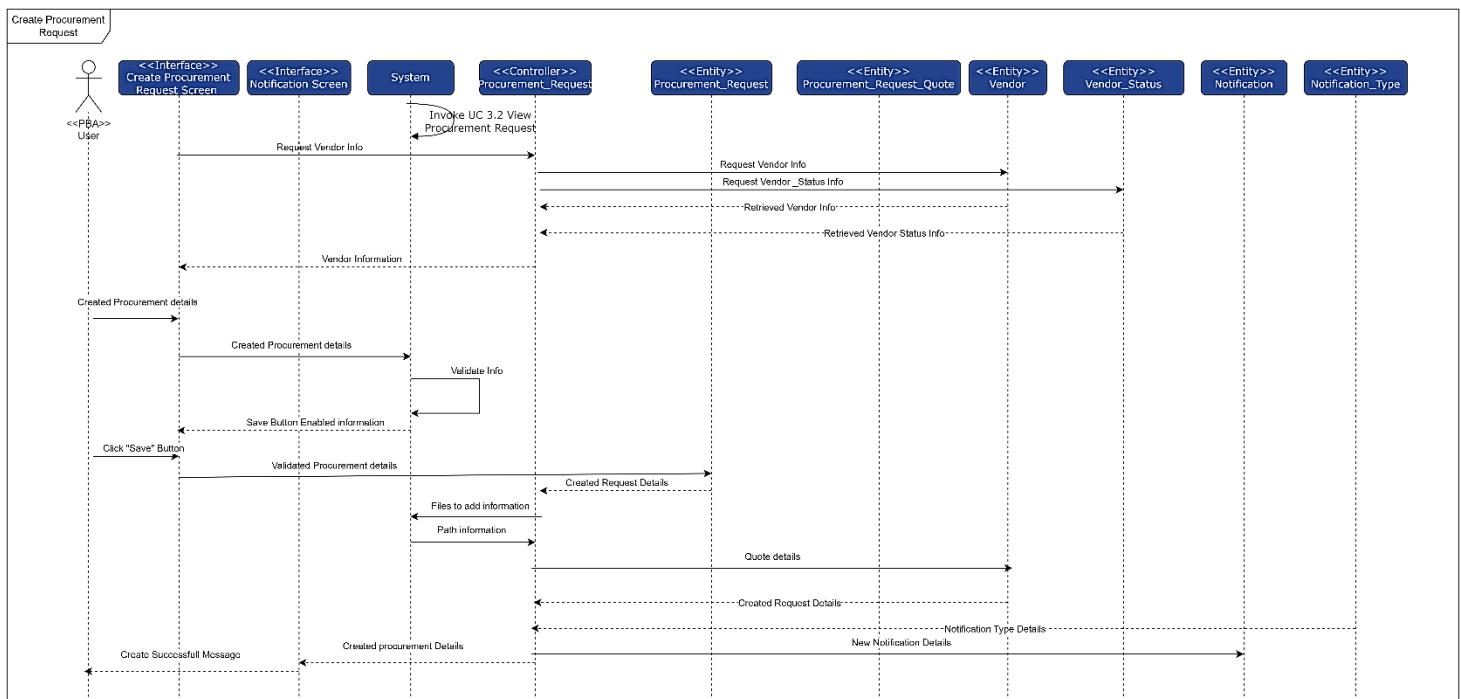


Figure 290 3.1 Create Procurement Request Sequence Diagram

### 3.2 VIEW PROCUREMENT REQUEST

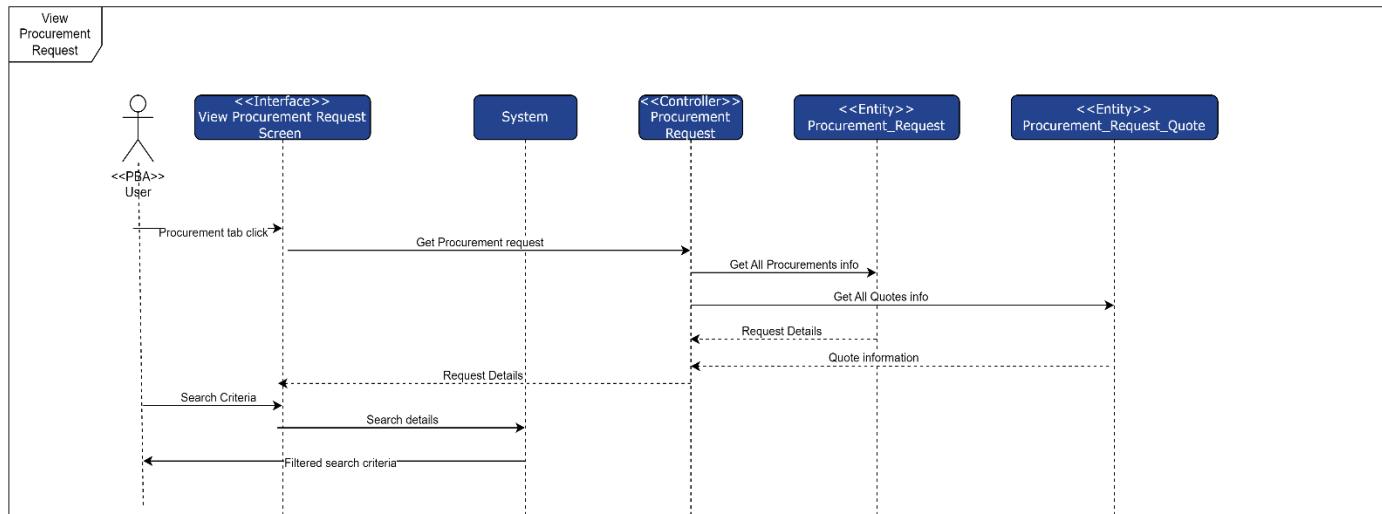


Figure 291 3.2 View Procurement Request Sequence Diagram

### 3.3 UPDATE PROCUREMENT REQUEST

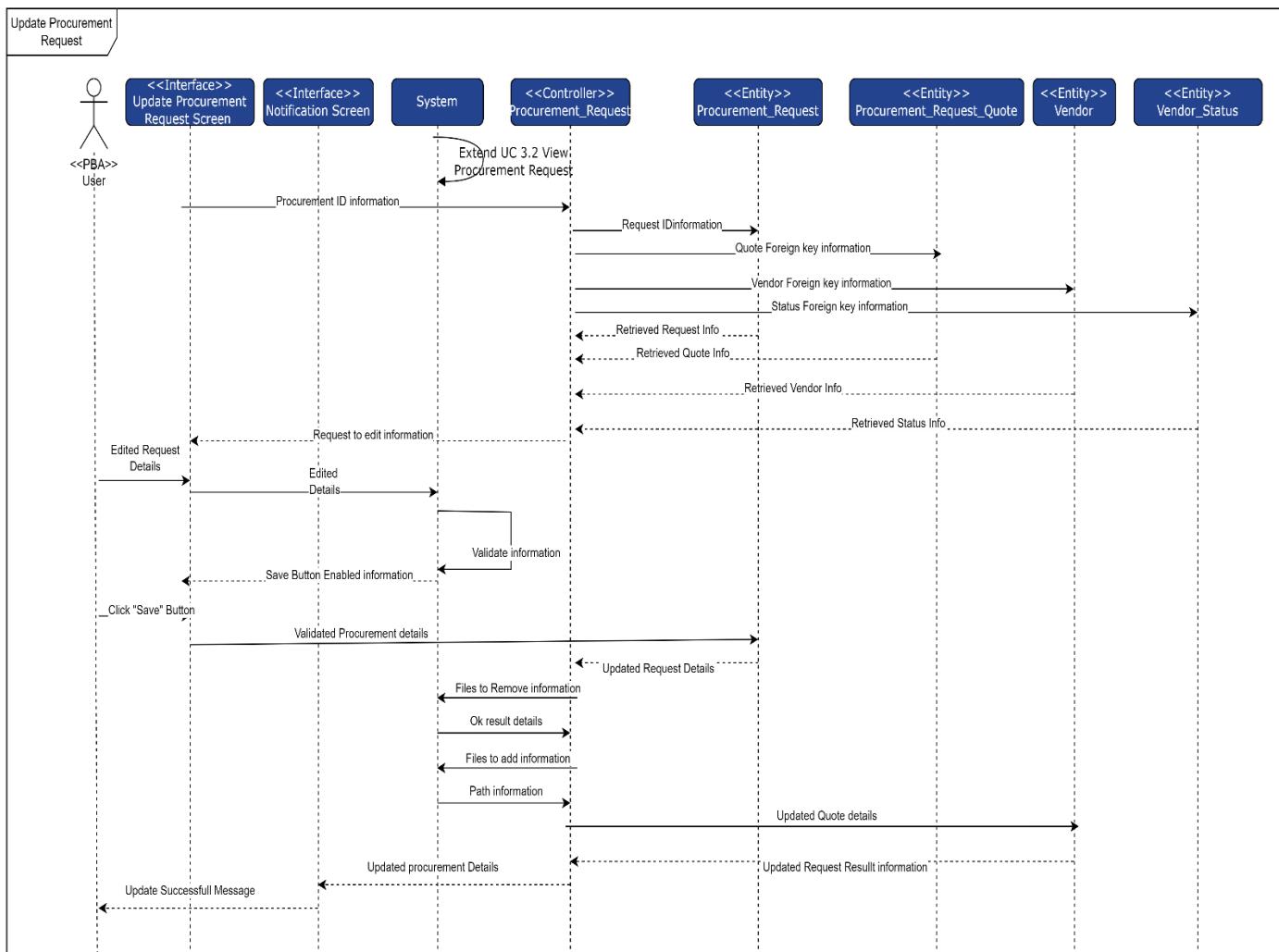


