|  |  |
| --- | --- |
|  | **MINISTRY OF EDUCATION AND TRAINING** |

FPT UNIVERSITY

**Capstone Project Document**

**Motorbike Accessories Inventory Management**

|  |  |
| --- | --- |
| **Group 7** | |
| **Group members** | Phạm Minh Hoàng – SE62769 (Leader)  Nguyễn Trường Thịnh – SE61887  Đỗ Trung Hiếu – SE61847  ~~Nguyễn Huy Hoàng – SE62442~~ |
| **Supervisor** | Lại Đức Hùng |
| **Ext. Supervisor** | N/A |
| **Project Code** | RFIM |

– **Ho Chi Minh City, 13 May, 2019** –

**ACKNOWLEDGEMENT**

In order to be able to finish this RFIM project, we have got a lot of help from other people. We would love to thank each person who had helped us on this project.

First of all, we would like to thank FPT University for giving us a change to work on this project, a chance to improve ourselves within 4 years.

Second of all, we are really thankful to Mr. Lai Duc Hung - our supervisor. He has been always with us giving a lot of useful suggestions from the very beginning till the end of this project. We would like to give a special thanks to him for supporting us with a suitable place to work, for helping us when we have no idea about some problems.

Next, we would like to thank all of the instructors who has lectured us in (about) 4 years.

And we shall not forget about all of the authors from the documentations we read.

Most importantly, we would like to dedicate this work to our parents. Thanks for supporting us from the first day when we were born.

In conclusion, thanks to everyone, it has been a pleasure working with all of you.

-RFIM Team-

# Table of Contents

[Table of Contents 3](#_Toc17067632)

[List of Tables 6](#_Toc17067633)

[List of Figures 10](#_Toc17067634)

[A. Introduction 14](#_Toc17067635)

[1. Project Information 14](#_Toc17067636)

[2. Introduction 14](#_Toc17067637)

[3. Current Situation 14](#_Toc17067638)

[4. Problem Definition 14](#_Toc17067639)

[5. Proposed Solution 15](#_Toc17067640)

[5.1. Feature Functions 15](#_Toc17067641)

[5.2. Advantages and Disadvantages 15](#_Toc17067642)

[6. Functional Requirements 16](#_Toc17067643)

[7. Role and Responsibility 16](#_Toc17067644)

[B. Software Project Management Plan 16](#_Toc17067645)

[1. Problem Definition 16](#_Toc17067646)

[1.1. Name of this Capstone Project 16](#_Toc17067647)

[1.2. Problem Abstract 17](#_Toc17067648)

[1.3. Project Overview 17](#_Toc17067649)

[2. Project Organization 20](#_Toc17067650)

[2.1. Software Process Model 20](#_Toc17067651)

[2.2. Roles and Responsibilities 20](#_Toc17067652)

[2.3. Tools and Techniques 21](#_Toc17067653)

[3. Project Management Plan 22](#_Toc17067654)

[3.1. Software development life cycle 22](#_Toc17067655)

[3.2. Phase Detail 23](#_Toc17067656)

[3.3. Meeting Minutes 25](#_Toc17067657)

[4. Coding Convention 25](#_Toc17067658)

[C. Software Requirement Specification 26](#_Toc17067659)

[1. User Requirement Specification 26](#_Toc17067660)

[1.1. Guest Requirement 26](#_Toc17067661)

[1.2. Admin Requirement 26](#_Toc17067662)

[1.3. Warehouse Accountant Requirement 27](#_Toc17067663)

[1.4. Stock-keeper Requirement 27](#_Toc17067664)

[2. System Requirement Specification 27](#_Toc17067665)

[2.1. External Interface Requirement 27](#_Toc17067666)

[2.2. System Overview Use Case 29](#_Toc17067667)

[2.3. List of Use Cases 30](#_Toc17067668)

[3. Software System Attribute 72](#_Toc17067669)

[3.1. Usability 72](#_Toc17067670)

[3.2. Reliability 72](#_Toc17067671)

[3.3. Availability 72](#_Toc17067672)

[3.4. Security 72](#_Toc17067673)

[3.5. Maintainability 72](#_Toc17067674)

[3.6. Portability 72](#_Toc17067675)

[3.7. Performance 73](#_Toc17067676)

[4. Conceptual Diagram 73](#_Toc17067677)

[D. Software Design Description 74](#_Toc17067678)

[1. Design Overview 74](#_Toc17067679)

[2. System Architectural Design 75](#_Toc17067680)

[2.1. Web Application Architecture Description 75](#_Toc17067681)

[2.2. Mobile Application Architecture Description 76](#_Toc17067682)

[3. Component Diagram 77](#_Toc17067683)

[4. Detail Description 77](#_Toc17067684)

[4.1. Class Diagram 77](#_Toc17067685)

[4.2. Class Diagram Explanation 79](#_Toc17067686)

[4.3. Sequence Diagram 89](#_Toc17067687)

[4.4. Activity Diagram 94](#_Toc17067688)

[5. User Interface Design 101](#_Toc17067689)

[5.1. Component Interface 101](#_Toc17067690)

[5.2. User Interface Design 104](#_Toc17067691)

[6. Database Design 139](#_Toc17067692)

[6.1. Entity Relationship Diagram (ERD) 139](#_Toc17067693)

[6.2. Entity Dictionary 139](#_Toc17067694)

[7. Algorithms 140](#_Toc17067695)

[7.1. Find box base on the longest stock in date and closest warehouse’s gate 140](#_Toc17067696)

[7.2. Convert shelves’ coordinates to matrix algorithm 141](#_Toc17067697)

[7.3. Dijkstra algorithms 143](#_Toc17067698)

[E. System Implementation & Testing 146](#_Toc17067699)

[1. Introduction 146](#_Toc17067700)

[1.1. Overview 146](#_Toc17067701)

[1.2. Test Approach: 147](#_Toc17067702)

[2. Database Relationship Diagram 148](#_Toc17067703)

[2.1. Physical Diagram 148](#_Toc17067704)

[2.2. Data Dictionary 149](#_Toc17067705)

[3. Test Plan 153](#_Toc17067706)

[3.1. Feature to be tested 153](#_Toc17067707)

[3.2. Features not to be tested 153](#_Toc17067708)

[4. System Testing Test Case 153](#_Toc17067709)

[4.1. Web System Test Cases 153](#_Toc17067710)

[4.2. Mobile Application Test Cases 166](#_Toc17067711)

[F. Software User’s Manual 176](#_Toc17067712)

[1. Installation Guide 176](#_Toc17067713)

[1.1. Setting up environment at server side 176](#_Toc17067714)

[1.2. Deployment at server side 177](#_Toc17067715)

[1.3. Setting up the environment at client side 178](#_Toc17067716)

[2. User Guide 180](#_Toc17067717)

[2.1. Web System 180](#_Toc17067718)

[2.2. Mobile Application 194](#_Toc17067719)

[3. 取扱説明書 202](#_Toc17067720)

[3.1. ウェブシステム 202](#_Toc17067721)

[3.2. モバイルアプリ 217](#_Toc17067722)

[G. Appendix 227](#_Toc17067723)

# List of Tables

[Table 1 – Role and Responsibility 16](#_Toc17066434)

[Table 2 - Hardware requirements for Server 19](#_Toc17066435)

[Table 3 - Hardware requirements for PC 19](#_Toc17066436)

[Table 4 - Hardware requirements for Mobile 19](#_Toc17066437)

[Table 5 - Software requirements for PC 20](#_Toc17066438)

[Table 6 - Role and Responsibilities 21](#_Toc17066439)

[Table 7 - Tools and Techniques 21](#_Toc17066440)

[Table 8 - Software development life cycle 23](#_Toc17066441)

[Table 9 - Phase 1: Requirements Analysis 23](#_Toc17066442)

[Table 10 - Phase 2: Design 24](#_Toc17066443)

[Table 11 - Phase 3: Implementation 24](#_Toc17066444)

[Table 12 - Phase 4: Testing 24](#_Toc17066445)

[Table 13 - Phase 5: Maintenance 25](#_Toc17066446)

[Table 14 - Conceptual Diagram Data Dictionary 74](#_Toc17066447)

[Table 15 - Component Dictionary 77](#_Toc17066448)

[Table 16 - Class Dictionary 79](#_Toc17066449)

[Table 17 - <Class Diagram> User Attributes 80](#_Toc17066450)

[Table 18 - <Class Diagram> User Methods 80](#_Toc17066451)

[Table 19 - <Class Diagram> Admin Attributes 80](#_Toc17066452)

[Table 20 - <Class Diagram> User Methods 81](#_Toc17066453)

[Table 21- <Class Diagram> Stock-keeper Attributes 81](#_Toc17066454)

[Table 22- <Class Diagram> Stock-keeper Methods 81](#_Toc17066455)

[Table 23- <Class Diagram> Warehouse Accountant Attributes 82](#_Toc17066456)

[Table 24- <Class Diagram> Stock-keeper Methods 82](#_Toc17066457)

[Table 25- <Class Diagram> Shelf Attributes 82](#_Toc17066458)

[Table 26- <Class Diagram> Shelf Methods 83](#_Toc17066459)

[Table 27- <Class Diagram> Floor Attributes 83](#_Toc17066460)

[Table 28- <Class Diagram>Floor Methods 83](#_Toc17066461)

[Table 29- <Class Diagram>Cell Attribute 83](#_Toc17066462)

[Table 30- <Class Diagram>Cell Methods 83](#_Toc17066463)

[Table 31- <Class Diagram>Package Attributes 84](#_Toc17066464)

[Table 32- <Class Diagram>Package Methods 84](#_Toc17066465)

[Table 33- <Class Diagram> Product Attribute 84](#_Toc17066466)

[Table 34- <Class Diagram> Product Methods 84](#_Toc17066467)

[Table 35- <Class Diagram>Category Attributes 85](#_Toc17066468)

[Table 36- <Class Diagram> Category Methods 85](#_Toc17066469)

[Table 37- <Class Diagram>Box Attributes 85](#_Toc17066470)

[Table 38- <Class Diagram> Box Method 85](#_Toc17066471)

[Table 39- <Class Diagram> Vendor Attributes 85](#_Toc17066472)

[Table 40- <Class Diagram> Vendor Methods 86](#_Toc17066473)

[Table 41- <Class Diagram>Goods Receipt Attributes 86](#_Toc17066474)

[Table 42- <Class Diagram>Goods Receipt Methods 86](#_Toc17066475)

[Table 43- <Class Diagram> Goods Issue Attributes 86](#_Toc17066476)

[Table 44- <Class Diagram> Goods Issue Methods 87](#_Toc17066477)

[Table 45- <Class Diagram>Product\_Goods Receipt Attributes 87](#_Toc17066478)

[Table 46- <Class Diagram>Product\_Goods Receipt Methods 87](#_Toc17066479)

[Table 47- <Class Diagram>Product\_Goods Issue Attributes 87](#_Toc17066480)

[Table 48- <Class Diagram>Product\_Goods Issue Methods 88](#_Toc17066481)

[Table 49- <Class Diagram>Stocktake History Attributes 88](#_Toc17066482)

[Table 50- <Class Diagram>Stocktake History Methods 88](#_Toc17066483)

[Table 51 - Component Interface 103](#_Toc17066484)

[Table 54- <UI Design>Main Menu Button 105](#_Toc17066485)

[Table 55- <UI Design> Register Shelf Fields 106](#_Toc17066486)

[Table 56- <UI Design>Register Shelf Buttons 106](#_Toc17066487)

[Table 57- <UI Design> Register Package and Box Fields 108](#_Toc17066488)

[Table 58- <UI Design> Register Package and Box Button 108](#_Toc17066489)

[Table 59- <UI Design> Goods Receipt Fields 109](#_Toc17066490)

[Table 60- <UI Design>Goods Receipt Button 109](#_Toc17066491)

[Table 61- <UI Design> Stock In Fields 110](#_Toc17066492)

[Table 62- <UI Design>Stock In Buttons 111](#_Toc17066493)

[Table 63- <UI Design> Stock Out Fields 112](#_Toc17066494)

[Table 64- <UI Design> Stock Out Buttons 112](#_Toc17066495)

[Table 65- <UI Design>Goods Issue Fields 113](#_Toc17066496)

[Table 66- <UI Design> Goods Issue Button 113](#_Toc17066497)

[Table 67- <UI Design> Suggest Box Fields 114](#_Toc17066498)

[Table 68- <UI Design> Suggest Box Button 114](#_Toc17066499)

[Table 69- <UI Design> Transfer Box Fields 115](#_Toc17066500)

[Table 70- <UI Design>Transfer Box Button 116](#_Toc17066501)

[Table 71- <UI Design> Transfer Package Fields 117](#_Toc17066502)

[Table 72- <UI Design> Transfer Package Buttons 117](#_Toc17066503)

[Table 73- <UI Design> Stocktake Inventory Fields 118](#_Toc17066504)

[Table 74- <UI Design> Stocktake Inventory Buttons 119](#_Toc17066505)

[Table 92- <UI Design>Accountant Homepage Fields 120](#_Toc17066506)

[Table 93- <UI Design>Accountant Homepage Button 120](#_Toc17066507)

[Table 104- <UI Design>Create Product Fields 122](#_Toc17066508)

[Table 105- <UI Design>Create Product Buttons 122](#_Toc17066509)

[Table 106- <UI Design>Edit Product Fields 123](#_Toc17066510)

[Table 107- <UI Design>Edit Product Buttons 123](#_Toc17066511)

[Table 110- <UI Design>Import Product Fields 124](#_Toc17066512)

[Table 111- <UI Design> 124](#_Toc17066513)

[Table 119- <UI Design>Deactivate Shelf Buttons 125](#_Toc17066514)

[Table 120- <UI Design> Activate Shelf Button 126](#_Toc17066515)

[Table 121- <UI Design>Detail Shelf Fields 126](#_Toc17066516)

[Table 122- <UI Design> Edit Standard Shelf Size Fields 127](#_Toc17066517)

[Table 123- <UI Design>Edit Standard Shelf Size Buttons 127](#_Toc17066518)

[Table 124- <UI Design>View Goods Receipt Fields 128](#_Toc17066519)

[Table 125- <UI Design>View Goods Receipt Buttons 129](#_Toc17066520)

[Table 126- <UI Design>View Goods Issue Fields 130](#_Toc17066521)

[Table 127- <UI Design>View Goods Issue Buttons 130](#_Toc17066522)

[Table 130- <UI Design> Edit Prefix Fields 131](#_Toc17066523)

[Table 131- <UI Design> Edit Prefix Buttons 131](#_Toc17066524)

[Table 132- <UI Design> Create Invoice Fields 132](#_Toc17066525)

[Table 133- <UI Design> Create Invoice Buttons 133](#_Toc17066526)

[Table 134- <UI Design>Choose Product Fields 134](#_Toc17066527)

[Table 135- <UI Design>Choose Product Button 134](#_Toc17066528)

[Table 136- <UI Design>Edit Invoice Fields 134](#_Toc17066529)

[Table 137- <UI Design>Edit Invoice Button 135](#_Toc17066530)

[Table 138- <UI Design>Delete Invoice Button 135](#_Toc17066531)

[Table 139- <UI Design>Detail Invoice Fields 137](#_Toc17066532)

[Table 140- <UI Design>Create Report Buttons 137](#_Toc17066533)

[Table 145- <UI Design>Update Stocktake Status Buttons 138](#_Toc17066534)

[Table 146 - Entity Data Dictionary 139](#_Toc17066535)

[Table 147- <Algorithm> Shelf Coordinate Mapping Table 142](#_Toc17066536)

[Table 148 Database Data Ditionary 149](#_Toc17066537)

[Table 149 Database Data Dictionary 152](#_Toc17066538)

[Table 157- <Test Case>Create Product 156](#_Toc17066539)

[Table 160- <Test Case>Create Shelf 157](#_Toc17066540)

[Table 161- <Test Case>Update Shelf 159](#_Toc17066541)

[Table 162- <Test Case>Deactivate Shelf 160](#_Toc17066542)

[Table 163- <Test Case>Create Good Receipt 161](#_Toc17066543)

[Table 164- <Test Case>Update Good Receipt 162](#_Toc17066544)

[Table 165- <Test Case>Create Good Issue 164](#_Toc17066545)

[Table 166- <Test Case>Update Good Issue 165](#_Toc17066546)

[Table 167- <Test Case> Delete Invoice 166](#_Toc17066547)

[Table 168- <Test Case>Mobile Login 166](#_Toc17066548)

[Table 169- <Test Case>Register Cell 167](#_Toc17066549)

[Table 170- <Test Case>Register Package and Box 169](#_Toc17066550)

[Table 171- <Test Case>Stock in Package 171](#_Toc17066551)

[Table 172- <Test Case>Stock out Box 172](#_Toc17066552)

[Table 173- <Test Case>Transfer Package 173](#_Toc17066553)

[Table 174- <Test Case>Transfer Box 175](#_Toc17066554)

[Table 175- <Test Case>Stocktake Inventory 175](#_Toc17066555)

[Table 176 - Hardware requirements for Server 176](#_Toc17066556)

[Table 177 - Hardware requirements for PC 176](#_Toc17066557)

[Table 178 - Hardware requirements for Mobile 176](#_Toc17066558)

[Table 179 – Software Requirement 177](#_Toc17066559)

[Table 190- <User Guide>Create Product 181](#_Toc17066560)

[Table 191- <User Guide>Update Product 183](#_Toc17066561)

[Table 193- <User Guide>Import Product 183](#_Toc17066562)

[Table 197- <User Guide>Deactivate/Activate Shelf 184](#_Toc17066563)

[Table 198- <User Guide>Edit Standard Shelf Size 185](#_Toc17066564)

[Table 200- <User Guide>Create Invoice 188](#_Toc17066565)

[Table 201- <User Guide>Update Invoice 190](#_Toc17066566)

[Table 202- <User Guide>Delete Invoice 191](#_Toc17066567)

[Table 203- <User Guide>Create Report 191](#_Toc17066568)

[Table 205- <User Guide>Update Stocktake Status 192](#_Toc17066569)

[Table 206- <User Guide>Edit Prefix 193](#_Toc17066570)

[Table 208- <User Guide>Register Cell 194](#_Toc17066571)

[Table 209- <User Guide>Register Package and Box 196](#_Toc17066572)

[Table 210- <User Guide> Stock In 197](#_Toc17066573)

[Table 211- <User Guide>Stock Out 199](#_Toc17066574)

[Table 212- <User Guide>Transfer Box 199](#_Toc17066575)

[Table 213- <User Guide>Transfer Package 200](#_Toc17066576)

[Table 214- <User Guide>Stocktake Inventory 201](#_Toc17066577)

[Table 225- <取扱説明書>製品を作成する 203](#_Toc17066578)

[Table 226- <取扱説明書>製品を更新する 206](#_Toc17066579)

[Table 228- <取扱説明書>商品ファイルをインポートする 207](#_Toc17066580)

[Table 232- <取扱説明書>棚を無効化/有効化する 208](#_Toc17066581)

[Table 233- <取扱説明書>標準棚サイズを編集する 209](#_Toc17066582)

[Table 235- <取扱説明書>請求書を作成する 211](#_Toc17066583)

[Table 236- <取扱説明書>請求書を更新する 213](#_Toc17066584)

[Table 237- <取扱説明書>請求書を削除する 214](#_Toc17066585)

[Table 238- <取扱説明書>レポートを作成する 215](#_Toc17066586)

[Table 240- <取扱説明書>ストックテイクの状態を更新する 216](#_Toc17066587)

[Table 241- <取扱説明書>プレフィックスを編集する 217](#_Toc17066588)

[Table 243- <取扱説明書>棚（セル）を登録する 218](#_Toc17066589)

[Table 244- <取扱説明書>パッケージとボックスを登録する 220](#_Toc17066590)

[Table 245- <取扱説明書>在庫に入れる 221](#_Toc17066591)

[Table 246- <取扱説明書>在庫に出す 223](#_Toc17066592)

[Table 247- <取扱説明書>ボックスを転送する 224](#_Toc17066593)

[Table 248- <取扱説明書>パッケージを転送する 225](#_Toc17066594)

[Table 249- <取扱説明書>棚卸在庫 226](#_Toc17066595)

# List of Figures

[Figure 1 - Modified Waterfall - Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms [1] 20](#_Toc17066596)

[Figure 2 - System Overview Use Case 29](#_Toc17066597)

[Figure 12 - <Warehouse Accountant> Overview Use Case 30](#_Toc17066598)

[Figure 18 - <Warehouse Accountant> Create Product 30](#_Toc17066599)

[Figure 19 - <Warehouse Accountant> Update Product 33](#_Toc17066600)

[Figure 22 - <Warehouse Accountant> Create Shelf 35](#_Toc17066601)

[Figure 23 - <Warehouse Accountant> Update Shelf 37](#_Toc17066602)

[Figure 24 - <Warehouse Accountant> Deactivate Shelf 40](#_Toc17066603)

[Figure 26 - <Warehouse Accountant> Create Goods Receipt 41](#_Toc17066604)

[Figure 27 - <Warehouse Accountant> Create Goods Issue 43](#_Toc17066605)

[Figure 28 - <Warehouse Accountant> Update Goods Receipt 46](#_Toc17066606)

[Figure 29 - <Warehouse Accountant> Update Goods Issue 48](#_Toc17066607)

[Figure 30 - <Warehouse Accountant> Delete Invoice 50](#_Toc17066608)

[Figure 31 - <Warehouse Accountant> Create Report 52](#_Toc17066609)

[Figure 33 - <Warehouse Accountant> Update Stocktake Status 53](#_Toc17066610)

[Figure 34 - <Stock-keeper> Overview Use Case 55](#_Toc17066611)

[Figure 35 - <Stock-keeper> Register Cell 55](#_Toc17066612)

[Figure 36 - <Stock-keeper> Register Package and Box 57](#_Toc17066613)

[Figure 37 - <Stock-keeper> Stock in Package 59](#_Toc17066614)

[Figure 38 - <Stock-keeper> Stock out Box 61](#_Toc17066615)

[Figure 39 - <Stock-keeper> Stocktake Inventory 63](#_Toc17066616)

[Figure 40 - <Stock-keeper> Suggest Box 65](#_Toc17066617)

[Figure 41 - <Stock-keeper> Transfer Box 67](#_Toc17066618)

[Figure 42 - <Stock-keeper> Transfer Package 69](#_Toc17066619)

[Figure 43 - <Stock-keeper> Clear RFID Tag 71](#_Toc17066620)

[Figure 44 - Conceptual Diagram 73](#_Toc17066621)

[Figure 45 - System Architectural Design 75](#_Toc17066622)

[Figure 46 – Component Diagram 77](#_Toc17066623)

[Figure 47 – Class Diagram 78](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066624)

[Figure 48- <Sequence Diagram> Create Category 89](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066625)

[Figure 54- <Sequence Diagram> Deactivate Product 90](#_Toc17066626)

[Figure 57- <Sequence Diagram> Delete Invoice 91](#_Toc17066627)

[Figure 59- <Sequence Diagram> Update Invoice 92](#_Toc17066628)

[Figure 66- <Sequence Diagram> View User 93](#_Toc17066629)

[Figure 67- <Activity Diagram> Register Cell 94](#_Toc17066630)

[Figure 68- <Activity Diagram> Register Box and Package 95](#_Toc17066631)

[Figure 69- <Activity Diagram> Stock in 96](#_Toc17066632)

[Figure 70- <Activity Diagram> Stock Out 97](#_Toc17066633)

[Figure 71- <Activity Diagram> Stocktake Inventory 98](#_Toc17066634)

[Figure 72- <Activity Diagram> Transfer Box 99](#_Toc17066635)

[Figure 73- <Activity Diagram> Transfer Package 100](#_Toc17066636)

[Figure 75- <UI Design> Main Menu 104](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066637)

[Figure 76- <UI Design> Register Shelf 105](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066638)

[Figure 77- <UI Design>Register Packages and Box 107](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066639)

[Figure 78- <UI Design>Goods Receipt 109](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066640)

[Figure 79- <UI Design> Stock In 110](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066641)

[Figure 80- <UI Design> Stock Out 111](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066642)

[Figure 81- <UI Design> Goods Issue 113](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066643)

[Figure 82- <UI Design> Suggest Box 114](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066644)

[Figure 83- <UI Design> Transfer Box 115](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066645)

[Figure 84- <UI Design> Transfer Package 116](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066646)

[Figure 85- <UI Design> Stocktake Inventory 118](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066647)

[Figure 96- <UI Design>Accountant Homepage 119](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066648)

[Figure 103- <UI Design> Create Product 121](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066649)

[Figure 104- <UI Design>Edit Product 122](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066650)

[Figure 107- <UI Design> Import Product 124](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066651)

[Figure 112- <UI Design> Deactivate Shelf 125](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066652)

[Figure 113- <UI Design> Activate Shelf 125](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066653)

[Figure 114- <UI Design>Detail Shelf 126](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066654)

[Figure 115- <UI Design>Edit Standard Shelf Size 127](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066655)

[Figure 116- <UI Design> View Goods Receipt 128](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066656)

[Figure 116- <UI Design> View Goods Issue 129](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066657)

[Figure 117- <UI Design>Edit Prefix 131](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066658)

[Figure 118- <UI Design> Create Invoice [2] 132](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066659)

[Figure 119- <UI Design> Create Invoice [1] 132](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066660)

[Figure 120- <UI Design>Choose Product 133](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066661)

[Figure 121- <UI Design>Edit Invoice 134](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066662)

[Figure 122- <UI Design>Delete Invoice 135](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066663)

[Figure 123- <UI Design>Detail Invoice 136](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066664)

[Figure 124- <UI Design>Create Report 137](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066665)

[Figure 126- <UI Design>Update Stocktake Status 138](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066666)

[Figure 127 - Entity Relationship Diagram 139](#_Toc17066667)

[Figure 128<Algorithm> Optimizing Stock Out Product Flowchart 141](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066668)

[Figure 129- <Algorithm> Shelf Coordinate Map 142](#_Toc17066669)

[Figure 130- <Algorithm> Shelf Coordinate Mapping Flowchart 143](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066670)

[Figure 131- <Algorithm> Dijkstra Example 144](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066671)

[Figure 132- <Algorithm> Dijkstra Example 144](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066672)

[Figure 133- <Algorithm> Dijkstra Example 145](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066673)

[Figure 134- <Algorithm> Dijkstra Example 145](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066674)

[Figure 135- <Algorithm> Dijkstra Example 145](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066675)

[Figure 136- <Algorithm> Dijkstra Flowchart 146](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066676)

[Figure 137 – Database Diagram 148](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066677)

[Figure 138 – Prepare deployment package 177](#_Toc17066678)

[Figure 139 – Create Virtual Private Server 177](#_Toc17066679)

[Figure 140 – Copy Jar file 178](#_Toc17066680)

[Figure 141 – Copy installation file to device 178](#_Toc17066681)

[Figure 142 – Open file manage 179](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066682)

[Figure 143 - Install 179](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066683)

[Figure 161- <User Guide>Create Product [2] 180](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066684)

[Figure 162- <User Guide> Create Product [1] 180](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066685)

[Figure 163- <User Guide> Update Product [1] 181](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066686)

[Figure 164- <User Guide> Update Product [2] 182](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066687)

[Figure 167- <User Guide>Import Product [2] 183](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066688)

[Figure 168- <User Guide>Import Product [1] 183](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066689)

[Figure 174- <User Guide>Deactivate/Activate Shelf [1] 184](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066690)

[Figure 175- <User Guide>Deactivate/Activate Shelf [2] 184](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066691)

[Figure 176- <User Guide>Edit Standard Shelf Size [1] 185](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066692)

[Figure 177- <User Guide>Edit Standard Shelf Size [2] 185](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066693)

[Figure 181- <User Guide>Create Invoice [2] 186](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066694)

[Figure 179- <User Guide>Create Invoice [1] 186](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066695)

[Figure 183- <User Guide>Create Invoice [4] 187](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066696)

[Figure 182- <User Guide>Create Invoice [3] 187](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066697)

[Figure 184- <User Guide>Update Invoice [1] 188](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066698)

[Figure 186- <User Guide>Update Invoice [3] 189](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066699)

[Figure 185- <User Guide>Update Invoice [2] 189](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066700)

[Figure 187- <User Guide>Delete Invoice [1] 190](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066701)

[Figure 188- <User Guide>Delete Invoice [2] 190](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066702)

[Figure 189- <User Guide>Create Report 191](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066703)

[Figure 191- <User Guide>Update Stocktake Status [2] 192](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066704)

[Figure 192- <User Guide>Update Stocktake Status [1] 192](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066705)

[Figure 177- <User Guide>Edit Prefix [2] 193](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066706)

[Figure 176- <User Guide>Edit Prefix [1] 193](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066707)

[Figure 194- <User Guide>Register Cell 194](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066708)

[Figure 195- <User Guide>Menu 194](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066709)

[Figure 198- <User Guide>Good Receipt Info 195](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066710)

[Figure 197- <User Guide>Register Package and Box 195](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066711)

[Figure 196- <User Guide> Menu 195](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066712)

[Figure 199- <User Guide>Stock In 196](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066713)

[Figure 200- <User Guide>Menu 196](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066714)

[Figure 201- <User Guide>Stock Out 197](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066715)

[Figure 202- <User Guide>Menu 197](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066716)

[Figure 203- <User Guide>Goods Issue 198](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066717)

[Figure 204- <User Guide>Suggest Box 198](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066718)

[Figure 205- <User Guide>Menu 199](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066719)

[Figure 206- <User Guide>Transfer Box 199](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066720)

[Figure 207- <User Guide>Menu 200](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066721)

[Figure 208- <User Guide>Transfer Package 200](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066722)

[Figure 210- <User Guide>Stocktake Inventory 201](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066723)

[Figure 209- <User Guide>Menu 201](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066724)

[Figure 161- <取扱説明書>製品を作成する[2] 202](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066725)

[Figure 162- <取扱説明書>製品を作成する[1] 202](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066726)

[Figure 163- <取扱説明書>製品を更新する[1] 204](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066727)

[Figure 164- <取扱説明書>製品を更新する[2] 205](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066728)

[Figure 167- <取扱説明書>商品ファイルをインポートする[2] 206](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066729)

[Figure 168- <取扱説明書>商品ファイルをインポートする[1] 206](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066730)

[Figure 174- <取扱説明書>棚を無効化/有効化する[1] 207](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066731)

[Figure 175- <取扱説明書>棚を無効化/有効化する[2] 207](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066732)

[Figure 176- <取扱説明書>標準棚サイズを編集する[1] 208](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066733)

[Figure 177- <取扱説明書>標準棚サイズを編集する[2] 208](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066734)

[Figure 179- <取扱説明書>請求書を作成する[1] 209](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066735)

[Figure 181- <取扱説明書>請求書を作成する[2] 209](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066736)

[Figure 183- <取扱説明書>請求書を作成する[5] 210](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066737)

[Figure 182- <取扱説明書>請求書を作成する[3] 210](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066738)

[Figure 184- <取扱説明書>請求書を更新する[1] 211](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066739)

[Figure 186- <取扱説明書>請求書を更新する[3] 212](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066740)

[Figure 185- <取扱説明書>請求書を更新する[2] 212](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066741)

[Figure 187- <取扱説明書>請求書を削除する[1] 213](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066742)

[Figure 188- <取扱説明書>請求書を削除する[2] 214](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066743)

[Figure 189- <取扱説明書>レポートを作成する 214](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066744)

[Figure 191- <取扱説明書>ストックテイクの状態を更新する[2] 215](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066745)

[Figure 192- <取扱説明書>ストックテイクの状態を更新する[1] 215](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066746)

[Figure 177- <取扱説明書>プレフィックスを編集する[2] 216](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066747)

[Figure 176- <取扱説明書>プレフィックスを編集する[1] 216](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066748)

[Figure 194- <取扱説明書>棚（セル）を登録する[1] 217](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066749)

[Figure 195- <取扱説明書>棚（セル）を登録する[2] 217](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066750)

[Figure 196- <取扱説明書>パッケージとボックスを登録する[1] 218](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066751)

[Figure 197- <取扱説明書>パッケージとボックスを登録する[2] 218](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066752)

[Figure 198- <取扱説明書>パッケージとボックスを登録する [3] 219](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066753)

[Figure 199- <取扱説明書>在庫に入れる[1] 220](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066754)

[Figure 200-<取扱説明書>在庫に入れる[2] 220](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066755)

[Figure 201- <取扱説明書>在庫に出す[1] 221](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066756)

[Figure 202- <取扱説明書>在庫に出す[2] 221](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066757)

[Figure 203- <取扱説明書>在庫に出す[3] 222](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066758)

[Figure 204- <取扱説明書>在庫に出す[4] 222](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066759)

[Figure 205- <取扱説明書>ボックスを転送する[1] 223](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066760)

[Figure 206- <取扱説明書>ボックスを転送する[2] 223](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066761)

[Figure 207- <取扱説明書>パッケージを転送する[1] 224](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066762)

[Figure 208- <取扱説明書>パッケージを転送する[2] 224](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066763)

[Figure 210- <取扱説明書>棚卸在庫[2] 226](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066764)

[Figure 209- <取扱説明書>棚卸在庫[1] 226](file:///C:\Users\KIROV\Desktop\RFIM_200_Page.DOCX#_Toc17066765)

# A. Introduction

## Project Information

* Project name:  **Motorbike Accessories Inventory Management**
* Project Code: **RFIM**
* Product Type: **Website Application, Android Application**
* Start Date: **May 13th, 2019**
* End Date: **April 27th, 2019**

## Introduction

Nowadays, the motorcycle industry has become one of the fastest growing industries. It means that the accessories contained in the inventory are getting larger and larger. Besides, the stock in and stock out processes also occur regularly. Therefore, the need for a method of managing motorbike’s accessories that is precise and quick is necessary. An effective motorbike’s accessories management software not only minimizes the need for manual work, but also increases productivity and reduces the loss of products.

In this document, we would like to introduce a solution that integrates RFID technology into this management software called Motorbike Accessories Inventory Management. This system will help the warehouse managers update as well as manage information of motorbike accessories, goods receipts, good issues, goods receipts, stock-take products easily. This document will also detail our work process in the last 4 months including our perspective on the system, component designs, and detailed core workflow.

## Current Situation

Most motorcycle accessory manufacturers and suppliers manage their products by using barcode “archetype” or even traditional excel files and papers.

With traditional excel files and papers, the stock-keeper has to count numerous of motorcycle accessories by his eyes then note those numbers to papers during goods receipt, goods issue or stocktaking process. With barcode archetype, the amount of time spending for counting items is reduced by technology. Stock-keeper will use a barcode scanner to scan instead of counting on his own, then the products’ information will be automatically updated into an inventory management system.

When those previous processes are completed and don’t have any mistakes, stock-keeper will create an excel format file instead of raw paper then transfer it to the accounting department for collating and printing the receipt as the finish step. The receipt has 3 copies: 1 was kept by the stock-keeper, 1 by the accountant and 1 by the deliverer.

## Problem Definition

After doing some research, we can see some disadvantages that still exist in current inventory management by using the barcode:

* Time-consuming: Barcode is designed to be scanned once at a time so that when we need to import a bunch of products, we have to spend a lot of time to scan each product.
* Unclear line of sight: Without the clear line of sight, the scanner cannot read the barcode. Sometimes the barcode was pasted in a convex or concave surface that makes the scanner difficult to scan.
* Physical damage: Barcodes are generally printed on paper or plastic, so they are easy to become blur and be torn when they are stored in stock for a long time or contact with dust and humidity. The damaged barcodes cannot be read by the scanner so that the user has to input information by hand.
* Security problem: Barcode can be easily reproduced or forged. Someone can use this problem to make fake products and sell them to the market that affects a lot to the enterprise’s benefit.

## Proposed Solution

To solve these problems, our proposed solution is to build a system called “RFIM”. The system supports enterprise organizes and manages the inventory by using RFID technology that helps reduce a lot of time when importing, exporting or inventorying products.

### Feature Functions

* + Managing motorbike accessories inventory’s information: product, category, quantity, shelf, etc.
  + Managing receipt and issue process of motorbike accessories, inventory control with stocktaking and transfer product feature
  + Using RFID technology to improve the performance when perform above tasks

### Advantages and Disadvantages

The advantages and disadvantages of the proposed solution:

* **Advantages**:
* Time-saving: The duration when the staff checks the quantity of importing or exporting products can be reduced because RFID tags can be read more than once at the same time.
* Wide range scanning: The RFID reader can be scanned with the range up to 100m.
* Durability: Because the RFID tags are made from durable material so that they are really difficult to be damaged.
* Security: RFID can be saved by using encryption.
* **Disadvantages**:
* The most disadvantageous of RFID is the investing cost. The RFID implementation’s cost is more expensive than the barcode itself because the price of RFID scanner devices and tags is too high.
* RFID scanners are affected by metal and liquid.
* If multiple tags in the same area respond at the same time, the collision can occur.

## Functional Requirements

Function requirements of the system are listed as below:

**Admin:**

* Manage users
* View current users list

**Warehouse Accountant:**

* View and manage products
* View and manage categories
* View cells and manage shelves
* View and manage goods receipt and goods issue.
* Create and export pdf reports (issue, receipt, stocktaking…)
* View stocktaking history and update stocktake status

**Stock-keeper:**

* Transfer products (transfer packages and transfer boxes)
* Stocktake inventory
* Stock in and stock out packages of product’s items with created goods receipt or issue
* Register packages
* Register cells
* Clear RFID tag
* Suggest Boxes

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Full Name** | **Role** | **Position** | **Contact** |
|  |  |  |  |  |
| 1 | Lại Đức Hùng | Project Manager | Supervisor | hungld5@fe.edu.vn |
| 2 | Phạm Minh Hoàng | Developer | Leader | hoangpmse62769@fpt.edu.vn |
| 3 | Nguyễn Trường Thịnh | Developer | Member | thinhntse61887@fpt.edu.vn |
| 4 | Đỗ Trung Hiếu | Developer | Member | hieudtse61847@fpt.edu.vn |

Table 1 – Role and Responsibility

# B. Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

* Official name: Motorbike Accessories Inventory Management With RFID.
* Vietnamese name: Quản lý kho phụ tùng xe máy với công nghệ RFID.
* Abbreviation: RFIM

### Problem Abstract

* Project is concerned about the development of motorbike leading to increasingly difficult accessories management, so we use RFID technology to handle this problem. We call it Motorbike Accessories Inventory Management with RFID (RFIM). RFIM will provide a user-friendly interface and easy-to-use solution to manage motorbike parts.
* Companies use warehouse management system for managing information of products, of users, etc. With the existing systems that use barcode, the can only scan one code at a time, scan in the clear line of sight, and the barcode is easy to become blur and be torn. Therefore, it is hard to manage the correct, authentic information, quantity of motorbike accessories.
* The Motorbike Accessories Inventory Management System will be a useful assistant for every stores, every industry to manage motorbike accessories easily and effectively.

### Project Overview

#### The Current Situation

Below are the problems of the current situation:

• Schedule of team members: team members have different time for studying and working so that it is hard to arrange the available time for the group offline meeting.

• New techniques: team members are all JS students and don’t have much experience to use embedded system. The team needs an amount of time to research and integrate into system.

• Lack of knowledge about warehouse management process: Our team members are students, and no one knows about the warehouse management business’s knowledge. Therefore, the team needs an amount of time to take survey of warehouse management process.

• Lack of budget: The team can’t afford modern and expensive RFID tools, just be able to buy cheap tools for the demonstration.

#### The Proposed System

According to the technology researches, we found out that RFID is an effective solution to solve motorbike warehouse problems. We build a web application that allows end users manage users, motorbike parts. End users include admin, warehouse accountant. Admin can manage users, permissions. Warehouse accountant can manage products, goods receipts, goods issues, shelves, categories, reports. We also build a mobile application that allows stock-keeper to stock in, stock out, register, and maintain products.

Tasks will be assigned vertically to team members, so that if one member quits, the team will not lack of resources.

#### Boundaries of the System

**The system can**:

* Allow Admin to manage user accounts .
* Allow Warehouse Accountant to manage products, shelves, categories, goods receipts, goods issues, stocktaking history and reports.
* Allow Stock-keeper to register cells, register package and boxes, stock in package, stock out box, stocktaking inventory, transfer products

**The system cannot:**

* RFID can’t scan multiple tags because of the cheap RFID scanner.
* Due to limitation of technology and funds, RFIM system can only bind RFID tags to product’s boxes, not the product items inside
* The stocktake process report all missing boxes while stocktaking but system doesn’t cover the case that how to handle missing boxes’s case when stocktake process report.
* The system can’t support when warehouse need to decrease shelf’s number of floor or shelf‘s number of cell because it can reduce to effective management.

#### Future Plans

* The existing systems can only scan one RFID tag at one time. We design our system to scan multiple RFID tags with a wide range.
* We can stick RFID tags to product items for managing effectively.
* We build system to minimize lost of products in warehouse.
* We build this system for medium motorbike companies or stores so it doesn’t support automated system and stock-keepers have to scan the tags manually. We will integrate automated system for managing warehouse.

