

**Revision and Sign Off Sheet**

**Change Record**

| **Author** | **Version** | **Change reference** | **Date** |
| --- | --- | --- | --- |
| Tran Le Yen Linh | 0.1.0 | Complete diagram | 10/11/2024 |
| Pham Nguyen Minh Khoa | 0.1.1 | Edit diagram | 20/11/2024 |
| Tran Le Yen Linh | 0.1.2 | Complete business rule | 16/12/2024 |

**Reviewers**

| **Name** | **Company** | **Version** | **Position** | **Date** |
| --- | --- | --- | --- | --- |
| Nguyen Hoang Duy |  | 0.1.0 | Project Owner | 05/12/2024 |
| Nguyen Hoang Duy |  | 0.1.2 | Project Owner | 17/12/2024 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Table of Contents**

# Introduction

# Purpose

This Software Requirements Specification and Design Document contains the software requirements for building a warehouse, logistics, and fleet management system. Firstly, this document along with the Notes database(s) and other reference documents are the complete requirements to carry out the process of building a warehouse, logistics and fleet management system. Second, technically, it defines how applications will function. Developers will rely on this document to plan development, assign tasks and deploy new applications.

# Scope

This document is prepared for the application Supply Chain, in scope of the project Supply Chain.

# Intended Audiences and Document Organization

This document is intended for:

* Development team: Responsible to develop detailed design, implement and perform unit test, integration test and system test for the migrated application
* Data Migration team: Responsible to create data migration scripts, and perform data migration for the application.
* Documentation Team: Responsible to writing User Guide for the application.
* UAT team: Responsible to conduct user acceptance test sessions with end users.

Below are main sections of the document:

* **1. Introduction**: This section describes the general introduction of this document.
* **2. Functional Requirements**: This section describesthe functional requirements in detail.
* **3. Non-functional Requirements:** This section describes the non-functional requirements of this application such as user access and security, interfaces, screens and performance.
* **4. Other Requirements:** This section describes other requirements such as archive or security audit function.
* **5.** **SharePoint Application Design:** This section describes the design of SharePoint application.
* **6. Appendixes**: This section describes other requirements for this application and other supporting information for this document**.**
* NOTE:Please refer to section 6.1 for all acronyms and abbreviations you may encounter within this document.

# References

| **#** | **Title** | **Version** | **File Name / Link** | **Description** |
| --- | --- | --- | --- | --- |
| 1 |  |  |  |  |
| 2 |  |  |  |  |

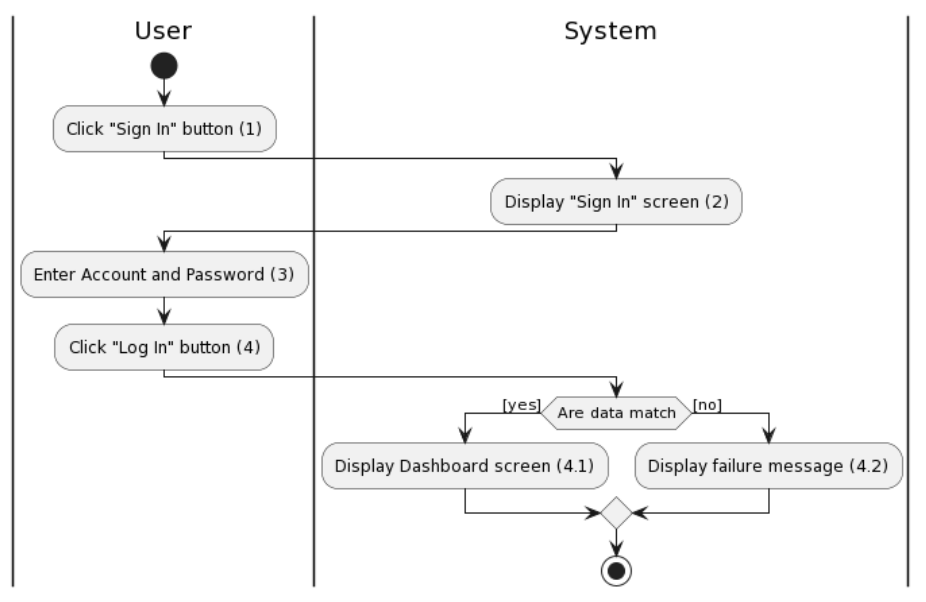
# Functional Requirements

# Use Case Description

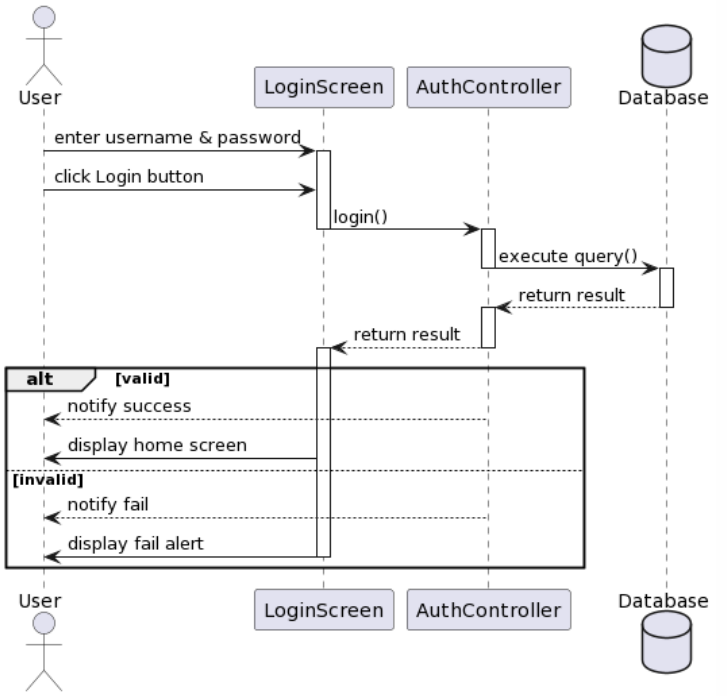
### UC0: Sign In

| **Name** | **Sign In** |
| --- | --- |
| **Description** | This use case allows users to log in to the system to access their personalized settings and features. |
| **Actor** | User |
| **Trigger** | * The user selects the option to sign in. |
| **Pre-condition** | * The user must be registered with the system. * The screen is Login Screen. |
| **Post-condition** | * **Successful**: The user gains access to the system and is redirected to their dashboard. * **Failure**: Corresponding error notification. |

#### Activities Flow



*Figure 1: Activities Flow*

**

*Figure 2: Sequence diagram*

#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system loads the “Sign In” screen. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the 'Login' button and waits for a response from the Server: The 'Login' button changes to a 'Loading' state. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * When the user presses the "Login" button, the Client confirms whether the [Username] and [Password] are in the correct format of the system.   Format of [Username]:  [Username] must be a set of 6-12 characters including at least 1 uppercase letter, 1 special character, and 1 digit.  Format of [Password]:  [Password] must be a set of 6 numeric characters ranging from 0 to 9.   * **Validation Steps**: * Account Validation:   If [Username].isEmpty() , system showMessage(**MSG1**) .  Else if [Username] is not in the correct format, system showMessage(**MSG2**).  Else, accept [Username].   * Password Validation:   If [Password].isEmpty(), system showMessage(**MSG6**).  Else if [Password] is not in the correct format, system showMessage(**MSG3**).  Else, accept [Password].   * Client Request to Server:   When both [Username] and [Password] are accepted, the Client sends a login request to the Server with [Account] and [Password].   * Server Response:   The Server confirms if the submitted [Account] matches any account in the User table.  If it’s no match, the Server sends [AuthenticationSuccess = false] and the Client , system showMessage(**MSG4**).  If it’s a match, the Server further confirms if the submitted [Password] matches the password of the User with the aforementioned account.  If the password matches, the Server sends [AuthenticationSuccess = true] and the Client transitions to the Home Screen, recognizing the current identity as that of the logged-in account.  If the password does not match, the Server sends [AuthenticationSuccess = false] and the Client system showMessage(**MSG5**) . |

### 

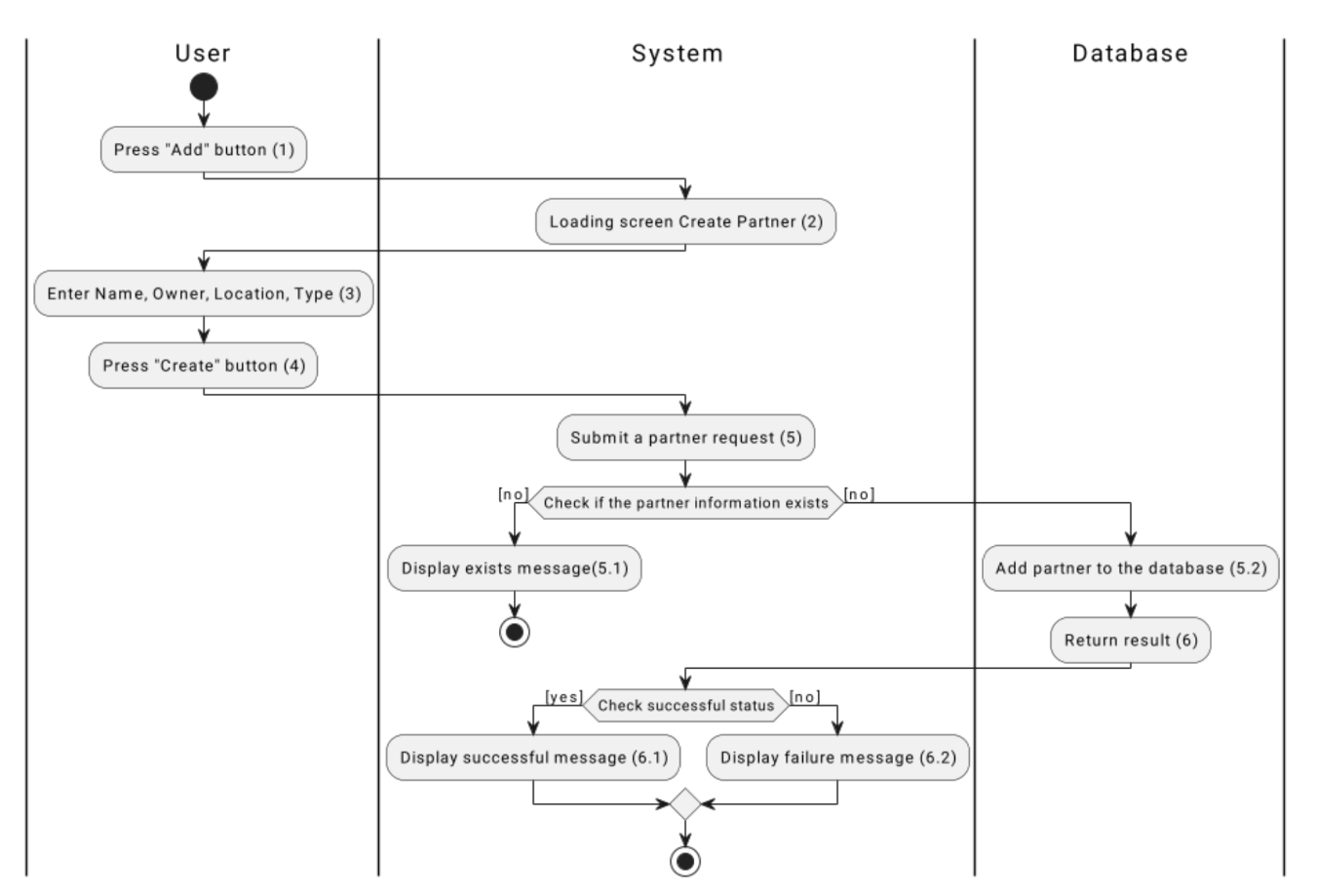
### UC1: Create Partner

### 

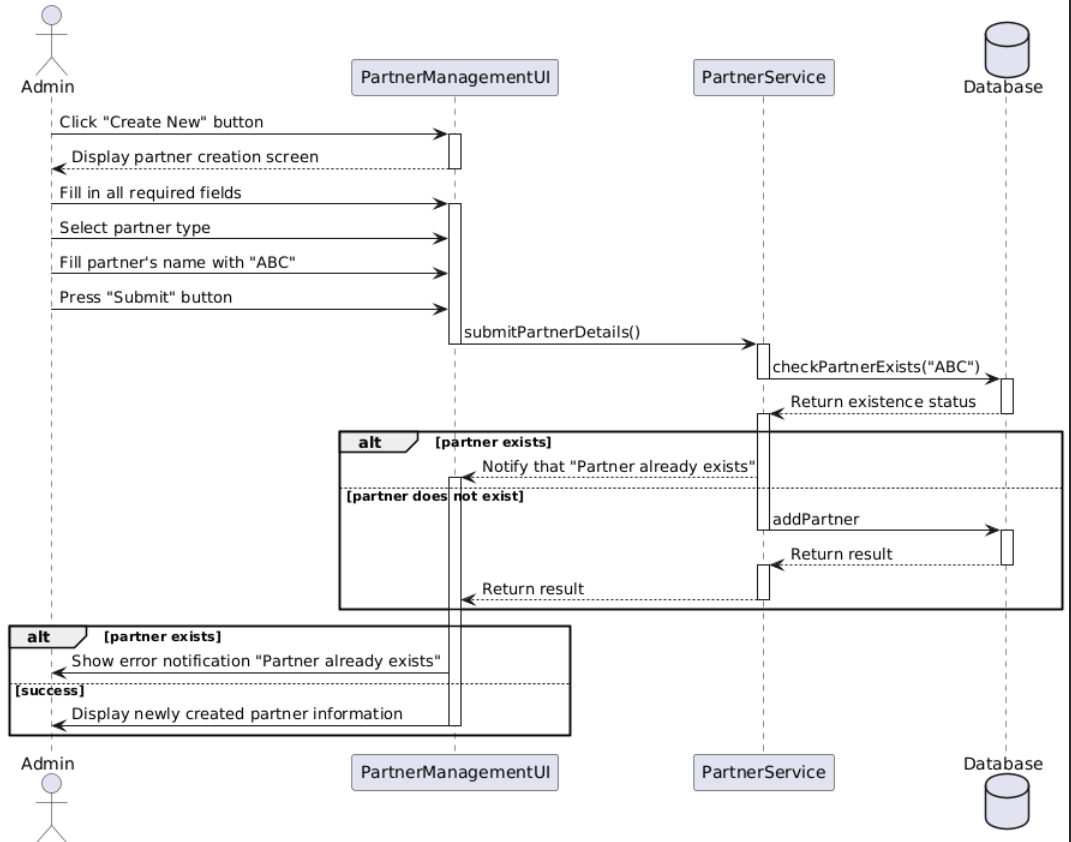
| **Name** | **Create Partner Information** |
| --- | --- |
| **Description** | This use case allows the user to create a new partner in the partner management system by entering all required information. The system validates the input to ensure completeness and uniqueness before saving the information to the database. |
| **Actor** | Admin, Partner Manager, Warehouse Manager, Logistic Manager |
| **Trigger** | The user access to Partner Management Page:   * Clicks the "Create New" * Fill in all required fields for the partner |
| **Pre-condition** | * The user is authenticated and has access to the partner management page. |
| **Post-condition** | * A new partner including its details (ID, Name, Type), is added to partner list and save in the system's database |

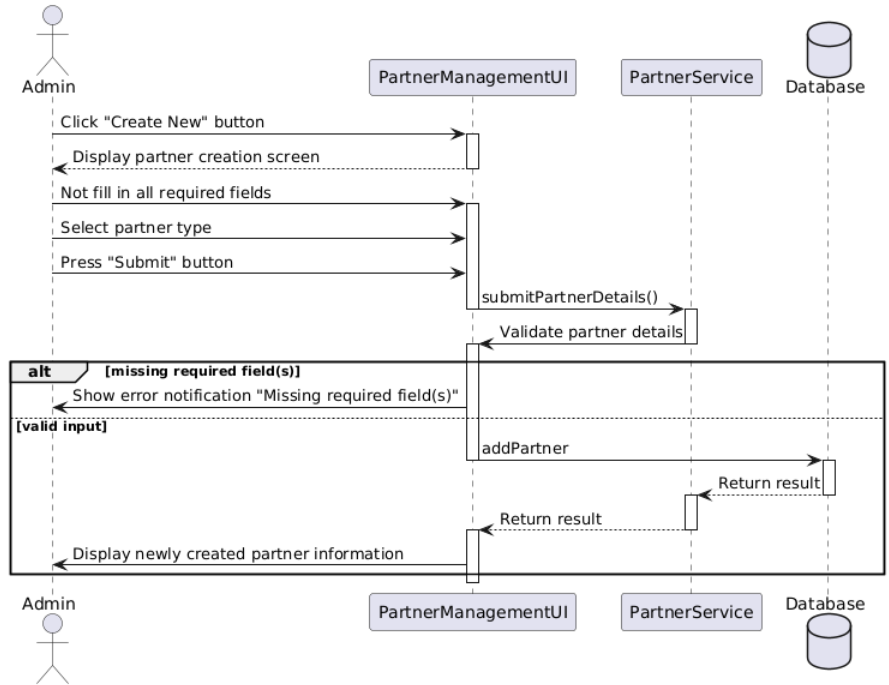
#### 

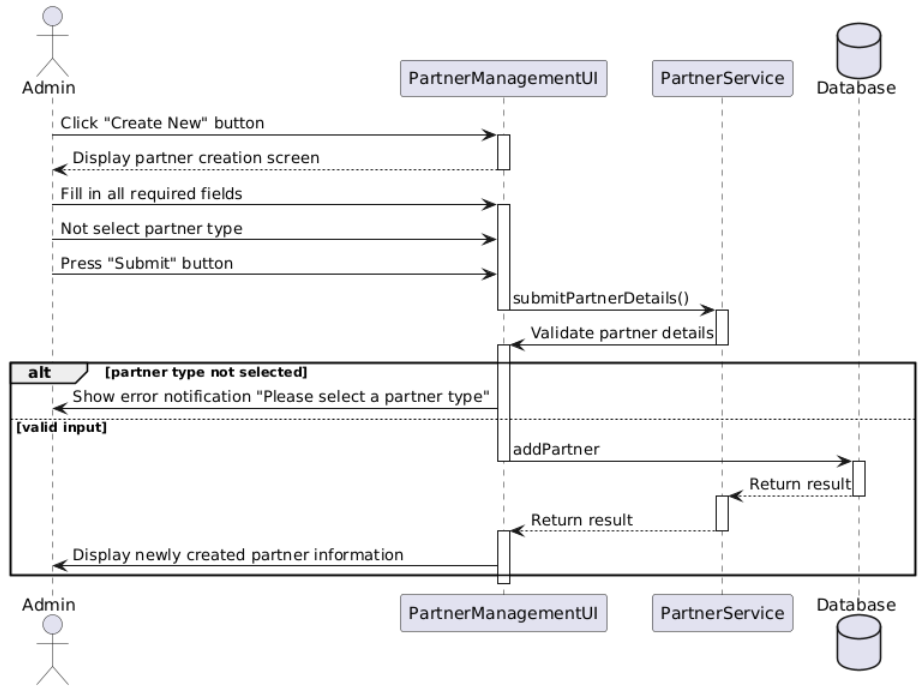
### Activities Flow

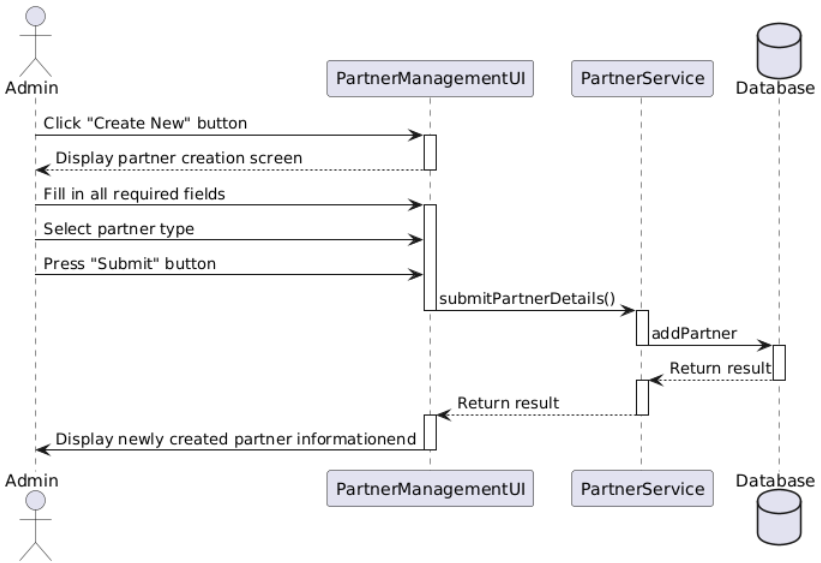


*Figure 1: Activities Flow*

**

**

**

**

#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **"Create Partner Information"** screen when the user clicks the **"Add"** button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the **"Create"** button, the system displays a loading indicator while waiting for a response from the server.. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the **"Create"** button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(Name)] && [checkIsEmpty(Owner)] && [checkIsEmpty(Location)] && [checkIsEmpty(Type)]  If [Name].isEmpty() && [Owner].isEmpty() && [Location].isEmpty() && [Type].isEmpty(), the system shows message **(MSG6)**  Else, do nothing   * Duplicate Validation:   The system checks for existing partners using:  [checkDuplicate(Name, Type)]  If [Name] and [Type] already exist in the database, the system shows message **(MSG7)**   * Data Submission:   If input is valid and no duplicates exist, the system saves the partner data.  On success, the system shows message **(MSG8)**  On failure, the system shows message **(MSG9)** |

### 

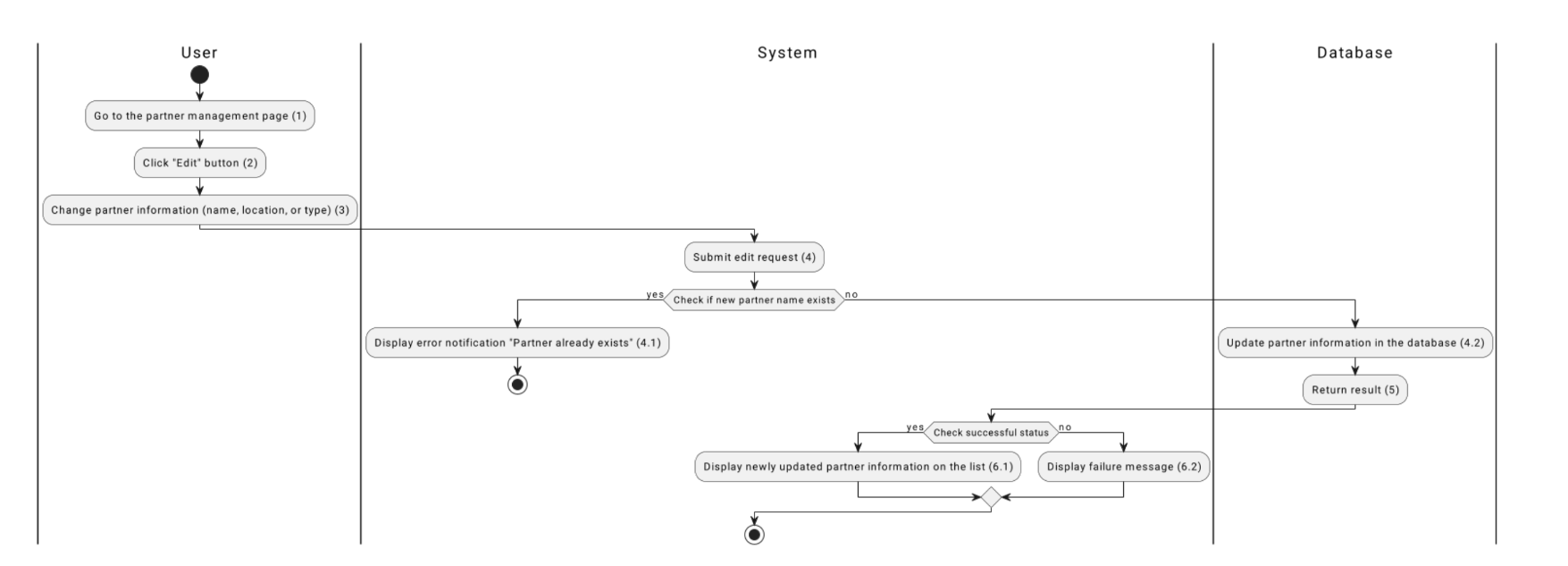
### UC2: Edit Partner

### 

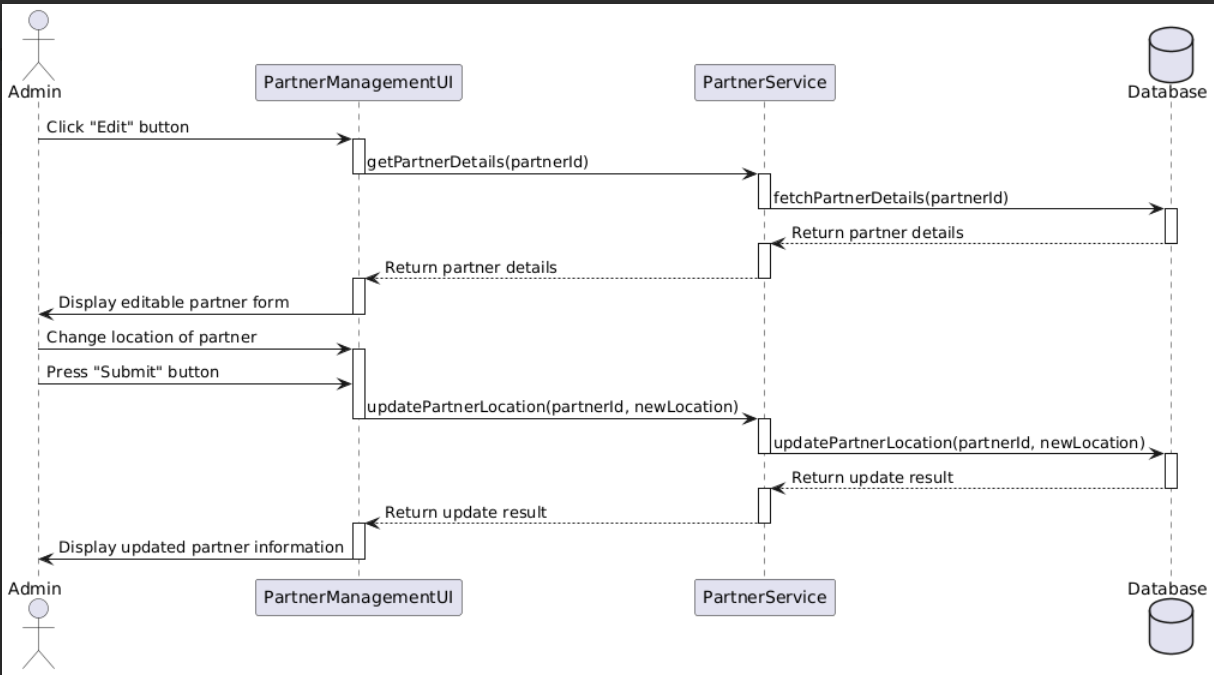
| **Name** | **Edit Partner** |
| --- | --- |
| **Description** | This use case allows the user to edit existing partner information within the partner management system. The user can change details such as location, type, and name. The system validates the input to ensure uniqueness of the partner name before saving any updates to the database. |
| **Actor** | Admin, Partner Manager, Warehouse Manager, Logistic Manager |
| **Trigger** | The user accesses the Partner Management Page:   * Clicks the "Edit" button on a selected partner, * Modifies one or more fields for the partner. |
| **Pre-condition** | The user must be logged in.  There is an existing partner record in the system. |
| **Post-condition** | The partner's information is updated in the database, reflecting changes made by the user, except if there is an error (e.g., duplicate name), in which case the original data remains unchanged. |

