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

|  |
| --- |
| **FPT UNIVERSITY** |
| Capstone Project Document |
| Blind Box System |

|  |  |
| --- | --- |
| **Group 03** | |
| **Group Members** | <Nguyễn Đức Huấn><SE170396 >  <Lê Minh Tân>< SE170587>  <Võ Đăng Khoa>< SE170395 >  <Nguyễn Khánh Linh><SE150873 > |
| **Supervisor** | Nguyễn Ngọc Lâm. |
| **Ext Supervisor** |  |
| **Capstone Project code** | Blind Box System |

- Ho Chi Minh, <03 >/<2025 > -

**Table of Contents**

[Definition and Acronyms 3](#_Toc194043520)

[I. Project Introduction 4](#_Toc194043521)

[1. Overview 4](#_Toc194043522)

[2. Product Background 4](#_Toc194043523)

[3. Existing Systems 6](#_Toc194043524)

[II. Project Management Plan 11](#_Toc194043528)

[1. Overview 11](#_Toc194043529)

[2. Management Approach 11](#_Toc194043530)

[3. Project Deliverables 11](#_Toc194043531)

[4. Responsibility Assignments 11](#_Toc194043532)

[5. Project Communications 11](#_Toc194043533)

[6. Configuration Management 11](#_Toc194043534)

[III. Software Requirement Specification 12](#_Toc194043535)

[1. Product Overview 12](#_Toc194043536)

[2. User Requirements 12](#_Toc194043537)

[3. Functional Requirements 12](#_Toc194043538)

[4. Non-Functional Requirements 12](#_Toc194043539)

[5. Requirement Appendix 12](#_Toc194043540)

[IV. Software Design Description 13](#_Toc194043541)

[1. System Design 13](#_Toc194043542)

[2. Database Design 35](#_Toc194043543)

[3. Detailed Design 35](#_Toc194043544)

# Definition and Acronyms

*[Fill all the definitions, acronyms,… used within the document] in the table format as below]*

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| PWM | Psychology website |
| AWS | Amazon Web Services |
| BA | Business Analysis |
| BR | Business Rule |
| ERD | Entity Relationship Diagram |
| GUI | Graphical User Interface |
| PM | Project Manager |
| SDD | Software Design Description |
| SPMP | Software Project Management Plan |
| SRS | Software Requirement Specification |
| UAT | User Acceptance Test |
| UC | Use Case |
| API | Application Program Interface |

# I. Project Introduction

The Blind Box Shopping System is an e-commerce platform designed to enhance the excitement of shopping through a blind box model. It offers users the thrill of unboxing surprise items while ensuring a seamless and secure shopping experience.

This system integrates a dynamic shopping interface with gamified elements like digital unboxing, reward systems, and interactive content. It supports both customers and administrators in managing orders, inventory, and promotional campaigns.

By adopting modern technologies, the system delivers an engaging user experience and optimizes operational efficiency for administrators.

## 1. Overview

### 1.1 Project Information

Project Name: BlindBox Shopping System.

Project Code:

* Server-side: C#
* Client-side: React
* Database: MySql

Group number: group 3

### 1.2 Project Team

**Project Manager:** Nguyễn Ngọc Lâm.

**Lead Developer:** Nguyễn Đức Huấn.

**Developer:** Nguyễn Khánh Linh, Võ Đăng Khoa, Lê Minh Tân.

## 2. Product Background

**2. 1. Product Overview**

* **Product Name**: Blind Box (Baby Tree, Labubu)
* **Product Type**: Surprise Toys (Blind Box)
* **Material**: Safe, non-toxic plastic, environmentally friendly.
* **Design**: Each box contains a random character with unique and adorable designs.

**2. 2. Product Story**

* **Inspiration**: The Blind Box concept was created to bring unexpected joy to players. Each box is a delightful surprise, sparking curiosity and excitement upon opening.
* **Featured Characters**: Baby Tree and Labubu are the standout characters in this product line. Baby Tree embodies the charm of young trees, while Labubu is an adorable bear with a variety of expressions.