Figure 292 3.3 Update Procurement Request Sequence Diagram

### 3.4 DELETE PROCUREMENT REQUEST

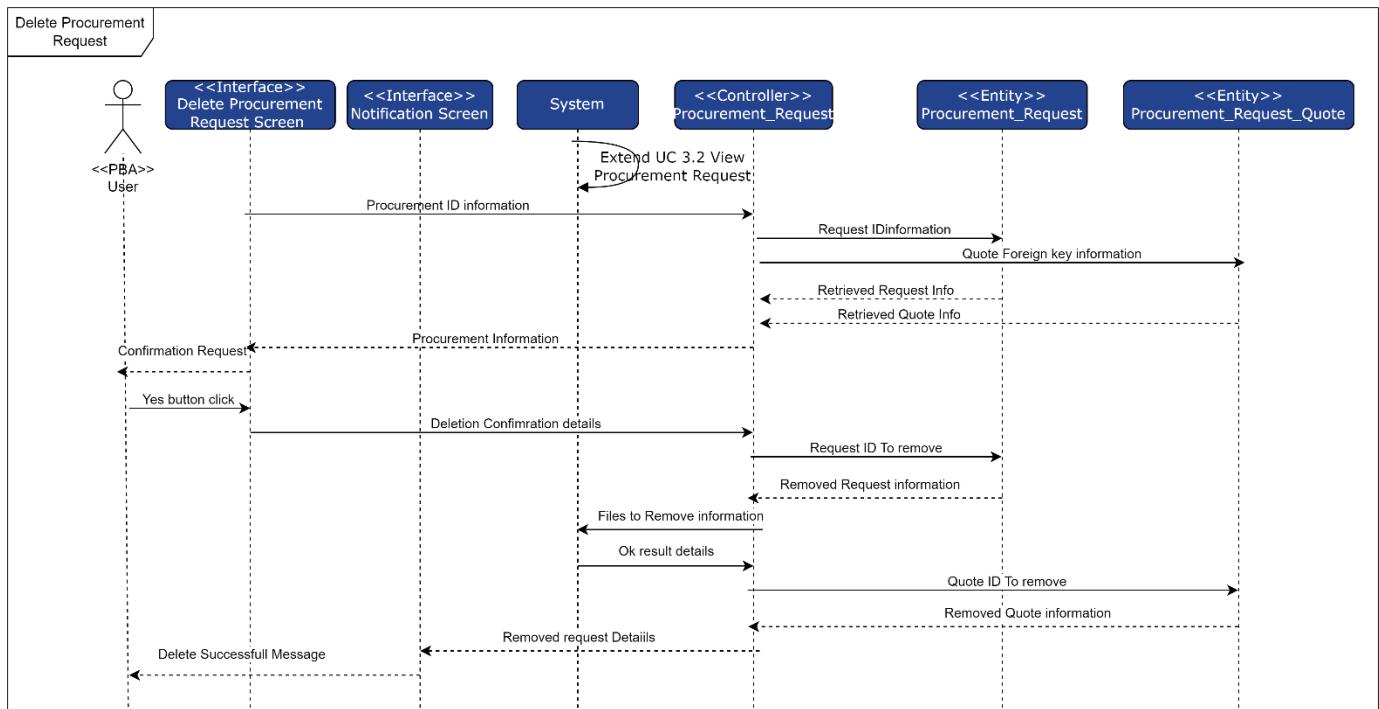


Figure 293 3.4 Delete Procurement Request Sequence Diagram

## USE CASE 3.5-3.7,6.5,6.10-6.14

### 3.5 APPROVE PROCUREMENT REQUEST

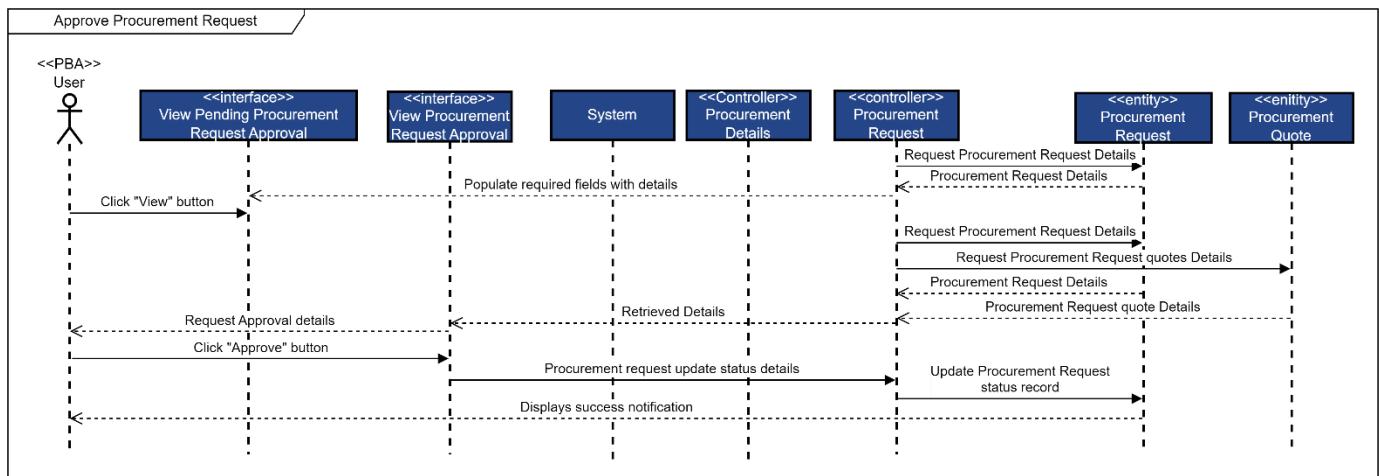


