

**Final Assignment**

**Information Systems for Agents Retail Stores**

**Requirements Analysis & Design (RAD)**

**By Students:**

1. **521H0494: Tran Quoc Bao**
2. **521H0276: Trang Si Nam**

|  |  |  |  |
| --- | --- | --- | --- |
| **Reference:** | **Team\_G2\_RAD\_Requirements\_Modelling\_v0.1** | | |
| **Audience:** | **Mr. Pham Thai Ky Trung** | **Document Version:** | **26th Dec, 2023** |
| **Outcome:** | **Your Project Title** | | |
| **Abstract:** | This document provides an in-depth analysis of a proposed video rental system with the requirements modelled utilizing the UML framework. The document is a collaboration between the members of my team. | | |
|  | | | |

# Intellectual Property

***Copyright 2023 Team G2.***

The following documentation, the content therein and/or the presentation of its information is proprietary to and embodies the confidential processes, designs, technologies and otherwise of Team G2. All copyright, trademarks, trade names, patents, industrial designs, and other intellectual property rights contained herein are, unless otherwise specified, the exclusive property of Team G2.

The ideas, concepts and/or their application, embodied within this documentation remain and constitute items of intellectual property which nevertheless belong to Team G2.

The information (including, but by no means limited to, data, drawings, specification, documentation, software listings, source and/or object code) shall not be disclosed, manipulated, disseminated or otherwise in any manner inconsistent with the nature and/or conditions under which this documentation has been issued.

The information contained herein is believed to be accurate and reliable. Team G2 accepts no responsibility for its use in any way whatsoever. Team Five shall not be liable for any expenses, damages and/or related costs which may result from the use of the information contained herein.

The information contained herein is subject to change without notice.

All Rights Reserved. Copyright herein is expressly protected at common law, statute and under various International and Multi-National Treatises (including, but by no means limited to, the Berne Convention for the Protection of Literary and Artistic Works).

# Table of Contents

[Intellectual Property 2](#_Toc154520833)

[Table of Contents 3](#_Toc154520834)

[Executive Summary 5](#_Toc154520835)

[I. Business Requirements 6](#_Toc154520836)

[1.1 Organization Chart / Project Chart/Gantt Chart 6](#_Toc154520837)

[1.2 Business Modelling / Requirements 7](#_Toc154520838)

[1.3 Business Processes / Flowchart of Requirements 8](#_Toc154520839)

[1.4 List of Requirements 9](#_Toc154520840)

[II. System Requirements Analysis 10](#_Toc154520841)

[2.1 Translate from Business Use Case 10](#_Toc154520842)

[2.1.1. System Narrative 10](#_Toc154520843)

[2.1.2. Users and their goals 11](#_Toc154520844)

[2.1.3. List of Events 12](#_Toc154520845)

[2.1.4. List of Actors 13](#_Toc154520846)

[2.1.5. List of Use Cases 13](#_Toc154520847)

[2.1.6. Use Case Diagram 14](#_Toc154520848)

[2.1.7. Domain Class Diagram 15](#_Toc154520849)

[2.2 Use Case Description 16](#_Toc154520850)

[2.2.1. Use Case 1: Login 16](#_Toc154520851)

[2.2.2. Use Case 2: Logout 19](#_Toc154520852)

[2.2.3. Use Case 3: Create Manager Account 22](#_Toc154520853)

[2.2.4. Use Case 4: Create Staff Account 25](#_Toc154520854)

[2.2.5. Use Case 5: Add New Product 28](#_Toc154520855)

[2.2.6. Use Case 6: View Product List 30](#_Toc154520856)

[2.2.7. Use Case 7: Update Product Information 31](#_Toc154520857)

[2.2.8. Use Case 8: Delete Product 33](#_Toc154520858)

[2.2.9. Use Case 9: Create New Customer 35](#_Toc154520859)

[2.2.10. Use Case 10: View Customer & Purchase History 38](#_Toc154520860)

[2.2.11. Use Case 11: Update Customer Information 39](#_Toc154520861)

[2.2.12. Use Case 12: Delete Customer 41](#_Toc154520862)

[2.2.13. Use Case 13: Manage Sale Transactions 42](#_Toc154520863)

[2.2.14. Use Case 14: View Total Sales 44](#_Toc154520864)

[2.2.15. Use Case 15: Manage Inventory 45](#_Toc154520865)

[2.2.16. Use Case 16: Create Goods Received 48](#_Toc154520866)

[2.2.17. Use Case 17: Create Goods Delivery Note 51](#_Toc154520867)

[2.2.18. Use Case 18: Generate Report 54](#_Toc154520868)

[2.3 Verifying use cases for Actor 57](#_Toc154520869)

[2.3.1. Verifying uses cases for Actor 1: Company Admin 57](#_Toc154520870)

[2.3.2. Verifying uses cases for Actor 2: Manager 57](#_Toc154520871)

[2.3.3. Verifying uses cases for Actor 3: Staff 57](#_Toc154520872)

[III. System Requirements Design 58](#_Toc154520873)

[3.1 Design Class for Use Case 1: Login 58](#_Toc154520874)

[3.2 Design Class for Use Case 2: Logout 61](#_Toc154520875)

[3.3 Design Class for Use Case 3: Create Manager Account 63](#_Toc154520876)

[3.4 Design Class for Use Case 4: Create Staff Account 65](#_Toc154520877)

[3.5 Design Class for Use Case 5: Add Product 67](#_Toc154520878)

[3.6 Design Class for Use Case 6: View Product 69](#_Toc154520879)

[3.7 Design Class for Use Case 7: Update Product Information 71](#_Toc154520880)

[3.8 Design Class for Use Case 8: Delete Product 73](#_Toc154520882)

[3.9 Design Class for Use Case 9: Create New Customer 75](#_Toc154520883)

[3.10 Design Class for Use Case 10: View Customer & Purchase History 76](#_Toc154520884)

[3.11 Design Class for Use Case 11: Update Customer Information 78](#_Toc154520885)

[3.12 Design Class for Use Case 12: Delete Customer 80](#_Toc154520886)

[3.13 Design Class for Use Case 13: Manage Sales Transaction 82](#_Toc154520887)

[3.14 Design Class for Use Case 14: View Total Sales 84](#_Toc154520888)

[3.15 Design Class for Use Case 15: Manage Inventory 85](#_Toc154520889)

[3.16 Design Class for Use Case 16: Create Goods Received 87](#_Toc154520890)

[3.17 Design Class for Use Case 17: Create Goods Delivery Note 89](#_Toc154520891)

[3.18 Design Class for Use Case 18: Generate Report 91](#_Toc154520892)

[IV. System Requirements Implementation 93](#_Toc154520893)

[4.1 Design Class for Sub System 93](#_Toc154520894)

[4.2 Package Diagram for Sub System 93](#_Toc154520895)

[4.3 Implementation 93](#_Toc154520896)

[4.3.1. Map persistent objects to the tables in a database 93](#_Toc154520897)

[4.3.2. Modifying sequence diagrams 93](#_Toc154520898)

[4.3.3. UI design 93](#_Toc154520899)

[4.3.4. SQL Code 93](#_Toc154520900)

[4.3.5. Software Classes Method Code 96](#_Toc154520901)

[V. SYSTEM TESTING, DEPLOYMENT AND DEMONSTRATION 103](#_Toc154520902)

[5.1 Testing: Test plan & Test case 103](#_Toc154520903)

[5.2 Deployment 103](#_Toc154520904)

[5.3 Demonstration 103](#_Toc154520905)

[Conclusions 104](#_Toc154520906)

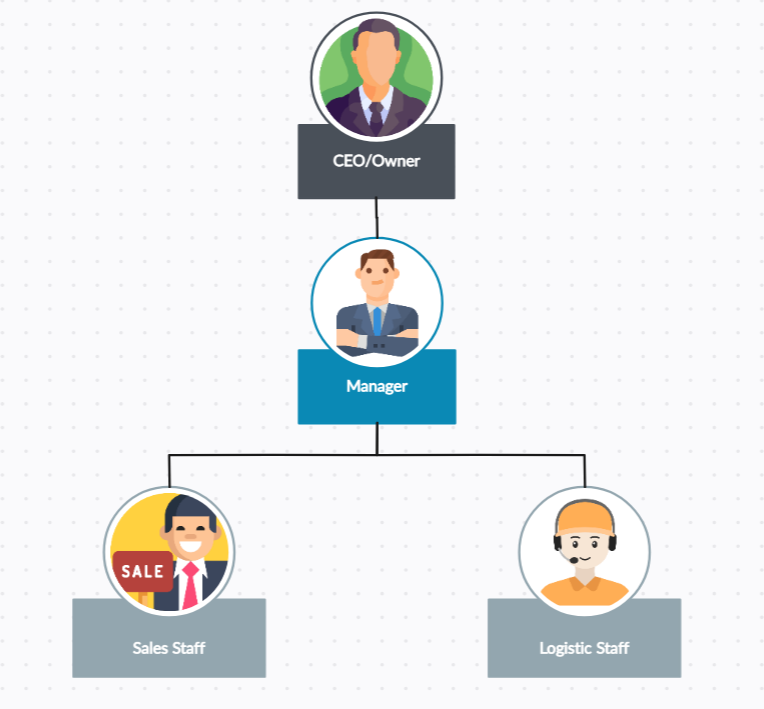
[References 104](#_Toc154520907)

# Executive Summary

* **Account Management**:
  + Company provides Admin account for each agents’ Retail store.
  + The Manager can create accounts for staff members at the store.
* **Product Management**:
  + Admins can perform basic operations on products, such as viewing product listings, adding new products, updating product information, and deleting products.
  + Products have information such as barcode/QR code, product name, import price, retail price, category, and creation date.
  + Products can only be deleted if they are not included in any orders.
  + Sales staff can view the product list but cannot modify its content or see the original price.
  + Irrelevant features for sales staff should be hidden, such as the Delete button for products.
* **Customer Management**:
  + Sales interface allows creating new customers for their first-time purchase.
  + Personal information of customers (name, phone number, address) and their purchase history can be viewed.
* **Sales Management**:
  + Manage sale transactions for each retail store, including generating receipts and reports.
  + Track the total sales of the company.
* **Inventory Management**:
  + Manage inventory across all retail stores.
  + Create Goods Received to record the import of goods from distributors.
  + Create Goods Delivery Notes to deliver goods to agents.
* **Reports**:
  + Generate reports for the company and individual retail stores.

# Business Requirements

# Organization Chart / Project Chart/Gantt Chart

****

# Business Modelling / Requirements

* **Account Management:** This is the first layer of the business model. It involves the creation and management of user accounts. Each retail store will have an Admin account provided by the company. The Manager of each retail store will be able to create accounts for each staff member. This ensures proper access control and accountability with the system.
* **Product Management:** This involves managing the lifecycle of products in store inventory. Administrators will have the ability to modify products. Each product will have a unique identifier (barcode/QR Code), name, import price, retail price, category, and creation date. Products can only be deleted when not in any order. Sales staff can view the product list but cannot change any content or see the original price.
* **Customer Management:** This involves managing customer data and interactions. The sales interface will allow for the creation of a new customer for the first-time purchase and viewing of customers’ personal information and purchase history. This helps in maintaining a good relationship with the customers and also aids in personalized marketing.
* **Sales Management:** This involves managing the sales transactions for each retail store and providing a total sale of the company. This helps in tracking the performance of each store and the overall performance of the company.
* **Inventory Management:** This involves managing the inventory, creating Goods Received when the distributor imports goods, and creating Goods Delivery Note to deliver goods to Agents. This ensures that there is always enough stock to meet the demand and helps in avoiding overstocking.
* **Reports:** This involves generating reports for the company & agents’ retail store. These reports can provide insights into sales trends, inventory levels, customer behaviour, etc. This can help in making informed business decisions.

# Business Processes / Flowchart of Requirements

The Agent’s Retail Stores major process consists of the following sub-processes:

A screenshot of a cell phone

Description automatically generated

# List of Requirements

| **Requirements** | **Functional** | **Non-Functional** |
| --- | --- | --- |
| **The company must provide a manage account for each retail store agent** | **X** |  |
| **The system must allow administrators to modify product** | **X** |  |
| **The system must allow company users to create a new customer for the first-time purchase** | **X** |  |
| **The system must allow company users to manage Sale transactions for each retail store agent** | **X** |  |
| **The system must allow company users to manage the inventory** | **X** |  |
| **The system must allow company users to make report for the company & retail store agents** | **X** |  |
| **Only administrator can change product information** |  | **X** |
| **Sale staff are allowed to view the product list but cannot change any content** |  | **X** |
| **Sale staff cannot see the original price of the product** |  | **X** |
| **The system must not show the Delete button with the Salesperson account** |  | **X** |

# System Requirements Analysis

# Translate from Business Use Case

## System Narrative

Company system

* A retail store agent logs in to the POS system using his or her account. The system verifies the account and displays the main menu with different options according to the agent’s role and permissions (admin, manager, staff).
* The agent can perform various tasks such as managing products, customers, sales, inventory, and reports. For each task, the agent can select the corresponding option from the main menu and the system will show the relevant interface.
* For example, if the agent wants to add a new product, he or she can select the product management option and then the add product option. The system will prompt the agent to enter the product information such as barcode, name, price, category, etc. The system will validate the input and store the product in the database.
* The agent can also view, update, or delete existing products from the product list. The system will display the product details and allow the agent to make changes or remove the product. The system will update or delete the product in the database accordingly.
* Similarly, the agent can manage customers, sales, inventory, and reports using the POS system. The system will provide the necessary functions and interfaces for each task and interact with the database to store or retrieve the data.
* The system will also generate receipts, invoices, and other documents for the transactions and print them out for the customers or the agents. The system will also keep track of the sales and inventory status and alert the agents if there are any issues or discrepancies. The system will also produce various reports for the company and the agents to monitor and analyse the business performance.

Alternate scenario - Customer cancels the order.

## Users and their goals

|  |  |
| --- | --- |
| **User/Actor** | **User Goal** |
| **Company** | **Manage Manager Account** |
| **Manager** | **Manage Staff Account** |
| **Manage Product** |
| **Manage Inventory** |
| **Staff** | **Manage Customer** |
| **Manager Transactions** |

## List of Events

List of events and its use case

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Trigger** | **Source** | **Use case** | **Response** | **Destination** |
| User want to login to the system | User login to the system | All users | **Login** | Logged in successfully | All users |
| User want to logout the system | User logout the system | All users | **Logout** | Logged out successfully | All users |
| Company has new manager | Admin creates new manager account | Admin | **Create Manager Account** | Create successfully | Manager |
| Company has new staff | Manager creates new manager account | Manager | **Create Staff Account** | Create successfully | Staff |
| The retail store has new product | Manager adds new product | Manager | **Add New Product** | Add product successfully | Product |
| User want to view product | User view product | Manager, Staff | **View Product List** | View product details | Product |
| The product needs to be updated | Manager updates product | Manager | **Update Product Information** | Update product successfully | New information of product |
| The product has not been ordered for a while | Manager deletes product | Manager | **Delete Product** | Delete product successfully | None |
| New customers buy products | Sales Staff create new customer | Sales Staff | **Create New Customer** | Add new customer successfully | Customer |
| User want to view customer | User view customer | Sales Staff | **View Customer & Purchase History** | View customer details & purchase history | Customer |
| Customer needs to be updated | User update customer details | Sales Staff | **Update Customer Information** | Update customer successfully | New information of customer |
| User want to delete customer | User delete customer | Sales Staff | **Delete Customer** | Delete customer successfully | Customer |
| Sales staff checks order of customer | Customer pays the bill | Sales Staff | **Manage Sale Transactions** | Pay the bill successfully | Purchase history |
| User want to view total sales | User view total sales | Manager, Staff | **View Total Sales** | Total sales details | Invoices |
| User want check list of products | User check list of products | Logistic Staff | **Manage Inventory** | View total of products | Inventory |
| The store receives new products | Logistic Staff creates Goods Received Note | Logistic Staff | **Create Goods Received** | Create Goods Received successfully | Report |
| The store sends new products | Logistic Staff creates Goods Delivery Note | Logistic Staff | **Create Goods Delivery Note** | Create Goods Delivery successfully | Report |
| Manage want to generate report | Manage generates report | Manager | **Generate Report** | Generate new report | Report |

## List of Actors

* **User**: There are three types of members.
* ***Admin***
* ***Manager***
* ***Staff***
  + ***Sales Staff***
  + ***Logistic Staff***