**2.3. Key Features**

* **Random Surprise**: Each Blind Box contains a randomly selected character, offering an element of delightful unpredictability.
* **Collectible**: Players can collect different characters to build a unique and diverse collection.
* **High Quality**: Made from safe, durable materials, suitable for both children and adults.

**2.4. Benefits**

* **Entertainment**: Brings joy and excitement when opening the box, creating memorable moments.
* **Encourages Collecting**: Inspires players to build their collections and connect with others who share similar interests.
* **Skill Development**: Helps children enhance thinking and creativity skills through play and collecting activities.

**2.5. Target Audience**

* **Primary Audience**: Children aged 5 and above, as well as collectors and toy enthusiasts.
* **Customer Psychology**: Appeals to those who enjoy surprises, love collecting, and seek unique items.

**2.6. Market Positioning**

* **Unique Selling Proposition (USP)**: The Blind Box is more than just a toy—it's an exciting experience that connects collectors and fosters a vibrant community.

**2.7. Customer Testimonials**

* Include positive reviews from satisfied customers, emphasizing the joy and fulfillment the product brings.

**2.8. Call to Action**

* Encourage customers to explore the Blind Box collection, join the collector community, and share their experiences. For instance: “Start your collection today! Visit [website link] to grab your Blind Box now!”

## 3. Existing Systems

3.1. E-commerce platforms, such as Shopee, Tiki, and Lazada

3.2. PopMart ( Link: [PopMart Link](https://www.popmart.com/) )

* Reference UI/UX elements from PopMart to craft visually appealing and functional interfaces for my system.
* Emphasize intuitive categorization and information hierarchy for better navigation and usability.
* Enhance user experience with interactive features like animations, hover effects, and smooth transitions.

## 4. Business Opportunity

The Blind Box (Baby Tree, Labubu) e-commerce platform addresses a rapidly growing niche in the global collectibles market, driven by Gen Z and millennial demand for surprise-driven, gamified shopping experiences. Below is a breakdown of the market opportunity, competitive landscape, and strategic value of the proposed system:

### 4.1. Market Opportunity

### A. Growing Collectibles & Toy Market

#### Market Size: The global collectibles market is expected to expand significantly, reaching $630 billion by 2033 with a compound annual growth rate (CAGR) of 5.2%. Asia-Pacific is predicted to lead this growth, driven by popular trends such as Blind Boxes and gacha culture.

#### Gen Z & Millennial Demand: More than 70% of Gen Z consumers prioritize "unique experiences" over generic products. The Blind Box, with its element of surprise and exclusivity, aligns perfectly with their preferences, making it a highly attractive choice.

#### Digital Shift: Following the pandemic, 65% of collectible purchases now occur online, fueling the need for dedicated and specialized e-commerce platforms to cater to this growing market.

#### B. Untapped Gaps in Existing Solutions

Current platforms (e.g., Amazon, AliExpress, niche toy stores) lack:

* Gamified Engagement: Limited integration of reward systems (e.g., spins, loyalty tiers) to drive repeat purchases.
* Community Features: No dedicated spaces for collectors to trade duplicates, share unboxing videos, or discuss trends.
* Trust & Authenticity: Counterfeit Blind Boxes plague marketplaces like eBay, eroding consumer trust.
* Personalization: Most platforms don’t offer subscription models or tailored recommendations for collectors.

### 2. Business Problem Being Solved

The platform solves three core challenges:

2.1Low Engagement: Traditional e-commerce lacks interactivity. Users buy once but rarely return.

1. Fragmented Community: Collectors rely on third-party apps (e.g., Discord, Reddit) to connect, creating friction.
2. Counterfeit Risk: Buyers struggle to verify authenticity, especially for rare Labubu/Baby Tree figures.

## 5. Software Product Vision

"To redefine the joy of collectibles by creating a world where every Blind Box unboxing is a gateway to discovery, connection, and trust. Our platform empowers collectors of Baby Tree, Labubu, and beyond to experience the thrill of surprise while fostering a global community built on authenticity, gamified engagement, and shared passion. By blending cutting-edge technology with human-centric design, we aim to become the trusted heartbeat of the collectibles revolution—where every user feels valued, every purchase sparks delight, and every rare find tells a story."

