HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and Communications Technology

Software Requirement Specification

Version 1.2

AIMS – An Internet Media Store

Subject: ITSS Software Development

Group 2

Đặng Văn Nhân – 20225990

Hà Việt Khánh – 20225979

Hồ Bảo Thư – 20226003

Trần Cao Phong – 20226061

Nguyễn Lan Nhi - 20225991

*Hanoi,* *March 2025*

Table of contents

Table of contents 1

1 Introduction 3

1.1 Objective 3

1.2 Scope 3

1.3 Glossary 4

1.4 References 6

2 Overall Description 7

2.1 Survey 7

2.2 Overall requirements 7

2.3 Business process 8

2.3.1 Business Operation – Place Order 9

2.3.2 Business Operation – Manage Products 9

2.3.3 Business Operation – Manage Users 10

3 Detailed Requirements 11

3.1 Use case “Place order” 11

3.2 Use case “Pay order” 15

3.3 Use case “Place rush order” 17

3.4 Use case “Cancel order” 20

3.5 Use case “View product details” 21

3.6 Use case “Create products” 22

3.7 Use case “Update products” 24

3.8 Use case “Delete products” 26

3.9 Use case “Approve order” 27

3.10 Use case “Reject order” 28

3.11 Use case “Create users” 30

3.12 Use case “View homepage” 31

3.13 Use case “Log in” 32

3.14 Use case “Search products” 33

3.15 Use case “Add products to cart” 34

3.16 Use case “Update products in cart” 36

3.17 Use case “View products in cart” 37

3.18 Use case “Delete products in cart” 38

3.19 Use case “View pending orders” 39

4 Supplementary specification 42

4.1 Functionality 42

4.2 Usability 42

4.3 Reliability 42

4.4 Performance 42

4.5 Supportability 42

4.6 Other requirements 43

# Introduction

## Objective

An Internet Media Store (AIMS) is an e-commerce desktop application designed to facilitate the buying and selling of physical media products such as books, CDs, LP records, and DVDs. This system provides a seamless shopping experience for customers while enabling product managers to efficiently manage inventory. The software is designed to handle up to 1,000 concurrent users without performance degradation and can operate continuously for 300 hours without failure.

This Software Requirements Specification (SRS) document defines the functional and non-functional requirements of the AIMS software, serving as a reference for developers, testers, product managers, and other stakeholders involved in its development and deployment.

## Scope

The AIMS software allows users to browse, search, purchase, and manage physical media products in a structured and user-friendly manner. The system supports both normal and rush orders, integrates with VNPay for payment processing, and provides robust inventory management capabilities. Key functionalities include:

* For Customers:
  + Searching and filtering products by attributes.
  + Viewing product details and adding items to the cart.
  + Normal or rush order options.
  + Making payment through VNPay.
  + Viewing and managing past orders, including order cancellations.
* For Product Managers:
  + Adding, editing, and removing products from the inventory.
  + Monitoring and updating product prices.
  + Reviewing and approving customer orders.
  + Managing stock availability.
* For Administrators:
  + Creating and managing user accounts.
  + Assigning roles and permissions to users.

## Glossary

| ***No*** | ***Term*** | ***Explanation*** | ***Example*** | ***Note*** |
| --- | --- | --- | --- | --- |
| 1 | session | A session is a temporary period of interaction between a user and a system, during which the user accesses and interacts with software, a website, or an application. The system maintains session-specific data, such as user authentication, preferences, and activity history, for the duration of the session. Once the session ends—either through user logout, session timeout, or browser closure—the stored session data is typically cleared unless explicitly saved. | software session |  |
| 2 | VAT (Value-added tax) | VAT (Value-Added Tax) is a consumption tax applied to the value added at each stage of production or distribution of goods and services. It is typically calculated as a percentage of the final selling price and is collected by businesses on behalf of the government. |  |  |
| 3 | API (Application Programming Protocol) | An API (Application Programming Interface) is a set of rules, protocols, and tools that enables different software applications to communicate and interact with each other. | VNPay API | AIMS connects API of VNPay for transaction. |
| 4 | Payment gateway | A payment gateway is a technology service that facilitates the secure transmission of payment information between a merchant's website or application and the financial institutions involved in processing payment transactions. |  |  |
| 5 | GUI (Graphical user interface) | A Graphical User Interface (GUI) refers to the visual elements and interactive components of a software application that allow users to interact with the system using graphical icons, buttons, menus, windows, and other visual elements. |  |  |
| 6 | Credit card | A credit card is a payment card issued by a financial institution, such as a bank or credit union, that allows cardholders to borrow funds up to a predetermined limit to make purchases or pay for goods and services. |  | AIMS currently supports for credit card payment through VNPay |
| 7 | Authentication | Authentication is the process of verifying the identity of a user, device, or system attempting to access a resource or service. |  |  |
| 8 | Response time | Response time refers to the amount of time it takes for a system to respond to a user's request or input. |  |  |

## References

# Overall Description

## Survey

The system under consideration is an internet media store designed to facilitate the desire to purchase digital media products online by the customers. This software serves as a comprehensive platform for not only customers but also the shop managers or product managers.

The system includes four actors:

- **Customer**: They can view, search for or sort in order by many factors the products that are available in the store. To place an order, they must add, update the products in the cart, and provide delivery information to the system. If the information is available, the customer needs to pay for the order through the VNPay platform. If the order is created successfully, the customer will get an email from AIMS about the order invoice. Moreover, customers can place orders with rush orders in some cases.

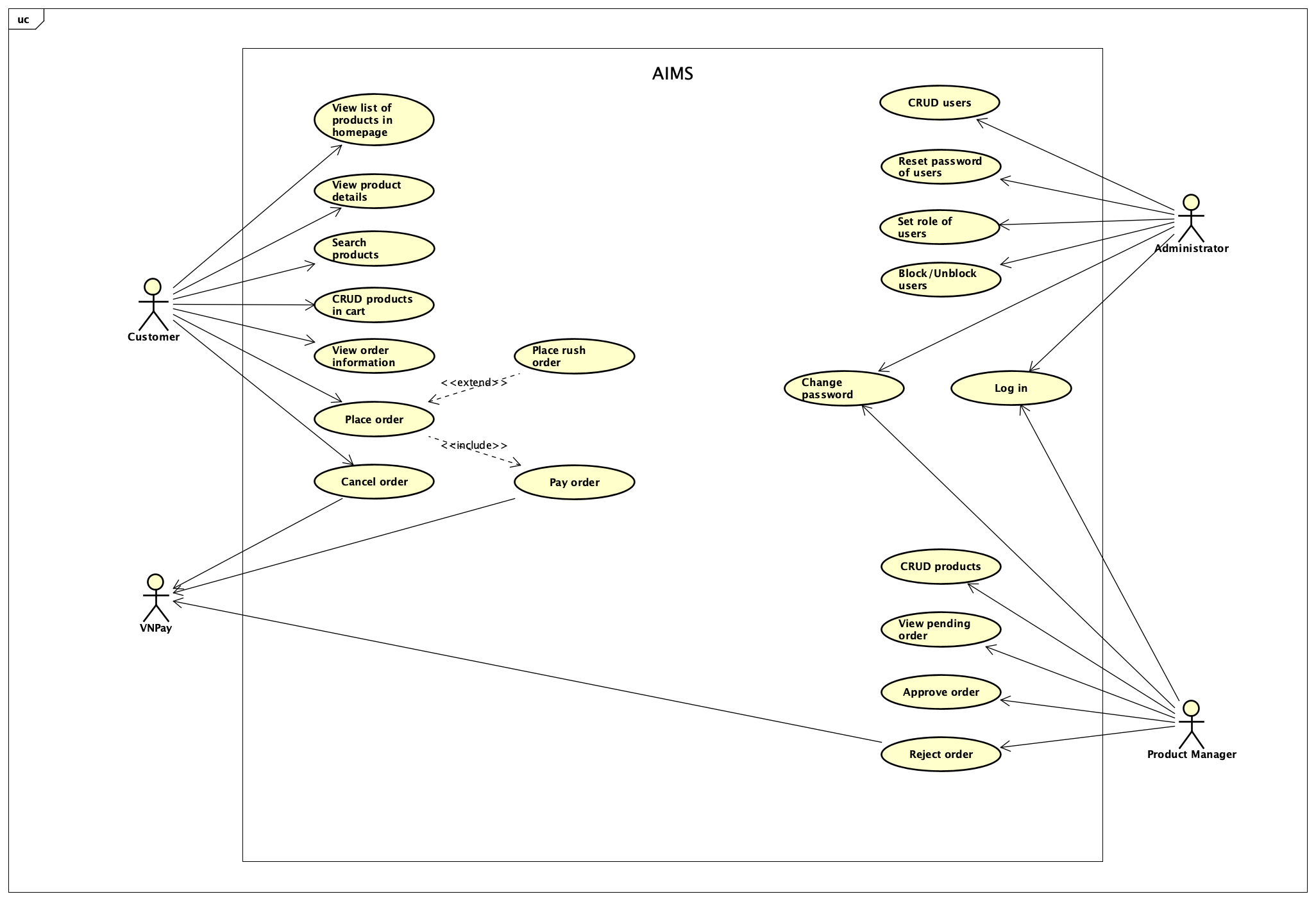
- **Product Manager**: They can manage their products in their shop through the user interface of manager supplied by AIMS software. They can add, remove, update information about the products. Moreover, under some circumstances, they can apply sales for many products.

