# HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and communications technology

Software Requirement Specification

Version 1.2

<AIMS>

Subject: <ITSS>

<Group 19>

*Hanoi,* *March 2*

Table of Contents

[1 HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY i](#_Toc192504063)

[2 Introduction 2](#_Toc192504064)

[2.1 Objective 2](#_Toc192504065)

[2.2 Scope 2](#_Toc192504066)

[2.3 Glossary 2](#_Toc192504067)

[2.4 References 3](#_Toc192504068)

[3 Overall Description 4](#_Toc192504069)

[3.1 Survey 4](#_Toc192504070)

[3.2 Overall requirements 5](#_Toc192504071)

[3.3 Business process 5](#_Toc192504072)

[4 Detailed Requirements 7](#_Toc192504073)

[4.1 Use case 1 7](#_Toc192504074)

[4.2 Use case 2 10](#_Toc192504075)

[4.3 Use case 3 13](#_Toc192504076)

[4.4 Use case 4 17](#_Toc192504077)

[4.5 Use case 5 22](#_Toc192504078)

[4.6 Use case 6 26](#_Toc192504079)

[4.7 Use case 7 28](#_Toc192504080)

[5 Supplementary specification 32](#_Toc192504081)

[5.1 Functionality 32](#_Toc192504082)

[5.2 Usability 32](#_Toc192504083)

[5.3 Reliability 33](#_Toc192504084)

[5.4 Performance 34](#_Toc192504085)

[5.5 Supportability 35](#_Toc192504086)

[5.6 Other requirements 36](#_Toc192504087)

# Introduction

## Objective

The purpose of this Software Requirements Specification (SRS) is to define the functional and non-functional requirements for the AIMS (An Internet Media Store) project, a desktop e-commerce software. This document serves as a comprehensive guide for developers to design and implement the system, and for testers to verify its functionality. The intended audience includes the development team, testing team, project stakeholders, and instructors of the Software Engineering course at Hanoi University of Science and Technology.

## Scope

The AIMS software facilitates the buying and selling of physical media products (books, CDs, LP records, DVDs) through a desktop application. It allows customers to browse, search, and purchase products without requiring login, while Product Managers manage inventory and Administrators oversee user accounts. Key features include product management, order placement (including rush delivery), payment processing via VNPay, and order cancellation with refunds. The system will not support digital media sales or mobile applications within this scope. Benefits include streamlined e-commerce operations, 24/7 availability, and support for up to 1,000 simultaneous users.

## Glossary

| ***No*** | ***Term*** | ***Explanation*** | ***Example*** | ***Note*** |
| --- | --- | --- | --- | --- |
| 1 | token | A piece of data created by server, and contains the user's information, as well as a special token code that user can pass to the server with every method that supports authentication, instead of passing a username and password directly. | JSON Web Token (JWT) | Compact, URL-safe and usable especially in web browser single sign-on (SSO) context. |
| 2 | VNPay | An external payment gateway for processing credit card transactions and refunds. | VNPay Sandbox | Integrated via API. |
| 3 | Product Manager | A user role responsible for managing product inventory and order approval. | N/A | Requires login. |
| 4 | Rush Order | A delivery option for eligible items within 2 hours in Hanoi inner districts. | N/A | Additional fee applied. |
| 5 | Pending Processing | Order status awaiting Product Manager approval or rejection. | N/A | Precedes fulfillment. |

## References

* AIMS Project Document by Prof. NGUYEN Thi Thu Trang ([trangntt@soict.hust.edu.vn](mailto:trangntt@soict.hust.edu.vn)).
* VNPay Sandbox Demo: <https://sandbox.vnpayment.vn/apis/vnpay-demo/>
* VNPay Payment API: <https://sandbox.vnpayment.vn/apis/docs/thanh-toanpay/pay.html>
* VNPay Refund API: <https://sandbox.vnpayment.vn/apis/docs/truy-van-hoan-tien/querydr&refund.html>

# Overall Description

## Survey

The AIMS (An Internet Media Store) project is a desktop e-commerce software designed to facilitate the buying and selling of physical media products such as books, CDs, LP records, and DVDs. The software operates 24/7, supports up to 1,000 simultaneous customers with minimal performance degradation, and ensures a maximum response time of 2 seconds under normal conditions or 5 seconds during peak hours. It is resilient, capable of running for 300 hours without failure, and can recover within 1 hour after an incident.

**Key actors**

**Customer** : *Interacts with the system to browse, search, add products to the cart, and pay for orders using VNPay. Customers do not need to log in to place an order within the scope of this course.*

**VNPay** : *An external payment gateway integrated with AIMS to process credit card payments and refunds.*

**Product Manager** : *Manages product inventory, reviews, and approves or rejects orders in the pending processing state.*

**Administrator** : *Manages user accounts, roles, and permissions (not directly involved in this usecase).*

## Overall requirements

A diagram of a diagram

AI-generated content may be incorrect.

## Business process

**Business Process for AIMS**:

1. Customer browses/searches products and views details.
2. Customer adds products to cart and places an order (regular or rush).
3. System checks inventory and delivery eligibility.
4. Customer pays via VNPay.
5. Post-payment: Order is created, cart is emptied, and confirmation is sent.
6. Customer may cancel order (pre-approval) with refund processed via VNPay.
7. Product Manager adds/updates/deletes products and approves orders.