## List of Use Cases

**Login**

**Logout**

**Create Manager Account**

**Create Staff Account**

**Add New Product**

**View Product List**

**Update Product Information**

**Delete Product**

**Create New Customer**

**View Customer & Purchase History**

**Update Customer Information**

**Delete Customer**

**Manage Sale Transactions**

**View Total Sales**

**Manage Inventory**

**Create Goods Received**

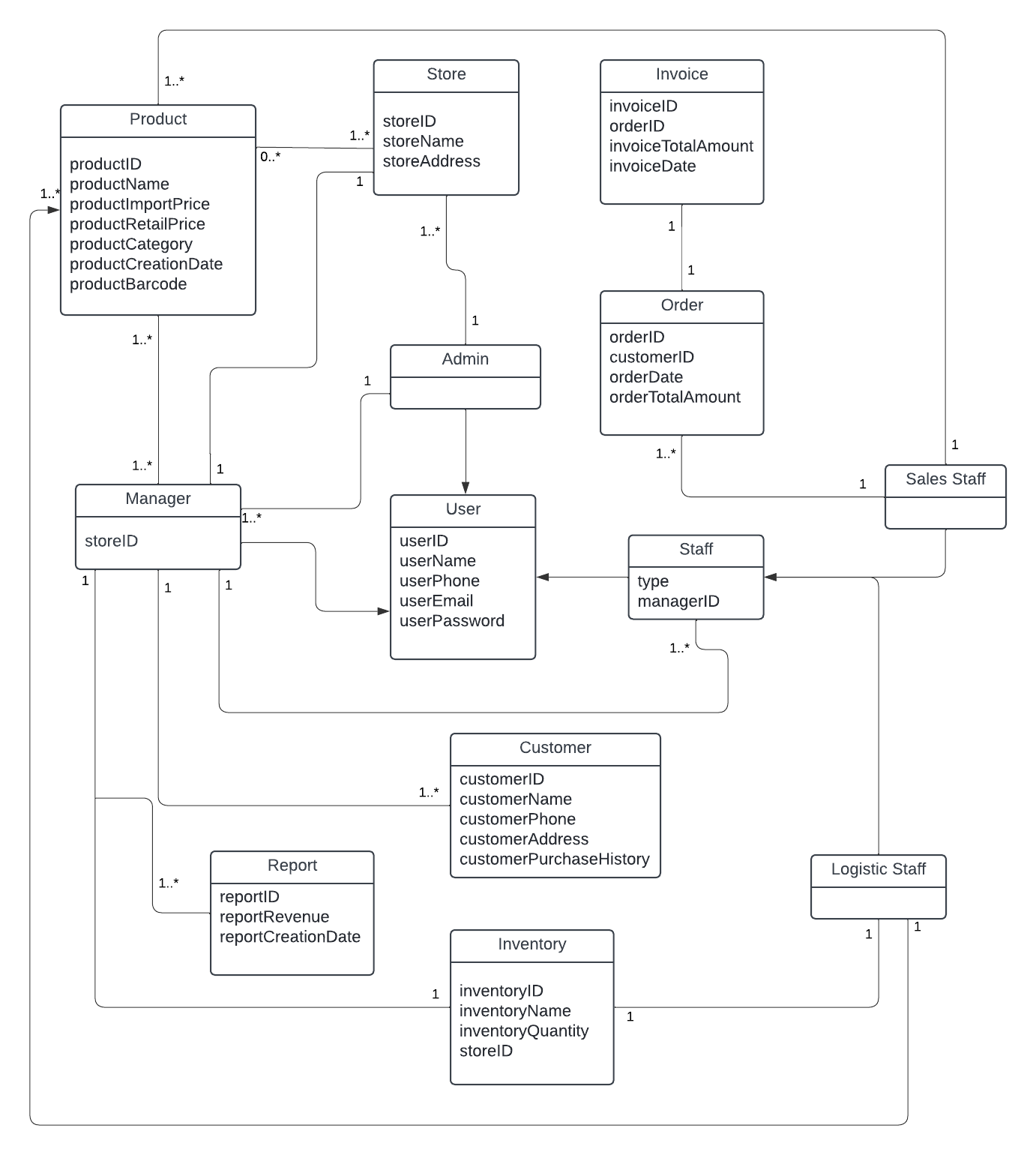
**Create Goods Delivery Note**

**Generate Report**

## Use Case Diagram



## Domain Class Diagram



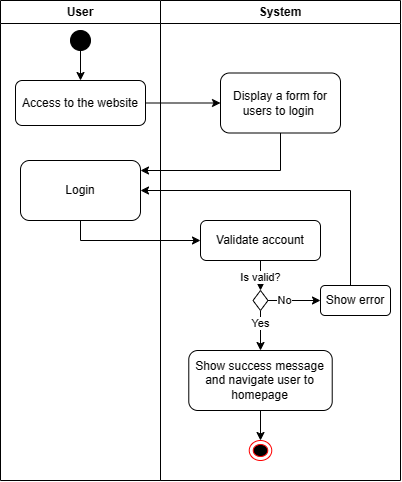
# Use Case Description

## Use Case 1: Login

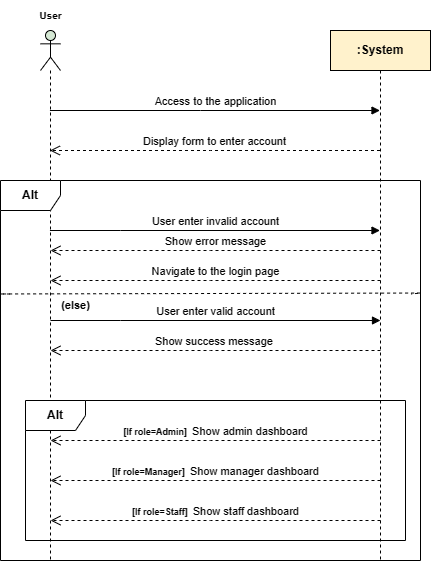
* 1. Use case 1: Login fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Login** | |
| **Scenario** | User login to the Point-of-Sale System | |
| **Triggering event** | The user intends to use the Point-of-Sale system and needs to log in to their account. | |
| **Brief description** | The User Login use case enables authorized users (Admin, Manager, Staff) to securely access the system and perform their respective tasks. | |
| **Actors** | User (Admin, Manager, Staff) | |
| **Related Use case** | None | |
| **Stakeholders** | User (Admin, Manager, Staff) | |
| **Pre-conditions** | The system is installed and running.  The user has a valid account created. | |
| **Post-conditions** | The user is successfully logged into the Point-of-Sale system.  The user can access their assigned functionalities and perform their tasks based on their role and permissions. | |
| **Flow of activities** | **Actor** | **System** |
| 1. User launches the system interface.  2. User logins | 1.1 The system presents a login screen.  2.1 The system verifies the credentials against the stored user database.  2.2 The system displays the user's dashboard with relevant information and available functionalities based on their assigned role and permissions. (Admin, Manager, Staff) |
| **Exception conditions** | 2.1 Invalid Credentials  2.2 Inactive Account | |

* 1. Activity Diagram for Use Case 1: Login



* 1. System sequence diagram for Use Case 1: Login

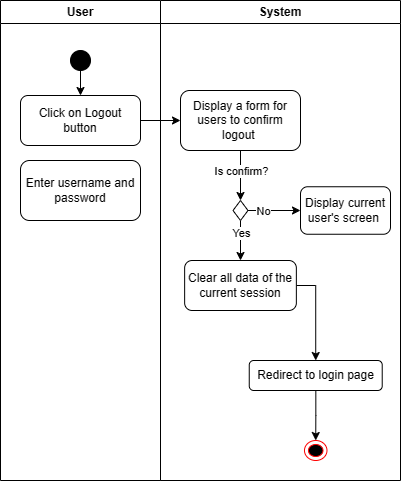


## Use Case 2: Logout

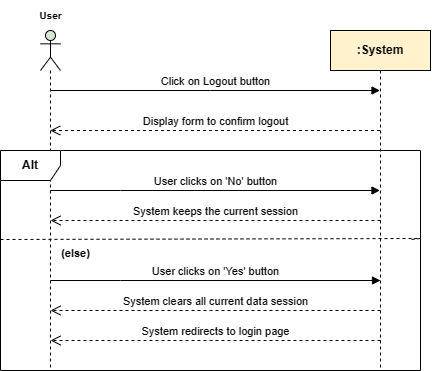
* 1. Use case 2: Logout fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Logout** | |
| **Scenario** | A user has successfully completed their tasks within the system and wishes to securely log out. | |
| **Triggering event** | The user selects the "Logout" option or clicks on a designated logout button within the system. | |
| **Brief description** | The system will terminate the user's current session if they confirm logout. Any user-specific data or temporary information related to the session will be deleted. The user is then redirected to the logout confirmation page, which displays a successful logout confirmation message. As a result, the user is no longer authenticated in the system and if they want to access the system again, they will have to log in again. | |
| **Actors** | User (Admin, Manager, Staff) | |
| **Related Use case** | Login | |
| **Stakeholders** | User (Admin, Manager, Staff) | |
| **Pre-conditions** | The user must be logged in to the Point-of-Sale system. | |
| **Post-conditions** | The user's session is terminated, and they are no longer authenticated within the Point-of-Sale system.  The user is redirected to the logout confirmation page or the login page. | |
| **Flow of activities** | **Actor** | **System** |
| 1. The user initiates the logout process by selecting the "Logout" option or clicking on a designated logout button within the system. | 1.1 The system recognizes the user's request and terminates the current session.  1.2 The system clears any user-specific data or temporary information associated with the session.  1.3 The system redirects users to the login page. |
| **Exception conditions** | 1.1 Session Timeout | |

* 1. Activity Diagram for Use Case 2: Logout



* 1. System sequence diagram for Use Case 2: Logout

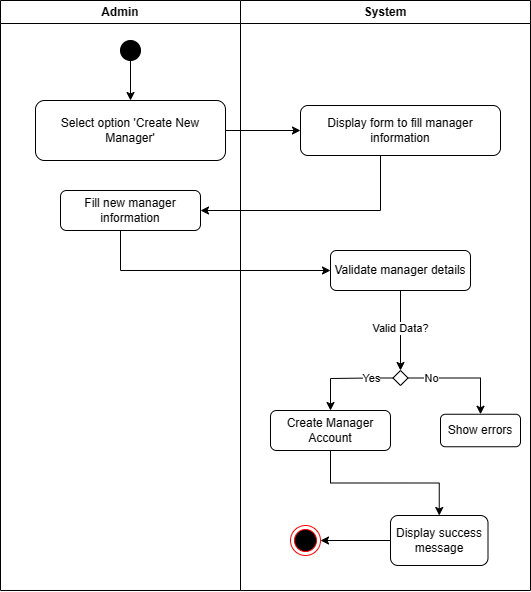


## Use Case 3: Create Manager Account

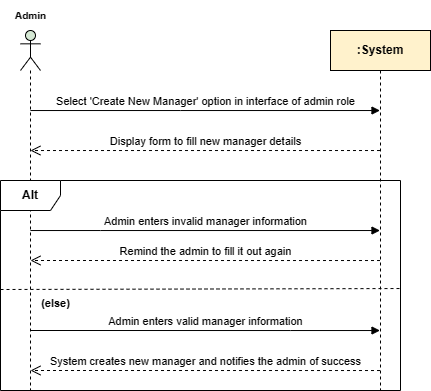
* 1. Use case 3: Create Manager Account fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Create New Manager** | |
| **Scenario** | The admin wants to create a new manager account. | |
| **Triggering event** | The admin adds a new manager account. | |
| **Brief description** | After hiring a new manager for each retail store, the administrator adds manager account, so the manager can work. | |
| **Actors** | Admin | |
| **Related Use case** | Manage User (Manager) | |
| **Stakeholders** | Admin, Manager | |
| **Pre-conditions** | The admin user is logged into the system.  The admin has all necessary permissions to create new accounts. | |
| **Post-conditions** | A new manager account is created in the system with the provided information.  The Admin can manage and modify the manager account as needed | |
| **Flow of activities** | **Actor** | **System** |
| 1. The admin user selects the option to create a new manager account for an agent's retail store.  2. The admin user fills in the necessary details and submits the form to create the new manager account.. | 1.1 The system presents a form for entering the required information for the new manager account.  2.1 The system validates the provided information, ensuring that all required fields are filled and that the username is unique.  2.2 The system creates a new manager account with the provided details.  2.3 The system generates a confirmation message indicating that the manager account has been successfully created. |
| **Exception conditions** | 2.1 Invalid input data or the manager account has existed. | |

* 1. Activity Diagram for Use Case 3: Create Manager Account



* 1. System sequence diagram for Use Case 3: Create Manager Account

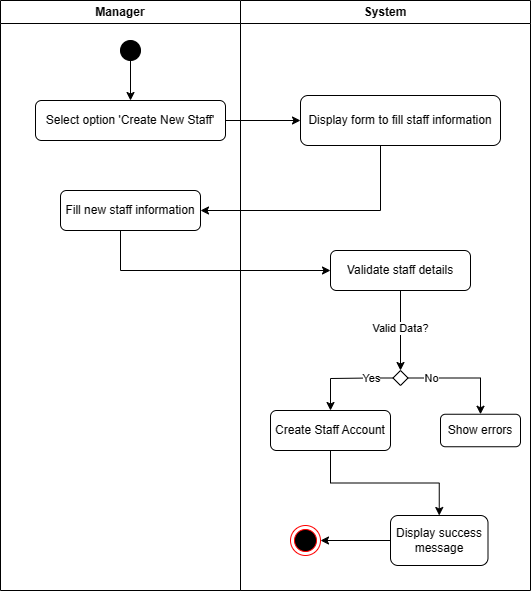


## Use Case 4: Create Staff Account

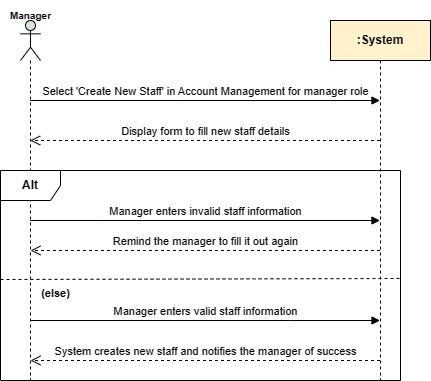
* 1. Use case 4: Create Staff Account fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Create New Staff** | |
| **Scenario** | The manager adds a new staff account. | |
| **Triggering event** | The manager wants to create a new staff account. | |
| **Brief description** | After hiring a new sales staff, the manager adds a staff account, so the staff can work. | |
| **Actors** | Manager | |
| **Related Use case** | Manage User (Staff) | |
| **Stakeholders** | Manager, Staff | |
| **Pre-conditions** | The manager user is logged into the system.  The manager has all necessary permissions to create new accounts. | |
| **Post-conditions** | A new staff account is created in the system with the provided information.  The manager can manage and modify the staff account as needed | |
| **Flow of activities** | **Actor** | **System** |
| 1. The manager navigates to the "Account Management" section of the system (for Manager role).  2. The manager user selects the option to create a new staff account for an agent's retail store.  3. The manager fills in the necessary details and submits the form to create the new staff account. | 1.1 The system shows "Account Management" of the manager role.  2.1 The system presents a form for the manager to enter the required information for the new staff account.  3.1 The system validates the provided information, ensuring that all required fields are filled and that the username is unique.  3.2 The system creates a new staff account with the provided details.  3.3 The system generates a confirmation message indicating that the staff account has been successfully created. |
| **Exception conditions** | 3.1 Invalid input data or the staff account has existed. | |

* 1. Activity Diagram for Use Case 4: Create Staff Account



* 1. System sequence diagram for Use Case 4: Create Staff Account

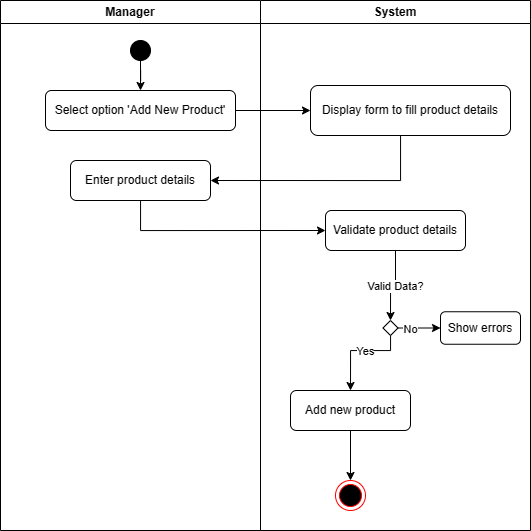


## Use Case 5: Add New Product

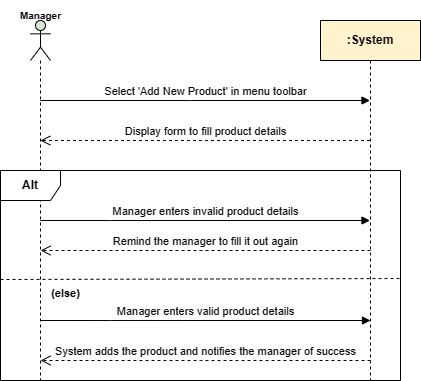
* 1. Use case 5: Add New Product fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Add New Product** | |
| **Scenario** | The manager adds a new product to the retail store's inventory. | |
| **Triggering event** | The manager selects the "Add New Product" option from the system's menu or toolbar. | |
| **Brief description** | The manager has recently received a shipment of new product items and wants to add them to the store's inventory. Manager logs into the store's management system and navigates to the "Add New Product" feature. | |
| **Actors** | Manager | |
| **Related Use case** | Manage Product, Manage Inventory | |
| **Stakeholders** | Manager, Customer, Staff, Accounting Department, IT Department | |
| **Pre-conditions** | The manager is logged into the system.  The manager has the necessary permissions to add a new product. | |
| **Post-conditions** | The new product is added to the retail store's inventory.  The inventory database is updated with the new product information.  The manager receives a confirmation message regarding the successful addition of the product. | |
| **Flow of activities** | **Actor** | **System** |
| 1. The manager selects the "Add New Product" option from the system's menu or toolbar.  2. The manager enters the required information for the new product and submits the form. | 1.1 The system displays a form for entering the details of the new product.  2.1 The system validates the entered data to ensure it meets the required criteria.  2.2 The system adds the new product to the inventory.  2.3 The system updates the inventory database with the new product information.  2.4 The system confirms the successful addition of the product and displays a success message to the manager |
| **Exception conditions** | 2.1 Invalid product details. | |

* 1. Activity Diagram for Use Case 5: Add New Product



* 1. System sequence diagram for Use Case 5: Add New Product

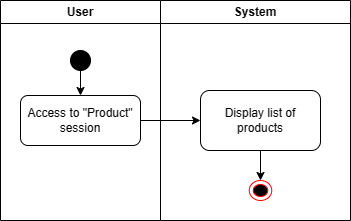


## Use Case 6: View Product List

* 1. Use Case 8: View Product List fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **View Product List** | |
| **Scenario** | The user views a list of products in the retail store's inventory. | |
| **Triggering event** | The user intends to find and explore products of interest by viewing the list of available products | |
| **Brief description** | The user accesses the retail store's website and view a visually appealing list of available products | |
| **Actors** | All Users | |
| **Related Use case** | Manage Product, Manage Inventory | |
| **Stakeholders** | Manager, Customer, Staff | |
| **Pre-conditions** | The user has access to the retail store's website.  The user is logged in (if required) or accessing the store as a guest. | |
| **Post-conditions** | The user has found and explored products of interest by viewing the list of available products in the retail store.  The user can proceed to view the detailed information of a specific product (if desired). | |
| **Flow of activities** | **Actor** | **System** |
| 1. The user navigates to the "Products" section. | 1.1 The system presents the list of available products. |
| **Exception conditions** | None | |