- **Administrator**: They can gain access to mange user in the system. They can also block or unblock the user (customer, product manager) and the system will send the mail to that account. In addition, admin can also change the role of the user like a user can be a customer and a product manager at the same time.

- **VNPay**: A stakeholder in the system when they supply the API to make the transaction in the AIMS software.

## Overall requirements

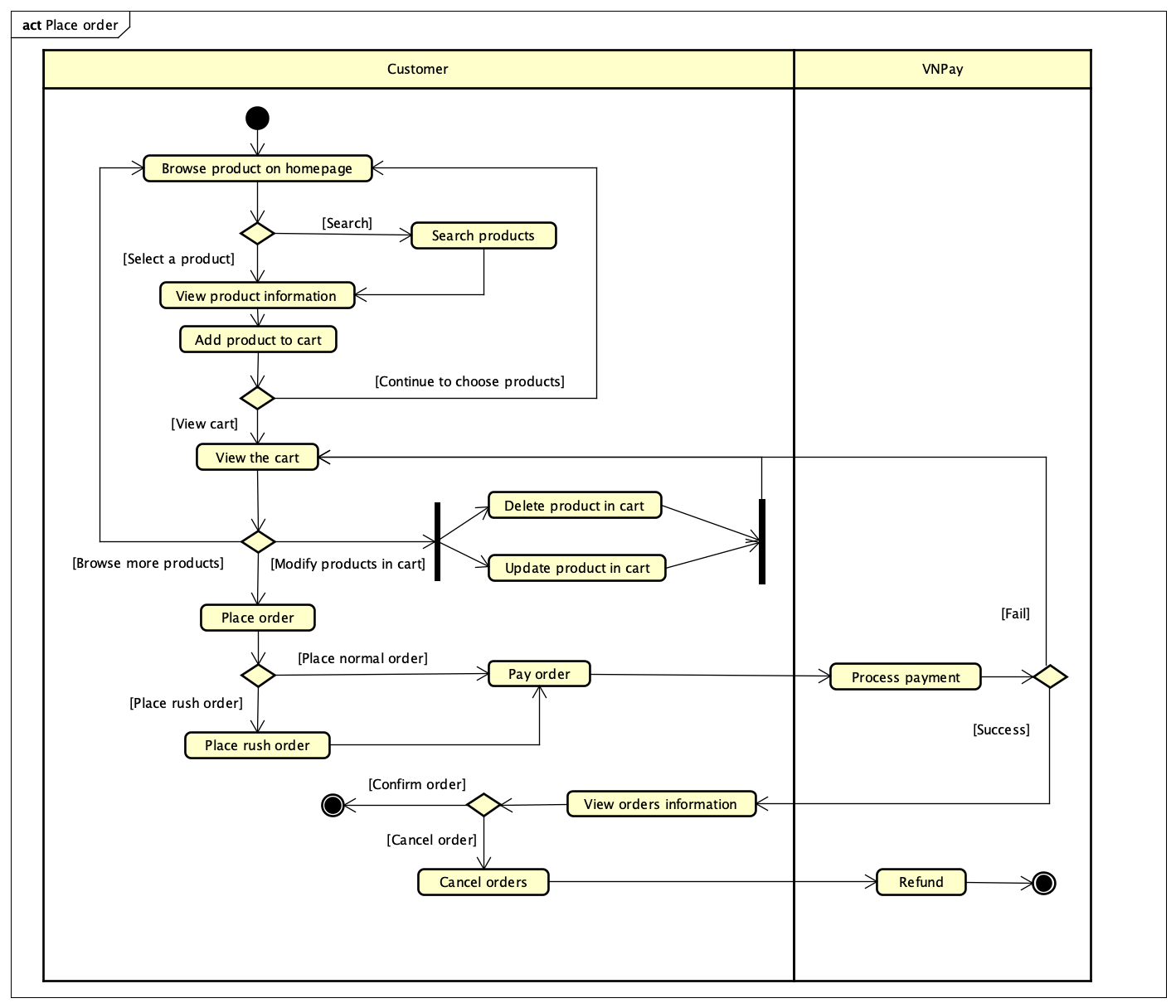
Use case diagram represents the interactions between actors and use cases. It represents the functional requirements of the system, showing the interaction between external and internal actors with the system.