*Activity Diagram*

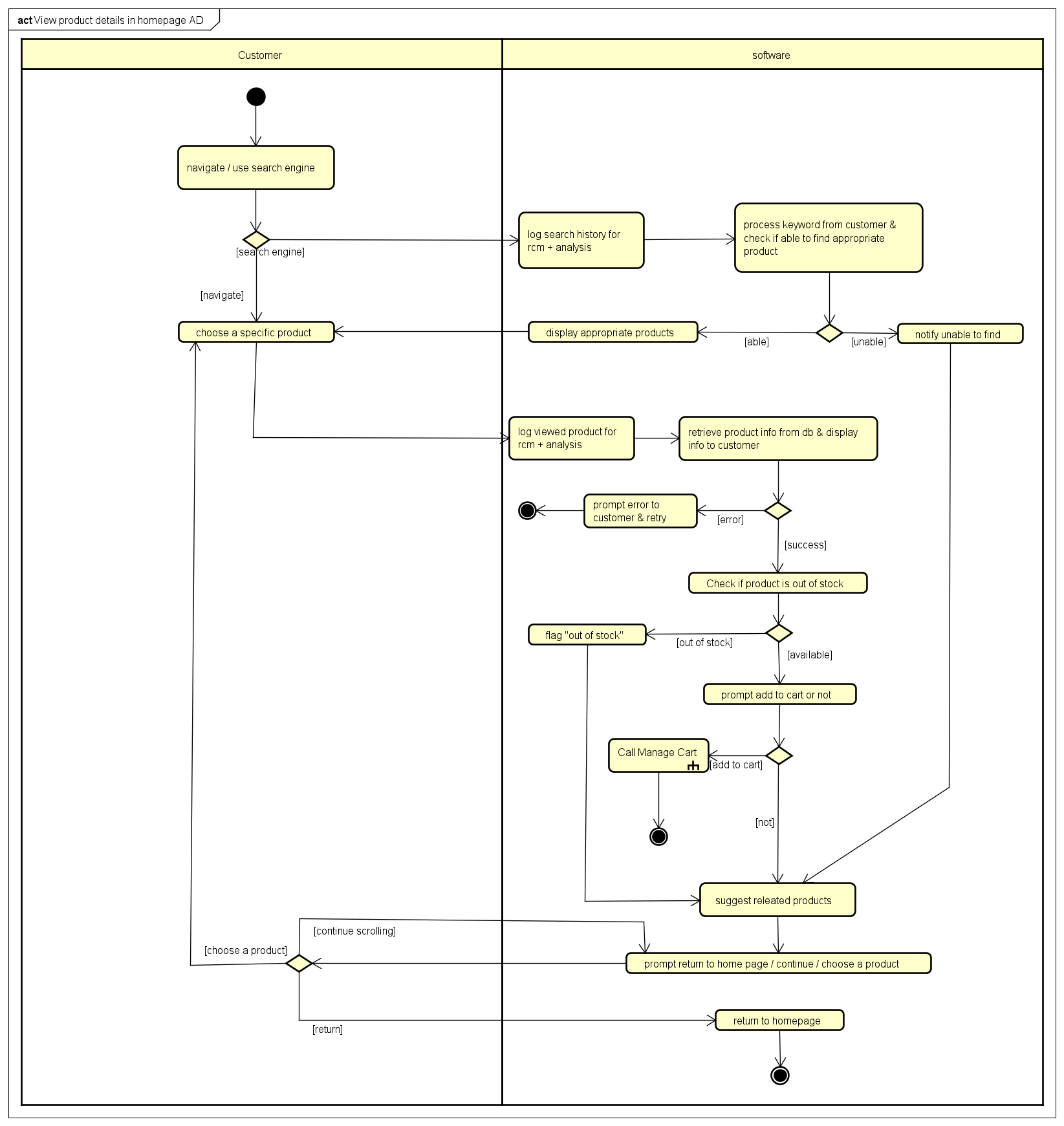
A diagram of a software project

AI-generated content may be incorrect.

# Detailed Requirements

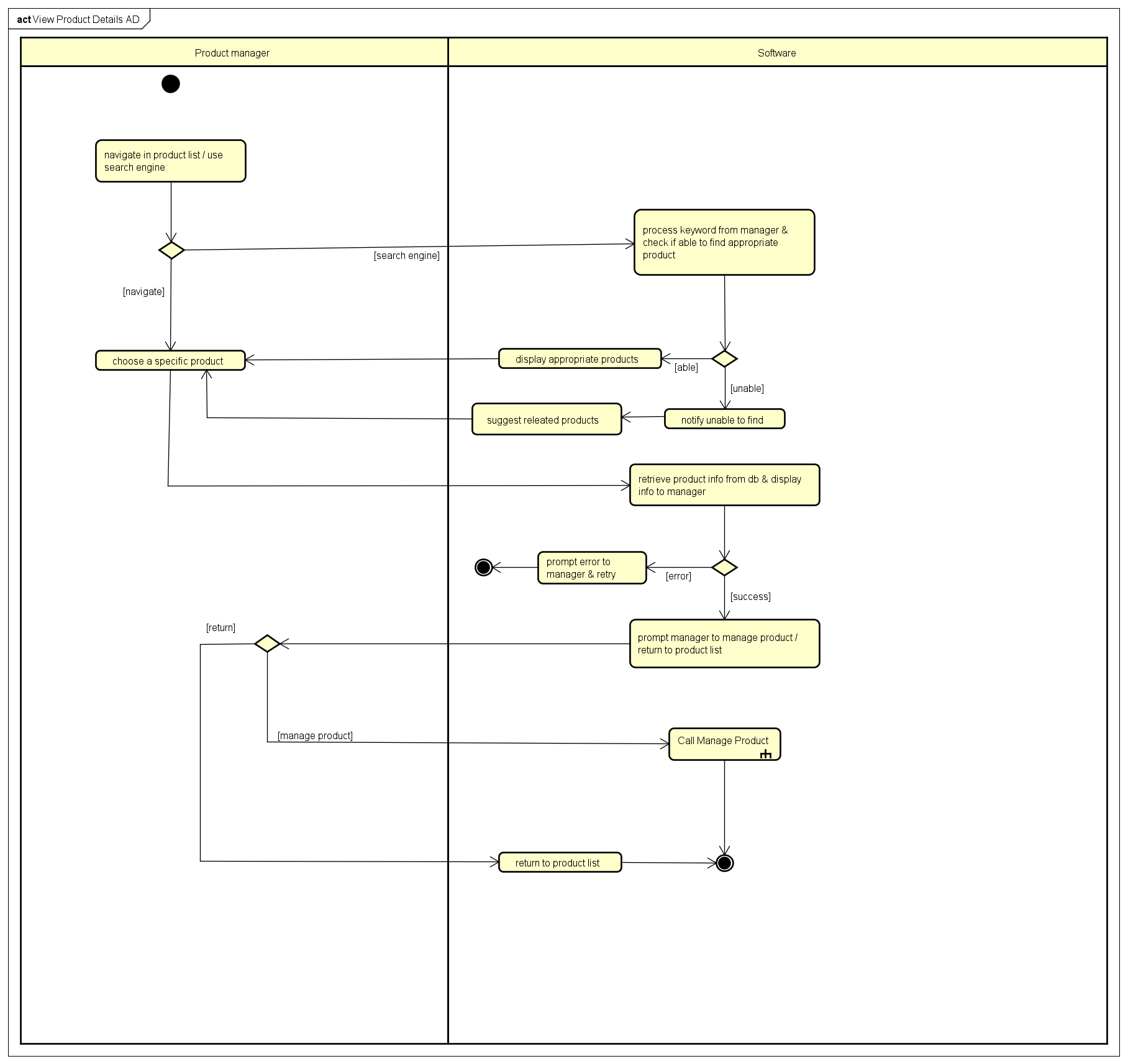
## Use case 1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| USE CASE “View Product Detail in homepage” ( Customer )   1. **Use case code:**   UC001   1. **Brief description**   This use case describes the interaction between a Customer and AIMS when the Customer wishes to view detailed information about a specific product to make a purchase.   1. **Actors**   Customer: the one who is browsing products in AIMS   1. **Preconditions** 2. The customer must have access to the system (login is optional for browsing products). 3. The system must have product details stored in the database. 4. **Basic flow** Step 1. The Customer navigates through the homepage or uses the search function.   **Search Engine Path**  Step 2a: Software logs search history for recommendation and analysis  Step 3a: Software process keyword from customer and check if able to find appropriate products  Step 4a: Software displays appropriate products  **Navigate Path**  Customer navigates through the homepage or categories.  Step 2b: The Customer selects a specific product to view details.  Step 3b: The Software logs the viewed product for recommendations and analysis.  Step 4b: The Software retrieves product information from the database.  Step 5: The Software displays detailed product information, including name, description, price, availability, specifications, and images.  Step 6: The Software checks if the product is available  Step 7: Software prompt customer to Add the product to the cart  Step 8. The system suggests related products  Step 9. The Customer can choose to choose a product or continue scrolling or return to homepage   1. **Alternative flow**  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | **1** | **Step 3a** | If the product does not exist or is no longer available | The system displays a message indicating that the product is unavailable and suggests alternatives or related product. | Resumes at Step 8 | | **2** | **Step 4b** | If the system fails to retrieve product data due to an error | The system displays an error message and prompts the Customer to retry. | Ends if unresolved | | **3** | **Step 6** | Product is out of stock | System flags "Out of stock" and suggests alternatives | Continues at Step 8 | | **4** | **Step 7** | Customer choose to manage Cart | Software Call Manage Cart | Ends | | **5** | **Step 9** | Customer choose to return to homepage | Software Returns to homepage | ends | | **6** | **Step 9** | Customer choose to choose a product | Return to choose a specific product screen | Resume at step 2b |  1. **Input data for deliveries**   **Table A - Input Data of View Product Detail (Customer)**   | **No** | **Data Fields** | **Description** | **Mandatory** | **Valid Condition** | **Example** | | --- | --- | --- | --- | --- | --- | | 1 | Product ID | Unique identifier for the product | Yes | Alphanumeric, exists in database | "BK001" |  1. **Output data**  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | 1 | Product Name | Name of the product | Text | "Harry Potter Book" | | 2 | Description | Product description | Text | "New, hardcover" | | 3 | Price | Selling price including VAT | Number (VND) | 165,000 VND | | 4 | Availability | Stock status | Text | "In stock" | | 5 | Specifications | Product-specific details | List | "Author: J.K. Rowling, Pages: 500" | | 6 | Images | Product images | Image | [Displayed Image] | | 7 | Related Products | Suggested alternative products | List | "Lord of the Rings, Hobbit" |  1. **Postconditions**    1. The Customer successfully views the detailed information of the selected product.    2. If the product is unavailable, the system suggests alternative options.    3. The system logs the Customer's interaction for recommendation and analytics purposes.    4. The Customer can add the product to the cart, proceed to checkout, or browse other products.    5. The system ensures real-time synchronization of product availability and pricing data.    6. The system maintains a log of viewed products to enhance personalized recommendations in future browsing sessions. |