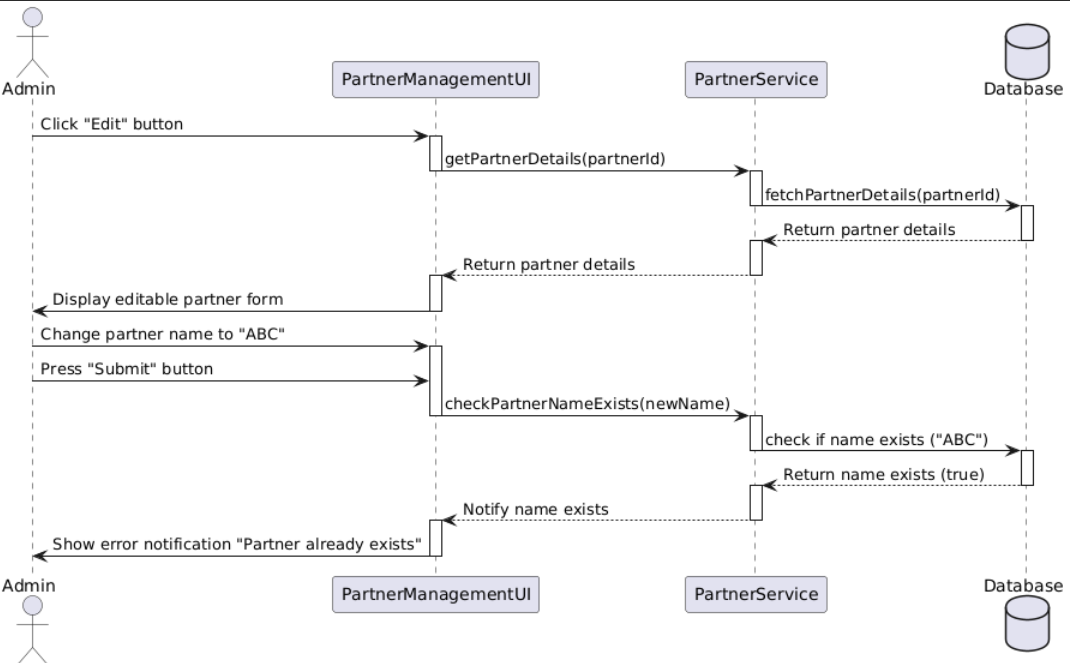
### 

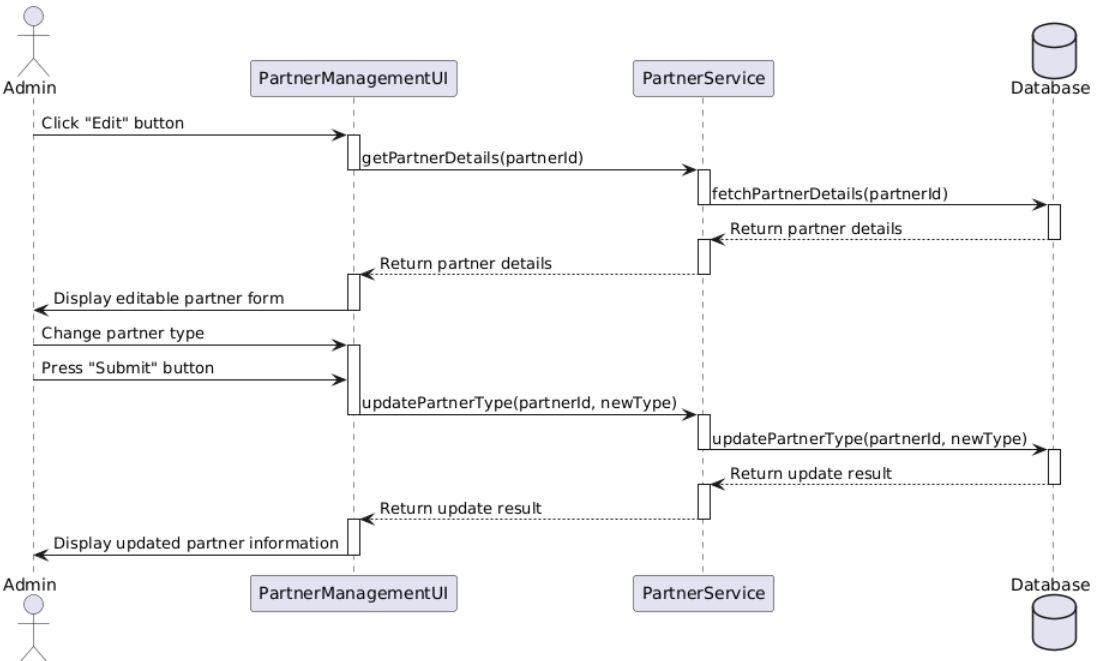
### Activities Flow



*Figure 1: Activities Flow*

**

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **"Edit Partner"** screen when the user clicks the **"Edit"** button on a selected partner record. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the **"Submit"** button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the **"Submit"** button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(Name)] && [checkIsEmpty(Location)] && [checkIsEmpty(Type)]  If [Name].isEmpty() || [Location].isEmpty() || [Type].isEmpty(), the system shows message **(MSG10)**  Else, do nothing.   * Duplicate Validation:   The system checks for existing partners using:  [checkDuplicate(Name, Type)]  If [Name] and [Type] already exist in the database, the system shows message **(MSG11)**   * Data Submission:   If input is valid and no duplicates exist, the system updates the partner data.   * On success, the system shows message **(MSG12)** * On failure, the system shows message **(MSG13)** |

### 

### UC3: Delete Partner

### 

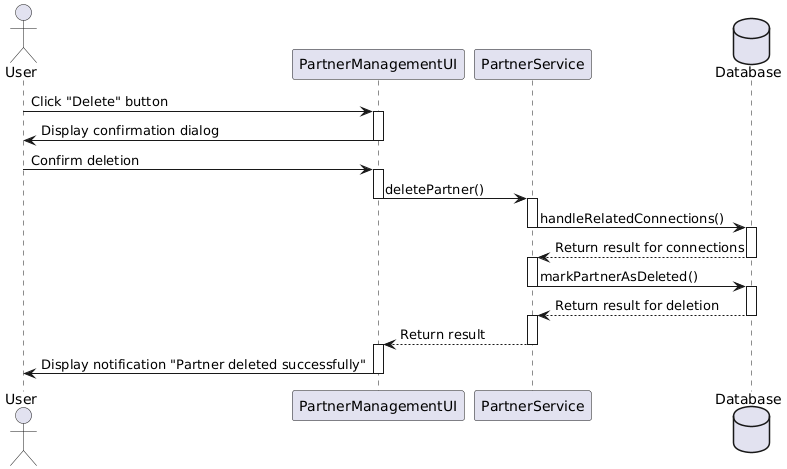
| **Name** | **Delete Partner Information** |
| --- | --- |
| **Description** | This use case allows the user to delete a partner’s information within the partner management system. The deletion process involves handling all related data (such as orders, products, and shipments) associated with the partner. Only after all dependencies are properly managed will the system successfully delete the partner by changing their status to "Deleted." |
| **Actor** | Admin, Partner Manager, Warehouse Manager, Logistic Manager |
| **Trigger** | The user accesses the Partner Management Page:   * Clicks the "Delete" button for a selected partner, * Confirms the deletion request. |
| **Pre-condition** | * The user must be login. * The partner to be deleted has existing records in the system, including dependencies like orders, products, or shipments. |
| **Post-condition** | * If successful, the partner’s status is updated to "Deleted," and a confirmation notification ("Partner deleted successfully") is displayed. * If unsuccessful, an error message will notify the user to handle remaining dependencies before proceeding. |

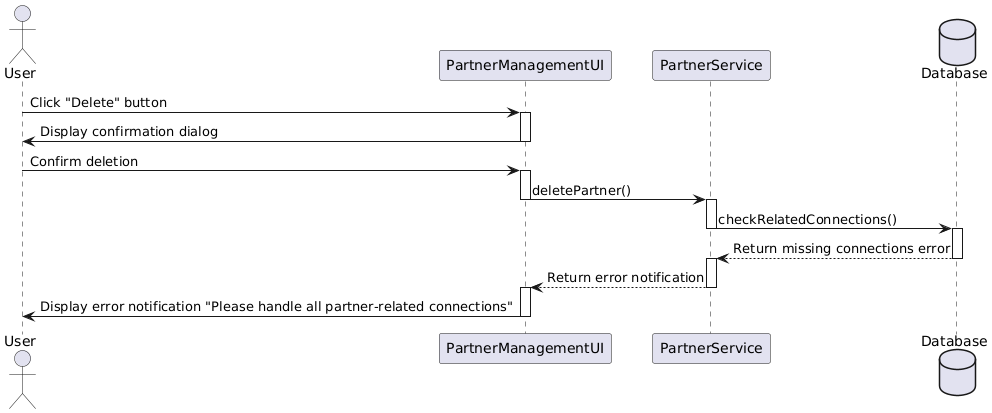
### 

### Activities Flow



*Figure 1: Activities Flow*

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Delete Partner Information** screen when the user clicks the **"Delete"** button for a selected partner. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user confirms the deletion request, the system displays a **loading indicator** while processing the deletion. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system checks for unresolved dependencies (e.g., orders, products, shipments) associated with the partner:   * Validation Step:   [checkDependencies(PartnerID)]  If unresolved dependencies exist, the system shows the message **(MSG14)**   * If all dependencies are resolved, the system updates the **partner status** to **"Deleted"**: * Submission Step:   If the deletion is successful, the system shows the message **(MSG15)**  If the deletion fails, the system shows the message **(MSG16)** |

### 

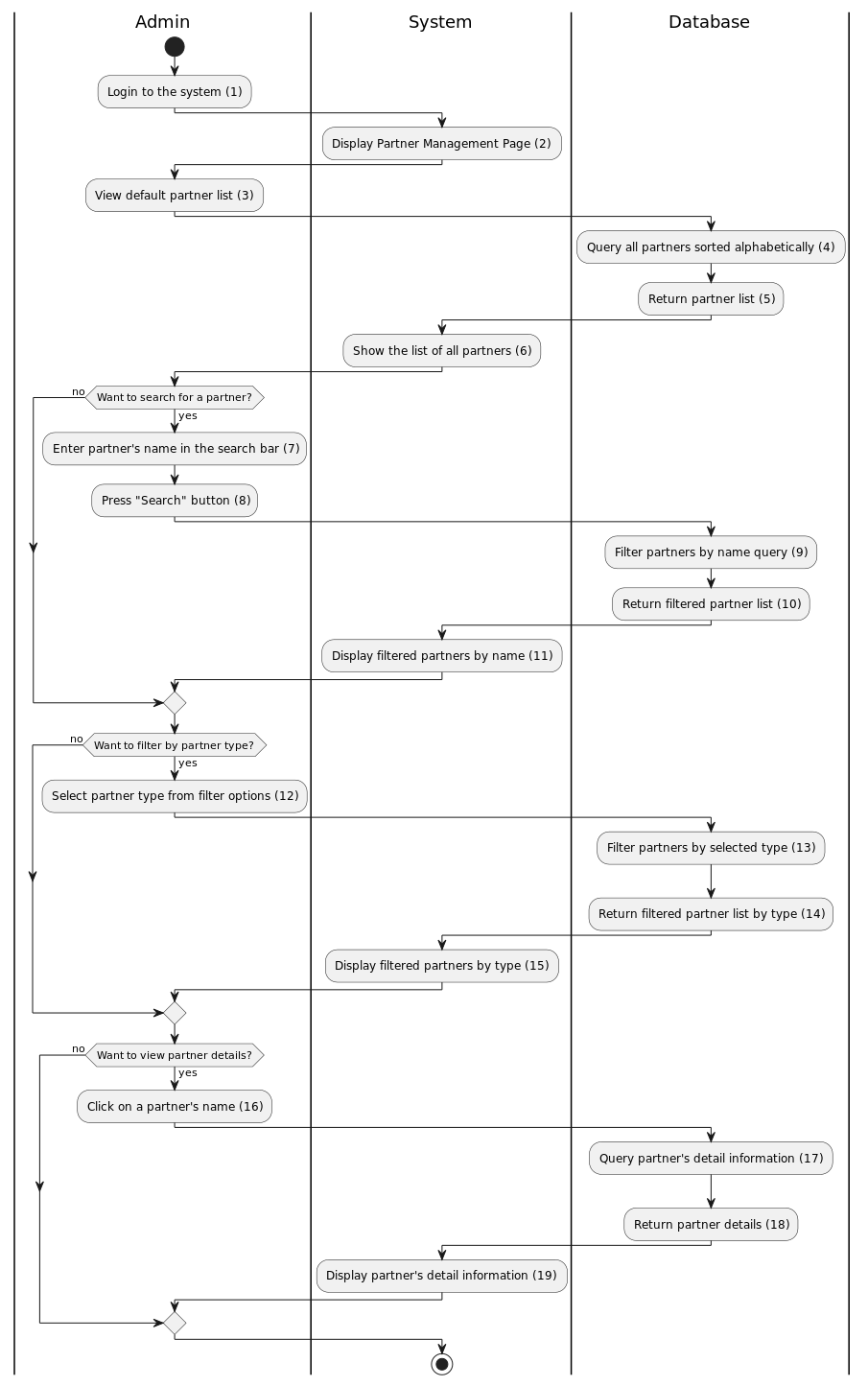
### UC4: View Partner List

### 

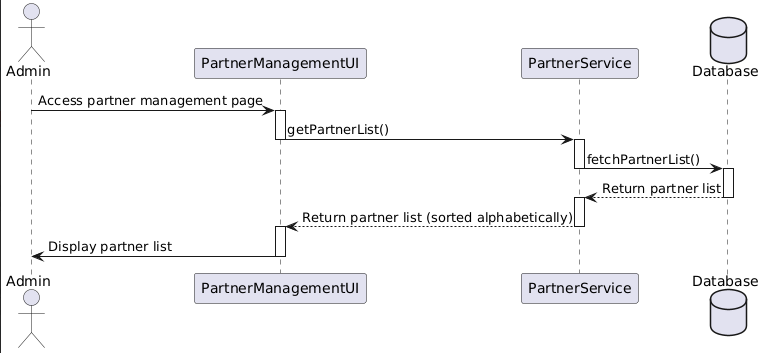
| **Name** | **View List Partner Information** |
| --- | --- |
| **Description** | This use case allows the user to view a default list of all existing partners on the partner management page, sorted by default criteria (e.g., alphabetically by name). |
| **Actor** | Admin, Partner Manager, Warehouse Manager, Logistic Manager, Sales Staff, Accountant |
| **Trigger** | * The users access the partner management page to view the partner list. |
| **Pre-condition** | * The user must be login. |
| **Post-condition** | * The system displays the complete list of partners, sorted by the default criteria. |

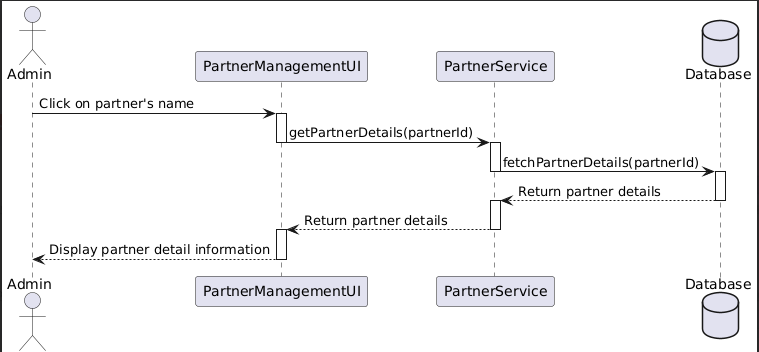
### 

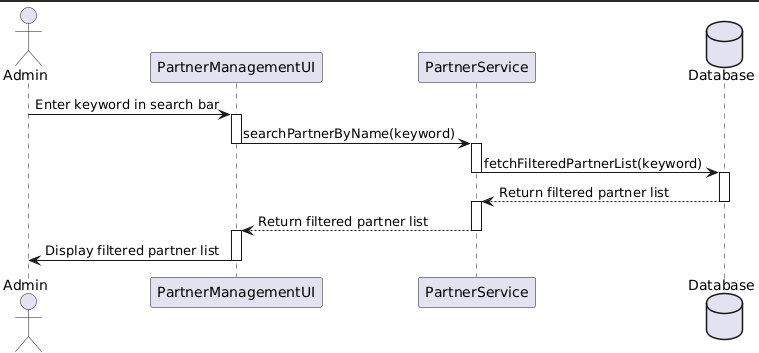
### Activities Flow

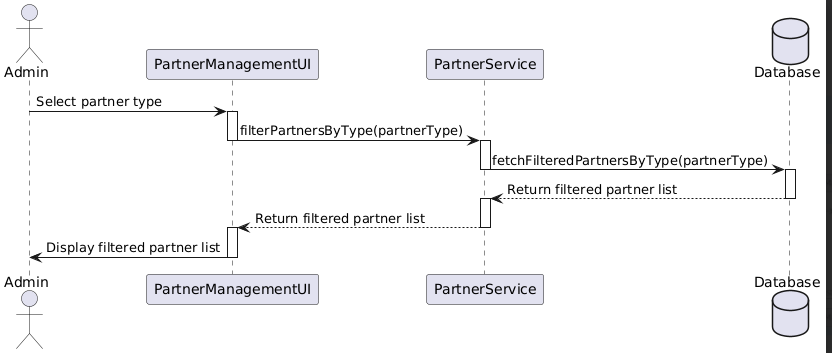


*igure 1: Activities Flow*

**

**

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Partner Management Page** when the user selects the "View Partner List" option. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the partner list from the database. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system checks for unresolved dependencies (e.g., orders, products, shipments) associated with the partner:   * Validation Step:   [checkPartnerListAvailability()]  If the partner list cannot be retrieved, the system shows the message **(MSG17)**   * If the partner list is retrieved successfully, the system displays it, sorted by the default criteria. |

### 

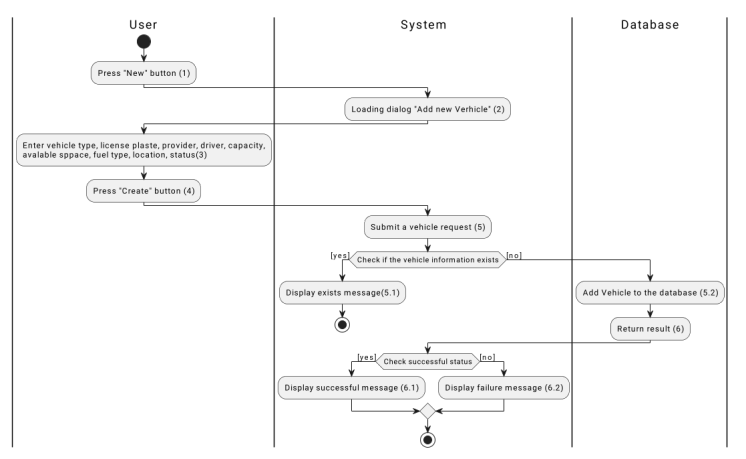
### UC5: Create Vehicle

### 

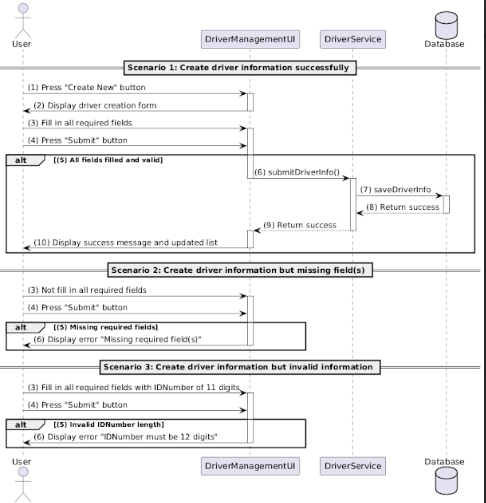
| **Name** | **Create Vehicle Information** |
| --- | --- |
| **Description** | This use case allows authorized users to add a new vehicle to the system by providing details such as LicensePlate, Driver, TransportProviderID, VehicleType, Status. |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user access into the Vehicle Management page:   * Click the “Create New” button * Fill in all required fields for the vehicle. * Select vehicle type |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role. |
| **Post-condition** | * A new vehicle, including its details (ID, LicensePlate, Driver, TransportProviderID, VehicleType, Status), is added to the vehicle list and saved in the system's database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Add New Vehicle" screen when the user clicks the "New" button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Create" button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Create" button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(VehicleType)] && [checkIsEmpty(LicensePlate)] && [checkIsEmpty(Driver)] && [checkIsEmpty(Status)]  If [VehicleType].isEmpty() || [LicensePlate].isEmpty() || [Driver].isEmpty() || [Status].isEmpty(), the system shows message **(MSG18):**  Else, proceed to the next step.   * Duplicate Validation:   [checkDuplicate(LicensePlate)]  If [LicensePlate] already exists in the database, the system shows message **(MSG19):**   * Data Submission:   If input is valid and no duplicates exist, the system saves the vehicle data.   * On success, the system shows message **(MSG20)** * On failure, the system shows message **(MSG21)** |

### 

### UC6: Edit Vehicle

### 

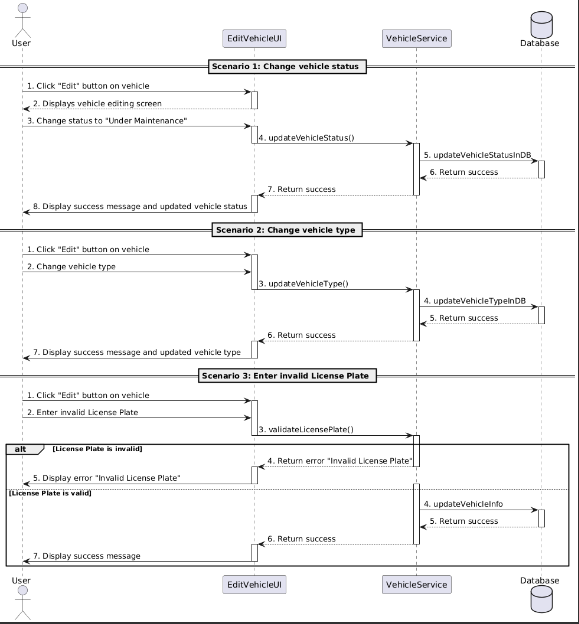
| **Name** | **Edit Vehicle Information** |
| --- | --- |
| **Description** | This use case allows authorized users to modify details of an existing vehicle, including its LicensePlate, Type and Status. |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user access into the Vehicle Management page:   * Click the “Edit” button * Change the properties of the vehicle |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role. |
| **Post-condition** | * The selected vehicle is updated with the new information and saved in the database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Vehicle Information Details" screen when the user clicks on a selected vehicle in the Vehicle Management page. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays the "Edit Vehicle" screen when the user clicks the "Edit" button on the "Vehicle Information Details" screen. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Save" button on the "Edit Vehicle" screen, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(LicensePlate)] && [checkIsEmpty(Type)] && [checkIsEmpty(Status)]  If LicensePlate, Type, or Status is empty, the system shows message **(MSG22)**  Else, do nothing.   * License Plate Validation:   The system checks if the entered License Plate format is valid using:   * [validateLicensePlate(LicensePlate)]   If LicensePlate is invalid, the system shows message **(MSG23)**  Else, do nothing.  **Duplicate Validation:**   * The system checks if a vehicle with the same LicensePlate already exists in the database using:   + [checkDuplicate(LicensePlate)] * If a duplicate LicensePlate is found, the system shows message **(MSG24)** * Else, do nothing. * Data Submission:   If input is valid and no duplicates exist, the system updates the vehicle details in the database:   * On success, the system shows message **(MSG25)** * On failure, the system shows message **(MSG26)** |