**

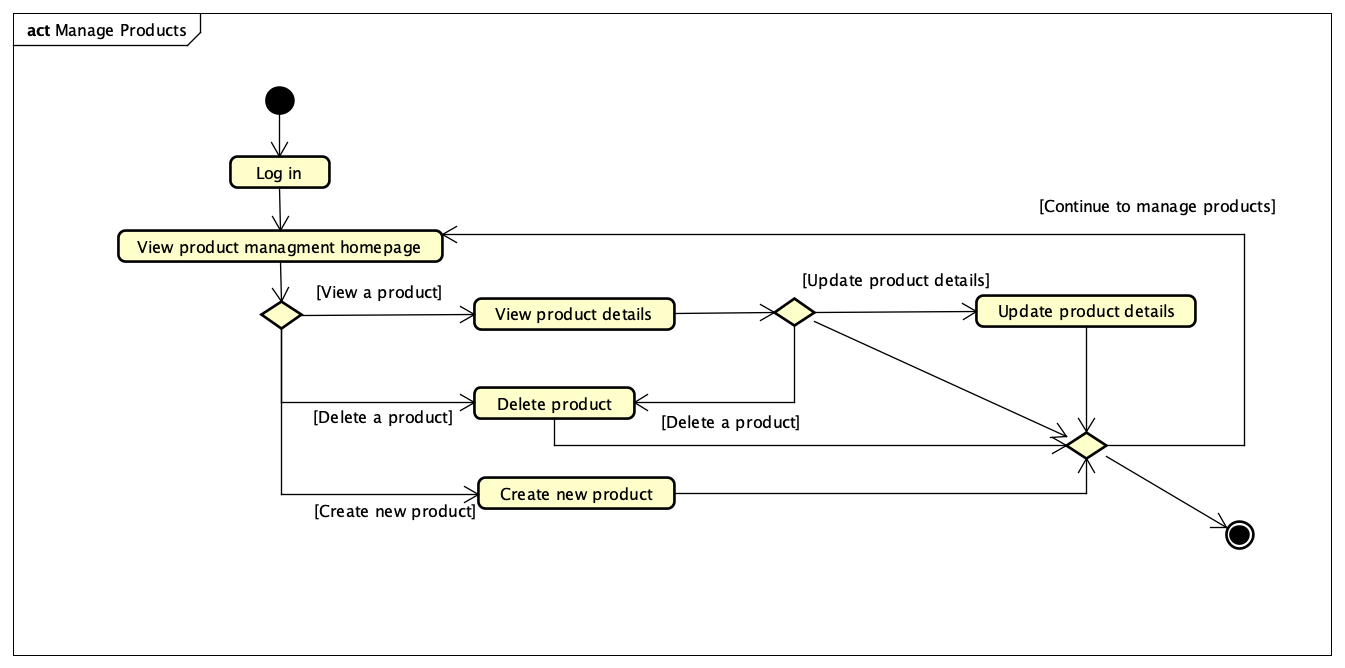
## Business process

In the AIMS software, there are three primary business operations: "**Place Order**" carried out by Customer, "**Manage Products**" handled by Product Managers, and "**Manage Users**" by Administrators. Each of these business processes is detailed through activity diagram in their respective sections.

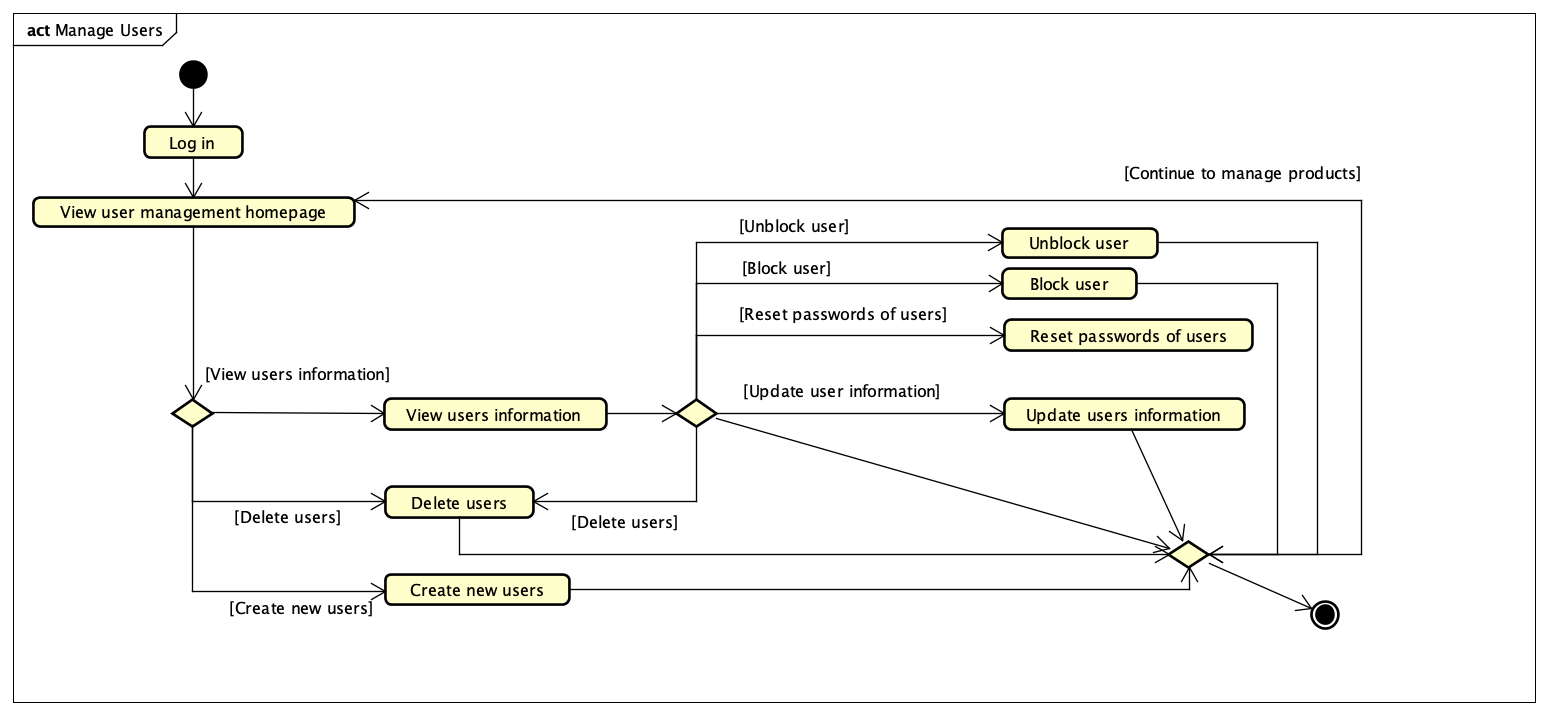
### Business Operation – Place Order



### Business Operation – Manage Products



### Business Operation – Manage Users



# Detailed Requirements

## Use case “Place order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Place order”**   1. **Use case code**   UC001   1. **Brief Description**   This use case describes the interaction between Customer and AIMS software when Customer wishes to place an order.   1. **Actors**    1. **Customer** 2. **Preconditions**   Cart is not currently empty; All selected items are in stock and available in the requested quantity; At least 1 item is selected by Customer.   1. **Basic Flow of Events** 2. Customer chooses products they want to purchase and requests to place an order 3. Software checks the availability of products in the cart 4. Software displays the form of delivery information with order information 5. Customer enters and submits delivery information (see Table 1) 6. Software check the submitted information 7. Software calculates and updates order information with shipping fee 8. Software displays and saves invoice information (see Table 2) 9. Customer chooses payment method and asks for payment 10. Software calls Use Case “Pay order” 11. Software creates and saves a new order 12. Software sends email to customer for the success order 13. Software makes the cart empty 14. Software displays the successful order notification, the order and the transaction information (see Table 3) 15. **Alternative flows**   Table A1 - Alternative flows of events for UC “Place order”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 3 | If the products are not available | The software displays the inventory quantity of each unmet product | Use case ends | | 2. | At Step 5 | If the delivery information is invalid | The software notifies that the delivery information is invalid | At Step 3 | | 3. | At Step 5 | If the user chooses to place a rush order | The software inserts use case “Place rush order” | At Step 6 | | 4. | At Step 6 | If the user adjusts the delivery method or the items | The software recalculates the delivery fees and display the corresponding invoice | At Step 7 | | 5. | At Step 8 | If the order payment is not successful or goes back from payment |  | At Step 5 |  1. **Input data**   Table 1 - Input data of Customer   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | customerName | Receiver’s full name | Yes | Non-empty string | Ho Bao Thu | |  | phoneNumber | Receiver’s phone number | Yes | Exacly 10 digits | 0123456789 | |  | deliveryProvince | Province or city | Yes | Must be one of the allowed list | Hanoi | |  | deliveryAddress | Full street address | Yes | Non-empty string | 123 ABC street, DEF district | |  | deliveryInstructions | Any extra instructions for delivery | Yes |  | "Please call on arrival" | |  | isRushOrder | Rush‐order flag | Yes | Boolean | false | |  | email | Contact email | Yes | Valid domain part with @ | [abc@gmail.com](mailto:abc@gmail.com) | |  | expectedDate | Expected delivery date (format yyyy-MM-dd) | No | ISO-date string | |  | | --- | |  |   "2025-07-05T14:30:00" |  1. **Output data**   Table 2 - Output data of order information and shipping fee   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Title | Title of a media product |  | Book Cây cam ngọt của tôi | |  | Price | Price of the corresponding media product | * Comma for thousands separator * Positive integer * Right alignment | 200,000 | |  | Quantity | Quantity of the corresponding media | * Positive integer * Right alignment | 1 | |  | Amount | Total money of the corresponding media | * Comma for thousands separator * Positive integer * Right alignment | 200,00 | |  | Subtotal before VAT | Total amount of all products in the order excluding VAT |  | 1,263,000 | |  | Subtotal after VAT | Total amount of all products in the order with VAT |  | 1,389,300 | |  | Shipping fee |  |  | 20,000 | |  | Total |  | * Comma for thousands separator * Positive integer * Right alignment | 1,409,300 | |  | Concurrency |  |  | VND |   Table 3 - Output data of general information of order and transaction information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Customer name |  |  | Ho Bao Thu | |  | Phone Number |  | 10 digits | 0123456789 | |  | Province |  |  | Hanoi | |  | Address |  |  | 123 ABC street, DEF district | |  | Total amount |  | * Comma for thousands separator * Positive integer * Right alignment | 1,283,004 | |  | Transaction ID |  |  | 00001 | |  | Transaction content |  |  | Order #00001 | |  | Transaction date |  | dd/mm/yyyy | 23/02/2025 | |  | Currency |  |  | VND | |  | Order status |  |  | Pending | |  | Order email |  |  | abc@gmail.com |  1. **Postconditions**   If the payment is successful, the cart is emptied; order, payment, and delivery information are stored; the user receives a confirmation, and stock levels are updated. If the payment fails, no changes occur. |