## Use case 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| USE CASE “View Product Detail”  ( Product Manager )   1. **Use case code:**   UC002   1. **Brief description**   This use case describes the interaction between the Product Manager and AIMS when the Product Manager wishes to view detailed information about a specific product for management purposes (e.g., updating or deleting).   1. **Actors**   Product manager: the one who responsible for managing products inventory in AIMS   1. **Preconditions**   The Product Manager is logged into the system with appropriate permissions.The system has product details stored in the database.   1. **Basic flow**   Step 1: The Product Manager navigates through the product list or uses the search function.  **Search Engine Path**  Step 2a: Software process keyword from manager and check if able to find appropriate products  Step 3a: Software displays appropriate products  **Navigate Path**  The Product Manager navigates through the product list  Step 2b: The Product Manager selects a specific product to view details.  Step 3b: The system retrieves product information from the database.  Step 4: The system displays detailed product information, including name, description, price, value, barcode, stock quantity, warehouse entry date, and edit history (see Table B).  Step 5: Software prompts The Product Manager choose to manage product, or return to the product list.   1. **Alternative flow**  | **No** | **Location** | **Condition** | **Action** | **Resume Location** | | --- | --- | --- | --- | --- | | 1 | At Step 2a | Product does not exist | System displays "Product not found" and suggest related products | Resumes at Step 2b | | 2 | At Step 3b | System fails to retrieve data | System displays "Error, please retry" | Ends if unresolved | | 3 | At Step 5 | Manager chooses to return to product list | Software Return to product list | Ends | | 4 | At step 5 | Manager chooses to manage product | Software calls to Manage Product UC | Ends |  1. **Input data for deliveries**  | **No** | **Data Fields** | **Description** | **Mandatory** | **Valid Condition** | **Example** | | --- | --- | --- | --- | --- | --- | | 1 | Product ID | Unique product identifier | Yes | Alphanumeric, exists in DB | "BK001" |  1. **Output data**  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | 1 | Product Name | Name of the product | Text | "Harry Potter Book" | | 2 | Description | Product description | Text | "New, hardcover" | | 3 | Price | Selling price (excl. VAT) | Number (VND) | 150,000 VND | | 4 | Value | Base value (excl. VAT) | Number (VND) | 135,000 VND | | 5 | Barcode | Unique identifier | Alphanumeric | "123456789012" | | 6 | Stock Quantity | Available stock | Number | 50 | | 7 | Warehouse Entry Date | Date added to inventory | DD/MM/YYYY | "01/01/2025" | | 8 | Edit History | Log of changes | List | "Price updated 01/02/2025" | | 9 | Specifications | Product-specific details | List | "Author: J.K. Rowling" |  1. **Postconditions**    1. The Product Manager successfully views the detailed information of the selected product.    2. The system logs the Product Manager’s interaction for audit purposes.    3. The Product Manager can proceed to update or delete the product, or return to the product list.    4. Product data remains synchronized with the database. |