### Key Elements of the Vision:

1. Purpose & Intent:
   * Transform Blind Box shopping from a transactional experience into an immersive journey of discovery and community.
2. Future Worldview:
   * Collectors seamlessly connect, trade, and celebrate rare finds in a secure, gamified ecosystem.
   * Counterfeit-free markets and AR-powered unboxing elevate trust and excitement.
   * Brands and creators thrive through collaborations, while users enjoy personalized, sustainable collectibles.
3. Balanced Realities:
   * Customer Needs: Combines surprise (randomized boxes), security (blockchain authentication), and social connection (community hubs).
   * Organizational Goals: Drives scalable revenue via subscriptions, peer-to-peer trading fees, and premium partnerships.
   * Market Trends: Aligns with Gen Z’s demand for digital-physical hybrid experiences and eco-conscious consumption.
   * Feasibility: Leverages existing tech (Shopify-like storefronts, Discord-style forums) while innovating in AR and blockchain.

## 6. Project Scope & Limitations

**6.1. Core Platform Features**

* **Blind Box Listings**: Dedicated product pages showcasing Baby Tree, Labubu, and other series, with randomized item displays to maintain the thrill of surprise.
* **Gamified Purchases**: Interactive purchase options, including virtual spin-to-win rewards, loyalty tiers, and exclusive limited-edition drops to engage users.
* **Community Hub**: Integrated forums, trading boards, and user-generated content (UGC) feeds, enabling unboxing video sharing and fostering community connections.
* **Authentication System**: Blockchain-powered verification for rare collectibles (e.g., Labubu exclusives), ensuring product authenticity and building user trust.

**6.2. User Roles & Permissions**

* **Buyers**: Individuals purchasing Blind Boxes and collectibles.
* **Sellers**: Approved partners responsible for offering collectibles and managing listings.
* **Moderators**: Community managers tasked with overseeing forums, resolving disputes, and maintaining a secure user environment.

**6.3. Monetization**

* **Subscription Plans**: Monthly bundles featuring curated Blind Boxes to incentivize recurring engagement.
* **Transaction Fees**: Fees applied to peer-to-peer trading activities, driving sustainable revenue growth.

**6.4. Technical Infrastructure**

* **Mobile-First Responsive Design**: A web-based platform optimized for mobile users, with no native application development planned for Phase 1.
* **Payment Gateway Integration**: Seamless connections to third-party payment services like Stripe and PayPal, ensuring secure transactions.