### 

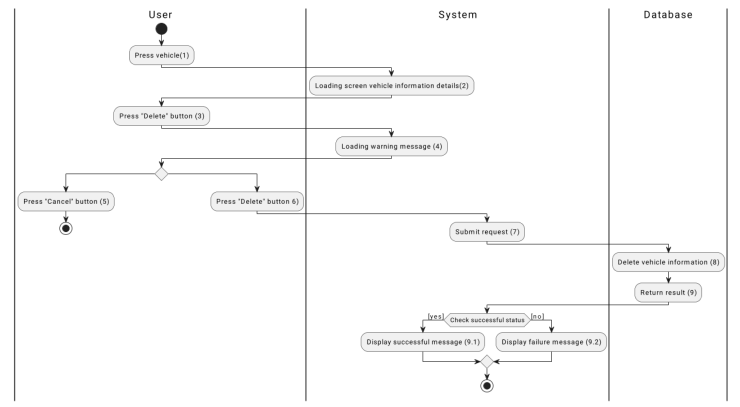
### UC7: Delete Vehicle

### 

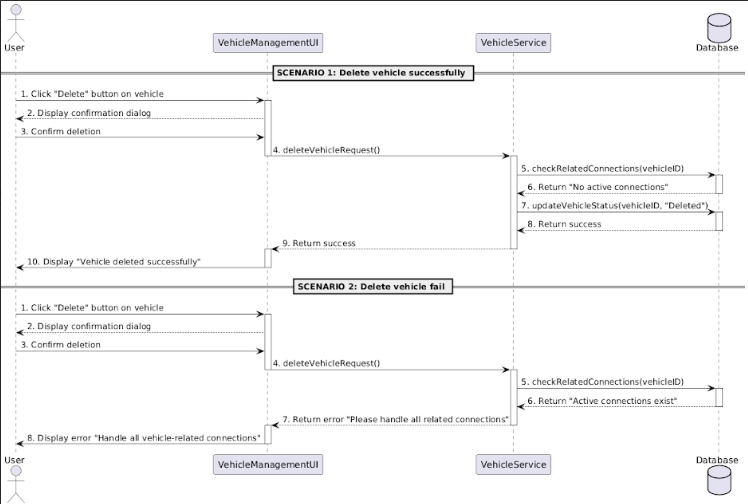
| **Name** | **Delete Vehicle Information** |
| --- | --- |
| **Description** | This use case allows authorized users to remove a vehicle from the system. |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user access into the Vehicle Management page:   * Click the “Delete” button * Confirm the deletion |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role.. |
| **Post-condition** | * The selected vehicle is removed from the vehicle list in the system's database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Delete Vehicle Information** screen when the user clicks the "Delete" button for a selected vehicle. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user confirms the deletion request, the system displays a loading indicator while processing the deletion. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system checks for unresolved dependencies (e.g., active orders, ongoing maintenance, related assignments) associated with the vehicle:   * Validation Step:   [checkDependencies(VehicleID)]   * If unresolved dependencies exist, the system shows the message: (**MSG27)**: * If all dependencies are resolved, the system updates the vehicle status to **"Deleted"**: * **Submission Step**:   + If the deletion is successful, the system shows the message: (**MSG28)**   + If the deletion fails, the system shows the message: (**MSG29)** |

### 

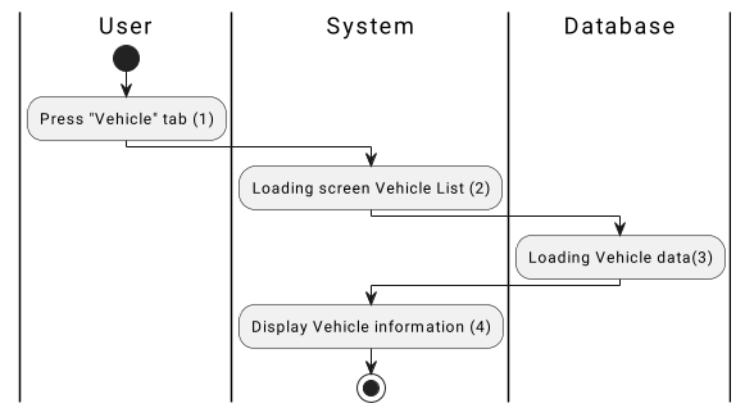
### UC8: View Vehicle List

### 

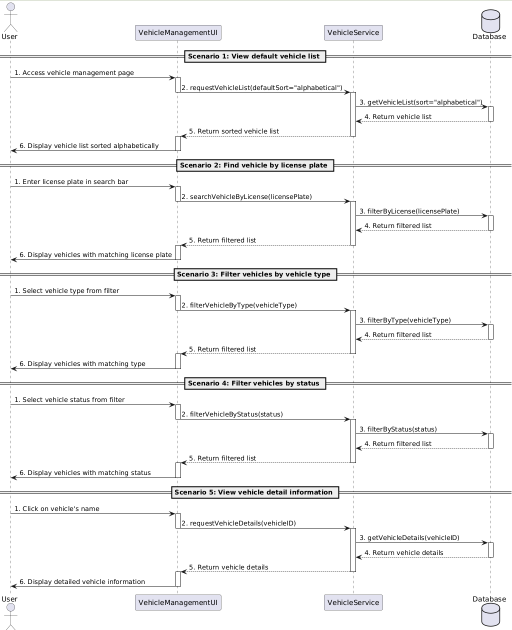
| **Name** | **View Vehicle List** |
| --- | --- |
| **Description** | This use case allows users to view the list of vehicles |
| **Actor** | Admin, Vehicle Manager, Logistics Manager, Warehouse Manager |
| **Trigger** | * When user accesses into the Vehicle Management page |
| **Pre-condition** | * The user must be logged in |
| **Post-condition** | * The system displays a list of all existing vehicles |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Vehicle Management Page** when the user selects the "Vehicle" tab. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the vehicle list data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules:  * The system retrieves the vehicle list data by invoking requestVehicleList() API, applying default sorting criteria (alphabetically) if no filter is selected. * If the data retrieval process fails, the system displays an error message:   + **Error Message:** **(MSG30)**  Filtering and Searching Rules:  * **License Plate Search:**   + The system filters and displays vehicles matching the license plate entered by the user in the search bar using searchVehicleByLicense() API. * **Filter by Vehicle Type:**   + The system filters and displays vehicles matching the selected type using filterVehicleByType() API. * **Filter by Status:**   + The system filters and displays vehicles matching the selected status using filterVehicleByStatus() API.  Vehicle Details Rules:  * When the user selects a vehicle, the system fetches its detailed information by invoking requestVehicleDetails() API. * The system displays a **loading indicator** while retrieving vehicle details. * If the retrieval process fails, the system displays an error message:   + **Error Message: (MSG31)**  Displaying Rules:  * If the data is successfully retrieved:   + The vehicle list is displayed alphabetically by default.   + Filters (type, status) or search criteria (license plate) are applied to display the filtered list.   + Detailed information for a selected vehicle is displayed in a new section or dialog. |

### 

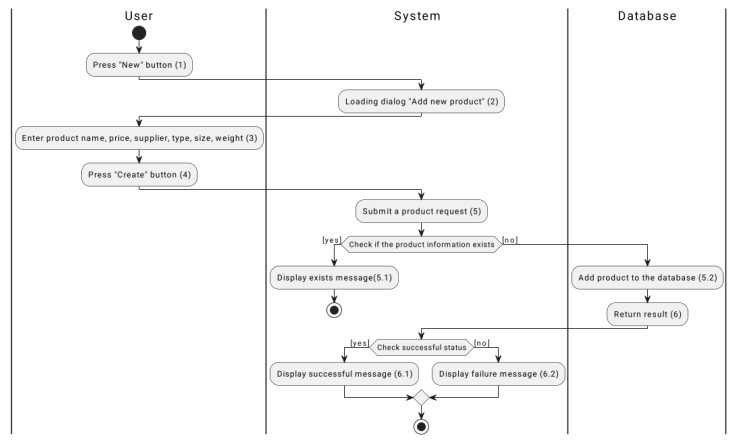
### UC9: Create Product

### 

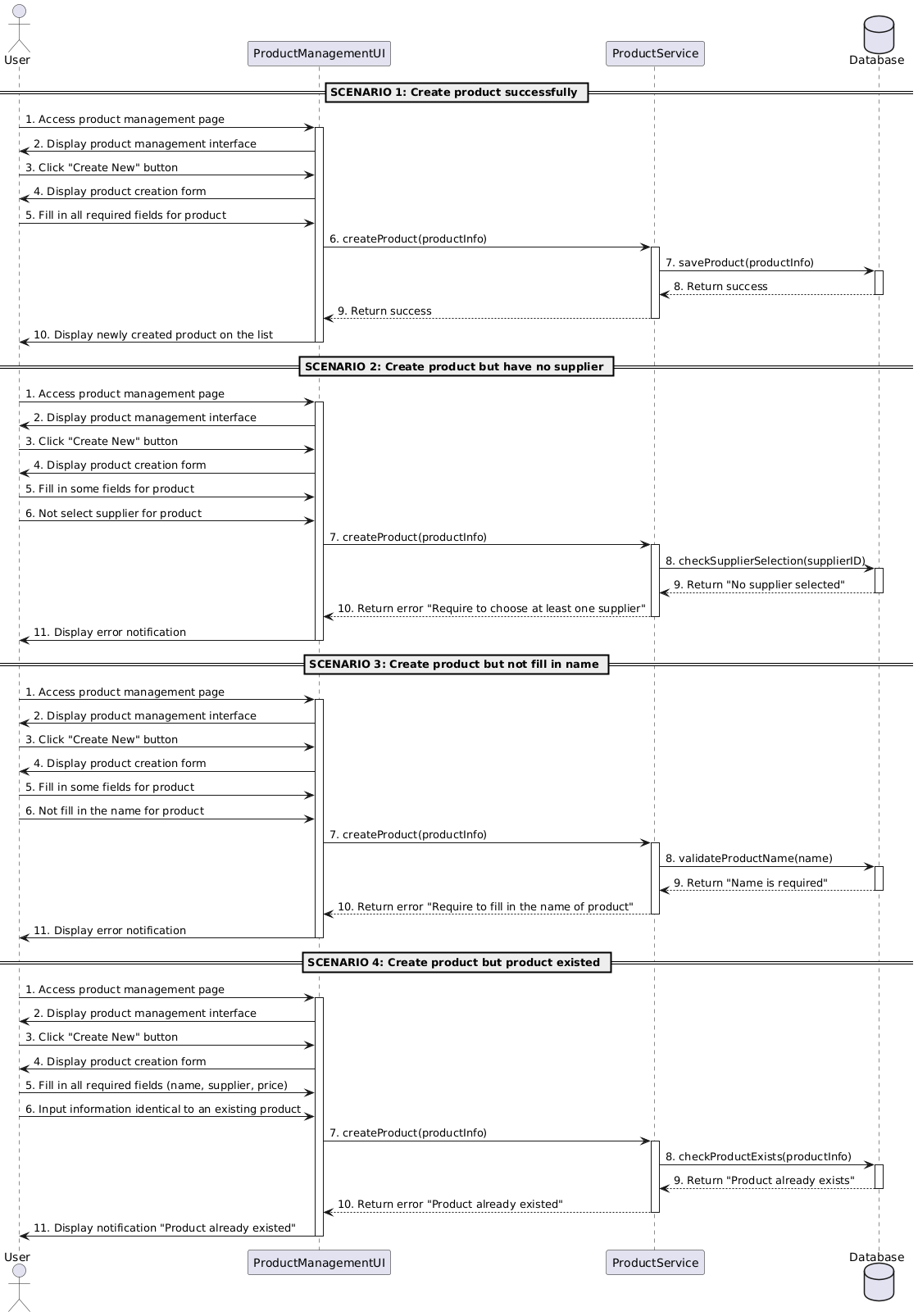
| **Name** | **Create Product Form** |
| --- | --- |
| **Description** | This use case allows authorized users to add a new product to the system by providing details such as product name, quantity, unit, status, supplier ID, type, size, and weight. |
| **Actor** | Admin, Sales staff |
| **Trigger** | When user accesses into the Product Management page:   * Click the “Create New” button * Fill in all required fields for the product |
| **Pre-condition** | * The user must be logged in with an Admin or Sales Staff role. |
| **Post-condition** | * A new product, including its details (ID, name, quantity, unit, status, supplier ID, type, size, and weight), is added to the product list and saved in the system's database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Add New Product" screen when the user clicks the "Create New" button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Create" button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Create" button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(Name)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(Unit)] && [checkIsEmpty(SupplierID)] && [checkIsEmpty(Type)] && [checkIsEmpty(Size)] && [checkIsEmpty(Weight)]   * If any of the required fields (Name, Quantity, Unit, SupplierID, Type, Size, Weight) are empty, the system shows the message **(MSG32):** * Else, proceed to the next step. * Duplicate Validation:   [checkDuplicate(Name, SupplierID)]   * If [Name] and [SupplierID] already exist in the database, the system shows the message **(MSG33):** * Data Submission:   If input is valid and no duplicates exist, the system saves the product data.  On success, the system shows the message **(MSG34)**  On failure, the system shows the message **(MSG35)**   * Error Message:   Scenario 1: Required fields not filled.   * The system returns the error message **(MSG36)**   Scenario 2: Supplier not selected.   * The system returns the error message **(MSG37)**   Scenario 3: Product name not provided.   * The system returns the error message **(MSG38)**   Scenario 4: Duplicate product.   * The system returns the error message **(MSG39)** |

### 

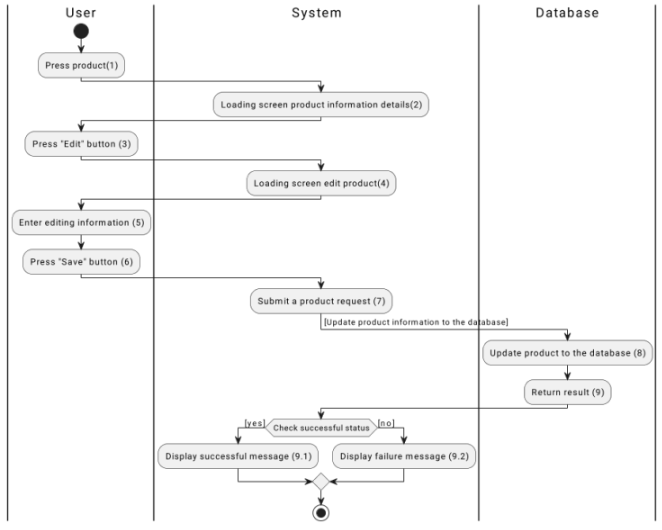
### UC10: Edit Product

### 

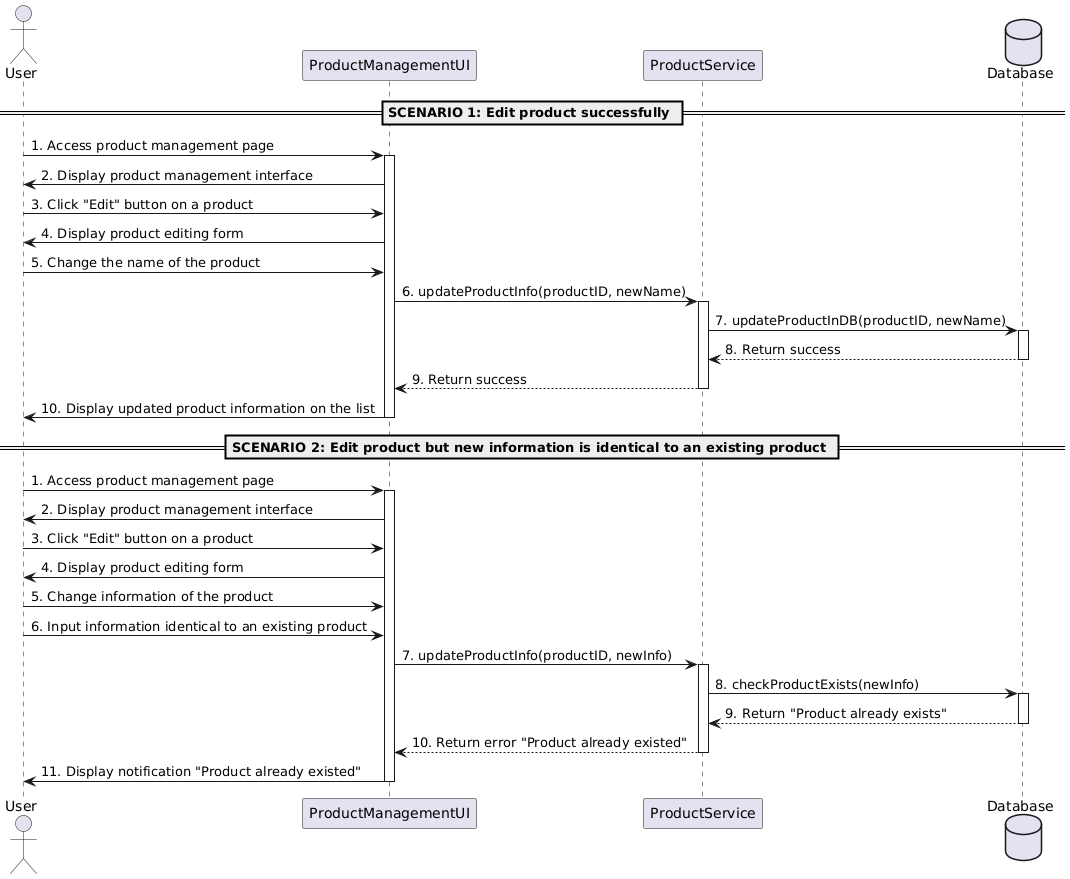
| **Name** | **Edit Product Form** |
| --- | --- |
| **Description** | This use case allows authorized users to modify details of an existing product, including its quantity, unit, status, supplier ID, type, size, and weight. |
| **Actor** | Admin, Sales staff |
| **Trigger** | When user accesses into the Product Management page:   * Click the “Edit” button * Change the properties of the product. |
| **Pre-condition** | * The user must be logged in with an Admin or Sales Staff role. |
| **Post-condition** | * The selected product is updated with the new information (name, quantity, unit, status, supplier ID, type, size, and weight) and saved in the database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Product Information Details" screen when the user clicks on a selected product in the Product Management page. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays the "Edit Product" screen when the user clicks the "Edit" button on the "Product Information Details" screen. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Save" button on the "Edit Product" screen, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(Name)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(Unit)] && [checkIsEmpty(Status)] && [checkIsEmpty(SupplierID)] && [checkIsEmpty(Type)] && [checkIsEmpty(Size)] && [checkIsEmpty(Weight)]  If any required field (Name, Quantity, Unit, Status, SupplierID, Type, Size, Weight) is empty, the system shows message **(MSG39)**  Else, do nothing.  **Duplicate Validation:**  The system checks if a product with the same Name and SupplierID already exists in the database using:  [checkDuplicate(Name, SupplierID)]  If a duplicate product is found, the system shows message **(MSG40)** Else, do nothing.   * Data Submission:   If input is valid and no duplicates exist, the system updates the product details in the database:   * On success, the system shows message **(MSG41)**   On failure, the system shows message **(MSG42)** |

### 

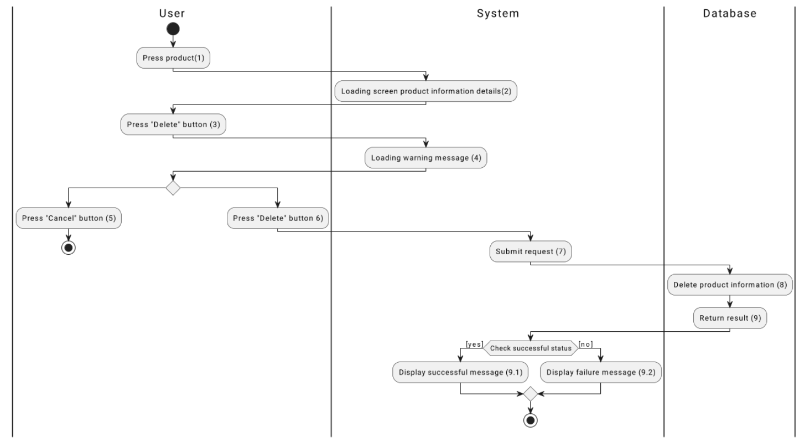
### UC11: Delete Product

### 

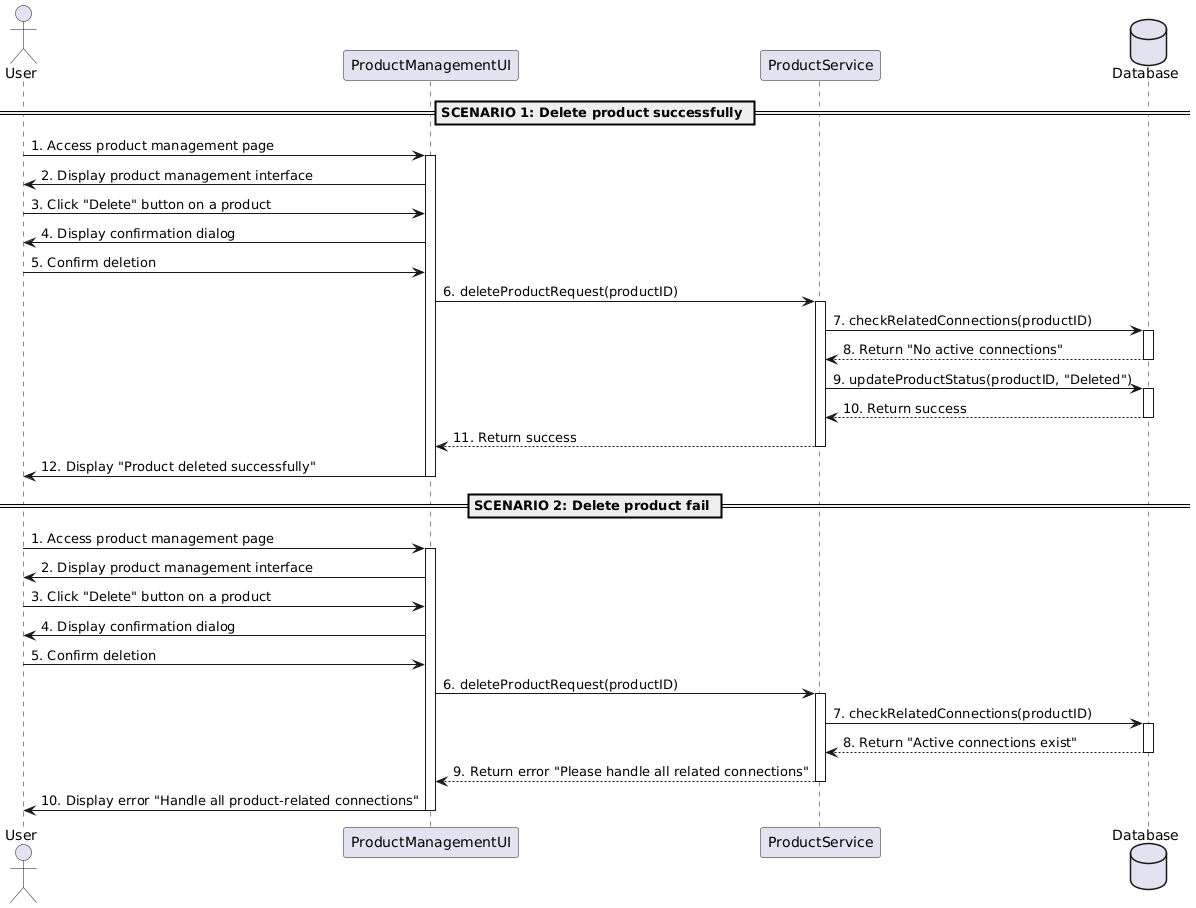
| **Name** | **Delete Product’s Form** |
| --- | --- |
| **Description** | This use case allows authorized users to remove a product from the system. |
| **Actor** | Admin, Sales staff |
| **Trigger** | When user accesses into the Product Management page:   * Click the “Delete” button * Confirm the deletion |
| **Pre-condition** | * The user must be logged in with an Admin or Sales Staff role. |
| **Post-condition** | * The selected product is removed from the product list in the system's database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Delete Product Information** screen when the user clicks the "Delete" button for a selected product. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user confirms the deletion request, the system displays a loading indicator while processing the deletion. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system checks for unresolved dependencies (e.g., orders, inventories, related connections) associated with the product:   * Validation Step:   [checkDependencies(ProductID)]   * If unresolved dependencies exist, the system shows the message: **(MSG43)** * If all dependencies are resolved, the system updates the product status to **"Deleted"**: **Submission Step**:   + If the deletion is successful, the system shows the message: **(MSG44)**   + If the deletion fails, the system shows the message: **(MSG45)** |