## Use case 3

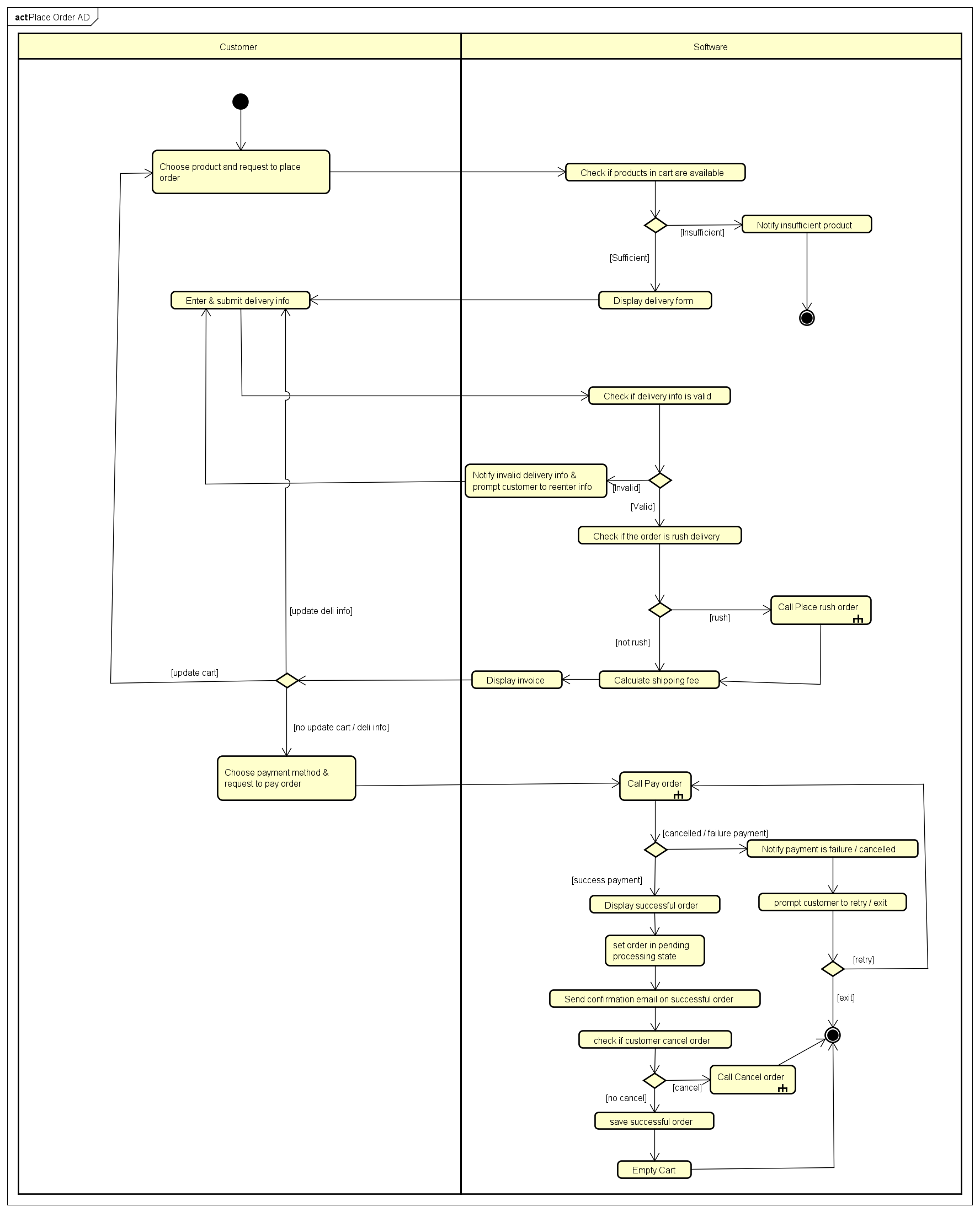
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **USE CASE “Manage Product”**  **1. Use case code:**  UC003  **2. Brief description**  This use case describes the interaction between the Product Manager and the system when the Product Manager wishes to add, update or delete a product.  **3. Actors**  Product Manager  **4. Preconditions**  The Product Manager must be logged into the system with the appropriate permissions.  **5. Basic flow**  **Step 1:** Product Manager selects "Manage Product" from the system menu.   **Add Product Path**  **Step 2a:** System prompts Product Manager to enter product information.  **Step 3a:** Product Manager enters product details.  **Step 4a:** System validates the information and confirms the product has been added successfully.  **Step 5a:** System updates the database and product catalog.  **Step 6a:** System log operation.  **Step 7a:** System prompts Product Manager to continue or exit.   **Update Product Path**  **Step 2b:** Product Manager selects a product to edit and updates the information.  **Step 3b:** System validates the updated information  **Step 4b:** System confirms the product has been updated successfully.  **Step 5b:** System updates the database and product catalog.  **Step 6b:** System log operation.  **Step 7b:** System prompts Product Manager to continue or exit.   **Delete Product Path**  **Step 2c:** Product Manager selects a product to delete.  **Step 3c:** System checks if the deletion is within allowed limits.  **Step 4c:** System checks if the deletion affects any active orders.  **Step 5c:** System confirms the product has been deleted successfully  **Step 6c:** System updates the database and product catalog.  **Step 7c:** System log operation.  **Step 8c:** System prompts Product Manager to continue or exit.  **6. Alternative flow**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | |  | |  | | --- | | Step 3a |  |  | | --- | |  | | If mandatory product fields are missing | The system displays an error message prompting the Product Manager to complete the missing fields. | Resumes at Step 2a | |  | Step 4a | If the entered data is invalid (e.g., price format incorrect) | The system highlights invalid fields and prompts for correction. | Resumes at Step 3a | |  | Step 5a | If the product fails to save due to system error | The system displays an error message and prompts for retry. | Resumes at Step 4a or ends if unrecoverable | |  | Step 3c | If exceeds deletion limit | The system aborts the deletion process. | Resumes at step 8c | |  | Step 4c | If the Product Manager cancels the deletion | The system displays an error message and prompts for continue or exit. | Resumes at step 8c | |  | Step 5c | If the product fails to delete due to system error | The system displays an error message and prompts for continue or exit. | Resumes at Step 8c | |  | Step 4c | If the product deletion affects existing orders or inventory | The system warns the Product Manager of dependencies and asks for confirmation before proceeding. | Resumes at Step 5c |   **7. Input Data of Manage Product**   | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | | --- | --- | --- | --- | --- | --- | | 1 | ProductName | Name of the product | Yes | string (100) | Harry Potter Book | | 2 | Description | Short description of the product | Yes | string (500) | New, hardcover | | 3 | Price | The price of the product | Yes | int | 150000 | | 4 | Value | Product base value | Yes | int | 135000 | | 5 | StockQuantity | |  | | --- | | Quantity in stock |  |  | | --- | |  | | Yes | int | 50 | | 6 | Category | |  | | --- | | Product category |  |  | | --- | |  | | Yes | enum | Book | | 7 | Barcode | |  | | --- | | Product barcode |  |  | | --- | |  | | Yes | string (13) | 1234567890123 | | 8 | Weight | |  | | --- | | Product weight |  |  | | --- | |  | | Yes | float | 0.5 | | 9 | Dimensions | |  | | --- | | Product dimensions |  |  | | --- | |  | | Yes | JSON | |  | | --- | | {"L": 20, "W": 15, "H": 5} |  |  | | --- | |  | | | 10 | AuthorArtist | |  | | --- | | Author/Artist |  |  | | --- | |  | | No | string (100) | |  | | --- | | J.K. Rowling |  |  | | --- | |  | | | 11 | PublisherLabel | |  | | --- | | Publisher/Label |  |  | | --- | |  | | No | string (100) | Bloomsbury | | 12 | PublicationDate | |  | | --- | | Publication date |  |  | | --- | |  | | No | date | 1997-01-01 | | 13 | ProductName | Name of the product | Yes | string (100) | Harry Potter Book |   **8. Output data**   | **No** | **Data fields** | **Description** | **Display format** | **Example** | | --- | --- | --- | --- | --- | | 1 | ProductID | |  | | --- | | Product ID |  |  | | --- | |  | | int | 101 | | 2 | ProductName | |  | | --- | | Product name |  |  | | --- | |  | | string (100) | Harry Potter Book | | 3 | Value | |  | | --- | | Product base value |  |  | | --- | |  | | int | 135000 | | 4 | Price | |  | | --- | | Product selling price |  |  | | --- | |  | | int | 150000 | | 5 | StockQuantity | |  | | --- | | Quantity in stock |  |  | | --- | |  | | int | 50 | | 6 | Category | |  | | --- | | Product category |  |  | | --- | |  | | enum | Book | | 7 | Barcode | |  | | --- | | Product barcode |  |  | | --- | |  | | string (13) | 1234567890123 | | 8 | Weight | |  | | --- | | Product weight |  |  | | --- | |  | | float | 0.5 | | 9 | Specifications | |  | | --- | | Product specifications |  |  | | --- | |  | | JSON | {"Author": "J.K. Rowling"} | | 10 | DateAdded | |  | | --- | | Date added to system |  |  | | --- | |  | | date | 2025-01-01 | | 11 | LastUpdated | |  | | --- | | Last update date |  |  | | --- | |  | | datetime | 2025-02-01 10:00:00 | | 12 | Status | |  | | --- | | Product status |  |  | | --- | |  | | enum | Active |   **9. Postconditions**   1. If a new product is added, it is successfully saved in the system's database. 2. If a product is updated, the updated product information is successfully saved in the system's database. 3. If a product is deleted, it is successfully removed from the system's database. 4. The product catalog is updated to reflect the addition, update, or deletion made by the Product Manager. 5. The updated or newly added product is made available for viewing and searching by other users (e.g., customers, other managers). 6. If the product was added, it appears as a new entry in the product list with the status "Active." 7. If the product was updated, all associated data (e.g., orders, inventory) are synchronized to reflect the new product details. 8. If the product was deleted, it is removed from the product list, and any related inventory or orders are updated accordingly. 9. A notification or confirmation message is displayed to the Product Manager confirming the success of the add, update, or delete operation. 10. Any inventory adjustments (if applicable) are reflected in the stock levels of the product. 11. The system logs the transaction for auditing and tracking purposes, including details such as who made the changes, the time of the update, deletion, or addition, and what was changed. |

A diagram of a flowchart

AI-generated content may be incorrect.