#### Development Environment

##### Hardware Requirements

* **For server**

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirement** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps or more) |
| **Operation System** | XP, Vista, 7, 10, Window Server 2008 | 10, Window server 2008 |
| **Computer Processor** | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz) |
| **Computer memory** | 4GB RAM | 32 GB RAM or more |
| **Storage space** | 1GB | 5GB or more |

Table 2 - Hardware requirements for Server

* **For PC**

|  |  |  |
| --- | --- | --- |
| **PC** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window 7 | Window 7 or more. |
| **Computer Processor** | Intel® Core i3 1.4GHz | Intel® Core i5 2.50GHz |
| **Computer Memory** | 1GB RAM | 2GB RAM or more |
| **Web Browser** | Chromes (v42 or higher) | Chrome latest stable version |

Table 3 - Hardware requirements for PC

* **For Mobile**

|  |  |  |
| --- | --- | --- |
| **Mobile** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Wi-Fi (4 Mbps) | Wi-Fi (8 Mbps) |
| **Operating System** | Android 6.0 | Android 7.1.1 |
| **Mobile Processor** | SnapDragon 625 | SnapDragon 625 |
| **Mobile Memory** | 2GB RAM | 4GB RAM or more |

Table 4 - Hardware requirements for Mobile

##### Software Requirements

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| **Operation System** | Window Server 2012 | Operating system and platform for development |
| **Environment** | .NET Framework, Java, C, C# | Specification for developing web application |
| **Modeling tool** | StarUML | Designing diagram |
| **IDE** | Android Studio 3.4.1, Visual Studio 2017, Arduino IDE, IntelliJ 2019.1.3 | Programming tools |
| **DBMS** | SQL Server 2017, Amazon Web Services | Creating & managing the database for system |
| **Source control** | Git on IDE (Sourcetree) | Controlling the source code |
| **Web browser** | Chrome 69 or above | Testing system on browser |

Table 5 - Software requirements for PC

## Project Organization

### Software Process Model

This project is developed using Waterfall model for Software development project. Our team choose Waterfall model because of the following reasons:

* Our team only has 4 members, and this project for medium motorcycle accessories warehouses, no multi-warehouses available.
* For a classic problem “Warehouse management” which most of the requirements were clarified in real life, we decided to choose Modified Waterfall Model
* Modified Waterfall model is easy to progress and maintain, also has lower risk when comparing to the classic Waterfall model.

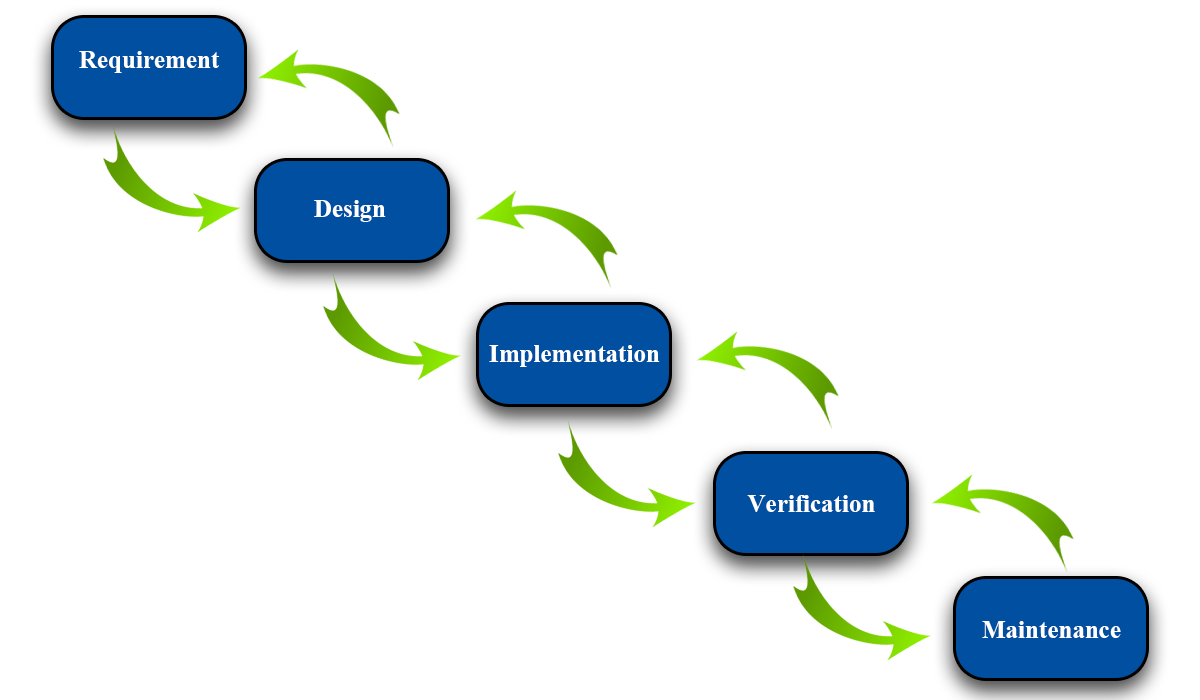


Figure 1 - Modified Waterfall - Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms [1]

### Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Lại Đức Hùng | Product Owner | * Specifying user requirements * Controlling the development process * Giving out technique and business analysis support |
| **2** | Phạm Minh Hoàng | Team Leader | * Managing process * Designing database * Clarifying requirements * Preparing documents * Designing GUI * Creating test plan * Coding * Testing * Arranging Meeting * Managing Risks |
| **3** | Nguyễn Trường Thịnh | Team Member | * Designing database * Clarifying requirements * Preparing documents * Designing GUI * Creating test plan * Coding * Testing |
| **4** | Đỗ Trung Hiếu | Team Member | * Designing database * Clarifying requirements * Preparing documents * Designing GUI * Creating test plan * Coding * Testing |

Table 6 - Role and Responsibilities

### Tools and Techniques

|  |  |
| --- | --- |
| **Tool/Technique** | **Name** |
| Front-end | HTML, CSS, JavaScript, jQuery |
| Back-end | ASP.NET Core 2.1, Entity Framework, Spring Boot , Java |
| IDE | Android Studio 3.4.1, Visual Studio 2017, Arduino IDE, IntelliJ 2019.1.3 |
| DBMS | SQL Server 2017, Amazon AWS |
| Source Control | Github, Sourcetree |
| Modeling tool | StarUML |

Table 7 - Tools and Techniques

## Project Management Plan

### Software development life cycle

| **Phase** | **Description** | **Deliverables** | **Resource needed** | **Dependencies and Constrains** | **Description** |
| --- | --- | --- | --- | --- | --- |
| Requirement Analysis | Collect and analyze system requirements | Software Requirement Specification | 20 man- days | N/A | * Missing requirement * Unclear scope project’s scope * Lack of member share of understanding |
| System and Software Design | * Design the structure of the system * Choose Architecture style | Software Design Document | 20 man- days | Depend on requirement after analyzed and system architecture design. | * Lack of experience from real project * Requirement is too complex and easy to misunderstand |
| Implementation and Unit Testing | * Implement all system’s function * Create test plan * Perform Unit test | Software package and hardware | 50 man- days | Depend on Software Requirement Specification, Software Design Description and Hardware | * Lack of experience and knowledge * Human mistake * Hardware error |
| Integration and System Testing | * Integration test the system * Alpha test * Correct bugs * Beta test | Test case | 20 man- days | Depend on “Implementation” | * Lack of experience * Missing test case |
| Operation and Maintenance | Deploy on sever, web and mobile | * Installation guide * Software User’s Manual | 10 man- days | Depend on “Testing” | * Lack of experience * Customers do not understand User’s Manual |

Table 8 - Software development life cycle

### Phase Detail

Phase 1: Requirements Analysis

| **Task** | **Description** | **Author** |
| --- | --- | --- |
| **1. Collect requirements** | Research similar application to collect requirement.  Observation real situation. | HoangPM, ThinhNT, HieuDT |
| **2. Identify and clarify main functions** | Define main system’s function. | HoangPM, ThinhNT, HieuDT |
| **3. Create System** **Introduction** | Complete Introduction Report. | HoangPM, ThinhNT, HieuDT |
| **4. Software Project**  **Management Plan** | Create Project Management Plan. | HoangPM, ThinhNT, HieuDT |
| **5. Prototype** | Build a prototype of proposed system (Website/Mobile). | HoangPM, ThinhNT, HieuDT |
| **6. SRS** | Create Software Requirement Specification document. | HoangPM, ThinhNT, HieuDT |

Table 9 - Phase 1: Requirements Analysis

Phase 2: Design

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| 1. Database Design | Design database for the RFIM system | HoangPM, ThinhNT, HieuDT |
| 2. Technology research | Study Arduino platform  Study Bluetooth on Mobile Device Technology | HoangPM |
| 3. Design Document | Create software design document | HoangPM, ThinhNT, HieuDT |

Table 10 - Phase 2: Design

Phase 3: Implementation

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. RFIM’s functions** | Implement function of RFIM system | HoangPM, ThinhNT, HieuDT |
| **2. Front-end web functions** | Implement front-end functions on web | ThinhNT, HieuDT |
| **3. Back-end web functions** | Implement back-end functions on web | ThinhNT, HieuDT |
| **4. Mobile functions** | Implement mobile application | HoangPM |
| **5. RFID Scanner** | Assembly RFID Scanner  Implement embedded code for RFID Scanner | HoangPM |
| **6. Unit testing** | Write test case and test for web functions | ThinhNT, HieuDT |
| Write test case and testing for mobile functions | HoangPM |

Table 11 - Phase 3: Implementation

Phase 4: Testing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Integration testing** | Write test case and test systems | ThinhNT, HieuDT |
| **2. Acceptance testing** | Let customer test real product to ensure the product meet user’s requirements | HoangPM, ThinhNT, HieuDT |

Table 12 - Phase 4: Testing

Phase 5: Operation and Maintenance

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| **1. Installation guide** | Write installation guide | HoangPM , HieuDT |
| **2. Create User’s Manual** | Write user manual | HieuDT, ThinhNT |

Table 13 - Phase 5: Maintenance

### Meeting Minutes

All meeting documents could be found here:

<https://drive.google.com/open?id=1tj0KZxtxWliY2fBINYbQjqLgHpzjM3op> (Security: Must be login with fpt university’s email).

## Coding Convention

* **For C#:**

1. **Naming Conventions:**
   * Use pascal case for class and method names.
   * Use camel case for method arguments and local variables.
   * Do not use underscores in identifiers. Except: can prefix private static variables with an underscore.
   * Prefix interfaces with the letter ‘I’.
   * Use noun or noun phrases to name a class.
2. **Layout Conventions:**
   * If continuation lines are not indented automatically, tabs must be set exactly every 4 spaces.
   * Avoid lines longer than 80 characters.
   * Write only one statement per line.
   * Write only one declaration per line.
   * Add at least one blank line between method definitions and property definitions.
3. **Declaration:**
   * Use implicit type ‘var’ for local variable declarations. Except: primitive types (int, string, double, etc) use predefined names.
   * Organize namespaces with a clearly defined structure.
4. **Commenting Conventions:**
   * Place the comment on a separate line.
   * Begin comment text with an uppercase letter and end with a period.
   * Add one space between the comment delimiter (//) and the comment text.

***References:*** *C# Coding Conventions* [2]

* **For Java**
* **Naming Conventions:**
  + Use camel case for method arguments, method name and local variables.
  + Use uppercase with words separated by underscores (“\_”) for constant
  + Use all-lowercase for package name
  + Use verbs for method name
  + Use nouns for class name and the first word should be uppercase
* **Layout Conventions:**
  + If continuation lines are not indented automatically, tabs must be set exactly every 4 spaces.
  + Avoid lines longer than 80 characters.
  + Write only one statement per line.
  + Write only one declaration per line.
  + Add at least one blank line between method definitions and property definitions.
* **Declaration:**
  + Use one declaration per line.
  + Put declarations only at the begin of the block
  + Do not use space between method name
  + Method are separated by black line
  + Open brace at the same line of method declaration
* **Commenting Conventions:**
  + Place the comment on a separate line.
  + Begin comment text with an uppercase letter and end with a period.
  + Add one space between the comment delimiter (//) and the comment text.

***References:*** *Java Coding Conventions* [3]

# C. Software Requirement Specification

## User Requirement Specification

### Guest Requirement

Guest is a person who doesn’t have any access to the system. Guest can only log in the system in order to be able to use all further functions of the system. Guest has the following function:

* Login

### Admin Requirement

Admin is a user who works directly with the system and has the rights to manage users’ information existed in the system. Admin can use the following functions:

* Manage Users
* Logout: after finishing all the activities on the system, admin can log out of the system

### Warehouse Accountant Requirement

Warehouse Accountant is a user who manages all the information of the products, goods, and accessories appearing in the warehouse, and has the responsibilities of managing the goods receipts, issues and reports. Warehouse Accountant has the following functions:

* View and Manage Products
* View and Manage Category
* View Cells and Manage Shelves
* View and Manage Goods Receipts, Goods Issues
* View Stocktake History and Update Stocktake Status
* Create Reports
* Logout: after finishing all the activities on the system, admin can log out of the system

### Stock-keeper Requirement

Stock-keeper is a user who has the responsibilities of works related to using RFID to manage products, packages, boxes. Stock-keeper has the following functions:

* Register Cells
* Register Packages
* Stock in Packages
* Stock out Boxes
* Clear RFID Tags (Automatically)
* Stocktake Inventory
* Transfer Products: transfer by boxes or packages
* Suggest Boxes
* Logout: after finishing all the activities on the system, admin can log out of the system

## System Requirement Specification

### External Interface Requirement

#### User Interface

* The general requirement of the GUI is it should be simple, clear, intuitive, and reminiscent.
* The interface design is an iterate process includes: design, sketching, prototyping, user assessment.
* The UI of both of mobile and web application use English language.

#### Hardware Interface

* PC: connected to Wifi, 4GB of RAM minimum, CPU 1.4GHz.
* Android device: connected to Wifi or LTE, connected to Bluetooth v4.0 or higher, 1GB of RAM minimum, CPU 1.2 GHz.
* RFID Scanner: HC-06 Bluetooth board, Arduino UNO ESP8266

#### Software Interface

* Web application: work with Chrome browser Firefox (v52 or higher), Chromes (v28 or higher), Internet Explorer (v10 or above) or with any web browsers that support HTML5 & CSS3.
* Mobile application: Android operating system (v6.0 or above).



#### Communication Protocol

* Use HTTP protocol 1.1 for communication between web system and cloud database.
* Use HTTP protocol 1.1 for communication between mobile application and web server.

### System Overview Use Case



Figure 2 - System Overview Use Case

### List of Use Cases

#### <Warehouse Accountant> Overview Use Case

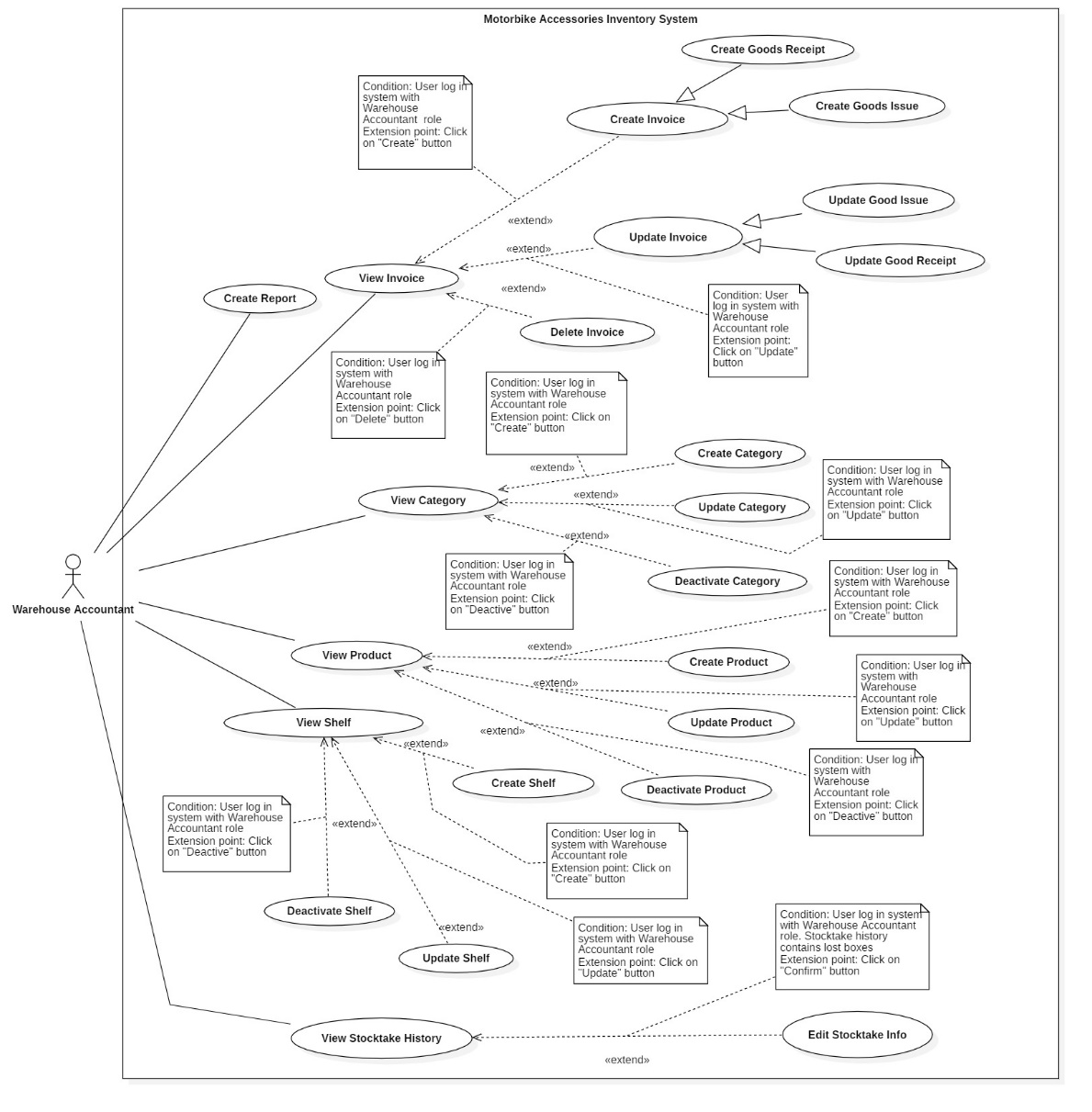
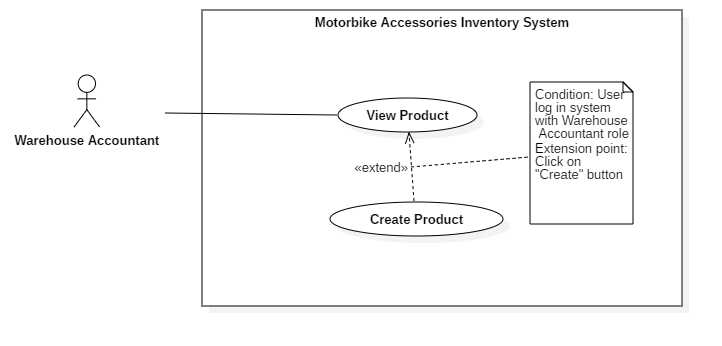


Figure 12 - <Warehouse Accountant> Overview Use Case

##### <Warehouse Accountant> Create Product

****Figure 18 - <Warehouse Accountant> Create Product

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_P2** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_P2** | **Use Case Version** | 2.0 |
| **Use Case Name** | Create Product | | |
| **Author** | HieuDT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to create new products.   **Goal:**   * Warehouse Accountant creates new products successfully.   **Triggers:**   * Warehouse Accountant clicks on “Create” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: New product is created successfully. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Create” button on “Products” screen. | “Add Product” screen is shown with following labels and fields:   * Product ID: text, required * Product Name: text, required * Category: select option, list Category * Length (cm): float, optional * Width (cm): float, optional * Height (cm): float, optional * Quantity per Box: integer, required * Vendor: select option, list Vendor * Weight (g): double, optional * Image: text, optional * Description: text, optional | | 2 | Warehouse Accountant fills out data fields and clicks on “Confirm” button.  [Alternative 1, 2] | System validates inputted data and shows “Products” screen.  [Exception 1, 2, 3, 4] |   **Alternative Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Import” button on “Products” screen. | “Import Product” screen is shown with following labels and fields:   * Upload file: file chooser, required. | | 2 | Warehouse Accountant chooses file and clicks on “Confirm” button. | System validates the uploaded file whether it contains the same fields with sample file and adds list of products, then shows “Products” screen.  [Exception 4, 5, 6, 7] |   **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | * Product ID is empty * Product Name is empty * Quantity per Box is empty | System shows error message "This field is required.” and highlights the empty fields. | | 2 | Product Id already exists. | System show error message “Product ID already exists!” and highlights the error field. | | 3 | Warehouse Accountant clicks on “Cancel” button. | System shows “Products” screen. | | 4 | Network connection problem. | System shows error message: “Please check your connection.” | | 5 | File is not chosen. | System shows error message “Please choose a file.” | | 6 | File extension is not .xls or .xlsx. | System shows error message “Please choose any file with .xls or .xlsx extension.” | | 7 | Uploaded file’s fields are not as same as the fields of “Sample File” | System shows error message “File has wrong format. Please check “Sample File” format.” |   **Relationships:** Extended from View Product use case.  **Business Rules:**   * In this system, only Warehouse Accountant can create new products. * Product ID will be generated automatically with RFIM prefix and number which is ascending. * Each product must belong to one category and one vendor. * One box only has one type of product, but can contain many items of that product. This rule also applies with packages. * “Import Products” feature requires Excel data file (.xls, .xlsx) that must follow the format of “Sample File”. * Quantity per Box determines how many items that product has in one box. * Length, Width and Height are the 3 dimensions of each product’s box. | | | |

<Warehouse Accountant> Create Product use case specification

##### <Warehouse Accountant> Update Product

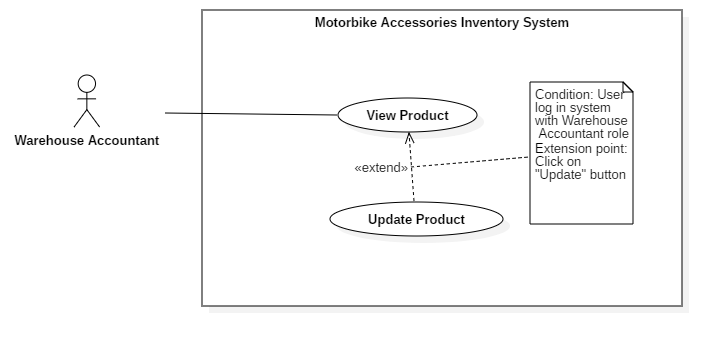


Figure 19 - <Warehouse Accountant> Update Product

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_P3** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_P3** | **Use Case Version** | 2.0 |
| **Use Case Name** | Update Product | | |
| **Author** | HieuDT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to update products’ information.   **Goal:**   * Warehouse Accountant updates information of products successfully.   **Triggers:**   * Warehouse Accountant clicks on “Update” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: Information of products will be updated. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Update” button on “Products” screen. | “Update Product” screen is shown with following labels and fields which have current product’s information:   * Product ID: prefilled, not editable * Product Name: text, required * Category: select option, list Category * Length (cm): float, optional * Width (cm): float, optional * Height (cm): float, optional * Quantity per Box: integer, required * Vendor: select option, list Vendor * Weight (g): double, optional * Image: text, optional * Description: text, optional | | 2 | Warehouse Accountant fills out data fields and clicks on “Confirm” button.  [Alternative 1, 2] | System validates inputted data and shows “Products” screen.  [Exception 1, 2, 3] |   **Alternative Scenarios:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Update” button in “Product Details” screen. | “Update Product” screen is shown with following labels and fields which have current product’s information:   * Product ID: fixed * Product Name: text, required * Category: select option, list Category * Length (cm): float, optional * Width (cm): float, optional * Height (cm): float, optional * Quantity per Box: integer, required * Vendor: select option, list Vendor * Weight (g): double, optional * Image: text, optional * Description: text, optional | | 2 | Warehouse Accountant fills out data fields and clicks on “Confirm” button. | System validates inputted data and updates product’s information then shows “Products” screen.  [Exception 1, 2, 3] |   **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Editing Product Name already exists | System shows error message “Product Name already exists!” | | 2 | Warehouse Accountant clicks on “Cancel” button | System shows “Products” screen. | | 3 | Network connection problem. | System shows error message: “Please check your connection”. |   **Relationships:** Extended from View Product use case.  **Business Rules:**   * In this system, only Warehouse Accountant can update prodcuts’ information. * Product ID is fixed and can’t be updated. * Each product must belong to one category and one vendor. * One box only has one type of product, but can contain many items of that product. This rule also applies with packages. * Quantity per Box determines how many items that product has in one box. * Length, Width and Height are the 3 dimensions of each product’s box. | | | |

<Warehouse Accountant> Update Product use case specification

##### <Warehouse Accountant> Create Shelf

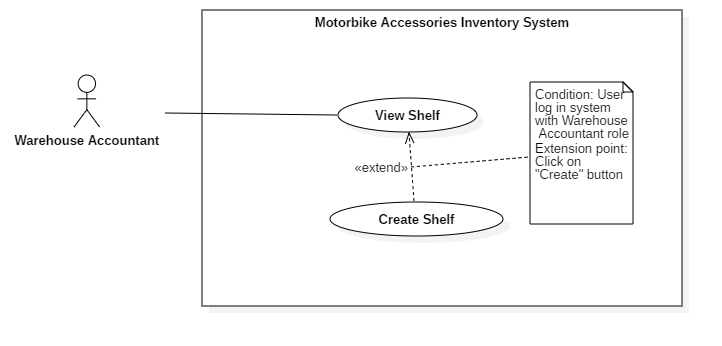


Figure 22 - <Warehouse Accountant> Create Shelf

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_S2** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_S2** | **Use Case Version** | 2.0 |
| **Use Case Name** | Create Shelf | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to create shelves.   **Goal:**   * Warehouse Accountant creates shelf successfully.   **Triggers:**   * Warehouse Accountant clicks on “Create” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: A new shelf with cells will be created. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Create” button in “Shelves” screen. | “Add Shelf” screen is shown with following labels and fields:   * Shelf ID: text, required * Description: text, optional * Number of Floor: integer, required * Number of Cell: integer, required * Cell Height (cm): float, optional * Cell Width (cm): float, optional * Cell Length (cm): float, optional * Coordinate X: integer, required * Coordinate Y: integer, required | | 2 | Warehouse Accountant fills out data fields and clicks on “Confirm” button. | System validates inputted data and shows “Shelves” screen.  [Exception 1, 2, 3, 4] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | * Shelf ID is empty * Coordinate X is empty * Coordinate Y is empty | System shows error message "This field is required.” and highlights the empty fields. | | 2 | Shelf ID already exists. | System shows error message “Shelf ID already exists!” and highlights the error field. | | 3 | Warehouse Accountant clicks on “Cancel” button. | System shows “Shelves” screen. | | 4 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Extended from View Shelf use case.  **Business Rules:**   * In this system, only Warehouse Accountant can create shelves. * Created shelves can be registered with a specified RFID tag. * “Number of Floor” field shows the number of floors that shelf has. This warehouse uses Selective Pallet Racking Systems which commonly has most maximum height is 5 floors because the forklift can’t bypass that height limit. * “Number of Cell” field shows the number of cells in each floor of that shelf. Number of Cell is a fixed number which is determined by the first created shelf. All shelves inside the warehouse must have the same number of cells per floor for easy maintenance, stocking in and stocking out process. * “Cell Width”, “Cell Length” and “Cell Height” are the 3 dimension of Cell size to let stock-keeper know when Cell is full. * “Number of Floor” and “Number of Cell” fields are prefilled by configurable value from “Standard Cell Size”. * Coordinate X is position of that cell in horizontal, Coordinate Y is position of that cell in vertical (ex: CoorX = 1, Coor Y = 1 => Shelf is the closest to the warehouse door) * Cell code will be created automatically with the following format: Shelf Code-Floor-Cell. (ex: A-4-5 means Shelf A - 4th floor of that shelf - 5th cell of that floor) | | | |

<Stock-keeper> Create Shelf use case specification

##### <Warehouse Accountant> Update Shelf

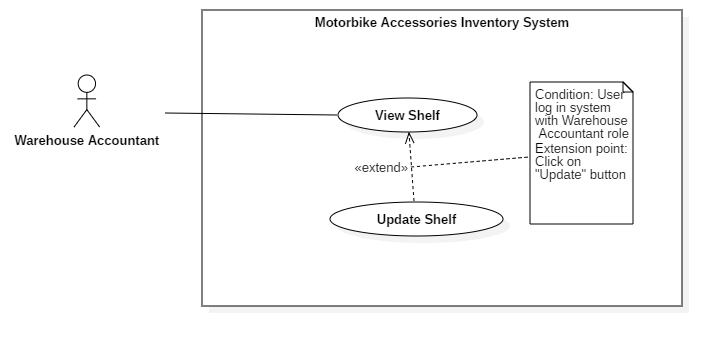


Figure 23 - <Warehouse Accountant> Update Shelf

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_S3** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_S3** | **Use Case Version** | 2.0 |
| **Use Case Name** | Update Shelf | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to update shelves’ information.   **Goal:**   * Warehouse Accountant updates the information of shelf successfully.   **Triggers:**   * Warehouse Accountant clicks on “Update” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: Selected shelf’s information will be successfully updated. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Update” button in “Shelves” screen. | “Update Shelf” screen is shown with following labels and fields which have current shelf information:   * Shelf ID: text, required * Description: text, optional * Number of Floor: integer, required * Number of Cell: integer, required * Cell Height (cm): float, required * Cell Width (cm): float, required * Cell Length (cm): float, required * Coordinate X: integer, required * Coordinate Y: integer, required | | 2 | Warehouse Accountant fills out data fields and clicks on “Confirm” button. | System validates inputted data and shows “Shelves” screen  [Exception 1, 2, 3, 4] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Editing Shelf Code already exists. | System show error message “Shelf Code already exists!” and highlight the error field. | | 2 | New editing Number of Floor/Number of Cell is lower than current ones. | System shows message “Number of Floor and Number of Cell can’t be decrease.” | | 3 | Warehouse Accountant clicks on “Cancel” button. | System shows “Shelves” screen. | | 4 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Manage Shelf use case.  **Business Rules:**   * In this system, only Warehouse Accountant can update shelves’ information. * “Number of Floors” field shows the number of floors that shelf has. This warehouse uses Selective Pallet Racking Systems which commonly has most maximum height is 5 floors because the forklift can’t bypass that height limit. * “Number of Cell” field shows the number of cells in each floor of that shelf. Number of Cell is a fixed number which is determined by the first created shelf. All shelves inside the warehouse must have the same number of cells per floor for easy maintenance, stocking in and stocking out process. * “Number of Floor” and “Number of Cell” are were prefilled by configurable value from “Standard Cell Size”. * “Cell Width”, “Cell Length” and “Cell Height” are the 3 dimension of Cell size to let stock-keeper know when Cell is full. * Coordinate X is position of that cell in horizontal, Coordinate is position of that cell in vertical * For the shelf that still has remained packages, only Shelf Code and Description, Coordinate X and Coordinate Y information can be changed. In order to be able to change any other attributes, all boxes belonging to that shelf must be removed. | | | |

<Warehouse Accountant> Update Shelf use case specification

##### <Warehouse Accountant> Deactivate Shelf

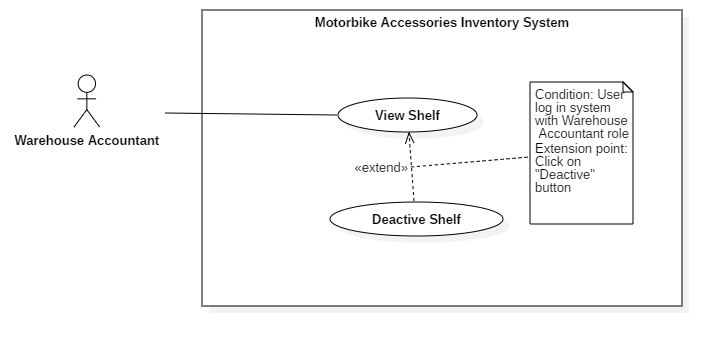


Figure 24 - <Warehouse Accountant> Deactivate Shelf

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_S4** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_S4** | **Use Case Version** | 2.0 |
| **Use Case Name** | Deactivate Shelf | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to delete shelves.   **Goal:**   * Warehouse Accountant deletes shelf successfully.   **Triggers:**   * Warehouse Accountant clicks on “Deactive” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: Selected shelf will be deleted from the current Shelf List. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Deactivate” button in “Shelves” screen. | Confirmation dialog is shown with the Y/N option: “You’re about to deactivate a shelf. Are you sure?” | | 2 | Warehouse Accountant clicks on “Confirm” button. | System validates whether the current shelf contains packages and shows “Shelves” screen.  [Exception 1, 2, 3] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Shelf currently contains packages. | System shows error message “The current shelf has one or more packages inside its cell. Please remove all packages and try again.” | | 2 | Warehouse Accountant clicks on “Cancel” button. | System shows “Shelves” screen. | | 3 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Extended from View Shelf use case.  **Business Rules:**   * In this system, only Warehouse Accountant can deactivate selected shelves. * All remained packages must be removed from the shelf before that shelf is being deactivated. * Deactivated shelf can be “Activate” again by click on same button name | | | |

<Warehouse Accountant> Deactivate Shelf use case specification

##### <Warehouse Accountant> Create Goods Receipt

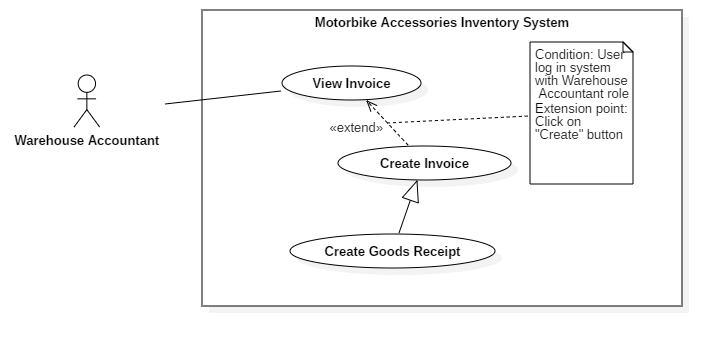


Figure 26 - <Warehouse Accountant> Create Goods Receipt

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_I2** | | | |
| **Use Case No.** | **RFIM\_UC\_WA\_I2** | **Use Case Version** | 1.0 |
| **Use Case Name** | Create Goods Receipt | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to create goods receipts.   **Goal:**   * Warehouse accountant creates goods receipt successfully.   **Triggers:**   * Warehouse accountant clicks on “Create” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: A new goods receipt will be created. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Create” button in “Goods Receipt” screen. | System shows “Create Receipt” screen with following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * Stock in Quantity: integer, required. | | 2 | Warehouse accountant clicks on “Choose Product” button in “Add Stock in Product” screen. | “Products List” screen is shown with following attribute of products: Product Id, Product Name, Quantity Per Box, Category, and Vendor. | | 3 | Warehouse Accountant check single or multiple checkboxes of products and clicks on “Confirm” button. | System shows “Create Receipt” screen with selected products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * Stock in Quantity: integer, required. | | 4 | Warehouse Accountant fills out “Stock in Quantity” field and clicks on “Finish” button. | System validates inputted data and shows “Goods Receipt” screen  [Exception 1, 2] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Cancel” button | System shows “Goods Receipt” screen. | | 2 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Create Invoice use case. Create Invoice use case was extended from View Invoice use case.  **Business Rules:**   * In this system, only Warehouse Accountant can create goods receipts. * “Stock in Quantity” is the number of boxes for stocking in products, maximum of 9999 boxes per Product per Good Receipt. * Newly created goods issue will have “Pending” status. * Goods receipts status will be changed to “Processing” during stock in process and “Done” when stock in process successfully finished. * Products of Good receipt does not contain deactivated products. * Goods Receipts Id was define automatically by this format: Configurable Prefix + Creation Date + “-” + Random 4 character code. Ex: RCE221019-A1x0. * Good Receipt Prefix can be change by using “Edit Receipt Prefix” feature | | | |

<Warehouse Accountant> Create Goods Receipt use case specification

##### <Warehouse Accountant> Create Goods Issue

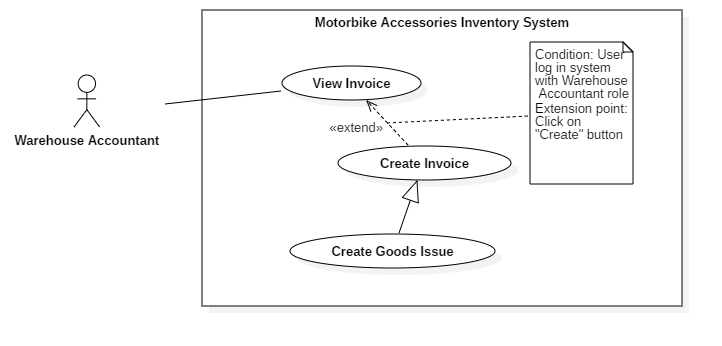


Figure 27 - <Warehouse Accountant> Create Goods Issue

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_WA\_I3** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_I3** | **Use Case Version** | 1.0 |
| **Use Case Name** | Create Goods Issue | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to create goods issue.   **Goal:**   * Warehouse accountant creates goods issue successfully.   **Triggers:**   * Warehouse accountant clicks on “Create” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: A new goods issue will be created. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Create” button in “Receipts/Issues” screen. | System shows “Create Issue” screen following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * On-hand Quantity: prefilled, not editable * Stock out Quantity: integer, required.. | | 2 | Warehouse accountant clicks on “Choose Product” button in “Add Stock out Product” screen. | “Products List” screen is shown with following attribute of products: Product ID, Product Name, Quantity Per Box, Category, Vendor and On-hand Quantity. | | 3 | Warehouse Accountant check single or multiple checkboxes of products and clicks on “Confirm” button. | System shows “Create Issue” screen with selected products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * On-hand Quantity: prefilled, not editable * Stock out Quantity: integer, required. | | 4 | Warehouse Accountant fills out “Stock out Quantity” field and clicks on “Finish” button. | System validates inputted data and shows “Receipts/Issues” screen.  [Exception 1, 2, 3] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Inputted “Stockout Quantity” is greater than “On-hand Quantity” | System shows error message: “Stockout Quantity must greater than 1 and smaller than On-hand Quantity: | | 2 | Warehouse accountant clicks on “Cancel” button | System shows “Receipts/Issues” screen. | | 3 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Create Invoice use case. Create Invoice use case was extended from View Invoice use case.  **Business Rules:**   * In this system, only Warehouse Accountant can create goods issue. * “On-hand Quantity” shows box quantity of that product in stock. * “Stock out Quantity” is the number of boxes for stocking in products, with the maximum value is equal to “On-hand Quantity”. * Newly created goods issue will have “Pending” status. * Goods issue status will be changed to “Done” when stock out process successfully finished. * Goods Issue Id was define automatically by this format: Configurable Prefix + Creation Date + “-” + Random 4 character code. Ex: ISE221019-A1x0. * Good Issue Prefix can be change by using “Edit Issue Prefix” feature | | | |

<Warehouse Accountant> Create Goods Issue use case specification

##### <Warehouse Accountant> Update Goods Receipt

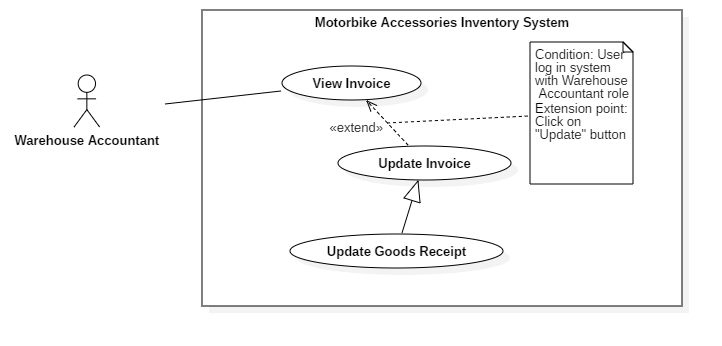


Figure 28 - <Warehouse Accountant> Update Goods Receipt

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_ WA\_I4** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_I4** | **Use Case Version** | 1.0 |
| **Use Case Name** | Update Goods Receipt | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to update goods receipts.   **Goal:**   * Warehouse accountant update goods receipt successfully.   **Triggers:**   * Warehouse accountant clicks on “Update” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role. * Goods Receipt’s status is “Pending”.   **Post conditions:**   * Selected good receipt information will be successfully updated. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Update” button in “Goods Receipt” screen. | System shows “Update Stock in Product” screen with products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * Stock in Quantity: integer, required. | | 2 | Warehouse accountant clicks on “Choose Product” button in “Update Stock in Product” screen. | “Products List” screen is shown with following attribute of products: Product ID, Product Name, Quantity Per Box, Category, and Vendor. | | 3 | Warehouse Accountant check single or multiple checkboxes and clicks on “Confirm” button. | System shows “Update Stock in Product” screen with new selected products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * Stock in Quantity: integer, required. | | 4 | Warehouse Accountant fills out “Stock in Quantity” field and clicks on “Finish” button. | System validates inputted data and shows “Goods Receipt” screen  [Exception 1, 2] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Cancel” button | System shows “Goods Receipt” screen. | | 2 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Update Invoice use case. Update Invoice use case was extended from View Invoice use case.  **Business Rules:**   * In this system, only Warehouse Accountant can update goods receipts. * “Stock in Quantity” is the number of boxes for stocking in products, maximum of 9999 boxes per Product per Good Receipt. * Goods receipts status will be changed to “Processing” during stock in process and “Done” when stock in process successfully finished. * Only Goods receipts with “Pending” status can be edited. Any further processes will cause the change of the status of the goods receipt and make the goods receipt not editable. * Products of Good receipt does not contain deactivated products. | | | |