* 1. Activity Diagram for Use Case 8: View Product List



* 1. System sequence diagram for Use Case 8: View Product List

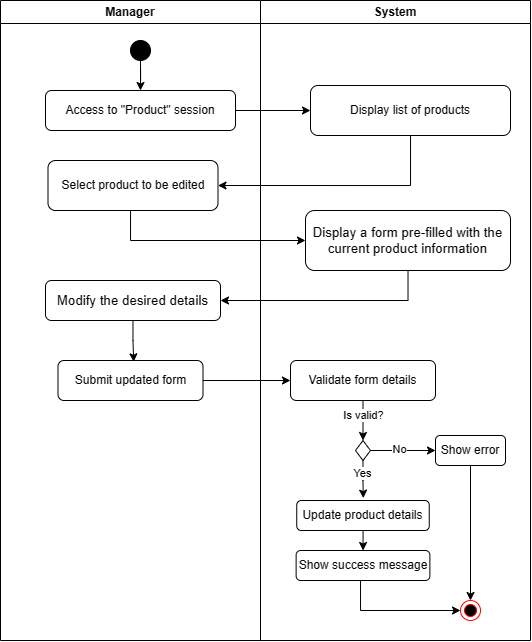


## Use Case 7: Update Product Information

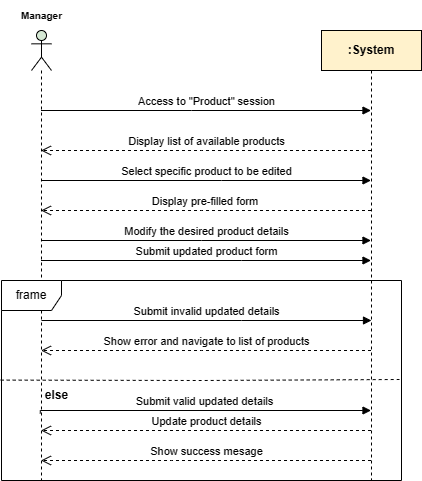
* 1. Use case 9: Update Product Information fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Update Product Information** | |
| **Scenario** | The manager updates product information | |
| **Triggering event** | The manager want to change product details | |
| **Brief description** | There are some changes to the product catalog or products that need to be updated. The manager will update the products to avoid misleading information. | |
| **Actors** | Manager | |
| **Related Use case** | Manage Product, Manage Inventory | |
| **Stakeholders** | Manager, Customer, Staff | |
| **Pre-conditions** | The manager is logged in to the retail store's admin panel.  The manager has the necessary permissions to update product information. | |
| **Post-conditions** | The manager has successfully updated the product information.  Customers and other users will see the updated product information on the store's website. | |
| **Flow of activities** | **Actor** | **System** |
| 1. The user navigates to the "Products" section.  2. The manager selects the specific product and clicks on the "Edit product" option.  3. The manager modifies the desired fields and submits the form to save the updated product information. | 1.1 The system presents the list of available products.  2.1 The system displays a form pre-filled with the current product information.  3.1 The system validates the entered information.  3.2 The system updates the product details in the store's inventory and sends a confirmation message indicating the successful update of the product information to the manager. |
| **Exception conditions** | 3.1 Invalid updated information | |

* 1. Activity Diagram for Use Case 9: Update Product Information



* 1. System sequence diagram for Use Case 9: Update Product Information

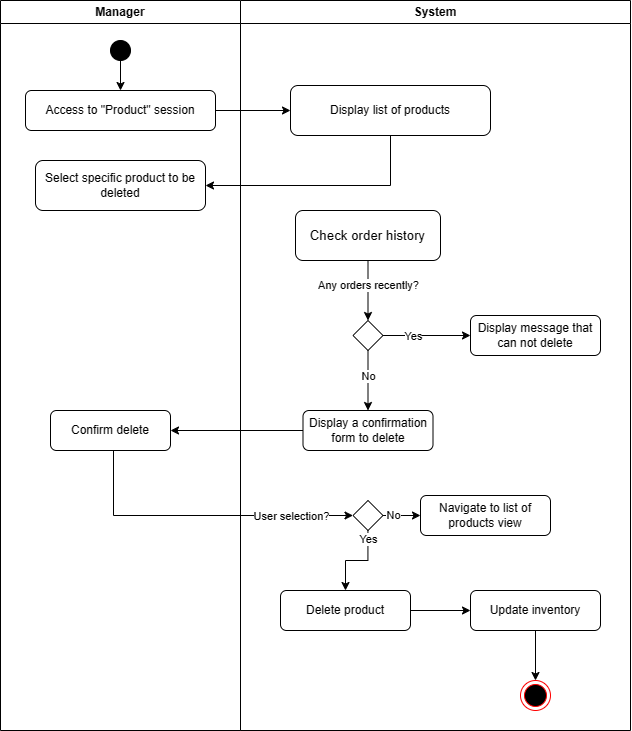


## Use Case 8: Delete Product

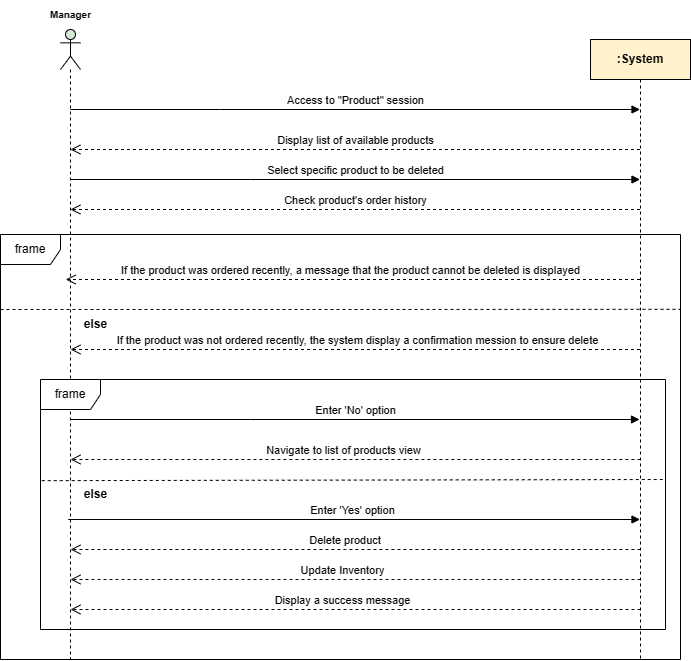
* 1. Use case 10: Delete Product fully description

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Delete Product** | |
| **Scenario** | The manager deletes product | |
| **Triggering event** | The manager want to delete product | |
| **Brief description** | That product has been out of order for a long time, so the manager needs to delete that product | |
| **Actors** | Manager | |
| **Related Use case** | Manage Product, Manage Inventory | |
| **Stakeholders** | Manager, Customer, Staff | |
| **Pre-conditions** | The manager is logged in to the retail store's admin panel.  The manager has the necessary permissions to delete the product. | |
| **Post-conditions** | The manager has successfully deleted the product.  The product is no longer available in the store's inventory.  Customers and other users will no longer see the deleted product on the store's website. | |
| **Flow of activities** | **Actor** | **System** |
| 1. The user navigates to the "Products" section.  2. The manager locates the product they want to delete and clicks on the 'Delete Product' button. | 1.1 The system presents the list of available products.  2.1 The system reviews the product's order history  2.2 The system prompts the administrator for confirmation to ensure deletion.  2.3 The system removes the product from the store's inventory and updates the product list accordingly.  2.4 The system sends a message indicating the successful deletion of the product. |
| **Exception conditions** | 2.1 The product has been included in an order within the specified period.  2.2 The manager chose the 'No' option. | |

* 1. Activity Diagram for Use Case 10: Delete Product



* 1. System sequence diagram for Use Case 10: Delete Product

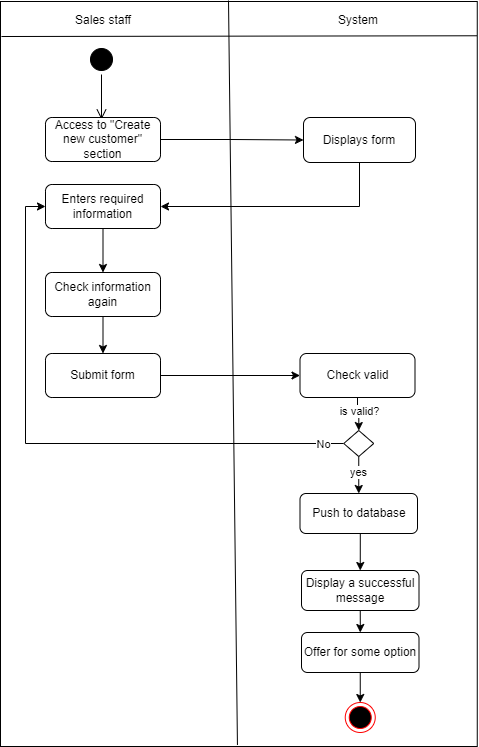


## Use Case 9: Create New Customer

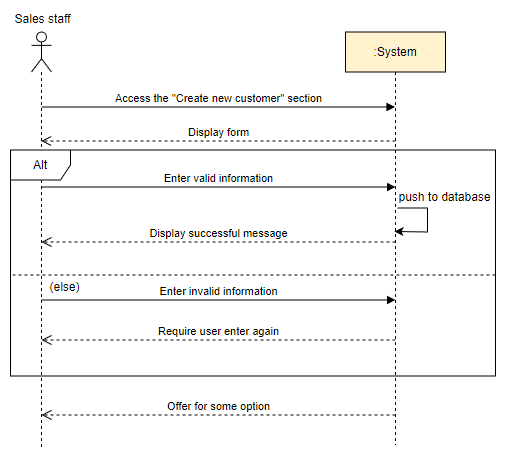
* 1. Use case 9: Create New Customer

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Create New Customer** | |
| **Scenario** | User want to create information for new customer | |
| **Triggering event** | The identification of a new customer who wishes to engage with the retail store. | |
| **Brief description** | This use case describes the process of creating a new customer record in a retail store's system. | |
| **Actors** | Sales staff | |
| **Related Use case** | Manage Customer, Manage Transactions | |
| **Stakeholders** | Sales staff | |
| **Pre-conditions** | System is available and accessible to the user  Users have access to the “Create New Customer” functionality. | |
| **Post-conditions** | A new customer record is created in the database with the provided information.  The user is informed of the result of the operation. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Access the “Create new customer” section.  2. Enters required information(name, email, phone, address,…).  3. Check the information again.  4. Submit the form. | 1. System displays a form for entering customer information.  2. System validates the submitted information.  3. If valid, the system creates new customer records in the database.  4. System displays a success message to the user.  5. System offers the user some options(view record of creation, create another customer, return previous page). |
| **Exception conditions** | System encounters technical issues or database errors during the creation process  Entered information is incomplete or contains errors. | |

* 1. Activity Diagram for Use Case 9: Create New Customer



* 1. System sequence diagram for Use Case 9: Create New Customer

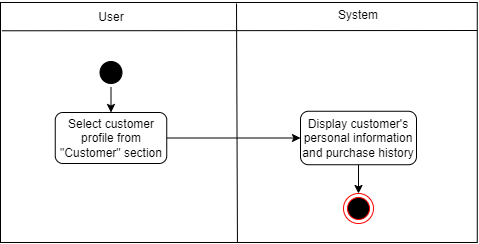


## Use Case 10: View Customer & Purchase History

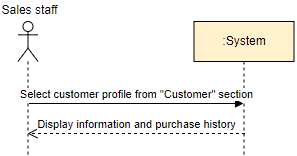
* 1. Use case 10: View Customer & Purchase History

|  |  |  |
| --- | --- | --- |
| **Use case name** | **View Customer & Purchase history** | |
| **Scenario** | User check customer information and purchase history | |
| **Triggering event** | User logs in to their account and navigates to their "Customers" section. | |
| **Brief description** | This use case allows users to view detailed information about a customer, including their information and purchase history. The history includes order details such as date, products bought, quantities, prices, and payment methods. | |
| **Actors** | Sales staff | |
| **Related Use case** | Create New Customer | |
| **Stakeholders** | Sales staff | |
| **Pre-conditions** | Users must have appropriate access rights to view customer information.  Customer profile information and purchase history data must be accurate and up-to-date. | |
| **Post-conditions** | Users gain insights into the customer's purchasing behavior and preferences.  Users can utilize this information for various purposes, such as targeted marketing, inventory management, and customer service. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Selects a customer profile from the "Customers" section. | 1. System displays the customer's personal information, including contact details and purchase history. |
| **Exception conditions** | System error prevents access to customer data or purchase history. | |

* 1. Activity Diagram for Use Case 10: View Customer & Purchase History



* 1. System sequence diagram for Use Case 10: View Customer & Purchase History

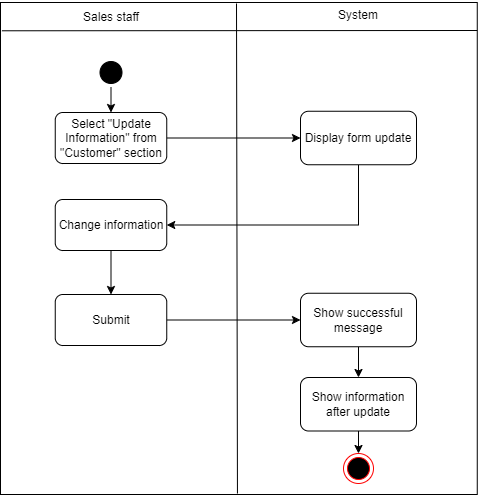


## Use Case 11: Update Customer Information

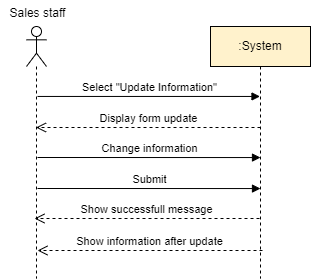
* 1. Use case 11

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Update Customer Information** | |
| **Scenario** | User update customer information | |
| **Triggering event** | User want to update customer information | |
| **Brief description** | This use case describes process of updating customer information | |
| **Actors** | Sales staff | |
| **Related Use case** | View Customer & Purchase History | |
| **Stakeholders** | Sales staff | |
| **Pre-conditions** | Customer information must be create previously  User must have appropriate access rights to view customer information | |
| **Post-conditions** | Updated information is reflected in other relevant systems | |
| **Flow of activities** | **Actor** | **System** |
| 1. Select "Update Information" in the "Customer" section.  2. Change information for customers.  3. Submit updates. | 1. System displays a form to update information.  2. System shows a success message.  3. System shows new information after updates. |
| **Exception conditions** | Invalid format, duplicate information | |

* 1. Activity Diagram for Use Case 11



* 1. System sequence diagram for Use Case 11

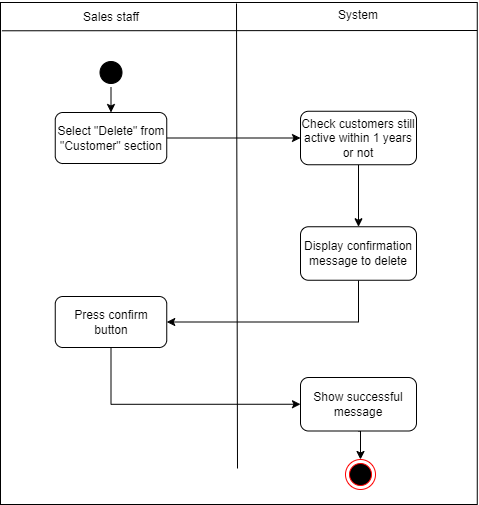


## Use Case 12: Delete Customer

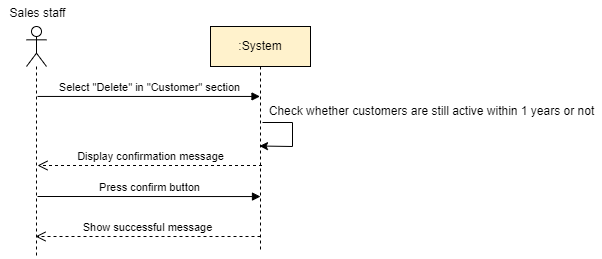
* 1. Use Case 12

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Delete Customer** | |
| **Scenario** | User delete customer | |
| **Triggering event** | User want to delete customer | |
| **Brief description** | This use case describes process of removing a customer information | |
| **Actors** | Sales staff | |
| **Related Use case** | View Customer & Purchase History | |
| **Stakeholders** | Sales staff | |
| **Pre-conditions** | Customer information must be create previously  User has appropriate system access and authority to delete customers | |
| **Post-conditions** | Customer information is permanently removed from the system  No further transactions or interactions can be associated with the deleted customer | |
| **Flow of activities** | **Actor** | **System** |
| 1. Select "Delete" in the "Customer" section.  2. Press the confirm delete button. | 1. System displays a confirmation message to delete.  2. System checks whether customers are still active within 1 year or not.  2. System shows a successful deletion message. |
| **Exception conditions** | User don’t confirm the message to delete | |