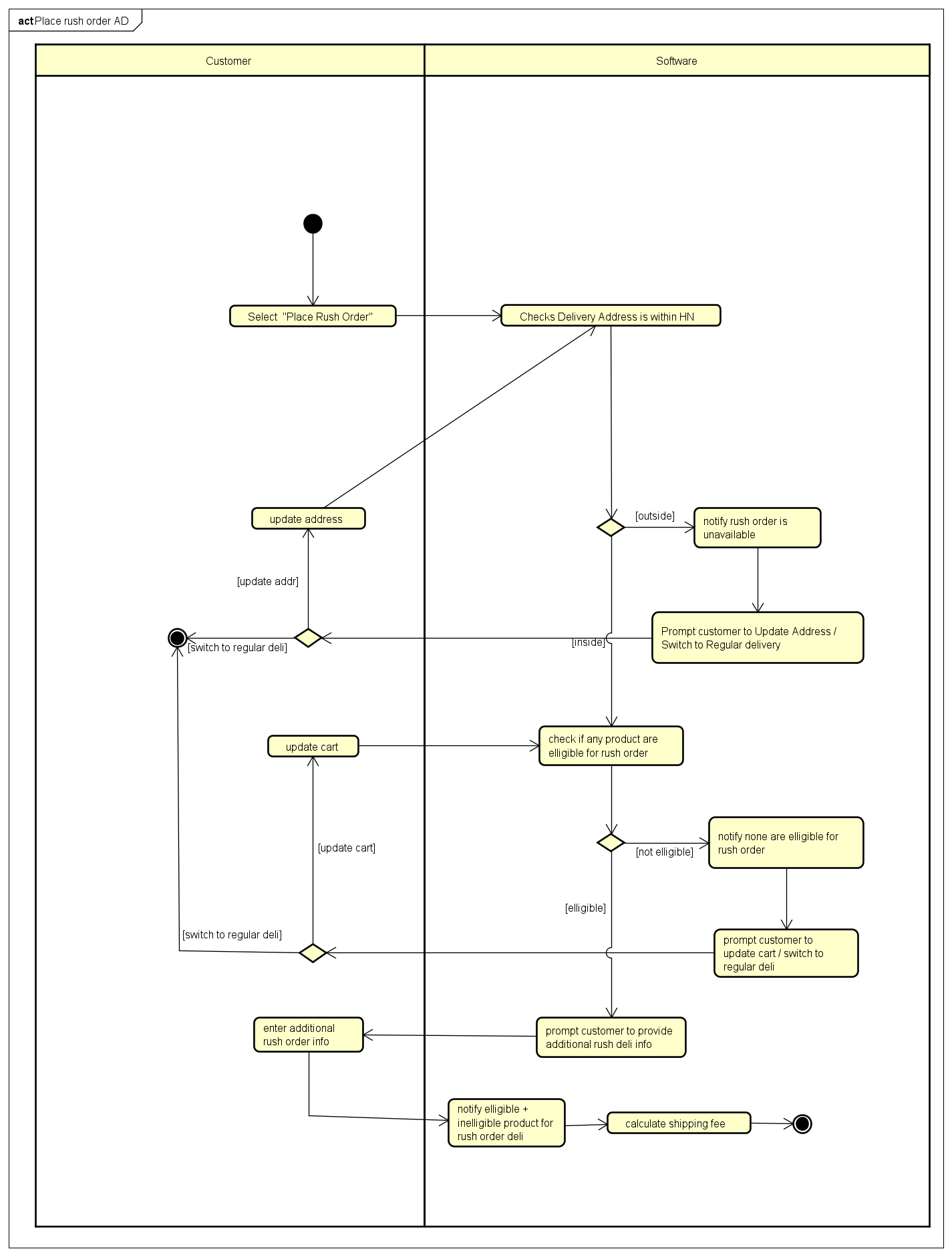
## Use case 4

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| USE CASE “PLACE ORDER”  **:**   1. **Use case code**   UC004   1. **Brief description**  * This use case describes the interaction bettween the “customer” and the AIMS software when the customer wants to place an order  1. **Actors**  * Customer * VNPay  1. **Preconditions**  * Customer has products in their cart * Customer is on the cart view screen  1. **Basic flow**   Step 1. Customer requests to place order in the cart  Step 2. AIMS software checks the availability of products in the cart  Step 3. AIMS software displays the form of delivery information with order information  Step 4. Customer enters and submits delivery information  Step 5. AIMS software calculates and updates order information with shipping fees  Step 6. The customer asks to pay order  Step 7. The AIMS software calls UC “Pay order”  Step 8: The AIMS software display successful order  Step 9: The AIMS software set order in pending processing state  Step 10. The AIMS software sends email about the order notification and information  Step 11: The AIMS software save order  Step 12. The AIMS empty the cart   1. **Alternative flow**  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 3 | If the products are not available | * The AIMS software notifies that the the products in the cart are not available and stay at the use case “View cart” | Use case ends | | 2. | At Step 5 | If the delivery info is invalid | * AIMS software notifies that the delivery info is invalid (blank or wrong format) | At Step 3 | | 3. | At Step 5 | If the user chooses to place a rush order | * AIMS software inserts use case “Place rush order” | At Step 6 | | 4. | At Step 8 | If the order payment is not successul or goes back from payment | * AIMS software notifies the customer that payment failed and go back to step 5 | At Step 5 |  1. **Input Data of Place Order**  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | | 1 | Recipient Name | Name of the delivery recipient | Yes | * String * Max 50 chars | Bui Minh Tuan | | 2 | Phone Number | Contact phone number | Yes | 10 digits, numeric | 0987654321 | | 3 | Email | Contact email for notifications | Yes | Valid email format | tuanbm@example.com | | 4 | Province/City | Delivery province or city | Yes | Valid Vietnamese province/city | Hanoi | | 5 | Address | Detailed delivery address | Yes | * String * Max 200 chars | 123 Dai La, Truong Dinh, Hai Ba Trung | | 6. | Shipping Instructions | Instruction by the customer for the shipper | No | * String * Max 100 chars | Deliver to the house with the red door |  1. **Output data**    1. Customer information  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | | 1 | CustomerID | ID of customer, auto-generated | * int | 1 | | 2. | FullName | Name of the delivery recipient | * String | "Bui Minh Tuan" | | 3. | Phone | Contact phone number | * String | "0987654321" | | 4. | Email | Contact email | * String | "[tuanbm@example.com](mailto:tuanbm@example.com)" | | 5. | ProvinceCity | Delivery province or city | String | "Hanoi" | | 6. | Address | Detailed delivery address | * String | "123 Dai La, Truong Dinh, Hai Ba Trung" |  * 1. Order infomation  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | | 1 | Order ID | The order unique identifier | * String * Max 30 char | OD00001 | | 2. | Product Title | Title of a media product | * String * Max 100 chars | DVD Rambo | | 3. | Unit Price(include VAT) | Price of the corresponding media product | * Comma for thousands separator * Positive integer | 105,000 | | 4. | Quantity | Quantity of the corresponding media | * Positive integer | 2 | | 5. | Line Total(include VAT) | Total money of the corresponding media | * Comma for thousands separator * Positive integer | 210,000 | | 6. | Subtotal | Total amount of all products in the order | 2,240,000 | | 7. | Shipping fee | Cost of delivery | 100,000 | | 8. | Total | Final amount to be paid | 2,340,000 |  * 1. Transaction info  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | **1** | Transaction ID | Unique ID of the payment | Alphanumeric | "TXN123456789" | | 2 | OrderID |  | String | "ORD001" | | 3 | Total Amount | Final amount paid | Number (VND), right-aligned | "2,500,000 VND" | | 4 | Transaction Content | Description of the transaction | String | "Order #ORD001 Payment" | | 5 | Transaction Date | Date and time of payment | DD/MM/YYYY HH:MM:SS | "23/02/2025 14:30:00" | | 6 | Payment Method | Method used for payment | String | "Credit Card" | | 7 | Payment Status | Status of the transaction | String | "Completed" | | 8 | Refund Status | Status of refund (if applicable) | String | "Not Refunded" | | 9 | Refund Date | Date of refund (if applicable) | DD/MM/YYYY HH:MM:SS | "N/A" |  1. **Postconditions**  * Order is recorded in system with "pending processing" status * Order awaits approval or rejection by a product manager |