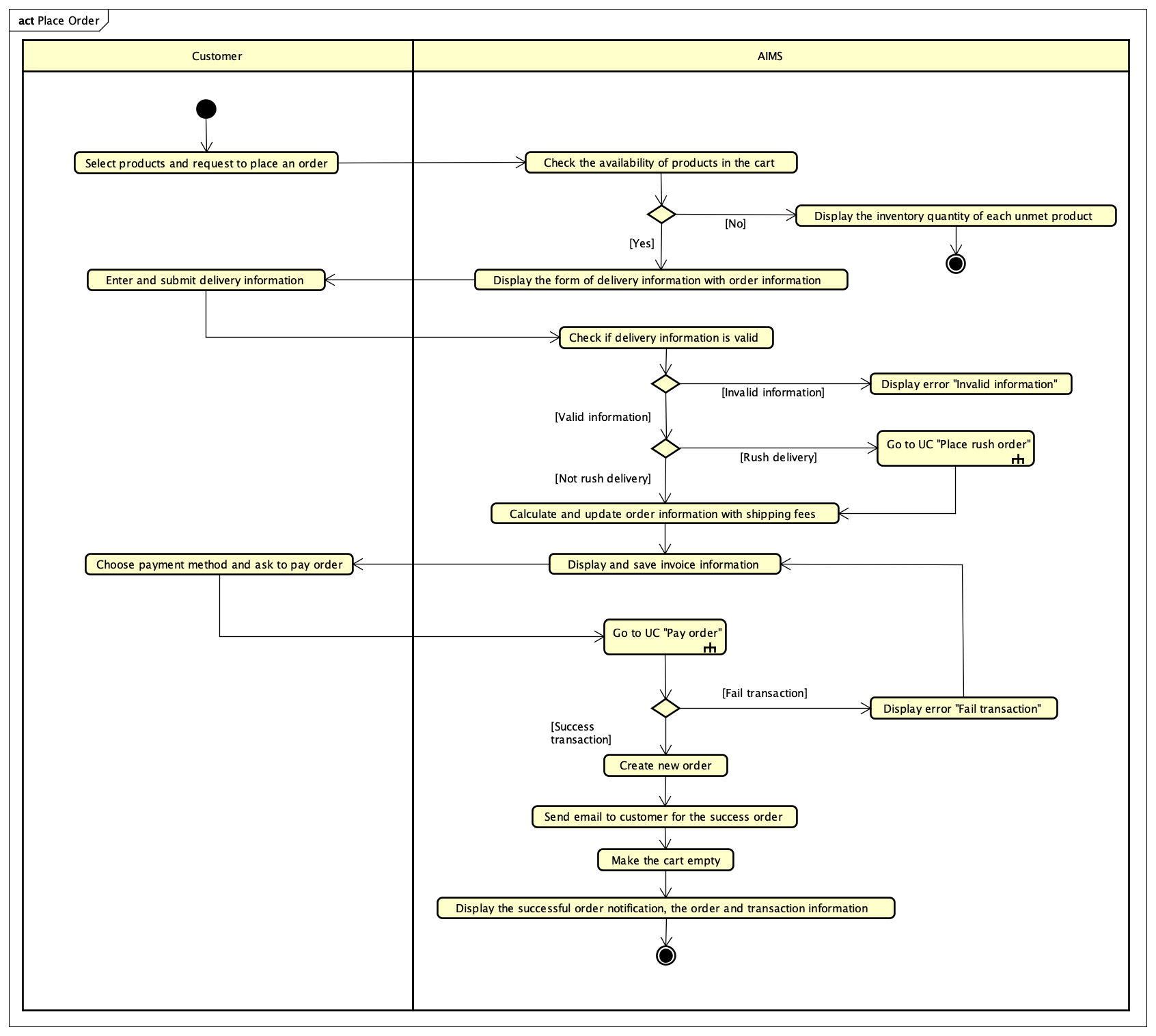


Figure 1: Activity Diagram for "Place Order"

## Use case “Pay order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Pay order”**   1. **Use case code**   UC002   1. **Brief Description**   This use case describes how software interacts with Customer and VNPay to pay the previous order.   1. **Actors**    1. **Customer**    2. **VNPay** 2. **Preconditions**   The order is in payment state and software can interact with VNPay.   1. **Basic Flow of Events**   1. Software redirects Customer to VNPay with payment information  2. Customer performs the transaction with VNPay  3. VNPay sends payment result to VNPay  4. Software updates and saves the invoice and payment transaction (see Table 1)  5. VNPay notifies the payment result   1. **Alternative flows**   Table A1 - Alternative flows of events for UC “Pay order”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1 | At Step 2 | Customer cancels payment | Software notices the cancellation. | Use case ends | | 2 | At Step 3 | Payment fails | Software notifies the Customer of failure. | Use case ends |  1. **Input data** 2. **Output data**   Table 1 - Output data of invoice   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Title | Title of the media product | Text | DVD Phim Vượt ngục | |  | Price | Price of the media product | Positive integer, right alignment | 123,000 | |  | Quantity | Quantity of the media product | Positive integer, right alignment | 2 | |  | Amount | Total money of the media product | Positive integer, right alignment | 246,000 | |  | Subtotal before VAT | Total price of products in the cart before VAT | Positive integer, right alignment | 2,106,000 | |  | Subtotal with VAT | Total price of products in the cart with VAT | Positive integer, right alignment | 2,316,600 | |  | Shipping fee | Shipping fee | Positive integer, right alignment | 30,000 | |  | Total | Total price of subtotal and shipping fee | Positive integer, right alignment | 2,346,600 | |  | Currency |  |  | VND |  1. **Postconditions**   If the payment is successful, the customer is notified and receives a bill detail email. If it fails, nothing happens. |