* 1. Activity Diagram for Use Case 12



* 1. System sequence diagram for Use Case 12

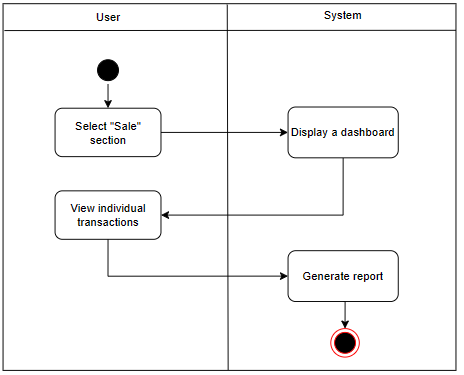


## Use Case 13: Manage Sale Transactions

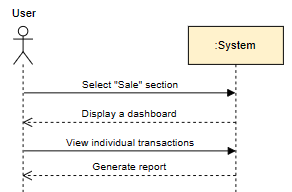
* 1. Use Case 13

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Manage Sale Transactions** | |
| **Scenario** | User needs to efficiently track and manage all sales transactions | |
| **Triggering event** | User want to manage all sale transactions | |
| **Brief description** | This use case focuses on providing all functionalities related to managing sales transactions | |
| **Actors** | Sales staff | |
| **Related Use case** | View Product, View Customer | |
| **Stakeholders** | Sales staff | |
| **Pre-conditions** | Sales data is accurately captured and entered into the system. | |
| **Post-conditions** | Receipts are generated accurately and easily accessible.  Reports provide valuable insights into sales performance and other key metrics. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Select the "Sale" section.  2. View individual transactions, generate receipts, and download reports. | 1. System displays a dashboard with key sales metrics and options for various tasks.  2. System generates reports with various metrics (sales volume, revenue, product performance, etc.). |
| **Exception conditions** | System have issues and don’t display sale information | |

* 1. Activity Diagram for Use Case 13



* 1. System sequence diagram for Use Case 13

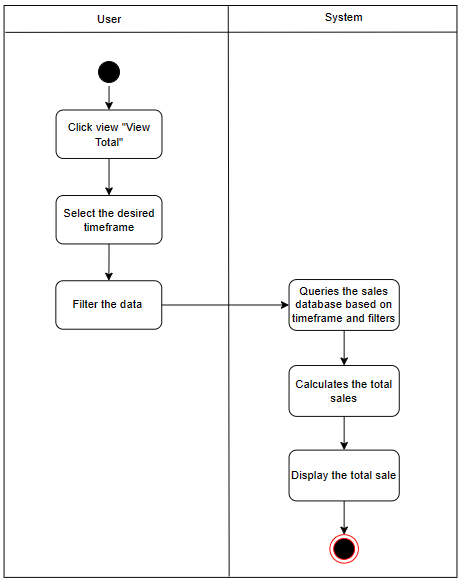


## Use Case 14: View Total Sales

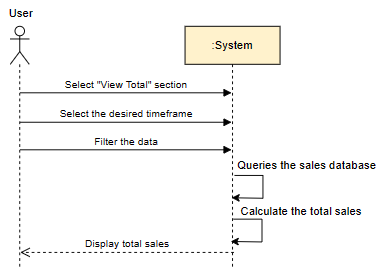
* 1. Use Case 14

|  |  |  |
| --- | --- | --- |
| **Use case name** | **View Total Sale** | |
| **Scenario** | User needs to get a quick overview of total sale for day,week,month or any specified date range | |
| **Triggering event** | User want to view total sale | |
| **Brief description** | This use case allows the user to retrieve the total sales amount for a chosen time frame | |
| **Actors** | User(Manager, Staff) | |
| **Related Use case** | Login | |
| **Stakeholders** | User(Manager, Staff) | |
| **Pre-conditions** | User has a valid account and permissions to access sales data.  Sales data is accurate and up-to-date. | |
| **Post-conditions** | User sees the total sales amount displayed on the screen.  Users can further analyze the data using other related use cases. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Click on the "View Total" button from the "Sale" section.  2. Selects the desired timeframe (day, week, month, date range).  3. Filter the data by product, category, or other criteria. | 1. Queries the sales database based on the specified timeframe and filters.  2. Calculates the total sales amount.  3. Displays the total sales amount on the screen. |
| **Exception conditions** | User enters an invalid timeframe or data format.  System encounters a database error or connectivity issue. | |

* 1. Activity Diagram for Use Case 14



* 1. System sequence diagram for Use Case 14

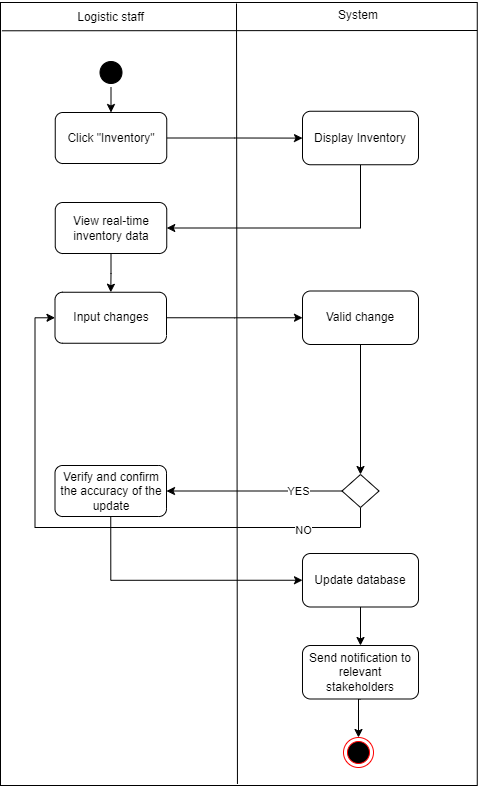


## Use Case 15: Manage Inventory

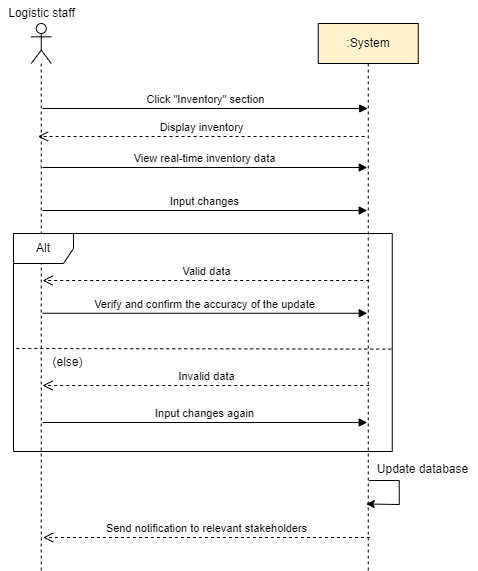
* 1. Use Case 15

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Manage Inventory** | |
| **Scenario** | User needs to manage the total inventory about stock levels, item quantities | |
| **Triggering event** | New items added so need to update quantities | |
| **Brief description** | This use case involves the systematic management of the total items in the inventory system, ensuring that each retail store's stock is accurately reflected and updated. | |
| **Actors** | Logistic staff | |
| **Related Use case** | View Product | |
| **Stakeholders** | Logistic staff | |
| **Pre-conditions** | The inventory system is operational.  Retail store agents have access to the inventory system. | |
| **Post-conditions** | Inventory records are updated and synchronized across all retail stores.  Notifications are sent to relevant stakeholders about inventory changes. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Click on the "Inventory" section.  2. Views real-time inventory data  3. Input changes in stock levels based on sales, returns, or new arrivals.  4. Verify and confirm the accuracy of the updated inventory information. | 1. Display Inventory Data  2. Validates the changes.  3. Updates the central inventory database.  4. Sends notifications to relevant stakeholders. |
| **Exception conditions** | System error leads to inaccurate inventory data.  Discrepancies are found between physical inventory counts and system data. | |

* 1. Activity Diagram for Use Case 15



* 1. System sequence diagram for Use Case 15

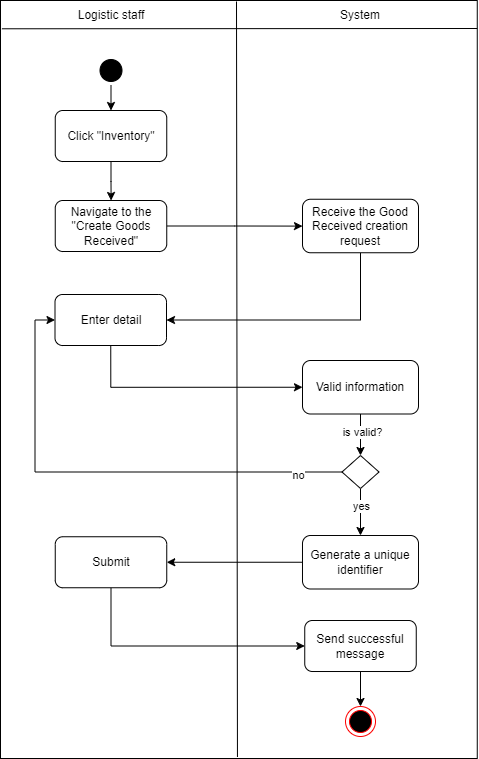


## Use Case 16: Create Goods Received

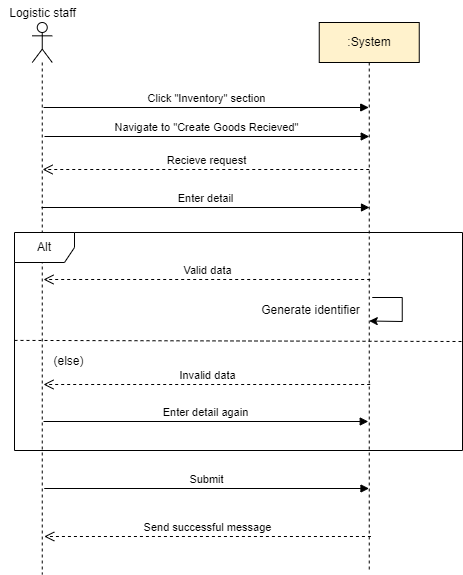
* 1. Use Case 16

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Create Goods Received** | |
| **Scenario** | User needs to create goods received when the distributor import new goods | |
| **Triggering event** | The user receives a shipment of goods from an external supplier. | |
| **Brief description** | This use case involves the creation of a Goods Received entry in the system to document the arrival of goods imported by the user. | |
| **Actors** | Logistic staff | |
| **Related Use case** | Manage Inventory | |
| **Stakeholders** | Logistic staff | |
| **Pre-conditions** | The user has a valid order or contract with the supplier for the import of goods.  The goods have been successfully delivered to the user's premises. | |
| **Post-conditions** | The inventory system is updated with the details of the received goods.  The user has a record of the imported goods. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Click on the “Inventory” section.  2. Navigates to the "Create Goods Received" module.  3. Enters details of the received goods (e.g., quantity, item codes, batch numbers).  4. Submits the Goods Received entry. | 1. Receives the Goods Received creation request.  2. Validates the entered information.  3. Updates the inventory database with the received goods information.  4. Generates a unique identifier for the Goods Received entry.  5. Send a successful message. |
| **Exception conditions** | Discrepancies between received quantities and purchase order.  Incomplete or incorrect import documentation. | |

* 1. Activity Diagram for Use Case 16



* 1. System sequence diagram for Use Case 16

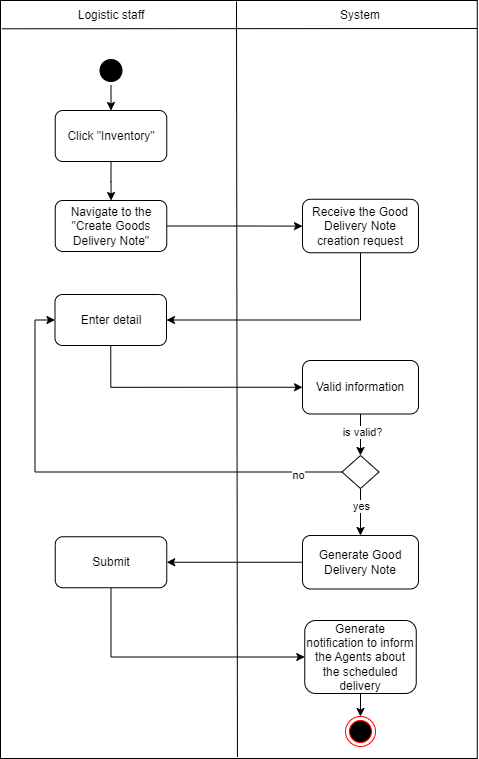


## Use Case 17: Create Goods Delivery Note

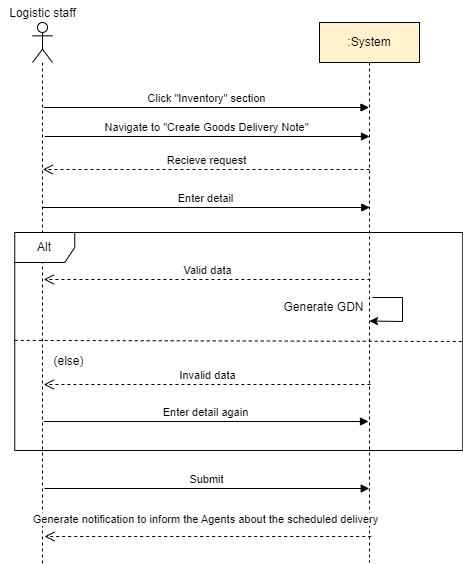
* 1. Use Case 17

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Create Goods Delivery Note** | |
| **Scenario** | User need to deliver goods to various sales agents | |
| **Triggering event** | Users deliver goods to Agents. | |
| **Brief description** | This use case involves the creation of a Goods Delivery Note (GDN) to facilitate the delivery of goods to Agents. | |
| **Actors** | Logistic staff | |
| **Related Use case** | Manage Inventory, Create Goods Received | |
| **Stakeholders** | Logistic staff | |
| **Pre-conditions** | Staff must be authenticated and have the necessary permissions to create a Goods Delivery Note.  The goods to be delivered must be available and prepared in the warehouse. | |
| **Post-conditions** | A Goods Delivery Note is created and stored in the system.  The goods are marked as scheduled for delivery. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Click on the “Inventory" section.  2. Navigates to the "Create Goods Delivery Note" module.  3. Enter details of the required information.  4. Confirms and submits. | 1. Receives the Goods Delivery Note creation request.  2. Validates the entered information.  3. System Generates GDN.  4. System generates notifications to inform the Agents about the scheduled delivery. |
| **Exception conditions** | Inventory levels are insufficient to fulfill the order.  System error occurs during GDN creation. | |

* 1. Activity Diagram for Use Case 17



* 1. System sequence diagram for Use Case 17

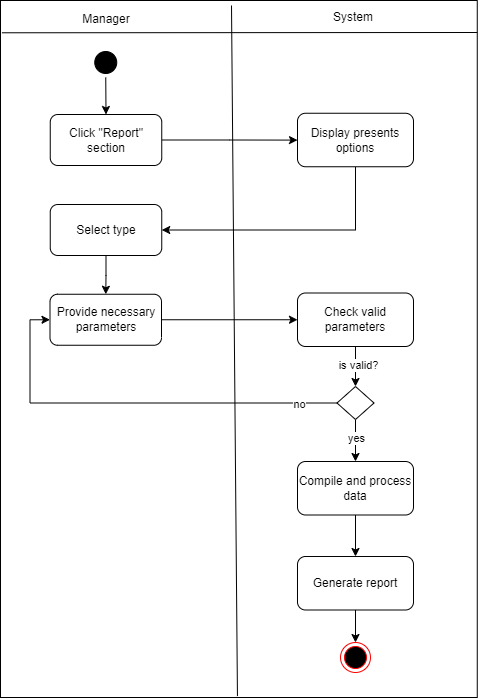


## Use Case 18: Generate Report

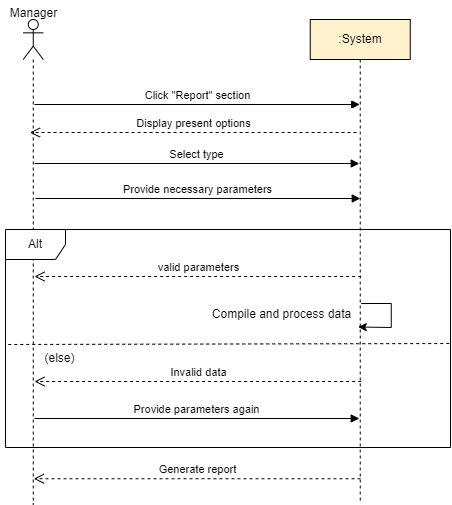
* 1. Use Case 18

|  |  |  |
| --- | --- | --- |
| **Use case name** | **Generate Report** | |
| **Scenario** | Manager need to deliver goods to various sales agents | |
| **Triggering event** | Manager request to generate reports or a predefined schedule | |
| **Brief description** | This use case describes the process of generating reports for a company and its agents' retail store. | |
| **Actors** | Manager | |
| **Related Use case** | Manage Inventory, View Total Sale | |
| **Stakeholders** | Manager | |
| **Pre-conditions** | The system is operational and connected to the retail store's data sources.  The necessary data, including sales transactions, inventory levels, and other relevant metrics, are available in the system. | |
| **Post-conditions** | The reports are generated and made available to the user.  Manager can access and analyze the reports through the system's user interface. | |
| **Flow of activities** | **Actor** | **System** |
| 1. Click on the "Report" section.  2. Selects the type of report needed.  3. Provides the necessary parameters for the report. | 1. System presents options for customizing the report, such as the time frame, specific stores, and performance metrics.  2. System validates the entered parameters.  3. System compiles and processes the data based on the specified parameters.  4. System generates reports, including relevant charts, graphs, and textual summaries. |
| **Exception conditions** | Invalid input, parameters when the user provides.  In case of system failures or interruptions during the report generation process. | |

* 1. Activity Diagram for Use Case 18



* 1. System sequence diagram for Use Case 18



# Verifying use cases for Actor

## Verifying uses cases for Actor 1: Company Admin

|  |  |  |
| --- | --- | --- |
| **Data entity/domain class** | **C R U D** | **Verified use case** |
| Company | Create | None |
|  | Read | None |
|  | Update | None |
|  | Delete | None |

## Verifying uses cases for Actor 2: Manager

|  |  |  |
| --- | --- | --- |
| **Data entity/domain class** | **C R U D** | **Verified use case** |
| Manager | Create | Create Manager Account |
|  | Read | None |
|  | Update | None |
|  | Delete | None |