<Warehouse Accountant> Update Goods Receipt use case specification

##### <Warehouse Accountant> Update Goods Issue

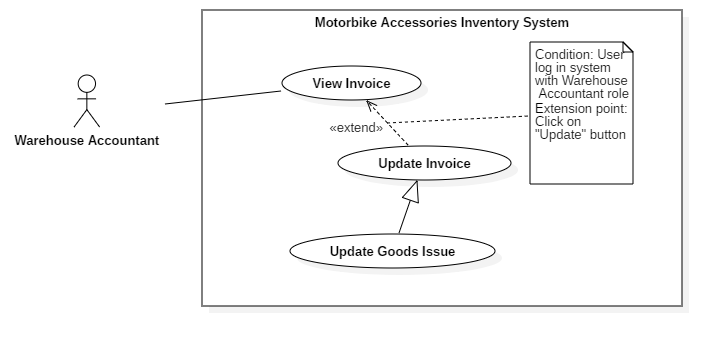


Figure 29 - <Warehouse Accountant> Update Goods Issue

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_ WA\_I5** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_I5** | **Use Case Version** | 1.0 |
| **Use Case Name** | Update Goods Issue | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to update goods issues   **Goal:**   * Warehouse accountant update goods issues successfully.   **Triggers:**   * Warehouse accountant clicks on “Update” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role. * Goods Issue’s status is “Pending”.   **Post conditions:**   * Selected good issue information will be successfully updated. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Update” button in “Goods Issues” screen. | System shows “Update Stock out Product” screen with products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * On-hand Quantity: prefilled, not editable * Stock in Quantity: integer, required. | | 2 | Warehouse accountant clicks on “Choose Product” button in “Update Stock out Product” screen. | “Products List” screen is shown with following attribute of products: Product ID, Product Name, Quantity Per Box, Category, Vendor and On-hand Quantity. | | 3 | Warehouse Accountant check single or multiple checkboxes of products and clicks on “Confirm” button. | System shows “Update Stock out Product” screen with new selected products and following labels, fields:   * Product ID: prefilled, not editable * Product Name: prefilled, not editable * Category: prefilled, not editable * Vendor: prefilled, not editable * On-hand Quantity: prefilled, not editable * Stock in Quantity: integer, required. | | 4 | Warehouse Accountant fills out “Stock out Quantity” field and clicks on “Finish” button. | System validates inputted data and shows “Goods Issues” screen  [Exception 1, 2] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Warehouse accountant clicks on “Cancel” button | System shows “Goods Issues” screen. | | 2 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Update Invoice use case. Update Invoice use case was extended from View Invoice use case.  **Business Rules:**   * In this system, only Warehouse Accountant can update goods issue. * “On-hand Quantity” shows box quantity of that product in stock. * “Stock out Quantity” is the number of boxes for stocking in products, with the maximum value is equal to “On-hand Quantity”. * Goods issue status will be changed to “Done” when stock out process successfully finished. * Only Goods issues with “Pending” status can be edited. Any further processes will cause the change of the status of the goods receipt and make the goods receipt not editable. | | | |

<Warehouse Accountant> Update Goods Issue use case specification

##### <Warehouse Accountant> Delete Invoice

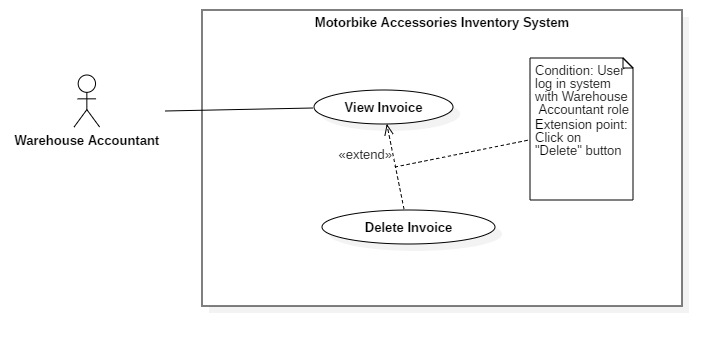


Figure 30 - <Warehouse Accountant> Delete Invoice

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_ WA\_I5** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_I5** | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete Invoice | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to delete invoices.   **Goal:**   * Warehouse accountant deletes invoice successfully.   **Triggers:**   * Warehouse accountant clicks on “Delete” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role. * Invoice’s status is “Pending”.   **Post conditions:**   * Success: Selected invoice will be deactivated * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Delete” button on “Goods Receipt” or “Goods Issue” screen. | Confirmation dialog is shown with the Y/N option: “You’re about to delete an invoice. Are you sure?” | | 2 | Warehouse Accountant clicks on “Confirm” button. | System validates invoice status then shows “Goods Receipt” or “Goods Issue” screen.  [Exception 1, 2, 3] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Invoice status is not “Pending”. | System shows error message: “Only invoice with ‘Pending’ status can be deleted.” | | 2 | Warehouse Accountant clicks on “Cancel” button. | System shows “Receipts/Issues” screen. | | 3 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Extended from View Invoice use case.  **Business Rules:**   * In this system, only Warehouse Accountant can delete invoices. * Only Goods invoice with “Pending” status can be deleted. * After deleted an invoice, user will be return to “Goods Receipt” or “Goods Issue” screen base on invoice type which user have just deleted | | | |

<Warehouse Accountant> Delete Invoice use case specification

##### <Warehouse Accountant> Create Report

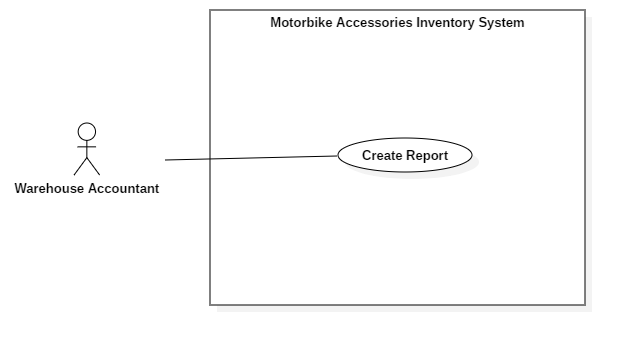


Figure 31 - <Warehouse Accountant> Create Report

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_ WA\_I6** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_I6** | **Use Case Version** | 2.0 |
| **Use Case Name** | Create Report | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows warehouse accountant to create a report file with pdf extension.   **Goal:**   * Warehouse accountant create report successfully.   **Triggers:**   * Warehouse accountant clicks on “Reports” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role.   **Post conditions:**   * Success: A report will be created. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on a report type button in “Reports” screen. | A specific report will be shown based on the report type that is chosen by Warehouse accountant.  [Exception 1] |   **Alternative Scenarios:** N/A  **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Warehouse Accountant actor.  **Business Rules:**   * In this system, only Warehouse Accountant can create reports. * There are 6 types of report can be chosen. * Goods Receipt, Goods Issue report helps Stock-keeper know what invoice they should progress. * Vendor, Category report helps Warehouse Accountant fill the excel file for “Import” feature. * Product report shows list of Products and Cells containing that Product. | | | |

<Warehouse Accountant> Create Report

##### <Warehouse Accountant> Update Stocktake Status

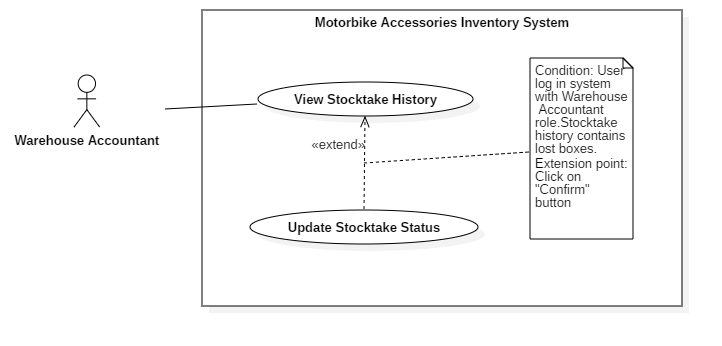


Figure 33 - <Warehouse Accountant> Update Stocktake Status

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_ WA\_H2** | | | |
| **Use Case No.** | **RFIM\_UC\_ WA\_H2** | **Use Case Version** | 2.0 |
| **Use Case Name** | Update Stocktake Status | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Warehouse Accountant   **Summary:**   * This use case allows Warehouse Accountant to update status of stoktake and status of lost and found boxes after stocktaking.   **Goal:**   * Warehouse Accountant update stocktake status and lost/found boxes status successfully.   **Triggers:**   * Warehouse Accountant clicks on “Confirm” button.   **Preconditions:**   * User must log in the system with Warehouse Accountant role. * Stocktake History status is “Pending”   **Post conditions:**   * Success: Stockatake status update successfully. Boxes status update successfully * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Confirm” button in “Awaiting Confirm” screen. | Confirmation dialog is shown with the Y/N option: “Are you sure you want to confirm status of this stocktake? This will cause boxes which not found during this stocktake become unacessible unless they was found again by next stocktake.  +Found box: +Lost box:” | | 2 | Warehouse Accountant clicks on “Confirm” button. | System shows “Awaiting Confirm” screen.  [Exception 1, 2] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Warehouse Accountant clicks on “Cancel” button. | System shows “Awaiting Confirm” screen. | | 2 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Extended from View Shelf use case.  **Business Rules:**   * In this system, only Warehouse Accountant can update stocktake status. * Boxes which exist in system but doesn’t exist during stocktaking will be set status to “Missing” after Update Stocktake Info. * Missing boxes can regain “Found” status with another Update Stocktake Status. * Only stocktake with “Pending” status can be confirm. * All confirmed Stocktake will be show in “History” screen | | | |

<Warehouse Accountant> Update Stocktake Status use case specification

#### <Stock-keeper> Overview Use Case

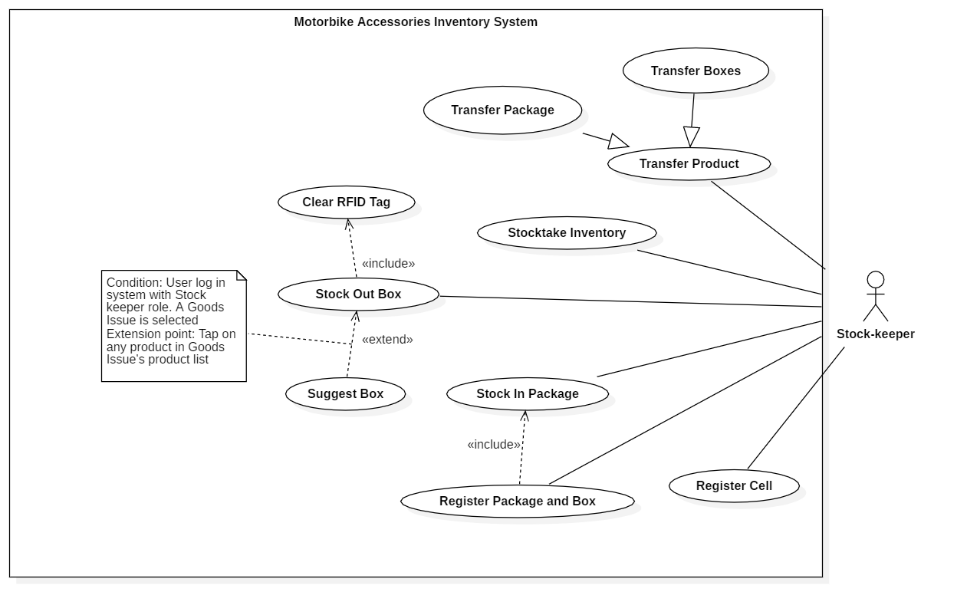


Figure 34 - <Stock-keeper> Overview Use Case

##### <Stock-keeper> Register Cell

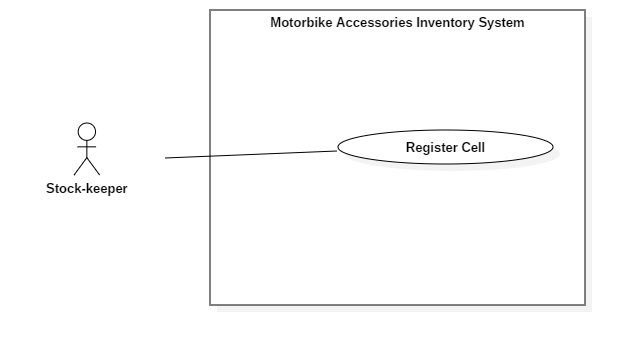


Figure 35 - <Stock-keeper> Register Cell

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK1** | | | |
| **Use Case No.** | **RFIM\_UC\_SK1** | **Use Case Version** | 2.0 |
| **Use Case Name** | Register Cell | | |
| **Author** | HoangNH | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper register cells with RFID tags.   **Goal:**   * Stock-keeper registers a cell with RFID tag successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the mobile system with Stock-keeper role. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Cell is linked RFID tag. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Register Cell” feature. | “Register Cell” screen is shown with following labels and fields:   * Shelf: select option, required * Floor: select option, required * Cell: select option, required * Cell RFID: text, automatically inputted when an RFID tag is scanned. | | 2 | Stock-keeper scans RFID tag and taps on “Save” button. | System validates inputted data and shows message “Register successfully.”  [Exception 1, 2, 3, 4, 5] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Cell RFID tags then taps on “Save” button | System shows error message "Please scan a Cell’s RFID.” | | 2 | Stock-keeper scans an RFID tags that is already registered with Cell ID | System shows error message "This RFID tag is already registered by another cell.” | | 3 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 4 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 5 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this mobile system, only Stock-keeper can register cells. * A registered cell can be used to bind packages. * Each Cell RFID tag can be changed by running “Register Cell” feature once again, but one RFID tag can’t be used to register 2 cells. | | | |

<Stock-keeper> Register Cell use case specification

##### <Stock-keeper> Register Package and Box

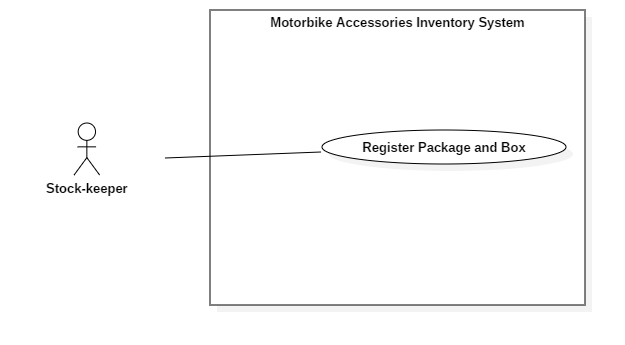


Figure 36 - <Stock-keeper> Register Package and Box

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK2** | | | |
| **Use Case No.** | **RFIM\_UC\_SK1** | **Use Case Version** | 2.0 |
| **Use Case Name** | Register Package and Box | | |
| **Author** | HoangPM | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to register packages and boxes.   **Goal:**   * Stock-keeper register package and box successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the mobile system with Stock-keeper role. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Boxes are successfully registered with a product and a RFID then mapped with a package RFID. * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Register Package and Box” button. | “Package Register” screen is shown with following labels and fields:   * Goods Receipt: select option, list Goods Receipt * Product Name: select option, list Product * Product Id: automatically show after selecting Product Name * Package RFID: automatically inputted when an RFID tag is scanned. One tag allowed only * Boxes RFID: text, automatically inputted when an RFID tag is scanned. Multiple tags are allowed. | | 2 | * Stock-keeper taps on “Scan” button and uses RFID scanner to scan Package tag. * Stock-keeper taps on “Scan” button and uses RFID scanner to scan Box tag. | * Package RFIDs are automatically inputted in the field. * List of Box RFIDs are automatically inputted in the field. | | 3 | Stock-keeper taps on “Save” button. | System validates inputted data and shows message “Register successfully.”  [Exception 1, 2, 3, 4, 5, 6] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Package RFID tags then taps on “Save” button. | System shows error message "Please scan a Package’s RFID.” | | 2 | Stock-keeper doesn’t scan any Box RFID tags then taps on “Save” button. | System shows error message “Please scan Box’s RFID.” | | 3 | Stock-keeper scan more boxes than required quantity | System shows error message “Enough Product.” | | 4 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 5 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 6 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this system, only Stock-keeper can register packages. * Stock-keeper can view List Product in Goods Receipt by pressing “Info” icon * “Registered Quantity” is number of boxes of that product which are registered successfully. * “Quantity” is number of boxes that the Good Receipt requires. * When any package is registered successfully, Good Receipt will have “Processing” status. If all packages in Goods Receipt are registered successfully then it will have “Done” status. * A package can contain many boxes but only have one type of product. | | | |

<Stock-keeper> Register Package and Box use case specification

##### <Stock-keeper> Stock in Package



Figure 37 - <Stock-keeper> Stock in Package

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK3** | | | |
| **Use Case No.** | **RFIM\_UC\_SK3** | **Use Case Version** | 2.0 |
| **Use Case Name** | Stock in Package | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to stock in packages.   **Goal:**   * Stock-keeper stocks in package successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the mobile system with Stock-keeper role. * Products are registered with RFID tag. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Package is mapped with a Cell RFID tag * Fail: System shows error messages.   **Main Success Scenarios:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Stock In” feature. | “Stock In” screen is shown with following labels and fields:   * Cell RFID: text, automatically inputted when an RFID tag is scanned. * Cell: text, automatically show after Cell RFID tag is scanned * Floor: text, automatically show after Cell RFID tag is scanned * Shelf: text, automatically show after Cell RFID tag scanned * Package RFID: text, automatically inputted when an RFID tag is scanned. | | 2 | * Stock-keeper taps on “Scan” button and uses RFID scanner to scan Cell RFID tag. * Stock-keeper taps on “Scan” button and uses RFID scanner to scan Package RFID tag. | * Cell RFIDs are automatically inputted in the field. * Package RFIDs are automatically inputted in the field. | | 3 | Stock-keeper taps on “Save” button. | System validates inputted data and shows message “Stock in successfully.”  [Exception 1, 2, 3, 4, 5, 6] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Cell RFID tags then taps on “Save” button. | System shows error message "Please scan Cell’s RFID.” | | 2 | Stock-keeper doesn’t scan any Package RFID tags then taps on “Save” button. | System shows error message "Please scan Package’s RFID.” | | 3 | Stock-keeper register package into a full cell. | System shows error message "Cell is full. Please register to another cell.” | | 4 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 5 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 6 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this mobile system, only Stock-keeper can stock in packages. * Unregistered packages cannot be stocked in. * Each cell only has one package. * Any further task such as merging packages should be accessed via “Transfer Product” feature. * Each package can contain many boxes but only have one type of product. | | | |

< Stock-keeper > Stock in Package use case specification

##### <Stock-keeper> Stock out Box

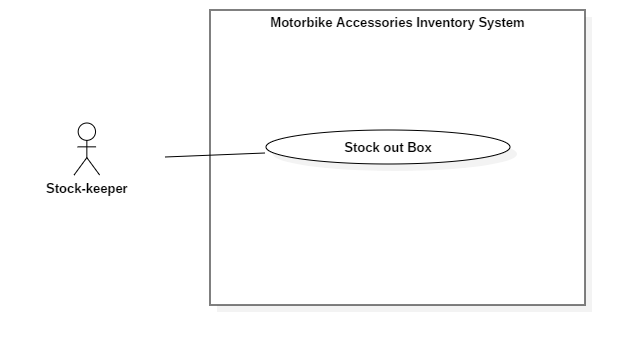


Figure 38 - <Stock-keeper> Stock out Box

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK4** | | | |
| **Use Case No.** | **RFIM\_UC\_SK4** | **Use Case Version** | 2.0 |
| **Use Case Name** | Stock out Box | | |
| **Author** | HoangPM | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to stock out boxes.   **Goal:**   * Stock-keeper stocks out box successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the system with Stock-keeper role. * Product is stocked in. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Product is stocked out. * Fail: System shows error messages.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Stock Out” feature. | “Stock Out” screen is shown with following labels and fields:   * Goods Issue: select option, list Goods Issue * Box Scanned Information: text, automatically inputted when an RFID tag is scanned | | 2 | Stock-keeper taps on “Scan” button and uses RFID scanner to scan boxes tag. | Scanner recognizes RFID and fills it to the “Box Scanned Information” field. | | 3 | Stock-keeper taps on “Save” button. | System shows message “Stock out successfully!”  [Exception 1, 2, 3, 4] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Box RFID tags then taps on “Save” button. | System shows error message "Please scan Box’s RFID.” | | 2 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 3 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 4 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this system only Stock-keeper can stock out boxes. * Unregistered boxes cannot be stocked out. * Box Scanned Information contains list of variety product types and its quantity, which are filled by scanning Box RFID tags. * When all products in Goods Issue are stocked out successfully, Goods Issue will have “Done” status. * Stock-keeper can scan any box of variety products as long as the invoice contains that product. | | | |

<Stock-keeper> Stock out Box use case specification

##### <Stock-keeper> Stocktake Inventory

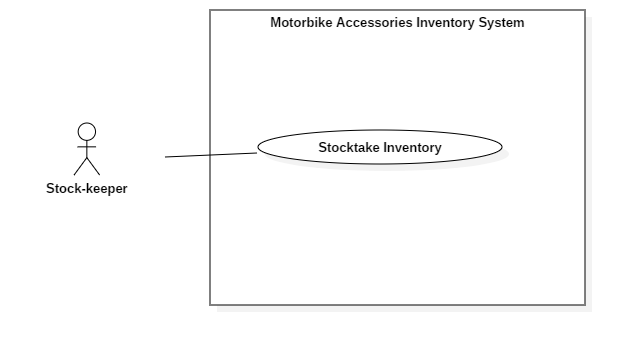


Figure 39 - <Stock-keeper> Stocktake Inventory

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK5** | | | |
| **Use Case No.** | **RFIM\_UC\_SK5** | **Use Case Version** | 2.0 |
| **Use Case Name** | Stocktake Inventory | | |
| **Author** | HoangPM | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to stocktake inventory products.   **Goal:**   * Stock-keeper stocktake inventory product successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the system with Stock-keeper role. * Product currently has boxes * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Product is stocked out. * Fail: System shows error messages.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Stocktake Inventory” feature. | “Stocktake Inventory” screen is shown with following labels and fields:   * Product Name: select option, list Product * Product ID: automatically show after selecting Product Name * Available Quantity: automatically show after selecting Product Name * Stocktake Quantity: integer, automatically increase when product’s RFID tag is scanned. | | 2 | Stock-keeper taps on “Scan Box RFID” button and uses RFID scanner to scan boxes of that products | Scanner recognizes RFID of selected product and increases “Stocktake Quantity” | | 3 | Stock-keeper taps on “Report” button. | Confirmation dialog is shown with the Y/N option: “Do you want to report?” | | 4 | Stock-keeper taps on “OK” button. | System shows message “Stocktake successfully!”  [Exception 1, 2, 3] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 2 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 3 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this system only Stock-keeper can stocktake inventory products. * “Available Quantity” is number of box which is archived by RFIM system. * “Stocktake Quantity” is number of box actually have in warehouse. * Stocktake Inventory can only stocktake one product at a time. | | | |

<Stock-keeper> Stocktake Inventory use case specification

##### <Stock-keeper> Suggest Box

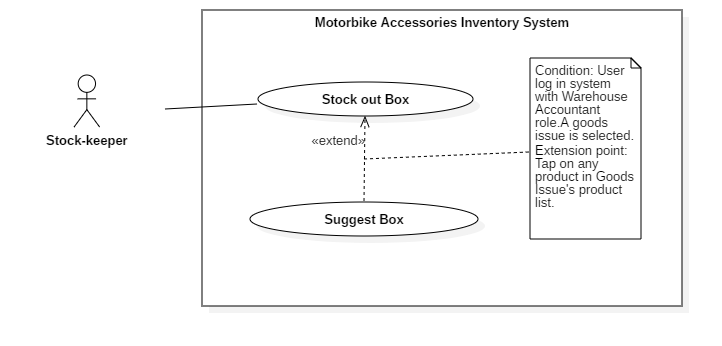


Figure 40 - <Stock-keeper> Suggest Box

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK6** | | | |
| **Use Case No.** | **RFIM\_UC\_SK6** | **Use Case Version** | 2.0 |
| **Use Case Name** | Suggest Box | | |
| **Author** | ThinhNT | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case show stock-keeper longest-lasting boxes and where to find them   **Triggers:**   * Stock-keeper tap on any product in “Goods Issue Info” screen   **Preconditions:**   * User must log in the mobile system with role Stock-keeper.   **Post conditions:**   * Success: List longest-lasting boxes was suggested * Fail: “Suggest Box” screen show nothing   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Goods Issue Info” button on “Good Issue” screen | “Goods Issue Info” screen is shown with list of products. | | 2 | Stock-keeper taps on any product on “Good Issue” screen | “Suggest Box” screen is shown with list of shelves contains that product. | | 3 | Stock-keeper taps on any shelf on “Suggest Box” screen | System show “List of Cell” contains boxes that satisfied both distance and lasting time |   **Alternative Scenarios: N/A**  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Network problem. | System shows error message: “Please check your connection.” | | 2 | Bluetooth problem. | System shows error message: “Please check your Bluetooth connection.” |     **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * If there is no box of that product, “Suggest Box” feature will leave an empty field. * “Suggest Box” feature list best options to stock out, making warehouse management much more effective | | | |

<Stock-keeper> Suggest Box use case specification

##### <Stock-keeper> Transfer Box

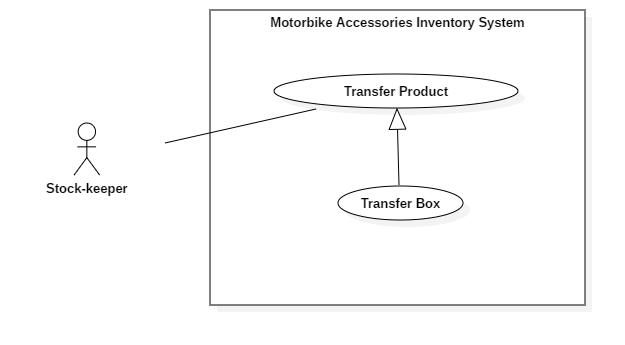


Figure 41 - <Stock-keeper> Transfer Box

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK7** | | | |
| **Use Case No.** | **RFIM\_UC\_SK7** | **Use Case Version** | 2.0 |
| **Use Case Name** | Transfer Box | | |
| **Author** | HoangNH | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to transfer product boxes to another package.   **Goal:**   * Stock-keeper transfer box of product successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the system with Stock-keeper role. * Product is stocked in. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Boxes of product is transferred to another package successfully. * Fail: System shows error messages.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Transfer Product” feature and selects “Transfer Box” tab. | “Transfer Box” screen is shown with following labels and fields:   * Package RFID: text, automatically inputted when package RFID tag is scanned. * Box RFID: list Boxes, automatically inputted when package RFID tag is scanned. | | 2 | Stock-keeper taps on “Scan” button and next to Package text field and uses RFID scanner to scan package’s tag. | Package RIFD is automatically inputted in the field. | | 3 | Stock-keeper taps on “Scan Box RFID” button and uses RFID scanner to scan boxes tag, which have same product as the destination package. | Boxes RFIDs are automatically inputted in the field. | | 4 | Stock-keeper taps on “Save” button. | System shows message “Transfer successfully!”  [Exception 1, 2, 3, 4, 5] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Package RFID tag then taps on “Save” button. | System shows error message "Please scan Package RFID.” | | 2 | Stock-keeper doesn’t scan any Box RFID tags then taps on “Save” button. | System shows error message "Please scan Box RFID.” | | 3 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 4 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 5 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Specialized from Transfer Product use case.  **Business Rules:**   * In this mobile system, only Stock-keeper can transfer boxes to another packages. * Only Boxes which have identical product with package will be shown on “Box Information”. Others types of boxes will not be scanned | | | |

<Stock-keeper> Transfer Box use case specification

##### <Stock-keeper> Transfer Package

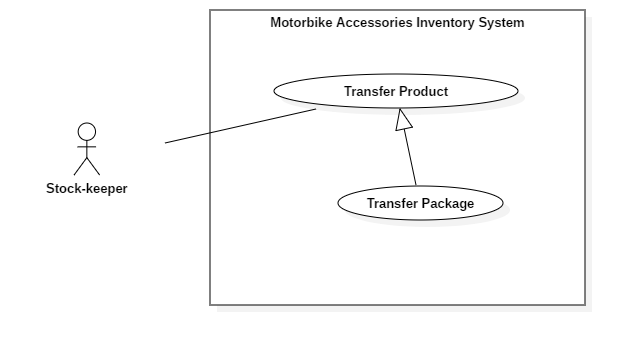


Figure 42 - <Stock-keeper> Transfer Package

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK8** | | | |
| **Use Case No.** | **RFIM\_UC\_SK8** | **Use Case Version** | 2.0 |
| **Use Case Name** | Transfer Package | | |
| **Author** | HoangNH | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows Stock-keeper to transfer product packages to another cell.   **Goal:**   * Stock-keeper transfer packages successfully.   **Triggers:**   * Stock-keeper taps on “Save” button.   **Preconditions:**   * User must log in the system with Stock-keeper role. * Product is stocked in. * The mobile device is connected with a RFID scanner.   **Post conditions:**   * Success: Product package is transferred to another cell successfully. * Fail: System shows error messages.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Transfer Product” feature and selects “Transfer Package” tab. | “Transfer Package” screen is shown with following labels and fields:   * Cell RFID: text, automatically inputted when an RFID tag is scanned. * Cell: text, automatically inputted after Cell RFID tag is scanned. * Floor: text, automatically inputted when an RFID tag is scanned. * Shelf: text, automatically inputted when an RFID tag is scanned. * Package: RFID of Package, text, automatically inputted when an RFID tag is scanned. | | 2 | Stock-keeper taps on “Scan” button next to Cell RFID text field and uses RFID scanner to scan cell’s tag. | Cell RIFD is automatically inputted in the field, followed by the position of the cell, floor, shelf associated with it. | | 3 | Stock-keeper taps on “Scan” button next to Package text field and uses RFID scanner to scan package’s tag. | Package RIFD is automatically inputted in the field | | 3 | Stock-keeper taps on “Save” button. | System shows message “Transfer successfully!”  [Exception 1, 2, 3, 4, 5, 6] |   **Alternative Scenarios:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | **No** | **Actor Action** | **System Response** | | 1 | Stock-keeper doesn’t scan any Cell RFID tags then taps on “Save” button. | System shows error message "Please scan Cell RFID tags.” | | 2 | Stock-keeper doesn’t scan any Package RFID tags then taps on “Save” button. | System shows error message "Please scan Package RFID tags.” | | 3 | Stock-keeper taps on “Cancel” button. | System shows “Home” screen. | | 4 | Stock-keeper scan Cell that already contains package | System shows error message: “Cell is full.” | | 5 | Bluetooth connection problem. | System shows error message: “Please check your Bluetooth connection.” | | 6 | Network connection problem. | System shows error message: “Please check your network connection.” |   **Relationships:** Associated with Stock-keeper actor.  **Business Rules:**   * In this mobile system, only Stock-keeper can transfer packages. * Unregistered packages cannot be transferred. | | | |

<Stock-keeper> Transfer Package use case specification

##### <Stock-keeper> Clear RFID Tag

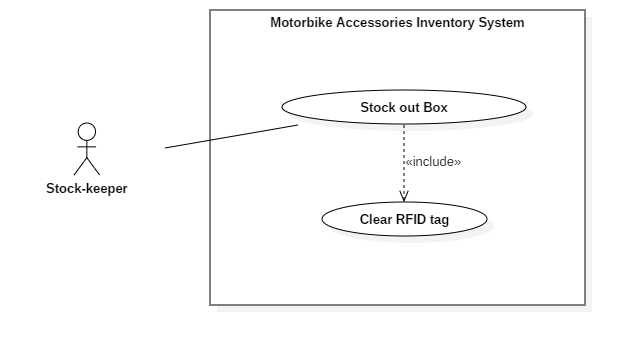


Figure 43 - <Stock-keeper> Clear RFID Tag

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – RFIM\_UC\_SK9** | | | |
| **Use Case No.** | **RFIM\_UC\_SK9** | **Use Case Version** | 2.0 |
| **Use Case Name** | Clear RFID Tag | | |
| **Author** | HoangPM | | |
| **Date** | 26/07/2019 | **Priority** | Normal |
| **Actor:**   * Stock-keeper   **Summary:**   * This use case allows system automatically clear RFID tag.   **Goal:**   * RFID tag successfully unbind from boxes and package for reusing purpose.   **Triggers:**   * Boxes: Stock out successfully. * Packages: Automatically cleared when contains no box.   **Preconditions:**   * User must log in the mobile system with role Stock-keeper.   **Post conditions:**   * Success: RFID Tag will be cleared successfully. * Fail: System shows error messages.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Stock-keeper taps on “Save” Button on Stock out screen | System automatically clear RFID tag of boxes |   **Alternative Scenarios: N/A**  **Exceptions: N/A**    **Relationships:** Included by Stock out Box usecase  **Business Rules:**   * Clear RFID Tag automatically trigger when stock out boxes successfully (Clear boxes RFIDs) or when package contains no box (Clear package RFID) | | | |

<Stock-keeper> Clear RFID Tag use case specification

**All Use Case Specification could be found here:** https://drive.google.com/open?id=1LQz30zdGvZ-penDFe\_-M2IEJCG8iBAKp

## Software System Attribute

### Usability

* The authorized users should need at least 2 days of training to be capable of using the system effectively and productively.
* The texts, labels must be all in English.
* The content of the description can be written in Vietnamese.

### Reliability

* Exception rate is at minimum 1% over all working time
* The RFID tag always contains and transfer correct information of the accessory it is attached to - to the server whenever the scanner detects the tag successfully to do the further works.

### Availability

Server available 24/7 including holidays with minimun downtime 1-2 days each 3 months for maintainance

### Security

* Privacy: Each role of the user has a specific permission to interact with the system. The system always checks the authorization and authentication before doing further work.

### Maintainability

* The system is divided into separated modules.
* The code is easy to maintain and upgrade.
* Require schedule maintainance every 6 month

### Portability

* Admin, customers and guests can use application on Google Chrome version 42 or above.
* Warehouse Staff can use mobile application on Android devices that support Android 6.0 or later.

### Performance

Requests from web application are responded in less than 5 seconds at 4 Mbps bandwidth speed and 1GHz processing speed of CPU.

## Conceptual Diagram

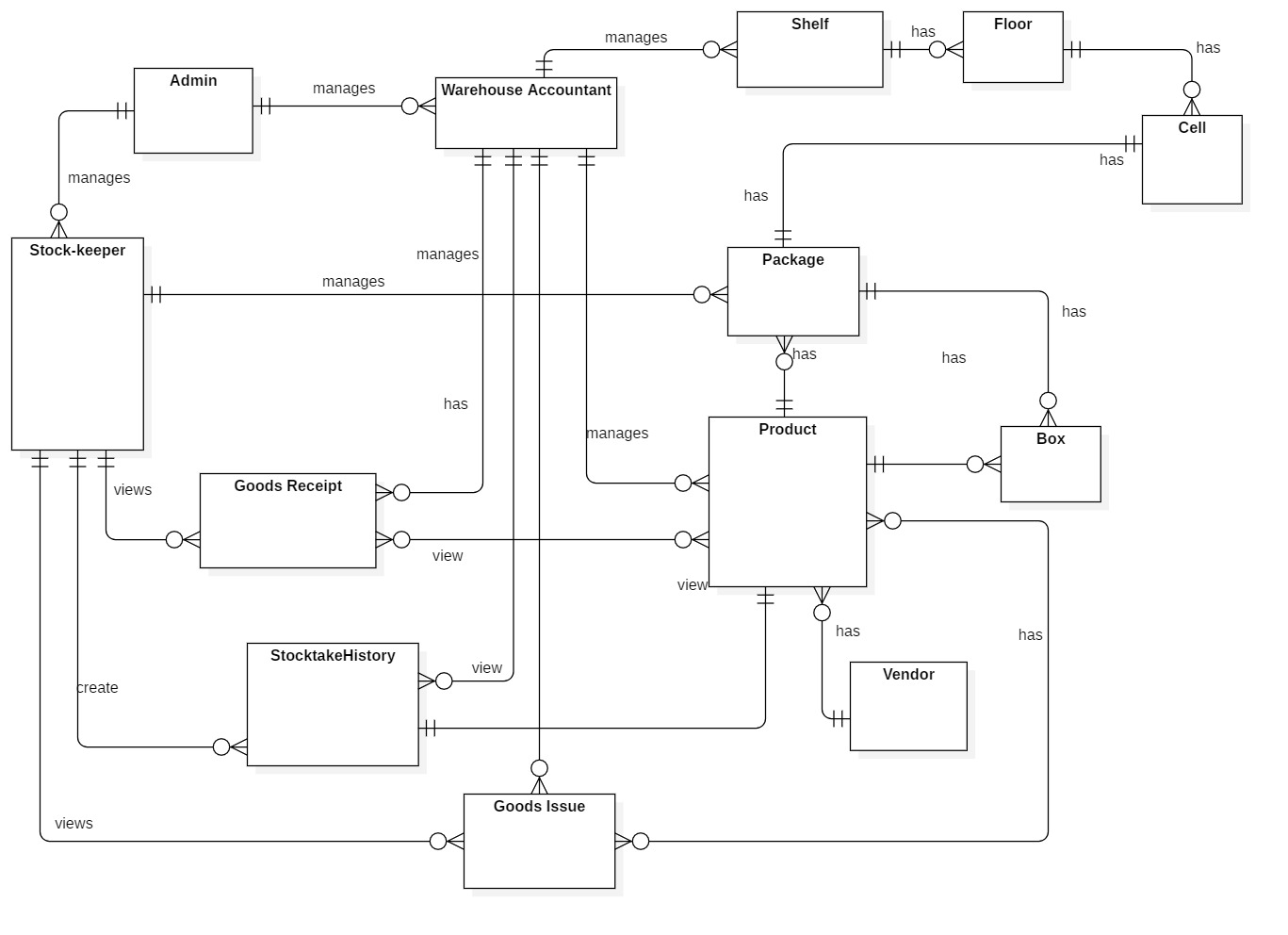


Figure 44 - Conceptual Diagram

|  |  |
| --- | --- |
| **ENTITY DATA DICTIONARY: DESCRIBE ALL CONTENT OF ALL ENTITIES** | |
| **Entity name** | **Description** |
| Admin | Contains the admin information |
| Stock-keeper | Contains the stock-keeper information |
| Warehouse Staff | Contains the warehouse staff information |
| Package | Contains the package information |
| Product | Contains the product information |
| Box | Contains the box information |
| Shelf | Contains the shelf information |
| Floor | Contains the floor information |
| Cell | Contains the cell information |
| Goods Receipt | Contains the goods receipt information |
| Goods Issue | Contains the goods issue information |
| Stocktake History | Contains the stocktake history information |
| Vendor | Contains the vendor information |

Table 14 - Conceptual Diagram Data Dictionary

# D. Software Design Description

## Design Overview

* This document describes the technical and user interface design of RFIM system. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.
* The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.
* The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.
* The database design describes the relationships between entities and details of each entity.
* Document overview:
  + Section 2: gives an overall description of the system architecture design.
  + Section 3: gives component diagram that describes the connection and integration of the system.
  + Section 4: gives the detail design description, which includes class diagram, class explanation and sequence diagram to detail the application functions.
  + Section 5: describes a fully attributed Entity Relationship Diagram.

## System Architectural Design

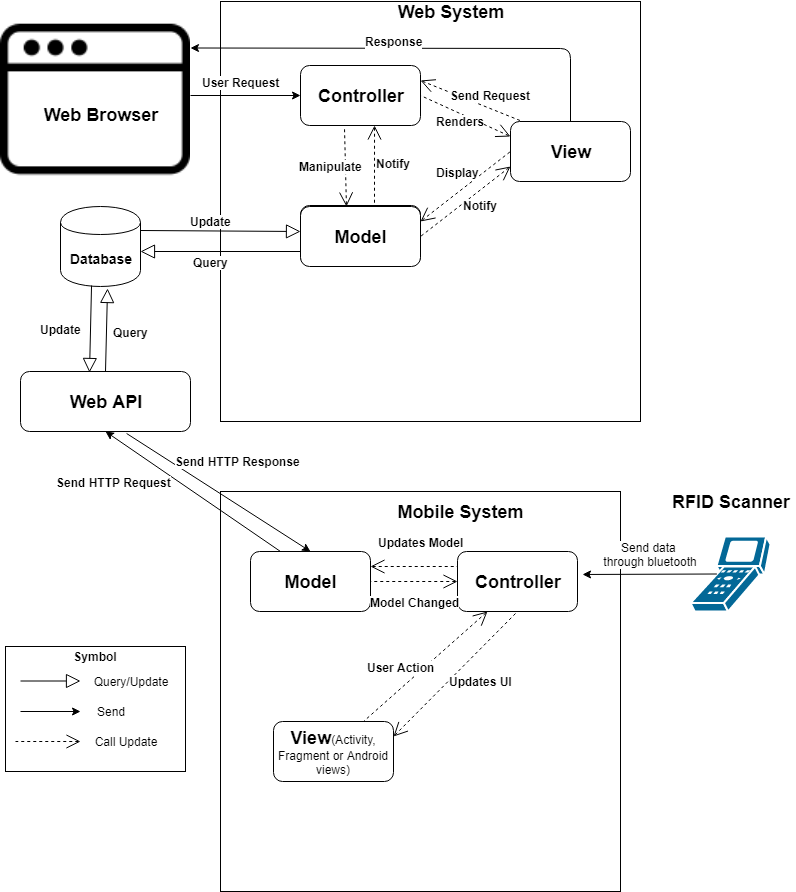


Figure 45 - System Architectural Design

### Web Application Architecture Description

In web application, our system is developed with ASP.NET Core MVC architecture. We decide to choose this architect for web application because of following advantages:

* ASP.NET Core is a cross-platform, high-performance, open-source framework for building application.
* Application is built-in dependency injection.
* ASP.NET Core has some technologies like entity framework to connect database.
* Support to develop and run on Windows, macOS, and Linux.
* Support to develop Web API and easily to build Web UI.
* ASP.NET Core is open-source and community-focused.