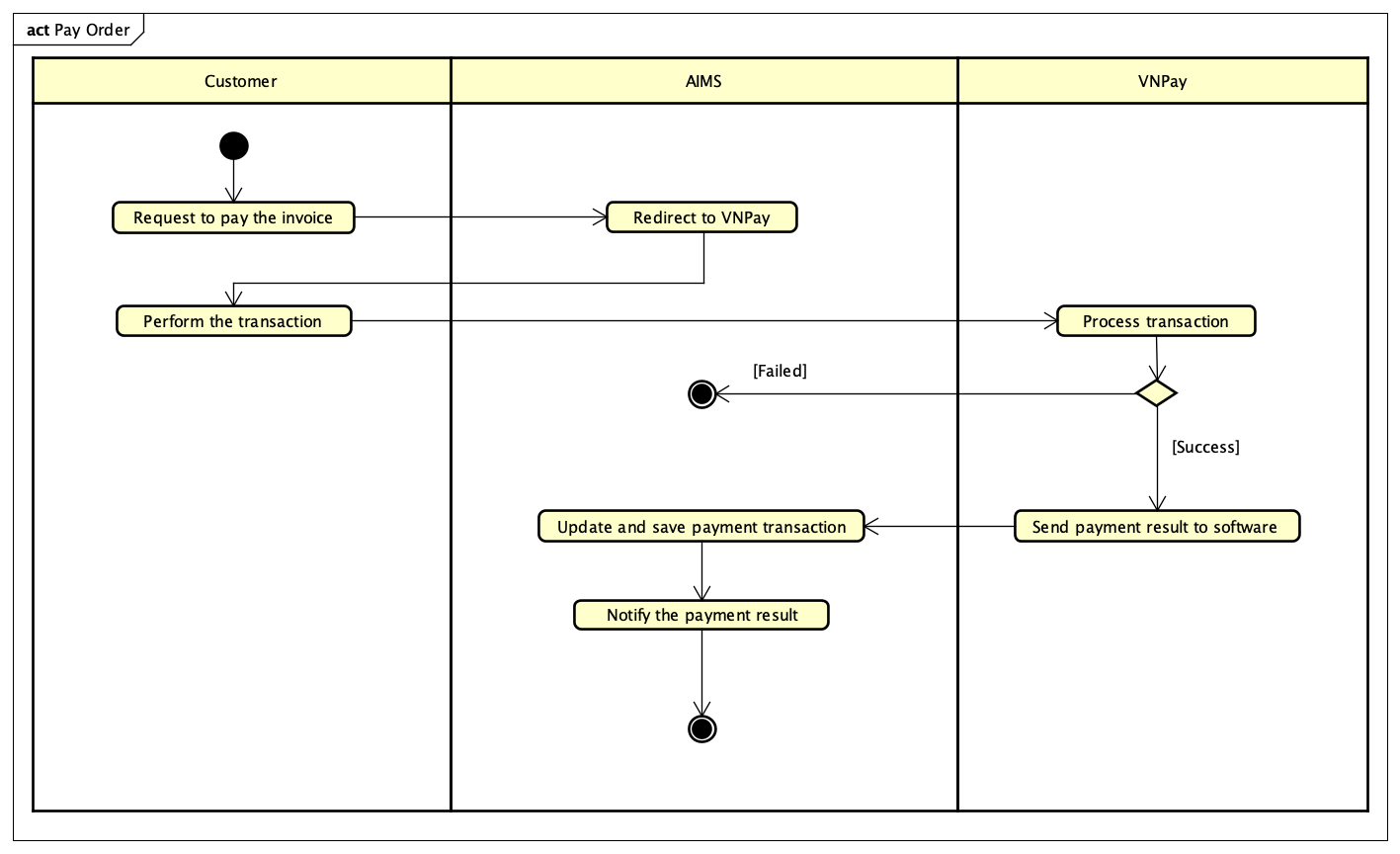


Figure 2: Activity diagram for "Pay Order"

## Use case “Place rush order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Place rush order”**   1. **Use case codes**   UC003   1. **Brief Description**   This use case describes the interaction between the customer and software when the customer wants to place an order with rush order.   1. **Actors**    1. **Customer** 2. **Preconditions**   Customer’s cart is not empty, the order is being placed by customer.   1. **Basic Flow of Events** 2. Customer requests to place rush order when placing the order 3. The software checks whether the delivery address supports these services and the availability of products 4. The software asks for additional information (see Table 1) 5. Customer enters and submits additional information 6. AIMS software calculates, updates, and displays the total order information with shipping fee (see Table 2) 7. **Alternative flows**   Table A1 - Alternative flows of events for UC “Place rush order”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | If the products or the address is not available for rush order | The software displays: Invalid product/Invalid address and returns to UC001 “Place order” | Use case ends | | 2. | At Step 4 | If the delivery information of Customer is not eligible | The software notifies to Customer and requires Customer to fill the information again | At Step 3 |  1. **Input data**   Table 1 - Input data of Customer   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | | 1. | Order instructions | Additional notes from customer | No |  |  | | 2. | Delivery time |  | Yes | hh:mm, 24-hour based. Delivery time within 2 hours. | 14:20 |  1. **Output data**   Table 2 - Output data of shipping fee   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Standard shipping fee | Shipping fee for non-eligible express shipping products | * Comma for thousands separator * Positive integer * Right alignment | 30,000 | |  | Express shipping fee | Express shipping fee for eligible products | * Comma for thousands separator * Positive integer * Right alignment | 50,000 |  1. **Postconditions**   If a user successfully places a rush order, the rush order details are temporarily stored, and the delivery fee is calculated separately and displayed. However, if the user is unable to place a rush order, no changes occur, and the order continues as a standard delivery without any modifications. |