Table 224: 3.5 Approve Procurement Request

### 3.6 PLACE PROCUREMENT DETAILS

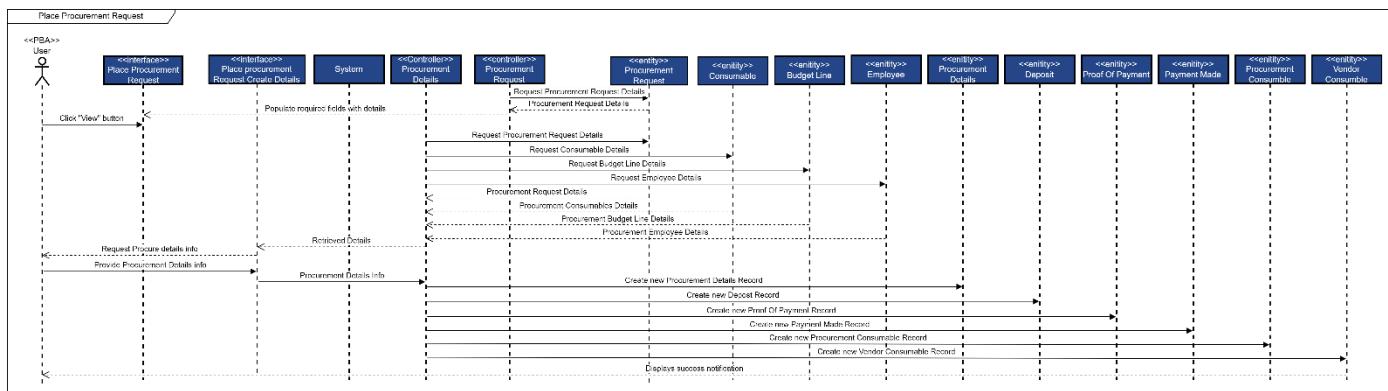


Table 225: 3.6 Place Procurement Details

### 3.7 APPROVE FLAGGED PROCUREMENT DETAILS

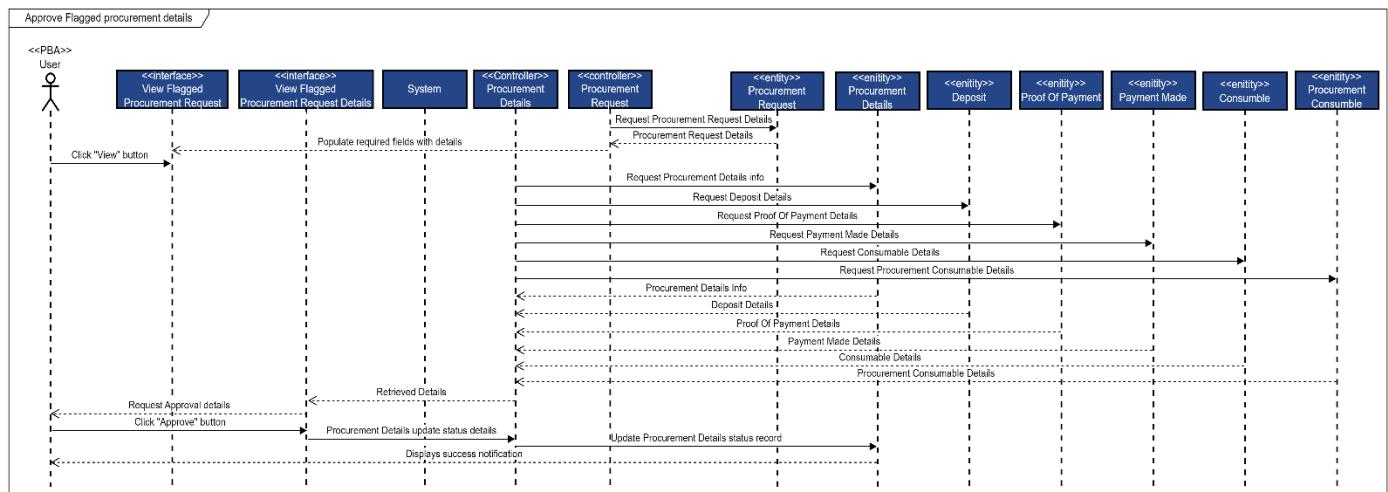


Table 226: 3.7 Approve Flagged Procurement Details