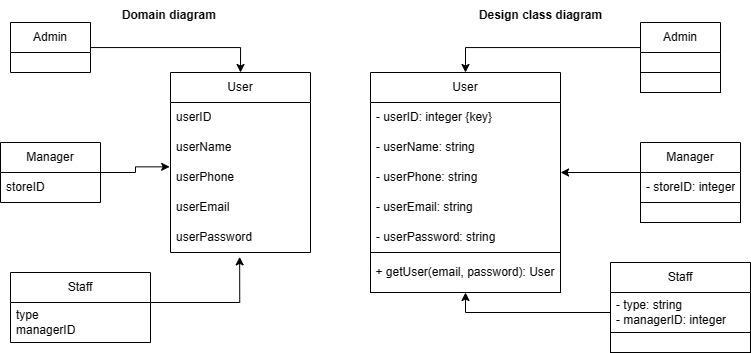
## Verifying uses cases for Actor 3: Staff

|  |  |  |
| --- | --- | --- |
| **Data entity/domain class** | **C R U D** | **Verified use case** |
| Staff | Create | Create Staff Account |
|  | Read | None |
|  | Update | None |
|  | Delete | None |

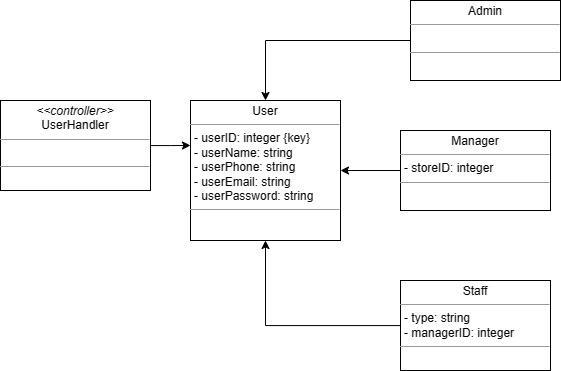
# System Requirements Design

# Design Class for Use Case 1: Login

* + 1. ***Design Class in Detailed Design***



* + 1. ***Design Class Diagram***
       1. *Domain Design Class*



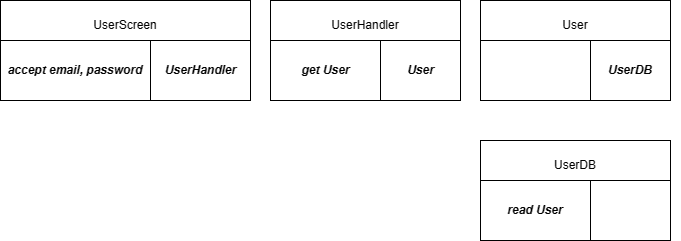
* + - 1. *Controller*



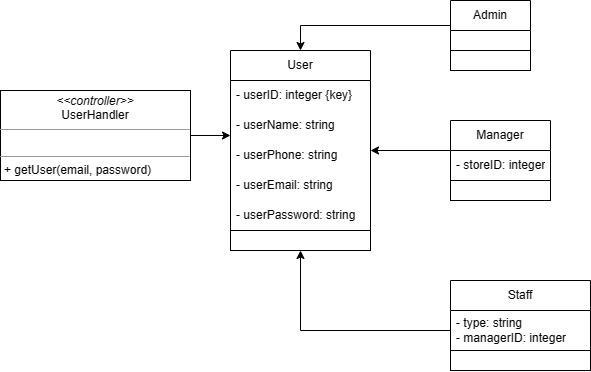
* + - 1. *UI*



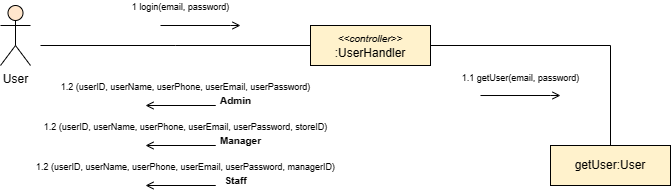
* + - 1. *Data Access*



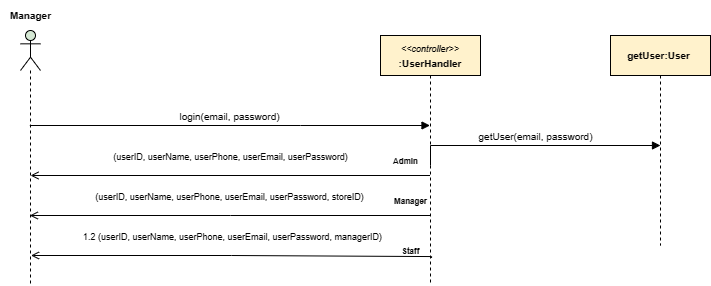
* + - 1. *Design Class*



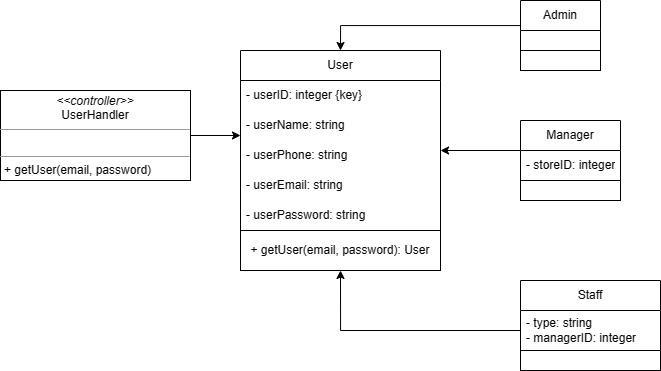
* + 1. ***OOD with Communication***



* + 1. ***OOD with Sequence Diagram***

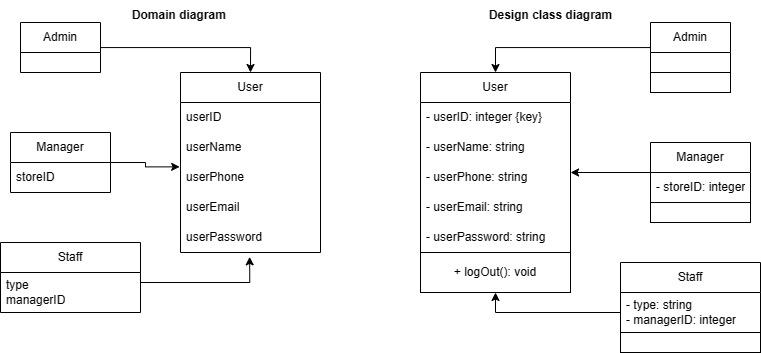


* + 1. ***Final Design Class Diagram***

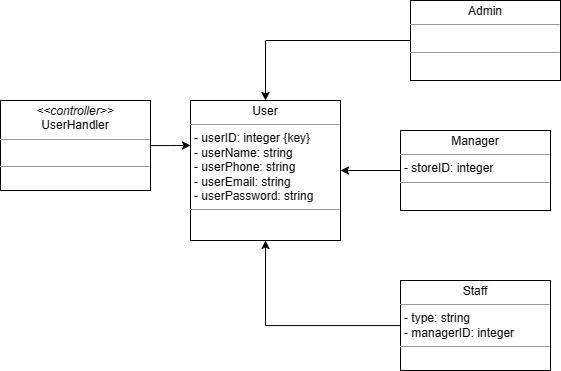


# Design Class for Use Case 2: Logout

* + 1. *Design Class in Detailed Design*



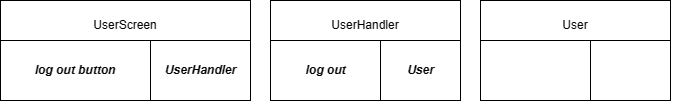
* + 1. *Design Class Diagram*
       1. *Domain Design Class*



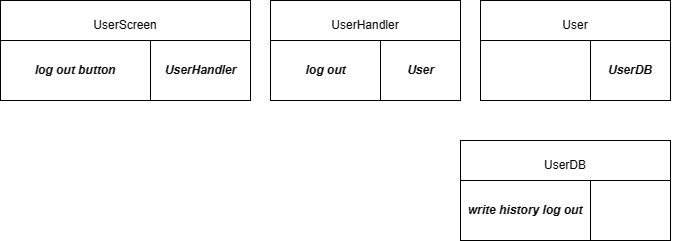
* + - 1. *Controller*



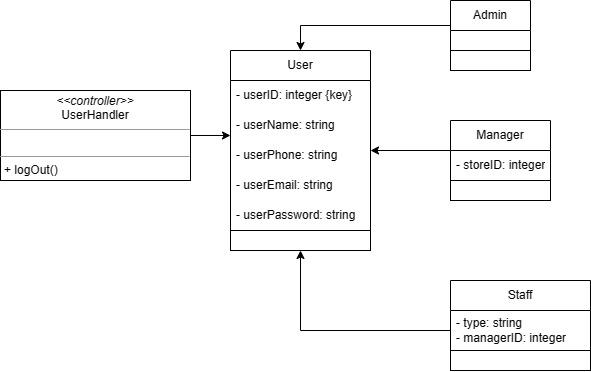
* + - 1. *UI*



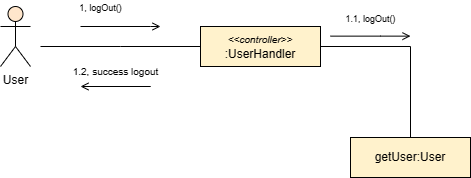
* + - 1. *Data Access*



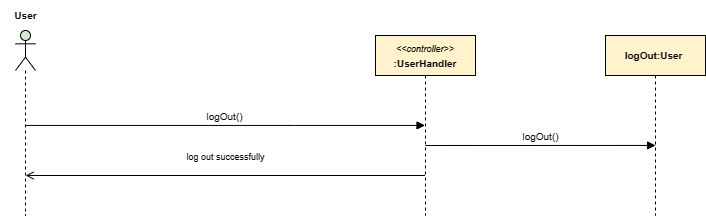
* + - 1. *Design Class*



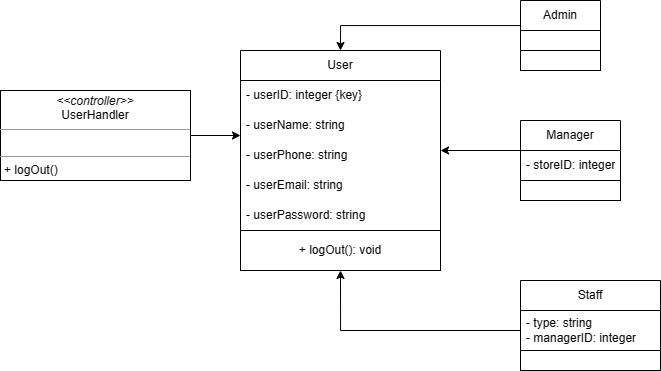
* + 1. *OOD with Communication Diagram*



* + 1. *OOD with Sequence Diagram*

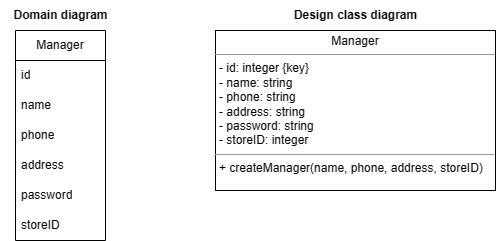


* + 1. *Final Design Class Diagram*

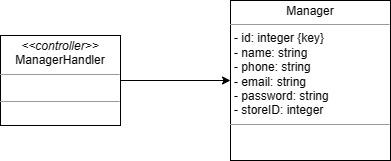


# Design Class for Use Case 3: Create Manager Account

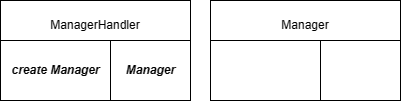
* + 1. Design Class in Detailed Design



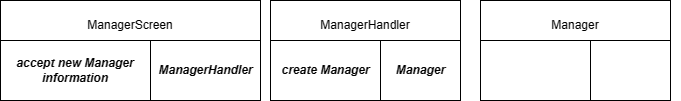
* + 1. Design Class Diagram
       1. Domain Design Class



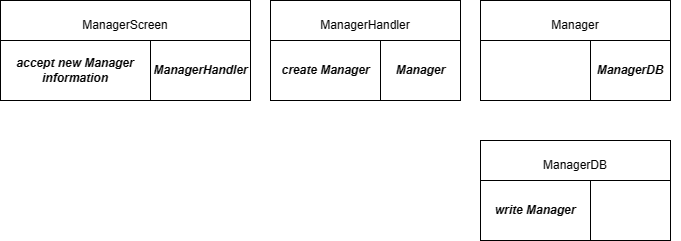
* + - 1. Controller



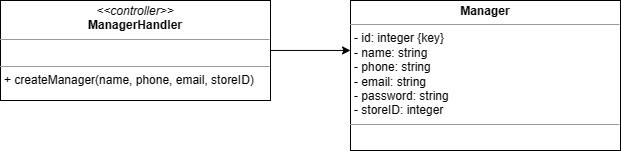
* + - 1. UI



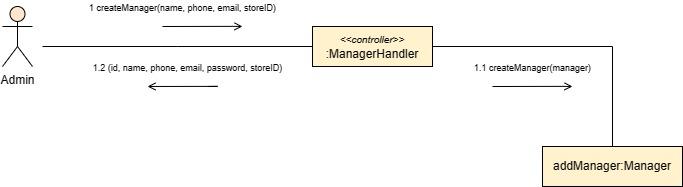
* + - 1. Data Access



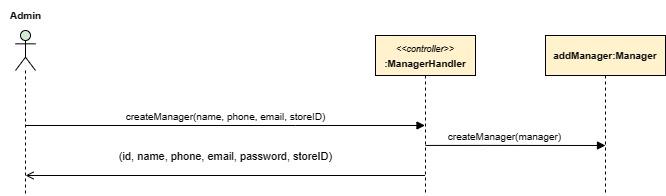
* + - 1. Design Class



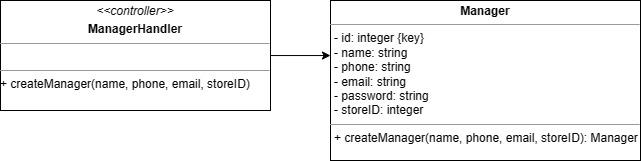
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

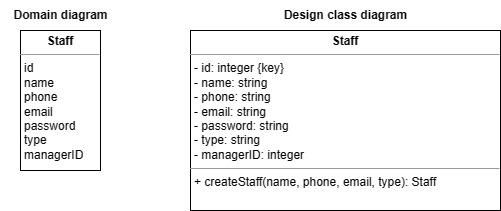


* + 1. Final Design Class Diagram

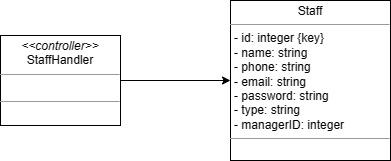


# Design Class for Use Case 4: Create Staff Account

* + 1. Design Class in Detailed Design



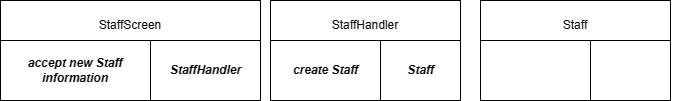
* + 1. Design Class Diagram
       1. Domain Design Class



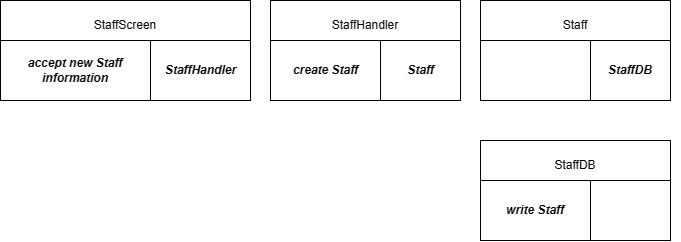
* + - 1. Controller



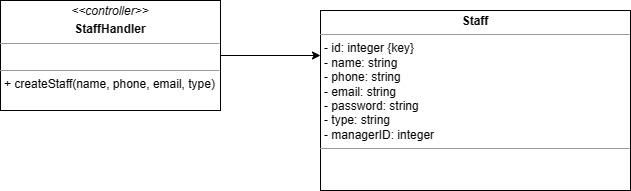
* + - 1. UI



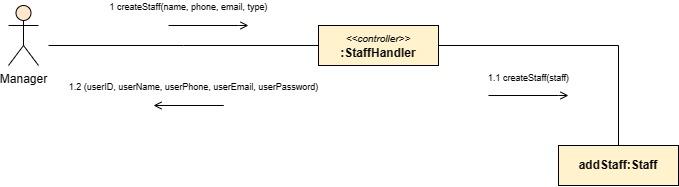
* + - 1. Data Access



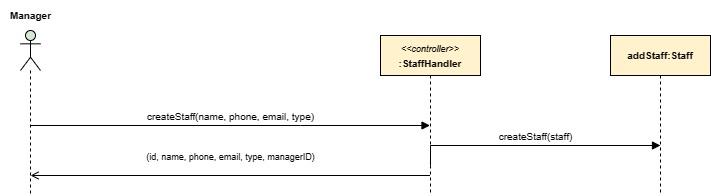
* + - 1. Design Class



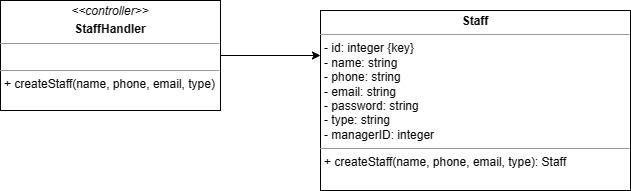
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