### 

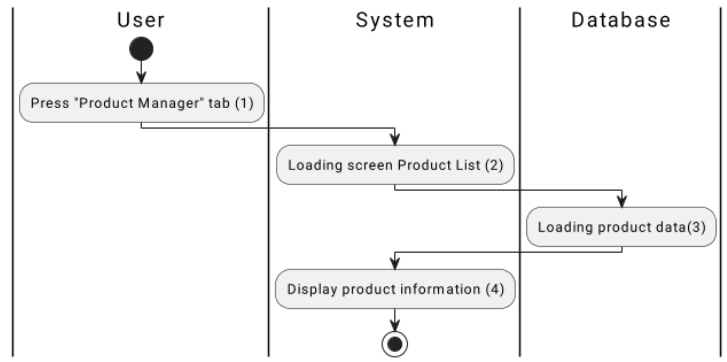
### UC12: View Product List

### 

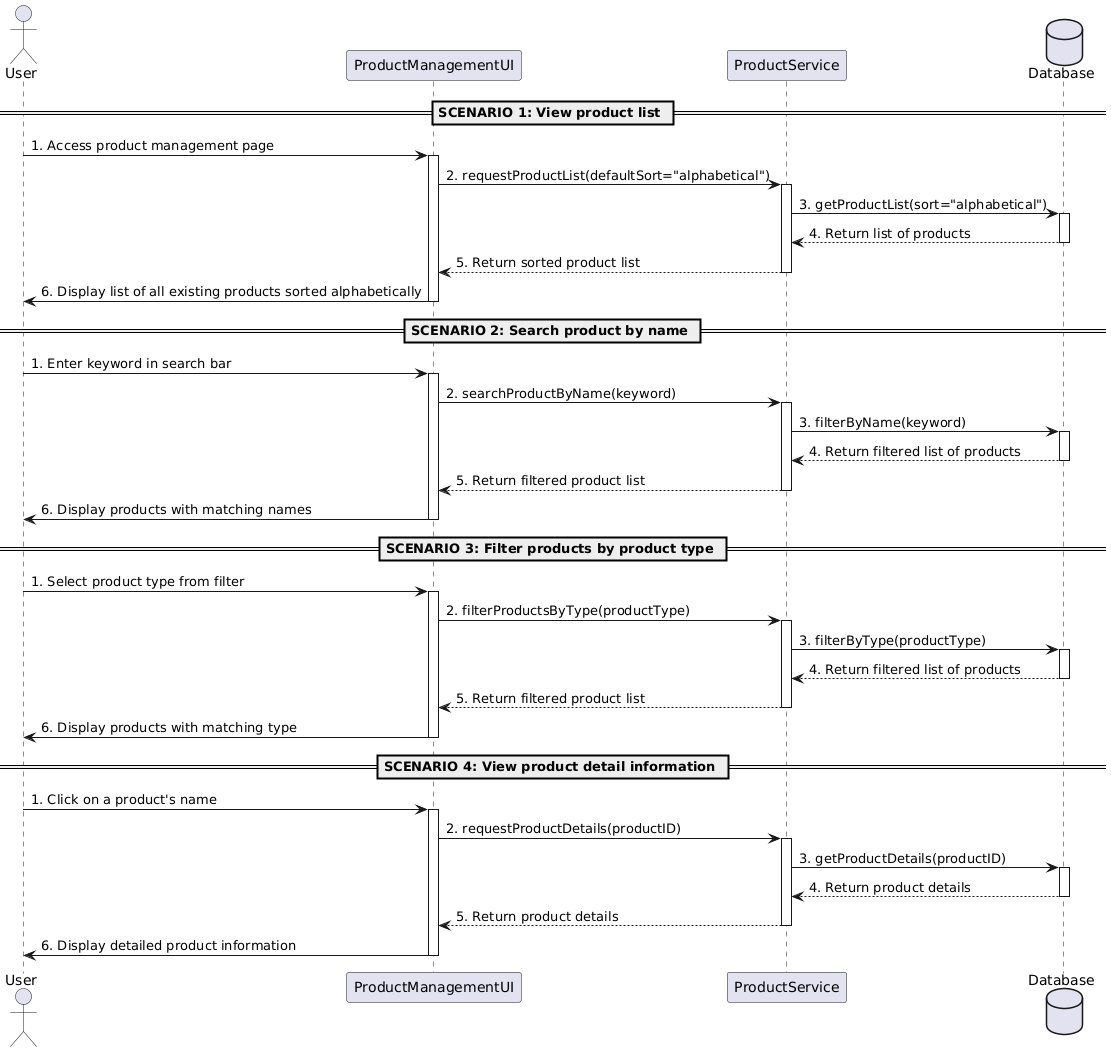
| **Name** | **View Product List** |
| --- | --- |
| **Description** | This use case allows users to view the list of products and their details, including ID, name, quantity, unit, status, supplier ID, type, size, and weight. |
| **Actor** | Admin, Sales Staff, Warehouse Manager, Accountant, Logistic Manager |
| **Trigger** | * When user accesses into the Product Management page |
| **Pre-condition** | * The user must be logged in. |
| **Post-condition** | * The system displays a list of all existing products. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Product Management Page** when the user selects the "Product Manager" tab. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the product list data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules: The system retrieves the product list data by invoking requestProductList() API, applying default sorting criteria (alphabetically) if no filter is selected.  If the data retrieval process fails, the system displays an error message:   * Error Message: ***(MSG46)***  Filtering and Searching Rules:  * Search by Product Name:   + The system filters and displays products matching the entered keyword in the search bar using searchProductByName() API. * Filter by Product Type:   + The system filters and displays products matching the selected type using filterProductsByType() API.  Product Details Rules:  * When the user selects a product, the system fetches its detailed information by invoking requestProductDetails() API. * The system displays a loading indicator while retrieving product details. * If the retrieval process fails, the system displays an error message:   + Error Message: **(MSG47)**  Displaying Rules:  * If the data is successfully retrieved:   + The product list is displayed alphabetically by default.   + Filters (type) or search criteria (product name) are applied to display the filtered list.   + Detailed information for a selected product is displayed in a new section or dialog. |

### 

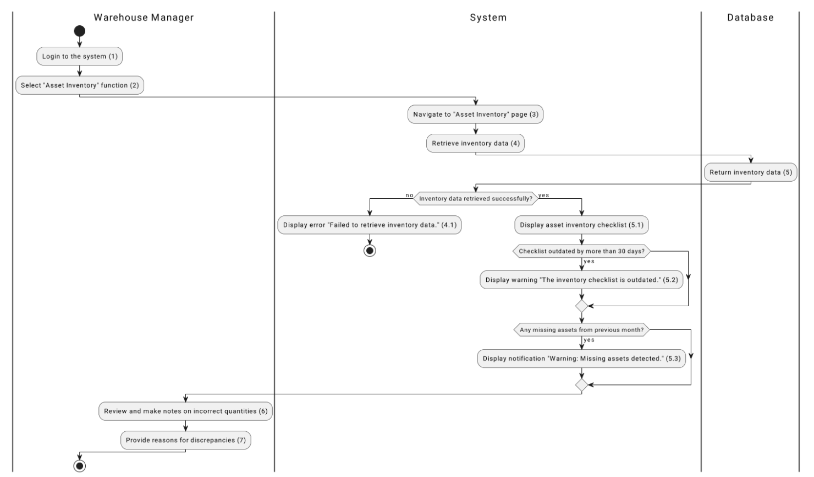
### UC13: Asset Inventory

### 

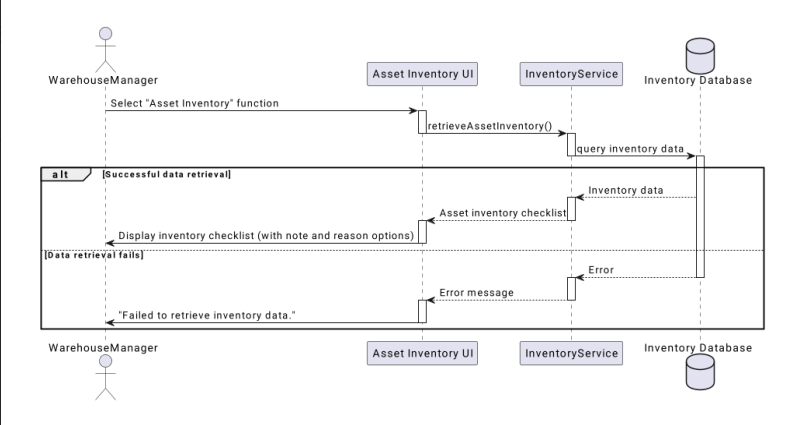
| **Name** | **Asset Inventory** |
| --- | --- |
| **Description** | This use case allows users to manage and conduct inventory checks for assets in the warehouse, ensuring that records are up to date and accurate. |
| **Actor** | Admin, Warehouse Manager, Accountant |
| **Trigger** | When a user accesses the inventory management section:   * Click the “Asset Inventory” button. * Check and update asset information. |
| **Pre-condition** | * The user must be logged in with a role that has permission to manage assets. |
| **Post-condition** | * The updated asset information is stored in the system’s database, with discrepancies flagged for review if necessary. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system loads the **Asset Inventory** screen when the user selects the "Asset Inventory" button in the Inventory Management section. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * While retrieving inventory data from the database, the system displays a **loading indicator** to notify the user of the ongoing process. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * Users are required to review and update the asset inventory checklist:   Make notes on any incorrect quantities.  Provide reasons for any discrepancies during the review process   * **Inventory Data Retrieval Rules**:   The system retrieves inventory data by invoking the retrieveAssetInventory() API.  If the data retrieval fails, the system displays (**MSG48)**  If the data retrieval is successful, the system proceeds to display the **Asset Inventory Checklist**.   * Checklist Rules:   If the retrieved inventory checklist is outdated (more than 30 days old), the system displays a warning (**MSG49)**  If missing assets from the previous month are detected, the system displays a notification (**MSG50)**   * Validation Steps   **Inventory Data Validation:**  The system validates the retrieved inventory data for completeness.  If discrepancies are detected, they are flagged for review.  **User Input Validation:**  The system verifies that user inputs (notes and reasons) are correctly formatted before submission.   * Error Handling Rules:   If an error occurs during the saving process, the system displays (**MSG51**) |

### 

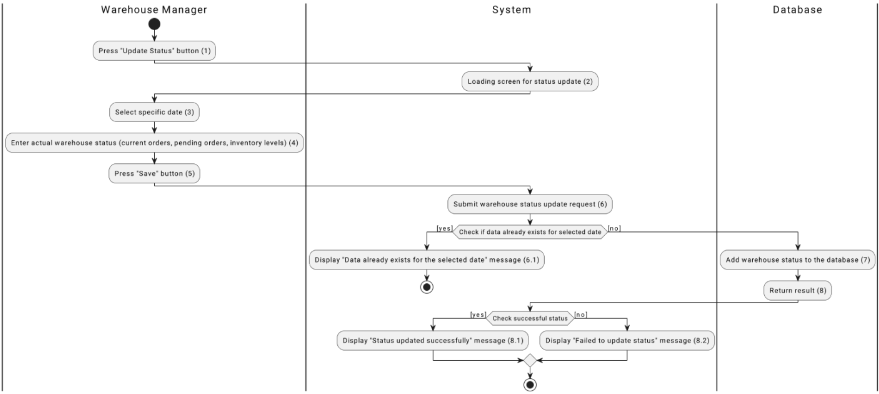
### UC14: Report status daily

### 

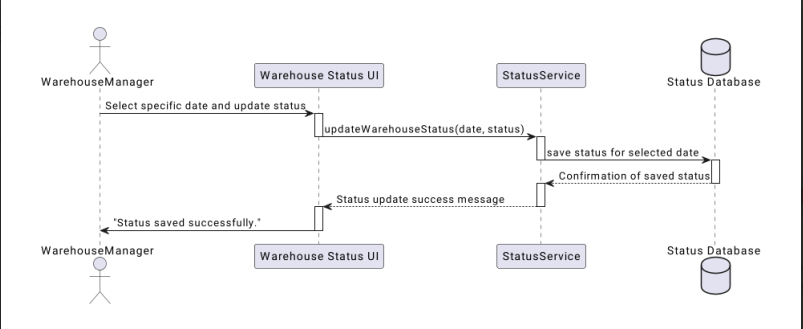
| **Name** | **Report Status Daily** |
| --- | --- |
| **Description** | This use case allows the user to create and submit a daily status report of the warehouse operations, including inventory changes and maintenance updates. |
| **Actor** | Admin, Warehouse Manager |
| **Trigger** | When a user finishes warehouse operations for the day:   * Click the “Report Status Daily” button. * Fill in required details and submit the report. |
| **Pre-condition** | * The user must have completed the day’s operations and be logged in as Warehouse Manager or Admin. |
| **Post-condition** | * The status report is saved in the system, and relevant parties are notified if necessary. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Report Status Daily** screen when the user selects the "Report Status Daily" button in the Warehouse Management section. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * While loading the status reporting screen, the system displays a **loading indicator** to inform the user of the process. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  The user is required to:   * Select a specific date for the report. * Enter the details of the daily warehouse status, including:   + Current orders.   + Pending orders.   + Inventory levels. * **Validation Steps:**   **Date Validation:**   * If no date is selected, the system prompts the user with (**MSG52)** * If a report for the selected date already exists, the system displays (**MSG53)**   **Data Validation:**   * If the required details are incomplete, the system displays (**MSG54)**   **Saving Rules:**  Once the user submits the report:   * The system saves the report to the database using the updateWarehouseStatus() API. * If the report is successfully saved, the system displays (**MSG55)** * If the save process fails, the system displays (**MSG56)** |

### 

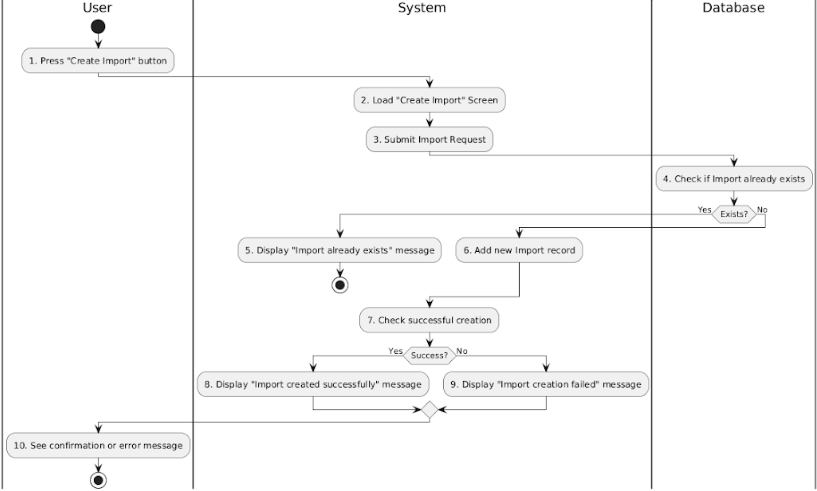
### UC15: Create Import Order

### 

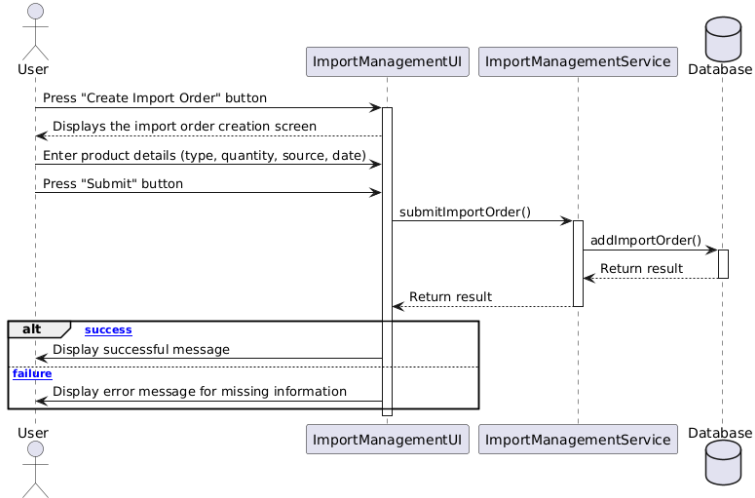
| **Name** | **Create Import Form** |
| --- | --- |
| **Description** | This use case allows authorized users to create a new import order, including details like product name, quantity, unit, supplier, date, and total cost. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | When the user accesses the Import Management page:   * Click the “Create Import” button * Fill in the necessary details (product, quantity, supplier, etc.). |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role. |
| **Post-condition** | * A new import order is created and saved in the system with the provided details (product name, quantity, unit, supplier, date, and total cost). |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Create Import Order" screen when the user clicks the "Create Import" button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Submit" button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Submit" button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(ProductName)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(Unit)] && [checkIsEmpty(SupplierID)] && [checkIsEmpty(Date)] && [checkIsEmpty(TotalCost)]   * If any required fields (ProductName, Quantity, Unit, SupplierID, Date, TotalCost) are empty, the system shows the message **(MSG57)** * Else, proceed to the next step. * Duplicate Validation:   If input is valid and no duplicates exist, the system saves the import order data.   * On success, the system shows the message **(MSG58)** * On failure, the system shows the message **(MSG59)** * Data Submission:   If input is valid and no duplicates exist, the system saves the product data.  On success, the system shows the message **(MSG60)**  On failure, the system shows the message **(MSG61)**   * Error Message:   Scenario 1: Missing required fields.   * The system returns the error message **(MSG62)**   Scenario 2: Duplicate import order.   * The system returns the error message **(MSG63)**   Scenario 3: Submission failure due to a system or database error.   * The system returns the error message **(MSG64)** |

### 

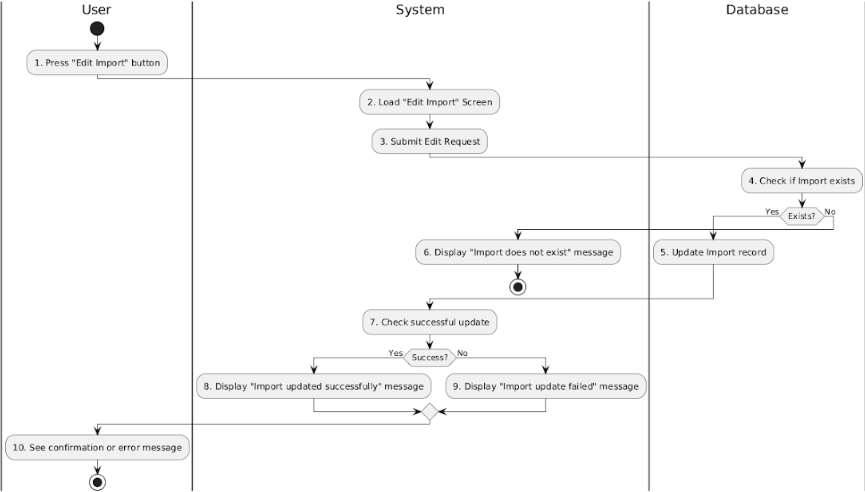
### UC16: Edit Import Order

### 

| **Name** | **Edit Import Form** |
| --- | --- |
| **Description** | This use case allows authorized users to modify details of an existing import order, including product name, quantity, unit, supplier, date, and total cost. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | * When the user accesses the Import Management page |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role. |
| **Post-condition** | * The selected import order is updated with the new information (product name, quantity, unit, supplier, date, and total cost) and saved in the system. |

### 

### Activities Flow



*Figure 1: Activities Flow*

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Import Management" page with a list of import orders when the user navigates to the Import Management module. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays the "Edit Import" screen when the user clicks the "Edit Import" button on a selected import order. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user submits the "Edit Import" request, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(ProductName)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(Unit)] && [checkIsEmpty(Supplier)] && [checkIsEmpty(Date)] && [checkIsEmpty(TotalCost)]  If any required field (ProductName, Quantity, Unit, Supplier, Date, or TotalCost) is empty, the system shows message **(MSG65)**  Else, do nothing.   * Existence Check   The system checks if the selected import order exists in the database using:  [checkExist(ImportOrderID)]  If the import order does not exist, the system shows message **(MSG66)**  Else, proceed to the next step.   * Data Submission:   If all validations pass and the import order exists, the system updates the import order details in the database:   * On success, the system shows message **(MSG67)** * On failure, the system shows message **(MSG68)** |

### 

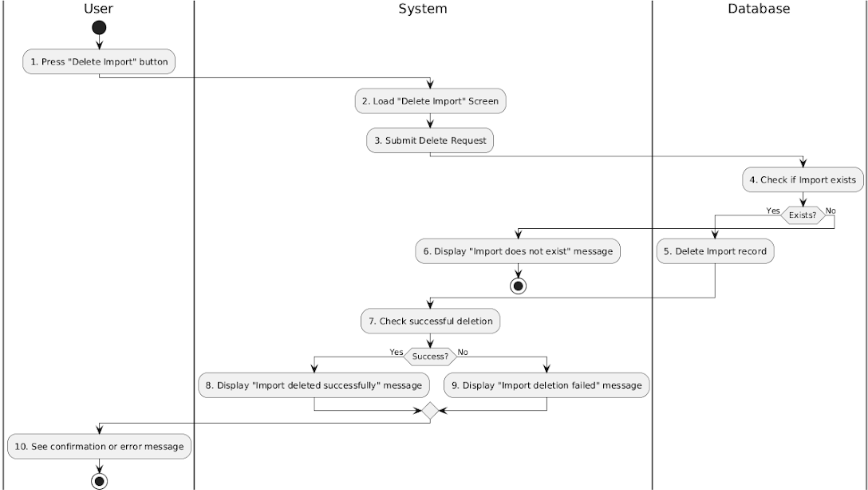
### UC17: Delete Import Order

### 

| **Name** | **Delete Import Order** |
| --- | --- |
| **Description** | This use case allows authorized users to cancel an existing import order, ensuring that any associated product quantities, suppliers, and total costs are updated accordingly in the inventory system. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | * When the user accesses the Import Management page and clicks the "Cancel" button. |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role. |
| **Post-condition** | * The import order is canceled, and its status is updated to "Canceled." Relevant product quantities are returned to inventory, and the cancellation details are recorded in the system. |

### 

### Activities Flow



*Figure 1: Activities Flow*

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Delete Import Order Information** screen when the user clicks the "Cancel" button for a selected import order. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays the **Delete Import Order Information** screen when the user clicks the "Cancel" button for a selected import order. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system verifies if the import order exists and checks for associated dependencies (e.g., inventory adjustments, supplier relations):   * Validation Step:   [checkDependencies(ImportOrderID)]   * If the import order does not exist, the system shows the message: **(MSG69)** * If the import order exists and is valid for deletion, the system cancels the order and updates the relevant product quantities, supplier data, and inventory system: **Submission Step**:   + If the deletion is successful, the system shows the message: **(MSG70)**   + If the deletion fails, the system shows the message: **(MSG71)** |

### 

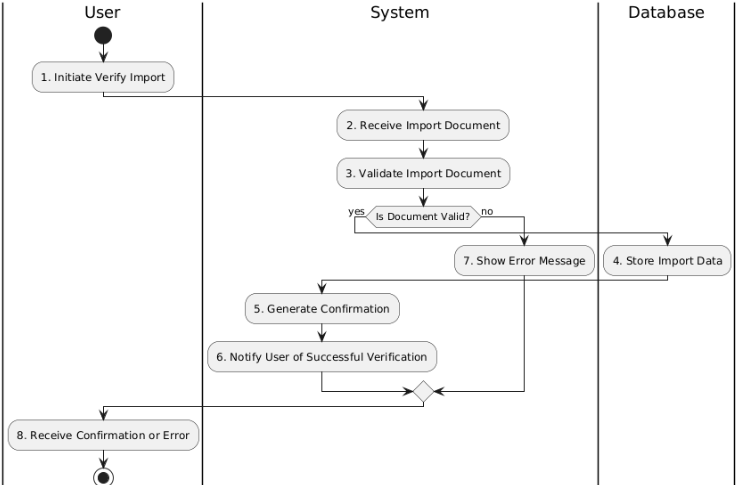
### UC18: Confirm Import Order

### 

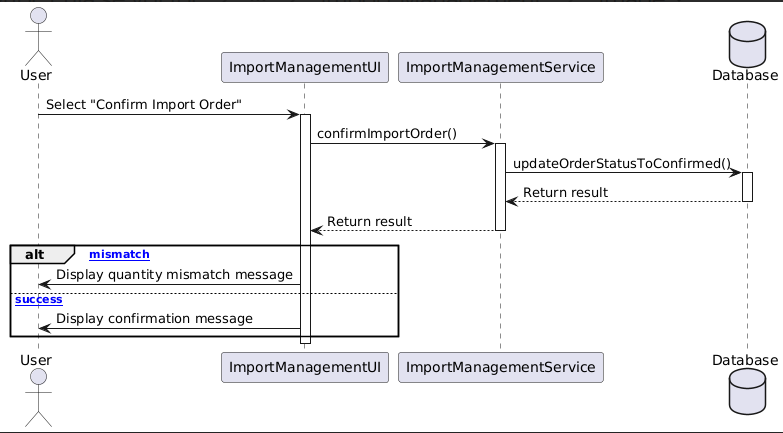
| **Name** | **Confirm Import Form** |
| --- | --- |
| **Description** | This use case allows authorized users to confirm the details of an import order, verifying product quantities, unit, supplier, and total cost before finalizing the import. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | * When the user accesses the Import Management page |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role.. |
| **Post-condition** | * The import order is confirmed, and its status is updated to "Confirmed." The confirmed details (product name, quantity, unit, supplier, date, and total cost) are saved in the system. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Confirm Import Form** screen when the user selects the "Confirm Import Order" button in the Import Management section. |
| *(4)* | *BR2* | **Loading Screen Rules:**  The system retrieves the details of the import order, including:   * Product name. * Quantity. * Unit. * Supplier. * Total cost. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * Validation Rules:   **Document Validation:**   * The system validates the import document to ensure all fields are complete and in the correct format. * If the document is invalid, the system displays (**MSG72)** * If the document is valid, the system proceeds to verify the order details.   **Quantity Verification:**   * The system verifies that the product quantities match the expected values. * If there is a mismatch, the system displays (**MSG73)**   If all validations pass:   * The system updates the order status to "Confirmed" using the updateOrderStatusToConfirmed() API. * The system stores the confirmed details (product name, quantity, unit, supplier, date, and total cost) in the database.   Once the import order is successfully confirmed, the system notifies the user with (**MSG74)** |