### 6.5 APPROVE ONBOARD REQUEST

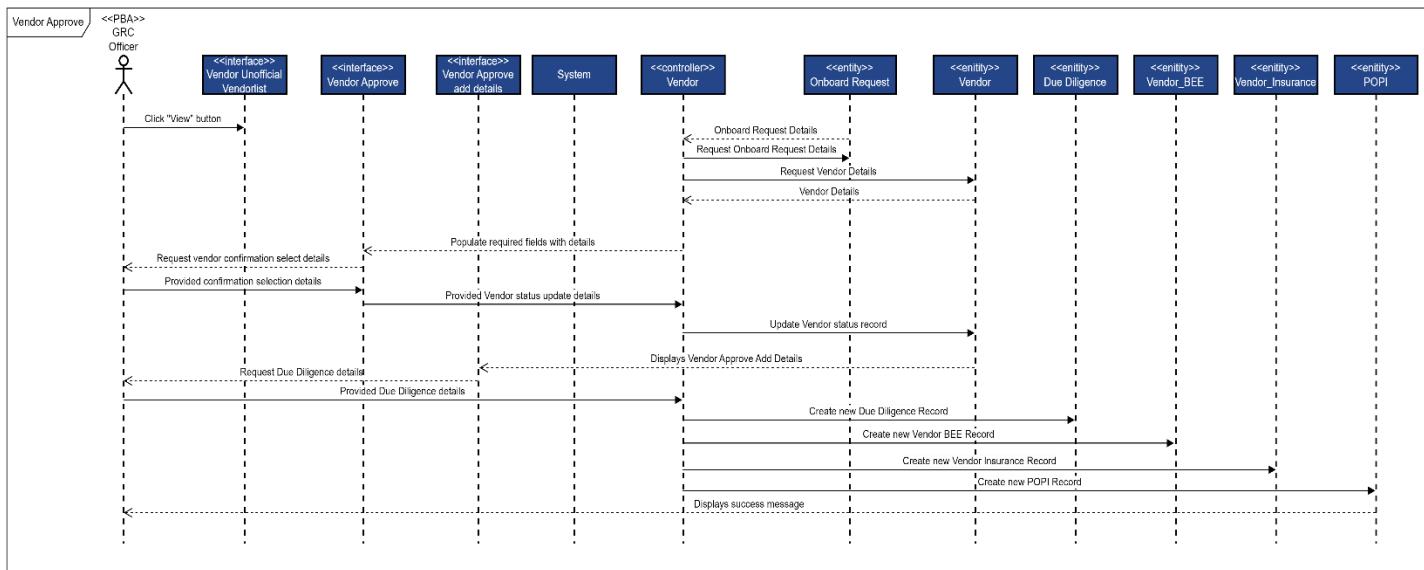
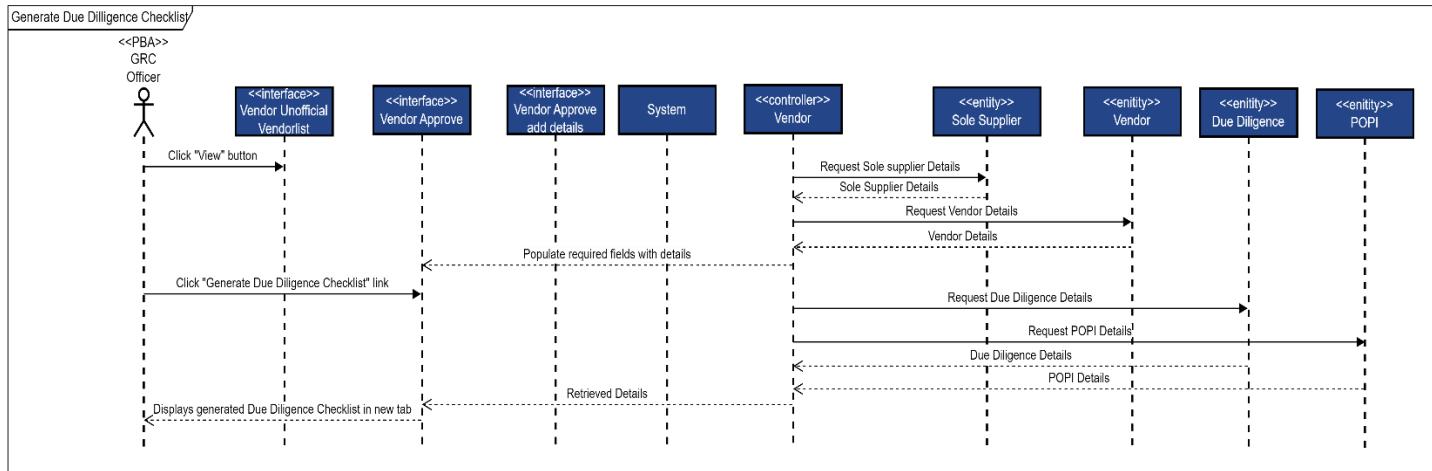
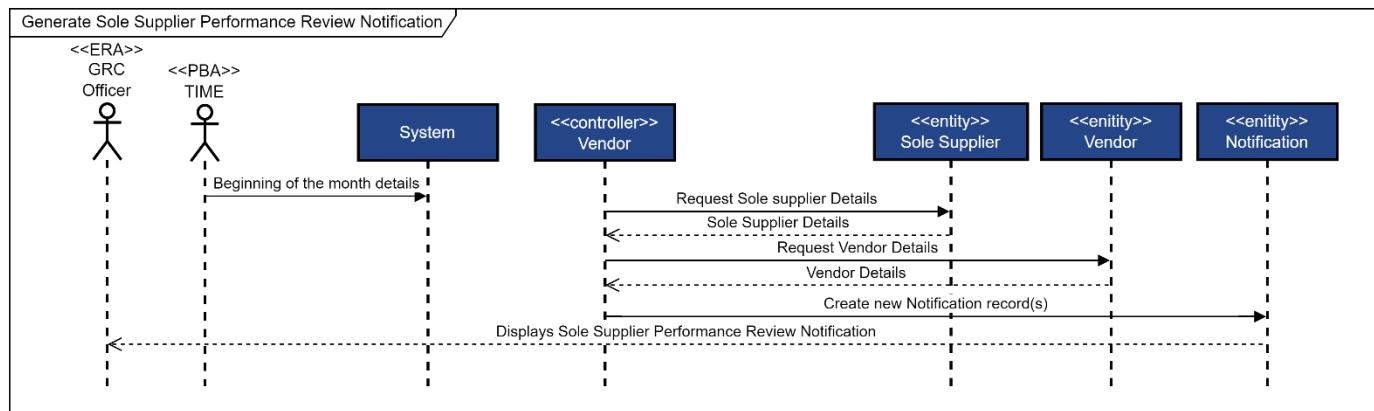


Table 227: 6.5 Approve Onboard Request

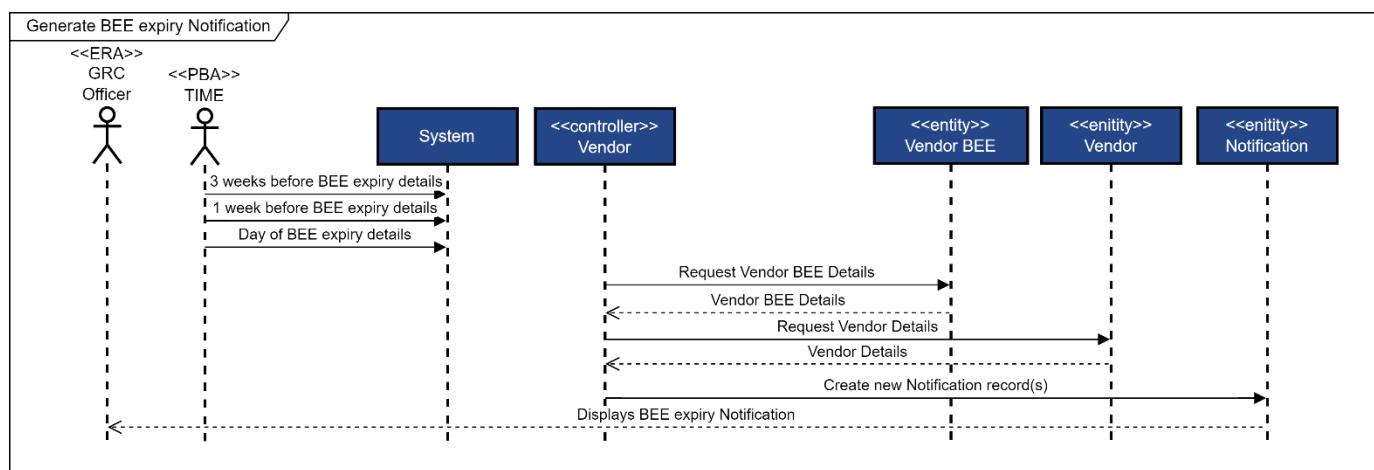
### 6.10 GENERATE DUE DILIGENCE CHECKLIST