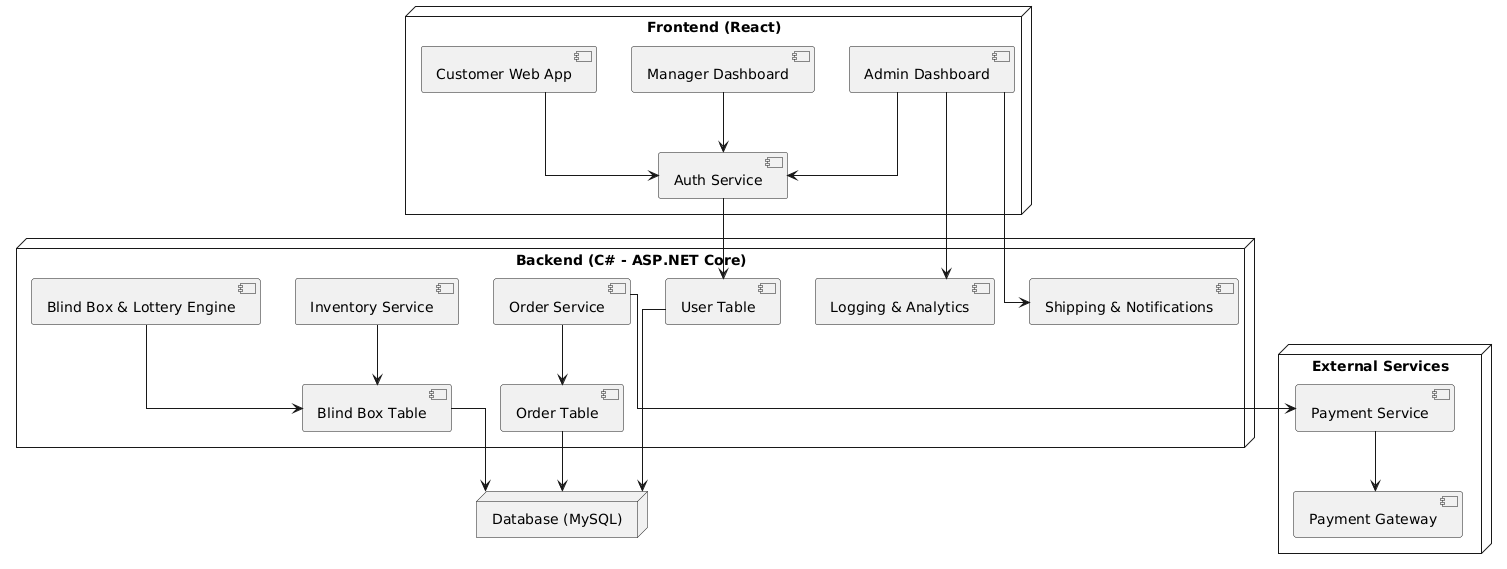
**6.5. Geographic Focus**

* **Initial Launch**: Focused rollout in Southeast Asia, starting with Vietnam and Thailand.
* **Global Expansion:** Progressive scaling into international markets following the successful Pilot phase.