### 

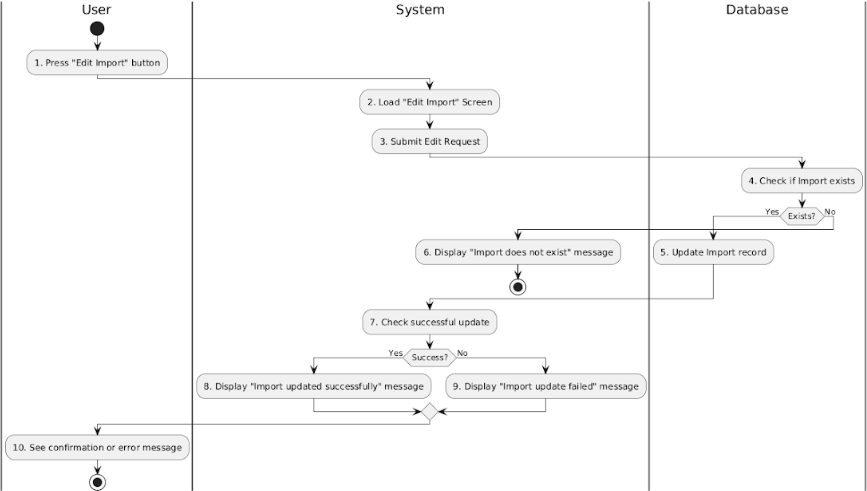
### UC19: View Import List

### 

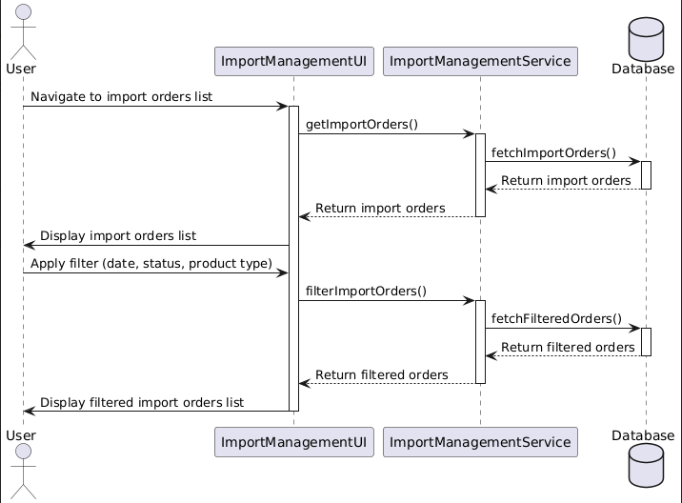
| **Name** | **View Import List** |
| --- | --- |
| **Description** | This use case allows authorized users to view a list of all import orders, including details such as product quantities, supplier information, and status of each order for effective inventory management. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | * When the user accesses the Import Management page and clicks on the "View Import List" option. |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role. |
| **Post-condition** | * The system displays a list of all import orders with relevant details, allowing the user to select and view specific import orders for more information. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Import Management Page** when the user selects the "View Import List" option. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the import orders data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules: The system retrieves the import orders list data by invoking getImportOrders() API.  If the data retrieval process fails, the system displays an error message:   * **Error Message: (MSG75)**  Filtering Rules: **Filter by Date:**   * The system filters and displays import orders based on the selected date range using the filterImportOrders(date) API.   **Filter by Status:**   * The system filters and displays import orders based on the selected status using the filterImportOrders(status) API.   **Filter by Product Type:**   * The system filters and displays import orders based on the selected product type using the filterImportOrders(productType) API.  Updating Import Records Rules: The system allows users to edit import records by navigating to the "Edit Import" screen.  Upon submitting the edit request:   * The system checks if the selected import order exists in the database by invoking checkImportExists(). * If the record does not exist, the system displays an error message:   + Error Message: ***(MSG76)*** * If the record exists and is successfully updated, the system displays a success message:   + Success Message***: (MSG77)*** * If the update fails, the system displays an error message:   + Error Message: ***(MSG78)***  Displaying Rules: If the data is successfully retrieved:   * The import orders list is displayed in its entirety, sorted alphabetically by default. * Filters (date, status, product type) are applied to display the filtered list. * For successfully edited import orders, updated details are reflected in the displayed list. |

### 

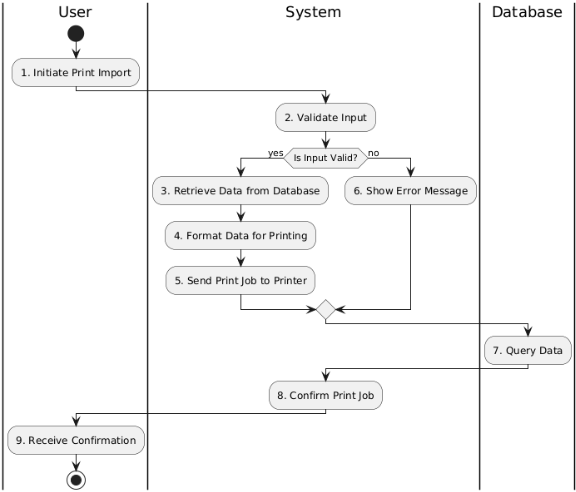
### UC20: Print Import Order

### 

| **Name** | **Print Import Order** |
| --- | --- |
| **Description** | This use case allows authorized users to print the details of an import order, including product quantities, unit, supplier, and total cost for record-keeping and verification purposes. |
| **Actor** | Admin, Inventory Manager |
| **Trigger** | * When the user accesses the Import Management page and clicks the "Print" button. |
| **Pre-condition** | * The user must be logged in with an Admin or Inventory Manager role, and there must be an existing import order to print.. |
| **Post-condition** | * The import order details are sent to the printer, and a physical copy of the import order is generated for the user. |

### 

### Activities Flow



*Figure 1: Activities Flow*

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Product Management Page** when the user selects the "Product Manager" tab. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the product list data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules: The system retrieves the product list data by invoking requestProductList() API, applying default sorting criteria (alphabetically) if no filter is selected.  If the data retrieval process fails, the system displays an error message:   * Error Message: ***(MSG79)***  Filtering and Searching Rules:  * Search by Product Name:   + The system filters and displays products matching the entered keyword in the search bar using searchProductByName() API. * Filter by Product Type:   + The system filters and displays products matching the selected type using filterProductsByType() API.  Product Details Rules:  * When the user selects a product, the system fetches its detailed information by invoking requestProductDetails() API. * The system displays a loading indicator while retrieving product details. * If the retrieval process fails, the system displays an error message:   + Error Message: **(MSG80)**  Displaying Rules:  * If the data is successfully retrieved:   + The product list is displayed alphabetically by default.   + Filters (type) or search criteria (product name) are applied to display the filtered list.   + Detailed information for a selected product is displayed in a new section or dialog. |

### UC21: Create Driver

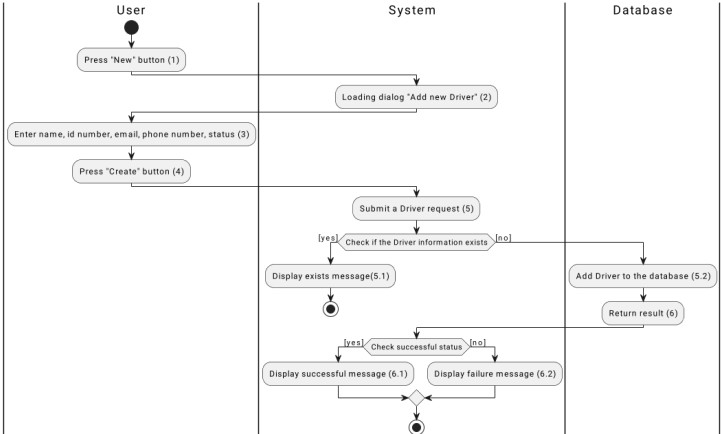
### 

| **Name** | **Create Driver’s Form** |
| --- | --- |
| **Description** | This use case allows authorized users to add a new driver to the system by providing details such as Name, IDNumber, Email, PhoneNumber |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user accesses into the Driver Management page:   * Click the “Create New” button * Fill in all required fields for the driver. |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role. |
| **Post-condition** | * A new driver, including its details (ID, Name, IDNumber, Email, PhoneNumber), is added to the driver list and saved in the system's database. |

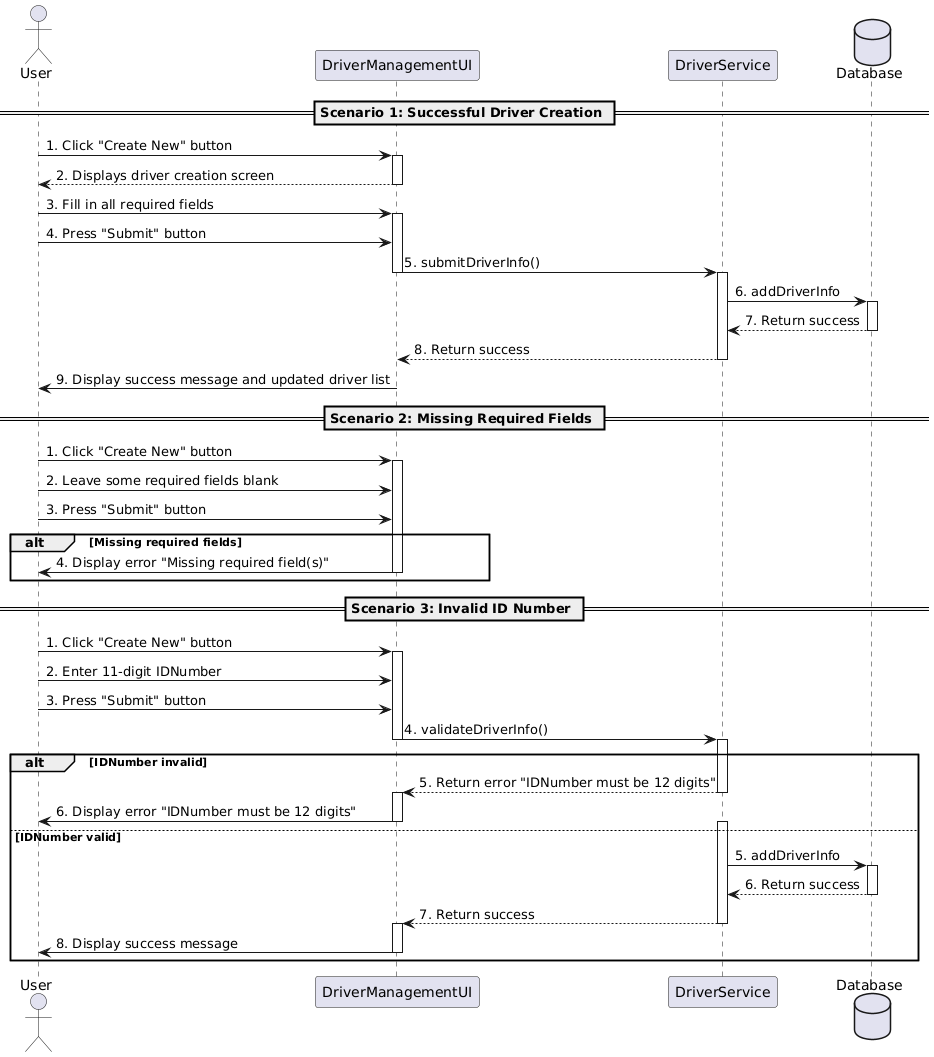
### 

3

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Add New Driver" screen when the user clicks the "New" button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Create" button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Submit" button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(Name)] && [checkIsEmpty(IDNumber)] && [checkIsEmpty(Email)] && [checkIsEmpty(PhoneNumber)]   * If any required fields (Name, IDNumber, Email, PhoneNumber) are empty, the system shows the message **(MSG81):** * Else, proceed to the next step.. * Duplicate Validation:   [validateIDNumberLength(IDNumber)]   * If IDNumber is not exactly 12 digits, the system shows the message **(MSG82)**   [checkDuplicate(IDNumber, Email)]   * If the IDNumber or Email already exists in the database, the system shows the message **(MSG83):** * Data Submission:   If input is valid and no duplicates exist, the system saves the driver data.   * On success, the system shows the message **(MSG84)** * On failure, the system shows the message **(MSG85)** * Error Message:   **Scenario 1:** Missing required fields.   * The system returns the error message **(MSG86)**   **Scenario 2:** Invalid IDNumber length.   * The system returns the error message **(MSG87)**   **Scenario 3:** Duplicate driver information.   * The system returns the error message **(MSG88)**   **Scenario 4:** Submission failure due to a system or database error.   * The system returns the error message **(MSG89)** |

### 

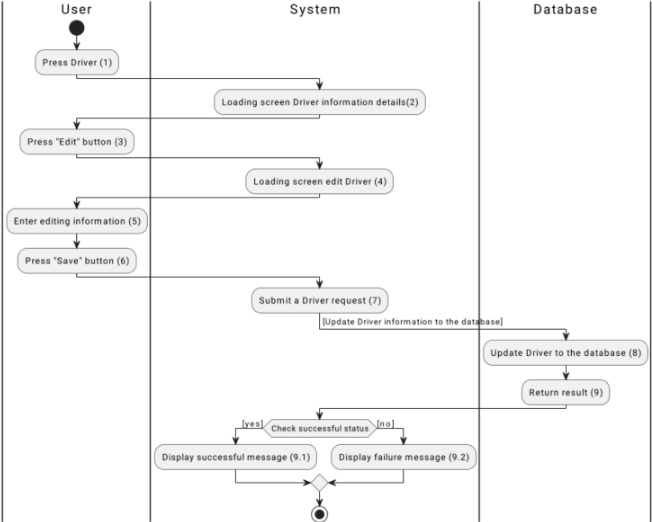
### UC22: Edit Driver

### 

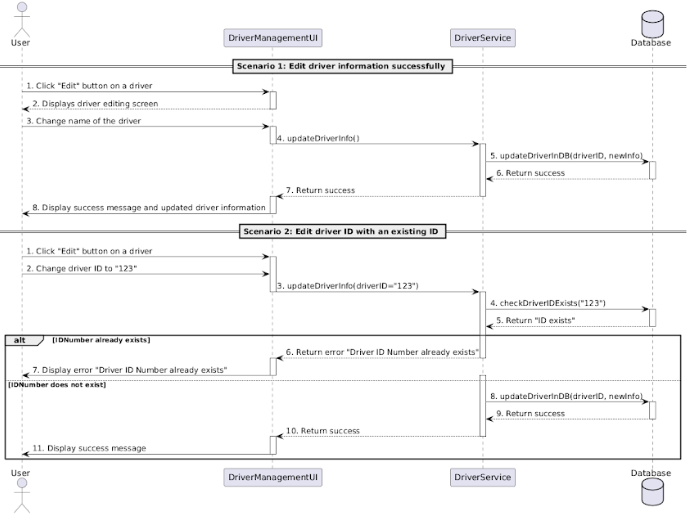
| **Name** | **Edit Driver’s Form** |
| --- | --- |
| **Description** | This use case allows authorized users to modify details of an existing driver, including its Name, IDNumber, Email, PhoneNumber. |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user access into the Driver Management page:   * Click the “Edit” button * Change the properties of the driver |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role. |
| **Post-condition** | * The selected driver is updated with the new information and saved in the database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Driver Information Details" screen when the user clicks on a driver in the Driver Management page. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a confirmation dialog with a warning message when the user clicks the "Delete" button on the "Driver Information Details" screen. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion of a driver, the system validates and processes the request as follows:   * Dependency Check:   The system checks if the selected driver has active connections (e.g., assigned vehicles or tasks) using:  [checkDriverConnections(driverID)]  If active connections exist, the system shows message **(MSG90)**  Else, proceed to the next step.   * Data Submission:   If no active connections exist, the system deletes the driver from the database:   * On success, the system shows message **(MSG91)** * On failure, the system shows message **(MSG92)** |

### 

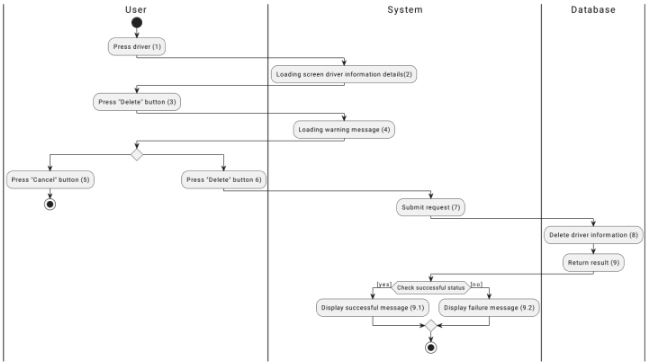
### UC23: Delete Driver

### 

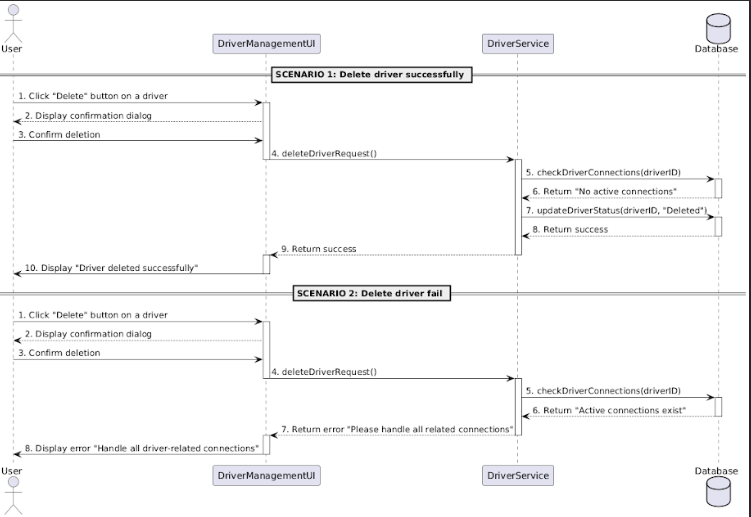
| **Name** | **Delete Driver’s Form** |
| --- | --- |
| **Description** | This use case allows authorized users to remove a driver from the system. |
| **Actor** | Admin, Vehicle Manager |
| **Trigger** | When user access into the Driver Management page:   * Click the “Delete” button * Confirm the deletion |
| **Pre-condition** | * The user must be logged in with an Admin or Vehicle Manager role. |
| **Post-condition** | * The selected driver is removed from the driver list in the system's database. |

### 

### Activities Flow

**

*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Delete Driver Information** screen when the user clicks the "Delete" button for a selected driver. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user confirms the deletion request, the system displays a loading indicator while processing the deletion. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the deletion, the system checks for unresolved dependencies (e.g., active trips, assignments, or vehicle connections) associated with the driver:   * Validation Step:   [checkDependencies(DriverID)]   * If unresolved dependencies exist, the system shows the message: **(MSG93)** * If all dependencies are resolved, the system updates the driver status to **"Deleted"**: **Submission Step**:   + If the deletion is successful, the system shows the message: **(MSG94)**   + If the deletion fails, the system shows the message: **(MSG95)** |

### 

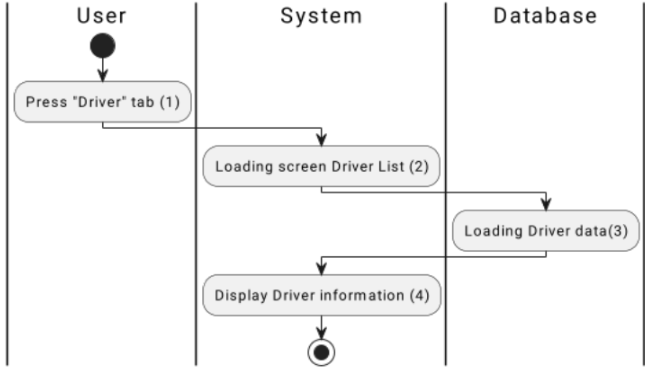
### UC24: View Driver List

### 

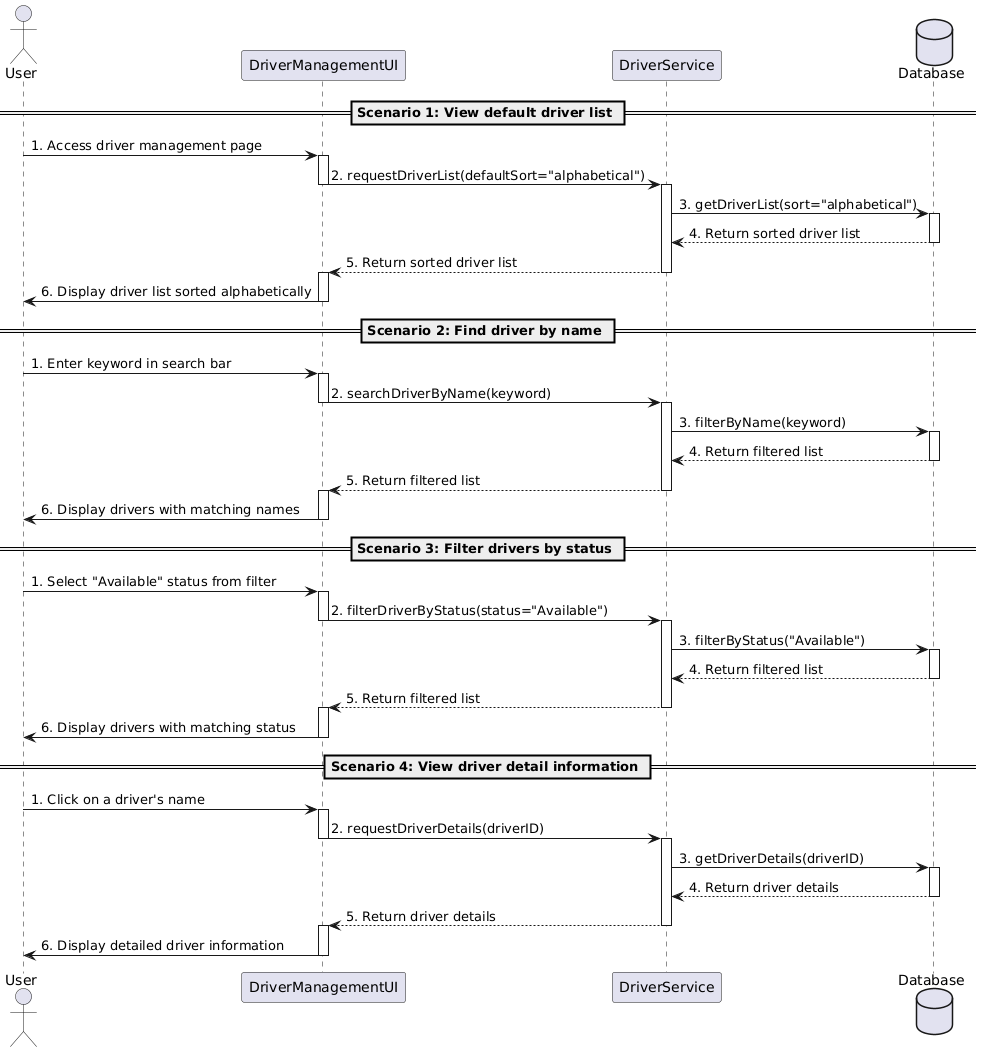
| **Name** | **View Driver List** |
| --- | --- |
| **Description** | This use case allows users to view the list of drivers. |
| **Actor** | Admin, Vehicle Manager, Logistics Manager, Warehouse Manager |
| **Trigger** | * When user access into the Driver Management page |
| **Pre-condition** | * The user must be logged in |
| **Post-condition** | * The system displays a list of all existing drivers |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Driver Management Page** when the user selects the "Driver" tab. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the driver list data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules: The system retrieves the driver list data by invoking requestDriverList() API, applying default sorting criteria (alphabetically) if no filter is selected.  If the data retrieval process fails, the system displays an error message:   * **Error Message:** **(MSG96)**  Filtering and Searching Rules: **Search by Driver Name:**   * The system filters and displays drivers matching the entered keyword in the search bar using searchDriverByName() API.   **Filter by Status:**   * The system filters and displays drivers matching the selected status (e.g., "Available") using filterDriverByStatus() API.  Driver Details Rules: When the user selects a driver, the system fetches the driver's detailed information by invoking requestDriverDetails() API.  The system displays a **loading indicator** while retrieving driver details.  If the retrieval process fails, the system displays an error message:   * **Error Message:** **(MSG97)**  Displaying Rules: If the data is successfully retrieved:   * The driver list is displayed alphabetically by default. * Filters (status) or search criteria (driver name) are applied to display the filtered list. * Detailed information for a selected driver is displayed in a new section or dialog. |