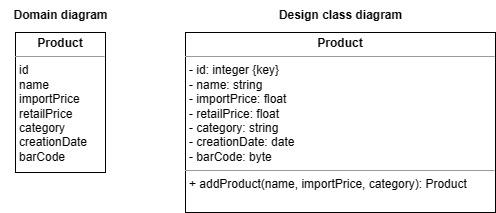


* + 1. Final Design Class Diagram

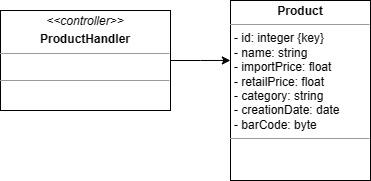


# Design Class for Use Case 5: Add Product

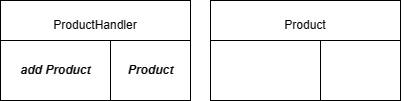
* + 1. Design Class in Detailed Design



* + 1. Design Class Diagram
       1. Domain Design Class



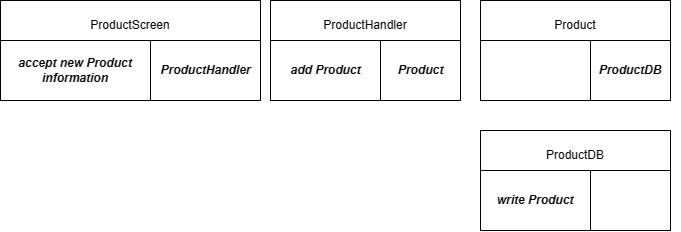
* + - 1. Controller



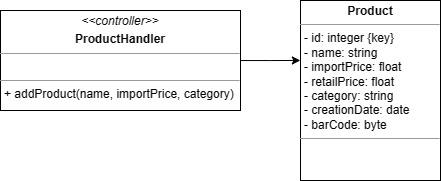
* + - 1. UI



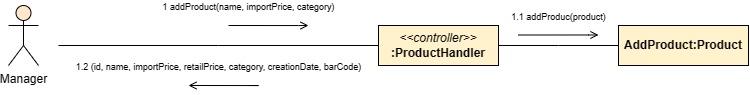
* + - 1. Data Access



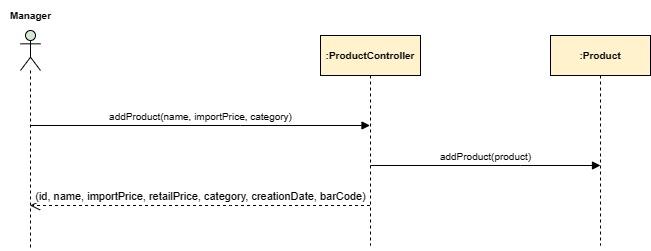
* + - 1. Design Class



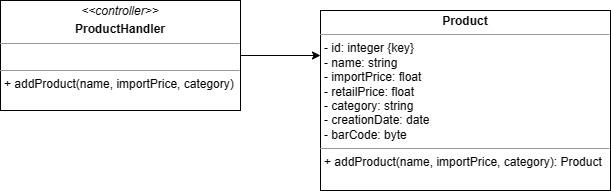
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

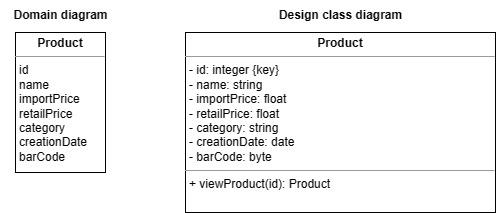


* + 1. Final Design Class Diagram

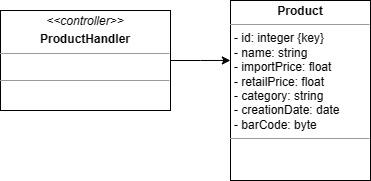


# Design Class for Use Case 6: View Product

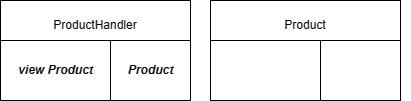
* + 1. Design Class in Detailed Design



* + 1. Design Class Diagram
       1. Domain Design Class



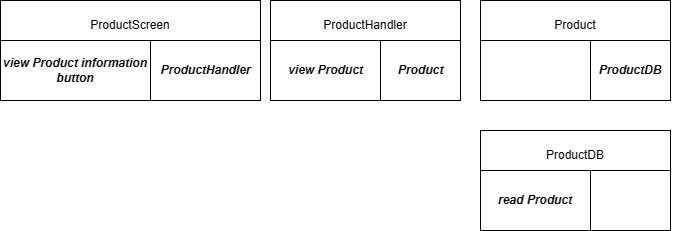
* + - 1. Controller



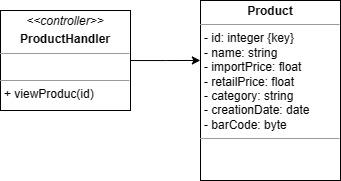
* + - 1. UI



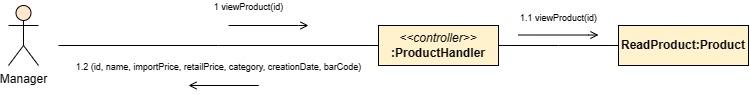
* + - 1. Data Access



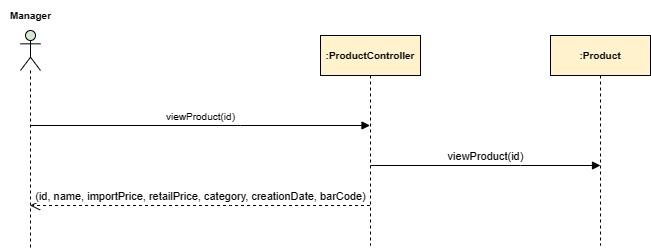
* + - 1. Design Class



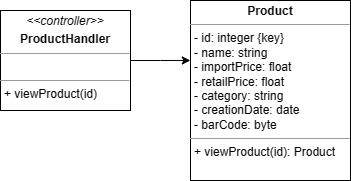
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

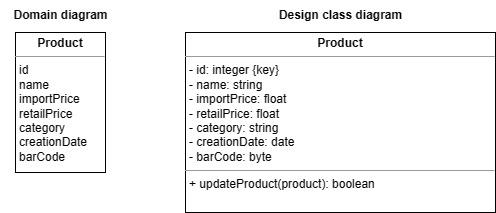


* + 1. Final Design Class Diagram

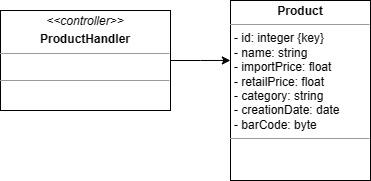


# Design Class for Use Case 7: Update Product Information

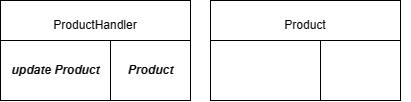
* + 1. Design Class in Detailed Design



* + 1. Design Class Diagram
       1. Domain Design Class



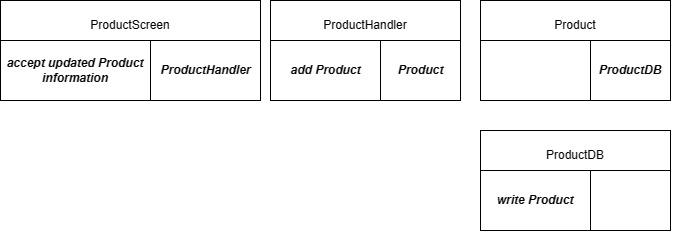
* + - 1. Controller



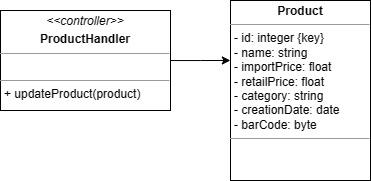
* + - 1. UI



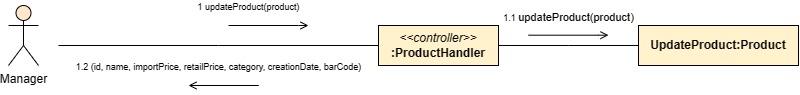
* + - 1. Data Access



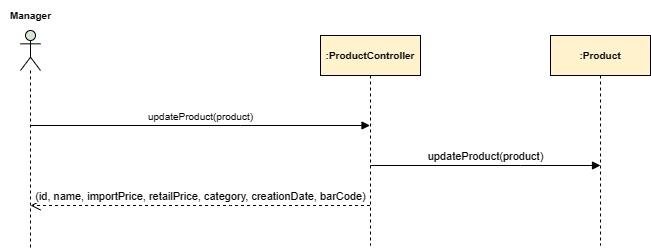
* + - 1. Design Class



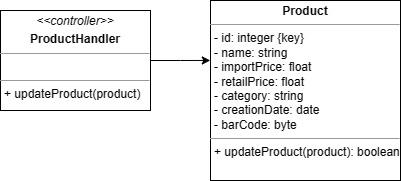
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

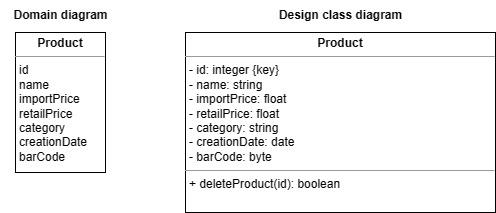


* + 1. Final Design Class Diagram

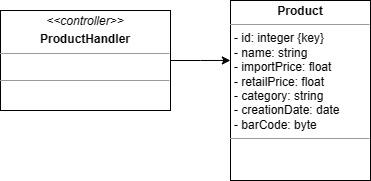


# Design Class for Use Case 8: Delete Product

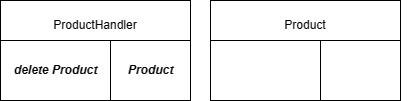
* + 1. Design Class in Detailed Design



* + 1. Design Class Diagram
       1. Domain Design Class



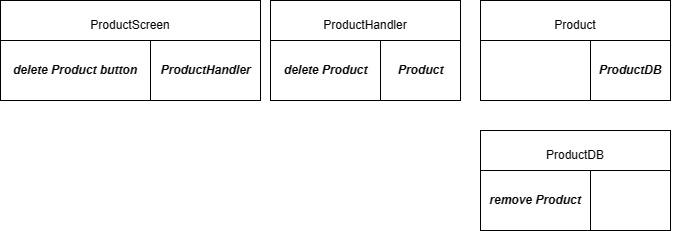
* + - 1. Controller



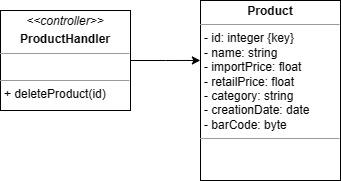
* + - 1. UI



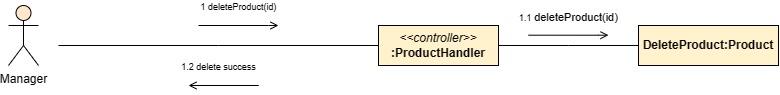
* + - 1. Data Access



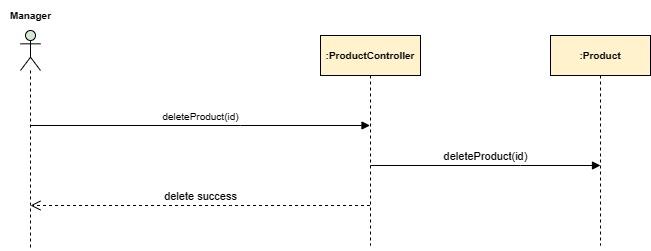
* + - 1. Design Class



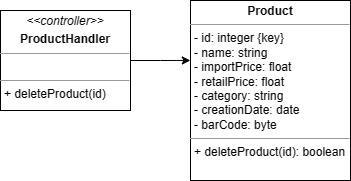
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

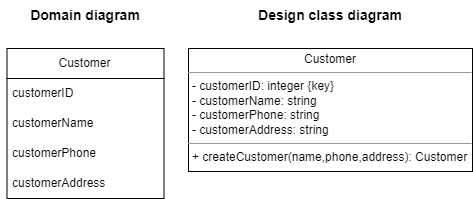


* + 1. Final Design Class Diagram

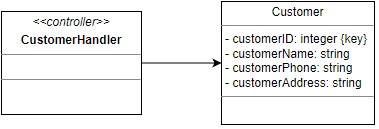


# Design Class for Use Case 9: Create New Customer

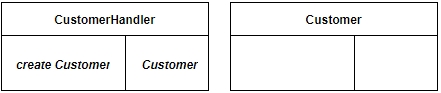
* + 1. Design Class in Detailed Design

****

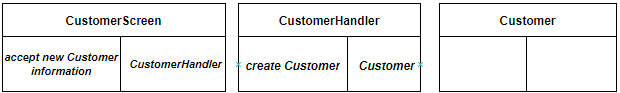
* + 1. Design Class Diagram
       1. Domain Design Class



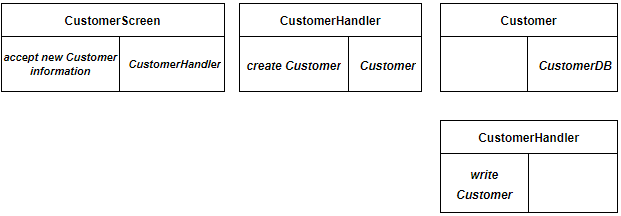
* + - 1. Controller



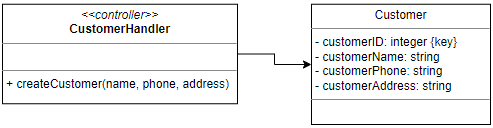
* + - 1. UI



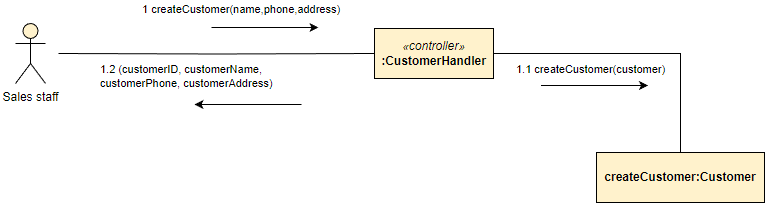
* + - 1. Data Access



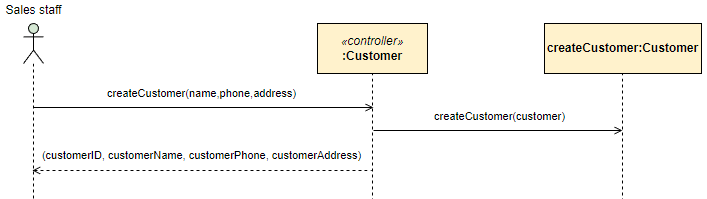
* + - 1. Design Class



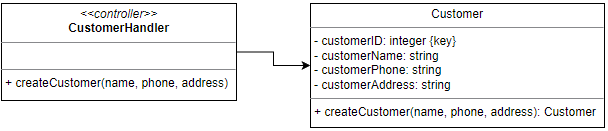
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

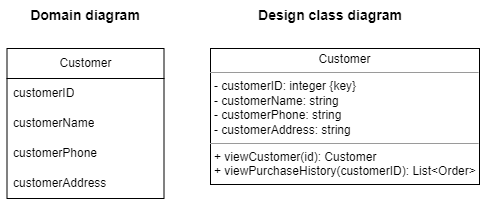


* + 1. Final Design Class Diagram

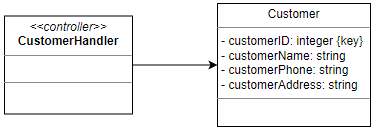


# Design Class for Use Case 10: View Customer & Purchase History

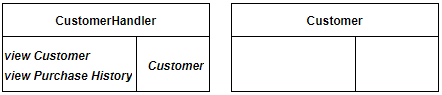
* + 1. Design Class in Detailed Design

****

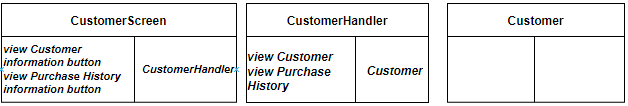
* + 1. Design Class Diagram
       1. Domain Design Class



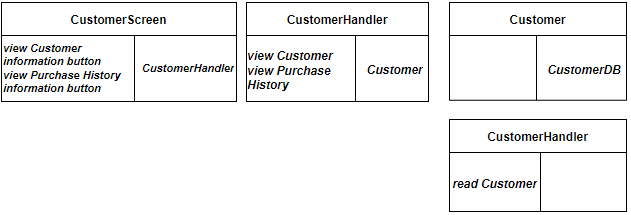
* + - 1. Controller



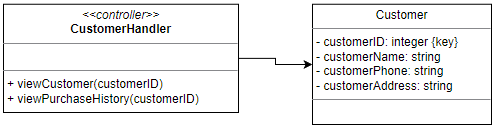
* + - 1. UI



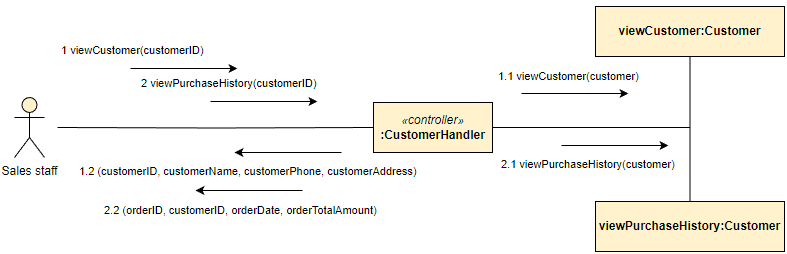
* + - 1. Data Access



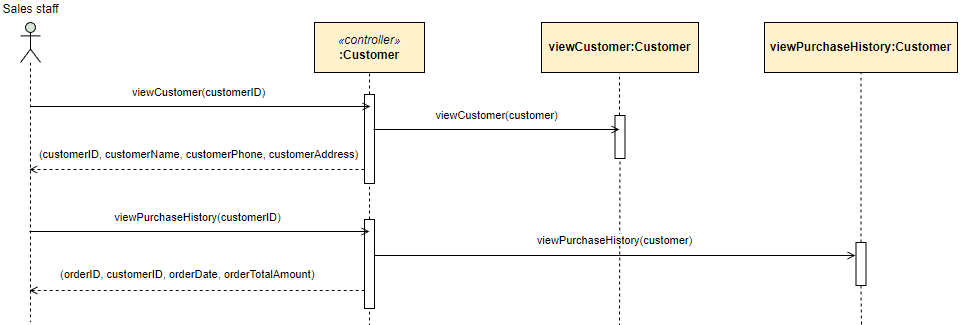
* + - 1. Design Class



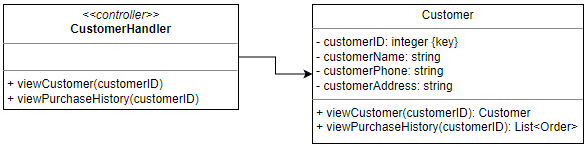
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