### 6.11 GENERATE SOLE SUPPLIER PERFORMANCE NOTIFICATION



### 6.12 GENERATE BEE EXPIRY NOTIFICATION



### 6.13 UPDATE APPROVE ONBOARD REQUEST

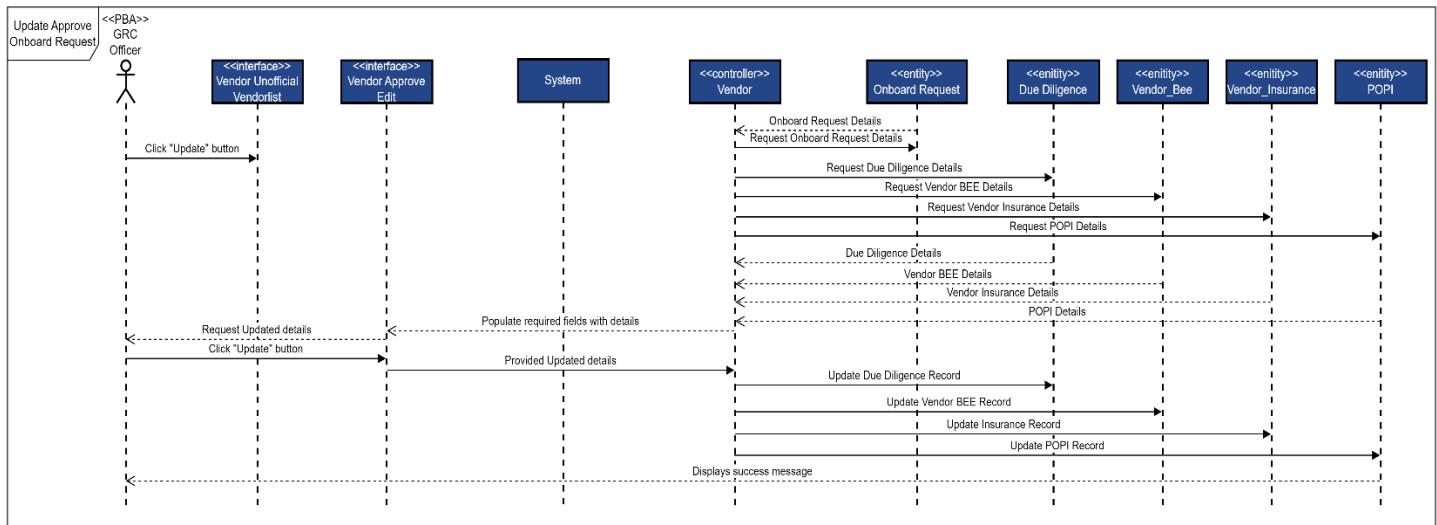


Figure 294: 6.13 Update Approve Onboard Request

### 6.14 VIEW APPROVED OR PENDING ONBOARD REQUEST

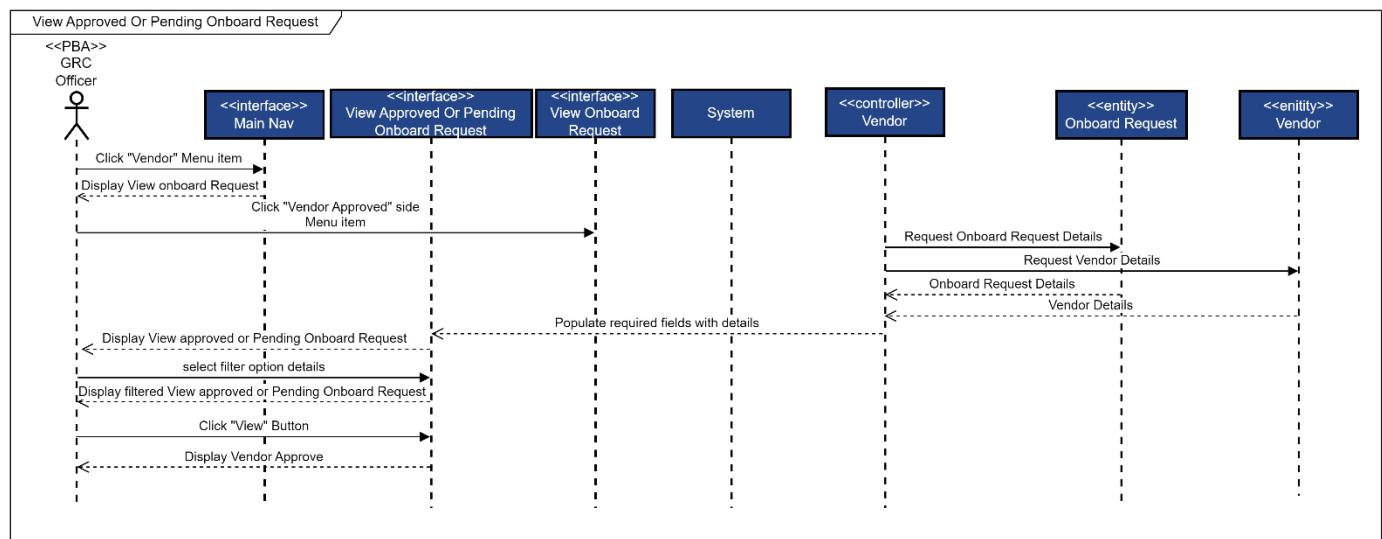


Figure 295: 6.14 View Approved Or Pending Onboard Request

### 9.3 CONCLUSION

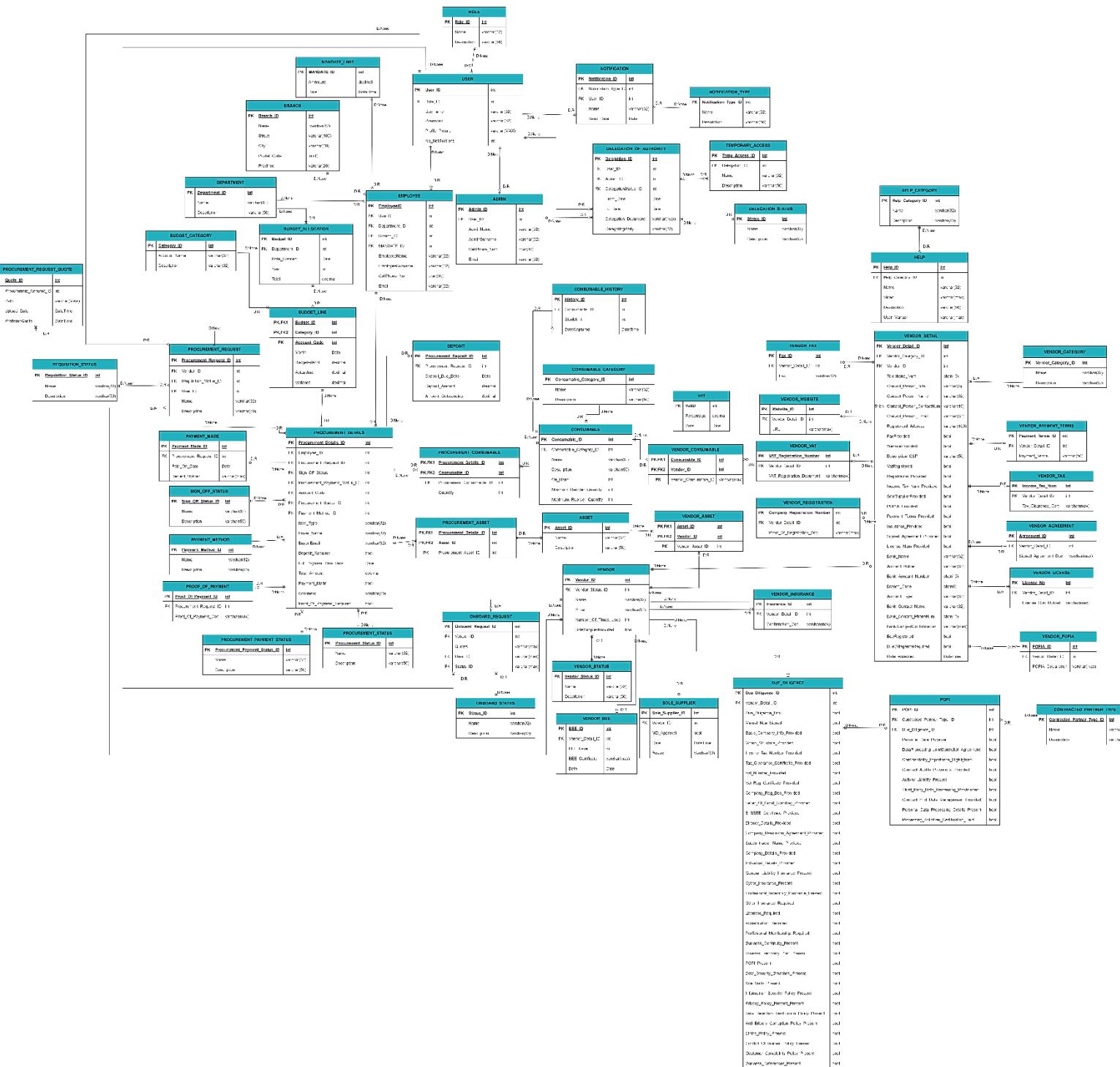
This Conclude the visual representation of the Sequence diagram with for the respective use case diagrams for the Procion system.