### 

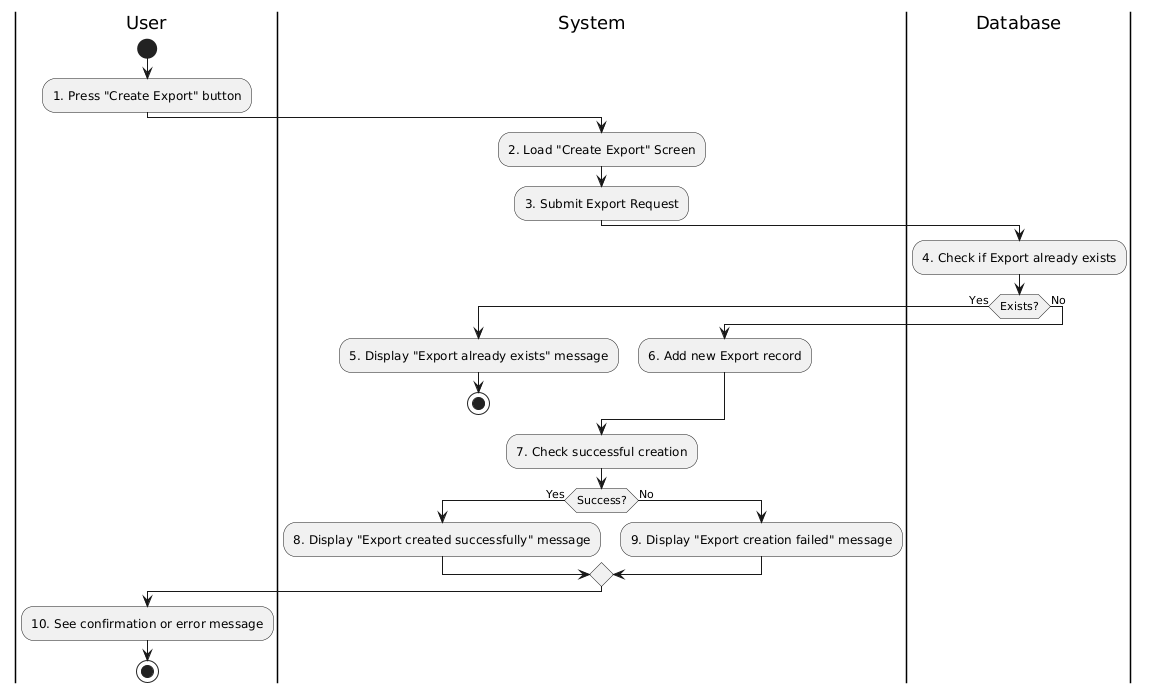
### UC25: Create Export Order

### 

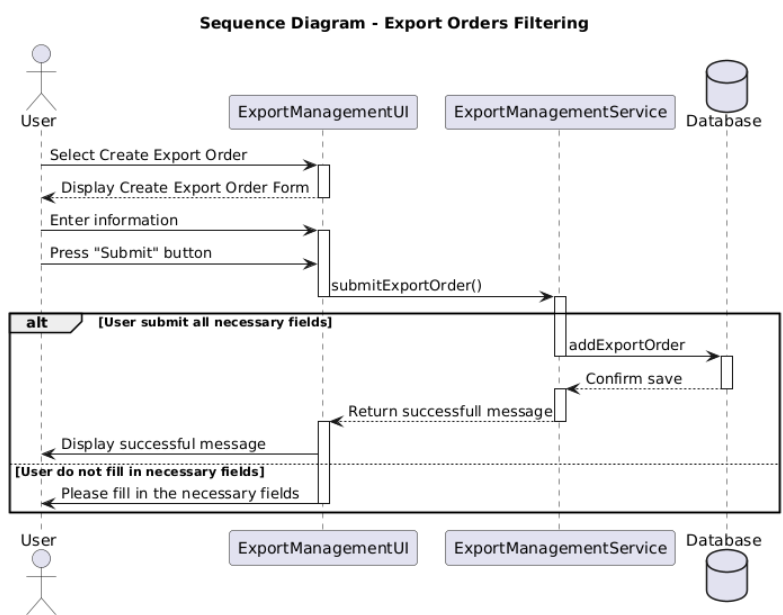
| **Name** | **Create Export Order** |
| --- | --- |
| **Description** | This use case allows users to add a new export order to the system by providing details such as product type, quantity, destination. |
| **Actor** | Admin, Sale staff, Customer, Warehouse manager |
| **Trigger** | When user accesses into the Import Management page:   * Click the "Create Import" * Enter the necessary details |
| **Pre-condition** | * The user must be logged in with an Admin, Sales staff, Customer, Warehouse Manager. |
| **Post-condition** | * A new Import order, including its detail (ID, orderID, salesID, customerID, destination, total price, status), is added to the Import order list and saved in the system's database. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Create Import Order" screen when the user clicks the "Create Import" button. (Refer to the link “UI Design” and section “View Description”) |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Submit" button, the system displays a loading indicator while waiting for a response from the server. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Submit" button, the system validates the entered data and performs the following actions:   * Input Validation:   [checkIsEmpty(ProductName)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(Unit)] && [checkIsEmpty(SupplierID)] && [checkIsEmpty(Date)] && [checkIsEmpty(TotalCost)]   * If any required fields (ProductName, Quantity, Unit, SupplierID, Date, TotalCost) are empty, the system shows the message **(MSG98)** * Else, proceed to the next step. * Duplicate Validation:   If input is valid and no duplicates exist, the system saves the import order data.   * On success, the system shows the message **(MSG99)** * On failure, the system shows the message **(MSG100)** * Data Submission:   If input is valid and no duplicates exist, the system saves the product data.  On success, the system shows the message **(MSG101)**  On failure, the system shows the message **(MSG102)**   * Error Message:   Scenario 1: Missing required fields.   * The system returns the error message **(MSG103)**   Scenario 2: Duplicate import order.   * The system returns the error message **(MSG104)**   Scenario 3: Submission failure due to a system or database error.   * The system returns the error message **(MSG105)** |

### 

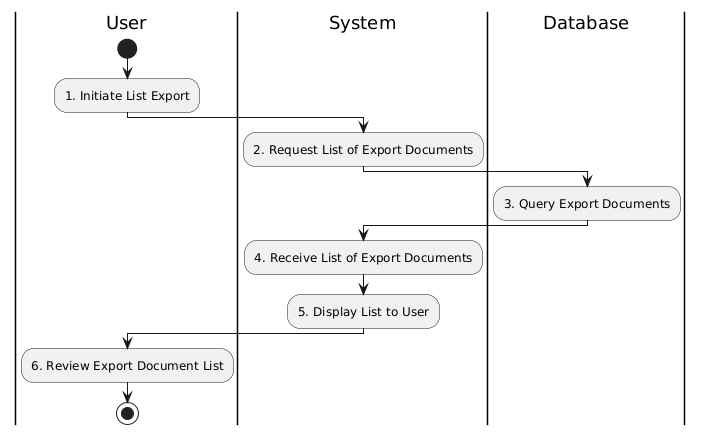
### UC26: View Export List

### 

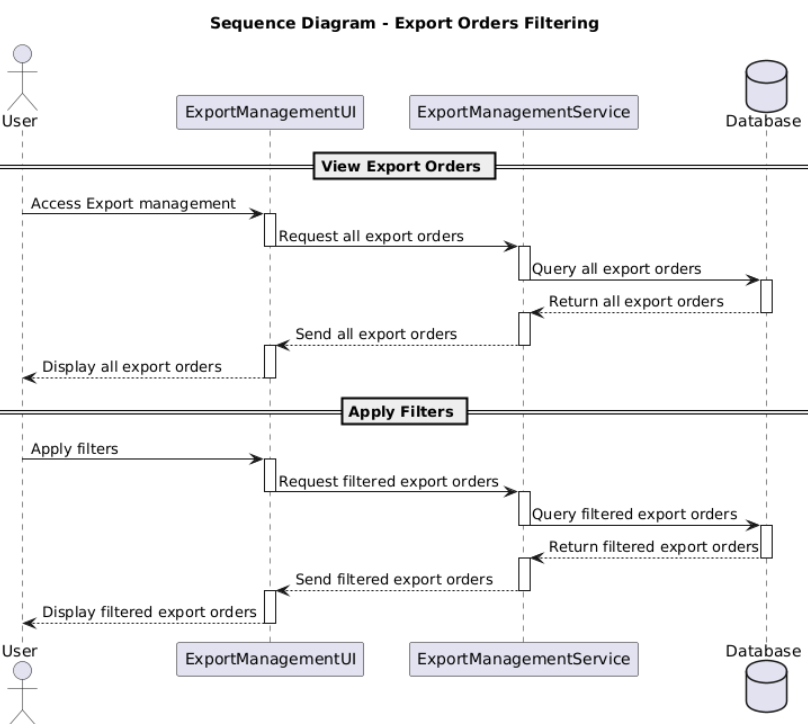
| **Name** | **View Import Order List** |
| --- | --- |
| **Description** | This use case allows user to view the list of Import orders and their details, including ID, orderID, customer, sale staff, destination, total price, status. |
| **Actor** | Admin, Warehouse manager, Logistic manager, User, Sale staff |
| **Trigger** | * When user accesses into the Import Management page. |
| **Pre-condition** | * The user must be logged in. |
| **Post-condition** | * The system displays a list of Import orders. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Import Management Page** when the user selects the "View Import List" option. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a **loading indicator** while fetching the import orders data from the database. |
| ***(4)*** | ***BR3*** | Data Fetching Rules: The system retrieves the Import orders list data by invoking getExportOrders() API.  If the data retrieval process fails, the system displays an error message **(MSG106)** Filtering Rules: **Filter by Date:**   * The system filters and displays Import orders based on the selected date range using the filterExportOrders(date) API.   **Filter by Status:**   * The system filters and displays Import orders based on the selected status using the filterExportOrders(status) API.   **Filter by Product Type:**   * The system filters and displays Import orders based on the selected product type using the filterImportOrders(productType) API.  Updating Import Records Rules: The system allows users to edit Import records by navigating to the "Edit Import" screen.  Upon submitting the edit request:   * The system checks if the selected Import order exists in the database by invoking checkImportExists(). * If the record does not exist, the system displays an error message: ***(MSG107)*** * If the record exists and is successfully updated, the system displays a success message ***(MSG108)*** * If the update fails, the system displays an error message ***(MSG109)***  Displaying Rules: If the data is successfully retrieved:   * The export orders list is displayed in its entirety, sorted alphabetically by default. * Filters (date, status, product type) are applied to display the filtered list. * For successfully edited export orders, updated details are reflected in the displayed list. |

### 

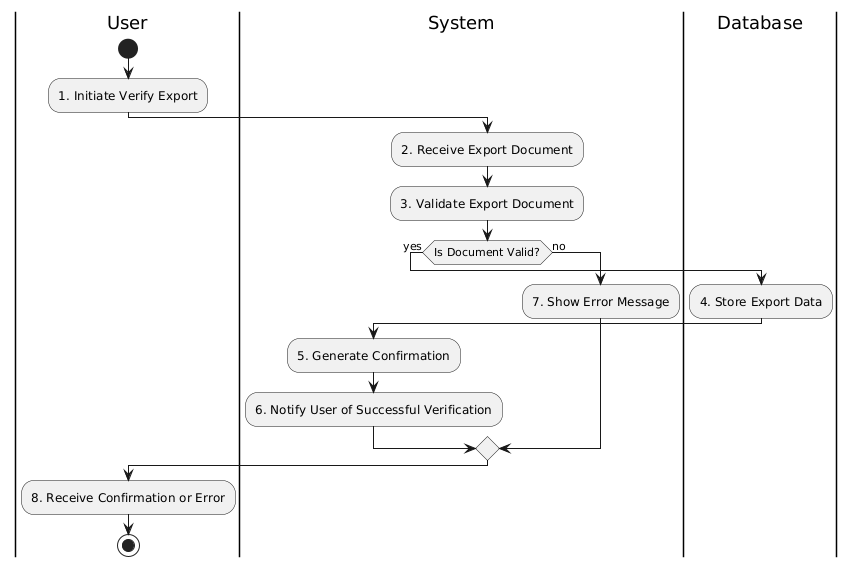
### UC27: Confirm Export Order

### 

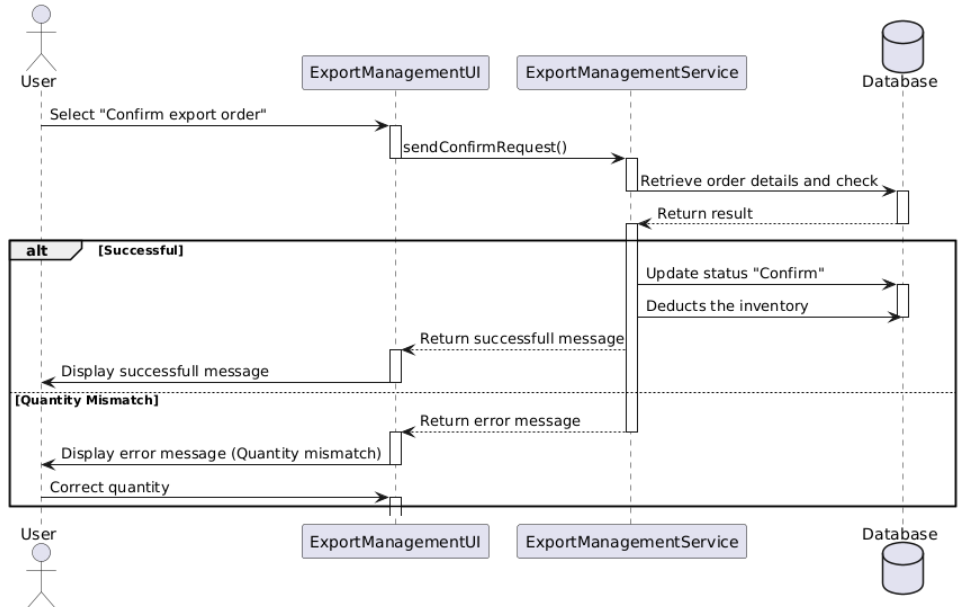
| **Name** | **Confirm Import Order** |
| --- | --- |
| **Description** | This use case describes the process for confirming an Import order. Upon confirmation, the system updates the order status and adjusts the inventory. |
| **Actor** | Admin, Warehouse manager, User, Sale staff |
| **Trigger** | When user accesses into the Import Management page:   * Click the "Confirm Import Order" for pending Import order |
| **Pre-condition** | * There is a pending Import order in the system. * The user has permission to confirm Import orders. * Sufficient inventory is available to fulfill the order. |
| **Post-condition** | * **Success**: The order status is updated to "Confirmed," and the inventory is deducted according to the dispatch details. * **Failure**: The order remains in pending, and user enters new details. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the **Confirm Import Form** screen when the user selects the "Confirm Import Order" button in the Import Management section. |
| *(4)* | *BR2* | **Loading Screen Rules:**  The system retrieves the details of the import order, including:   * Product name. * Quantity. * Unit. * Supplier. * Total cost. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * Validation Rules:   **Document Validation:**   * The system validates the import document to ensure all fields are complete and in the correct format. * If the document is invalid, the system displays (**MSG110)** * If the document is valid, the system proceeds to verify the order details.   **Quantity Verification:**   * The system verifies that the product quantities match the expected values. * If there is a mismatch, the system displays (**MSG111)**   If all validations pass:   * The system updates the order status to "Confirmed" using the updateOrderStatusToConfirmed() API. * The system stores the confirmed details (product name, quantity, unit, supplier, date, and total cost) in the database.   Once the import order is successfully confirmed, the system notifies the user with (**MSG112)** |

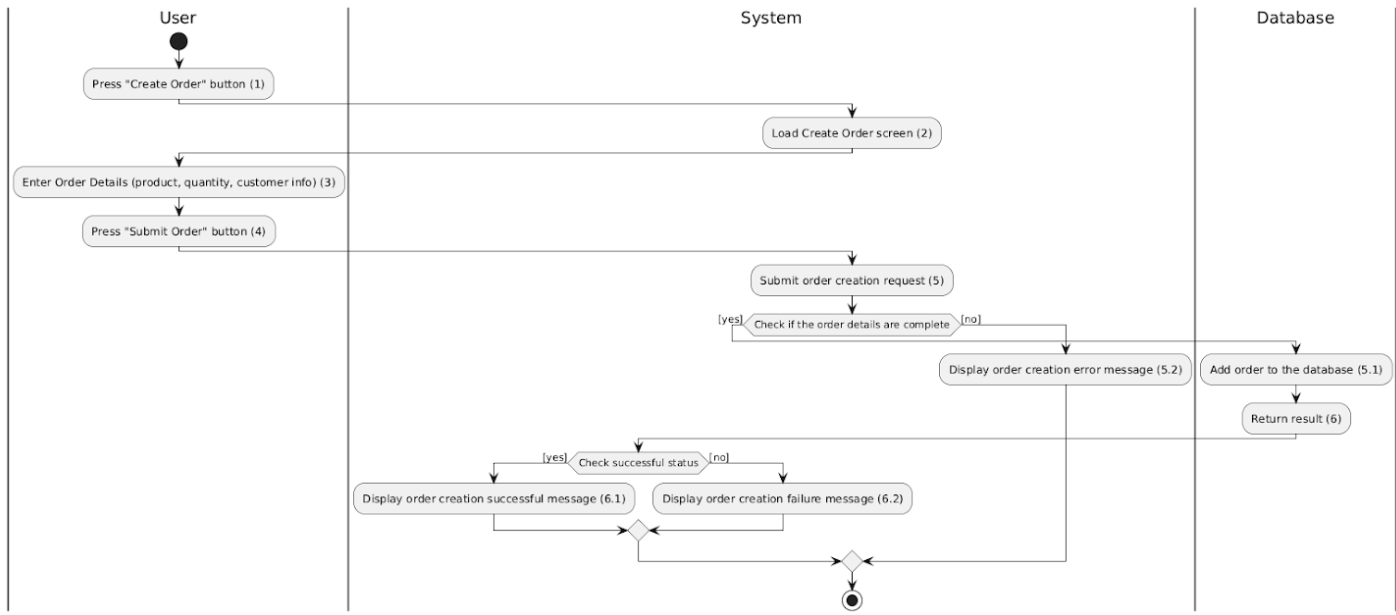
### UC28: Create Order

### 

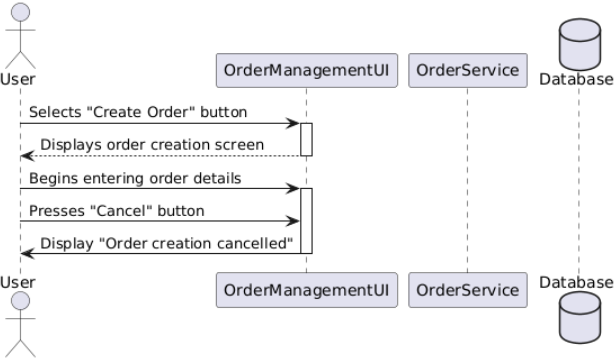
| **Name** | **Create Order** |
| --- | --- |
| **Description** | This use case describes the process where an authorized user creates a new order in the system. The system collects necessary order details, validates the input, and stores the order in the database for further processing. |
| **Actor** | Admin, Logistic manager |
| **Trigger** | * The user accesses the "Order Management" page and clicks the "Create New Order" button. |
| **Pre-condition** | * The user must be logged in with appropriate permissions (Admin or Sales Staff). * The required customer and product information is available in the system. * The inventory has sufficient stock to fulfill the order. |
| **Post-condition** | * Success: The order is created and stored in the database with a "Pending" status. The system updates relevant inventory quantities and provides the user with an order confirmation. * Failure: If validation fails or inventory is insufficient, the system notifies the user, and the order is not created. |

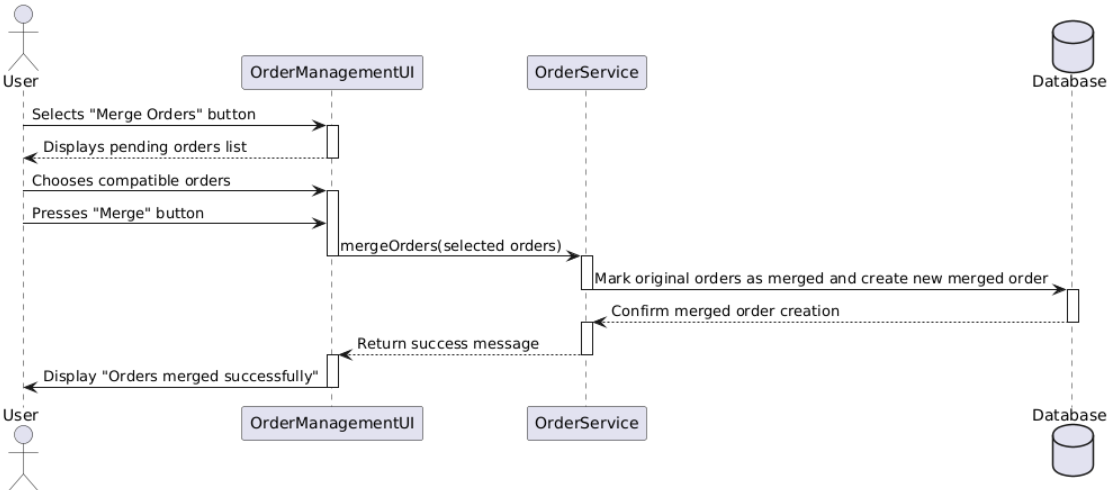
### 

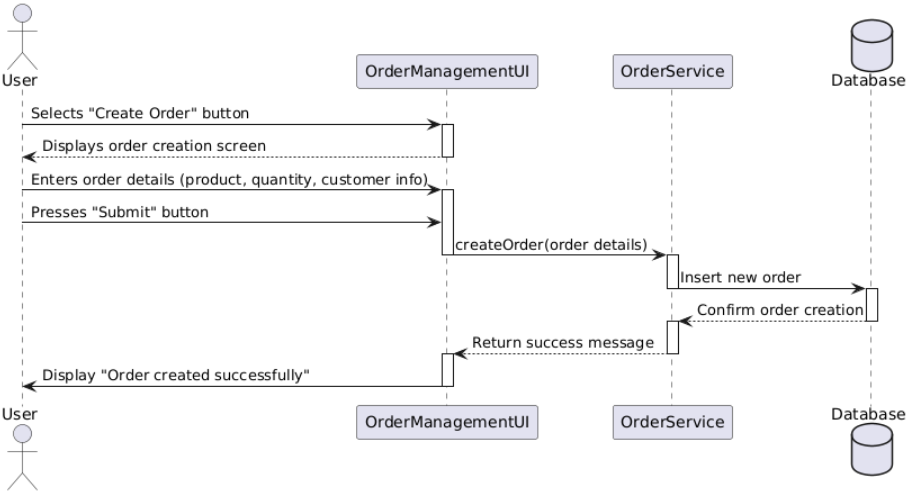
### Activities Flow



*Figure 1: Activities Flow*

**

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Create New Order" screen when the user clicks the "Create New Order" button. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Submit" button, the system displays a loading indicator while processing the order creation request. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * When the user clicks the "Submit" button, the system validates the entered data and performs the following actions:   **Input Validation:**  [checkIsEmpty(ProductDetails)] && [checkIsEmpty(Quantity)] && [checkIsEmpty(CustomerInfo)]   * If any required fields (ProductDetails, Quantity, CustomerInfo) are empty, the system shows the message **(MSG113)** * Else, proceed to the next step.   **Duplicate Validation:**  [checkDuplicateOrder(CustomerID, ProductID)]   * If an identical order already exists in the system, the system shows the message **(MSG114) Data Submission:** * If input is valid, inventory is sufficient, and no duplicate exists, the system saves the order data.   + On success, the system shows the message **(MSG115)**   + On failure, the system shows the message **(MSG116)** |