The MVC pattern (Model-View-Controller) separates application into 3 main components: Model-View-Controller. MVC patter helps you to build application more testable and easier. MVC-based app contains:

* Controller: is a part of application like a handler browser to handle and response user interaction and input. Controller handles route data and call repository’s method to get business data for handling bussines logic. Then select view to render to web browser for user.
* View: is a part of application for displaying the app’s user interface. It get data from model and specifies how that data should be presented. It updates UI when model changes.
* Model: is a part of application for represents the business data. The model classes use validation logic to enforce business data. The model notifies views when it changes and lets view query the model.

(Reference: [4])

### Mobile Application Architecture Description

We decided to choose Android and Arduino to implement our system because of the following benefits:

* MVC is a popular architecture which can help device code in to Model, View and Controller.
* We focus on the user who has to work in warehouse and has to move too much. Therefore we choose Android mobile device to develop our system.
* Android is the popular handheld operating system in the word so that user can install easily and use our system. In addition we can develop Android application easily on Window, Mac OS and Linux.
* Compare with React Native and other hybrid framework on mobile, Android is more stable and faster.
* Arduino framework is integrated a lot of modules like RFID scanner, Bluetooth, wifi, etc… which are affordable.
* Bluetooth technology helps transfer data between Android handheld and RFID scanner without using cable.

## Component Diagram

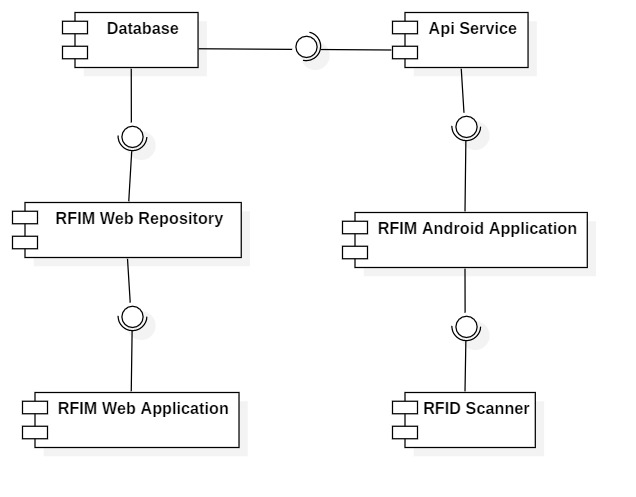


Figure 46 – Component Diagram

|  |  |
| --- | --- |
| COMPONENT DICTIONARY: DESCRIBES COMPONENTS | |
| Component name | **Description** |
| RFID Scanner | RFID scanner device |
| API Service | Provide API for Android application. Interact with Database |
| RFIM Android Application | Receive data from RFID scanner. Interact with API |
| RFIM Web Application | Handle business |
| RFIM Web Repository | Interact with Database |
| Database | Store data |

Table 15 - Component Dictionary

## Detail Description

### Class Diagram

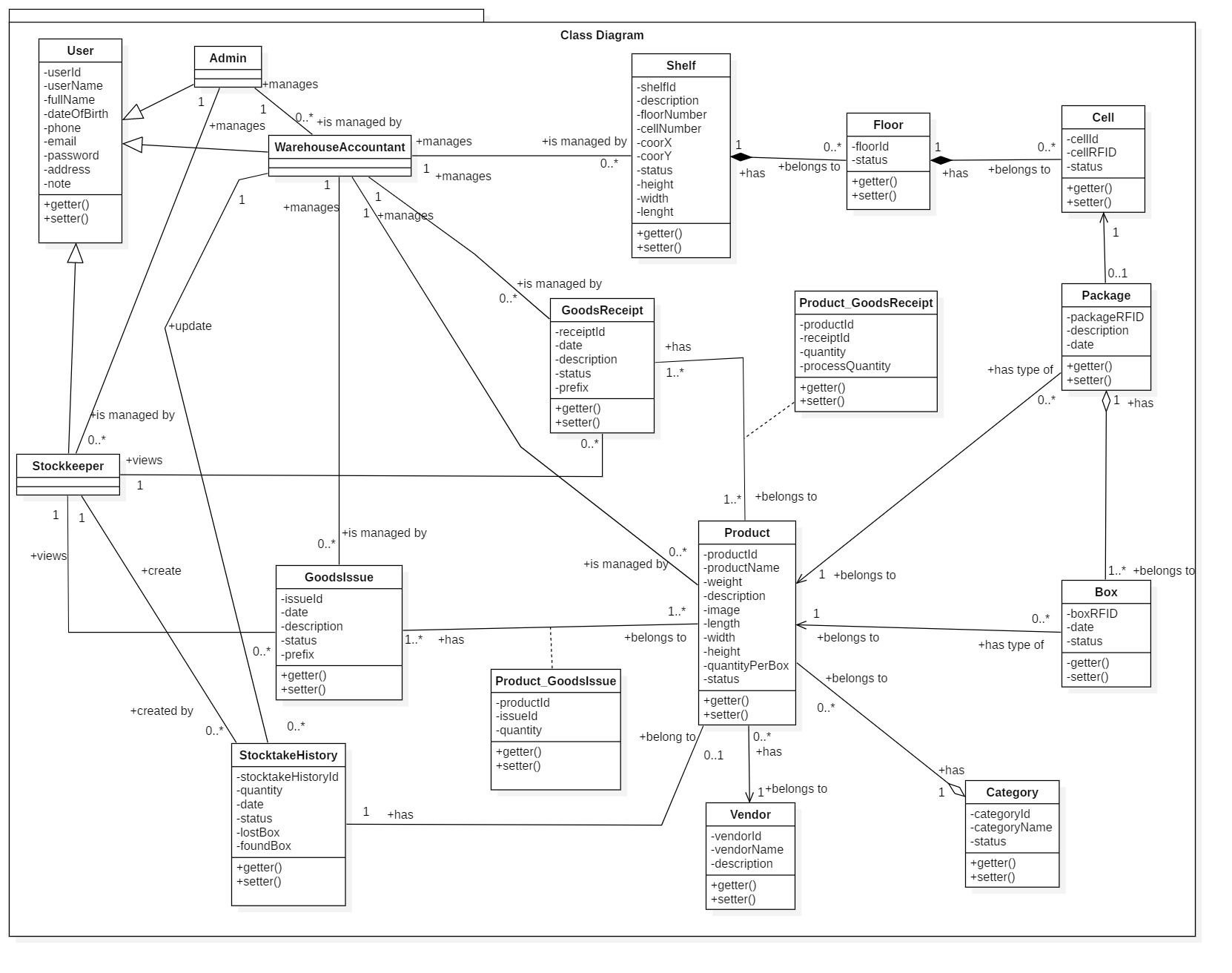


Figure 47 – Class Diagram

|  |  |  |
| --- | --- | --- |
| **CLASS DICTIONARY: DESCRIBE CLASS** | | |
| ***Class Name*** | ***Mapping column with Conceptual diagram*** | ***Description*** |
| **User** | N/A | Not exist in conceptual diagram, but needed in class diagram to contain the user information in general |
| **Admin** | Admin | Contains the admin information |
| **Stock-keeper** | Stock-keeper | Contains the stock-keeper information |
| **Warehouse Accountant** | Warehouse Accountant | Contains the warehouse accountant information |
| **Shelf** | Shelf | Contains the shelf information |
| **Floor** | Floor | Contains the floor information |
| **Cell** | Cell | Contains the cell information |
| **Package** | Package | Contains the package information |
| **Product** | Product | Contains the product information |
| **Category** | N/A | Not exist in conceptual diagram, but needed in class diagram to contain the category information |
| **Box** | Box | Contains the box information |
| **Vendor** | Vendor | Contains the vendor information |
| **Goods Receipt** | Goods Receipt | Contains the goods receipt information |
| **Goods Issue** | Goods Issue | Contains the goods issue information |
| **Product \_ Goods Receipt** | N/A | Not exist in conceptual diagram, but needed in class diagram to contain the attribute to define many to many relationship |
| **Product \_ Goods Issue** | N/A | Not exist in conceptual diagram, but needed in class diagram to contain the attribute to define many to many relationship |
| **Stocktake History** | StocktakeHistory | Contains the stocktake history information |

Table 16 - Class Dictionary

### Class Diagram Explanation

#### User

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| userId | string | private | Unique identifier of user |
| userName | string | private | Username of user |
| password | string | private | Password of user |
| fullName | string | private | Full name of user |
| dateOfBirth | date | private | Date of birth of user |
| phone | string | private | Phone number of user |
| email | string | private | Email of user |
| address | string | private | Address of user |
| note | string | private | Notes about user |

Table 17 - <Class Diagram> User Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get user attributes |
| setter() | void | public | Set value of user attributes |

Table 18 - <Class Diagram> User Methods

#### Admin

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| userId | string | private | Unique identifier of admin |
| userName | string | private | Username of admin |
| password | string | private | Password of admin |
| fullName | string | private | Full name of admin |
| dateOfBirth | date | private | Date of birth of admin |
| phone | string | private | Phone number of admin |
| email | string | private | Email of admin |
| address | string | private | Address of admin |
| note | string | private | Notes about admin |

Table 19 - <Class Diagram> Admin Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get admin attributes |
| setter() | void | public | Set value of admin attributes |

Table 20 - <Class Diagram> User Methods

#### Stock-keeper

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| userId | string | private | Unique identifier of stock-keeper |
| userName | string | private | Username of stock-keeper |
| password | string | private | Password of stock-keeper |
| fullName | string | private | Full name of stock-keeper |
| dateOfBirth | date | private | Date of birth of stock-keeper |
| phone | string | private | Phone number of stock-keeper |
| email | string | private | Email of stock-keeper |
| address | string | private | Address of stock-keeper |
| note | string | private | Notes about stock-keeper |

Table 21- <Class Diagram> Stock-keeper Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get stock-keeper attributes |
| setter() | void | public | Set value of stock-keeper attributes |

Table 22- <Class Diagram> Stock-keeper Methods

#### Warehouse Accountant

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| userId | string | private | Unique identifier of warehouse staff |
| userName | string | private | Username of warehouse staff |
| password | string | private | Password of warehouse staff |
| fullName | string | private | Full name of warehouse staff |
| dateOfBirth | date | private | Date of birth of warehouse staff |
| phone | string | private | Phone number of warehouse staff |
| email | string | private | Email of warehouse staff |
| address | string | private | Address of warehouse staff |
| note | string | private | Notes about warehouse staff |

Table 23- <Class Diagram> Warehouse Accountant Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get warehouse staff attributes |
| setter() | void | public | Set value of warehouse staff attributes |

Table 24- <Class Diagram> Stock-keeper Methods

#### Shelf

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| shelfId | string | private | Unique identifier of shelf |
| description | string | private | Description of shelf |
| floorNumber | integer | private | Number of floor of each shelf |
| cellNumber | integer | private | Number of cell per floor of each shelf |
| coorX | integer | private | Coordinate X of shelf, determine by position in horizontal |
| coorY | integer | private | Coordinate Y of shelf, determine by position in vertical |
| status | boolean | private | Status of shelf |
| height | float | private | Height of shelf |
| width | float | private | Width of shelf |
| lenght | float | private | Length of shelf |

Table 25- <Class Diagram> Shelf Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get shelf attributes |
| setter() | void | public | Set value of shelf attributes |

Table 26- <Class Diagram> Shelf Methods

#### Floor

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| floorId | string | private | Unique identifier of floor |
| status | boolean | private | Status of floor |

Table 27- <Class Diagram> Floor Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get floor attributes |
| setter() | void | public | Set value of floor attributes |

Table 28- <Class Diagram>Floor Methods

#### Cell

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| cellId | string | private | Unique identifier of cell |
| cellRFID | string | private | RFID bind to cell |
| status | boolean | private | Status of cell |

Table 29- <Class Diagram>Cell Attribute

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get cell attributes |
| setter() | void | public | Set value of cell attributes |

Table 30- <Class Diagram>Cell Methods

#### Package

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| packageRFID | string | private | Unique RFID of package |
| date | date | private | Package creation date |
| description | string | private | Description of package |

Table 31- <Class Diagram>Package Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get package attributes |
| setter() | void | public | Set value of package attributes |

Table 32- <Class Diagram>Package Methods

#### Product

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| productId | string | private | Unique identifier of product |
| productName | string | private | Name of product |
| weight | float | private | Weight of product |
| description | string | private | Description of product |
| image | string | private | Image of product |
| length | float | private | Product box length |
| width | float | private | Product box width |
| height | float | private | Product box height |
| quantityPerBox | integer | private | Number of item per product box |
| status | boolean | private | Product status |

Table 33- <Class Diagram> Product Attribute

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get product attributes |
| setter() | void | public | Set value of product attributes |

Table 34- <Class Diagram> Product Methods

#### Category

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| categoryId | int | private | Unique identifier of category |
| categoryName | string | private | Name of category |
| status | boolean | private | Status of category |

Table 35- <Class Diagram>Category Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get category attributes |
| setter() | void | public | Set value of category attributes |

Table 36- <Class Diagram> Category Methods

#### Box

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| boxRFID | string | private | Unique RFID of box |
| date | date | private | Box stock in date |
| status | boolean | private | Status of box |

Table 37- <Class Diagram>Box Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get box attributes |
| setter() | void | public | Set value of box attributes |

Table 38- <Class Diagram> Box Method

#### Vendor

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| vendorId | int | private | Unique identifier of vendor |
| vendorName | string | private | Name of vendor |
| description | string | private | Description of vendor |

Table 39- <Class Diagram> Vendor Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get vendor attributes |
| setter() | void | public | Set value of vendor attributes |

Table 40- <Class Diagram> Vendor Methods

#### Goods Receipt

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| receiptId | string | private | Unique identifier of goods receipt |
| date | dateTime | private | Goods receipt creation date |
| description | string | private | Description of goods receipt |
| status | int | private | Status of goods receipt |
| prefix | string | private | Prefix of goods receipt |

Table 41- <Class Diagram>Goods Receipt Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get goods receipt attributes |
| setter() | void | public | Set value of goods receipt attributes |

Table 42- <Class Diagram>Goods Receipt Methods

#### Goods Issue

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| issueId | string | private | Unique identifier of goods issue |
| date | dateTime | private | Date of creating goods issue |
| description | string | private | Description of goods issue |
| status | int | private | Status of goods issue |
| prefix | string | private | Prefix of goods issue |

Table 43- <Class Diagram> Goods Issue Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get goods issue attributes |
| setter() | void | public | Set value of goods issue attributes |

Table 44- <Class Diagram> Goods Issue Methods

#### Product\_Goods Receipt

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| receiptId | string | private | Unique identifier of goods receipt |
| productId | string | private | Unique identifier of product |
| quantity | integer | private | Planning quantity of stock in |
| processQuantity | integer | private | Current scanned quantity of stock in |

Table 45- <Class Diagram>Product\_Goods Receipt Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get product\_receipt attributes |
| setter() | void | public | Set value of product\_receipt attributes |

Table 46- <Class Diagram>Product\_Goods Receipt Methods

#### Product\_Goods Issue

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| issueId | string | private | Unique identifier of goods issue |
| productId | string | private | Unique identifier of issue |
| quantity | integer | private | Planning quantity of stock out |

Table 47- <Class Diagram>Product\_Goods Issue Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get product\_issue attributes |
| setter() | void | public | Set value of product\_issue attributes |

Table 48- <Class Diagram>Product\_Goods Issue Methods

#### Stocktake History

* **Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| stocktakeHistoryId | int | private | Unique identifier of stocktake history |
| quantity | string | private | Stocktaked quantity |
| date | date | private | Date of Stocktake History |
| status | boolean | private | Status of Stocktake |
| lostBox | string | private | Boxes that didn’t appear in stocktake process |
| foundBox | string | private | Lost boxes that was found during stocktake process |

Table 49- <Class Diagram>Stocktake History Attributes

* **Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter() | attribute type | public | Get stocktake attributes |
| setter() | void | public | Set value of stocktake attributes |

Table 50- <Class Diagram>Stocktake History Methods

### Sequence Diagram

Figure 48- <Sequence Diagram> Create Category

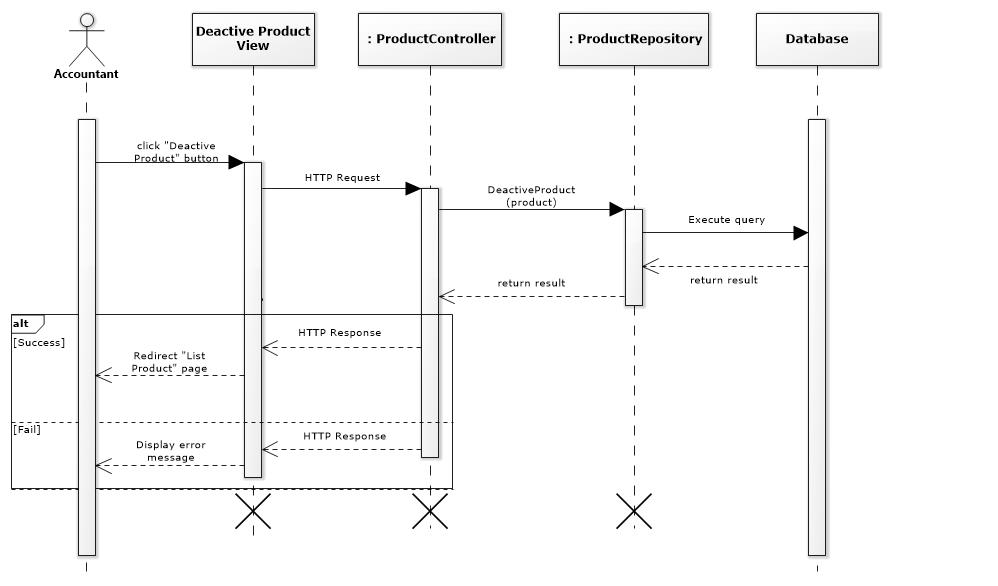


Figure 54- <Sequence Diagram> Deactivate Product

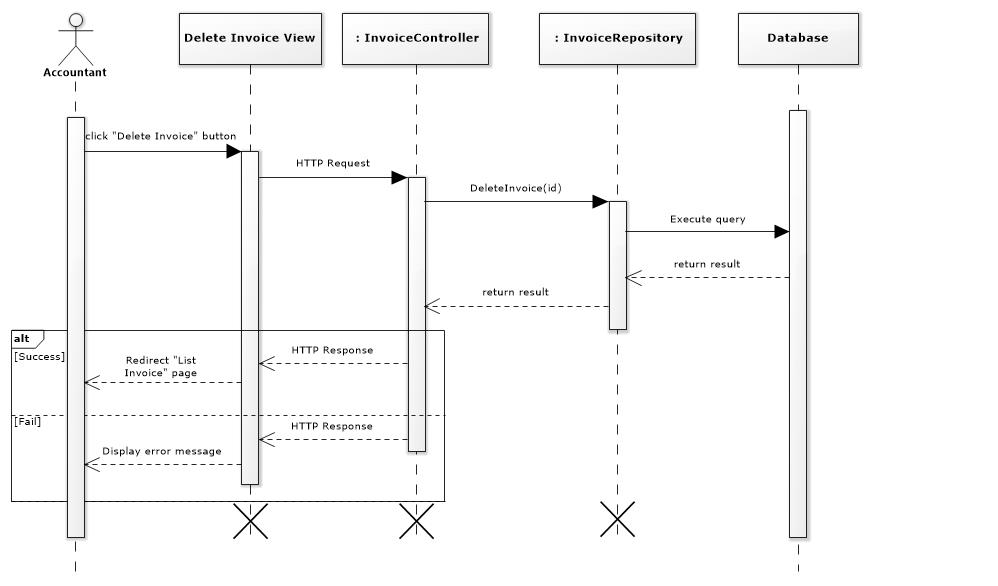


Figure 57- <Sequence Diagram> Delete Invoice

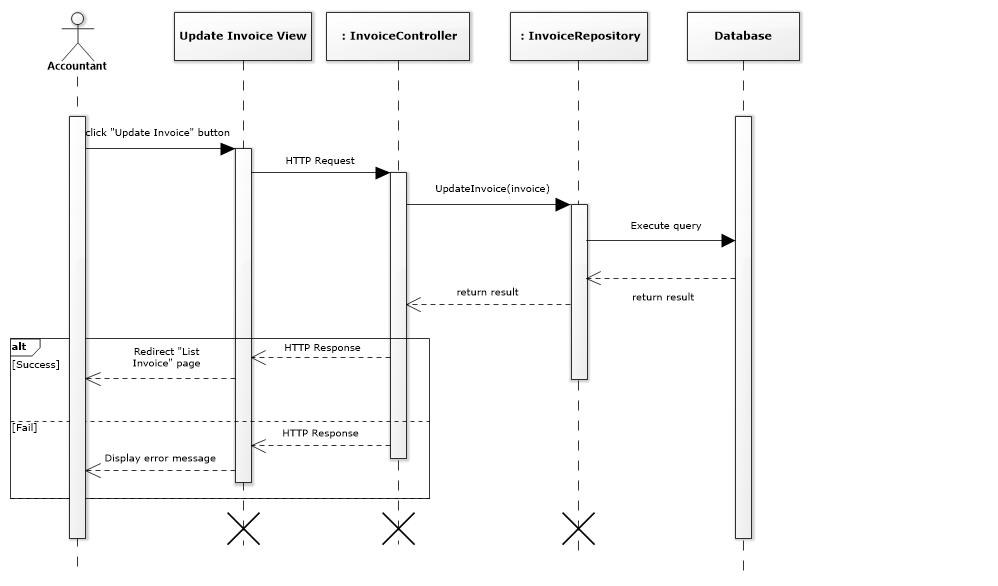


Figure 59- <Sequence Diagram> Update Invoice

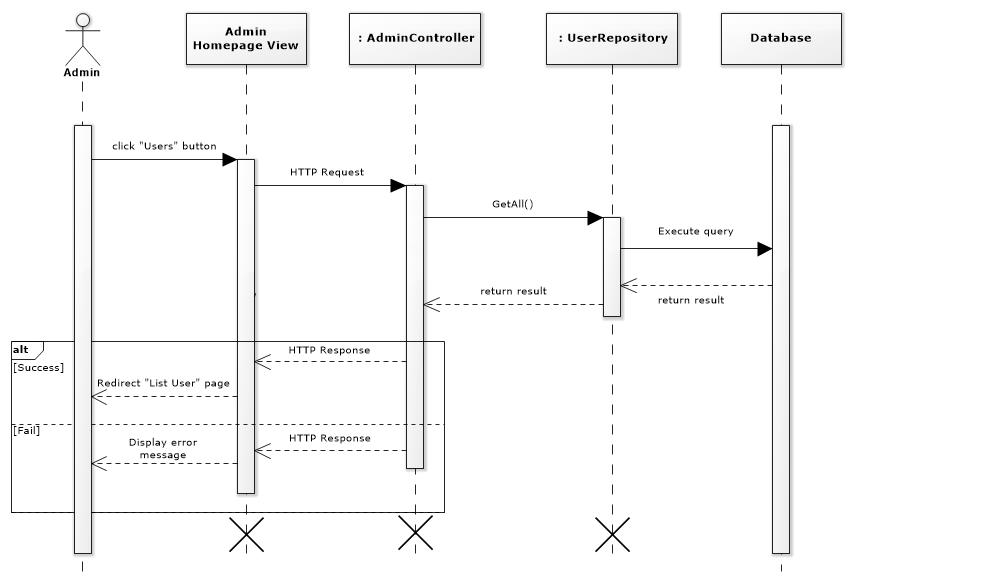


Figure 66- <Sequence Diagram> View User

**All Sequence Diagram could be found here:** https://drive.google.com/open?id=1tZ4yjZRzz0DGS093IR\_NqZk0rnEmpqky

### Activity Diagram

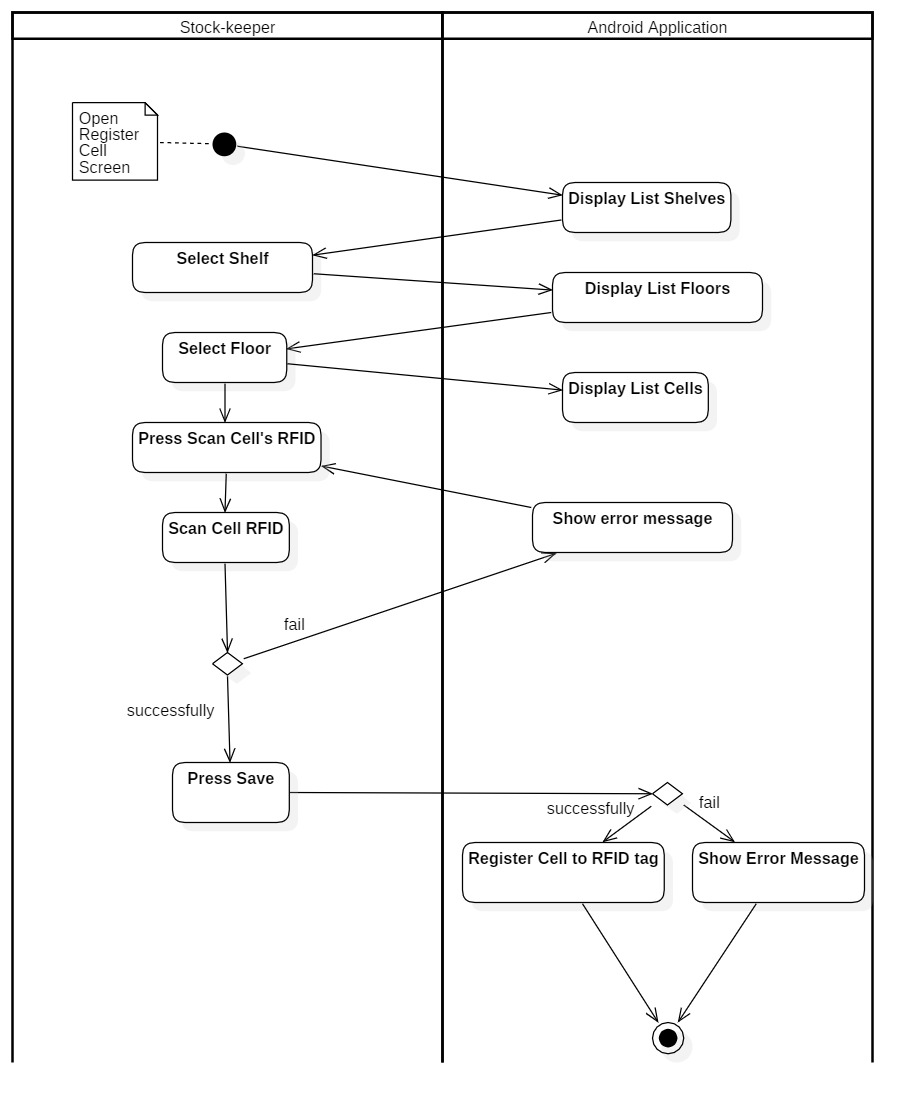


Figure 67- <Activity Diagram> Register Cell

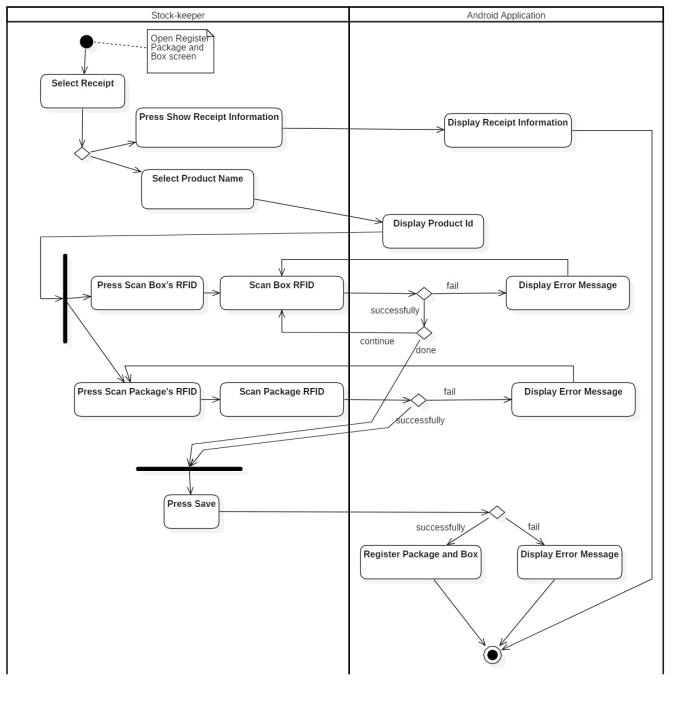


Figure 68- <Activity Diagram> Register Box and Package

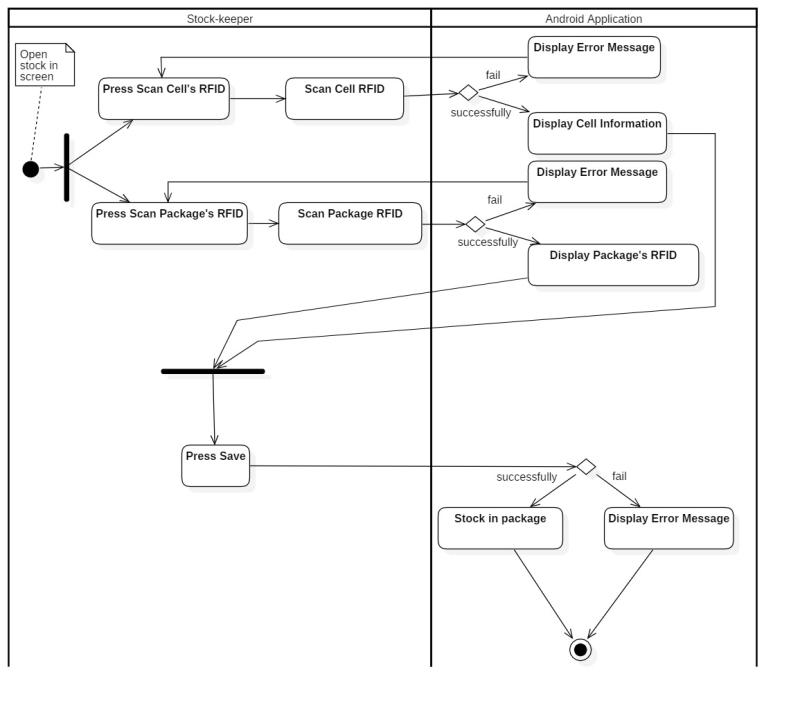


Figure 69- <Activity Diagram> Stock in

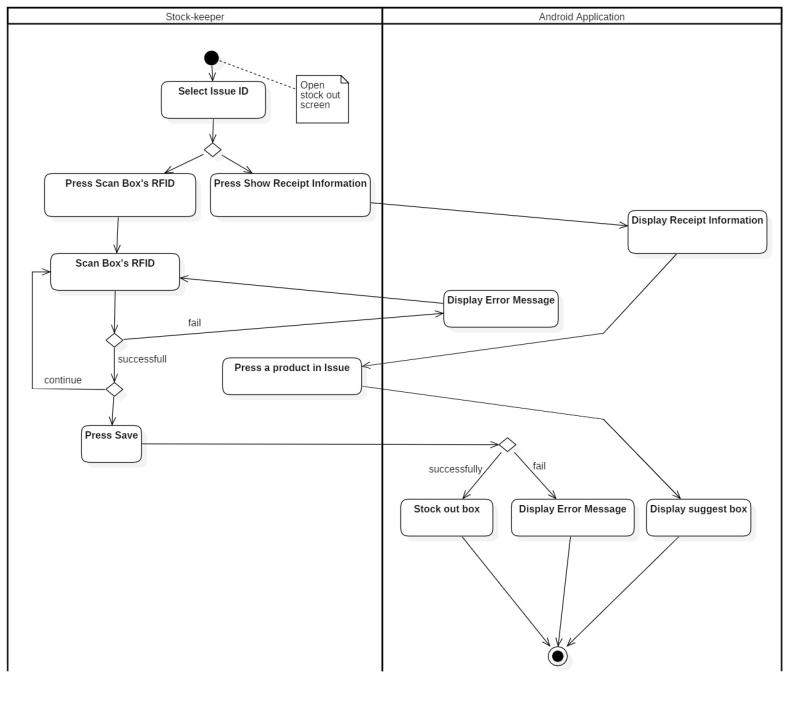


Figure 70- <Activity Diagram> Stock Out

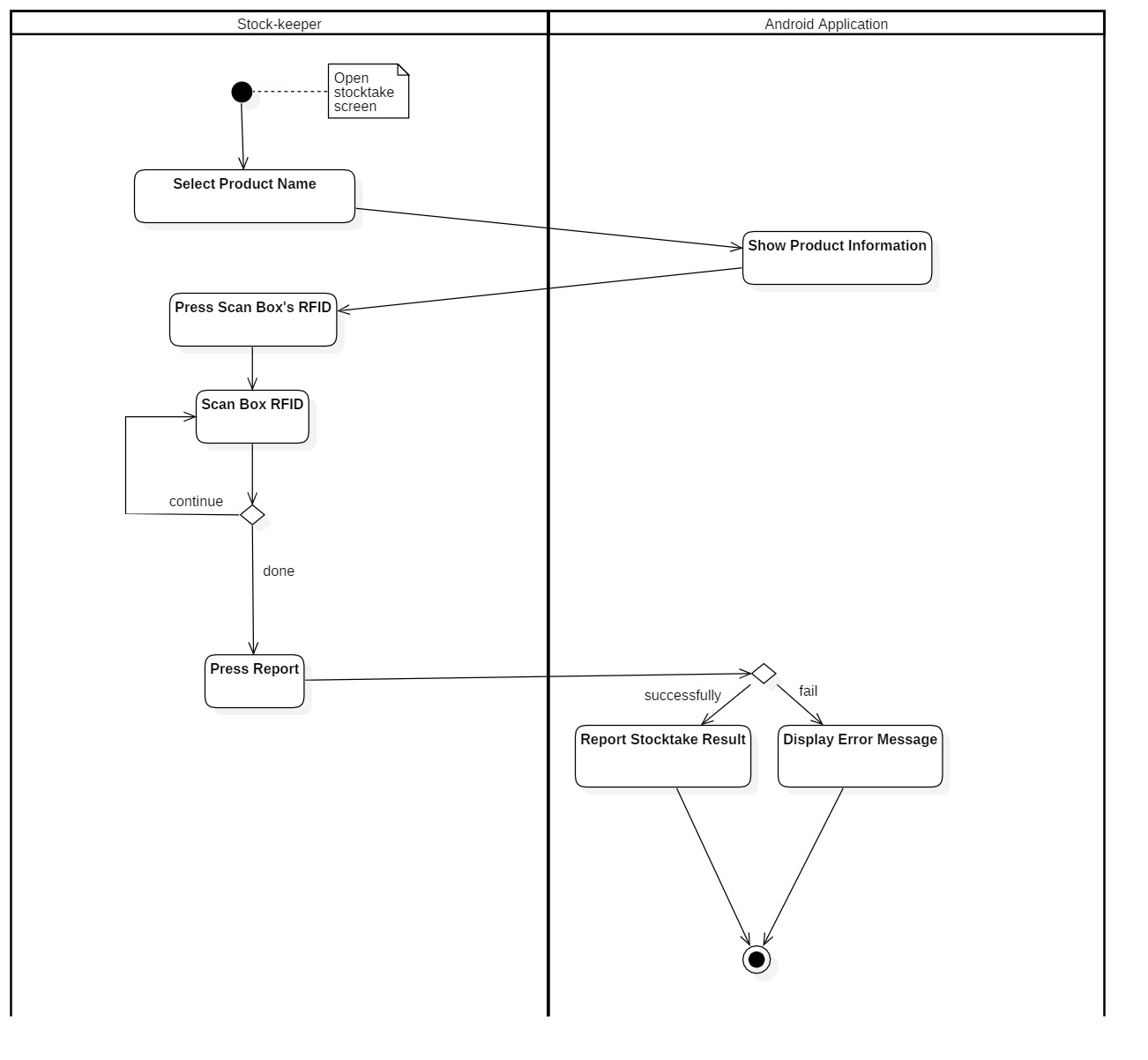


Figure 71- <Activity Diagram> Stocktake Inventory

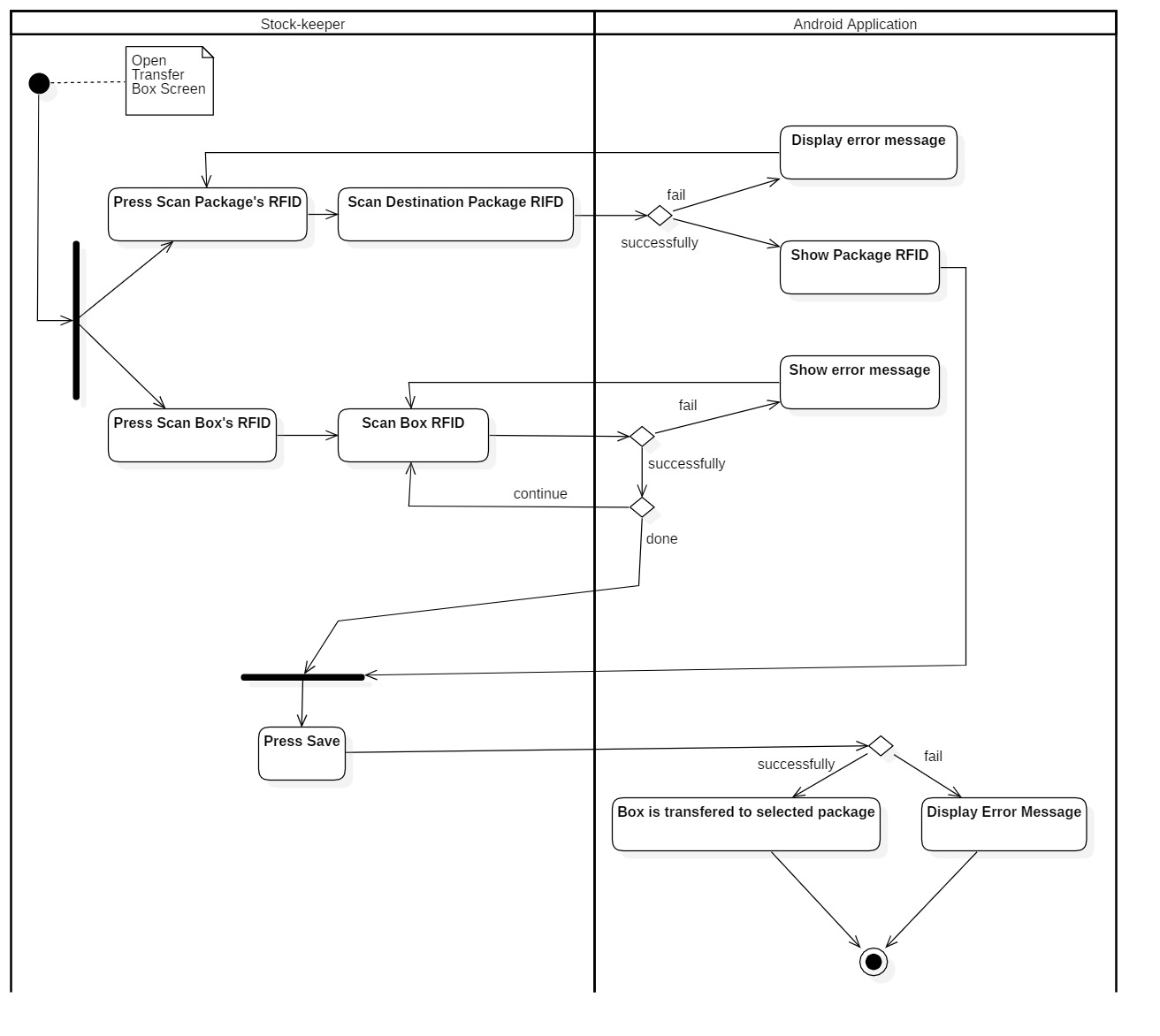


Figure 72- <Activity Diagram> Transfer Box

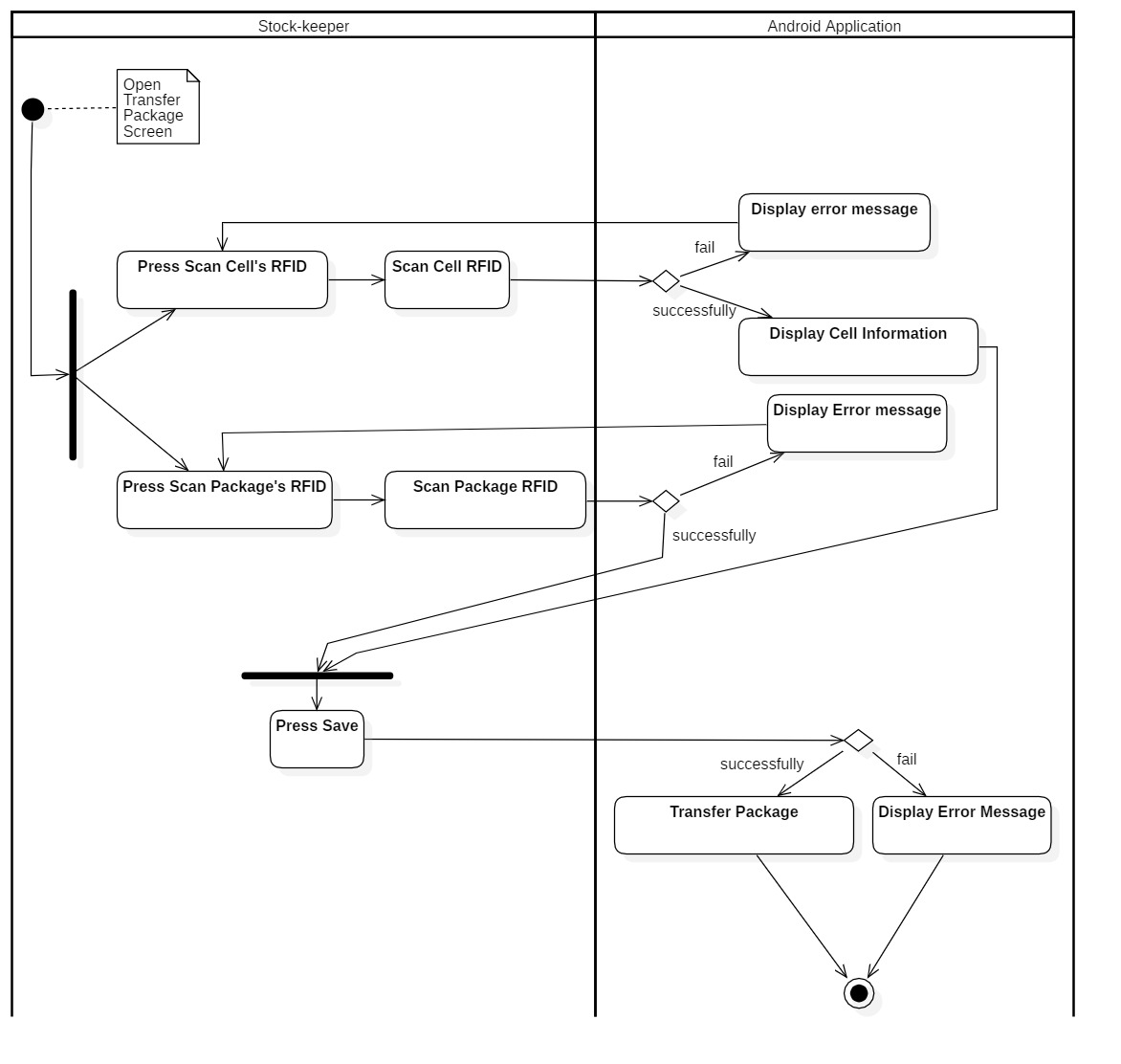


Figure 73- <Activity Diagram> Transfer Package

## User Interface Design

### Component Interface

Component Interface are Web Service API which will be used by Mobile Application.