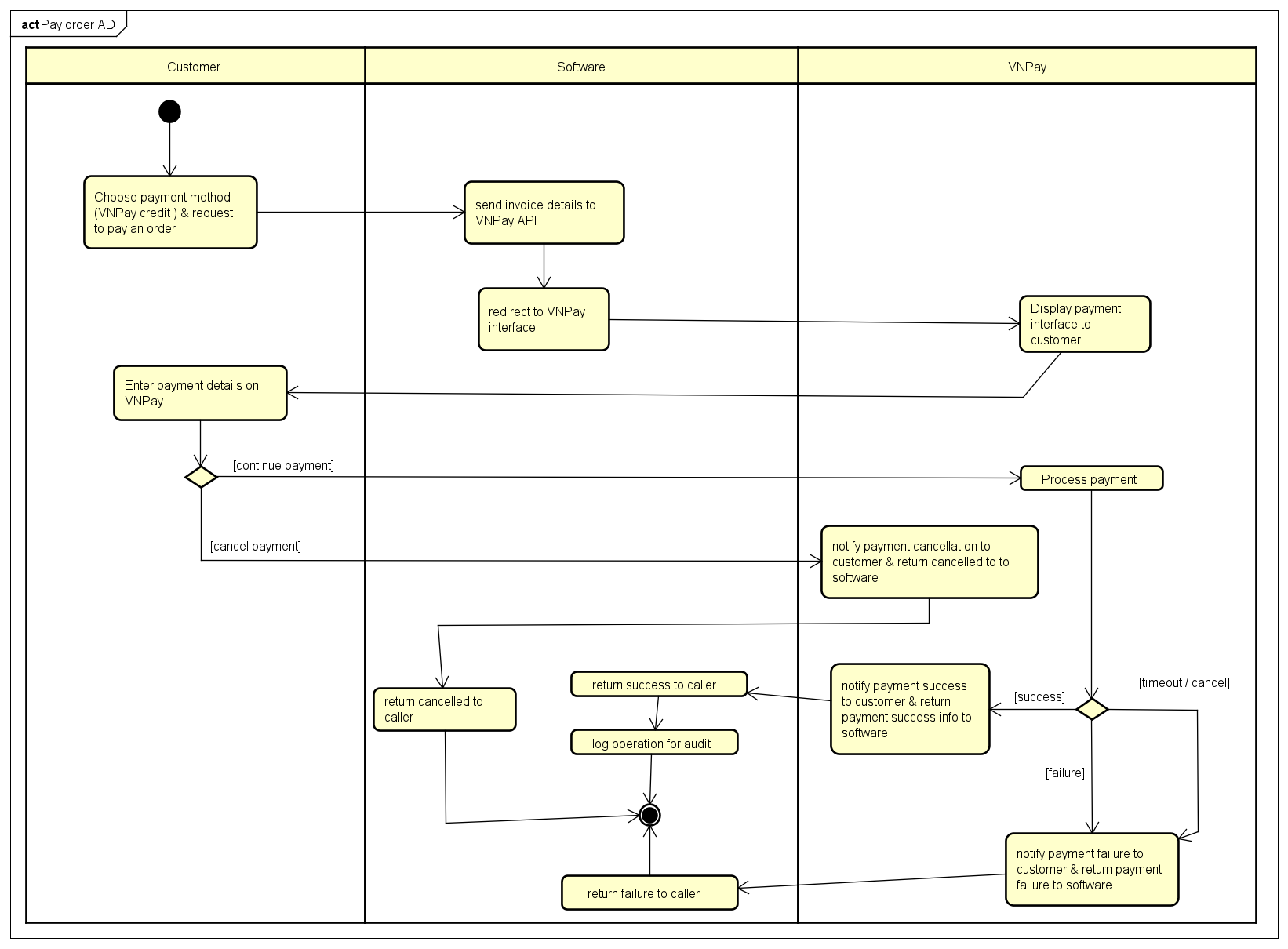
## Use case 5

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Place Rush Order”**   1. **Use case code**   UC005   1. **Brief Description**  * This use case describes the process by which a customer places an order with the option for rush order delivery, allowing eligible items to be delivered within a 2-hour prearranged timeframe to addresses within the inner city of Hanoi districts. The system verifies eligibility, calculates fees, and prompts the customer for additional information as needed.  1. **Actors**  * **Customers**  1. **Preconditions**  * The customer is logged into the system. * The customer has added at least one item to their shopping cart. * The system has access to the customer's delivery address and product eligibility data.  1. **Basic Flow of Events** 2. The customer initiates the checkout process and selects the "Rush Order Delivery" option. 3. The system checks if the delivery address is within the inner city of Hanoi districts. 4. The system verifies whether any products in the cart are eligible for rush order delivery. 5. If both the address and at least one product are eligible:    1. The system prompts the customer to provide additional rush order delivery information (e.g., delivery time within the next 2 hours, delivery instructions).    2. The system informs the customer:       1. Which items are eligible for rush order delivery and will be delivered together within 2 hours.       2. Which items (if any) are ineligible and will follow regular delivery.       3. Delivery fees for rush order items (including an additional 10,000 VND per rush order item) and regular delivery items (if applicable), calculated based on weight and location. 6. The customer submits the rush order delivery information. 7. The system calculates total fees:    1. Rush order delivery fees: Initial fee (e.g., 22,000 VND for the first 3kg in Hanoi) + 2,500 VND per additional 0.5kg + 10,000 VND per rush order item.    2. Regular delivery fees (if applicable): Based on location and weight, with free shipping up to 25,000 VND for orders over 100,000 VND (excluding rush order items). 8. The system displays the updated invoice with separate delivery fees and total cost.  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1 | At Step 2 | If the delivery address is outside the inner city of Hanoi districts | - The system notifies the customer that rush order delivery is unavailable for their address.  - The system prompts the customer to update the delivery address or switch to regular delivery.  - If the customer updates the address and it becomes eligible, return to step 3.  - If the customer switches to regular delivery, the system proceeds with standard checkout. | Resumes at Step 3 | | 2 | At Step 3 | If no products in the cart are eligible for rush order delivery | - The system notifies the customer that none of their items qualify for rush order delivery.  - The system prompts the customer to adjust their cart (e.g., add eligible items) or switch to regular delivery  - If the customer adjusts the cart and adds eligible items, return to step3.  - If the customer switches to regular delivery, the system proceeds with standard checkout. | Resume at Step 3 | | 3 | At Step 7 | If the customer modifies the delivery method or items in the cart | - The system recalculates eligibility and fees based on the updated selections.  - The system redisplays the invoice with updated costs.  - Return to step 8 foìnoconfirmation. | Resume at Step 8 |  1. **Input data**  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data Fields** | **Description** | **Mandatory** | **Valid Condition** | **Example** | | 1 | Delivery Method | Customer's choice of delivery method | Yes | Enum:  Must be “Rush Order" or "Regular" | "Rush Order" | | 2 | Delivery Address | Detailed address for delivery | Yes | string (200) | "97 Dai Co Viet, Bach Khoa, Hanoi" | | 3 | Cart Item List | List of product IDs and quantities to purchase | Yes | JSON: Item, Time, Address | "BK001 (Qty: 2), CD002 (Qty: 1)" | | 4 | Rush Order Delivery Time | Specific time for rush delivery within 2 hours | Yes (if rush) | Datetime  DD/MM/YYHH:MM:SS | "14:00 PM, 27/02/2025" | | 5 | Delivery Instructions | Additional notes for delivery | No | string (100) | "Leave at gate" |  1. **Output data**  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | 1 | Order Confirmation | Message confirming order placement | String | "Your order #ORD123 has been placed successfully" | | 2 | Rush Order Delivery Details | Details of rush items, time, and address | JSON: Item, Time, Address | {"Item": "Cream Cake", "Time": "2025-02-27T14:00:00", "Address": "97 Dai Co Viet"} | | 3 | Regular Delivery Details | Details of regular items and estimated delivery | JSON: Item, Time, Address | {"Item": "Gaming Desk", "EstDelivery": "2025-02-28"} | | 4 | Rush Order Delivery Fee | Fee for rush items (base + surcharge) | Int | "32,000 VND" | | 5 | Regular Delivery Fee | Fee for regular items or free shipping note | int | "0 VND (Free shipping applied)" | | 6 | Invoice Breakdown | Detailed breakdown of costs | JSON: Table: Category, Amount | {"Items": 150000, "RushFee": 32000, "RegularFee": 0, "Total": 182000} | | 7 | Total Order Cost | Total cost including items and fees | Int | "182,000 VND" |  1. **Postconditions:**  * The order is successfully placed with rush order delivery for eligible items (if applicable) and regular delivery for ineligible items (if any). * The customer receives a confirmation with delivery times and fees. * The system updates the order status and notifies relevant parties (e.g., delivery team). |