### 

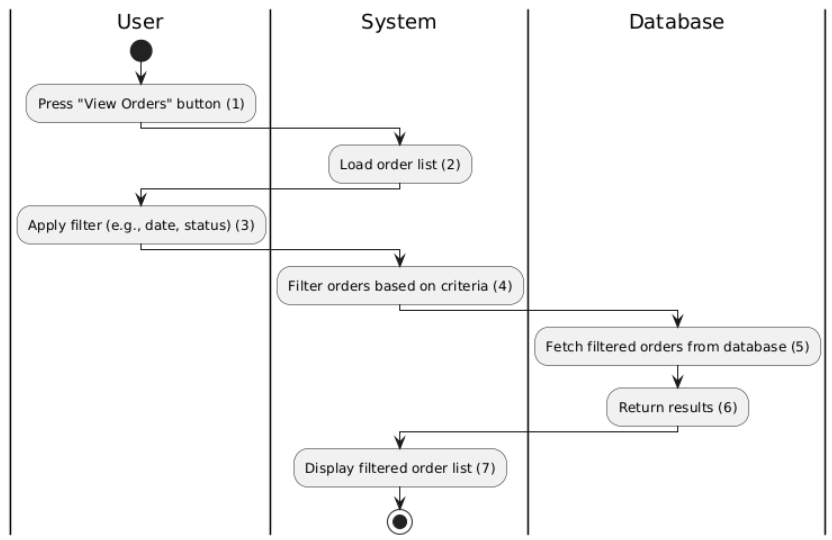
### UC29: View Order

### 

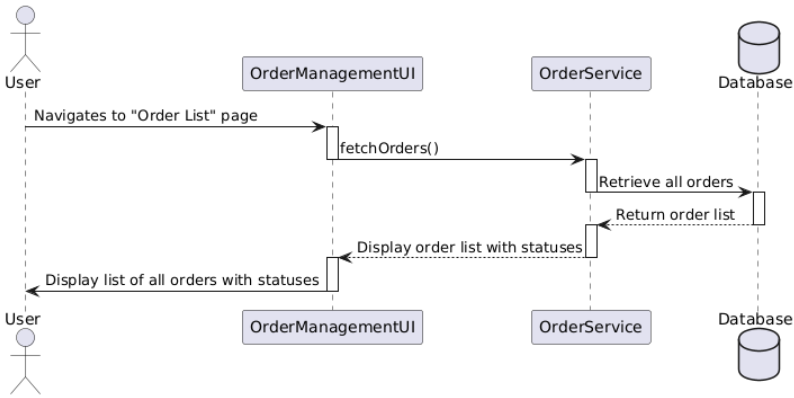
| **Name** | **View Order** |
| --- | --- |
| **Description** | This use case enables authorized users to view a list of orders stored in the system. Users can filter orders by criteria such as date, status, or customer name to streamline order management and decision-making. |
| **Actor** | Admin, Logistic manager, Warehouse manager, Sale staff, Vehicle Manager |
| **Trigger** | * The user navigates to the "Order Management" page and selects the "View Orders" option. |
| **Pre-condition** | * The user must be logged in with sufficient permissions to access order information (Admin or Inventory Manager role). * Orders must exist in the database. |
| **Post-condition** | * Success: The system displays the order list, allowing users to apply filters and view specific order details as needed. * Failure: If no orders exist or an error occurs, the system displays a message notifying the user. |

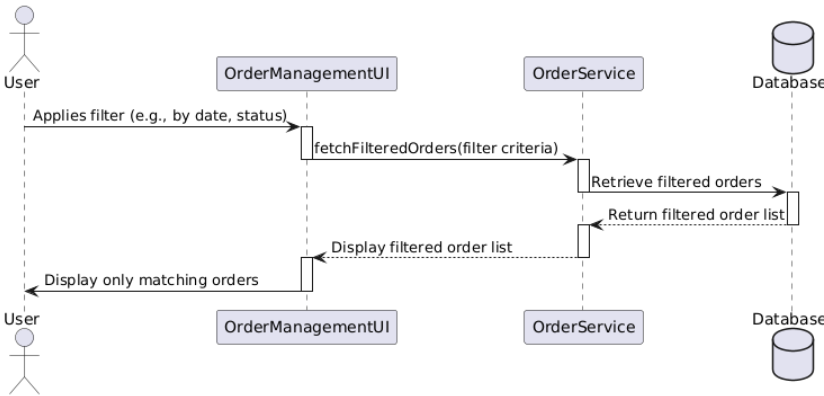
### 

### Activities Flow



*Figure 1: Activities Flow*

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Order List" screen when the user selects the "View Orders" option. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user applies filters (e.g., by date or status), the system displays a loading indicator while retrieving the filtered results from the database. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * The system validates the filter criteria, retrieves matching orders, and performs the following actions:   **Filter Validation:**  [validateFilterCriteria(Criteria)]   * If the entered filter criteria are invalid (e.g., unsupported date format, non-existent status), the system shows the message **(MSG117)** * Else, proceed to the next step.   **Order Retrieval:**  [fetchOrdersBasedOnCriteria(Criteria)]   * If no orders match the filter criteria, the system shows the message **(MSG118)**   **Data Display**:  If matching orders are found, the system displays the filtered order list.   * On success, the system allows the user to view the details of a specific order. * If an error occurs during retrieval, the system shows the message **(MSG119)** |

### 

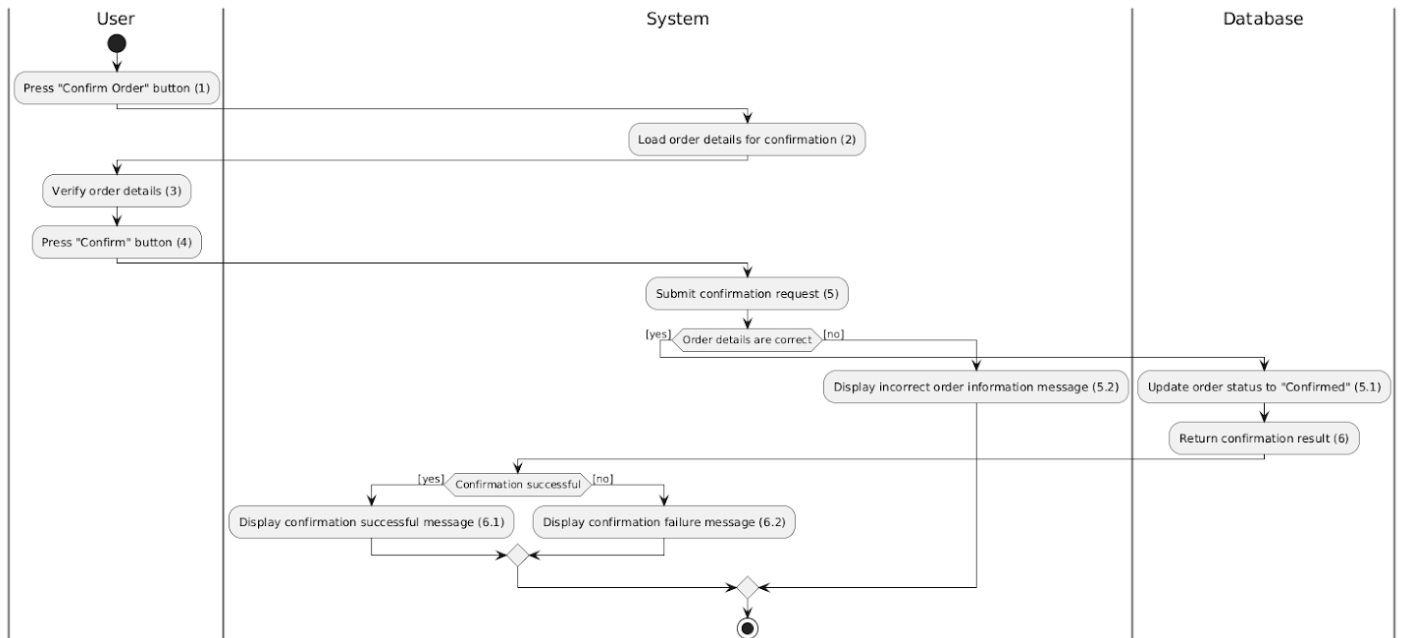
### UC30: Confirm Order

### 

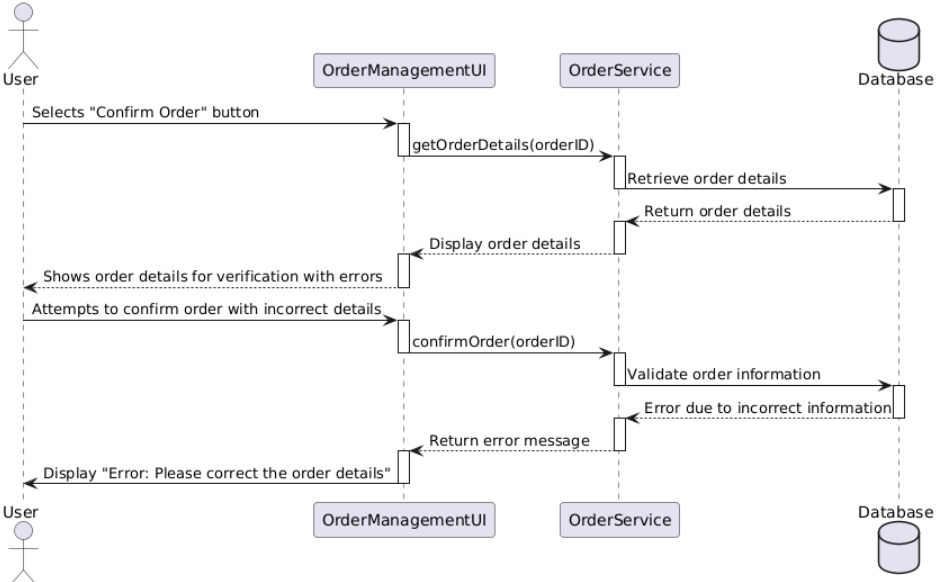
| **Name** | **Confirm Order** |
| --- | --- |
| **Description** | This use case allows authorized users to confirm pending orders. During confirmation, the system verifies order details, updates the order status, and adjusts related inventory data if applicable. |
| **Actor** | Admin, Logistic manager |
| **Trigger** | * The user accesses the "Order Management" page and clicks the "Confirm Order" button for a specific order. |
| **Pre-condition** | * The user must be logged in with sufficient permissions to confirm orders. * The selected order must be in a "Pending" status. * All required order details (e.g., products, quantities) must be accurate and complete. |
| **Post-condition** | * Success: The order status is updated to "Confirmed," and a confirmation message is displayed. * Failure: If any details are incorrect or validation fails, the system displays an error message, and the order remains unconfirmed. |

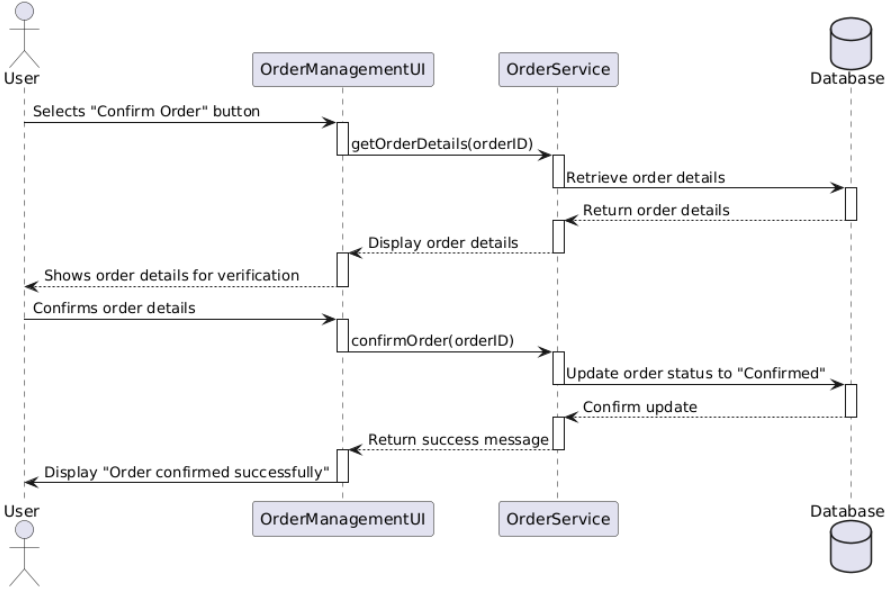
### 

### Activities Flow



*Figure 1: Activities Flow*

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Order Details" screen for confirmation when the user clicks the "Confirm Order" button. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a loading indicator while validating the order details and updating the order status. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user confirms the order, the system validates the order details and performs the following actions:   * Order Status Validation:   [validateOrderStatus(orderID)]  If the order status is not "Pending," the system shows the message **(MSG120)**   * Order Details Validation:   [validateOrderDetails(ProductDetails, Quantity)]  If the product details or quantities are incorrect, the system shows the message **(MSG121)**   * Inventory Validation:   [checkInventoryAvailability(ProductID, Quantity)]  If inventory levels are insufficient, the system shows the message **(MSG122)**   * Confirmation Submission:   If all validations pass, the system updates the order status to "Confirmed."   * On success, the system shows the message **(MSG123)** * On failure, the system shows the message **(MSG124)** |

### 

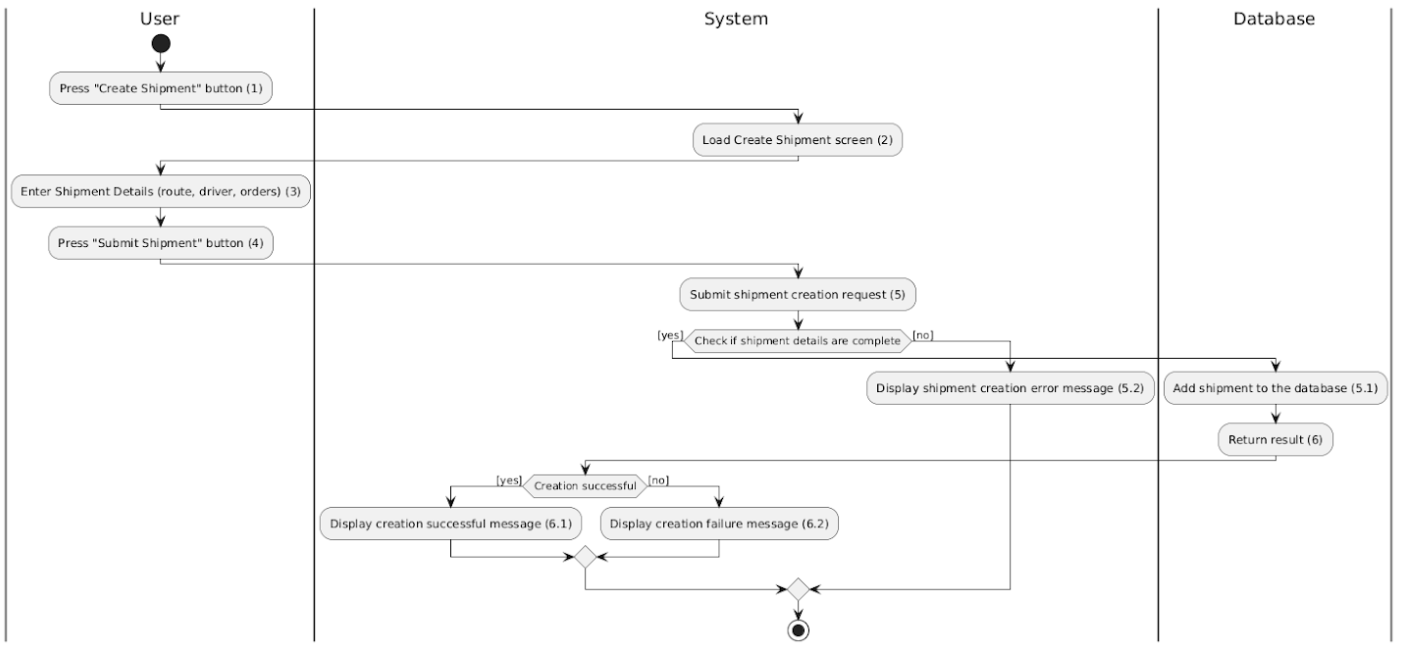
### UC31: Create Shipment

### 

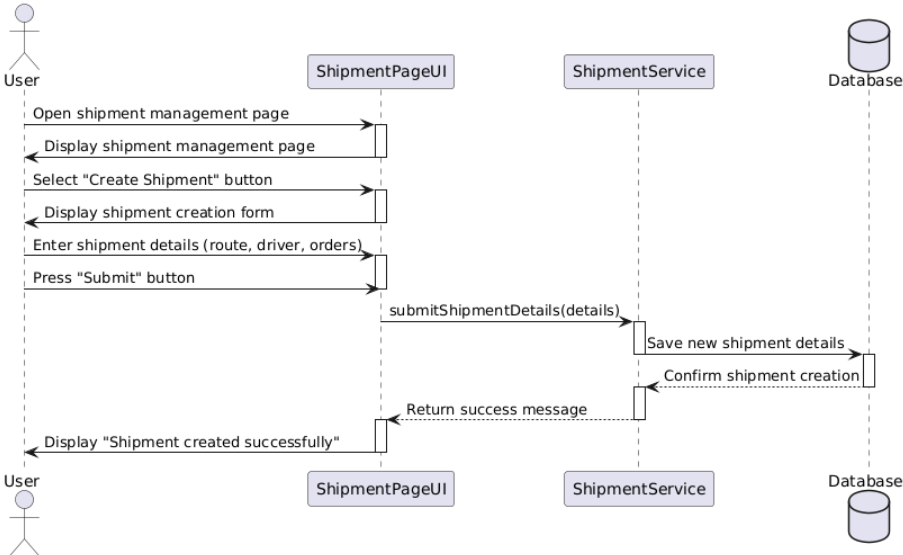
| **Name** | **Create Shipment** |
| --- | --- |
| **Description** | This use case enables authorized users to create a new shipment in the system by providing shipment details such as route, driver, and related orders. The system validates the input, saves the shipment information to the database, and confirms the creation. |
| **Actor** | Admin, Logistic manager |
| **Trigger** | * The user accesses the "Shipment Management" page and clicks the "Create Shipment" button. |
| **Pre-condition** | * The user must be logged in with sufficient permissions to manage shipments. * Necessary data (e.g., order details, driver information) must exist in the system. |
| **Post-condition** | * Success: A new shipment is created, saved in the database, and a success message is displayed to the user. * Failure: If input validation fails, the system displays an error message, and the shipment is not created. |

### 

### Activities Flow



*Figure 1: Activities Flow*

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Create Shipment" screen when the user clicks the "Create Shipment" button. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user clicks the "Submit" button, the system displays a loading indicator while processing the shipment creation request. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**   * When the user clicks the "Submit" button, the system validates the entered data and performs the following actions:   **Input Validation:**  [checkIsEmpty(Route)] && [checkIsEmpty(Driver)] && [checkIsEmpty(OrderDetails)]   * If any required fields (Route, Driver, OrderDetails) are empty, the system shows the message **(MSG125)** * Else, proceed to the next step.   **Driver Availability Validation:**  [checkDriverAvailability(DriverID)]   * If the assigned driver is not available, the system shows the message **(MSG126)**   **Data Submission:**  If input is valid and no errors are found, the system saves the shipment data.   * On success, the system shows the message **(MSG127)** * On failure, the system shows the message **(MSG128)** |

### 

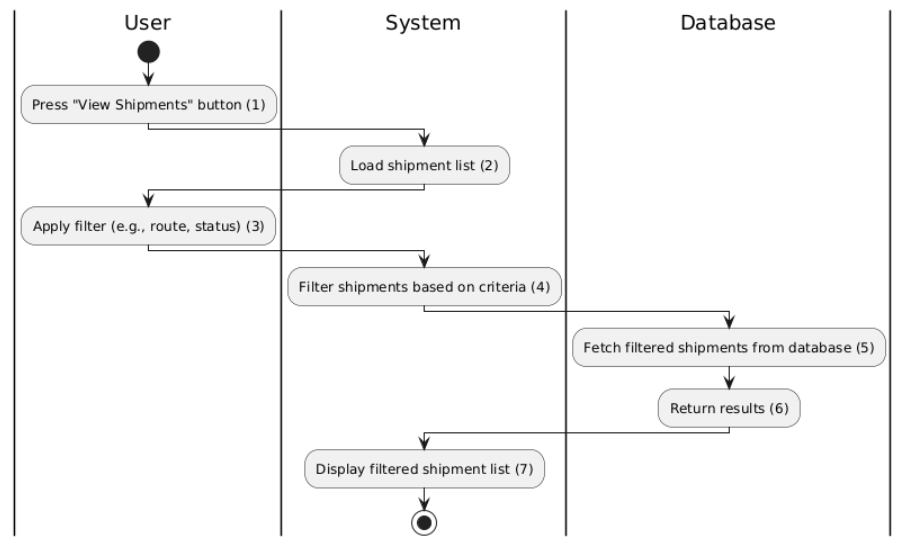
### UC32: View Shipment

### 

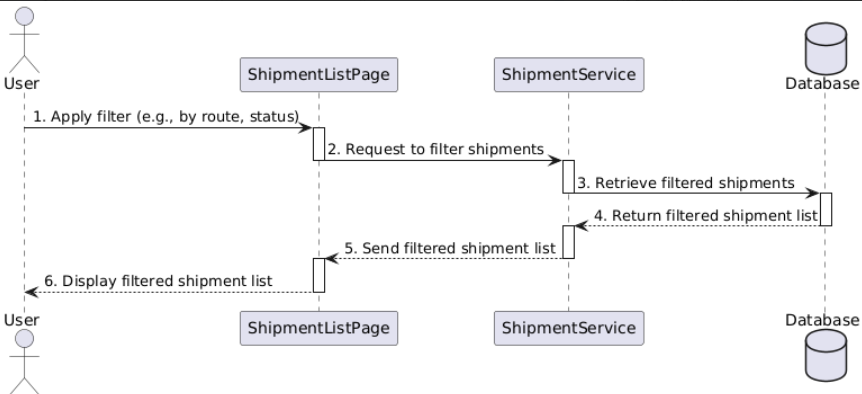
| **Name** | **View Shipment** |
| --- | --- |
| **Description** | This use case allows authorized users to view a list of shipments stored in the system. Users can filter the shipment list by criteria such as route, status, or assigned driver for efficient shipment management and tracking. |
| **Actor** | Admin, Logistic manager, Warehouse manager, Sale staff, Vehicle Manager |
| **Trigger** | * The user accesses the "Shipment Management" page and selects the "View Shipments" option. |
| **Pre-condition** | * The user must be logged in with sufficient permissions to access shipment data. * Shipment records must exist in the database. |
| **Post-condition** | * Success: The system displays a filtered or complete list of shipments, depending on the user's input criteria. * Failure: If no shipments match the criteria or an error occurs, the system notifies the user. |

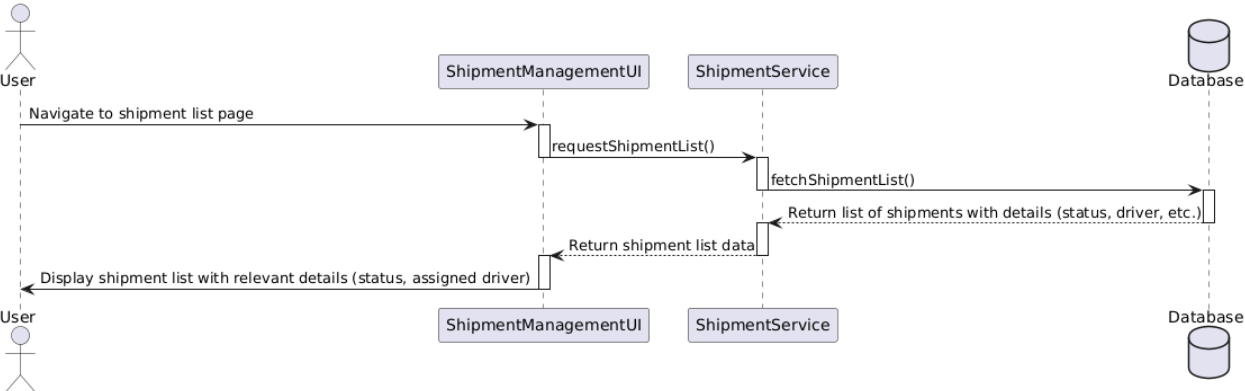
### 

### Activities Flow



*Figure 1: Activities Flow*

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Shipment List" screen when the user selects the "View Shipments" option. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * When the user applies filters (e.g., by route, status, assigned driver), the system displays a loading indicator while retrieving the filtered shipment data. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  The system validates the filter criteria and performs the following actions:   * Filter Validation:   [validateFilterCriteria(Criteria)]  If the entered filter criteria are invalid (e.g., unsupported date range, invalid status), the system shows the message **(MSG129)**   * Shipment Retrieval:   [fetchShipmentsBasedOnCriteria(Criteria)]  If no shipments match the filter criteria, the system shows the message **(MSG130)**   * Data Display:   If matching shipments are found, the system displays the filtered shipment list.   * On success, the system allows the user to view shipment details. * If an error occurs during retrieval, the system shows the message **(MSG131)** |