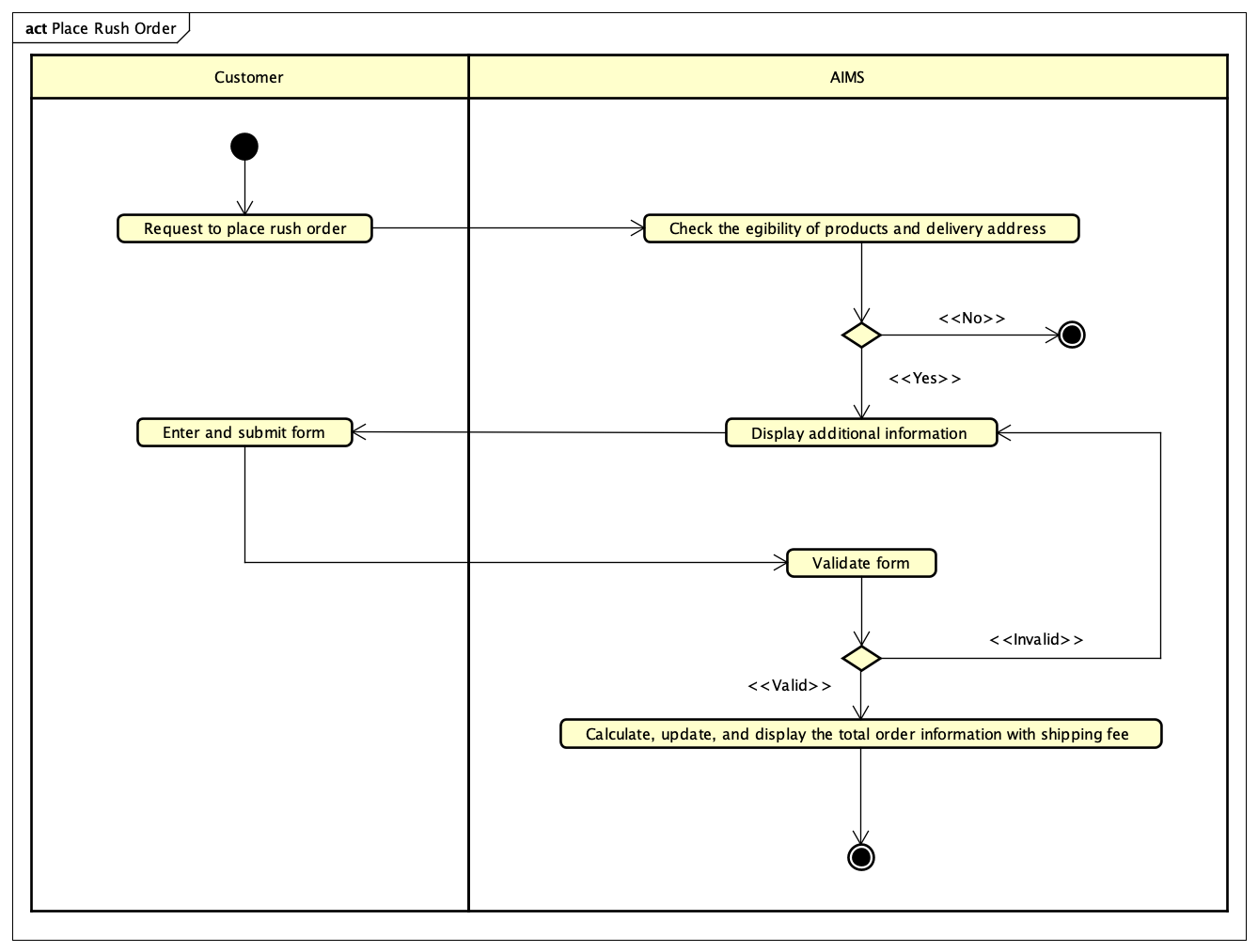


Figure 3: Activity Diagram for "Place Rush Order"

## Use case “Cancel order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Cancel order”**   1. **Use case code**   UC004   1. **Brief Description**   This use case describes the interaction between the customer and software when the customer wants to cancel an order.   1. **Actors**    1. **Customer** 2. **Preconditions**   Customer selects to view an order successfully.   1. **Basic Flow of Events** 2. Customer selects to cancel a specific order 3. The software checks the status of selected order 4. The software displays all relevant order details (see Table 1) 5. The software changes the order’s status to “Cancelled” 6. The software calls VNPay to refund the amount of that order. 7. **Alternative flows**   Table A1 - Alternative flows of events for UC “Cancel order”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | The order has been accepted/rejected by Product Manager before | The software informs to customer: “This order can not be cancelled” | Use case ends |  1. **Input data** 2. **Output data**   Table 1 - Output data of order details   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Transaction ID | Unique identifier for the order transaction | Numeric | 000018 | |  | Customer’s name |  | Text | Nguyen Lan Nhi | |  | Itemized list of products |  | Text | DVD | |  | Quantity | Number of units for each product | Positive integer | 1 | |  | Price | Price of each product | Positive integer, comma for thousands separator | 200,000 | |  | Total amount | Total cost including tax and shipping | Positive integer, comma for thousands separator | 260,000 | |  | Tax |  | Positive integer, comma for thousands separator | 30,000 | |  | Shipping fee |  | Positive integer, comma for thousands separator | 30,000 | |  | Shipping address | Delivery address provided by the customer | Text | Hoang Mai, Ha Noi | |  | Payment method |  | Text | VNPay | |  | Refund ID | Unique identifier for the refund transasction for VNPay | Numeric | 00812 | |  | Amount paid | Total amount paid by the customer | Positive integer, comma for thousands separator | 260,000 | |  | Refund status | Status of the refund process | Text (e.g., "Pending", "Failed", "Successful") | Successfully |  1. **Postconditions** |

## Use case “View product details”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “View product details”**   1. **Use case code**   UC005   1. **Brief Description**   This use case describes the interaction between Customer/Product Manager and software when Customer/Product Manager wants to view product details.   1. **Actors**    1. **Product Manager/Customer** 2. **Preconditions**   Customer/Product Manager has already accessed theo product listing screen.   1. **Basic Flow of Events** 2. Customer/Manager selects a product from the product listing screen 3. Software retrieves the product details from database and displays the product details (see Table 1) 4. **Alternative flows** 5. **Input data** 6. **Output data**   Table 1 - Output data of product details   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Title of product |  |  | Movie | |  | Category | Category of products |  | DVD | | 3. | Rush order support |  | Yes/No | Yes | | 4. | Price | Price of each product | Positive integer, comma for thousands separator | 200,000 | | 5. | Currency | Currency of price | Text | VND | | 6 | Availability status | In stock/Out of Stock |  | In stock | | 7 | Product image |  | Image |  | |  | Quantity | Quantity of product | Integer | 12 | |  | Special information fields for the product type |  | Text |  |  1. **Postconditions** |

## Use case “Create products”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Create products”**   1. **Use case code**   UC006   1. **Brief Description**   This use case describes the interaction between Product Manager and AIMS software when Product Manager wishes to create a new producet.   1. **Actors**    1. **Product Manager** 2. **Preconditions**   Product manager must log in with a product manager account.   1. **Basic Flow of Events** 2. The manager requests the creation of a new product 3. The system displays a form for the manager to enter product information 4. The manager needs to select the type of product he wants to add so that the specific information form for that product will appear 5. The manager enters information (see Table 1) 6. The system checks the validity of entered product information 7. The system notifies the successful creation of new product 8. **Alternative flows**   Table A1 - Alternative flows of events for UC “Create products”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 5 | If the user leaves any of the required information blank or invalid | The software reports an error and requires the manager to fill in all the information and correct it. | At Step 4 |  1. **Input data**   Table 1 - Input data of Product Manager   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Title | Title of media product | Yes |  | SHINee WORLD VI | |  | Price |  | Yes | Numeric | 20.000 | |  | Total quantity |  | Yes |  | 10 | |  | Weight | Weight of a product | Yes | Numeric, in kilogram unit | 2kg | |  | Rush order support |  | Yes | Yes/No | Yes | |  | Image URL |  | Yes |  |  | |  | Barcode |  | Yes |  | 250508 | |  | Description |  | Yes |  |  | |  | Dimension |  | Yes |  |  | |  | Category |  | Yes |  |  | |  | Special information fields for the product type |  | Yes |  |  |  1. **Output data** 2. **Postconditions**   A new product will be created, if the creation is successful. |

## Use case “Update products”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Update products”**   1. **Use case code**   UC007   1. **Brief Description**   This use case describes the interaction between Product Manager and Software when there is a need to update products.   1. **Actors**    1. **Product Manager** 2. **Preconditions**   Product manager must log in with a product manager account.   1. **Basic Flow of Events** 2. The manager selects the product that needs updating 3. The system displays the product information form for the manager (see Table 1) 4. The manager fills in the information that needs to be updated 5. The system checks the validity of the entered information 6. The system updates information, announces success and returns the manager to the product manager screen 7. **Alternative flows**   Table A1 - Alternative flows of events for UC “Update products”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | If the product manager has updated more than 30 products per day | The software reports an error and returns the manager back to the product manager screen | Use case ends | | 2. | At Step 3 | If the product manager leaves any required information blank or enters invalid data | The software reports an error and requires the manager to fill in all the information or correct it. | At Step 2 | | 3. | At Step 3 | If the product manager has updated price more than twice per day | The software reports an error and returns the manager back to the product manager screen. | Use case ends |  1. **Input data**   Table 1 - Input data of Product Manager   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Title | Title of media product | Yes |  | SHINee WORLD VI | |  | Price |  | Yes | Numeric | 20.000 | |  | Total quantity |  | Yes |  | 10 | |  | Weight | Weight of a product | Yes | Numeric, in kilogram unit | 2kg | |  | Rush order support |  | Yes | Yes/No | Yes | |  | Image URL |  | Yes |  |  | |  | Barcode |  | Yes |  | 250508 | |  | Description |  | Yes |  |  | |  | Dimension |  | Yes |  |  | |  | Category |  | Yes |  |  | |  | Special information fields for the product type |  | Yes |  |  |  1. **Output data** 2. **Postconditions**   The product will be updated if the operation success. |