| **Signature** | **Description** | **Input** | **Output** | **Output Format** |
| --- | --- | --- | --- | --- |
| getBoxRfidsByProductId(id) | Get box's rfid by using product’s id | id: String | Json list box’s rfid | String |
| getCellByFloorId(id) | Get cell by floor’s id | id: String | Json list cell’s information | String |
| registerCell(cell) | Register cell with rfid tag | cell: CellDTO | Http status code, String | String |
| getCellByCellRfid(id) | Get cell by cell’s rfid | id: String | Json cell’s information | String |
| checkCellIsEmpty(id) | Check cell is empty or not | id: String | Http status code, String | String |
| getFloorByShelfId(id) | Get floors by shelf’s id | id: String | Json list floor’s information | String |
| getFloorByCellId(id) | get floor by cell’s id | id: String | Json floor’s information | String |
| getAllActiveIssues() | Get all active issue (pending and processing) | N/A | Json list active (pending, processing) goods issue information | String |
| getAllActiveReceipt() | Get all active receipt (pending) | N/A | Json list active (peding) goods receipt’s information | String |
| suggestBox(request) | Suggest box using dijkstra and weighted average number | request: InvoiceInfoItem | Json suggesting box’s information | String |
| registerPackage(request) | Register package and box into system | request: RequestPackageInfo | Http status code, String | String |
| getPackageByPackageRfid(id) | Get package by package’s rfid | id: String | Json package’s information | String |
| stockIn(pac) | Stock in Package by mapping package with cell’s id | pac: PackageDTO | Http status code, String | String |
| stockOut(request) | Stock out box by remove box | request: RequestStockoutInfo | Http status code, String | String |
| transferPackages(pac) | Transfer package by update cell’s rfid | pac: PackageDTO | Http status code, String | String |
| transferBoxes(request) | Transfer box between packaged by update package’s rfid | request: RequestPackageInfo | Http status code, String | String |
| getAllProduct() | Get all products | N/A | Json list product’s information | String |
| getProudctByBoxId(id) | Get product by box’s id | id: String | Json product’s information | String |
| getAllShelves() | Get all shelves | N/A | Json list shelves’s information | String |
| getShelfByCellId(id) | Get shelves by floor’s id | id: String | Json shelf’s information | String |
| addStocktakeHistory(request) | Create stocktake history | request: StocktakeHistoryDTO | Http status code, String | String |
| login(request) | Login api | request: UserDTO | Json user’s information | String |

Table 51 - Component Interface

### User Interface Design

#### Mobile Application

##### Main Menu

Figure 75- <UI Design> Main Menu

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Register Package & Box | Register Package & Box into system | N/A | Transfer to Register Package & Box Screen |
| 2 | Stock In | Stock in Box into system | N/A | Transfer to Stock In Screen |
| 3 | Stock Out | Stock out box system | N/A | Transfer to Stock Out Screen |
| 4 | Transfer Product | Transfer boxes between packages and package between cells | N/A | Transfer to Transfer Product Screen |
| 5 | Stocktake Inventory | Check the current quantity of product’s box in warehouse | N/A | Transfer to Stocktake Inventory Screen |
| 6 | Register Cell (Shelf) | Register Cell into system | N/A | Transfer to Register Cell Screen |

Table 54- <UI Design>Main Menu Button

##### Register Shelf

Figure 76- <UI Design> Register Shelf

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Shelf | Shelf to register | Yes | Yes | Spinner | String |
| 2 | Floor | Floor to register | Yes | Yes | Spinner | String |
| 3 | Cell | Cell to register | Yes | Yes | Spinner | String |
| 4 | Cell RFID | Cell RFID to register with Cell | Yes | Yes | TextView | String |

Table 55- <UI Design> Register Shelf Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | Save | Register Cell into system | N/A | Message from system |
| 6 | Clear | Clear all filled information in screen | N/A | Cell RFID will be blank |
| 7 | Cancel | Exit Register Cell screen | N/A | Transfer to Main Menu |
| 8 | Scan Cell RFID | Scan cell RFID and fill into Cell RFID field | N/A | Cell RFID String |

Table 56- <UI Design>Register Shelf Buttons

##### Register Package and Box

Figure 77- <UI Design>Register Packages and Box

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Goods Receipt | Receipt Id to register | Yes | Yes | Spinner | String |
| 2 | Product Name | Product Name to register | Yes | Yes | Spinner | String |
| 3 | Product Id | Product Id to register | Yes | Yes | TextView | String |
| 4 | Package RFID | Package RFID to register | Yes | Yes | TextView | String |
| 5 | Box RFID | Boxes RFID to register | Yes | Yes | TextView | String |

Table 57- <UI Design> Register Package and Box Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 6 | Receipt Information | View Receipt Information | N/A | Transfer to Goods Receipt Screen |
| 7 | Scan Package RFID | Scan Package RFID and fill into Package RFID field | N/A | Package RFID String |
| 8 | Scan Box RFID | Scan Box RFID and fill into Box RFID field | N/A | Box RFID String |
| 9 | Save | Register package and boxes into system | N/A | Message from system |
| 10 | Clear | Clear all filled information in screen | N/A | Clear all RFID field and all spinner will select the first item |
| 11 | Cancel | Exit Register Package & Box screen | N/A | Transfer to to Main Menu |

Table 58- <UI Design> Register Package and Box Button

##### Goods Receipt

Figure 78- <UI Design>Goods Receipt

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Goods Receipt Information | Show all goods receipt info | Yes | Yes | RecyclerView | String |

Table 59- <UI Design> Goods Receipt Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Exit | Exit Goods Receipt Screen | N/A | Transfer to Register Package & Box screen |

Table 60- <UI Design>Goods Receipt Button

##### Stock In

Figure 79- <UI Design> Stock In

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Cell RFID | Cell RFID for stock in | No | Yes | TextView | String |
| 2 | Cell | Cell ID for stock in | No | Yes | TextView | String |
| 3 | Floor | Floor ID for stock in | No | Yes | TextView | String |
| 4 | Shelf | Shelf ID for stock in | No | Yes | TextView | String |
| 5 | Package RFID | Package RFID for stock in | No | Yes | TextView | String |

Table 61- <UI Design> Stock In Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 6 | Scan Cell RFID | Scan Cell RFID and fill into Cell RFID field | N/A | Cell RFID String |
| 7 | Scan Package RFID | Scan Package RFID and fill into Package RFID field | N/A | Package RFID String |
| 8 | Save | Stock in package into warehouse | N/A | Message from system |
| 9 | Clear | Clear all filled information in screen | N/A | Clear all RFID field and all spinner will select the first item |
| 10 | Cancel | Exit Stock in screen | N/A | Transfer to to Main Menu |

Table 62- <UI Design>Stock In Buttons

##### Stock out

Figure 80- <UI Design> Stock Out

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Goods Issue | Issue Id to stock out | No | Yes | TextView | String |
| 2 | Box Scanned Information | Information of box and quantity of scanned boxes | No | Yes | RecyclerView | String |

Table 63- <UI Design> Stock Out Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Scan Box RFID | Scan Box RFID and fill product information of this box into product field | N/A | Box RFID String and product information |
| 4 | Goods Issue Information | View Goods Issue Information | N/A | Transfer to Issue Invoice Screen |
| 5 | Save | Stock out package from warehouse | N/A | Message from system |
| 6 | Clear | Clear all filled information in screen | N/A | Clear all RFID field |
| 7 | Cancel | Exit Stock Out screen | N/A | Transfer to Main Menu |

Table 64- <UI Design> Stock Out Buttons

##### Goods Issue

Figure 81- <UI Design> Goods Issue

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Goods Issue Information | Show all goods issue info | Yes | Yes | RecyclerView | String |

Table 65- <UI Design>Goods Issue Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Exit | Exit Goods Issue Screen | N/A | Transfer to Stock Out screen |

Table 66- <UI Design> Goods Issue Button

##### Suggest Box

Figure 82- <UI Design> Suggest Box

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Suggest Information | Show information about suggest box | Yes | Yes | ExpandableListView | String |

Table 67- <UI Design> Suggest Box Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Exit | Exit Suggest Box Screen | N/A | Transfer to Goods Issue screen |

Table 68- <UI Design> Suggest Box Button

##### Transfer Box

Figure 83- <UI Design> Transfer Box

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Package RFID | Package RFID which will be transfer | No | Yes | TextView | String |
| 2 | Scanned box Information | Information of box and quantity | No | Yes | RecyclerView | String |

Table 69- <UI Design> Transfer Box Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Scan Package RFID | Scan Package RFID and fill into Package RFID field | N/A | Package RFID String |
| 4 | Scan Box RFID | Scan Box RFID and fill product information of this box into product field | N/A | Box RFID String and product information |
| 5 | Save | Transfer box between package | N/A | Message from system |
| 6 | Clear | Clear all filled information in screen | N/A | Clear all RFID field and all spinner will select the first item |
| 7 | Cancel | Exit Transfer Box screen | N/A | Transfer to Main Menu |
| 8 | Transfer Box | Switch to Transfer Box Tab | N/A | Transfer to Transfer Box Tab |
| 9 | Transfer Package | Switch to Transfer Package Tab | N/A | Transfer to Transfer Package Tab |

Table 70- <UI Design>Transfer Box Button

##### Transfer Package

Figure 84- <UI Design> Transfer Package

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Cell RFID | Cell RFID of target cell | No | Yes | TextView | String |
| 2 | Cell | Cell Id of target cell | No | Yes | TextView | String |
| 3 | Floor | Floor Id contain target cell | No | Yes | TextView | String |
| 4 | Shelf | Shelf Id contain target cell | No | Yes | TextView | String |
| 5 | Package RFID | Package RFID to transfer | No | Yes | TextView | String |

Table 71- <UI Design> Transfer Package Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 6 | Scan Cell RFID | Scan Cell RFID and fill into Cell RFID field | N/A | Cell RFID String |
| 7 | Scan Package RFID | Scan Package RFID and fill into Package RFID field | N/A | Package RFID String |
| 8 | Save | Transfer package between cells | N/A | Message from system |
| 9 | Clear | Clear all filled information in screen | N/A | Clear all RFID field |
| 10 | Cancel | Exit Transfer Package screen | N/A | Transfer to Main Menu |
| 11 | Transfer Box | Switch to Transfer Box Tab | N/A | Transfer to Transfer Box Tab |
| 12 | Transfer Package | Switch to Transfer Package Tab | N/A | Transfer to Transfer Package Tab |

Table 72- <UI Design> Transfer Package Buttons

##### Stocktake inventory

Figure 85- <UI Design> Stocktake Inventory

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Product Name | Product name to stocktake | No | Yes | TextView | String |
| 2 | Product Id | Product Id to stocktake | No | Yes | TextView | String |
| 3 | Available Quantity | Quantity archived in system | No | Yes | TextView | String |
| 4 | Scanned Quantity | Real quantity in warehouse | No | Yes | TextView | String |

Table 73- <UI Design> Stocktake Inventory Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | Scan Box RFID | Scan Box RFID and fill into Box RFID field | N/A | Scanned quantity increase 1 |
| 6 | Report | Report stocktake result to the system | N/A | Message from system |
| 7 | Clear | Clear all filled information in screen and scanned quantity | N/A | Scanned quantity return zero and all spinner will select the first item |
| 8 | Cancel | Exit Stocktake Inventory screen | N/A | Transfer to to Main Menu |

Table 74- <UI Design> Stocktake Inventory Buttons

#### Web System

##### Accountant Homepage

Figure 96- <UI Design>Accountant Homepage

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Total Categories | Total number of categories | Yes | No | Text | String |
| 2 | Total products | Total number of products | Yes | No | Text | String |
| 3 | Total Shelfs | Total number of shelf | Yes | No | Text | String |
| 4 | Total Vendors | Total number of vendor | Yes | No | Text | String |
| 5 | Pending Goods Receipt List | Total Goods Receipt with pending status | Yes | No | Text | String |
| 6 | Pending Goods Issue List | Total Goods Issue with pending status | Yes | No | Text | String |

Table 92- <UI Design>Accountant Homepage Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 7 | Show All Goods Receipt and Issue | Show all goods receipt and issue | N/A | Transfer to “Receipts/Issues” page |
| 8 | Show Categories | Show all categories | N/A | Transfer to “Categories” page |
| 9 | Show All Products | Show all products | N/A | Transfer to “Products” page |
| 10 | Show All Shelves | Show all shelves | N/A | Transfer to “Shelves” page |
| 11 | Show All Stocktake History | Show all stocktake history | N/A | Transfer to “Stocktake Histories” page |
| 12 | Show Report Type | Show all report type | N/A | Transfer to “Reports” page |

Table 93- <UI Design>Accountant Homepage Button

##### Create Product

Figure 103- <UI Design> Create Product

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Product Id | Id of product | No | Yes | Text | String |
| 2 | Product Name | Name of product | No | Yes | Text | String |
| 3 | Image | Image of product | No | No | File | String |
| 4 | Description | Description of product | No | No | Text | String |
| 5 | Weight | Weight of product | No | No | Text | Float |
| 6 | Height | Height of product | No | No | Text | Float |
| 7 | Width | Width of product | No | No | Text | Float |
| 8 | Length | Length of product | No | No | Text | Float |
| 9 | Quantity per box | Quantity product item per product’s box | No | Yes | Text | Integer |
| 10 | Category | Category of product | No | Yes | Select | String |
| 11 | Vendor | Vendor of product | No | Yes | Select | String |

Table 104- <UI Design>Create Product Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 12 | Cancel | Cancel create product process | N/A | Transfer to list all product’s screen |
| 13 | Confirm | Create new product successfully | N/A | Transfer to list all product’s screen with new product which was created |

Table 105- <UI Design>Create Product Buttons

##### Edit Product

Figure 104- <UI Design>Edit Product

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Product Id | Id of product | Yes | Yes | Text | String |
| 2 | Product Name | Name of product | No | Yes | Text | String |
| 3 | Image | Image of product | No | No | File | String |
| 4 | Description | Description of product | No | No | Text | String |
| 5 | Weight | Weight of product | No | No | Text | Float |
| 6 | Height | Height of product | No | No | Text | Float |
| 7 | Width | Width of product | No | No | Text | Float |
| 8 | Length | Length of product | No | No | Text | Float |
| 9 | Quantity per box | Quantity product item per product’s box | No | Yes | Text | Integer |
| 10 | Category | Category of product | No | Yes | Select | String |
| 11 | Vendor | Vendor of product | No | Yes | Select | String |

Table 106- <UI Design>Edit Product Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 12 | Cancel | Cancel edit product process | N/A | Transfer to list all product’s screen |
| 13 | Confirm | Edit product successfully | N/A | Transfer to list all product’s screen with product that was edited |

Table 107- <UI Design>Edit Product Buttons

##### Import Product

Figure 107- <UI Design> Import Product

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Upload File | File list of products | No | Yes | File | String |

Table 110- <UI Design>Import Product Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Sample File | Download sample file | N/A | A sample file will be downloaded |
| 3 | Confirm | Import product successfully | N/A | Transfer to list all product’s screen with products that are created |
| 4 | Cancel | Cancel import product process | N/A | Transfer to list all product’s screen |

Table 111- <UI Design>

##### Deactivate Shelf

Figure 112- <UI Design> Deactivate Shelf

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Deactivate Shelf | Deactivate shelf’s status | N/A | Transfer to list all shelf’s screen with shelf’s status is deactivated |
| 2 | Cancel | Cancel deactivate shelf process | N/A | Transfer to list all shelf screen |

Table 119- <UI Design>Deactivate Shelf Buttons

##### Activate Shelf

Figure 113- <UI Design> Activate Shelf

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Activate Shelf | Activate shelf’s status | N/A | Transfer to list all shelf’s screen with shelf’s status is activated |
| 2 | Cancel | Cancel activate shelf process | N/A | Transfer to list all shelf screen |

Table 120- <UI Design> Activate Shelf Button

##### Detail Shelf

Figure 114- <UI Design>Detail Shelf

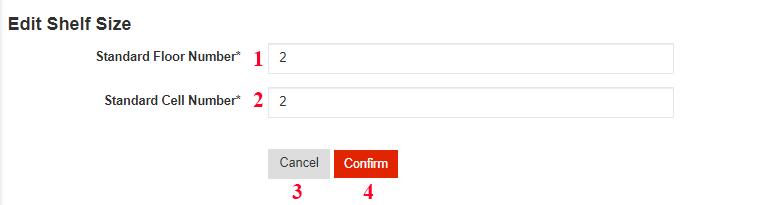
* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Floor Id | Floor Id of shelf | Yes | Yes | Text | String |
| 2 | Cell Id | Cell Id of shelf | Yes | No | Text | String |
| 3 | Cell RFID | RFID tags in cell | Yes | Yes | Text | String |
| 4 | Status | Status of cell | Yes | No | Text | String |

Table 121- <UI Design>Detail Shelf Fields

##### Edit Standard Shelf Size

Figure 115- <UI Design>Edit Standard Shelf Size



* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Standard Floor Number | Floor number of shelf | No | Yes | Text | Integer |
| 2 | Standard Cell Number | Cell number per Floor | No | Yes | Text | Integer |

Table 122- <UI Design> Edit Standard Shelf Size Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Cancel | Cancel edit shelf size process | N/A | Transfer to list all shelf’s screen |
| 4 | Confirm | Edit shelf size successfully | N/A | Transfer to list all shelf’s screen with new shelf size that is edited |

Table 123- <UI Design>Edit Standard Shelf Size Buttons

##### View Goods Receipt

Figure 116- <UI Design> View Goods Receipt

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Receipt Id | Id of receipt | Yes | Yes | Text | String |
| 2 | Date | Creation date of receipt | Yes | Yes | Text | Date |
| 3 | Status | Status of receipt | Yes | Yes | Text | String |
| 4 | Created by | Creator of receipt | Yes | Yes | Text | String |

Table 124- <UI Design>View Goods Receipt Fields

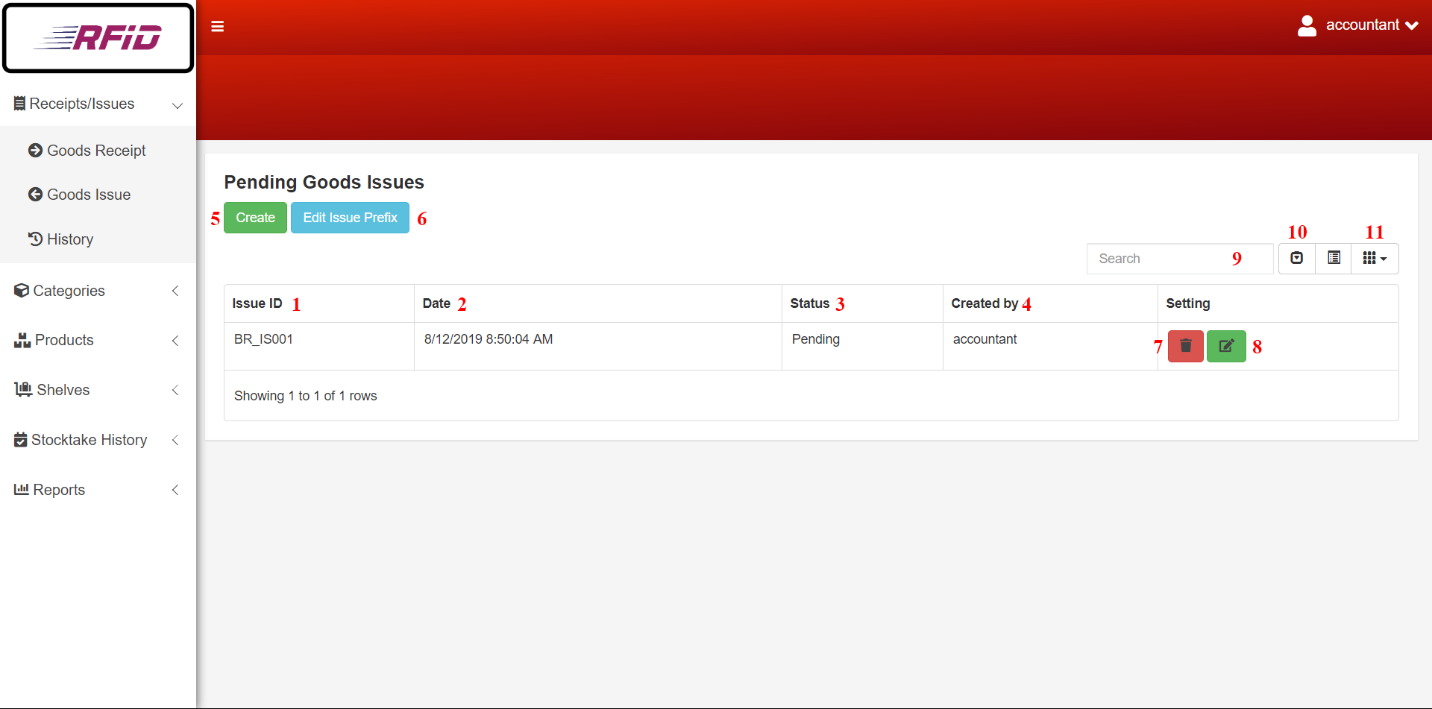
* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | Create Receipt | Create new receipt | N/A | Transfer to create receipt screen |
| 6 | Edit Prefix | Edit prefix of receipt | N/A | Transfer to edit prefix screen |
| 7 | Delete receipt | Delete receipt | N/A | Show delete receipt dialog |
| 8 | Edit Receipt | Edit receipt | N/A | Transfer to edit receipt screen |
| 9 | Search | Search receipt by keyword | N/A | Show list of receipt are filtered by keyword |
| 10 | Show pagination | Show pagination | N/A | Show pagination |
| 11 | Show data columns | Show columns are checked | N/A | Display all columns are checked |

Table 125- <UI Design>View Goods Receipt Buttons

##### View Goods Issue

Figure 116- <UI Design> View Goods Issue



* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Issue Id | Id of issue | Yes | Yes | Text | String |
| 2 | Date | Creation date of issue | Yes | Yes | Text | Date |
| 3 | Status | Status of issue | Yes | Yes | Text | String |
| 4 | Created by | Creator of issue | Yes | Yes | Text | String |

Table 126- <UI Design>View Goods Issue Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | Create Issue | Create new issue | N/A | Transfer to create issue screen |
| 6 | Edit Prefix | Edit prefix of issue | N/A | Transfer to edit prefix screen |
| 7 | Delete Issue | Delete issue | N/A | Show delete issue dialog |
| 8 | Edit Issue | Edit issue | N/A | Transfer to edit issue screen |
| 9 | Search | Search issue by keyword | N/A | Show list of issues are filtered by keyword |
| 10 | Show pagination | Show pagination | N/A | Show pagination |
| 11 | Show data columns | Show columns are checked | N/A | Display all columns are checked |

Table 127- <UI Design>View Goods Issue Buttons

##### Edit Prefix

Figure 117- <UI Design>Edit Prefix

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Prefix | Prefix string of invoice Id | No | Yes | Text | String |

Table 130- <UI Design> Edit Prefix Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Cancel | Cancel edit prefix process | N/A | Transfer to view goods receipt or to view goods issue screen |
| 3 | Confirm | Edit Prefix of Invoice | N/A | Transfer to view goods receipt or to view goods issue screen |

Table 131- <UI Design> Edit Prefix Buttons

##### Create Invoice

Figure 118- <UI Design> Create Invoice [2]

Figure 119- <UI Design> Create Invoice [1]

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Product Id | Product id of product | Yes | Yes | Text | String |
| 2 | Product Name | Product name of product | Yes | Yes | Text | String |
| 3 | Category | Category of product | Yes | Yes | Text | String |
| 4 | Vendor | Vendor of product | Yes | Yes | Text | String |
| 5 | Quantity | Quantity of product | No | Yes | Text | Integer |

Table 132- <UI Design> Create Invoice Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 6 | Choose Product | Choose products to add into invoice | N/A | Transfer to choose product screen |
| 7 | Cancel | Cancel create invoice process | N/A | Transfer to list all invoice screen |
| 8 | Finish | Create new invoice successfully | N/A | Transfer to list all invoices’ screen with new invoice that is created |

Table 133- <UI Design> Create Invoice Buttons

##### Choose Product

Figure 120- <UI Design>Choose Product

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Checkbox | Checkboxes for choose products | No | Yes | Checkbox | Boolean |
| 2 | Available Quantity | Available Quantity in Warehouse. Only appear in Stock out | Yes | Yes | Text | String |

Table 134- <UI Design>Choose Product Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Confirm | Add chosen products to invoice | N/A | Transfer to add product to invoice screen |
| 4 | Search | Filter Product list with key | N/A | Filtered product list |
| 5 | Cancel | Cancel choosing products | N/A | Transfer to add product to invoice screen |

Table 135- <UI Design>Choose Product Button

##### Edit Invoice

Figure 121- <UI Design>Edit Invoice

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Quantity | Quantity of product | No | Yes | Text | Integer |

Table 136- <UI Design>Edit Invoice Fields

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Choose Product | Choose products for edit invoice | N/A | Transfer to choose product screen |
| 3 | Cancel | Cancel edit invoice process | N/A | Transfer to list all invoice screen |
| 4 | Finish | Edit invoice successfully | N/A | Transfer to list all invoices’ screen with edited invoice |

Table 137- <UI Design>Edit Invoice Button

##### Delete Invoice

Figure 122- <UI Design>Delete Invoice

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Delete | Delete invoice | N/A | Transfer to list all invoice screen and remove this invoice |
| 2 | Cancel | Cancel delete invoice process | N/A | Transfer to list all invoice screen |

Table 138- <UI Design>Delete Invoice Button

##### Detail Invoice

Figure 123- <UI Design>Detail Invoice

* **Fields**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** |
| 1 | Date | Date of invoice | Yes | Yes | Text | Date |
| 2 | Type | Type of invoice | Yes | Yes | Text | String |
| 3 | Status | Status of invoice | Yes | Yes | Text | String |
| 4 | Description | Description of invoice | Yes | Yes | Text | String |
| 5 | User create | User that create invoice | Yes | Yes | Text | String |
| 6 | Product Id | Product id of product | Yes | Yes | Text | String |
| 7 | Product Name | Product name of product | Yes | Yes | Text | String |
| 8 | Category | Category of product | Yes | Yes | Text | String |
| 9 | Vendor | Vendor of product | Yes | Yes | Text | String |
| 10 | Quantity | Quantity of product | Yes | Yes | Text | Integer |

Table 139- <UI Design>Detail Invoice Fields

##### Create Report

Figure 124- <UI Design>Create Report

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Good Receive/Issue | Good receive/issue report | N/A | Transfer to good receive/issue pdf layout |
| 2 | Product | Product report | N/A | Transfer to good product pdf layout |
| 3 | Category | Category report | N/A | Transfer to category pdf layout |
| 4 | Vendor | Vendor report | N/A | Transfer to vendor pdf layout |

Table 140- <UI Design>Create Report Buttons

##### Update Stocktake Status

Figure 126- <UI Design>Update Stocktake Status

* **Button**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Update | Update stocktake status | N/A | Transfer to list all stocktake screen and update this stocktake status |
| 2 | Cancel | Cancel update stocktake status process | N/A | Transfer to list all stocktake screen |

Table 145- <UI Design>Update Stocktake Status Buttons

**All UI Design could be found here:** https://drive.google.com/open?id=1tZ4yjZRzz0DGS093IR\_NqZk0rnEmpqky

## Database Design

### Entity Relationship Diagram (ERD)

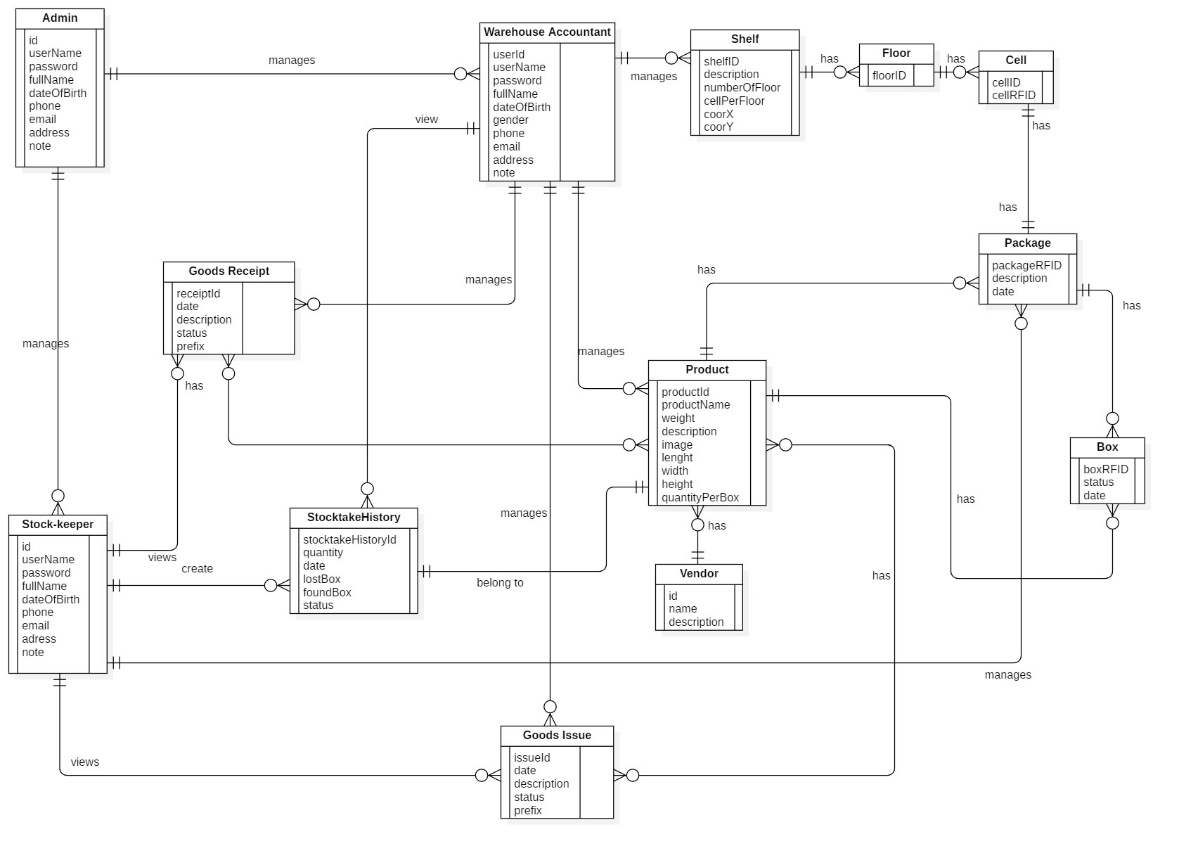


Figure 127 - Entity Relationship Diagram

### Entity Dictionary

| **Entity Data Dictionary: Describe content of all entities** | | |
| --- | --- | --- |
| **Entity Name** | **Mapping entity with conceptual diagram** | **Description** |
| Admin | Admin | Contains information of admin. |
| Stock-keeper | Stock-keeper | Contains information of stock-keeper. |
| Warehouse Accountant | Warehouse Accountant | Contains information of warehouse accountant. |
| GoodsReceipt | GoodsReceipt | Contains information of goods receipt. |
| GoodsIssue | GoodsIssue | Contains information of goods issue. |
| Product | Product | Contains information of product. |
| StocktakeHistory | Stocktake History | Contains information of stocktake history. |
| Vendor | Vendor | Contains information of vendor. |
| Shelf | Shelf | Contains information of shelf. |
| Floor | Floor | Contains information of floor. |
| Cell | Cell | Contains information of cell. |
| Package | Package | Contains information of package. |
| Box | Box | Contains information of box. |

Table 146 - Entity Data Dictionary

## Algorithms

### Find box base on the longest stock in date and closest warehouse’s gate

#### Definition

This algorithm intended to suggest stock-keeper the suitable box when they need to find box to stock out.

#### Problem Definition

When stock-keeper get an issue invoice, the stock-keeper do not know where to pick the box to stock out. In addition, a box have longest stock in date need to be stock out first but stock-keeper also do not know about this. This algorithm, provide information that help stock-keeper can choose the closest box and have longest date in warehouse.

#### Solution

To suggest box information for stock-keeper, we use weighted mean formula to solve this problem:­

**With:**

**: the weighted mean**

**x: the value of criteria**

**w: the weighted number**

In current problem, we have two criteria need to consider there are:

- The first criteria is the difference days between stock in date and now.

- The second criteria is the distance from source shelf to target shelf. (Because the farther away the distance, the lower priority so that the distance value need to be multiply with -1)

The priority of shelves is based on weighted mean values. The shelf that have larger weighted mean is better.

#### Example

**We have list shelves:**

|  |  |  |
| --- | --- | --- |
| **Criteria**  **Shelf** | **Days (70%)** | **Distance (30%)** |
| A | 20 | 10 |
| B | 25 | 8 |
| Shelf C | 20 | 8 |

**Calculate weighted mean values:**

**Order weighted mean values descending:**

|  |  |
| --- | --- |
| **Shelf** | **Weighted Average Number** |
| **B** | **15.1** |
| **C** | **11.6** |
| **A** | **11** |

#### Complexity

In total, the complexity of this algorithm is O(n).

#### Flowchart

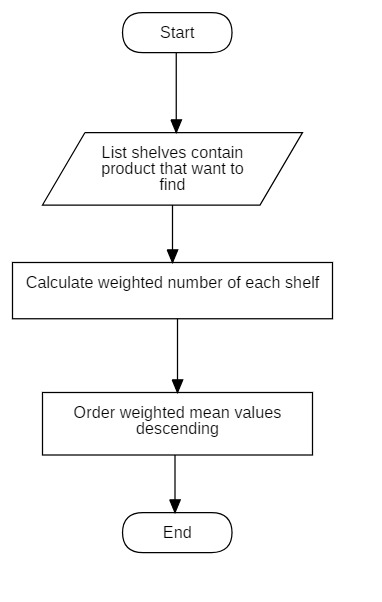
****

Figure 128<Algorithm> Optimizing Stock Out Product Flowchart

### Convert shelves’ coordinates to matrix algorithm

#### Definition

This algorithm converts shelves’ coordinates into matrix.

#### Definition Problem

Each shelf in the warehouse has its coordinates and we need to convert them into matrix for Dijkstra algorithm to analyze.

#### Solution

To convert all shelves’ coordinates into matrix, we follow these steps:

* Assume the coordinates of two shelves are (XA, YA), (XB, YB), distance between two column is x and distance between two row is y
* Compare the coordinates:
* If XA = XB and |YA – YB| = 1: two shelves are adjacent with distance between two shelves is x
* If XA = XB and |YA – YB| > 1: two shelves are not adjacent
* If XA = XB and |YA – YB| = 0: two shelves are one
* If YA = YB and |XA – XB| = 1: two shelves are adjacent with distance between two shelves is y
* If YA = YB and |XA – XB| > 1: two shelves are not adjacent
* If YA = YB and |XA – XB| = 0: two shelves are one

**Example:**

Assume thatdistance between two columns is x = 2 and distance between two rows is y = 1



Figure 129- <Algorithm> Shelf Coordinate Map

After calculating:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **A1** | **A2** | **A3** | **B1** | **B2** | **B3** | **C1** | **C2** | **C3** |
| **A1** | **0** | **2** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |
| **A2** | **2** | **0** | **2** | **0** | **1** | **0** | **0** | **0** | **0** |
| **A3** | **0** | **2** | **0** | **0** | **0** | **1** | **0** | **0** | **0** |
| **B1** | **1** | **0** | **0** | **0** | **2** | **0** | **1** | **0** | **0** |
| **B2** | **0** | **1** | **0** | **2** | **0** | **2** | **0** | **1** | **0** |
| **B3** | **0** | **0** | **1** | **0** | **2** | **0** | **0** | **0** | **1** |
| **C1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **2** | **0** |
| **C2** | **0** | **0** | **0** | **0** | **1** | **0** | **2** | **0** | **2** |
| **C3** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **2** | **0** |

Table 147- <Algorithm> Shelf Coordinate Mapping Table

#### Complexity

In total, the complexity of this algorithm is O(n2).

#### Flowchart

Figure 130- <Algorithm> Shelf Coordinate Mapping Flowchart

### Dijkstra algorithms

#### Definition

This algorithm is used to find the shortest path.

#### Problem Definition

Our use of this algorithm is to calculate the shortest distance from the source shelf (1; 1) to the other shelves.

#### Solution

In order to calculate the shortest distance, we use Dijkstra algorithm.

Create a list ListVertices which contains all vertices that already had been calculated the shortest distance.

Set the source vertex 0 and distance equals 0. It will be picked first and added to ListVertices;

While ListVertices does not contain all vertexes:

* Choose a vertice with minimum distance value.
* Add it to ListVertices.
* Find its adjacent vertices and update distance value of its adjacent vertice. (If adjacent vertice has the distance value already, compare the new distance with the old distance. If the new distance is smaller, update the new distance, if the new distance is smaller, keep the old distance)

#### Example

We have the following example:

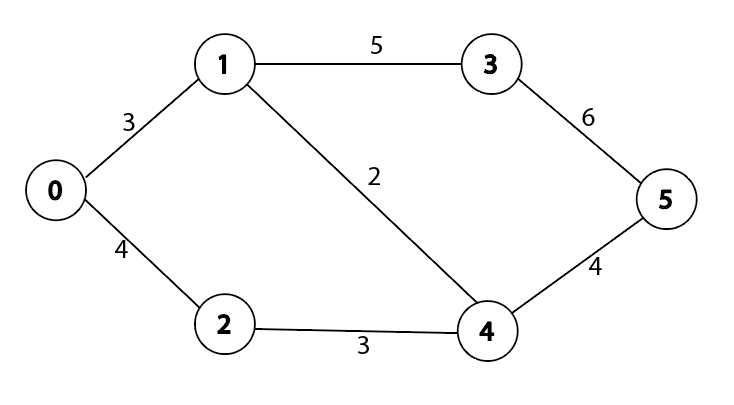


Figure 131- <Algorithm> Dijkstra Example

First we choose 0 as the source vertex, add it into ListVertices. Its adjacent is 1 and 2 and the distance value are updated as 3 and 4.

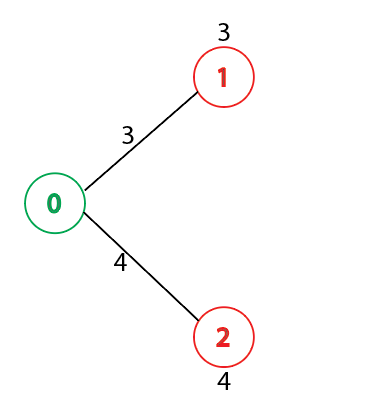


Figure 132- <Algorithm> Dijkstra Example

Pick the vertex with minimum distance value and add it to ListVertices. In the current situation vertex with minimum distance is 1 and ListVertices become {0, 1}. Update the distance value of vertex 1’s adjacent vertices are vertex 3 and vertex 4 as 8 and 5.