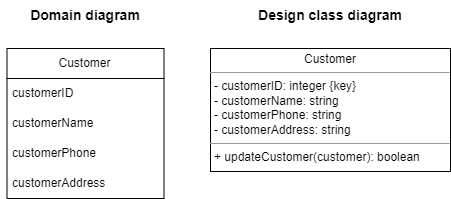


* + 1. Final Design Class Diagram

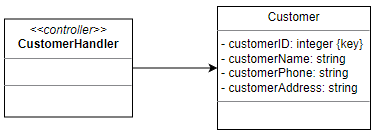


# Design Class for Use Case 11: Update Customer Information

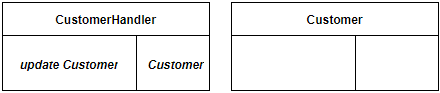
* + 1. Design Class in Detailed Design



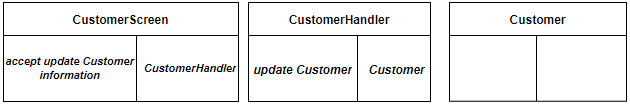
* + 1. Design Class Diagram
       1. Domain Design Class



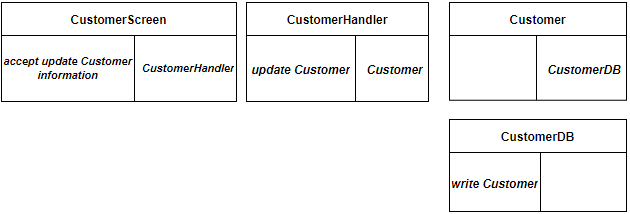
* + - 1. Controller



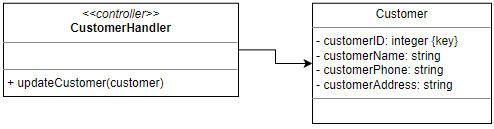
* + - 1. UI



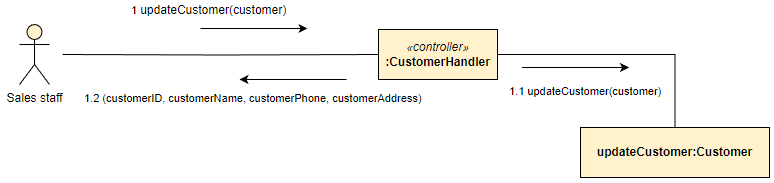
* + - 1. Data Access



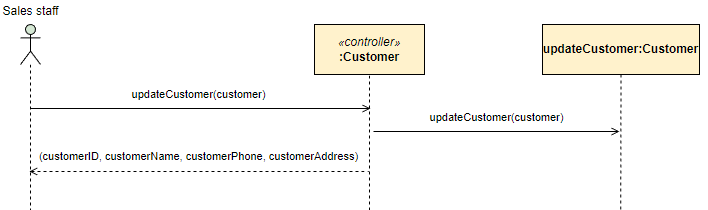
* + - 1. Design Class



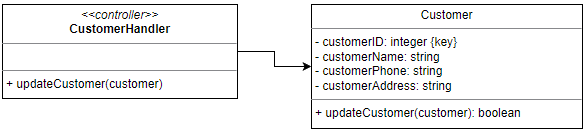
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

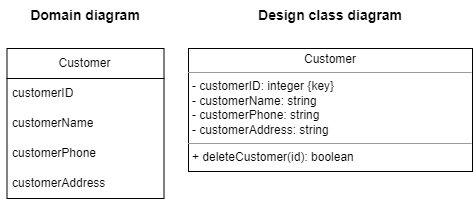


* + 1. Final Design Class Diagram

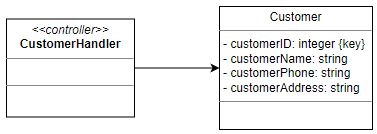


# Design Class for Use Case 12: Delete Customer

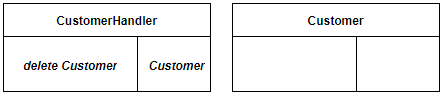
* + 1. Design Class in Detailed Design

****

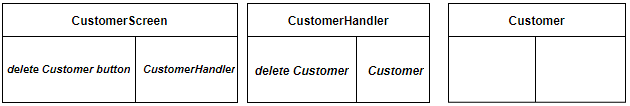
* + 1. Design Class Diagram
       1. Domain Design Class



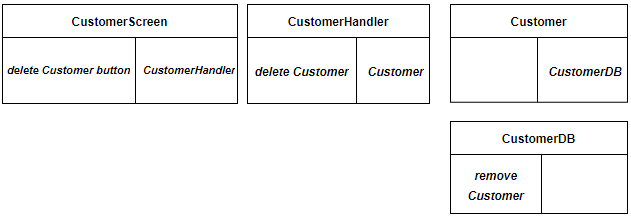
* + - 1. Controller



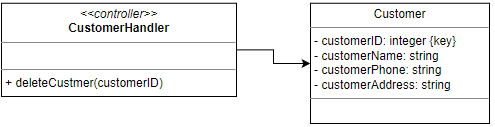
* + - 1. UI



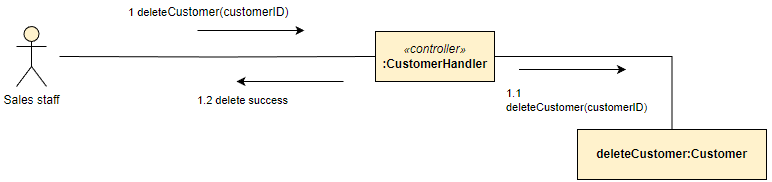
* + - 1. Data Access



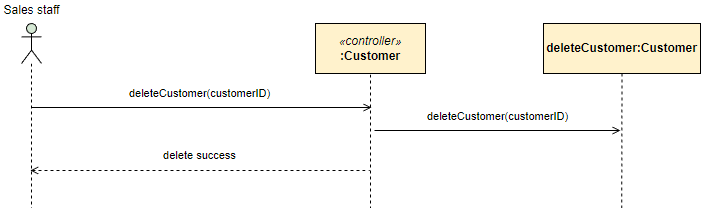
* + - 1. Design Class



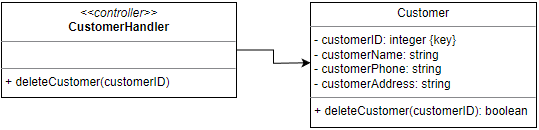
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

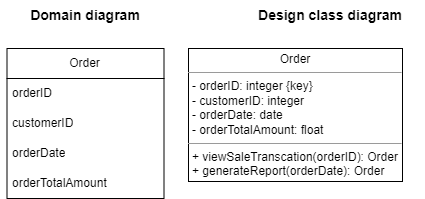


* + 1. Final Design Class Diagram

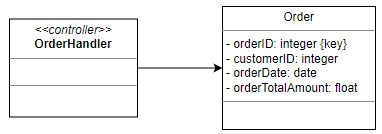


# Design Class for Use Case 13: Manage Sales Transaction

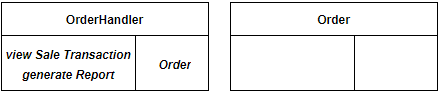
* + 1. Design Class in Detailed Design

****

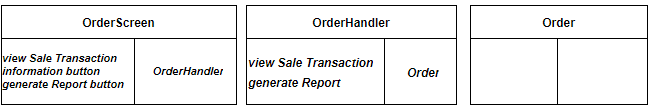
* + 1. Design Class Diagram
       1. Domain Design Class



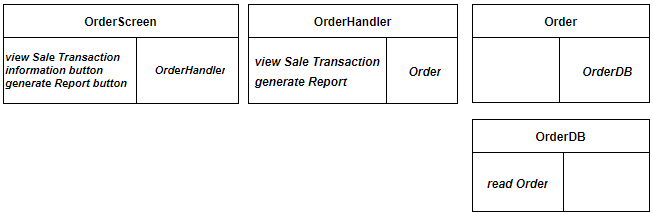
* + - 1. Controller



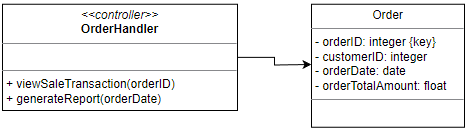
* + - 1. UI



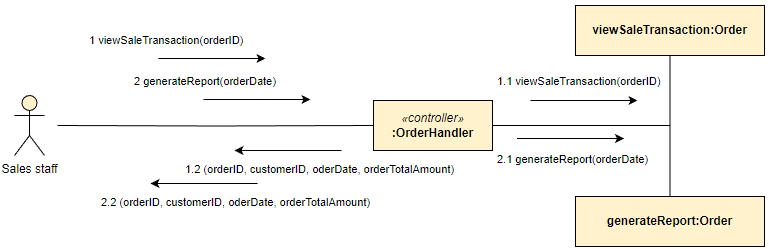
* + - 1. Data Access



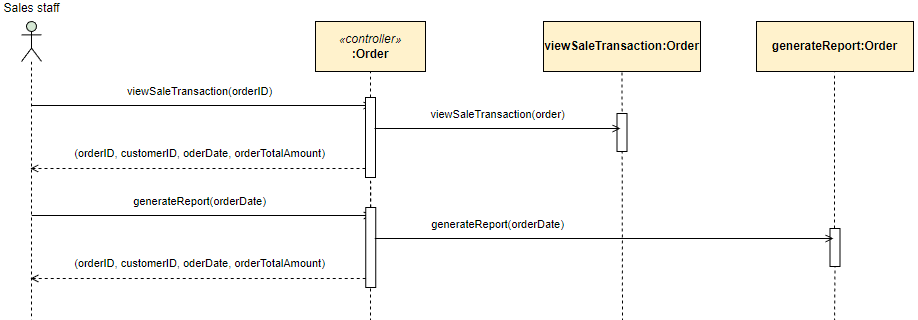
* + - 1. Design Class



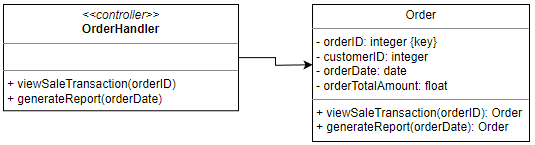
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

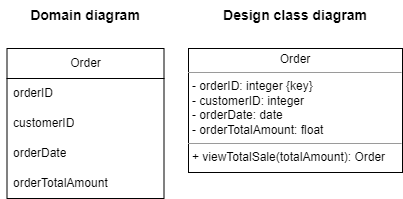


* + 1. Final Design Class Diagram

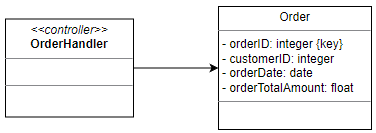


# Design Class for Use Case 14: View Total Sales

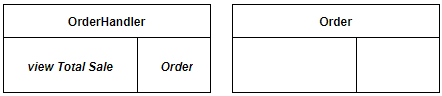
* + 1. Design Class in Detailed Design

****

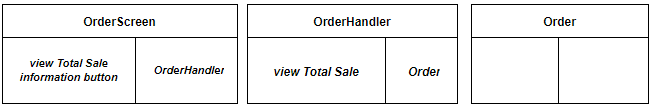
* + 1. Design Class Diagram
       1. Domain Design Class



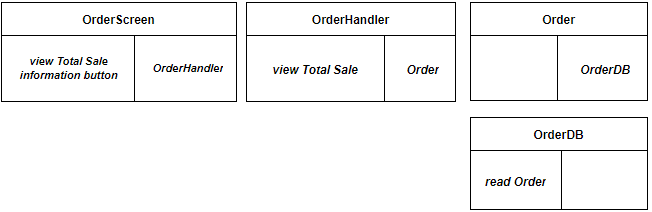
* + - 1. Controller



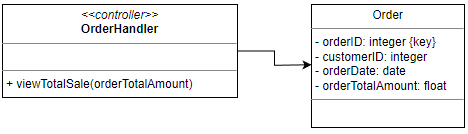
* + - 1. UI



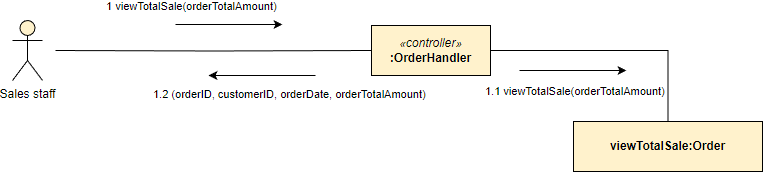
* + - 1. Data Access



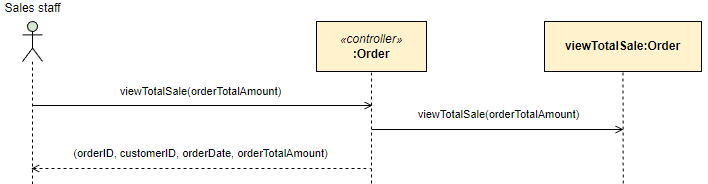
* + - 1. Design Class



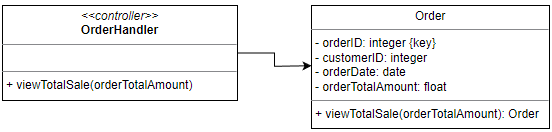
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

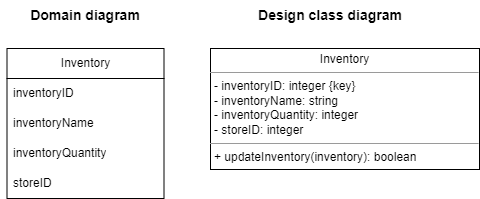


* + 1. Final Design Class Diagram

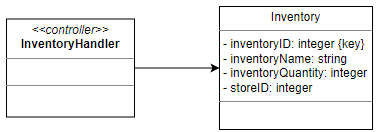


# Design Class for Use Case 15: Manage Inventory

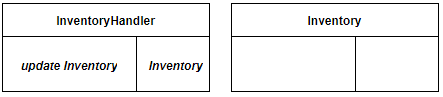
* + 1. Design Class in Detailed Design

****

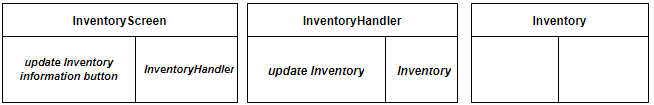
* + 1. Design Class Diagram
       1. Domain Design Class



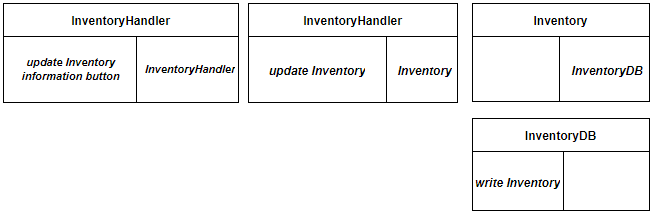
* + - 1. Controller



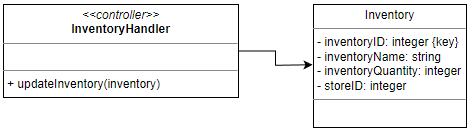
* + - 1. UI



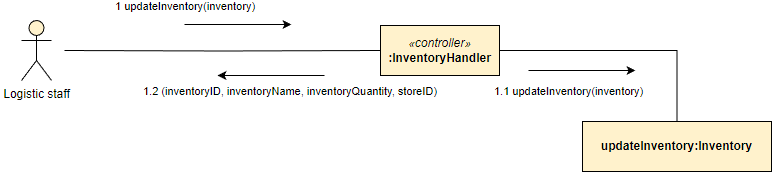
* + - 1. Data Access



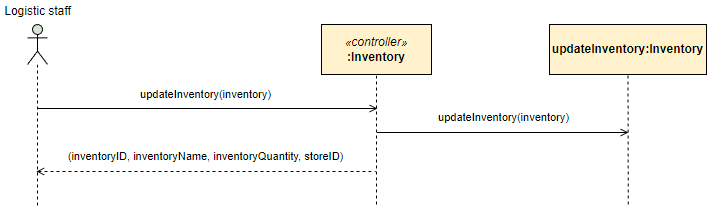
* + - 1. Design Class



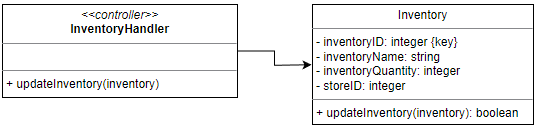
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

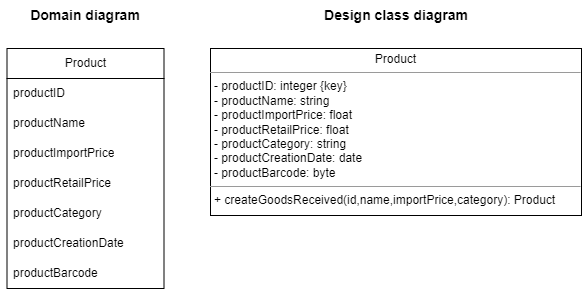


* + 1. Final Design Class Diagram

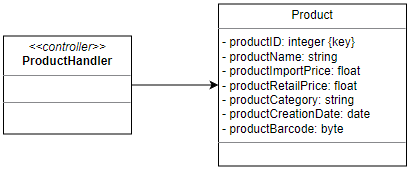


# Design Class for Use Case 16: Create Goods Received

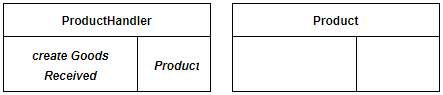
* + 1. Design Class in Detailed Design

****

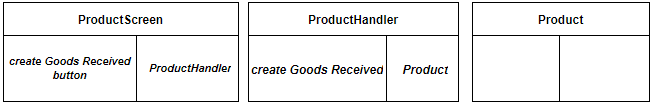
* + 1. Design Class Diagram
       1. Domain Design Class