# 10. UPDATED ENTITY RELATIONSHIP DIAGRAM

## 10.1 INTRODUCTION

This section goes in depth into a visual representation of our updated technical ERD. The ERD complies with referential integrity and is fully attributed as well as in 3<sup>rd</sup> normal form. This serves as an indication of how we will structure our database.

## 10.2 ENTITY RELATIONSHIP DIAGRAM



*Figure 296 Technical ERD*

### 10.3 CONCLUSION

This concludes the updated entity relationship diagram with all the entity tables and relationships between the entities within the system that we will develop.

# 11. STATE DIAGRAM

## 11.1 INTRODUCTION

In this section we will display the visual representation of the State diagram for the vendor subsystem.

## 11.2 STATE DIAGRAM

### DELEGATION OF AUTHORITY STATE DIAGRAM

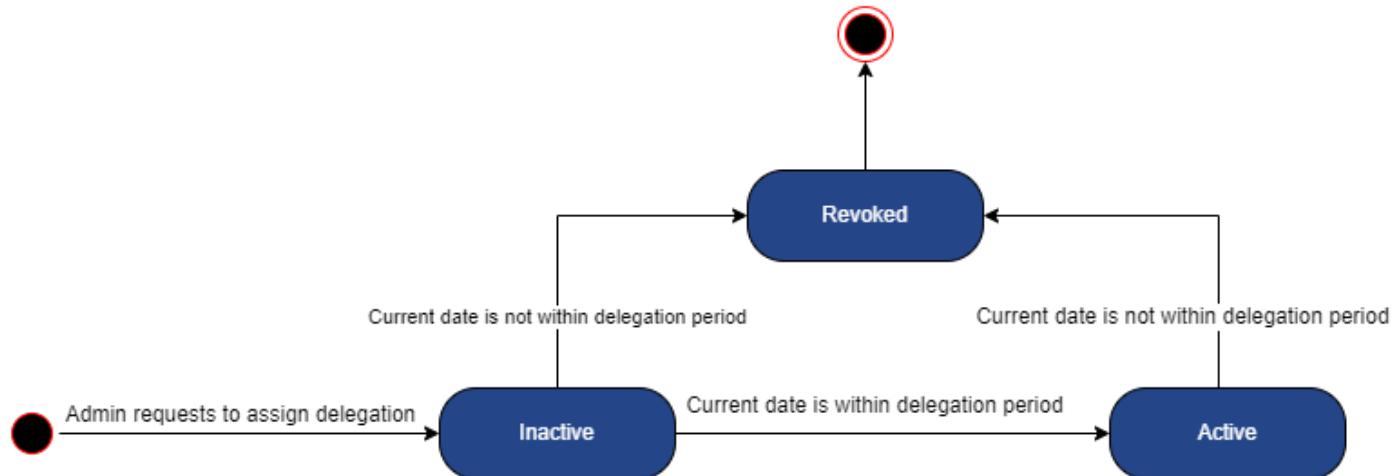


Table 228: Delegation of Authority State Diagram