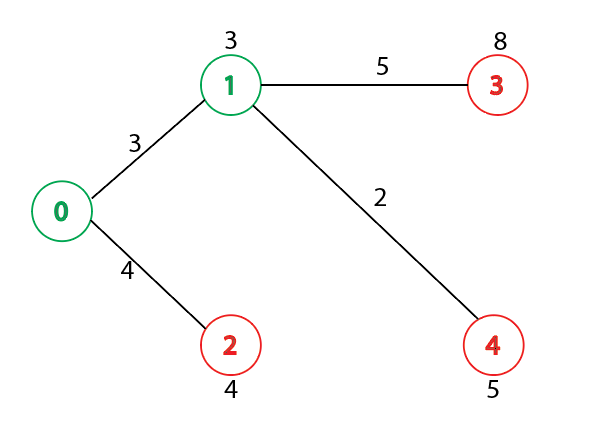


Figure 133- <Algorithm> Dijkstra Example

Pick the next vertex with minimum distance value and add it to ListVertices and the ListVertices becomes {0, 1, 2}. We pick vertex 2 with distance value is 4. The adjacent of vertex 2 is vertex 4 but the distance from vertex 2 to vertex 4 (4 + 3 = 7) is larger than old distance (5) so we keep old distance.

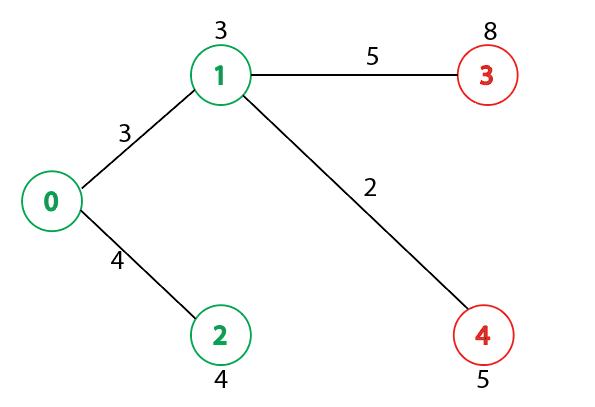


Figure 134- <Algorithm> Dijkstra Example

Iterative it until all ListVertices contain all of vertices we have the shortest distance value from source vertex to other vertices.

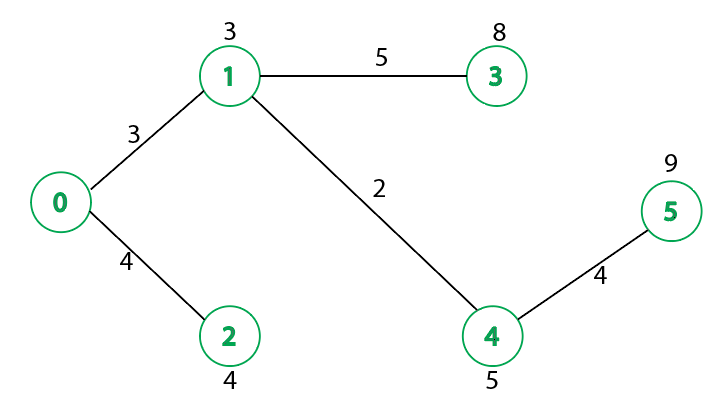


Figure 135- <Algorithm> Dijkstra Example

#### Complexity

In total, the comlexity of this algorithm is O(n2)

#### C:\Users\Ryan\Desktop\Chart\Dijkstra.jpgFlowchart

Figure 136- <Algorithm> Dijkstra Flowchart

# E. System Implementation & Testing

Introduction

### Overview

This section describes the approach and methodologies used by group to plan out, organize and manage the testing of RFIM system. It provides in the detail all necessary information about the implementation and testing procedure of the system included test plans, test cases, test result, test environments, pass/fail criteria and risks estimations as well as a checklist to cover all possible cases.

### Test Approach:

* **Goal:** Test all features in the whole RFIM system based on the core flow.
* **Method**: black-box testing
* **Technique:** check list

The testing for this project will consists of Integration System test level. Testing the program which was integrated and as a complete system to ensure that the software requirements have been met.

Integration testing would be performed by all member of team and approved by team leader.

System testing is focused on assessing the system’s reliability. This process is concerned with finding errors that result from unanticipated interactions between components and component interface problems.

## Database Relationship Diagram

### Physical Diagram

Figure 137 – Database Diagram

### Data Dictionary

| **Entity Data Dictionary: Describe content of all entities** | | |
| --- | --- | --- |
| **Entity Name** | **Mapping table with class diagram** | **Description** |
| User | User | Contains the information of user. |
| Role | Role | Contains the information of role. |
| Invoice | Goods Receipt Goods Issue | Contains the information of goods receipt and Issue |
| StocktakeHistory | StocktakeHistory | Contains the information of stocktake history. |
| Product | Product | Contains the information of product. |
| Vendor | Vendor | Contains the information of vendor. |
| Category | Category | Contains the information of category. |
| Invoice\_Product | Product\_GoodsReceipt Product\_GoodsIssue | Contains mapping between good receipt/issue and product. Contains info of products in goods receipt/issue |
| InvoiceType | N/A | Contains the type of goods receipt/issue. |
| InvoiceStatus | N/A | Contains the status of goods receipt/issue. |
| Shelf | Shelf | Contains the information of shelf. |
| Floor | Floor | Contains the information of floor. |
| Cell | Cell | Contains the information of cell. |
| Package | Package | Contains information of package. |
| Box | Box | Contains information of box. |
| StandardShelfSize | N/A | Contains information of standard shelf size |
| ConstantTable | N/A | Contains contstant generated from algorithm |

Table 148 Database Data Ditionary

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Allow Null** |
| --- | --- | --- | --- | --- |
| User | UserId {PK} | Unique identifier of user | Integer | No |
| UserName | Username of user | String | No |
| FullName | Fullname of user | String | No |
| DateOfBirth | Date of birth of user | Date | No |
| Phone | Phone of user | String | Yes |
| Email | Email of user | String | Yes |
| Password | Password of user | String | No |
| Address | Address of user | String | Yes |
| Note | Note of user | String | Yes |
| RoleId | Role of user | Integer | No |
| Status | Status of user | Boolean | No |
| Role | RoleId {PK} | Unique identifier of role | Integer | No |
| RoleName | Role name of role | String | Yes |
| Invoice | InvoiceId {PK} | Unique identifier of invoice | String | No |
| Date | Date of invoice | Date | No |
| Description | Description of invoice | String | Yes |
| StatusId | Status of invoice | Integer | No |
| UserId | Creator of invoice | Integer | No |
| InvoiceTypeID | Type of invoice | Integer | No |
| InvoiceStatus | StatusId | Unique identifier of invoice status | Integer | No |
| Status | Status name of invoice | String | Yes |
| InvoiceType | InvoiceTypeId {PK} | Unique identifier of invoice type | Integer | No |
| InvoiceTypes | Type name of invoice | String | Yes |
| InvoicePrefix | Prefix of invoice type | String | Yes |
| Stocktake History | StocktakeHistoryId {PK} | Unique identifier of stocktake history | Integer | No |
| Quantity | Quantity of box after stocktaking | Integer | No |
| Date | Date of stocktake history | Date | Yes |
| Status | Status of stocktake history | Boolean | No |
| LostBox | Lost box’s RFID after stocktake | String | Yes |
| FoundBox | Missing box RFID was found after stocktake | String | Yes |
| UserId | Creator of stocktake history | Integer | No |
| ProductId | Product Id of stocktake | String | No |
| Product | ProductId {PK} | Unique identifier of product | String | No |
| ProductName | Product name of product | String | No |
| Weight | Weight of product | Float | Yes |
| Description | Description of product | String | Yes |
| Image | Image of product | String | Yes |
| Length | Length of product box | Float | Yes |
| Width | Width of product box | Float | Yes |
| Height | Height of product box | Float | Yes |
| QuantityPerBox | Number of items per product box | Integer | No |
| CategoryId | Category of product | Integer | No |
| VendorId | Vendor of product | Integer | No |
| Status | Status of product | Boolean | No |
| Vendor | VendorId {PK} | Unique identifier of vendor | Int | No |
| VendorName | Vendor name of vendor. | String | No |
| Description | Description of vendor | String | Yes |
| Category | CategoryId {PK} | Unique identifier of category | Int | No |
| CategoryName | Category name of category | String | No |
| Description | Description of category | String | Yes |
| Status | Status of category | Boolean | No |
| Invoice\_Product | ProductId | Product Id from product | String | No |
| InvoiceId | Invoice Id from invoice | String | No |
| Quantity | Quantity of box for stock in/out | Integer | No |
| ProcessQuantity | Current scanned box quantity | Integer | Yes |
| Shelf | ShelfId {PK} | Unique identifier of shelf | String | No |
| Description | Description of shelf | String | Yes |
| FloorNumber | Number of floor | Integer | No |
| CallNumber | Number of cell per floor | Integer | No |
| CoorX | Coordinate X of shelf | Integer | No |
| CoorY | Coordinate Y of shelf | Integer | No |
| Length | Length of shelf | Float | Yes |
| Width | Width of shelf | Float | Yes |
| Height | Height of shelf | Float | Yes |
| Status | Status of shelf | Boolean | Yes |
| Floor | FloorId {PK} | Unique identifier of floor | String | No |
| ShelfId | Shelf Id of floor | String | No |
| Status | Status of floor | Boolean | Yes |
| Cell | CellId {PK} | Unique identifier of cell. | String | No |
| FloorId | Floor Id of cell | String | No |
| Status | Status of cell | Boolean | Yes |
| CellRFID | RFID of cell | String | Yes |
| Package | PackageRFID {PK} | Unique identifier of package | String | No |
| Description | Description of package | String | Yes |
| CellId | Cell Id of package | String | Yes |
| ProductId | Product Id of package | String | Yes |
| Date | Creation Date of package | Date | Yes |
| Box | BoxRFID {PK} | Unique identifier of box. | String | No |
| PackageRFID | Package RFID of box | String | No |
| ProductId | Product Id of box | String | No |
| Date | Date of box | Date | Yes |
| Status | Status of box | Boolean | Yes |
| StandardShelfSize | StandardShelfId {PK} | Unique identifier of shelf size | Integer | No |
| StandardFloor | Standard number of floor | Integer | No |
| StandardCell | Standard number of cell per floor | Integer | No |
| ConstantTable | WeightedDate | Weight number of Date in Algorithm | Float | No |
| WeightedHeight | Weight number of Height in Algorithm | Float | No |
| SourceShelfId | Destination ShelfId of Date in Algorithm | String | No |
| ColumnDistance | Distance between two column of shelves | Integer | No |
| RowDistance | Distance between two row of shelves | Integer | No |

Table 149 Database Data Dictionary

## Test Plan

### Feature to be tested

* **Guest**: Login
* **Admin**: Create User, Update User, Deactivate User
* **Stock-keeper**: Create Product, Update Product, Deactivate Product, Create Category, Update Category, Deactivate Category, Create Shelf, Update Shelf, Deactivate Shelf, Create Goods Receipt, Update Goods Receipt, Delete Invoice
* **Warehouse Staff**: Register Cell, Register Package and Box, Stock in Package, Stock out Box, Transfer Package, Transfer Box, Stocktake Inventory

### Features not to be tested

Logout, View Product, View Category, View Shelf, View Invoice, Create Report, View Stocktake History, Clear RFID Tag, Edit Stamdard Shelf Size, Edit Invoice Prefix, Update Stocktake Status

## System Testing Test Case

### Web System Test Cases

#### <Warehouse Accountant> Create Product

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| CR\_PRO\_1 | Create a new product successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Product List” screen and clicks on “Create Product” button | 1. Enter “Nhớt Castrol” into “Product Name”  2. Select “Banh xe” from the “Category” list  3. Enter “10” into “Quantity per Box”  4. Select “Yamaha” from the “Vendor” list  5. Click “Confirm” button | Product is created and system shows “Products” screen | N/A | Pass | 8/8/2019 |
| CR\_PRO\_2 | Create product with empty required data  field | Login with Warehouse Accountant role  Warehouse Accountant is at “Product List” screen and clicks on “Create Product” button | 1. Enter nothing into “Product Name”  2. Enter data into the remaining fields  3. Click “Confirm” button | System shows error message "This field is required.” | N/A | Pass | 8/8/2019 |
| CR\_PRO\_3 | Create product with an existing Product Name | Login with Warehouse Accountant role  Warehouse Accountant is at “Product List” screen and clicks on “Create Product” button  The system database already has the “Nhớt Castrol”” Product Name | 1. Enter “Nhớt Castrol” into “Product Name”  2. Enter data into the remaining fields  3. Click “Confirm” button | System shows error message “Product name already exists!” | N/A | Pass | 8/8/2019 |
| CR\_PRO\_4 | Create new products by “Importing Products File” feature but no file is chosen | Login with Warehouse Accountant role  Warehouse Accountant is at “Product List” screen and clicks on “Create Product” button | 1. Click “Import” button.  2. Choose no file  3. Click “Confirm” button. | System shows error message “File is not existed or fail to upload.” | N/A | Pass | 8/8/2019 |
| CR\_PRO\_5 | Create new product by “Importing Products File” feature successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Product List” screen and clicks on “Create Product” button | 1. Click “Import” button.  2. Pick a file with correct format  3. Click “Confirm” button | Product is added and system shows “Products” screen. | N/A |  | 8/8/2019 |

Table 157- <Test Case>Create Product

#### <Warehouse Accountant> Create Shelf

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| CR\_SH\_1 | Create a new shelf successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen and clicks on “Create Shelf” button | 1. Enter “A” into “Shelf ID”  2. Enter “80” into “Cell Height (cm)”  3. Enter “120” into “Cell Width (cm)”  4. Enter “140” into “Cell Length (cm)”  5. Enter “1” into “Coordinate X”  6. Enter “2” into “Coordinate Y”  7. Click “Confirm” button | Shelf is created and system shows “Shelf List” screen. | N/A | Pass | 8/8/2019 |
| CR\_SH\_2 | Create shelf with empty required data  field | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen and clicks on “Create Shelf” button | 1. Enter nothing into “Shelf Id”  2. Enter data into the remaining fields  3. Click “Confirm” button | System shows error message "This field is required.” | N/A | Pass | 8/8/2019 |
| CR\_SH\_3 | Create shelf with an existing Shelf ID | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen and click on “Create Shelf” button  The system database already has the “A” Shelf ID | 1. Enter “A” into “Shelf ID”  2. Enter data into the remaining fields  3. Click “Confirm” button | System shows error message “Shelf ID already exists!” | N/A | Pass | 8/8/2019 |

Table 160- <Test Case>Create Shelf

#### <Warehouse Accountant> Update Shelf

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| UP\_SH\_1 | Update shelf with an existing Shelf ID | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen and click on “Update” button  The system database already has the “B” Shelf ID | 1. Enter “B” into “Shelf ID”  2. Click “Confirm” button | System shows error message “Shelf ID already exists!” | N/A | Pass | 8/8/2019 |
| UP\_SH\_2 | Update shelf with the smaller number of floors / cells | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen and click on “Update” button  The current shelf has 4 floors | 1. Enter “2” into “Number of Floor”  2. Click “Confirm” button | System shows error message “Can’t decrease shelf size, please remove all packages!!!” | N/A | Pass | 8/8/2019 |

Table 161- <Test Case>Update Shelf

#### <Warehouse Accountant> Deactivate Shelf

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| DE\_SH\_1 | Deactivate a shelf successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen | 1. Choose a shelf  2. System shoes confirmation dialog: “Are you sure to deactivate shelf?”  3. Clicks “Confirm” button | Shelf is deactivated and system shows “Shelf List” screen | N/A | Pass | 8/8/2019 |
| DE\_SH \_2 | Deactivate a shelf while shelf still contains packages | Login with Warehouse Accountant role  Warehouse Accountant is at “Shelf List” screen | 1. Choose a shelf that is still containing packages | System shows error message “The current shelf has packages inside.” | N/A | Pass | 8/8/2019 |

Table 162- <Test Case>Deactivate Shelf

#### <Warehouse Accountant> Create Goods Receipt

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| CR\_GRI\_1 | Create a new goods receipt successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Create” button | 1. Click “Choose Product”  2. Add a product  3. Enter “5” into “Stock in Quantity”  4. Click “Confirm” button | Goods receipt is created and system shows “Good Receipt screen” screen | N/A | Pass | 8/8/2019 |
| CR\_GRI\_2 | Create a new goods receipt with no product | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Create” button | 1. Click “Choose Product”  2. Add none product | “Confirm” button is disable | N/A | Pass | 16/8/2019 |
| CR\_GRI\_3 | Create a new goods receipt with stockin quantity is 0 | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Create” button | 1. Click “Choose Product”  2. Add a product  3. Enter “0” into “Stock in Quantity”  4. Click “Confirm” button | System show error message: “number must between 1 and 9999” | N/A | Pass | 16/8/2019 |

Table 163- <Test Case>Create Good Receipt

#### <Warehouse Accountant> Update Goods Receipt

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | | Result | Test Date | |
| UP\_GRI\_1 | | Update stock in quantity of products in goods receipt successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Update” button | 1. Choose existing product on the list  2. Enter “6” into “Stock in Quantity”  3. Click “Confirm” button | Goods receipt is updated and system shows “Goods Receipt” screen | N/A | | Pass | 8/8/2019 | |
| UP\_GRI\_2 | | Update goods receipt by adding products successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Update” button | 1. Click “Choose Product”  2. Add a product  3. Enter “5” into “Stock in Quantity”  4. Click “Confirm” button | Goods receipt is updated and system shows “Goods Receipt” screen | N/A | | Pass | 8/8/2019 | |
| UP\_GRI\_3 | Update goods receipt with stockin quantity is 0 | | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Receipt” screen and clicks on “Update” button | 1. Enter “0” into “Stock in Quantity”  2. Click “Confirm” button | System show error message: “number must between 1 and 9999” | N/A | Pass | | | 16/8/2019 |

Table 164- <Test Case>Update Good Receipt

#### <Warehouse Accountant> Create Goods Issue

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| CR\_GIS\_1 | Create a new goods issue successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Create” button  Products have at least 1 box | 1. Click “Choose Product”  2. Add a product  3. Enter “1” into “Stock out Quantity”  4. Click “Confirm” button | Goods issue is created and system shows “Good Issue” screen | N/A | Pass | 8/8/2019 |
| CR\_GIS \_2 | Create a new goods issue with no product | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Create” button | 1. Click “Choose Product”  2. Add none product | “Confirm” button is disable | N/A | Pass | 16/8/2019 |
| CR\_GIS \_3 | Create a new goods issue with stock out quantity is larger than on-hand quantity | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Create” button  Products currently have 5 box | 1. Click “Choose Product”  2. Add a product  3. Enter “10” into “Stock out Quantity”  4. Click “Confirm” button | System show error message: “Number must between 1 and 5” | N/A | Pass | 16/8/2019 |

Table 165- <Test Case>Create Good Issue

#### <Warehouse Accountant> Update Goods Issue

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| UP\_GIS\_1 | Update stock out quantity of products in goods issue successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Update” button  Product have more than 6 boxes | 1. Choose existing product on the list  2. Enter “6” into “Stock out Quantity”  3. Click “Confirm” button | Goods issue is updated and system shows “Goods Issue” screen | N/A | Pass | 8/8/2019 |
| UP\_ GIS \_2 | Update goods issue by adding products successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Update” button  Product have more than 6 boxes | 1. Click “Choose Product”  2. Add a product  3. Enter “5” into “Stock out Quantity”  4. Click “Confirm” button | Goods issue is updated and system shows “Goods Issue” screen | N/A | Pass | 8/8/2019 |
| UP\_ GIS \_3 | Update goods issue with stock out quantity is larger than on-hand quantity | Login with Warehouse Accountant role  Warehouse Accountant is at “Goods Issue” screen and clicks on “Update” button  Products currently have 5 box | 1. Enter “10” into “Stock out Quantity”  2. Click “Confirm” button | System show error message: “number must between 1 and 5” | N/A | Pass | 16/8/2019 |

Table 166- <Test Case>Update Good Issue

#### <Warehouse Accountant> Delete Invoice

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| DE\_ GRI\_1 | Delete an invoice successfully | Login with Warehouse Accountant role  Warehouse Accountant is at “Receipts/Issues” screen | 1. Choose an invoice  2. System shoes confirmation dialog: “Are you sure to delete invoice?”  3. Clicks “Confirm” button | Invoice is deleted and system shows “Receipts/Issues” screen | N/A | Pass | 8/8/2019 |

Table 167- <Test Case> Delete Invoice

### Mobile Application Test Cases

#### <Guest> Login

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inner-test case dependence** | **Result** | **Test date** |
| ***LM\_1*** | Login successfully | Guest is at “Login” screen | 1. Enter username  “stockkeeper”  2. Enter password “123456”  3. Click “Log In” button | System shows the main screen for Stock-keeper. | N/A | Pass | 8/8/2019 |
| ***LM\_3*** | Login with invalid username and password | Guest is at “Login” screen | 1. Enter username “stockkeeper123”  2. Enter password “password”  3. Click “Log In” button | System shows error message “Invalid username or password. Please try again!” | N/A | Pass | 8/8/2019 |

Table 168- <Test Case>Mobile Login

#### <Stock-keeper> Register Cell

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| RGS\_CEL\_1 | Register cell with RFID tag successfully | Login with Stock-keeper role  Stock-keeper is at “Register Cell” screen | 1. Tap to choose Shelf, Floor and Cell  2. Tap “Scan” button to scan RFID  3. Tap “Save” button | System shows message “Register successfully!” | N/A | Pass | 8/8/2019 |
| RGS\_CEL\_2 | No RFID tag is scanned | Login with Stock-keeper role  Stock-keeper is at “Register Cell” screen | 1. Tap to choose Shelf, Floor and Cell  2. Tap “Save” button | System shows error message "Please scan a Cell’s RFID.” | N/A | Pass | 8/8/2019 |
| RGS\_CEL\_3 | Register cell with RFID tag which has already registered with another cell | Login with Stock-keeper role  Stock-keeper is at “Register Cell” screen | 1. Tap to choose Shelf, Floor and Cell  2. Tap “Scan” button to scan registered RFID  3. Tap “Save” button | System shows error message "This RFID tag is already registered with another cell.” | N/A | Pass | 8/8/2019 |

Table 169- <Test Case>Register Cell

#### <Stock-keeper> Register Package and Box

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| RGS\_PCK\_1 | Register package with RFID tag successfully | Login with Stock-keeper role  Stock-keeper is at “Register Package” screen | 1. Tap to select “Good Receipt”  2. Tap to search and select “Product Name”  3. Tap “Scan” button next to “Package RIFD” field to scan RFID tag for package  4. Tap “Scan” button next to “Box RIFD” field to scan RFID tag for boxes  5. Tap “Save” button | System shows message “Register successfully.” | N/A | Pass | 8/8/2019 |
| RGS\_PCK\_2 | No RFID tag for package is scanned | Login with Stock-keeper role  Stock-keeper is at “Register Package” screen | 1. Tap to select “Good Receipt”  2. Tap to search and select “Product Name”  3. Tap “Scan” button next to “Box RIFD” field to scan RFID tag for boxes  4. Tap “Save” button | System shows error message "Please scan a package RFID.” | N/A | Pass | 8/8/2019 |
| RGS\_PCK\_3 | No RFID tag for boxes is scanned | Login with Stock-keeper role  Stock-keeper is at “Register Package” screen | 1. Tap to select “Goods Receipt”  2. Tap to search and select Product Name  3. Tap “Scan” button next to “Package RIFD” field to scan RFID tag for package  4. Tap “Save” button | System shows error message "Please scan a product RFID.” | N/A | Pass | 8/8/2019 |
| RGS\_PCK\_５ | Scan package RFID that is already registered with another product | Login with Stock-keeper role  Stock-keeper is at “Register Package” screen | 1. Tap to select “Good Receipt”  2. Tap to search and select “Product Name” that is not in the receipt  3. Tap “Scan” button next to “Package RIFD” field to scan RFID tag for package but is already registered with another product  4. Tap “Scan” button next to “Box RIFD” field to scan RFID tag for boxes  5. Tap “Save” button | System shows error message “Package is already registered with another product.” | N/A | Pass | 8/8/2019 |

Table 170- <Test Case>Register Package and Box

#### <Stock-keeper> Stock in Package

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| SI\_PCK\_1 | Stock in package successfully | Login with Stock-keeper role  Stock-keeper is at “Stock In” screen | 1. Tap “Scan” button next to “Cell RIFD” field to scan RFID tag of cell  2. Tap “Scan” button next to “Package RIFD” field to scan RFID tag of package  3. Tap “Save” button | System shows message “Stock in successfully!” | N/A | Pass | 8/8/2019 |
| SI\_PCK\_2 | No RFID tag of cell is scanned | Login with Stock-keeper role  Stock-keeper is at “Stock In” screen | 1. Tap “Scan” button next to “Package RIFD” field to scan RFID tag of package  2. Tap “Save” button | System shows error message "Please scan a cell’s RFID.” | N/A | Pass | 8/8/2019 |
| SI\_PCK\_3 | No RFID tag of package is scanned | Login with Stock-keeper role  Stock-keeper is at “Stock In” screen | 1. Tap “Scan” button next to “Cell RIFD” field to scan RFID tag of cell  2. Tap “Save” button | System shows error message "Please scan a package RFID.” | N/A | Pass | 8/8/2019 |
| SI\_PCK\_4 | Stock in package into a cell contains package | Login with Stock-keeper role  Stock-keeper is at “Stock In” screen | 1. Tap “Scan” button next to “Cell RIFD” field to scan RFID tag of a full cell  2. Tap “Scan” button next to “Package RIFD” field to scan RFID tag of package  3. Tap “Save” button | System shows error message "Cell is not empty”. | N/A | Pass | 8/8/2019 |
| SI\_PCK\_5 | Stock in unregistered package | Login with Stock-keeper role  Stock-keeper is at “Stock In” screen | 1. Tap “Scan” button next to “Cell RIFD” field to scan RFID tag of cell  2. Tap “Scan” button next to “Package RIFD” field to scan RFID tag of an unregistered package  3. Tap “Save” button | System shows error message "Package is unregistered.” | N/A | Pass | 8/8/2019 |

Table 171- <Test Case>Stock in Package

#### <Stock-keeper> Stock out Box

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| SO\_B\_1 | Stock out box successfully | Login with Stock-keeper role  Stock-keeper is at “Stock Out” screen | 1. Tap to select “Goods Issue”  2. Tap “Scan” button to scan RFID tags of boxes  3. Tap “Save” button | System shows message “Stock out successfully!” | N/A | Pass | 8/8/2019 |
| SO\_B\_2 | No RFID tag of boxes is scanned | Login with Stock-keeper role  Stock-keeper is at “Stock Out” screen | 1. Tap to select “Goods Issue”  2. Tap “Save” button | System shows error message "Please scan a product’s RFID.” | N/A | Pass | 8/8/2019 |
| SO\_B\_3 | Stock out boxes that are not in the invoice | Login with Stock-keeper role  Stock-keeper is at “Stock Out” screen | 1. Tap to select “Goods Issue”  2. Tap “Scan” button to scan RFID tags of boxes that are not in the invoice  3. Tap “Save” button | System shows error message “This product is not in the invoice.” | N/A | Pass | 8/8/2019 |
| SO\_B\_4 | Stock out the quantity of boxes that is not the same as the quantity in the invoice | Login with Stock-keeper role  Stock-keeper is at “Stock Out” screen | 1. Tap to select “Goods Issue”  2. Tap “Scan” button to scan RFID tags of boxes that has smaller quantity than in the invoice  3. Tap “Save” button | System shows error message “Not enough product.” | N/A | Pass | 8/8/2019 |

Table 172- <Test Case>Stock out Box

#### <Stock-keeper> Transfer Package

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| TRS\_PCK\_1 | Transfer package successfully | Login with Stock-keeper role  Stock-keeper is at “Transfer Package” screen | 1. Tap “Scan” button to scan cell’s RFID  2. Tap “Scan” button to scan package’s RFID  3. Tap “Save” button | System shows message “Transfer successfully!” | N/A | Pass | 8/8/2019 |
| TRS\_PCK\_2 | No RFID tag of cell is scanned | Login with Stock-keeper role  Stock-keeper is at “Transfer Package” screen | 1. Tap “Scan” button to scan package’s RFID  2. Tap “Save” button | System shows error message "Please scan a cell’s RFID.” | N/A | Pass | 8/8/2019 |
| TRS\_PCK\_3 | No RFID tag of package is scanned | Login with Stock-keeper role  Stock-keeper is at “Transfer Package” screen | 1. Tap “Scan” button to scan cell’s RFID  2. Tap “Save” button | System shows error message "Please scan a package’s RFID.” | N/A | Pass | 8/8/2019 |

Table 173- <Test Case>Transfer Package

#### <Stock-keeper> Transfer Box

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| TRS\_B\_1 | Transfer box successfully | Login with Stock-keeper role  Stock-keeper is at “Transfer Box” screen | 1. Tap “Scan” button to scan package’s RFID  2. Tap “Scan” button to scan boxes’ RFID  3. Tap “Save” button | System shows message “Transfer successfully!” | N/A | Pass | 8/8/2019 |
| TRS\_B\_2 | No RFID tag of package is scanned | Login with Stock-keeper role  Stock-keeper is at “Transfer Box” screen | 1. Tap “Scan” button to scan boxes’ RFID  2. Tap “Save” button | System shows error message "Please scan a package’s RFID.” | N/A | Pass | 8/8/2019 |
| TRS\_B\_3 | No RFID tag of boxes is scanned | Login with Stock-keeper role  Stock-keeper is at “Transfer Box” screen | 1. Tap “Scan” button to scan package’s RFID  2. Tap “Save” button | System shows error message "Please scan a product’s RFID.” | N/A | Pass | 8/8/2019 |
| TRS\_B\_4 | Transfer box that is not the same product with the destination package | Login with Stock-keeper role  Stock-keeper is at “Transfer Box” screen | 1. Tap “Scan” button to scan package’s RFID  2. Tap “Scan” button to scan boxes’ RFID that are not the same product with the destination package  3. Tap “Save” button | System shows error message “This product does not the same product with the destination package.” | N/A | Pass | 8/8/2019 |

Table 174- <Test Case>Transfer Box

#### <Stock-keeper> Stocktake Inventory

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Precondition | Test case procedure | Expected output | Inner-test case dependence | Result | Test Date |
| STK\_INV\_1 | Stocktake Inventory Successfully | Login with Stock-keeper role  Stock-keeper is at “Stocktake Inventory” screen | 1. Tap to select Product name  2. Tap “Scan” button to scan boxes’ RFID that macth with chosed Product  3. Tap “Report” button | System shows message “Report successfully!” | N/A | Pass | 8/8/2019 |

Table 175- <Test Case>Stocktake Inventory

**All Test Cases could be found here:** https://drive.google.com/open?id=1MG9bSDlDIwt\_SLsthsxQmv5nVXjcW7vJ

# F. Software User’s Manual

## Installation Guide

### Setting up environment at server side

#### Hardware requirements

* **For server**

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirement** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps or more) |
| **Operation System** | XP, Vista, 7, 10, Window Server 2008 | 10, Window server 2008 |
| **Computer Processor** | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz) |
| **Computer memory** | 4GB RAM | 32 GB RAM or more |
| **Storage space** | 1GB | 5GB or more |

Table 176 - Hardware requirements for Server

* **For PC**

|  |  |  |
| --- | --- | --- |
| **PC** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| **Operating System** | Window 7 | Window 7 or more. |
| **Computer Processor** | Intel® Core i3 1.4GHz | Intel® Core i5 2.50GHz |
| **Computer Memory** | 1GB RAM | 2GB RAM or more |
| **Web Browser** | Chromes (v42 or higher) | Chrome latest stable version |

Table 177 - Hardware requirements for PC

* **For Mobile**

|  |  |  |
| --- | --- | --- |
| **Mobile** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | Wi-Fi (4 Mbps) | Wi-Fi (8 Mbps) |
| **Operating System** | Android 6.0 | Android 7.1.1 |
| **Mobile Processor** | SnapDragon 625 | SnapDragon 625 |
| **Mobile Memory** | 2GB RAM | 4GB RAM or more |

Table 178 - Hardware requirements for Mobile

#### Software requirements

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| **Operation System** | Window Server 2012 | Operating system and platform for development |
| **Environment** | .NET Framework, Java, C, C# | Specification for developing web application |
| **Modeling tool** | StarUML | Designing diagram |
| **IDE** | Android Studio 3.4.1, Visual Studio 2017, Arduino IDE, IntelliJ 2019.1.3 | Programming tools |
| **DBMS** | SQL Server 2017, Amazon Web Services | Creating & managing the database for system |
| **Source control** | Git on IDE (Sourcetree) | Controlling the source code |
| **Web browser** | Chrome 69 or above | Testing system on browser |

Table 179 – Software Requirement

### Deployment at server side

#### Prepare deployment package

* Build \*.jar file for web server and \*.apk file for android device

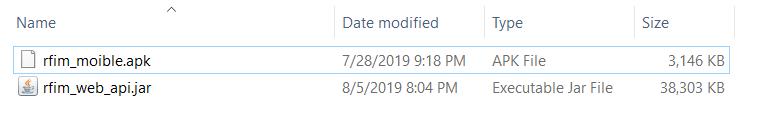


Figure 138 – Prepare deployment package

#### Configure Server before deploy

* Create a Ubuntu 18.04 **Virtual Private Server of ASW (Amazon Web Service)**

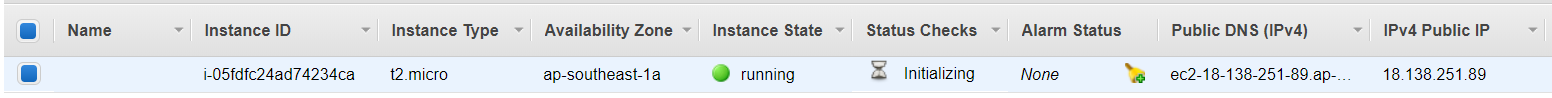


Figure 139 – Create Virtual Private Server

* Install open jdk using command

sudo apt-get update



sudo apt-get install openjdk-8-jdk



* Copy \*.jar file into server using FileZilla

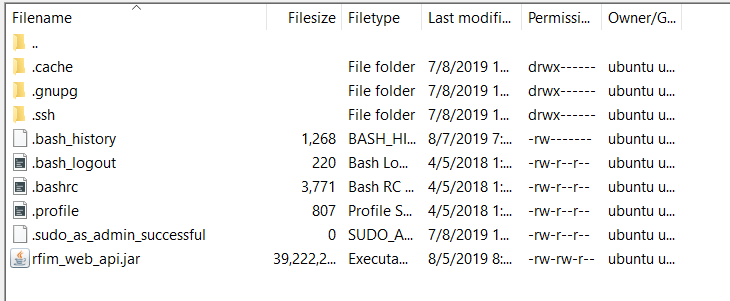


Figure 140 – Copy Jar file

* Run server by using command

java –jar \*.jar



### Setting up the environment at client side

* Copy installation file into android device

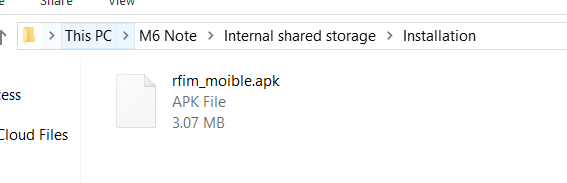


Figure 141 – Copy installation file to device

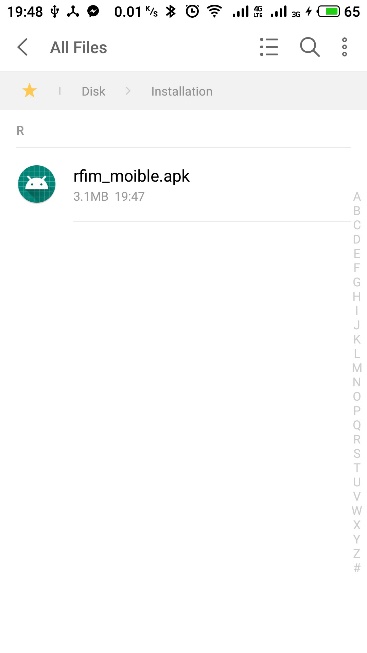
* Open file manage on android device

Figure 142 – Open file manage

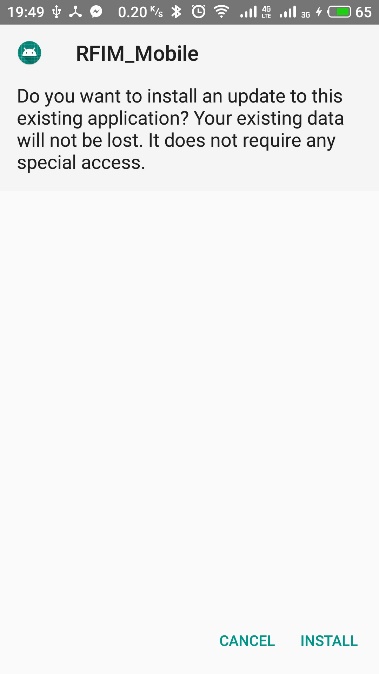
* Open installation file then install

Figure 143 - Install

## User Guide

### Web System

#### Create Product

Figure 161- <User Guide>Create Product [2]

Figure 162- <User Guide> Create Product [1]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Create Product” button |
| 2 | Fill in field “Product Id” |
| 3 | Fill in field “Product Name” |
| 4 | Click “Chọn tệp” to upload image |
| 5 | Fill in field “Description” |
| 6 | Fill in field “Weight (g)” |
| 7 | Fill in field “Height (cm)” |
| 8 | Fill in field “Width (cm)” |
| 9 | Fill in field “Length (cm)” |
| 10 | Fill in field “Quantity per Box” |
| 11 | Choose “Category” |
| 12 | Choose “Vendor” |
| 13 | Click on “Confirm” button |

Table 190- <User Guide>Create Product

#### Update Product

Figure 163- <User Guide> Update Product [1]

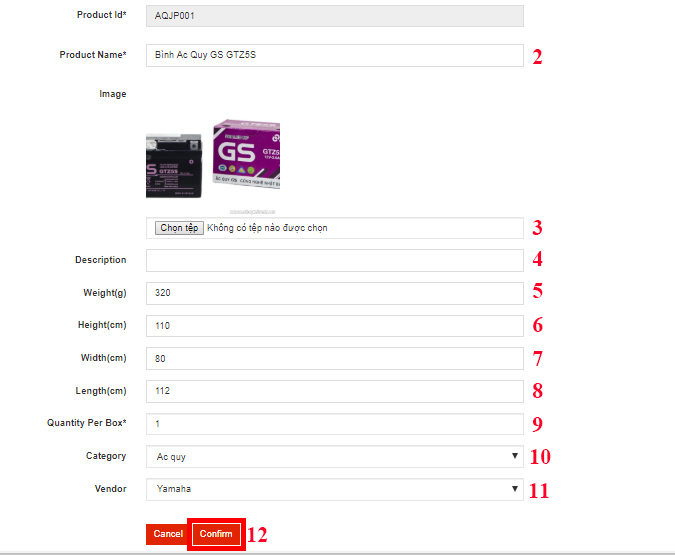
****

Figure 164- <User Guide> Update Product [2]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Edit” button |
| 2 | Fill in field “Product Name” |
| 3 | Click “Chọn tệp” to upload image |
| 4 | Fill in field “Description” |
| 5 | Fill in field “Weight (g)” |
| 6 | Fill in field “Height (cm)” |
| 7 | Fill in field “Width (cm)” |
| 8 | Fill in field “Length (cm)” |
| 9 | Fill in field “Quantity per Box” |
| 10 | Choose “Category” |
| 11 | Choose “Vendor” |
| 12 | Click on “Confirm” button |

Table 191- <User Guide>Update Product

#### Import Product (With Excel file)

Figure 167- <User Guide>Import Product [2]

Figure 168- <User Guide>Import Product [1]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Import Product” button |
| 2 | Click on “Chọn tệp” to choose file |
| 3 | Click on “Confirm” button |

Table 193- <User Guide>Import Product

#### Deactivate/Activate Shelf

Figure 174- <User Guide>Deactivate/Activate Shelf [1]

****

Figure 175- <User Guide>Deactivate/Activate Shelf [2]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Deactivate” button (after shelf is deactivated, click on “Activate” button if want to activate shelf again) |
| 2 | Click on “Deactive” button to confirm |

Table 197- <User Guide>Deactivate/Activate Shelf

#### Edit Standard Shelf Size

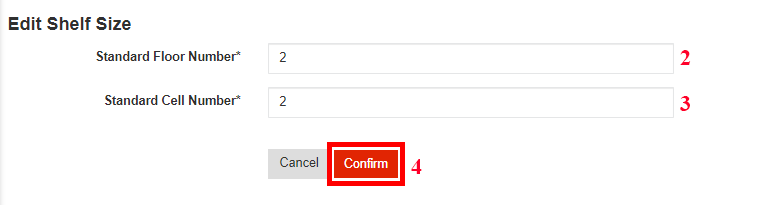
****

Figure 176- <User Guide>Edit Standard Shelf Size [1]

Figure 177- <User Guide>Edit Standard Shelf Size [2]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Standard Shelf Size” button |
| 2 | Fill in field “Standard Floor Number” |
| 3 | Fill in field “Standard Cell Number” |
| 4 | Click on “Confirm” button |