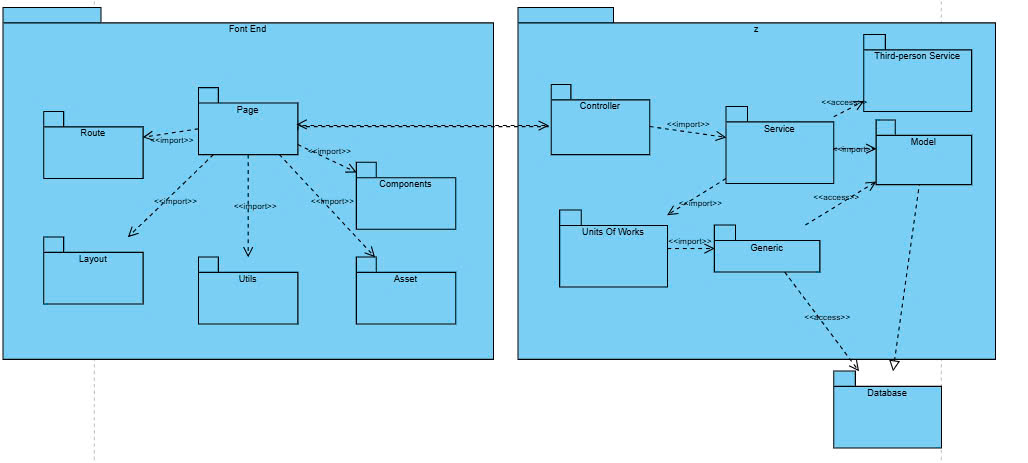
# II. Software Design Description

## 1. System Design

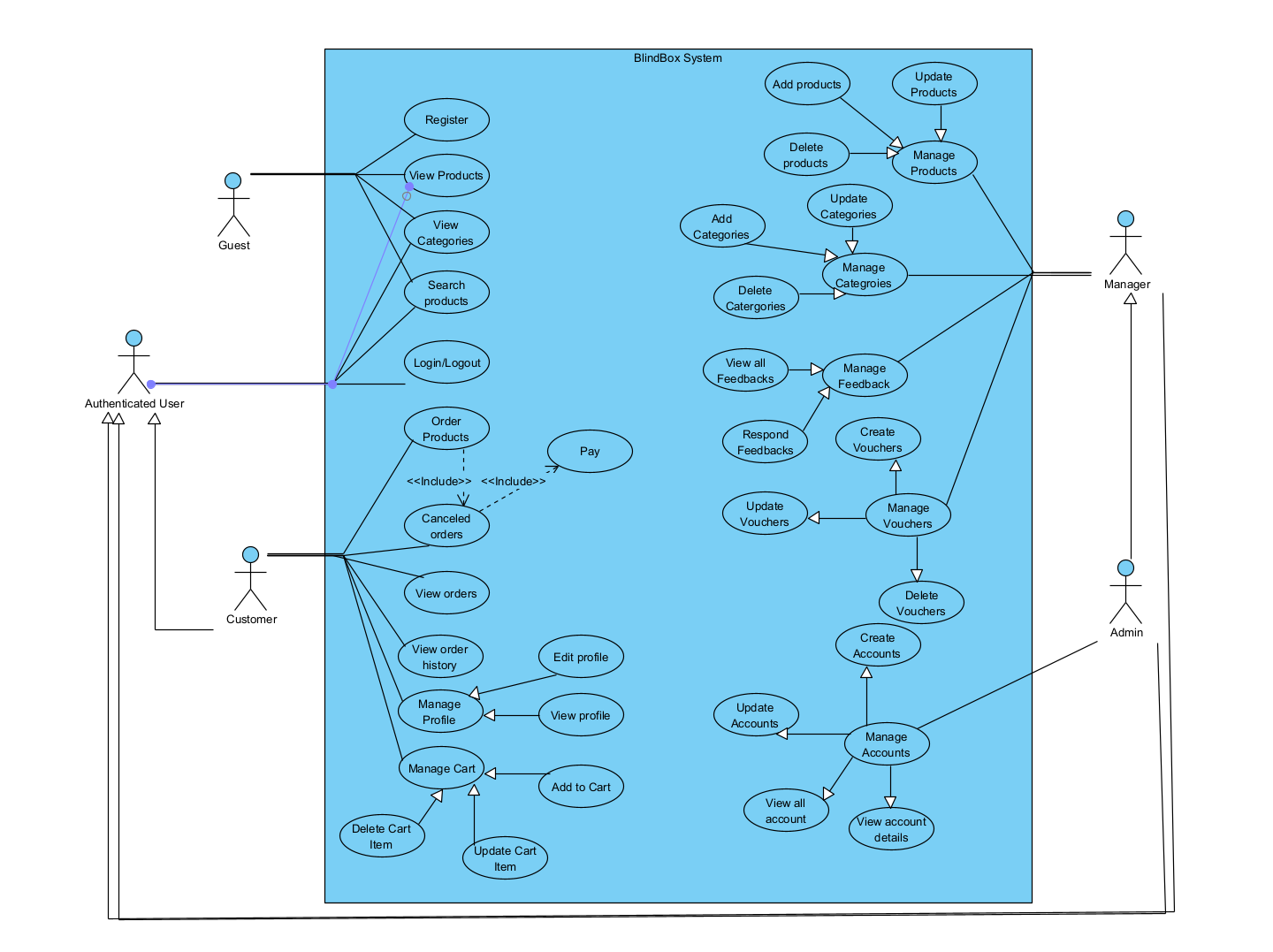
### 1.1 System Architecture

**

### 1.2 Package Diagram



**1.3 UseCase Diagram**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 1** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Guest | **Secondary Actors:** | None |
| **Trigger:** | A guest wants to create an account. | | |
| **Description:** | This use case allows a new user to register an account in the system. | | |
| **Preconditions:** | 1. The user must be on the registration page.  2. The user must provide valid information (email, password, name, etc.). | | |
| **Postconditions:** | 1. A new account is created and stored in the database.  2. The user receives a confirmation email.  3. The user can log in using their credentials. | | |
| **Normal Flow:** | 1. The user navigates to the "Register" page. 2. The user fills in the required information. 3. The user clicks the "Sign Up" button. 4. The system validates the input. 5. If valid, the system creates the user account. 6. A confirmation email is sent to the user. 7. The system displays a success message. | | |
| **Alternative Flows:** | 1. If the email is already registered:   * The system displays an error message.   2. If the password is too weak:   * The system prompts the user to use a stronger password. | | |
| **Exceptions:** | If the email confirmation fails:   * The user cannot log in until verification is complete. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily | | |
| **Business Rules:** | 1. The email must be unique. 2. The password must meet security requirements. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The user has internet access. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 2: Edit Product** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Manager | **Secondary Actors:** | None |
| **Trigger:** | The Manager selects a product from the product list to edit its details. | | |
| **Description:** | This use case allows the Manager to modify the details of an existing product. | | |
| **Preconditions:** | 1. The Manager must be logged into the system. 2. The product to be edited must already exist in the database. | | |
| **Postconditions:** | 1. The product details are updated in the database. 2. The updated product information is reflected in the product list. 3. A success notification is displayed to the Manager. | | |
| **Normal Flow:** | 1. The Manager navigates to the "Product Management" page. 2. The Manager selects a product from the list to edit. 3. The system displays a form with the product's existing details (name, price, description, category, size, color,…). 4. The Manager modifies the desired fields. 5. The Manager clicks the "Save" button. 6. The system validates the input, updates the product in the database, and displays a success message. | | |
| **Alternative Flows:** | 1. If the Manager inputs invalid details (missing required fields or incorrect formats):  * The system highlights errors and displays appropriate messages.  1. If the selected category does not exist:  * The system prompts the Manager to create a category first.  1. If the database connection fails:  * The system displays an error message and logs the issue. | | |