### 

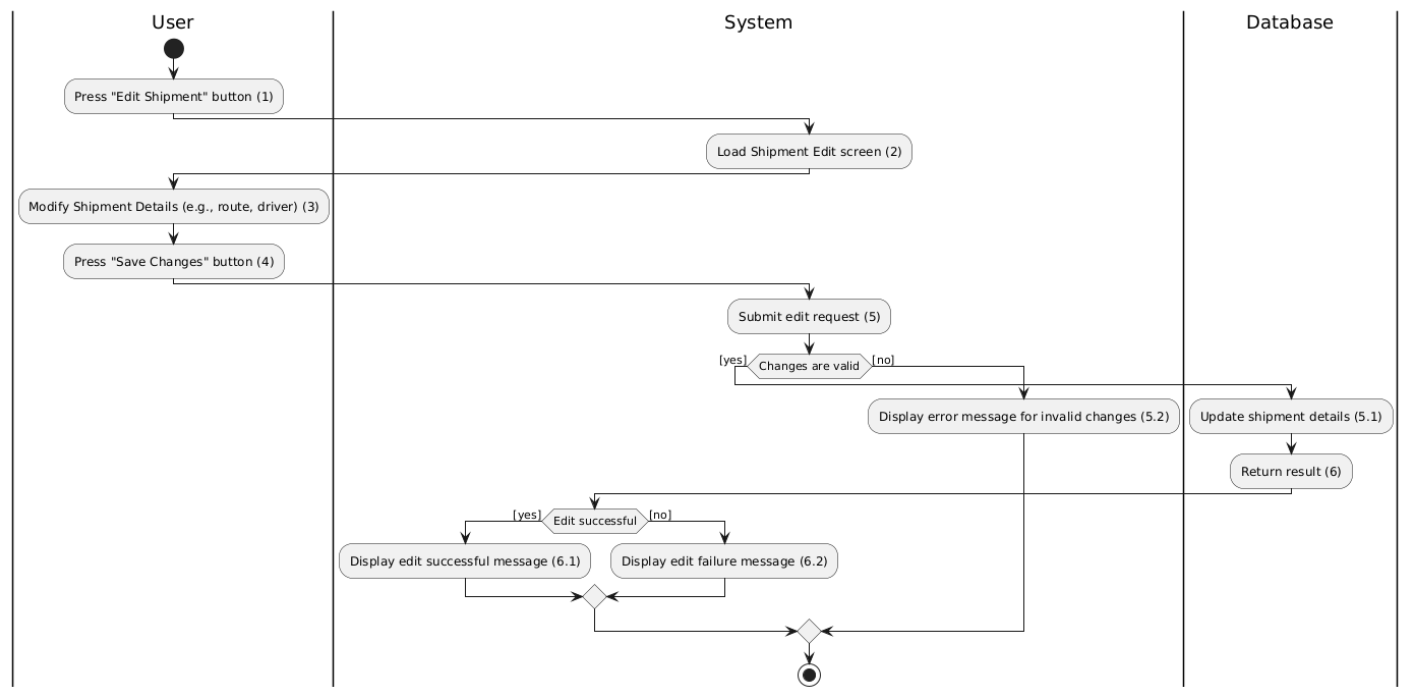
### UC33: Edit Shipment

### 

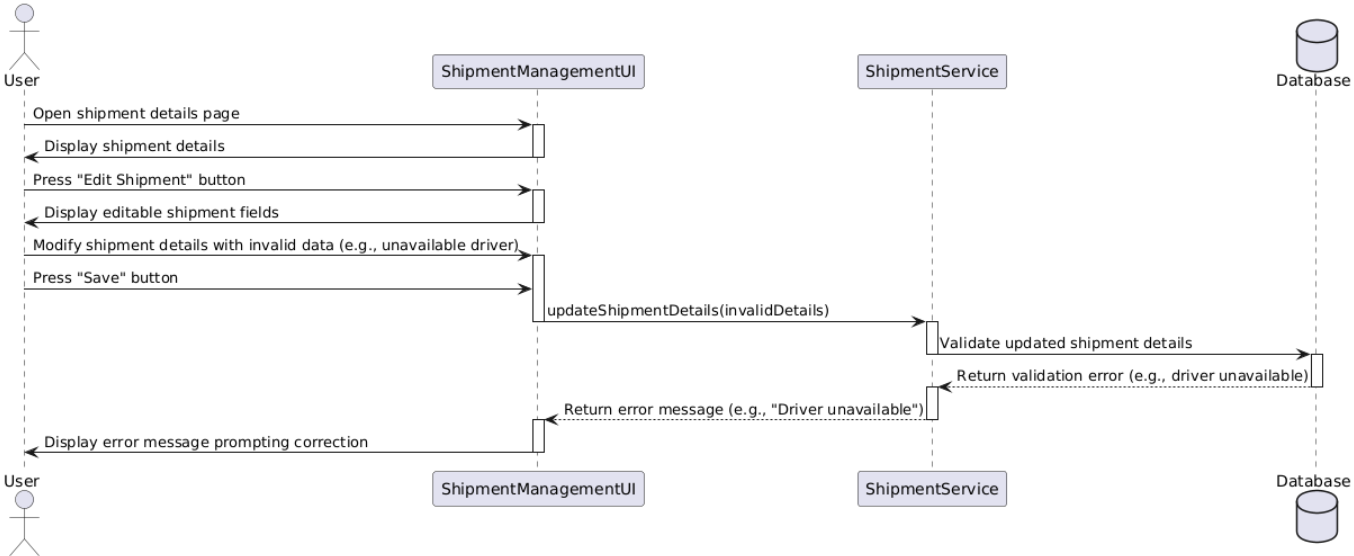
| **Name** | **Edit Shipment** |
| --- | --- |
| **Description** | This use case allows authorized users to modify shipment details, such as the route, driver, and orders. Changes are validated and updated in the database to ensure accurate shipment tracking and management. |
| **Actor** | Admin, Logistic manager |
| **Trigger** | * The user accesses the shipment details page and clicks the "Edit Shipment" button. |
| **Pre-condition** | * The user must be logged in with appropriate permissions to edit shipment details. * The shipment to be edited must exist in the system. |
| **Post-condition** | * Success: The shipment details are updated in the database, and a success message is displayed to the user. * Failure: If validation fails or the input data is invalid, the system displays an error message, and the shipment details are not updated. |

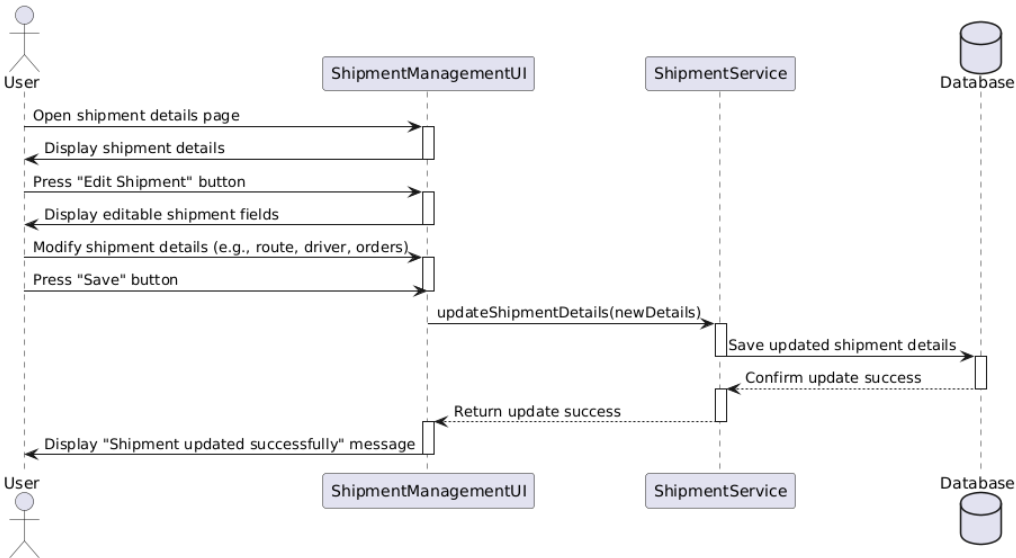
### 

### Activities Flow



*Figure 1: Activities Flow*

**

**

#### 

#### Business Rules

#### 

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR1* | **Loading Screen Rules:**   * The system displays the "Shipment Edit" screen when the user clicks the "Edit Shipment" button. |
| *(4)* | *BR2* | **Loading Screen Rules:**   * The system displays a loading indicator while validating and saving the updated shipment details. |
| ***(4)*** | ***BR3*** | **Submitting Rules:**  When the user clicks the "Save Changes" button, the system validates the input and performs the following actions:   * Input Validation:   [validateShipmentDetails(Route, Driver, Orders)]   * If any required fields (e.g., Route, Driver, Orders) are missing, the system shows the message **(MSG132)** * Order Validation:   [validateOrderDetails(OrderIDs)]  If any orders are invalid (e.g., already assigned to another shipment), the system shows the message **(MSG133):**   * Data Update Submission:   If all validations pass, the system updates the shipment details in the database.   * On success, the system shows the message **(MSG134)** * On failure, the system shows the message **(MSG135)** |

### 

# List Description

[List Description.xlsx](https://docs.google.com/spreadsheets/d/1ckGOHhHtBXDwclqPNK0Cx3NlQ59XjeQl/edit?usp=drive_link&ouid=116590313859269109657&rtpof=true&sd=true)

# View Description



# Non-functional Requirements

| Actor  Function | Admin | Warehouse Manager | Vehicle Manager | Sale | Accountant | Human Resource | Logistic Manager |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Authentication** |  |  |  |  |  |  |  |
| Sign Up | x | x | x | x | x | x | x |
| Sign In | x | x | x | x | x | x | x |
| Forgot password | x | x | x | x | x | x | x |
| Change password | x | x | x | x | x | x | x |
| **Partner Management** |  |  |  |  |  |  |  |
| Create information | x |  |  |  |  |  | x |
| Edit information | x |  |  |  |  |  | x |
| Delete information | x |  |  |  |  |  | x |
| View list | x |  |  |  |  |  | x |
| Create partner type | x |  |  |  |  |  | x |
| Edit partner type | x |  |  |  |  |  | x |
| View list | x |  |  | x | x | x | x |
| **Product Management** |  |  |  |  |  |  |  |
| Create information | x |  |  | x |  |  |  |
| Edit information | x |  |  | x |  |  |  |
| Delete information | x |  |  | x |  |  |  |
| View list | x | x | x | x | x | x | x |
| **Warehouse Management** |  |  |  |  |  |  |  |
| Create information | x | x |  |  |  |  |  |
| Edit information | x | x |  |  |  |  |  |
| Delete information | x | x |  |  |  |  |  |
| View list | x | x | x | x | x | x | x |
| Update the actual status of the warehouse for the day | x | x |  |  |  |  |  |
| Check the asset list of inventory | x | x |  |  |  |  |  |
| **Import Management** |  |  |  |  |  |  |  |
| Create information | x |  |  |  |  |  |  |
| Edit information | x |  |  |  |  |  |  |
| Delete information | x |  |  |  |  |  |  |
| View list | x | x |  | x |  |  | x |
| Confirm | x | x |  |  |  |  |  |
| Print | x | x |  |  |  |  |  |
| **Export Management** |  |  |  |  |  |  |  |
| Create information | x |  |  |  |  |  |  |
| Edit information | x |  |  |  |  |  |  |
| Delete information | x |  |  |  |  |  |  |
| View list | x | x |  | x |  |  | x |
| Confirm | x | x |  |  |  |  |  |
| Print | x | x |  |  |  |  |  |
| **Vehicle Management** |  |  |  |  |  |  |  |
| Create information | x |  | x |  |  |  |  |
| Edit information | x |  | x |  |  |  |  |
| Delete information | x |  | x |  |  |  |  |
| View list | x | x | x |  |  |  | x |
| **Driver Management** |  |  |  |  |  |  |  |
| Create information | x |  | x |  |  |  |  |
| Edit information | x |  | x |  |  |  |  |
| Delete information | x |  | x |  |  |  |  |
| View list | x | x | x |  |  |  | x |
| Vehicle coordination | x |  | x |  |  |  |  |
| **Order Management** |  |  |  |  |  |  |  |
| Create information | x |  |  |  |  |  | x |
| Edit information | x |  |  |  |  |  | x |
| Delete information | x |  |  |  |  |  | x |
| View list | x | x | x | x | x |  | x |
| **Shipment Management** |  |  |  |  |  |  |  |
| Create information | x |  |  |  |  |  | x |
| Edit information | x |  |  |  |  |  | x |
| View list | x | x | x | x | x | x | x |

# 

X: User has full permission to do the action.

X(\*): User has permission to do the action on his own items.

X(\*\*): User has permission to do the action on items sent to him only.

X (1): reading permission is specified for each Error Form item regarding to Location of the item. For example, items, which have Location = “London”, are only read by employees of the location.

## Performance Requirements

**Number of user**

* Number of concurrent user: 1000
* Number of business user: 500

**Data volume**

* Number of documents: 20 000 000
* Data growth rate: 4 500 000

**Level of availability**

❖ Actual uptime = 365 days \* 24 hours/day - 10 hours = 8,730 hours - 10 hours = 8,720 hours

❖ Total possible uptime = 365 days \* 24 hours/day = 8,760 hours

❖ Availability = (8,720 hours / 8,760 hours) x 100 = 99.54%

**Usage frequency**

Daily

## Implementation Requirements

**Location**

Viet Nam

**Read-only Duration**

1 day

**Read-only Timeframe**

4 hours

**Maintenance Window**

Annually

**Overall conversion timeline**

25th of every month, the Viet Nam users are payment dates.

**Other plans and activities**

[N/A]

# Other Requirements

| **List** | **Actor** | **Condition** |
| --- | --- | --- |
| *List warehouse* | *Warehouse Manager* | Able to archive items in “*list warehouse*” list by created date. |
| *List product* | *Warehouse Manager, Warehouse Staff* | *Can view, add, edit product on warehouse* |
| *List import order* | *Admin, Warehouse Manager, Logistic Manager* | Able to view, add, update import list |
| *List export order* | *Admin, Warehouse Manager, Logistic Manager* | Able to view, add, update export list |
| *List account* | *Admin* | Able to view, add, lock account |
| *List partner* | *Admin, Logistic Manager, Accountant* | Able to view, add, update partner |
| *List order* | *Logistic Manager, Sale* | Able to view, create order |
| *List shipment* | *Logistic Manager, Sale* | Able to view, update, create shipment |
| *List driver* | *Fleet Manager, Driver* | Able to view, update driver |
| *List vehicle* | *Fleet Manager* | Able to view, create, update, remove vehicle |

## 

## **Security Audit Function**

Check who has updated confirm import/export order.

Check who has updated shipment status.

Check who has updated the actual status of the warehouse for the day.

Check who has viewed or uploaded the product.

Check who has edited or deleted the shipment, warehouse.

# System Requirements

| **#** | **Site Name** | **Description** |
| --- | --- | --- |
| *1* | ***Server System*** | Robust server architecture optimised for logistics operations, supporting the complex processing of inventory management, order tracking, and fleet deployment. |
| *2* | ***Database*** | Dynamic Database Management System tailored for handling extensive logistics data, including inventory levels, shipment tracking, vehicle information, and user activity. Expected to manage detailed records of inventory and fleet movements. |
| *3* | ***Bandwidth*** | High-capacity network infrastructure to facilitate seamless data flow for tracking shipments, updating inventory, and managing fleet communications. Critical for supporting the data throughput necessary for logistics and warehouse operations.  Required Bandwidth: Sufficient to support real-time data synchronization across multiple locations, recommended to be at least 1 Gbps. |
| *4* | ***Security*** | Enhanced security framework to secure logistics data, protect against breaches, and ensure the confidentiality and integrity of operational data. Includes measures to prevent unauthorized vehicle tracking and data tampering. |
| *5* | ***Fast Response*** | The system is engineered for rapid data processing, enabling immediate updates from IoT-enabled warehouses and quick dispatching within the fleet management module.  Target Response Time: Under 2 seconds for critical logistics operations and data retrieval. |
| *6* | ***Backup and Data Recovery System*** | The system is engineered for rapid data processing, enabling immediate updates from IoT-enabled warehouses and quick dispatching within the fleet management module.  Target Response Time: Under 2 seconds for critical logistics operations and data retrieval. |
| *7* | ***Performance Management System*** | Advanced performance management tools designed for logistics, providing insights into warehouse operations, fleet efficiency, and user interactions to optimise system performance and resource allocation.  Monitoring Requirements: Capacity for real-time monitoring of up to 50,000 concurrent sessions during peak operational hours. |

# Appendixes

## Glossary

The list below contains all the necessary terms to interpret the document, including acronyms and abbreviations.

| **Term** | **Description** |
| --- | --- |
| *BR* | **B**usiness **R**ule |
| *CBR* | **C**ommon **B**usiness **R**ule |
| *DB* | Notes **D**ata**b**ase |
| *MSG* | **M**es**s**a**g**e |
| *UC* | **U**se **C**ase |
| *N/A* | **N**ot **A**vailable or **N**ot **A**pplicable, used to indicate when information in a certain section could not be provided because it does not apply to this application. |
| *UI* | **U**ser **I**nterface |
| *SRS* | **S**oftware **R**equirements **S**pecification |
| *TBD* | **T**o **b**e **d**etermined or **t**o **b**e **d**efined |

## **Messages**

This section describes the details of messages used in business rules e.g. error messages, confirmation messages, etc.

| **Message Code** | **Message Content** | **Button** |
| --- | --- | --- |
| MSG 1 | “Account Required” |  |
| MSG 2 | “Invalid Account” |  |
| MSG 3 | “Invalid Password” |  |
| MSG 4 | “No account matched” |  |
| MSG 5 | “Your password is wrong” |  |
| MSG 6 | “Missing required field(s)” |  |
| MSG 7 | “Partner already exists” |  |
| MSG 8 | “Partner created successfully” |  |
| MSG 9 | “Failed to create partner” |  |
| MSG 10 | “Missing required field(s)” |  |
| MSG 11 | “Partner already exists” |  |
| MSG 12 | “Partner updated successfully” |  |
| MSG 13 | “Failed to update partner” |  |
| MSG 14 | "Please handle all partner-related connections." |  |
| MSG 15 | "Partner deleted successfully." |  |
| MSG 16 | "Failed to delete partner." |  |
| MSG 17 | "Unable to load partner list. Please try again later." |  |
| MSG 18 | "Missing required field(s)." |  |
| MSG 19 | "Vehicle information already exists." |  |
| MSG 20 | "Vehicle added successfully." |  |
| MSG 21 | "Failed to add vehicle. Please try again later." |  |
| MSG 22 | “Missing required field(s).” |  |
| MSG 23 | “Invalid License Plate and prevents further actions.” |  |
| MSG 24 | “Vehicle already exists.” |  |
| MSG 25 | “Vehicle updated successfully.” |  |
| MSG 26 | “ Failed to update vehicle.” |  |
| MSG 27 | "Please handle all vehicle-related connections." |  |
| MSG 28 | "Vehicle deleted successfully." |  |
| MSG 29 | "Failed to delete vehicle." |  |
| MSG 30 | "Unable to load vehicle list. Please try again later." |  |
| MSG 31 | "Unable to load vehicle details. Please try again later." |  |
| MSG 32 | "Missing required field(s)." |  |
| MSG 33 | "Product already exists." |  |
| MSG 34 | "Product created successfully." |  |
| MSG 35 | "Failed to create product. Please try again later." |  |
| MSG 36 | "Missing required field(s).” |  |
| MSG 37 | "Require to choose at least one supplier." |  |
| MSG 38 | "Product already exists." |  |
| MSG 39 | “Missing required field(s).” |  |
| MSG 40 | “Product already exists and prevents saving.” |  |
| MSG 41 | “Product updated successfully.” |  |
| MSG 42 | “Failed to update product.” |  |
| MSG 43 | "Please handle all product-related connections." |  |
| MSG 44 | "Product deleted successfully." |  |
| MSG 45 | "Failed to delete product." |  |
| MSG 46 | "Unable to load product list. Please try again later." |  |
| MSG 47 | "Unable to load product details. Please try again later." |  |
| MSG 48 | "Failed to retrieve inventory data. Please try again later." |  |
| MSG 49 | "The inventory checklist is outdated." |  |
| MSG 50 | "Warning: Missing assets detected." |  |
| MSG 51 | "Failed to save inventory updates. Please try again later."v |  |
| MSG 52 | "Please select a date for the status report." |  |
| MSG 53 | "Data already exists for the selected date." The system prevents duplicate entries.” |  |
| MSG 54 | "All fields are mandatory. Please fill in the required details." |  |
| MSG 55 | "Status updated successfully." |  |
| MSG 56 | "Failed to update status. Please try again later." |  |
| MSG 57 | "Missing required field(s)." |  |
| MSG 58 | "Import order created successfully." |  |
| MSG 59 | "Failed to create import order. Please try again later." |  |
| MSG 60 | "Product created successfully." |  |
| MSG 61 | "Failed to create product. Please try again later." |  |
| MSG 62 | "Missing required field(s)." |  |
| MSG 63 | "Import order already exists." |  |
| MSG 64 | "Failed to create import order. Please try again later." |  |
| MSG 65 | “Missing required field(s).” |  |
| MSG 66 | “Import does not exist and prevents saving.” |  |
| MSG 67 | “Import updated successfully.” |  |
| MSG 68 | “Failed to update import order.” |  |
| MSG 69 | "Import order does not exist." |  |
| MSG 70 | "Import order deleted successfully." |  |
| MSG 71 | "Failed to delete import order." |  |
| MSG 72 | "Invalid document format. Please check the details and try again." |  |
| MSG 73 | "Quantity mismatch detected. Please review the order." |  |
| MSG 74 | "Import order confirmed successfully." |  |
| MSG 75 | "Unable to load import orders list. Please try again later." |  |
| MSG 76 | "The selected import order does not exist." |  |
| MSG 77 | "Import order updated successfully." |  |
| MSG 78 | "Failed to update import order. Please try again." |  |
| MSG 79 | "Unable to load product list. Please try again later." |  |
| MSG 80 | "Unable to load product details. Please try again later." |  |
| MSG 81 | "Missing required field(s)." |  |
| MSG 82 | "IDNumber must be 12 digits." |  |
| MSG 83 | "Driver information already exists." |  |
| MSG 84 | "Driver added successfully." |  |
| MSG 85 | "Failed to add driver. Please try again later." |  |
| MSG 86 | "Missing required field(s)." |  |
| MSG 87 | "IDNumber must be 12 digits." |  |
| MSG 88 | "Driver information already exists." |  |
| MSG 89 | "Failed to add driver. Please try again later." |  |
| MSG 90 | “Please handle all driver-related connections and prevents deletion.” |  |
| MSG 91 | “Driver deleted successfully.” |  |
| MSG 92 | “Failed to delete driver.” |  |
| MSG 93 | "Please handle all driver-related connections." |  |
| MSG 94 | "Driver deleted successfully." |  |
| MSG 95 | "Failed to delete driver." |  |
| MSG 96 | "Unable to load driver list. Please try again later." |  |
| MSG 97 | "Unable to load driver details. Please try again later." |  |
| MSG 98 | "Missing required field(s)." |  |
| MSG 99 | "Import order created successfully." |  |
| MSG 100 | "Failed to create import order. Please try again later." |  |
| MSG 101 | "Product created successfully." |  |
| MSG 102 | "Failed to create product. Please try again later." |  |
| MSG 103 | "Missing required field(s)." |  |
| MSG 104 | "Import order already exists." |  |
| MSG 105 | "Failed to create import order. Please try again later." |  |
| MSG 106 | "Unable to load import orders list. Please try again later." |  |
| MSG 107 | "The selected import order does not exist." |  |
| MSG 108 | "Import order updated successfully." |  |
| MSG 109 | "Failed to update import order. Please try again." |  |
| MSG 110 | "Invalid document format. Please check the details and try again." |  |
| MSG 111 | "Quantity mismatch detected. Please review the order." |  |
| MSG 112 | "Import order confirmed successfully." |  |
| MSG 113 | "Missing required field(s)." |  |
| MSG 114 | "Order already exists." |  |
| MSG 115 | "Order created successfully." |  |
| MSG 116 | "Failed to create order. Please try again later." |  |
| MSG 117 | "Invalid filter criteria. Please try again." |  |
| MSG 118 | "No orders found matching the specified criteria." |  |
| MSG 119 | "Failed to load orders. Please try again later." |  |
| MSG 120 | "Order cannot be confirmed as it is not in a 'Pending' status." |  |
| MSG 121 | "Invalid order details. Please review and correct the information." |  |
| MSG 122 | "Insufficient inventory to fulfill the order." |  |
| MSG 123 | "Order confirmed successfully." |  |
| MSG 124 | "Failed to confirm the order. Please try again later." |  |
| MSG 125 | "Missing required field(s)." |  |
| MSG 126 | "Driver is not available for the selected time period." |  |
| MSG 127 | "Shipment created successfully." |  |
| MSG 128 | "Failed to create shipment. Please try again later." |  |
| MSG 129 | "Invalid filter criteria. Please try again." |  |
| MSG 130 | "No shipments found matching the specified criteria." |  |
| MSG 131 | "Failed to load shipment data. Please try again later." |  |
| MSG 132 | "Missing required field(s). Please fill in all fields." |  |
| MSG 133 | "Invalid order(s) selected. Please review and update the shipment details." |  |
| MSG 134 | "Shipment updated successfully." |  |
| MSG 135 | "Failed to update shipment. Please try again later." |  |

## Issues List

N/A