## Use case “Delete products”

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Delete products”**   1. **Use case code**   UC008   1. **Brief Description**   This use case describes the interaction between Product Manager and Software when there is a need to delete products.   1. **Actors**    1. **Product Manager** 2. **Preconditions**   Product manager must log in with a product manager account.   1. **Basic Flow of Events** 2. The manager selects the product that needs deleting. 3. The system warns the manager and asks the manager to be sure with the above choice. 4. The system deletes products, announces success and returns the manager to the product manager screen. 5. **Alternative flows**   Table A1 - Alternative flows of events for UC “Update products”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 1 | If the product manager has deleted more than 30 products per day | The software reports an error and returns the manager back to the product manager screen | Use case ends |  1. **Input data** 2. **Output data** 3. **Postconditions**   The products will be deleted if the operation success. |

## Use case “Approve order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Approve order”**   1. **Use case code**   UC009   1. **Brief Description**   This use case describes the interaction between Product Manager, VNPay and software when the product manager wants to approve a pending order.   1. **Actors**    1. **Product Manager**    2. **VNPay** 2. **Preconditions**   The list of pending orders is not empty, product manager logged in before accessing the pending orders list.   1. **Basic Flow of Events** 2. Product manager views the list of pending orders and selects a specific order 3. Product manager views the details of this order 4. Product manager approves the order 5. The software checks the approved order 6. The software sends email of order information to customers to notice that the order was approved 7. **Alternative flows**   Table A1 - Alternative flows of events for UC “Approve order”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 4 | If there are not enough products in stock | The software rejects accepting order and sends notification to customer, and asks VNPay to refund to customer | Use case ends |  1. **Input data** 2. **Output data**   Table 1 - Output data of general information of order and transaction information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Order ID |  |  | 1 | |  | Transaction date |  | Timestamp | YY-MM-DDTHH:MM:SS | |  | Order Type |  |  | Normal or Rush | |  | List of products in order |  |  | Product name | |  | Total amount |  | * Comma for thousands separator * Positive integer * Right alignment | 1,283,004 | |  | Currency |  |  | VND |  1. **Postconditions**   If approval is successful, the status of selected order is updated. If approval is fail, no changes are made. |

## Use case “Reject order”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Reject order”**   1. **Use case code**   UC010   1. **Brief Description**   This use case describes the interaction between Product Manager, VNPay and software when the product manager wants to reject a pending order.   1. **Actors**    1. **Product Manager**    2. **VNPay** 2. **Preconditions**   The list of pending orders is not empty, product manager logged in before accessing the pending orders list.   1. **Basic Flow of Events** 2. Product manager requests to reject the order and submits the reason to reject (see Table 1) 3. Software changes the status of the order to ‘Rejected – Not refunded’ 4. The software sends email of order information to customer to notice that the order was rejected (see Table 2) 5. The software asks VNPay to refund the customer 6. Software saves transaction and updates status to ‘Rejected – Refunded’ 7. Software sends email of order transaction to customer to notice that the order was refunded (see Table 2) 8. **Alternative flows** 9. **Input data**   Table 1 – Input data of Product Manager   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Reason to reject |  | Yes | Text |  |  1. **Output data**   Table 2 - Output data of general information of order and transaction information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | | 1. | Order ID |  |  | 1 | | 2. | Transaction date |  | Timestamp | YY-MM-DDTHH:MM:SS | | 3. | Order Type |  |  | Normal or Rush | | 4. | List of products in order |  |  | Product name | | 5. | Total amount |  | * Comma for thousands separator * Positive integer * Right alignment | 1,283,004 | | 6. | Currency |  |  | VND |  1. **Postconditions**   If rejection is successful, the status of selected order is updated. If rejection is fail, no changes are made. |

## Use case “Create users”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Create users”**   1. **Use case code**   UC011   1. **Brief Description**   This use case describes how Administrator creates a new user account.   1. **Actors**    1. **Administrator** 2. **Preconditions**   Admin must be logged in.   1. **Basic Flow of Events** 2. Admin navigates to the Create User screen 3. Admin enters user details (see Table 1) 4. Software validates the input 5. Software creates the new user account 6. Software displays a success message to Administrator 7. Software sends email to the new user, notifies that the account has been created successfully 8. **Alternative flows**   Table A1 - Alternative flows of events for UC “Create users”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 3 | Email is already in use | Software notifies admin | Resumes at Step 2 | | 2. | At Step 3 | Invalid input (e.g., weak password) | Software prompts for correction | Resumes at Step 2 |  1. **Input data**   Table 1 - Input data of Administrator   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Full name | User's full name | Yes | Non-empty | Viet Khanh Ha | |  | Email | User's email address | Yes | Valid email format | Havietkhanh49@gmail.com | |  | Role | User’s role in the system | Yes | Must in a list | Adminstrator | |  | Password | User's initial password | Yes | Have at least 8 character | 123456789 |  1. **Output data**   Table 2 - Output data of user’s information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | User ID | Unique for user | Text | U1234 | |  | Registration Date | Date when the account was created | dd/mm/yy | 04/09/2024 | |  | Confirmation Notify | State message | Text | Successful |  1. **Postconditions**   If the account is successfully created, an email verification is sent, and the user can log in after verification. If validation fails or the email is not verified, the account remains inactive. |

## Use case “View homepage”

|  |
| --- |
| **Use Case “View homepage”**   1. **Use case code**   UC012   1. **Brief Description**   This use case describes how a customer interacts with the software to view a list of random products on the homepage.   1. **Actors**    1. **User** 2. **Preconditions** 3. **Basic Flow of Events** 4. The customer opens the application. 5. The system displays a list of 20 random products on the homepage. 6. **Alternative flows** 7. **Input data** 8. **Output data**   A list of 20 randomly selected products.   1. **Postconditions**   The homepage is displayed with product information. |

## Use case “Log in”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Log in”**   1. **Use case code**   UC013   1. **Brief Description**   This use case describes how an admin or product manager logs in to the system to access privileged features.   1. **Actors**    1. **Administrator**    2. **Product Manager** 2. **Preconditions** 3. **Basic Flow of Events** 4. The user enters login credentials. 5. The system validates the input. 6. The system checks the email and password with the database. 7. The user is redirected to the corresponding dashboard. 8. **Alternative flows**   Table A1 - Alternative flows of events for UC “Log in”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | Invalid input | System displays input error message and asks user to enter information again | Resumes at Step 1 | | 2. | At Step 3 | Email/password is incorrect | System displays authentication error message and asks user to enter information again | Resumes at Step 1 |  1. **Input data**   Table 1 - Input data of Administrator/Product Manager   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Email | User's email address | Yes | Valid email format | Havietkhanh49@gmail.com | |  | Password | User's initial password | Yes | Have at least 8 character | 123456789 |  1. **Output data** 2. **Postconditions**   The user is authenticated and can access the system with proper permissions if the information is correct. Otherwise, nothing happened. |