Table 198- <User Guide>Edit Standard Shelf Size

#### Create Invoices

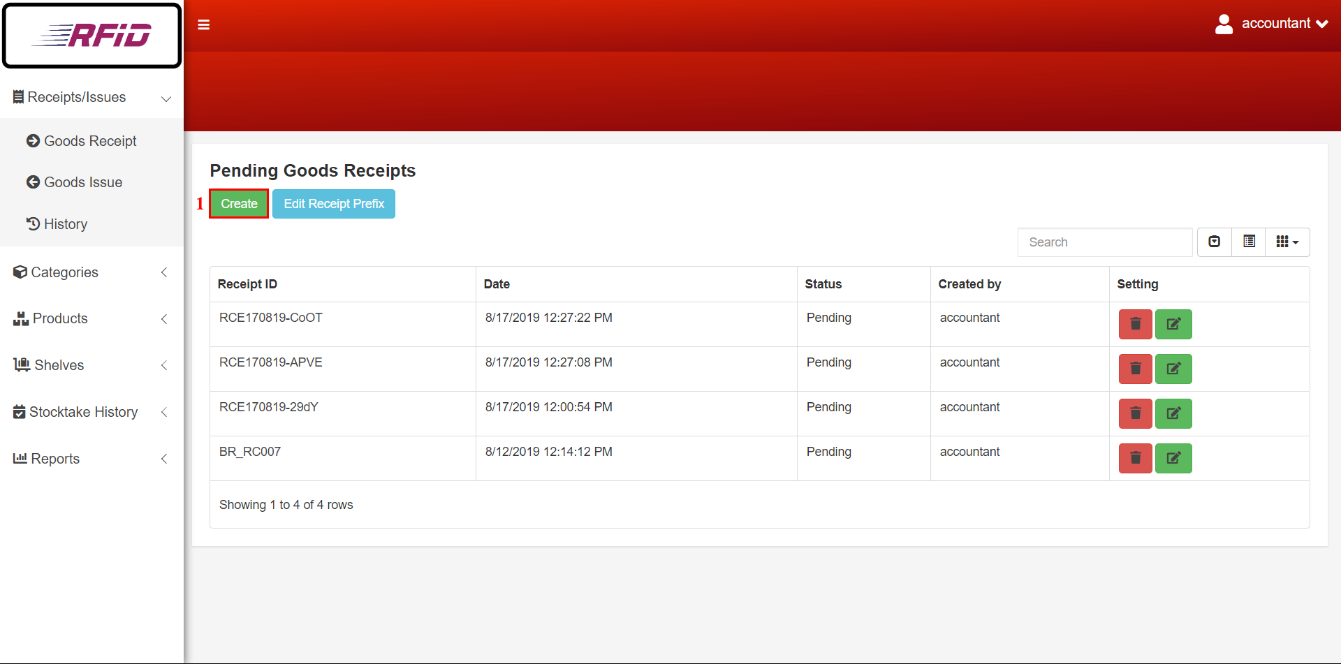
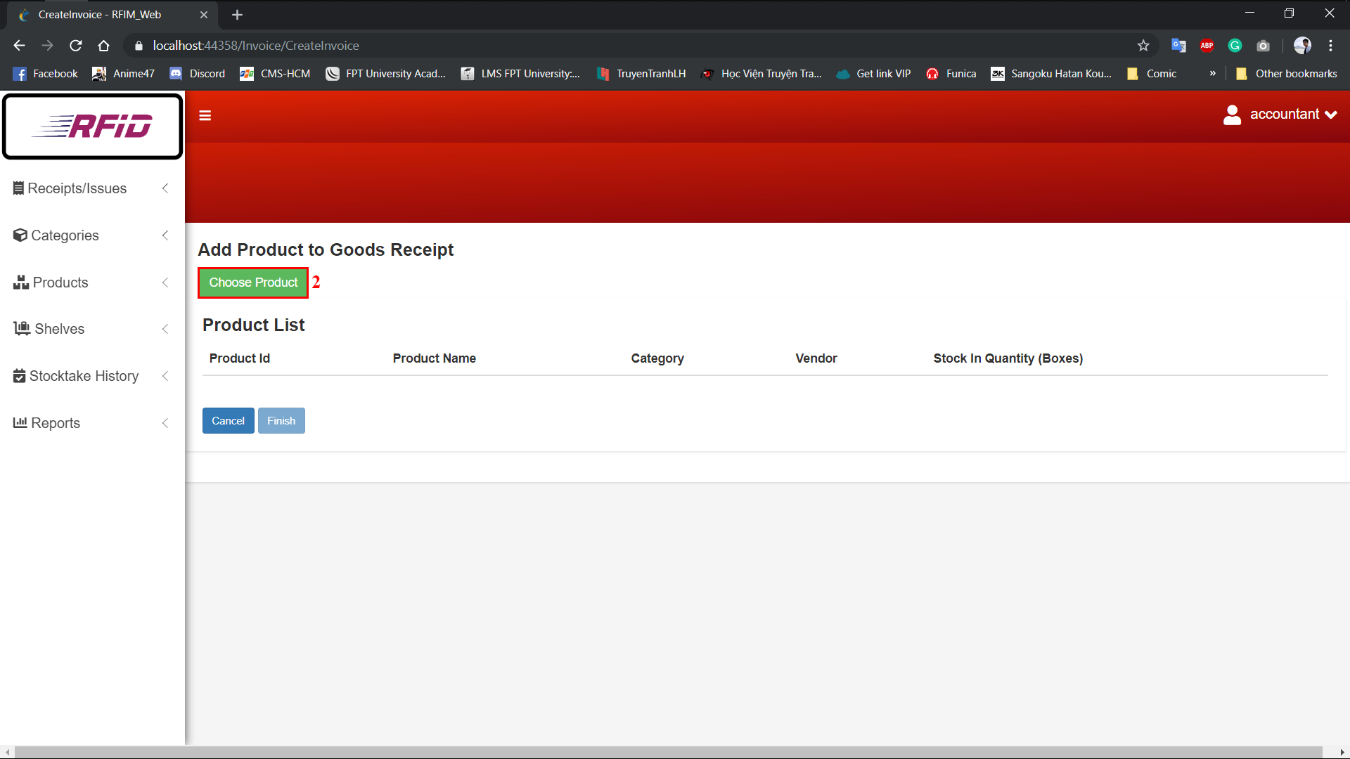
****

Figure 181- <User Guide>Create Invoice [2]

Figure 179- <User Guide>Create Invoice [1]

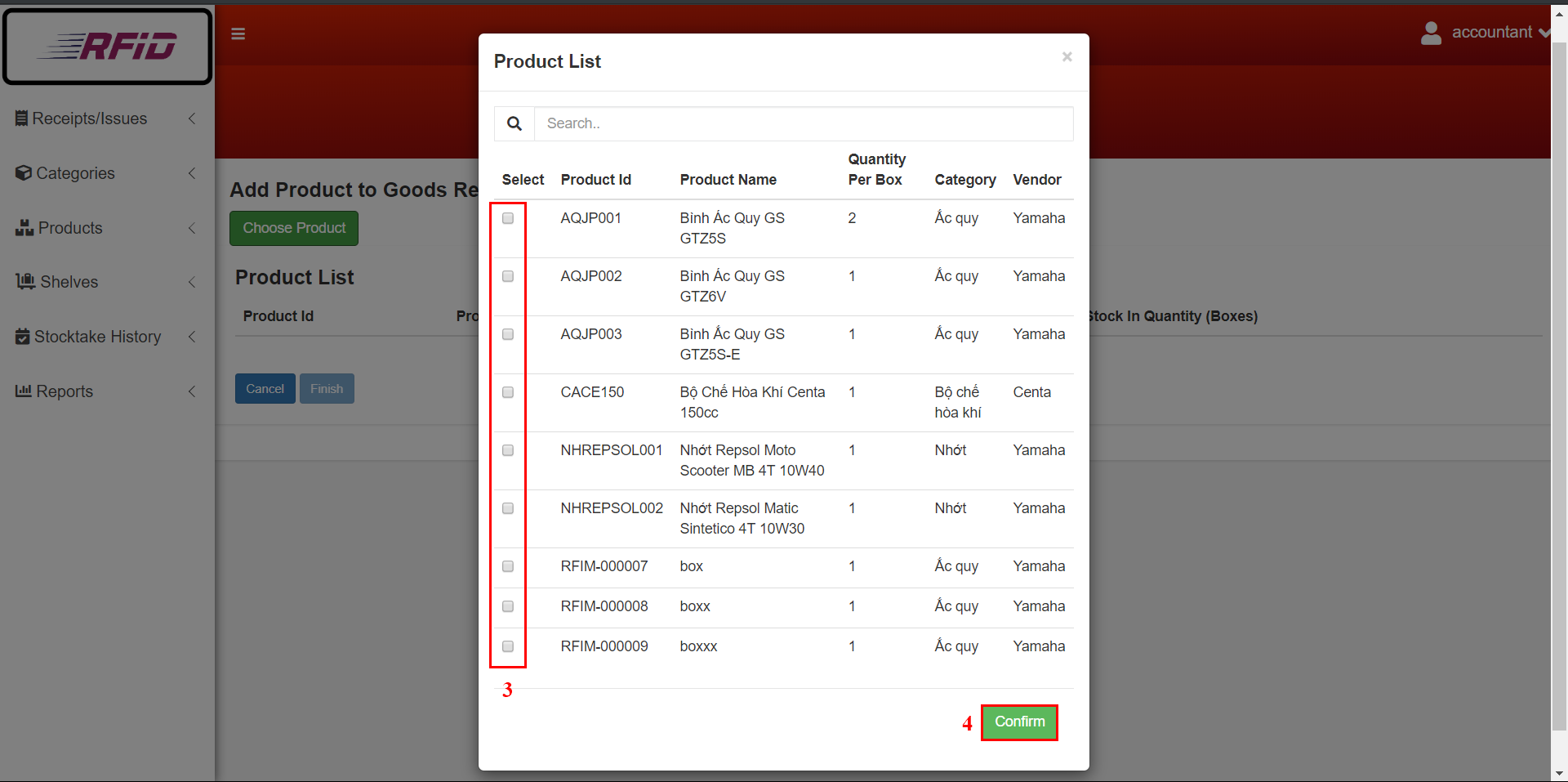
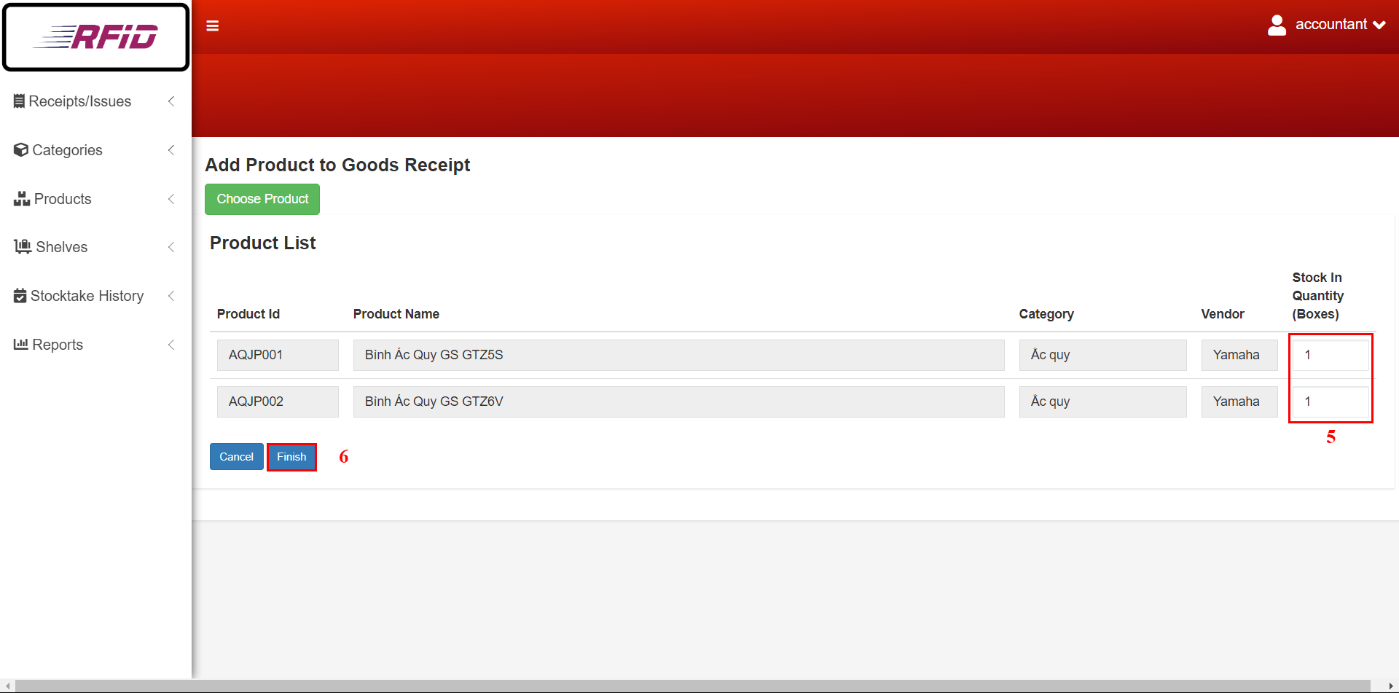
****

Figure 183- <User Guide>Create Invoice [4]

Figure 182- <User Guide>Create Invoice [3]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Create” button |
| 2 | Click on “Choose Product” button |
| 3 | Click on checkbox(es) to select product(s) |
| 4 | Click on “Confirm” button |
| 5 | Fill in field “Stock In Quantity (boxes)” for Goods Receipt or “Stock out Quantity (boxes)” for Goods Issue |
| 6 | Click on “Finish” button |

Table 200- <User Guide>Create Invoice

#### Update Invoice

Figure 184- <User Guide>Update Invoice [1]

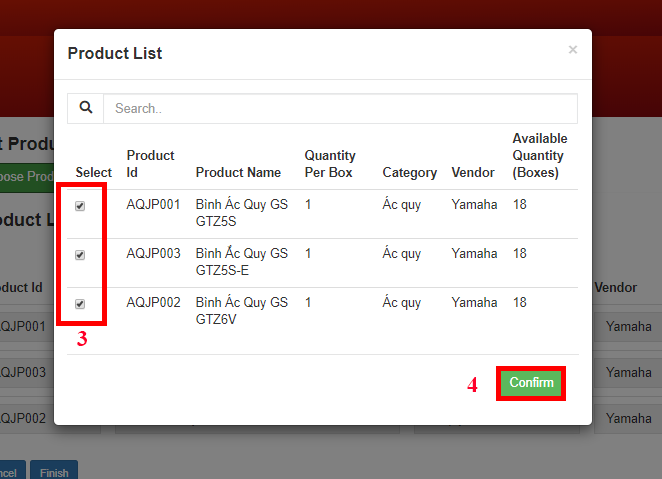
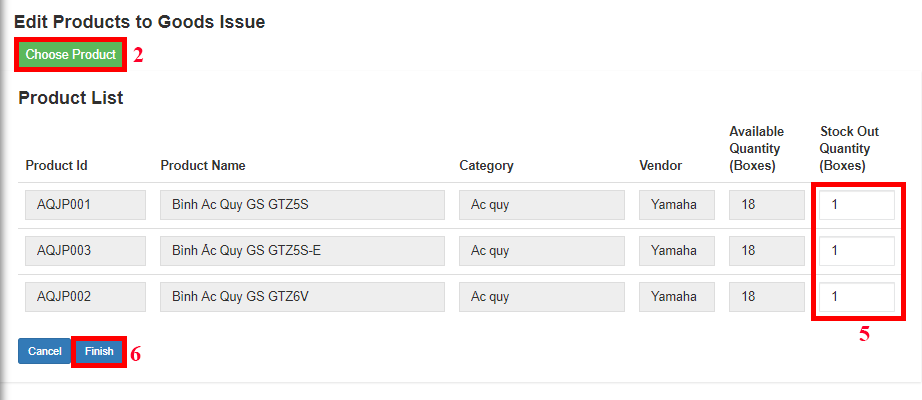
****

Figure 186- <User Guide>Update Invoice [3]

Figure 185- <User Guide>Update Invoice [2]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Edit” button |
| 2 | Click on “Choose Products” button |
| 3 | Click on checkbox(es) to select/unselect product(s) |
| 4 | Click on “Confirm” button |
| 5 | Fill in field “Stock In Quantity (boxes)” for Goods Receipt or “Stock out Quantity (boxes)” for Goods Issue |
| 6 | Click on “Finish” button |

Table 201- <User Guide>Update Invoice

#### Delete Invoice

Figure 187- <User Guide>Delete Invoice [1]

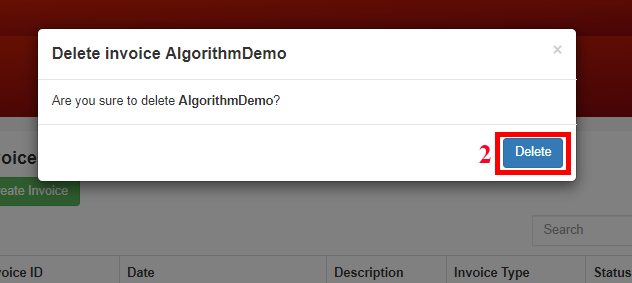
****

Figure 188- <User Guide>Delete Invoice [2]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Delete” button |
| 2 | Click on “Delete” button to confirm |

Table 202- <User Guide>Delete Invoice

#### Create Report

Figure 189- <User Guide>Create Report

|  |  |
| --- | --- |
| Step | Description |
| 1 | Choose “Goods Recept/Issue” to choose type of invoice that want to create |
| 2 | Click on “Product” button to create product report |
| 3 | Click on “Category” button to create category report |
| 4 | Click on “Vendor” button to create vendor report |
| 5 | Click on “Stocktake” button to create stocktake report |

Table 203- <User Guide>Create Report

#### Update Stocktake Status

Figure 191- <User Guide>Update Stocktake Status [2]

Figure 192- <User Guide>Update Stocktake Status [1]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Confirmation” button |
| 2 | Click on “Update” button to confirm |

Table 205- <User Guide>Update Stocktake Status

#### Edit Invoice Prefix

Figure 177- <User Guide>Edit Prefix [2]

Figure 176- <User Guide>Edit Prefix [1]

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click on “Edit Prefix” button |
| 2 | Fill in field “Prefix” |
| 3 | Click on “Confirm” button |

Table 206- <User Guide>Edit Prefix

### Mobile Application

#### Register Cell (Shelf)

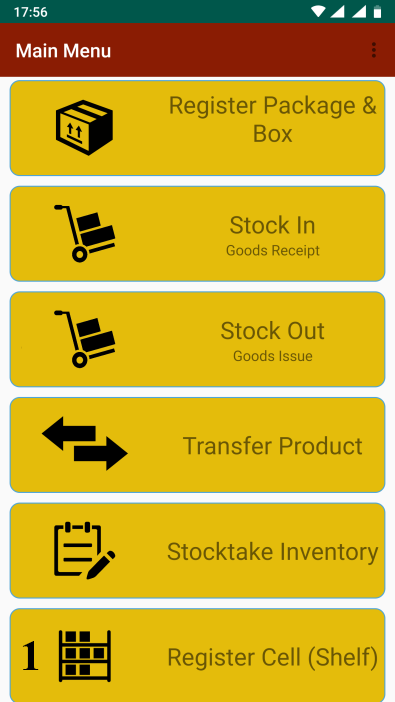
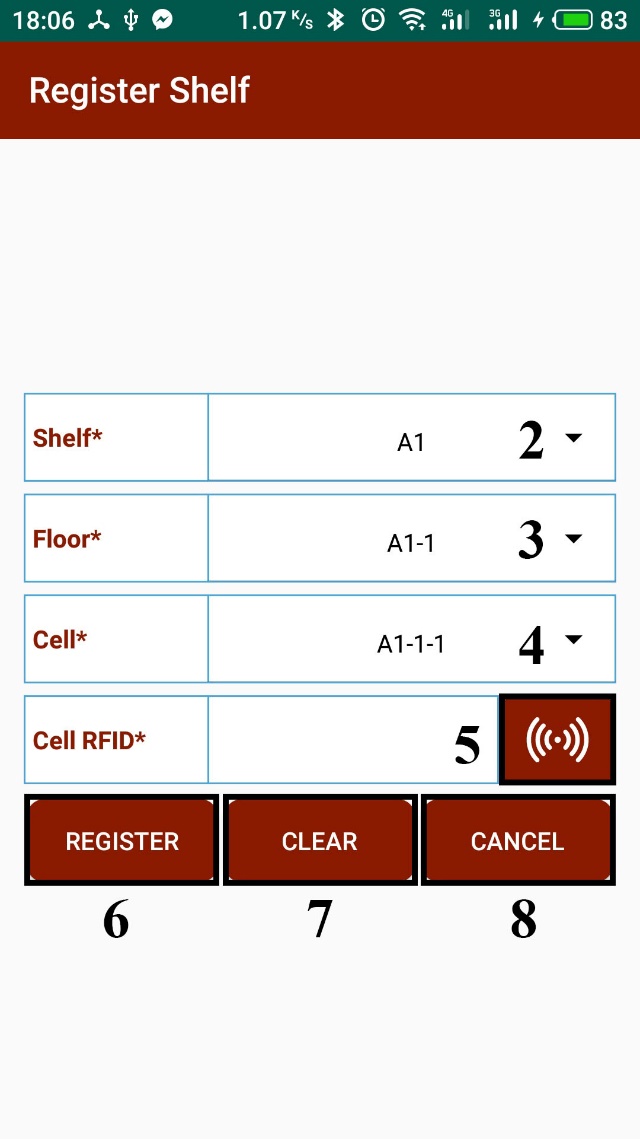
****

Figure 194- <User Guide>Register Cell

Figure 195- <User Guide>Menu

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Register Shelf (Cell)” feature |
| 2 | Choose “Shelf” |
| 3 | Choose “Floor” |
| 4 | Choose “Cell” |
| 5 | Tap on “Scan” button to scan “Cell RFID” |
| 6 | Tap on “Register” button to register |
| 7 | Tap on “Clear” button to reset data |
| 8 | Tap on “Cancel” button to exit screen |

Table 208- <User Guide>Register Cell

#### Register Package and Box

Figure 198- <User Guide>Good Receipt Info

Figure 197- <User Guide>Register Package and Box

Figure 196- <User Guide> Menu

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Register Package and Box” feature |
| 2 | Choose “Goods Receipt” |
| 3 | Tap on “Info” button to see goods receipt’s details |
| 4 | Tap on “X” button to exit |
| 5 | Choose “Product Name” |
| 6 | Tap on “Scan” button to scan “Package RFID” |
| 7 | Tap on “Scan” button to scan “Box RFID” |
| 8 | Tap on “Save” button to register |
| 9 | Tap on “Clear” button to reset data |
| 10 | Tap on “Cancel” button to exit screen |

Table 209- <User Guide>Register Package and Box

#### Stock In

Figure 199- <User Guide>Stock In

Figure 200- <User Guide>Menu

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Stock In” feature |
| 2 | Tap on “Scan” button to scan “Cell RFID” |
| 3 | Tap on “Scan” button to scan “Package RFID” |
| 4 | Tap on “Save” button to stock in |
| 5 | Tap on “Clear” button to reset data |
| 6 | Tap on “Cancel” button to exit screen |

Table 210- <User Guide> Stock In

#### Stock Out

Figure 201- <User Guide>Stock Out

Figure 202- <User Guide>Menu

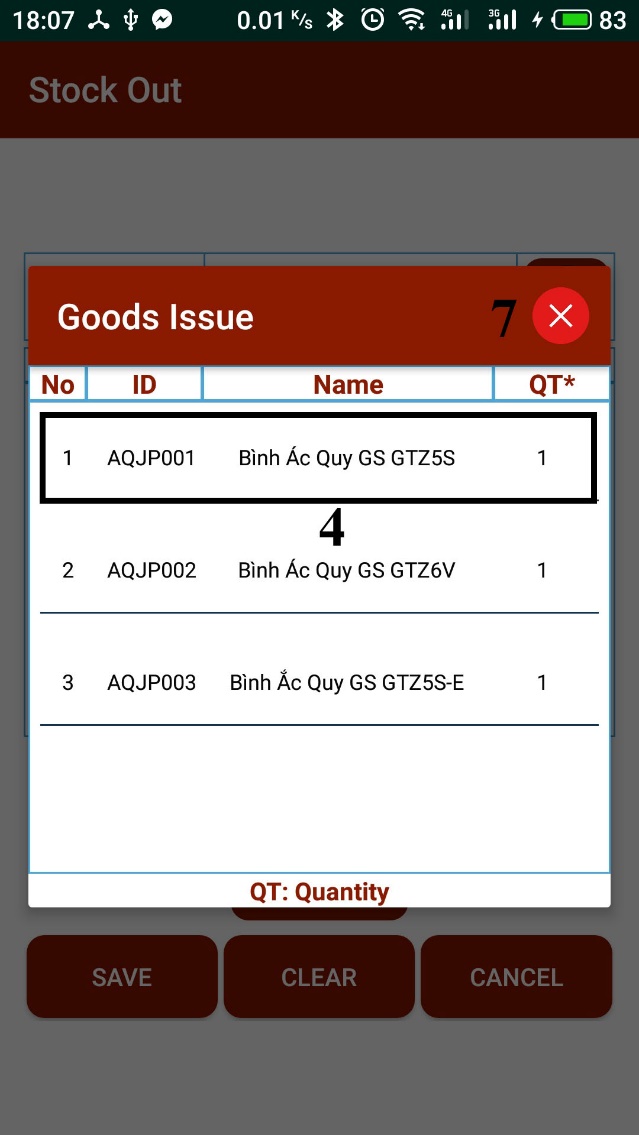
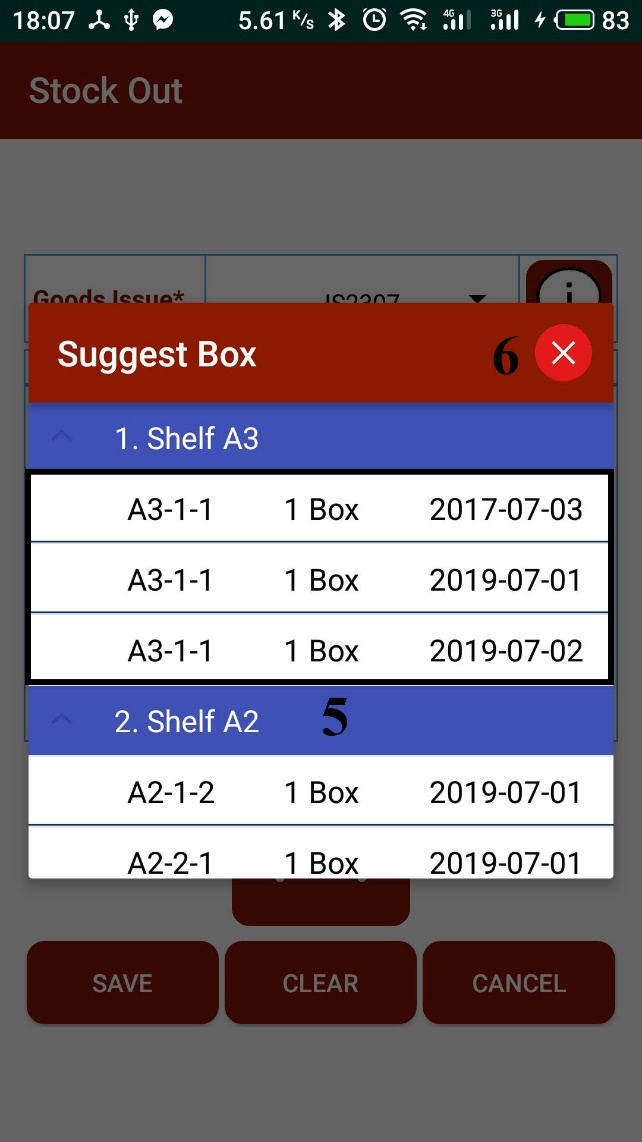
****

Figure 203- <User Guide>Goods Issue

Figure 204- <User Guide>Suggest Box

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Stock Out” feature |
| 2 | Choose “Issue Invoice” |
| 3 | Tap on “Info” button to see goods issue’s details |
| 4 | Tap on a product to see “Suggest Box” |
| 5 | Tap on a box that want to be stocked out |
| 6 | Tap on “X” button to exit “Suggest Box” screen |
| 7 | Tap on “X” button to exit “Issue Invoice” detail screen |
| 8 | Tap on “Scan” button to scan products |
| 9 | Tap on “Save” button to stock out |
| 10 | Tap on “Clear” button to reset data |
| 11 | Tap on “Cancel” button to exit screen |

Table 211- <User Guide>Stock Out

#### Transfer Box

Figure 205- <User Guide>Menu

Figure 206- <User Guide>Transfer Box

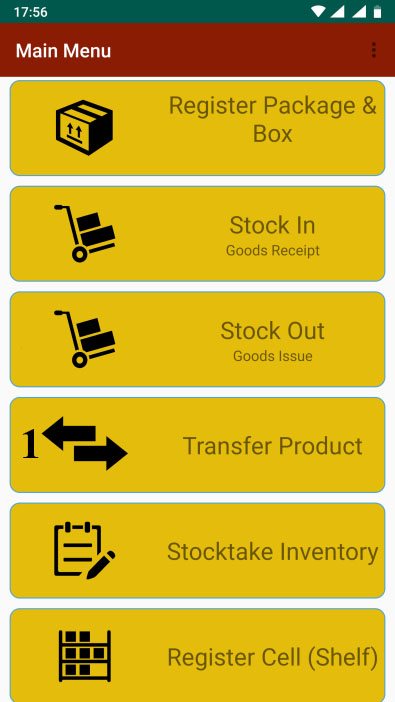
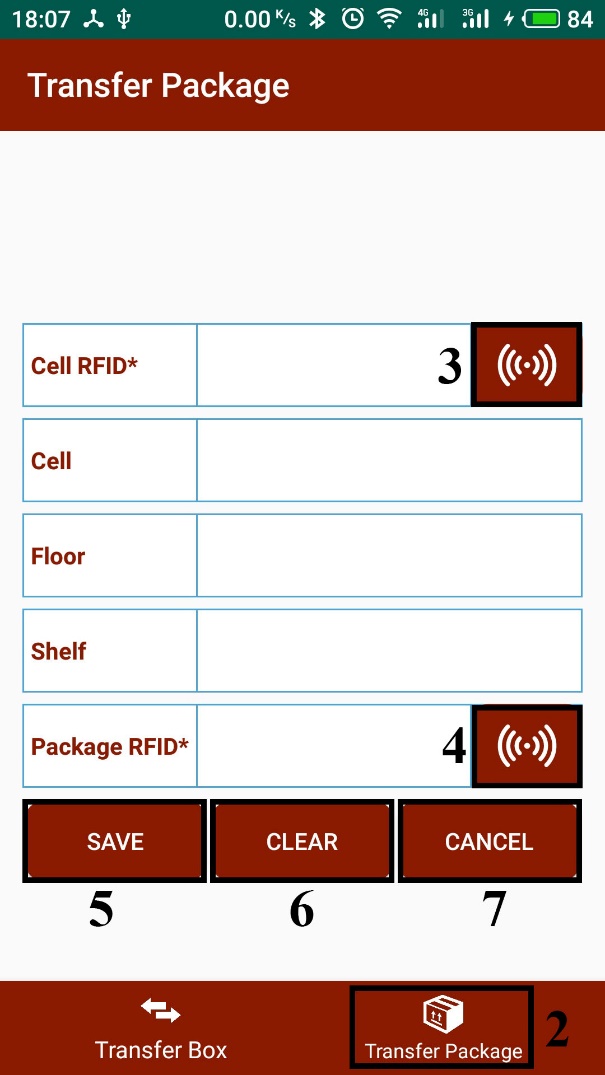
|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Transfer Product” feature |
| 2 | Tap on “Transfer Box” tab |
| 3 | Tap on “Scan” button to scan package RFID |
| 4 | Tap on “Scan” button to scan box RFID |
| 5 | Tap on “Save” button to transfer |
| 6 | Tap on “Clear” button to reset data |
| 7 | Tap on “Cancel” button to exit screen |

Table 212- <User Guide>Transfer Box

#### Transfer Package

Figure 207- <User Guide>Menu

Figure 208- <User Guide>Transfer Package

****

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Transfer Product” feature |
| 2 | Tap on “Transfer Package” tab |
| 3 | Tap on “Scan” button to scan “Cell RFID” |
| 4 | Tap on “Scan” button to scan “Package RFID” |
| 5 | Tap on “Save” button to transfer |
| 6 | Tap on “Clear” button to reset data |
| 7 | Tap on “Cancel” button to exit screen |

Table 213- <User Guide>Transfer Package

#### Stocktake Inventory

Figure 210- <User Guide>Stocktake Inventory

Figure 209- <User Guide>Menu

|  |  |
| --- | --- |
| Step | Description |
| 1 | Tap on “Stocktake Inventory” feature |
| 2 | Choose “Product Name” |
| 3 | Tap on “Scan” button to scan product RFID to check quantity |
| 4 | Tap on “Report” button to report |
| 5 | Tap on “Clear” button to reset data |
| 6 | Tap on “Cancel” button to exit screen |

Table 214- <User Guide>Stocktake Inventory

**All User Guide could be found here:** https://drive.google.com/open?id=1ZaObq4ZIR0Hnr82ru\_N6jxvg-bLga4BX

## 取扱説明書

ウェブシステム

#### 製品を作成する

Figure 161- <取扱説明書>製品を作成する[2]

Figure 162- <取扱説明書>製品を作成する[1]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Create Product」ボタンをクリックします |
| 2 | 「Product Id」フィールドに記入します |
| 3 | 「Product Name」フィールドに記入します |
| 4 | 「Chọn tệp」をクリックして画像をアップロードします |
| 5 | 「Description」フィールドに記入します |
| 6 | 「Weight (g)」フィールドに記入します |
| 7 | 「Height (cm)」フィールドに記入します |
| 8 | 「Width(cm)」フィールドに記入します |
| 9 | 「Length(cm)」フィールドに記入します |
| 10 | 「Quantity per Box」フィールドに記入します |
| 11 | 「Category」を選択します |
| 12 | 「Vendor」を選択します |
| 13 | 「Confirm」ボタンをクリックします |

Table 225- <取扱説明書>製品を作成する

#### 製品を更新する

Figure 163- <取扱説明書>製品を更新する[1]

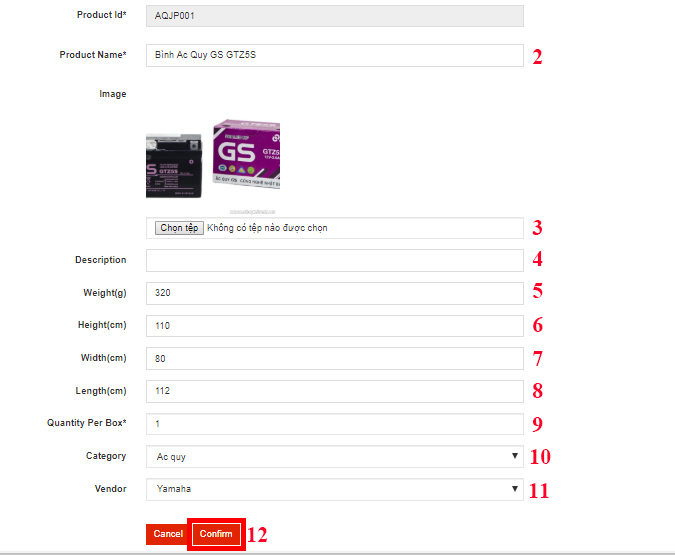
****

Figure 164- <取扱説明書>製品を更新する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Edit」ボタンをクリックします |
| 2 | 「Product Name」フィールドに記入します |
| 3 | 「Chọn tệp」をクリックして画像をアップロードします |
| 4 | 「Description」フィールドに記入します |
| 5 | 「Weight (g)」フィールドに記入します |
| 6 | 「Height (cm)」フィールドに記入します |
| 7 | 「Width(cm)」フィールドに記入します |
| 8 | 「Length(cm)」フィールドに記入します |
| 9 | 「Quantity per Box」フィールドに記入します |
| 10 | 「Category」を選択します |
| 11 | 「Vendor」を選択します |
| 12 | 「Confirm」ボタンをクリックします |

Table 226- <取扱説明書>製品を更新する

#### 商品ファイルをインポートする

Figure 167- <取扱説明書>商品ファイルをインポートする[2]

Figure 168- <取扱説明書>商品ファイルをインポートする[1]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Import Product」ボタンをクリックします |
| 2 | ファイルを選択するには、「Chọn tệp」をクリックします |
| 3 | 「Confirm」ボタンをクリックします |

Table 228- <取扱説明書>商品ファイルをインポートする

#### 棚を無効化/有効化する

Figure 174- <取扱説明書>棚を無効化/有効化する[1]

****

Figure 175- <取扱説明書>棚を無効化/有効化する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | [Deactivate]ボタンをクリックします（棚を無効にした後、もう一度有効にする場合は[Activate]ボタンをクリックします） |
| 2 | 確認するには「Deactive」ボタンをクリックしてください |

Table 232- <取扱説明書>棚を無効化/有効化する

#### 標準棚サイズを編集する

Figure 176- <取扱説明書>標準棚サイズを編集する[1]

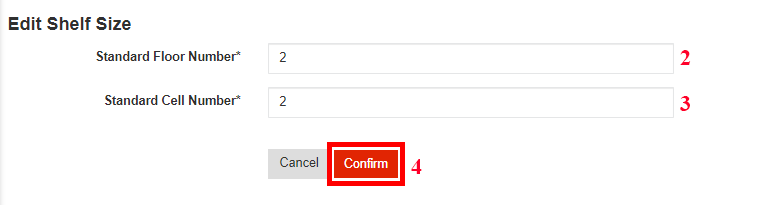
****

Figure 177- <取扱説明書>標準棚サイズを編集する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Standard Shelf Size」ボタンをクリックします |
| 2 | 「Standard Floor Number」フィールドに記入します |
| 3 | 「Standard Cell Number」フィールドに記入します |
| 4 | 「Confirm」ボタンをクリックします |

Table 233- <取扱説明書>標準棚サイズを編集する

#### 請求書を作成する

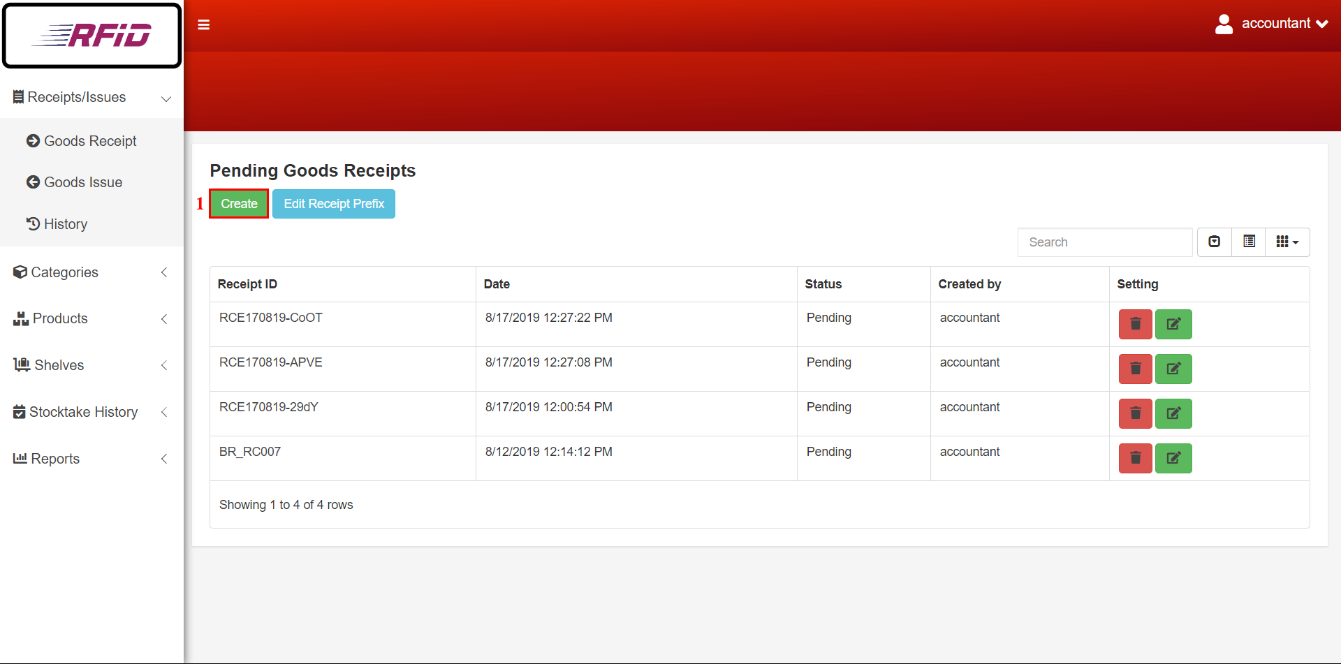
****

Figure 179- <取扱説明書>請求書を作成する[1]