### VENDOR STATE DIAGRAM

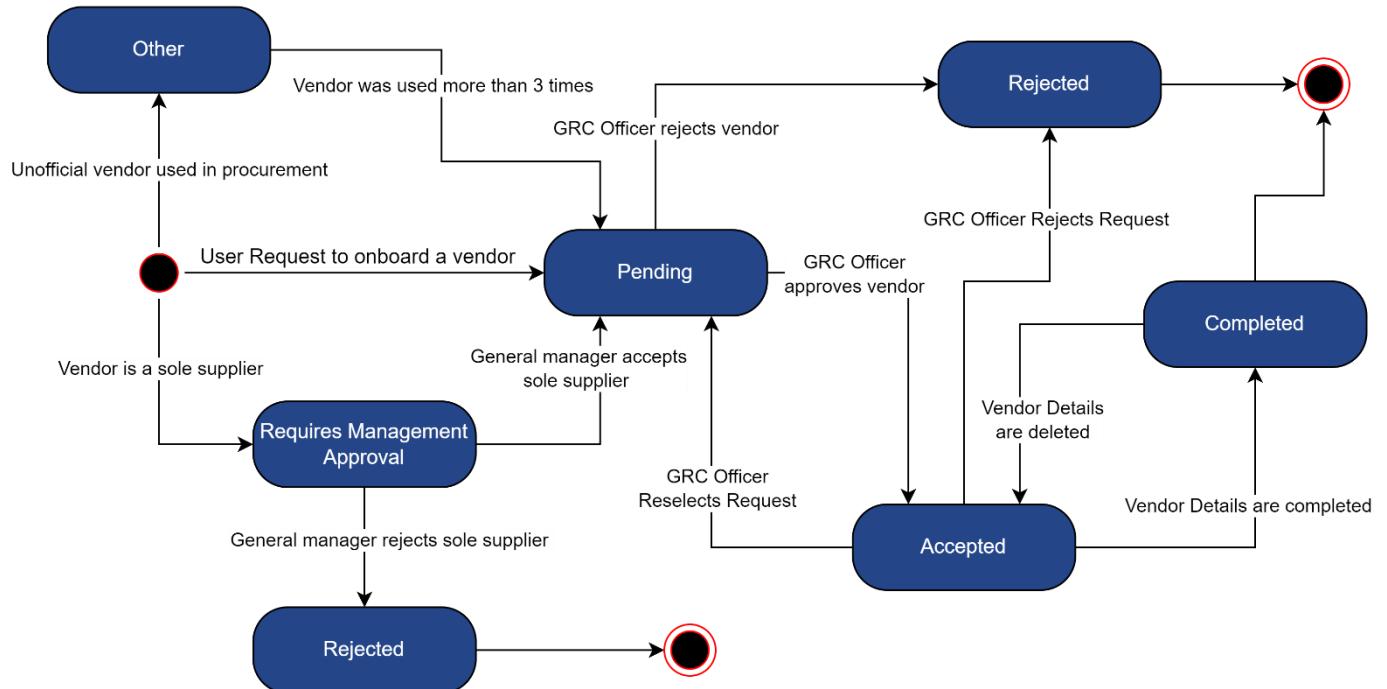


Table 229: Vendor State Diagram

### ONBOARD REQUEST STATE DIAGRAM

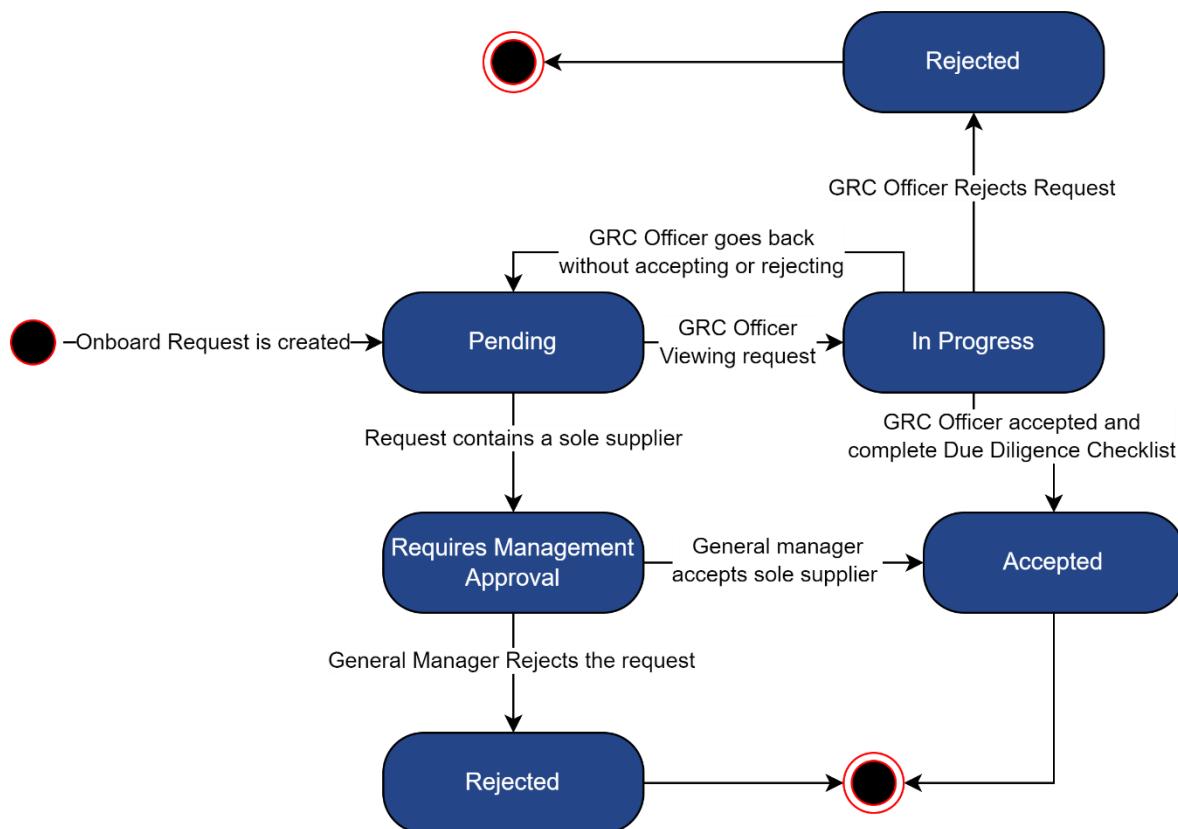


Table 230: Onboard Request State Diagram

### PROCUREMENT REQUEST STATE DIAGRAM

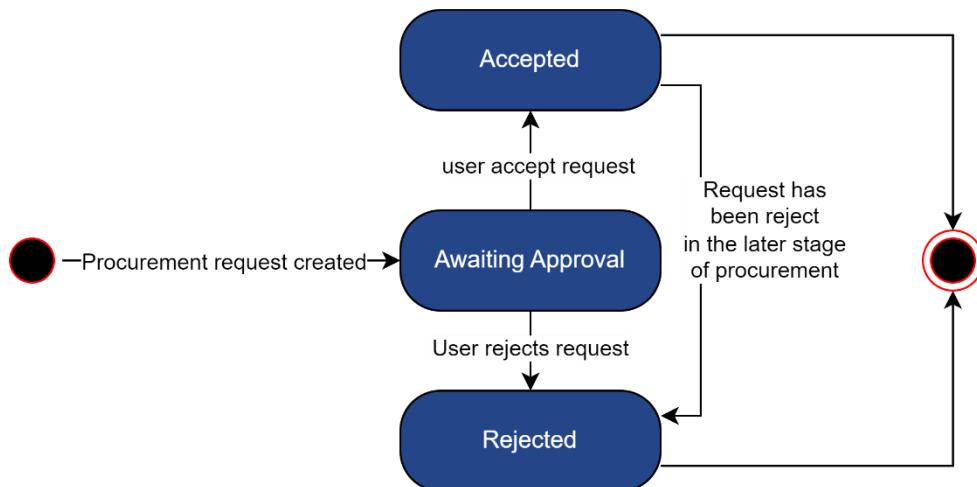


Table 231: Procurement Request State Diagram

### PROCUREMENT DETAILS STATE DIAGRAM

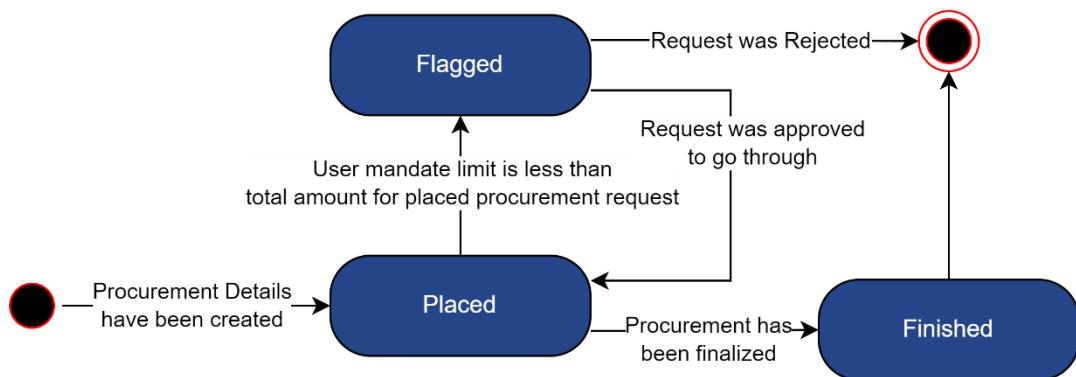


Table 232: Procurement Details State Diagram

### 11.3 CONCLUSION

This concludes the updated entity relationship diagram with all the entity tables and relationships between the entities within the system that we will develop.

# 12.UPDATED COMPLEXITY MATRIX

## 12.1 INTRODUCTION

In this section we show the updated complexity matrix with all the technology that we will be using, the Procion system. This shows each technical requirements that the system will have and the Total complexity mark of the system.