## Use case 6

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **USE CASE “Pay order”**   1. **Use Case Code** UC006 2. **Brief Description** This use case describes the interaction between the **Customer** and **AIMS** when the customer wishes to pay for an order using the VNPay payment gateway. The process involves selecting VNPay, processing the payment, and returning the result (success or failure) to the calling usecase "Place Order" for further handling. 3. **Actors**    * **Customer**: The individual purchasing products from AIMS.    * **VNPay**: The external payment gateway that processes credit card payments. 4. **Preconditions**    * The customer has completed the "Place Order" steps up to payment request.    * The system has verified sufficient inventory for all products in the cart.    * Delivery information is valid, and shipping fees (regular and/or rush) are calculated.    * The invoice details (products, total amount including VAT and fees) are ready to display. 5. **Basic Flow of Events** 6. The customer selects VNPay credit payment method and request to pay 7. Software send invoice details to VNPay API 8. VNPay display VNPay payment interface to customer. 9. The customer enters payment details as required by VNPay (e.g., card number, expiration date, CVV). 10. VNPay processes the payment and returns a success or failure response to AIMS. 11. The software notifies customer successful payment and return payment success result to the "Place Order" usecase 12. Software return success to caller and log operation for audit 13. **Alternative Flows**  | **No** | **Location** | **Condition** | **Action** | **Resume Location** | | --- | --- | --- | --- | --- | | 1 | At step 4 | Customer chooses to cancel payment | VNPay notify payment cancellation to customer & return cancelled to software  Software return cancelled to caller | Use case ends | | 1 | At Step 6 | Payment fails / time out (e.g., invalid card) | VNPay returns failure to Software and software return failure to caller | Use case ends |  1. **Input Data**  | **No** | **Data Fields** | **Description** | **Mandatory** | **Valid Condition** | **Example** | | --- | --- | --- | --- | --- | --- | | 1 | Payment Method | Selected payment gateway | Yes | Enum: Must be "VNPay" (only option) | "VNPay" |  1. **Output Data**  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | 1 | Invoice Details | Products, quantities, prices (incl. VAT), fees, total | JSON: Item, Qty, Price (VND) | [{"Item": "Book A", "Qty": 2, "Price": 55000}] | | 2 | Total Amount | Total incl. VAT and delivery fee | Int (VND), comma-separated | "125,000 VND" | | 3 | Transaction ID | Unique ID from VNPay (if success) | String | "TXN123456789" | | 4 | Transaction Content | Description of the payment (if success) | String | "Payment for Order #ORD001" | | 5 | Transaction Datetime | Date and time of payment (if success) | Datetime : DD/MM/YYYY HH:MM:SS | "27/02/2025 14:30:00" | | 6 | Payment Status | Result returned to "Place Order" | Enum | "Success" or "Failure" |  1. **Postconditions**    * If payment is successful:      + The software returns a success status with transaction details to "Place Order."      + The calling usecase proceeds to create/save the order, empty the cart, send a confirmation email, and display a success message.    * If payment fails:      + The software returns a failure status to "Place Order."      + The cart remains intact, and no order is created.    * The "Pay Order" usecase ends, and control returns to "Place Order." |



## Use case 7

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **USE CASE “Cancel Order”**   1. **Use Case Code** UC007 2. **Brief Description** This use case describes the interaction between the Customer and AIMS when the customer wishes to cancel an order that has been successfully paid using the VNPay payment gateway. The process involves verifying the order status, requesting a refund via VNPay, and updating the order status, with the result returned to the customer. 3. **Actors**    * **Customer**: The individual who placed and now wishes to cancel the order.    * **VNPay**: The external payment gateway that processes the refund for the cancelled order. 4. **Preconditions**    * The customer has successfully placed an order through the "Place Order" and "Pay Order" use cases.    * The order is in the "pending processing" status and has not been approved or rejected by a Product Manager.    * The system has a valid transaction ID from the "Pay Order" use case for refund processing.    * The customer has access to the order information (e.g., via email link or AIMS interface). 5. **Basic Flow of Events** 6. The customer accesses the order information (e.g., via email link or AIMS interface) 7. Customer selects the option to cancel the order ( click into link ) 8. Software display confirmation to customer 9. Customer confirm cancellation 10. The software checks the order status to confirm it is "pending processing". 11. The software sends a refund request to VNPay, including the transaction ID and order details. 12. VNPay processes the refund and returns a success response to AIMS. 13. The software updates the order status to "cancelled" & log operation for audit & display refund succeed and sends a confirmation email to the customer. 14. **Alternative Flows**  | **No** | **Location** | **Condition** | **Action** | **Resume Location** | | --- | --- | --- | --- | --- | | 1 | At step 3 | Customer doesn’t confirm |  | Use case ends | | 1 | At Step 4 | Order is not "pending processing" | AIMS displays "Order cannot be cancelled" | Use case ends | | 2 | At Step 7 | Refund fails (e.g., bank error / timeout ) | Software display refund failed and prompt customer to try again or exit | Resumes at Step 2 or Use case ends |  1. **Input Data**  | **No** | **Data Fields** | **Description** | **Mandatory** | **Valid Condition** | **Example** | | --- | --- | --- | --- | --- | --- | | 1 | Order ID | Unique identifier of the order | Yes | String: Exists in system, paid, "pending processing" status | "ORD001" | | 2 | Cancellation Confirmation | Customer confirms cancellation | Yes | Enum : "Yes" or "No" selection; "No" ends UC | "Yes" |  1. **Output Data**  | **No** | **Data Fields** | **Description** | **Display Format** | **Example** | | --- | --- | --- | --- | --- | | 1 | Order Status | Updated status after cancellation (if success) | Enum | "Cancelled" | | 2 | Refund Status | Result of refund process | Enum | "Refunded" or "Failed" | | 3 | Confirmation Message | Feedback to customer based on result | String | "Order cancelled successfully" or "Cancellation failed, please retry" | | 4 | Transaction ID | Refund transaction ID (if success) | String | "RFND123456789" | | 5 | Refund Datetime | Time of refund (if success) | Datetime : DD/MM/YYYY HH:MM:SS | "01/03/2025 15:00:00" |  1. **Postconditions** 2. If cancellation and refund are successful:    1. The order status is updated to "cancelled" in the system.    2. The Customer receives a confirmation email with refund details.    3. The refund is processed, and funds are returned to the Customer’s payment method via VNPay.    4. The system logs the cancellation operation for auditing purposes. 3. If cancellation or refund fails:    1. The order remains in "pending processing" status.    2. The Customer is prompted to retry or contact support. 4. The "Cancel Order" use case ends, and the Customer can take further actions (e.g., retry or browse products). |
|  |

# Supplementary specification

## Functionality

1. **Data Validation**

* System shall validate all user inputs for format and content
* System shall provide appropriate error messages for invalid inputs
* System shall prevent submission of incomplete required information

1. **Transaction Logging**

* System shall maintain logs of all transactions
* System shall record timestamps for all operations

1. **Notification Management**

* System shall send email notifications for significant events
* System shall display on-screen notifications for immediate user feedback
* System shall provide confirmation messages for completed actions

1. **Inventory Control**

* System shall maintain accurate product inventory counts
* System shall update inventory when orders are placed

1. **Search & Filtering**

* System shall provide search functionality across appropriate data
* System shall support filtering of results

## Usability

1. **Ease of Use**:

* The system must provide a simple, intuitive desktop interface, allowing new users (customers, product managers, administrators) to familiarize themselves within 10 minutes without detailed instructions, with key functions (search, add product, checkout) accessible in no more than 2 clicks.
* Notifications (e.g., "Insufficient inventory quantity," "Invalid delivery address") must use clear, understandable language, accompanied by explicit action buttons (e.g., "Update cart," "Re-enter information") to guide users in quickly resolving issues.
* The order placement, product management, and user administration processes must follow a linear workflow with "Back" and "Next" buttons at each step, facilitating easy navigation.