## Use case “Search products”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Search products”**   1. **Use case code**   UC014   1. **Brief Description**   This use case describes how the user searches or filters products by title, price, or category.   1. **Actors**    1. **User** 2. **Preconditions**   The user must in the session.   1. **Basic Flow of Events** 2. The customer enters a search query or selects filters. 3. The system validates and interprets the search/filter request. 4. The system queries the database for matching products. 5. The system returns a list of matching products.   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “Search products”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 1 | Search input is empty | System returns full list of products or random products | Use case ends | | 2. | At Step 3 | No products match query | System displays message indicating that no products were found | Use case ends |  1. **Input data**   Table 1 - Input data of User   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Title | Keyword to search product title | No |  | Harry Potter | |  | Category | Category to filter products | No | Must be in category list | Book | |  | Sort by | Field to sort | No | Title/price | A to Z | |  | Sort direction | Sorting direction | No | Asc or desc | asc |  1. **Output data**   A list of matching products.   1. **Postconditions**   The homepage is displayed with matching product informations. |

## Use case “Add products to cart”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Add products to cart”**   1. **Use case code**   UC015   1. **Brief Description**   This use case describes how a user adds one or more products to their shopping cart.   1. **Actors**    1. **User** 2. **Preconditions**   The user must in the session.   1. **Basic Flow of Events** 2. The user selects a product from the product list or product detail page 3. The user specifies the desired quantity 4. The user clicks the “Add to Cart” button and enters the quantity 5. The system validates product availability and quantity 6. The system adds the product to the shopping cart 7. The system confirms the action and updates the cart view   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “Add products to cart”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 4 | Product is out of stock | System shows error message indicating product is unavailable | Use case ends | | 2. | At Step 4 | Quantity exceeds available stock | System shows error message indicating insufficient stock | Use case ends |  1. **Input data**   Table 1 - Input data of User   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Product ID | Identifier of product to be added | Yes | Integer | 21 | |  | Quantity | Number of units to add | Yes | Integer | 10 |  1. **Output data** 2. **Postconditions**   The selected product is stored in the user’s current shopping cart session. |

## Use case “Update products in cart”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Update products in cart”**   1. **Use case code**   UC016   1. **Brief Description**   This use case describes how a user updates the quantity of products already in the shopping cart.   1. **Actors**    1. **User** 2. **Preconditions**   The user must be in the session and must have at least one product in the cart.   1. **Basic Flow of Events** 2. The user accesses the shopping cart. 3. The user modifies the quantity of one or more products in the cart. 4. The system validates the new quantity. 5. The system updates the quantity of the selected product(s) in the cart. 6. The system confirms the update and refreshes the cart view.   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “Update products to cart”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 3 | New quantity is 0 | System removes the product from the cart | At Step 5 | | 2. | At Step 3 | New quantity exceeds stock | System shows error message about insufficient stock | Use case ends |  1. **Input data**   Table 1 - Input data of User   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Product ID | Identifier of product to be added | Yes | Integer | 21 | |  | New quantity | Updated number of units | Yes | Integer | 10 |  1. **Output data** 2. **Postconditions**   The shopping cart reflects the new quantity for the selected product(s), or the product is removed if quantity is 0. |

## Use case “View products in cart”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “View products in cart”**   1. **Use case code**   UC017   1. **Brief Description**   This use case describes how the user views the list of products currently in their shopping cart.   1. **Actors**    1. **User** 2. **Preconditions**   The user must be in the session.   1. **Basic Flow of Events** 2. The user navigates to the shopping cart. 3. The system retrieves the current cart content. 4. The system displays the list of products with quantity, price, and total.   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “View products to cart”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | Cart is empty | System displays a message: "Your cart is empty" | Use case ends |  1. **Input data** 2. **Output data**   Table 2 - Output data of cart information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | |  | Product Name | Name of the product in cart | Text | "Effective Java - Programming Guide" | |  | Product Image | Product thumbnail image | Image | Product photo | |  | Unit Price | Price per individual item | Currency format | VND 450,000 | |  | Quantity | Number of items selected | Numeric | 2 | |  | Subtotal | Price × Quantity for each item | Currency format | VND 900,000 | |  | Total Amount | Sum of all subtotals | Currency format | 2,250,000 | |  | Currency | Price currency | Text | VND |  1. **Postconditions**   The cart content is displayed to the user. |

## Use case “Delete products in cart”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “Delete products in cart”**   1. **Use case code**   UC018   1. **Brief Description**   This use case describes how the user removes products from their shopping cart.   1. **Actors**    1. **User** 2. **Preconditions**   The user must be in the session and must have at least one product in the cart.   1. **Basic Flow of Events** 2. The user opens the cart. 3. The user selects one or more products to remove. 4. The user confirms the deletion. 5. The system removes the selected products from the cart. 6. The system confirms and updates the cart view.   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “Delete products in cart”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | No product selected | System shows message to select a product | Use case ends |  1. **Input data**   Table 1 - Input data of User   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** | |  | Product ID | Identifier of product to be deleted | Yes | Must exist in cart | 21 |  1. **Output data** 2. **Postconditions**   The selected product(s) are no longer in the cart. |

## Use case “View pending orders”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “View pending orders”**   1. **Use case code**   UC019   1. **Brief Description**   This use case describes the interaction between the product manager and software when the product manager wants to view pending orders.   1. **Actors**    1. **Product Manager** 2. **Preconditions**   The Product Manager must be logged in.   1. **Basic Flow of Events** 2. The Product Manager navigates to the order management screen. 3. The system retrieves all pending orders 4. The system displays a list of pending orders with details   **5. Alternative flows**  Table A1 - Alternative flows of events for UC “View pending orders”   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Location** | **Condition** | **Action** | **Resume location** | | 1. | At Step 2 | No pending orders exist | System displays message: "No pending orders found" | Use case ends |  1. **Input data** 2. **Output data**   Table 2 - Output data of pending orders information   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | | 1. | Order ID | Unique identifier for order | Text | #12345 | | 2. | Order Date | Date when order was created | dd/mm/yyyy HH:mm | 04/09/2024 14:30 | | 3. | Order Type | Rush or Normal order | String | Rush, Normal | | 4. | Total Amount | Total price with currency | Currency format | VND 1,500,000 | | 5. | Product List | List of ordered products | Table | Product details with quantity and price | | 6. | Rush Delivery Time | Expected delivery time for rush orders | dd/mm/yyyy HH:mm | 05/09/2024 10:00 |  1. **Postconditions**   The Product Manager sees a list of all pending orders and the Product Manager can proceed to approve or reject orders. |

# Supplementary specification

Supplementary specifications are additional details that complement the main Software Requirements Specification (SRS). They often include information about external interfaces, performance requirements, quality attributes, design constraints, and other supplementary details that are necessary for a comprehensive understanding of the software system.

## Functionality

* The software should validate all user inputs for correct data formats and fill required fields.
* The software should maintain a record of product modifications, order transactions, and administrative activities for audit trails to support traceability and compliance

## Usability

* AIMS software must be of an intuitive layout so that new users can familiarize themselves with ease.
* The functions need to be designed for ease of operation.
* The system must have tooltips and help documentation for essential features.

## Reliability

* The system should run 24/7 and support up to 1,000 customers concurrently without degradation in performance.
* AIMS should run for at least 300 hours continuously before maintenance is needed.
* In the event of failure, the system should be back to normal within 1 hour.
* Ordering and payment processing should be atomic to avoid inconsistencies in data.

## Performance

* Response time should not exceed 2 seconds per action under normal conditions. During peak hours, response time should not exceed 5 seconds.
* Searching for products should produce results in 2 seconds.
* Processing an order, from checkout to payment confirmation, should complete within 10 seconds.
* The system should be able to handle product stock checks effectively, updating real-time inventory.

## Supportability

* The software should support database backups and restoration.
* At the same time, whenever customers need to upgrade or maintain any module, the development team will support them.

## Other requirements

* Payment transactions must be securely processed via VNPay.
* Shipping fees should be dynamically calculated based on product weight and customer location.
* Rush order delivery should only be available for eligible products within Hanoi's inner districts.
* Refunds for canceled/rejected orders must be processed via VNPay within a predefined timeframe.