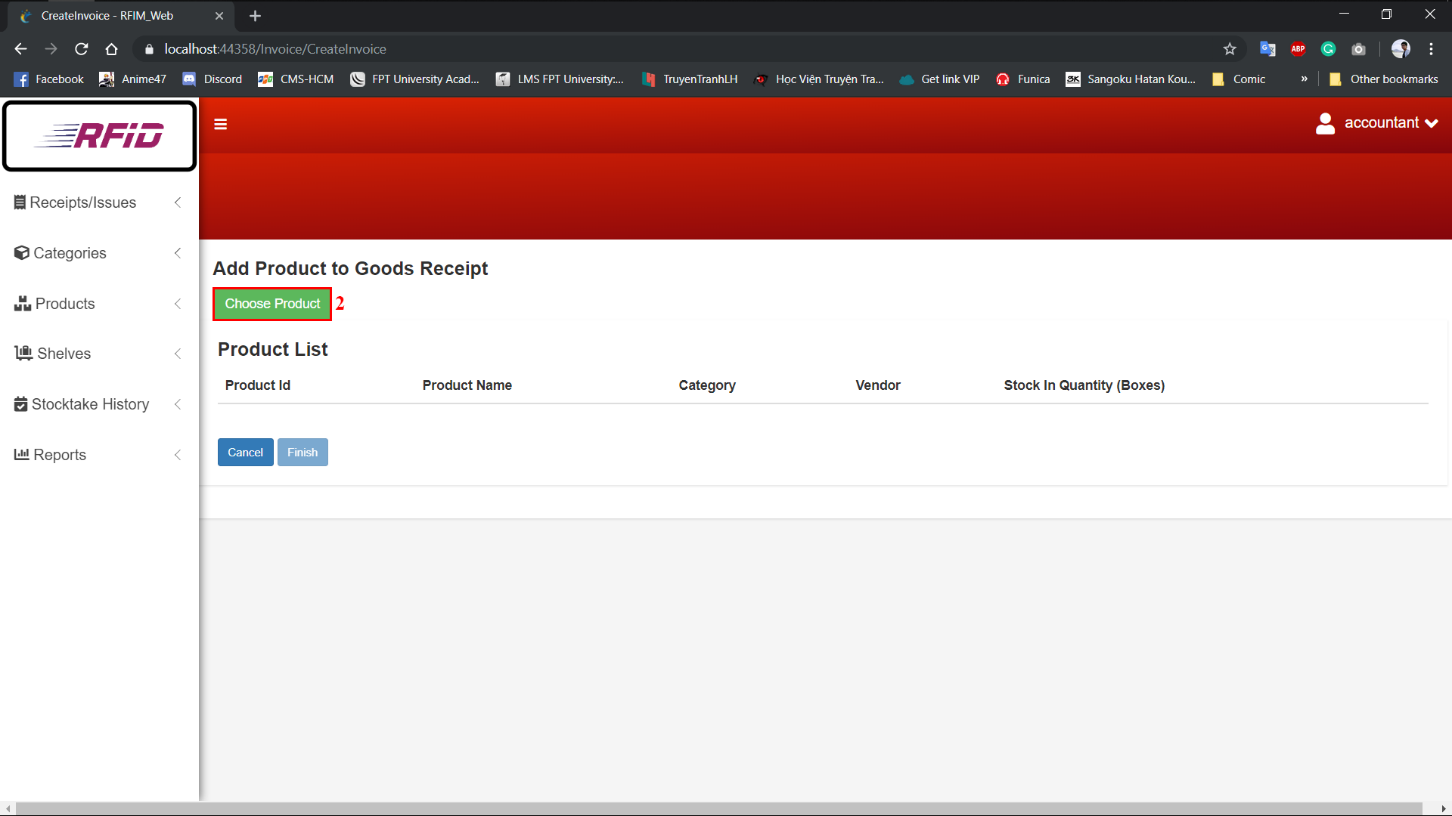
****

Figure 181- <取扱説明書>請求書を作成する[2]

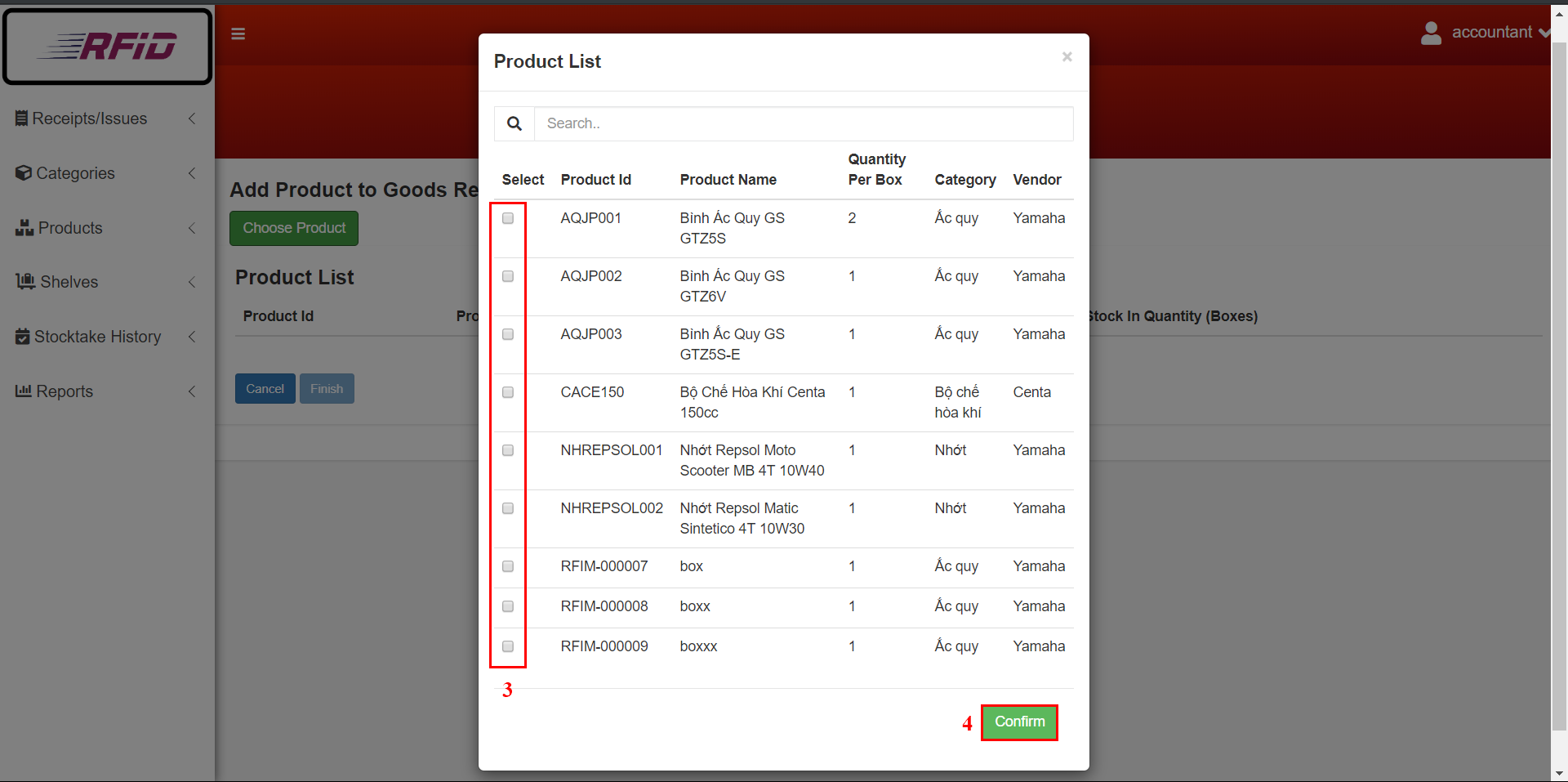
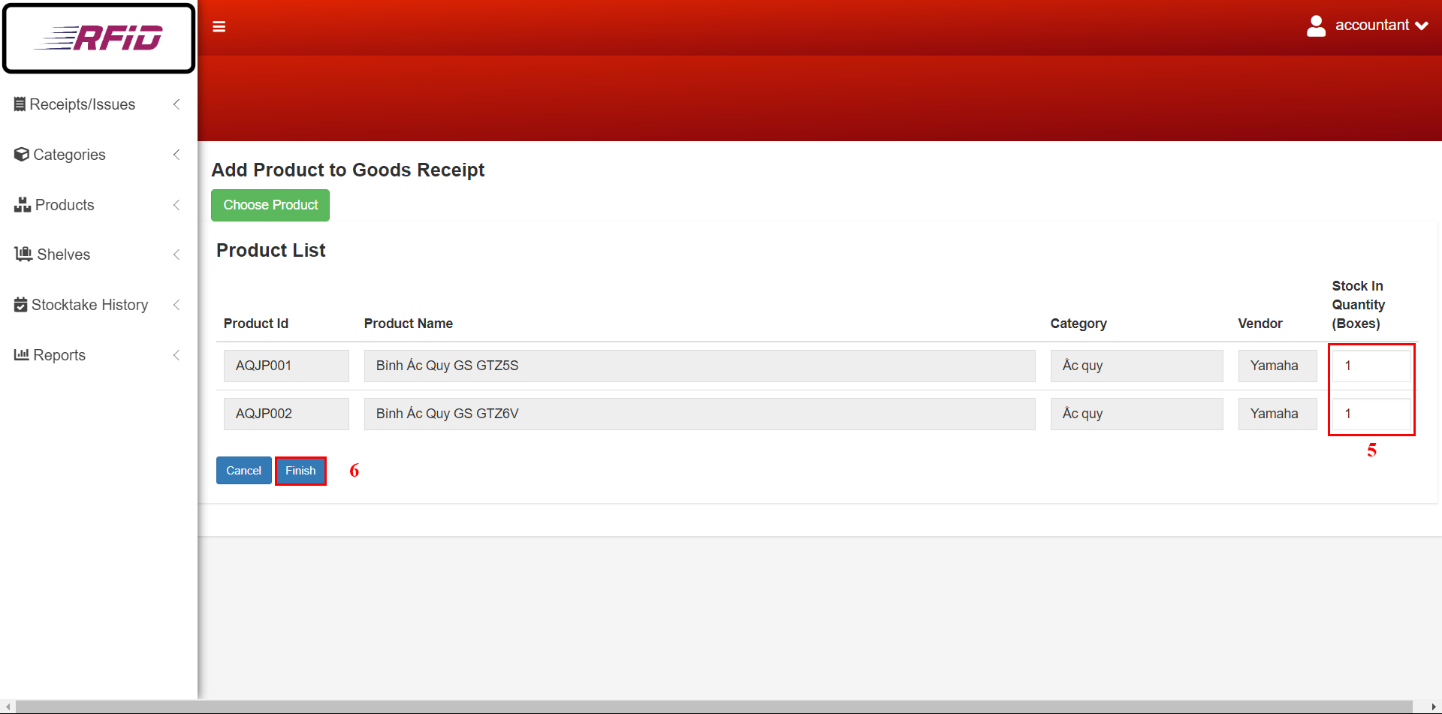
****

Figure 183- <取扱説明書>請求書を作成する[5]

Figure 182- <取扱説明書>請求書を作成する[3]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Create」ボタンをクリックします |
| 2 | 「Choose Products」ボタンをクリックします |
| 3 | チェックボックスをクリックして製品を選択します |
| 4 | 「Confirm」ボタンをクリックします |
| 5 | 入庫の場合は項目「Stock in Quantity (Boxes)」または出庫の場合は「Stock out Quantity (Boxes)」フィールドに入力します |
| 6 | 「Finish」ボタンをクリックします |

Table 235- <取扱説明書>請求書を作成する

#### 請求書を更新する

Figure 184- <取扱説明書>請求書を更新する[1]

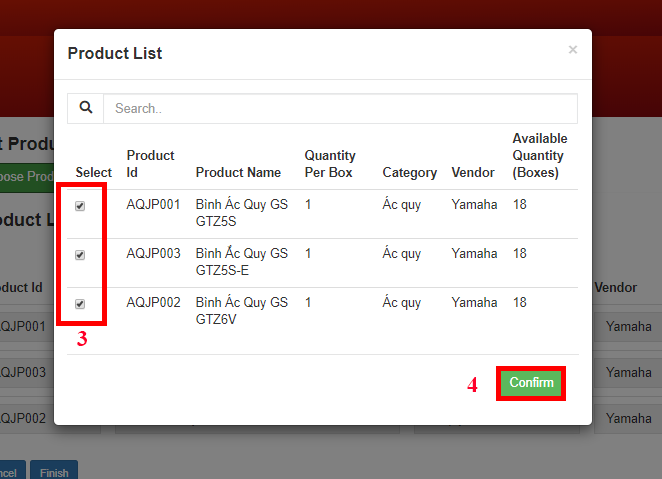
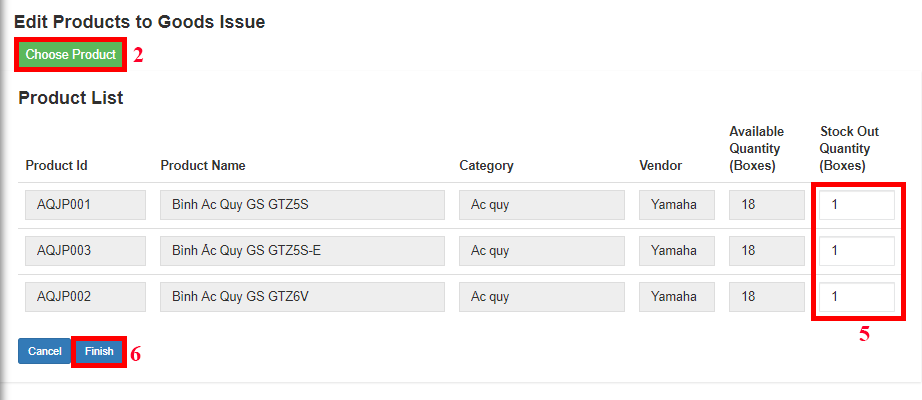
****

Figure 186- <取扱説明書>請求書を更新する[3]

Figure 185- <取扱説明書>請求書を更新する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Edit」ボタンをクリックします |
| 2 | 「Choose Products」ボタンをクリックします |
| 3 | チェックボックスをクリック/クリック解除して製品を選択/選択解除します |
| 4 | 「Confirm」ボタンをクリックします |
| 5 | 入庫の場合は項目「Stock in Quantity (Boxes)」または出庫の場合は「Stock out Quantity (Boxes)」フィールドに入力します |
| 6 | 「Finish」ボタンをクリックします |

Table 236- <取扱説明書>請求書を更新する

#### 請求書を削除する

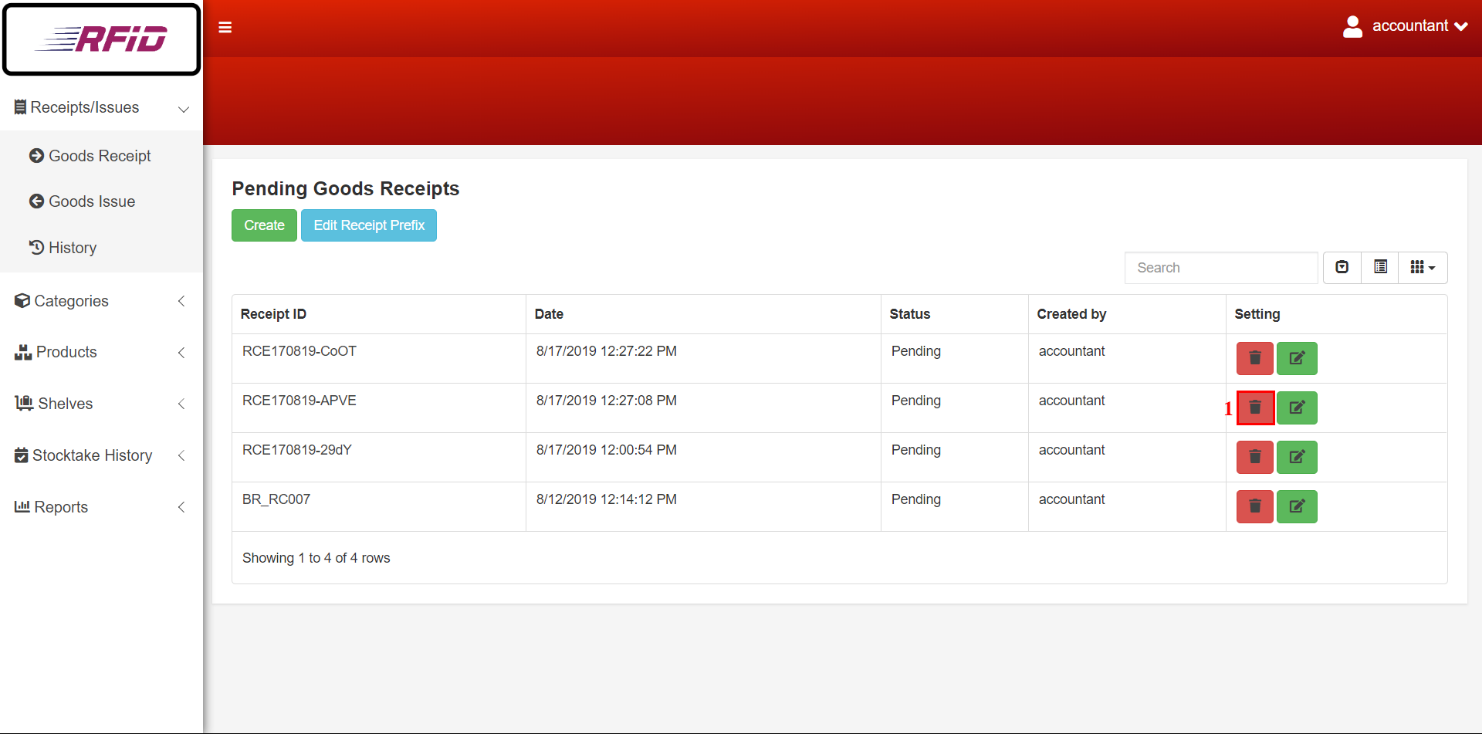
****

Figure 187- <取扱説明書>請求書を削除する[1]

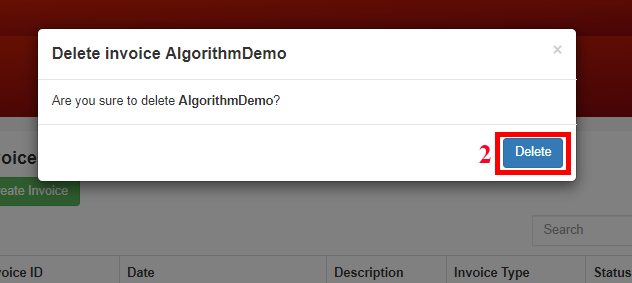
****

Figure 188- <取扱説明書>請求書を削除する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Delete」ボタンをクリックします |
| 2 | 「Delete」ボタンをクリックして確定します |

Table 237- <取扱説明書>請求書を削除する

#### レポートを作成する

Figure 189- <取扱説明書>レポートを作成する

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 作成する請求書のタイプを選択するには、「Goods Receipt/Issue」を選択します |
| 2 | 製品レポートを作成するには、[Product]ボタンをクリックします |
| 3 | カテゴリーレポートを作成するには、[Category]ボタンをクリックします |
| 4 | 仕入先レポートを作成するには、[Vendor]ボタンをクリックします |
| 5 | 棚卸レポートを作成するには、[Stocktake]ボタンをクリックします |

Table 238- <取扱説明書>レポートを作成する

#### ストックテイクの状態を更新する

Figure 191- <取扱説明書>ストックテイクの状態を更新する[2]

Figure 192- <取扱説明書>ストックテイクの状態を更新する[1]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Comfirmation」ボタンをクリックします |
| 2 | 「Update」ボタンをクリックして確定します |

Table 240- <取扱説明書>ストックテイクの状態を更新する

#### プレフィックスを編集する

Figure 177- <取扱説明書>プレフィックスを編集する[2]

Figure 176- <取扱説明書>プレフィックスを編集する[1]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Edit Invoice Prefix」ボタンをクリックします |
| 2 | 「Prefix」フィールドに記入します |
| 3 | 「Confirm」ボタンをクリックします |

Table 241- <取扱説明書>プレフィックスを編集する

### モバイルアプリ

#### 棚（セル）を登録する

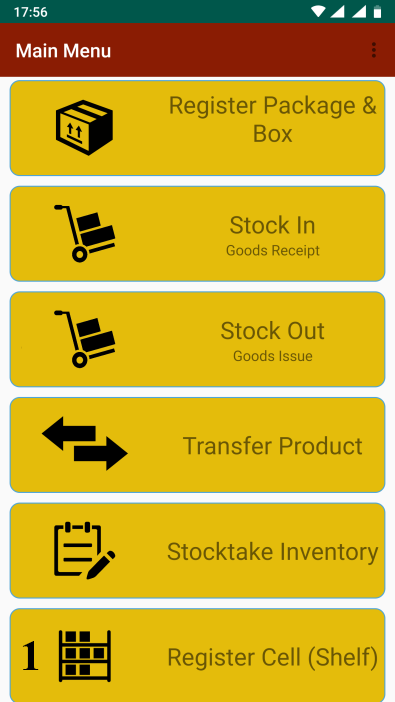
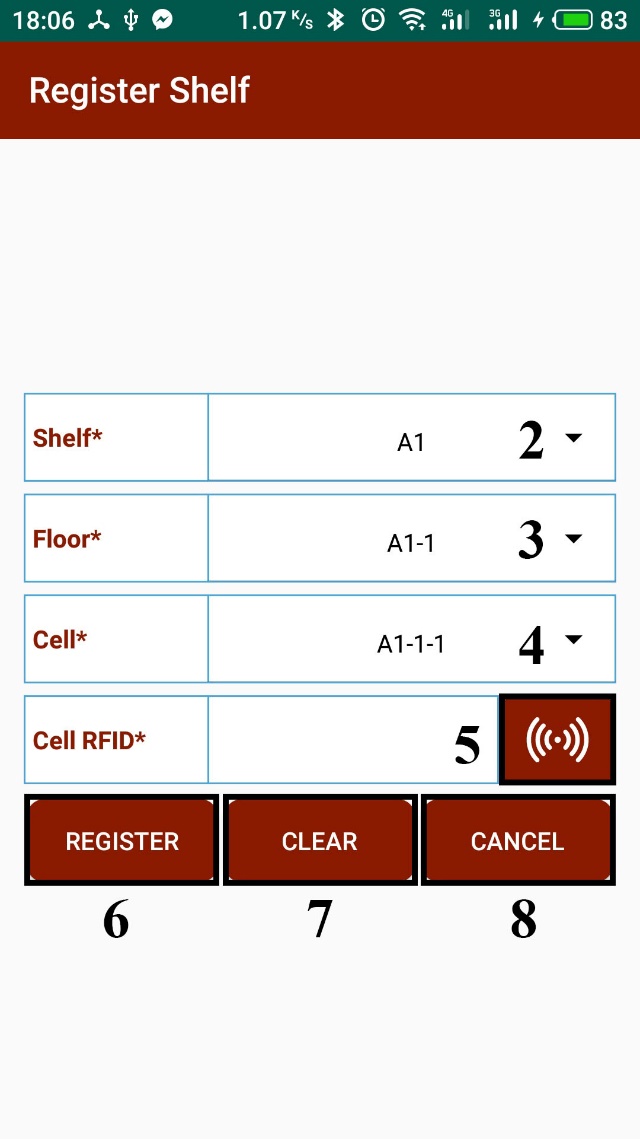
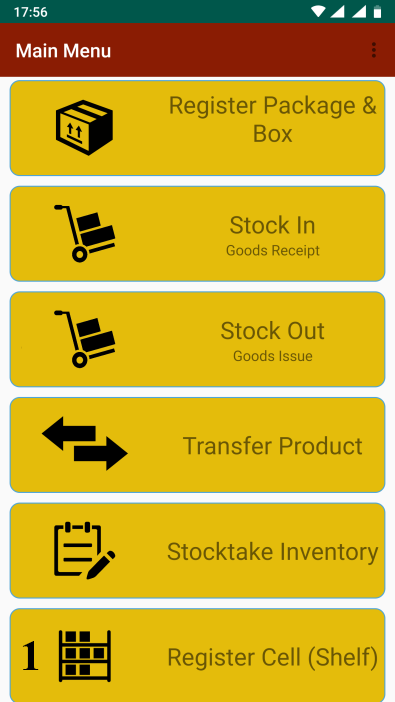
****

Figure 194- <取扱説明書>棚（セル）を登録する[1]

Figure 195- <取扱説明書>棚（セル）を登録する[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Register Shelf (Cell)」機能をタップします |
| 2 | 「Shelf」を選択します |
| 3 | 「Floor」を選択します |
| 4 | 「Cell 」を選択します |
| 5 | 「Cell」をスキャンするために「Scan」ボタンをタップします |
| 6 | 登録するには「Register」ボタンをタップします |
| 7 | データをリセットするには「Clear」ボタンをタップします |
| 8 | 画面を終了するには「Cancel」ボタンをタップします |

Table 243- <取扱説明書>棚（セル）を登録する

#### パッケージとボックスを登録する

Figure 196- <取扱説明書>パッケージとボックスを登録する[1]

Figure 197- <取扱説明書>パッケージとボックスを登録する[2]

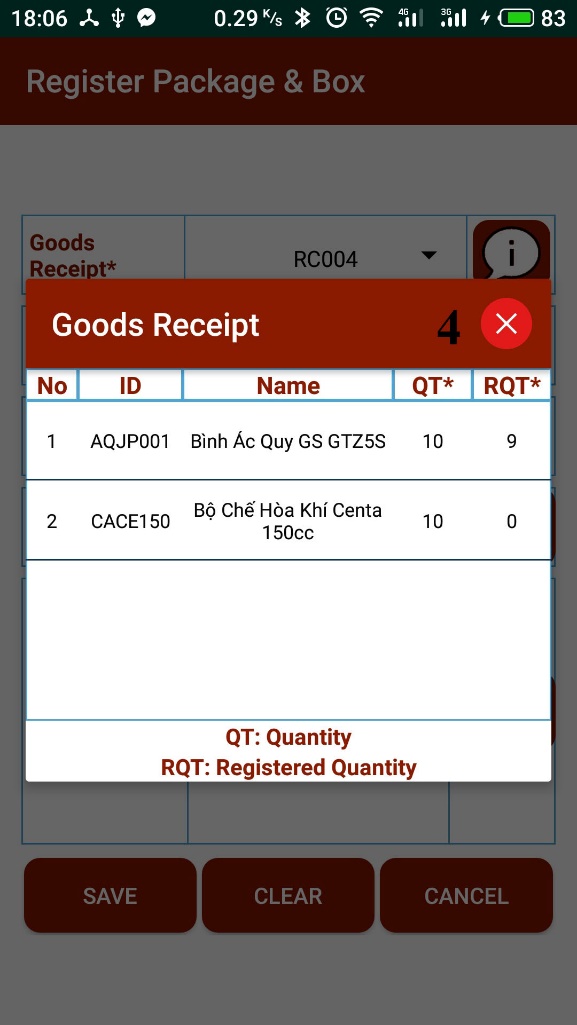
****

Figure 198- <取扱説明書>パッケージとボックスを登録する [3]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Register Package and Box」機能をタップします |
| 2 | 「Goods Receipt」を選択します |
| 3 | 領収書の詳細を見るには「Info」ボタンをタップします |
| 4 | 終了するには「X」ボタンをタップします |
| 5 | 「Product Name」を選択します |
| 6 | 「Package RFID」をスキャンするには「Scan」ボタンをタップします |
| 7 | 「Box RFID」をスキャンするために「Scan」ボタンをタップします |
| 8 | 登録するには「Save」ボタンをタップしてします |
| 9 | データをリセットするには「Clear」ボタンをタップしてします |
| 10 | 画面を終了するには「Scan」ボタンをタップしします |

Table 244- <取扱説明書>パッケージとボックスを登録する

#### 在庫に入れる

Figure 199- <取扱説明書>在庫に入れる[1]

Figure 200-<取扱説明書>在庫に入れる[2]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Stock In」機能をタップします |
| 2 | 「Cell RFID」をスキャンするために「Scan」ボタンをタップします |
| 3 | 「Package RFID」をスキャンするには「Scan」ボタンをタップします |
| 4 | 「Save」ボタンをタップして仕入れます |
| 5 | データをリセットするには「Clear」ボタンをタップします |
| 6 | 画面を終了するには「Cancel」ボタンをタップします |

Table 245- <取扱説明書>在庫に入れる

#### 在庫に出す

Figure 201- <取扱説明書>在庫に出す[1]

Figure 202- <取扱説明書>在庫に出す[2]

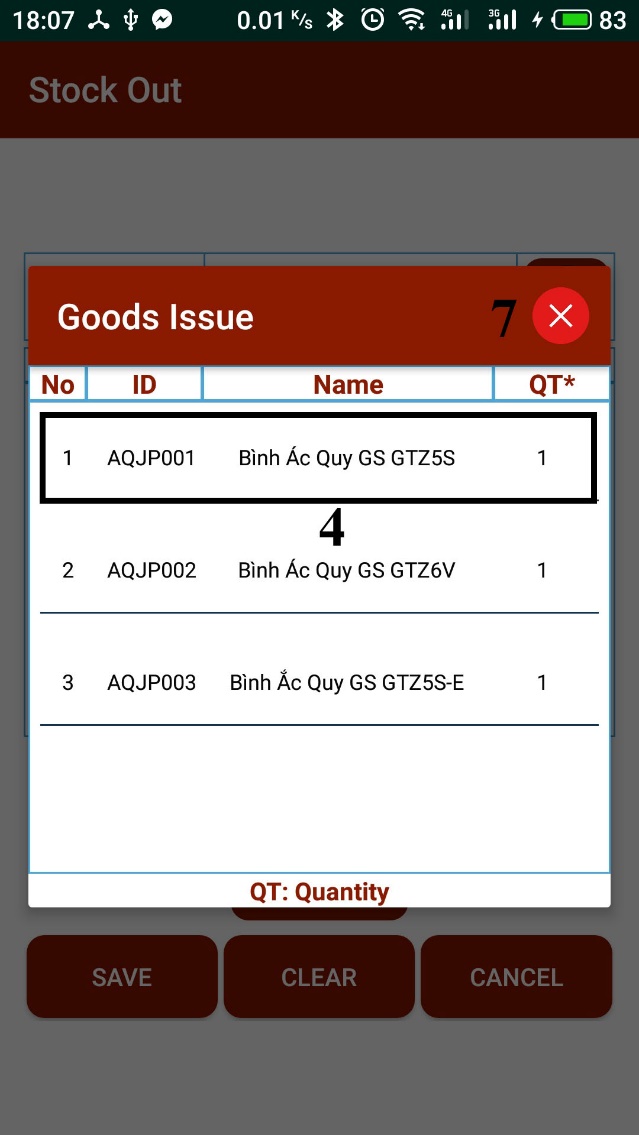
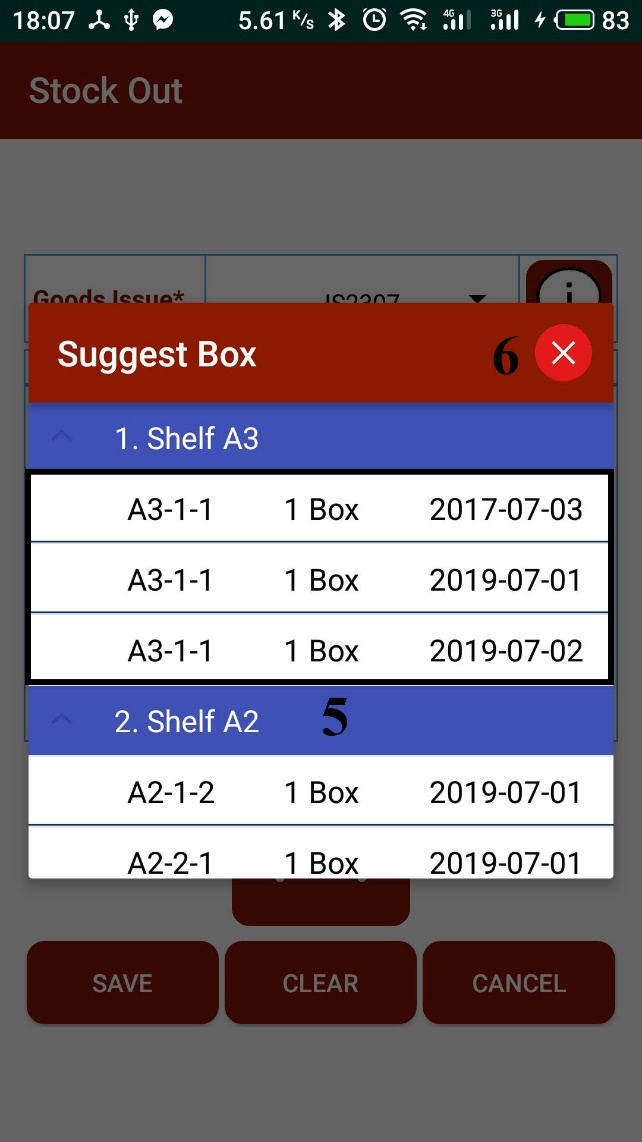
****

Figure 203- <取扱説明書>在庫に出す[3]

Figure 204- <取扱説明書>在庫に出す[4]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Stock Out」機能をタップします |
| 2 | 「Issue Invoice」を選択します |
| 3 | 出庫の詳細を見るには「Info」ボタンをタップします |
| 4 | 商品をタップして「Suggest Box」を表示します |
| 5 | 在庫切れにしたいボックスをタップします |
| 6 | 「Suggest Box」画面を終了するには、「X」ボタンをタップします |
| 7 | 「Goods Issue Detail」詳細画面を終了するには、「X」ボタンをタップします |
| 8 | 製品をスキャンするには「Scan」ボタンをタップします |
| 9 | 「Save」ボタンをタップして在庫切れ |
| 10 | データをリセットするには「Clear」ボタンをタップします |
| 11 | 画面を終了するには「Cancel」ボタンをタップします |

Table 246- <取扱説明書>在庫に出す

#### ボックスを転送する

Figure 205- <取扱説明書>ボックスを転送する[1]

Figure 206- <取扱説明書>ボックスを転送する[2]

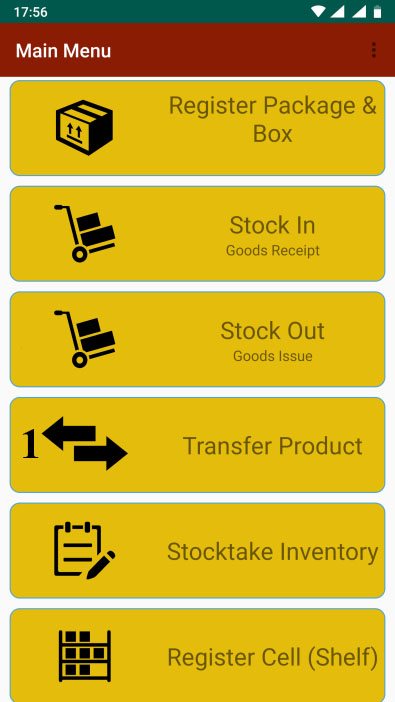
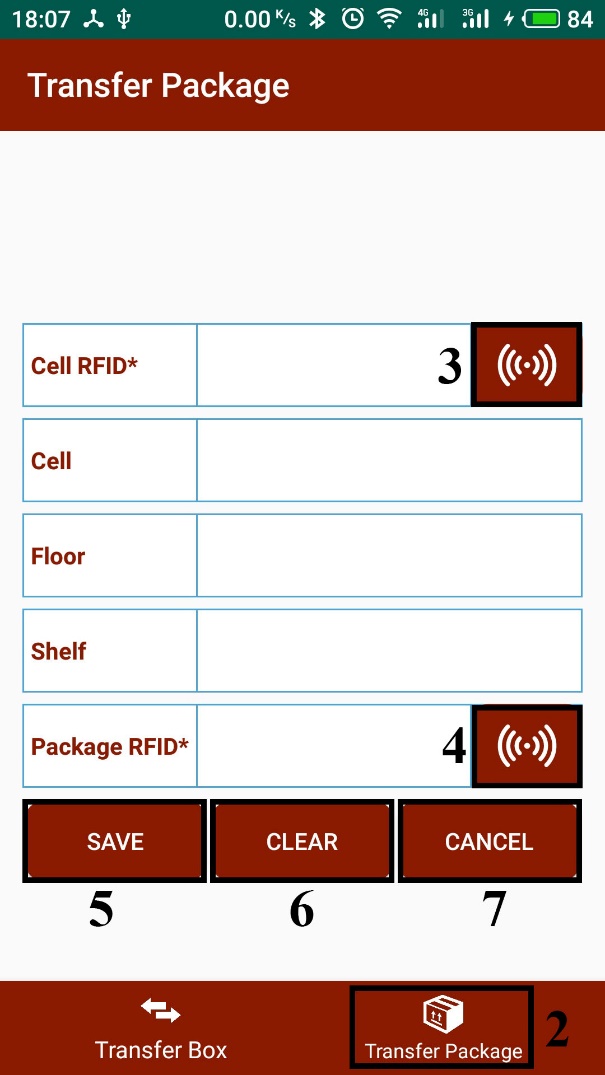
|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Transfer Product」機能をタップします |
| 2 | 「Transfer Box」タブをタップします |
| 3 | パッケージのRFIDをスキャンするために「Scan」ボタンをタップします |
| 4 | 「Scan」ボタンをタップしてボックスのRFIDをスキャンします |
| 5 | 転送するには「Save」ボタンをタップします |
| 6 | データをリセットするには「Clear」ボタンをタップします |
| 7 | 画面を終了するには「Cancel」ボタンをタップします |

Table 247- <取扱説明書>ボックスを転送する

#### パッケージを転送する

Figure 207- <取扱説明書>パッケージを転送する[1]

Figure 208- <取扱説明書>パッケージを転送する[2]

****

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Transfer Product」機能をタップします |
| 2 | 「パッケージ転送」タブをタップ |
| 3 | 「セルRFID」をスキャンするために「スキャン」ボタンをタップします |
| 4 | 「パッケージRFID」をスキャンするには「スキャン」ボタンをタップします |
| 5 | 転送するには「Save」ボタンをタップします |
| 6 | データをリセットするには「Clear」ボタンをタップします |
| 7 | 画面を終了するには「Cancel」ボタンをタップします |

Table 248- <取扱説明書>パッケージを転送する

#### 棚卸在庫

Figure 210- <取扱説明書>棚卸在庫[2]

Figure 209- <取扱説明書>棚卸在庫[1]

|  |  |
| --- | --- |
| ステップ | 説明 |
| 1 | 「Stocktake Inventory」機能をタップします |
| 2 | 「Product Name」を選択します |
| 3 | 製品のRFIDをスキャンして数量を確認するには、「Scan」ボタンをタップします |
| 4 | 報告するには「Report」ボタンをタップします |
| 5 | データをリセットするには「Clear」ボタンをタップします |
| 6 | 画面を終了するには「Cancel」ボタンをタップします |

Table 249- <取扱説明書>棚卸在庫

**すべてのユーザーガイドはこちらにあります**<https://drive.google.com/open?id=1ZaObq4ZIR0Hnr82ru_N6jxvg-bLga4BX>

# G. Appendix

|  |  |
| --- | --- |
| [1] | L. H. S. Peter DeGrace, Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms, Yourdon Press, 1990. |
| [2] | Microsoft, "C# Coding Conventions (C# Programming Guide)," Microsoft, 20 July 2015. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions. |
| [3] | Oracle, "Code Conventions for the Java TM Programming Language," Oracle, 20 April 1999. [Online]. Available: https://www.oracle.com/technetwork/java/javase/documentation/codeconvtoc-136057.html. |
| [4] | Daniel Roth, Rick Anderson, and Shaun Luttin, "Introduction to ASP.NET Core," Mircrosoft, 04 July 2019. [Online]. Available: https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-2.2. |
| [5] | Pivotal, "Spring Framework," Pivotal Software, 2019. [Online]. Available: https:/spring.io/docs/reference. |
| [6] | JetBrains, "IntelliJ IDEA," JetBrains s.r.o, 2019. [Online]. Available: https://www.jetbrains.com/idea/documentation/. |
| [7] | "HtppCode," [Online]. Available: https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html. |
| [8] | Google, "Android Bluetooth," Google Developers, [Online]. Available: https://developer.android.com/guide/topics/connectivity/bluetooth. |
| [9] | Chivorns, "Smart Material Spinner," [Online]. Available: https://github.com/Chivorns/SmartMaterialSpinner. |
| [10] | A. Shvets, "Dive Into Design Pattern," Sourcemaking.com, [Online]. Available: https://sourcemaking.com/design\_patterns/observer. |
| [11] | N. D. Duy, "AsynTask," Viblo, 28 October 2016. [Online]. Available: https://viblo.asia/p/xu-ly-da-tien-trinh-trong-android-bang-asynctask-ZDEeLRMpvJb. |
| [12] | Ita\_c, estenger, ChristianHelms, "Dijkstra Algorithm," GeeksForGeeks, [Online]. Available: https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/. |
| [13] | Statistics How To , "Weighted Mean," [Online]. Available: https://www.statisticshowto.datasciencecentral.com/weighted-mean/. |
| [14] | ANATOLIY, "Airduino Bluetooth," 1 November 2012. [Online]. Available: http://solderer.tv/data-transfer-between-android-and-arduino-via-bluetooth/. |
| [15] | Thái, "Arduino Bluetooth," 14 February 2016. [Online]. Available: http://arduino.vn/bai-viet/628-gui-va-nhan-du-lieu-tu-arduino-va-smartphone-qua-bluetooth. |
| [16] | R. Santos, "Arduino RFID," [Online]. Available: https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/. |
| [17] | C. Wong, "Using EPPlus To Import And Export Data In ASP.NET Core," [Online]. Available: https://www.c-sharpcorner.com/article/using-epplus-to-import-and-export-data-in-asp-net-core/. |
| [18] | M. Spasojevic, "How to Easily Create a PDF Document in ASP.NET Core Web API," [Online]. Available: https://code-maze.com/create-pdf-dotnetcore/. |