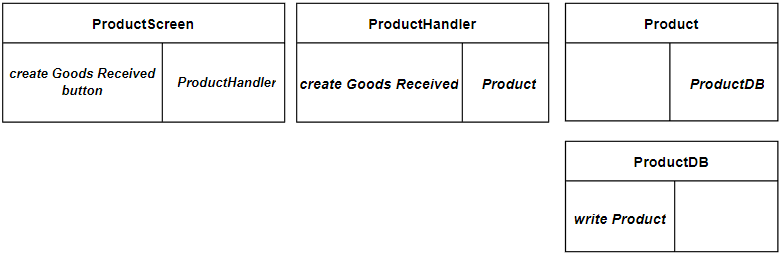
* + - 1. Controller



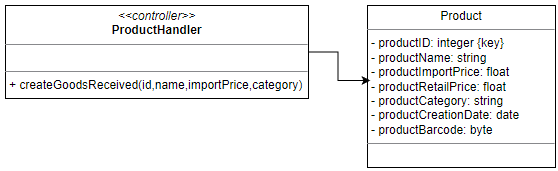
* + - 1. UI



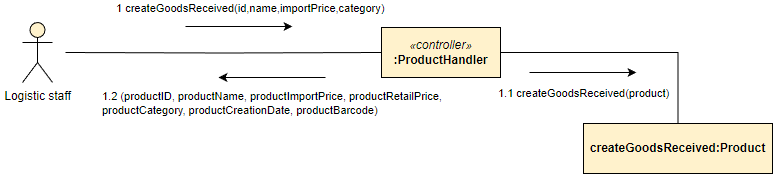
* + - 1. Data Access



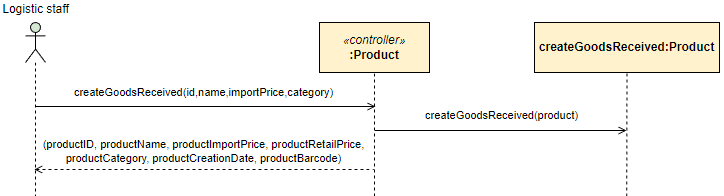
* + - 1. Design Class



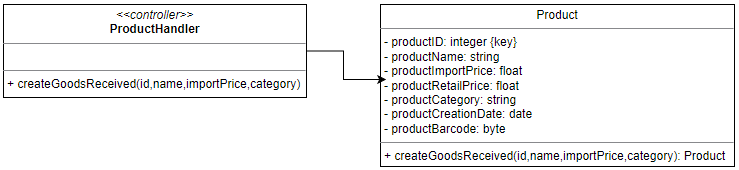
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

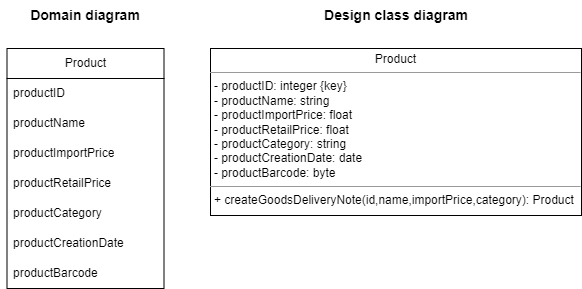


* + 1. Final Design Class Diagram

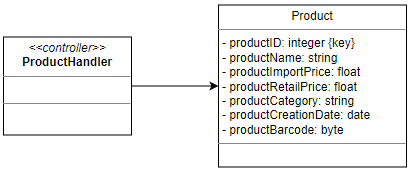


# Design Class for Use Case 17: Create Goods Delivery Note

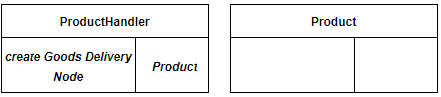
* + 1. Design Class in Detailed Design

****

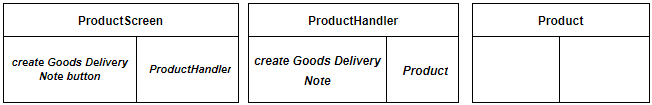
* + 1. Design Class Diagram
       1. Domain Design Class



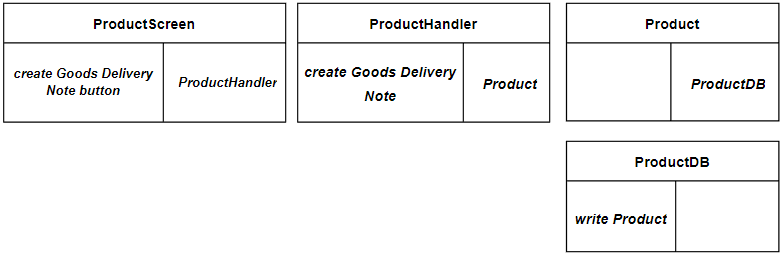
* + - 1. Controller



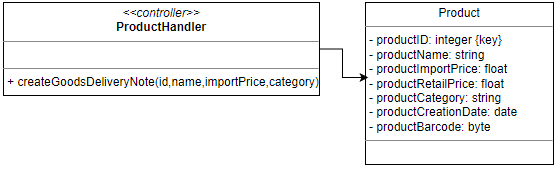
* + - 1. UI



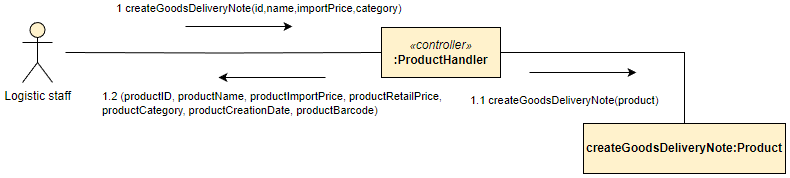
* + - 1. Data Access



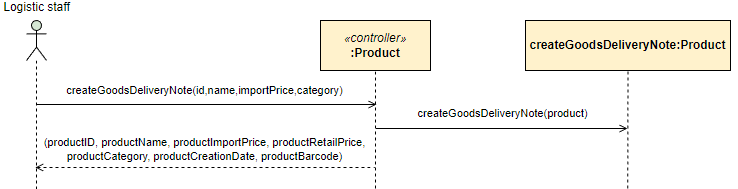
* + - 1. Design Class



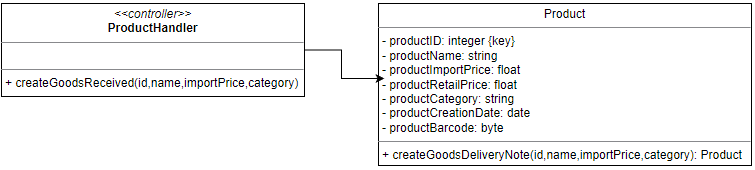
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram

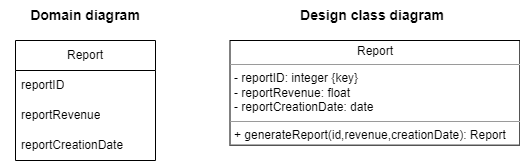


* + 1. Final Design Class Diagram

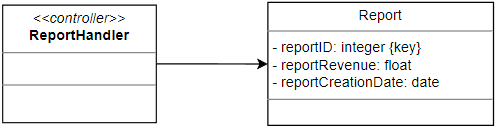


# Design Class for Use Case 18: Generate Report

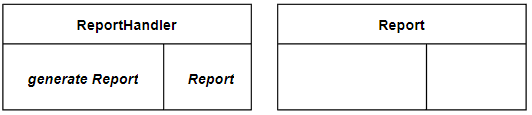
* + 1. Design Class in Detailed Design

****

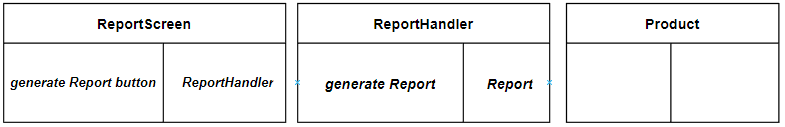
* + 1. Design Class Diagram
       1. Domain Design Class



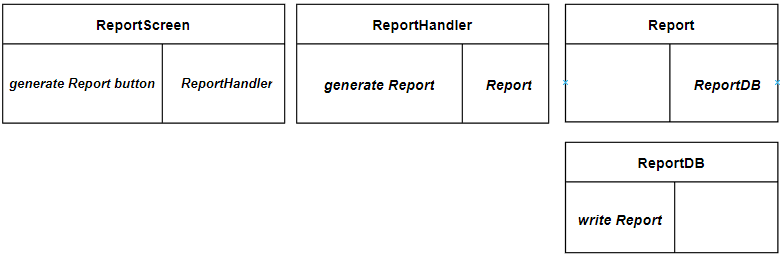
* + - 1. Controller



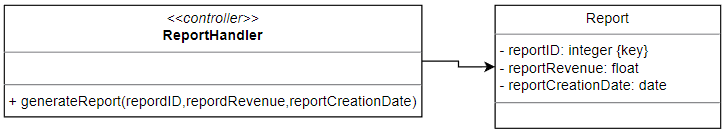
* + - 1. UI



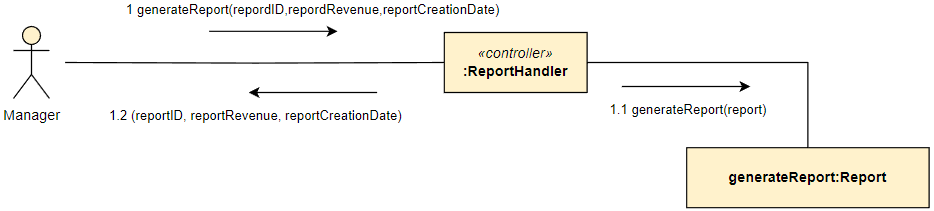
* + - 1. Data Access



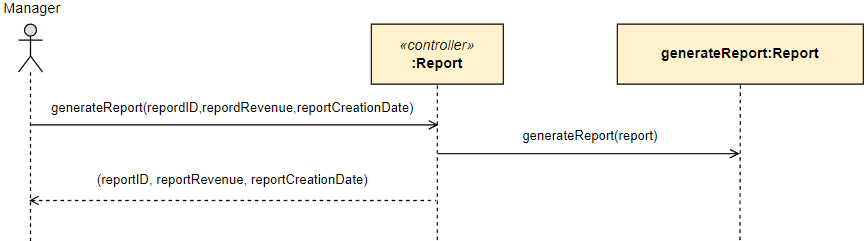
* + - 1. Design Class



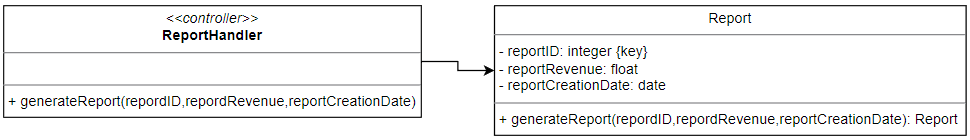
* + 1. OOD with Communication Diagram



* + 1. OOD with Sequence Diagram



* + 1. Final Design Class Diagram



# System Requirements Implementation

# Design Class for Sub System

# Package Diagram for Sub System

# Implementation

# Map persistent objects to the tables in a database

# Modifying sequence diagrams

# UI design

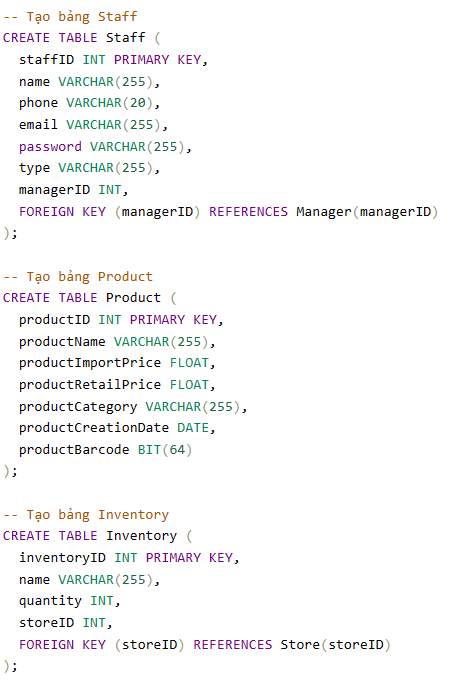
# SQL Code

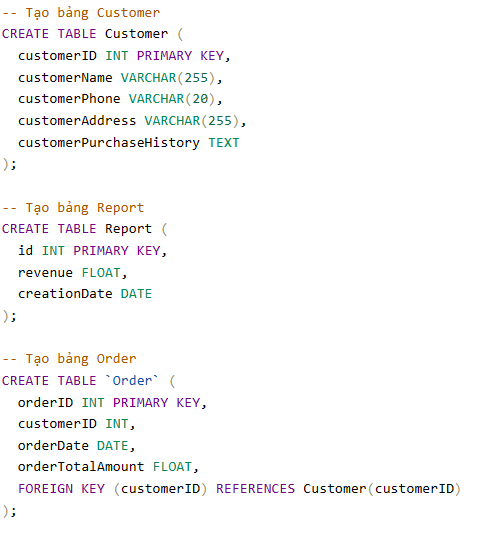
1. Create Database

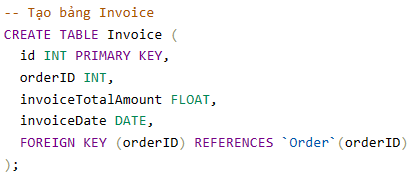


1. Create Table









# Software Classes Method Code

1. Store

import lombok.\*;

import javax.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "store")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Store {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "store\_id")

private int id;

private String name;

public Store(String name) {

this.name = name;

}

}

1. Admin

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "admin")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Admin {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "admin\_id")

private int id;

private String name;

private String phone;

private String email;

private String password;

public Admin(String name, String phone, String email, String password) {

this.name = name;

this.phone = phone;

this.email = email;

this.password = password;

}

}

1. Manager

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "manager")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Manager {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "manager\_id")

private int id;

private String name;

private String phone;

private String email;

private String password;

private int storeID;

public Manager(String name, String phone, String email, String password, int storeID) {

this.name = name;

this.phone = phone;

this.email = email;

this.password = password;

this.storeID = storeID;

}

}

1. Staff

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "staff")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Staff {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "staff\_id")

private int id;

private String name;

private String phone;

private String email;

private String password;

private String type;

private int managerID;

public Staff(String name, String phone, String email, String password, String type, int managerID) {

this.name = name;

this.phone = phone;

this.email = email;

this.password = password;

this.type = type;

this.managerID = managerID;

}

}

1. Product

import lombok.\*;

import javax.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "product")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "product\_id")

private int id;

private String name;

private float importPrice;

private float retailPrice;

private String category;

private Date creationDate;

private byte[] barCode;

public Product(String name, float importPrice, float retailPrice; String category, Date date, byte[] code) {

this.name = name;

this.importPrice = importPrice;

this.retailPrice = retailPrice;

this.category = category;

this.creationDate = date;

this.barCode = code;

}

}

1. Inventory

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "inventory")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Inventory {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "inventory\_id")

private int id;

private String name;

private int quantity;

private int storeID;

public Inventory(String name, int quantity, int storeID) {

this.name = name;

this.quantity = quantity;

this.storeID = storeID;

}

}

1. Customer

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "customer")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Customer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "customer\_id")

private int id;

private String name;

private String phone;

private String address;

private String purchaseHistory;

public Customer(String name, String phone, String address, String purchaseHistory) {

this.name = name;

this.phone = phone;

this.address = address;

this.purchaseHistory = purchaseHistory;

}

}

1. Report

import lombok.\*;

import javax.persistence.\*;

@Entity

@Table(name = "report")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Report {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "report\_id")

private int id;

private float revenue;

private Date creationDate;

public Report(float revenue, Date date) {

this.revenue = revenue;

this.creationDate = date;

}

}

1. Order

import lombok.\*;

import javax.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "order")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Order {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "order\_id")

private int id;

private int customerID;

private Date orderDate;

private float orderTotalAmount;

public Order(int customerID, Date date, float total) {

this.customerID = customerID;

this.orderDate = date;

this.orderTotalAmount = total;

}

}

1. Invoice

import lombok.\*;

import javax.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "invoice")

@Getter

@Setter

@AllArgsConstructor

@NoArgsConstructor

@Data

public class Invoice {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "invoice\_id")

private int id;

private int orderID;

private float totalAmount;

private Date invoiceDate;

public Invoice(int orderID, float total, Date date) {

this.orderID = orderID;

this.totalAmount = total;

this.invoiceDate = date;

}

}

# SYSTEM TESTING, DEPLOYMENT AND DEMONSTRATION

# Testing: Test plan & Test case

# Deployment

# Demonstration

# Conclusions

Overall, my team has roughly outlined the main aspects of retail operations management. It includes admin account management, allowing store managers to create accounts for employees. The product management system allows administrators to perform essential operations such as viewing, adding, updating, and deleting products. Sales staff have limited access to view product listings without modifying content or viewing original prices. Customer management allows creating new customer accounts and tracking their purchase history. Sales management facilitates the creation of invoices and reports for sales transactions, while inventory management ensures effective inventory tracking and control across retail stores. Finally, the reporting system allows generating reports for both the company and individual retail stores.

However, because of the tight schedule and many projects, the team was unable to complete system development on time.

# References

1. *Slide* ***Requirements Modelling***
2. *Book-MultiAuthor, Gra*
3. Book-System Analysis Design with UML\_2nd\_Edition

***Internet Link:***

1. *<http://www.uml-dox.net/Addison.Wesley-UML.for.Mere.Mo/0321246241/ch02lev1sec6.html>*
2. [UML Class Diagram Tutorial (visual-paradigm.com)](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/)
3. [What is Architecture Diagramming? - Software & System Architecture Diagramming Explained - AWS (amazon.com)](https://aws.amazon.com/what-is/architecture-diagramming/)