1. **Information Display**:

* Product lists (20 items per page) and pending order lists (30 orders per page) must be displayed in a table or grid format, with key columns (name, price, quantity, status) adjustable in width and supporting sorting (e.g., by price, warehouse entry date).
* Invoices, cart details, and product information must present data in a structured format (table or hierarchical list), separating product value, VAT, delivery fees, and total for transparency.

1. **Input Simplification**:

* Input forms (e.g., product details, delivery information, user data) must offer auto-suggestions (e.g., province/city list) and real-time format validation (e.g., date DD/MM/YYYY, 10-digit phone number), minimizing manual entry errors.
* Product managers can only add/edit one product at a time, with mandatory fields (e.g., title, value, barcode) clearly marked with an asterisk (\*) and immediate alerts if data is missing before saving.

1. **Accessibility**:

* The interface must use a minimum font size of 14px, a text contrast ratio of at least 4.5:1, and support keyboard shortcuts (e.g., Ctrl+S to save a product, Ctrl+D to delete) to enhance efficiency for desktop users.

1. **Training Requirements**:

* Customers require no prior training; the system must include in-interface guidance via tooltips or a "Help" link for functions like product search, order placement, and cancellation.
* Product managers and administrators must be trained within 4 hours to proficiently perform tasks such as adding/editing/deleting products, managing users, and approving orders, with concise documentation (under 10 pages).

1. **Language Support**:

* The system must default to Vietnamese, with standardized terms like "Add Product," "Checkout," "Cancel Order" to avoid confusion; English support must be available as an option for future expansion.

## Reliability

**System Availability**:

* The system must operate continuously 24/7, maintaining functionality for at least 300 hours without critical failure (per AIMS requirements), ensuring customers can shop and product managers/administrators can perform tasks at all times.

**Error Handling**:

* The system must detect and handle input errors (e.g., incorrect date format, product value outside 30%-150% range) with a failure rate below 0.5%, displaying specific error messages (e.g., "Price must be between 30% and 150% of original value") and allowing immediate correction.
* Inventory checks before order placement must be 99.9% accurate, automatically rejecting orders if stock is insufficient and notifying customers/product managers of the shortage quantity.

**Mean Time Between Failures (MTBF)**:

* The MTBF must be at least 300 hours (per AIMS), ensuring system stability while serving up to 1,000 concurrent customers and handling management/administrative operations.

**Recovery After Incidents**:

* Following an incident (e.g., server failure, VNPay disconnection), the system must resume normal operation within a maximum of 1 hour (per AIMS), preserving cart data, orders, and operation history without loss, and notifying affected users via email if necessary.

**Data Integrity**:

* Product data (price, quantity, barcode) and order data (payment, delivery) must be synchronized between the interface and database with a discrepancy rate below 0.1%, preventing mismatches in displayed information.
* Operation history (add/edit/delete products, place/cancel orders) must be fully recorded with 100% reliability and stored for at least 6 months for auditing purposes.

**Security Limits**:

* The system must enforce that product managers cannot delete/update more than 30 products daily, with an error rate for exceeding this limit below 0.01%, and immediately notify users if violated (e.g., "Limit of 30 products reached").

## Performance

1. Response Time:
   * Displaying a list of 20 products (random or search results) must complete within 2 seconds under normal conditions or 5 seconds during peak hours (per AIMS).
   * Adding/editing/deleting products by product managers must save to the database within 2 seconds (normal) or 5 seconds (peak).
   * Calculating and displaying invoices (including product prices, VAT, delivery fees) must complete within 2 seconds (normal) or 5 seconds (peak).
   * Sending order confirmation emails (post-payment) must complete within 3 seconds (normal) or 5 seconds (peak).
2. Throughput:
   * The system must handle up to 1,000 concurrent customers (per AIMS) without significant performance degradation (response time increase not exceeding 10%), supporting product searches, order placements, and payments.
3. Database Performance:
   * Data queries (products, inventory, user information) must complete within 1 second, even with a database containing 10,000 products or 5,000 orders, ensuring fast processing for all operations.
4. Order Processing:
   * Inventory checks before order placement must complete within 1 second, supporting up to 500 simultaneous orders during peak hours without delays.
5. Payment Integration:
   * Integration with VNPay for credit card payments must complete within 5 seconds (normal) or 10 seconds (peak), ensuring smooth transaction processing without interruptions.
6. Peak Load Management:
   * The system must maintain performance with a 50% increase in user volume (e.g., from 1,000 to 1,500 concurrent customers), with response time increases not exceeding 20%, validated under high-load testing conditions.

## Supportability

**1. Maintainability:**

The system should allow for easy updates, modifications, and deletions in the product catalog. This includes version control for tracking changes, the ability to rollback product updates, and logging product deletions when necessary.

1. **Documentation:**

Comprehensive documentation should be provided, detailing how Product Managers can add, update, delete, and manage product information. It should also cover system maintenance procedures for future administrators.

1. **System Updates:**

The system must be capable of supporting future feature updates and product management enhancements (including add/update/delete functions) without significant downtime, ensuring minimal impact on users during product catalog changes or improvements.

1. **Technical Support:**

A technical support system should be in place, allowing Product Managers to report any issues with adding, updating, or deleting products and receive timely assistance.

1. **Modularity:**

The product management module should be designed in a modular fashion, allowing for future enhancements (e.g., new product attributes, integration with external systems) without disrupting the overall system architecture.

1. **Scalability:**

The system should be able to scale with increasing numbers of products, product updates, and deletions, ensuring performance remains optimal as the product catalog grows and changes.

## Other requirements

* + - 1. **Security**:

The system must ensure that only authorized Product Managers can access, modify, or delete product information. Data encryption should be used for sensitive fields such as pricing.

* + - 1. **Data Integrity**:

Data consistency must be maintained during product updates and deletions. Any interruptions during the update or deletion process should trigger an automatic rollback to prevent partial updates or deletions.

* + - 1. **Performance**:

The system should handle product additions, updates, and deletions efficiently, with response times under 2 seconds for basic operations such as saving, validating, or deleting product information.

* + - 1. **Backup and Recovery**:

A backup mechanism should be in place to ensure that product information is not lost in case of a system failure. Regular backups should be scheduled, and recovery should be quick and reliable, including restoring deleted product information if needed.

* + - 1. **Compliance**:

The system should comply with industry standards and regulations regarding data handling, such as GDPR for protecting personal data, especially when adding, updating, or deleting sensitive information.

* + - 1. **Integration**:

The product management system should support integration with external systems such as inventory management, e-commerce platforms, or third-party analytics tools, and should ensure that product deletions are reflected in these external systems