## 12.2 COMPLEXITY MATRIX



### INF 370

#### Complexity Matrix Mark Sheet

| Topic   | Level  |   | Marks | Indicated by group | Marks given | MAX | Total |
|---|--|---|-------|--------------------|-------------|-----|-------|
|   | For online applications: Responsive web design   |   | 3     | 3                  |             | 42  | 0     |
|   | For desktop applications: Form design according to design principles (Schneiderman's golden rule on navigation applies here) | * |       | 3                  |             |     |       |
|   | Appropriate use of grids/tables  |   | 3     | 3                  |             |     |       |
|   | Appropriate use of tabs/links  |   | 3     | 3                  |             |     |       |
|   | Use of graphs in an appropriate business context   |   | 4     | 4                  |             |     |       |
|   | The storage and display of graphical information, like photos with a good business reason                                    |   | 3     | 3                  |             |     |       |
|   | Working e-mail automatically generated from the database in an appropriate business context                                  |   | 2     | 2                  |             |     |       |
|   | SMS messages automatically generated from the system in an appropriate business context                                      |   | 2     |                    |             |     |       |
|   | Extensive user-friendly search facility  |   | 3     | 3                  |             |     |       |
|   | At least one use of a tree to display data from the database   |   | 3     | 0                  |             |     |       |
|   | Able to dynamically modify a data tree structure and in doing so adjusting the data in the database                          |   | 4     | 0                  |             |     |       |
|   | At least one use of a calendar view of data (not a date/time picker; not a plug-in such as Google calendar)                  |   | 3     | 3                  |             |     |       |
|   | Uploading a file into the system with appropriate business reason  |   | 3     | 3                  |             |     |       |
|   | The use of audio/video in an appropriate business context  |   | 3     | 0                  |             |     |       |
|   | At least one use of an administrator configurable timer in an appropriate business context                                   |   | 3     | 3                  |             |     |       |
| 1. Special GUI  |  |   |       |                    |             |     |       |
|   | At least 30 tables used (4 member groups) or 40 tables used (5 member groups)  | * | 6     | 6                  |             | 15  | 0     |
|   | Full referential integrity on all tables   | * | 6     | 6                  |             |     |       |
|   | At least one use of master-detail table relationships (Schneiderman's golden rule on system status applies here)             | * | 3     | 3                  |             |     |       |
|   | At least 3 simple list reports in a reporting tool (no control breaks, no graphs, single table)                              | * | 3     | 3                  |             | 15  | 0     |
|   | At least 2 transactional report with 2 or more control breaks (with heading and calculated values/totals, multiple tables)   | * | 6     | 6                  |             |     |       |
|   | At least 1 report with adjustable criteria   |   | 3     | 3                  |             |     |       |
| 3. Reports  | At least 1 management report using a graph   |   | 3     | 3                  |             |     |       |
| 4. Flexibility  | hard coded but maintained in a sub-module of the system.   |   | 6     | 6                  |             | 12  | 0     |
|   | maintained in a sub-module of the system.  |   | 6     | 6                  |             |     |       |
|   | consistent, user-friendly error messages are   |   | 6     | 6                  |             | 12  | 0     |
| 5. Error handling   | Appropriate data validation on <b>all</b> input fields   |   | 6     | 6                  |             |     |       |
|   | up a complete help document (HTML, PDF, Help-file)   |   | 3     | 3                  |             | 15  | 0     |
|   | a specific screen/function will automatically open   |   | 6     | 6                  |             |     |       |
|   | Search Facility on Help  |   | 3     | 3                  |             |     |       |
| 6. Help   | Extensive use of hints   |   | 3     | 3                  |             |     |       |
|   | user profiles  |   | 3     | 3                  |             | 13  | 0     |
|   | business reason.   |   | 3     | 3                  |             |     |       |
| 7. Security   | Encrypted passwords in database  | * | 1     | 1                  |             |     |       |
|   | user profiles that will enable/disable access to   |   | 6     | 6                  |             |     |       |
|   | showing at least date, time, user, transaction type, following: date, user, transaction type                                 |   | 6     | 6                  |             | 9   | 0     |
| 8. Audit Trail  | installation disks that take care of application   |   | 3     | 0                  |             | 15  | 0     |
|   | application to a publicly accessible web server  |   | 3     | 3                  |             |     |       |
|   | Market place (such as the PlayStore or the   |   | 6     | 0                  |             |     |       |
| 9. Deployment   | server   |   | 3     | 3                  |             |     |       |
| 10. Backup and Restore  | backup/restore all data (system may exit during  |   | 3     | 3                  |             | 3   | 0     |
|   | data in it based on the parameters provided (with a  |   | 6     | 6                  |             | 9   | 0     |
| 11. Import/Export Data  | Importing or Exporting of data (with good business   |   | 3     | 3                  |             |     |       |
|   | and-play technology, such as a swipe card reader,  |   | 3     | 0                  |             | 18  | 0     |
| 12. External INPUT device                                       | specific software. Data or images must seamlessly  |   | 6     | 0                  |             |     |       |
|   | device specific software. Data or images must  |   | 9     | 0                  |             |     |       |
| 13. External APPLICATION / Services                             | application (with good business reason)  |   | 3     | 3                  |             | 9   | 0     |
|   | application system exists and the interface must be  |   | 6     | 0                  |             |     |       |
|   | alternative platform.  |   | 3     | 3                  |             | 27  | 0     |
|   | platform (i.e. data is displayed from the system's   |   | 3     | 3                  |             |     |       |
| 14. Multiplatform processing for an appropriate business reason | alternative platform (i.e. both reading and writing  |   | 9     | 3                  |             |     |       |
|   | onto the system's database.  |   | 3     | 3                  |             |     |       |
|   | reading and writing data from the system's   |   | 9     | 9                  |             |     |       |
|   | between your database and your business layer  |   | 3     | 3                  |             | 12  | 0     |
|   | your business layer and your presentation layer  |   | 6     | 6                  |             |     |       |
| 15. Programming Principles                                      | triggers and/or jobs.  |   | 3     | 3                  |             |     |       |
| system  | system (e.g. machine learning, AI, block chain, text   | # | 1-9   | 9                  |             | 9   | 0     |
|   | Complexity Marks Obtained  |   |       | 187                |             | 0   | 235   |
|   | Maximum Complexity Marks   |   | 235   |                    |             |     |       |
|   | Complexity Marks Required for Del 5 (5 members in team)  |   | 150   |                    |             |     |       |
|   | Complexity Marks Required for Del 5 (4 members in team)  |   | 130   |                    |             |     |       |

**12.3 CONCLUSION**

This concludes the update complexity matrix section of this document.

## 12. DOCUMENT CONCLUSION

This concludes the iteration 5 document regarding the Design and the development of the following Use cases: 1.5-1.6, 1.8 & 2.17-2.20 & 2.35-2.36(Emil), 1.7 & 2.25, 2.27-2.30(Leon), 3.9-3.13 & 4.1-4.2(Bupe), 1.1-1.4 & 3.1-3.4 & 5.9-5.10(Jason), 3.5-3.8 & 6.5-6.12(Werner) of the Procion system that we are developing. Here we try and gather insight about the third update functional requirements, Logical Use Case Narratives, Logical context diagrams for those narrative, Screen designs for those narrative, Technical Use Case Narrative, Technical Context diagram for those narratives, Technical primitive level diagram for those narrative, Activity diagram for those narratives, Sequence diagram for those narratives, State diagram for those narratives(only complex narratives that has a change in state involved), and Demonstrate Functionality of the use cases, and the updated complexity matrix with the updated ERD to help us achieve the start for the development of the Procion system. This will bring us closer towards our end goal of building the Procion system.

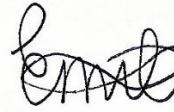
## 13. TEAM SIGN OFF

### PROCION SYSTEM: Design & Development

We, team 11, understand and acknowledge, have reviewed the information contained in the document (Technical Specifications), and we confirm and understand the contents and that the information provided is true.

#### **BY SIGNING BELOW, WE AGREE TO THE FOLLOWING:**

1. We understand the goals and the objectives of the project, as well as the specific tasks and responsibilities assigned to us.
2. We agree that our assigned tasks have been completed and meet the required quality standards.
3. We agree that any issues, risks, and concerns have been appropriately addressed and resolved.
4. Acknowledge that any changes or revisions to the document have been made and agreed upon by the project team.
5. That each team member has contributed equally to the iteration

| <b>Signed by:</b>  |  |  |  |  |
|--|--|--|--|--|
| <b>Name:</b> Jason vd Merwe  | <b>Name:</b> Leon Combrinck  | <b>Name:</b> Bupe Chindongo  | <b>Name:</b> Werner Schutte  | <b>Name:</b> Emil Wonigkeit  |
| <b>Date:</b> 2023-07-30  | <b>Date:</b> 2023-07-30  | <b>Date:</b> 2023-07-30  | <b>Date:</b> 2023-07-30  | <b>Date:</b> 2023-07-30  |
| <b>Signature:</b><br> | <b>Signature:</b><br> | <b>Signature:</b><br> | <b>Signature:</b><br> | <b>Signature:</b><br> |

## 14. CLIENT DOCUMENT SIGN OFF

PROCION SYSTEM: Design and development

Business Name: MOYO Business Advisory

I, Vumile Gumbi, Hereby confirm that I have read the document and agree that the information contained therein is complete, accurate and meets the requirements of the project.

**BY SIGNING I CONFIRM THAT:**

1. I have reviewed the deliverable as outlined in the project.
2. I accept the deliverable as complete and satisfactory.

| Signed by: |   |
|------------|---|
| Name:      | Vumile Gumbi  |
| Date:      | 28/07/2023  |
| Signature: |  |