| **Exceptions:** | 1. The updated product name conflicts with another product in the same category:  * The system displays a warning message and prevents duplication.  1. Missing required fields:  * The system displays validation messages and prevents submission. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily or as needed to manage products. | | |
| **Business Rules:** | 1. The product name must remain unique within its category. 2. The price must be a positive value. 3. Products cannot be edited if they are linked to ongoing transactions. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Manager has permission to edit products. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 3: Purchase Blind Box** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The customer wants to purchase a blind box. | | |
| **Description:** | This use case allows the customer to buy a blind box from the store. | | |
| **Preconditions:** | 1. The user must be logged in. 2. The user must have a sufficient wallet balance or valid payment method | | |
| **Postconditions:** | 1. The order is successfully created and recorded. 2. Payment is processed. 3. The inventory is updated. | | |
| **Normal Flow:** | 1. The user navigates to the "Blind Box Store" page. 2. The user selects a blind box. 3. The system displays the product details. 4. The user clicks "Buy Now". 5. The system prompts the user to confirm payment. 6. The payment is processed. 7. The system creates an order and updates inventory. 8. The user receives a confirmation | | |
| **Alternative Flows:** | 1. If the user has insufficient balance:  * The system prompts the user to add funds.  1. If the payment fails:  * The system displays an error message. | | |
| **Exceptions:** | If the blind box is out of stock:   * The system prevents purchase and notifies the user. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily or as needed to manage products. | | |
| **Business Rules:** | A user cannot buy more than the available stock. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The user has a valid payment method. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 4: Apply Voucher** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The customer enters a voucher code during checkout. | | |
| **Description:** | This use case allows the Manager to modify the details of an existing product. | | |
| **Preconditions:** | 1. The customer must be logged into the system. 2. The voucher must be valid and active. | | |
| **Postconditions:** | 1. The discount is applied to the order. 2. The voucher usage is recorded in the system | | |
| **Normal Flow:** | 1. The customer navigates to the checkout page. 2. The customer enters a voucher code. 3. The system validates the voucher. 4. The system applies the discount if the voucher is valid. 5. The system updates the order total. 6. The system displays the final price with the applied discount. | | |
| **Alternative Flows:** | 1. If the voucher is expired or invalid:  * The system notifies the customer and prevents usage.  1. If the voucher has already been used:  * The system displays an error message. | | |
| **Exceptions:** | If the database connection fails:   * The system logs the issue and notifies the customer. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | Occasionally | | |
| **Business Rules:** | 1. Vouchers must have a validity period. 2. Some vouchers may have a minimum purchase requirement. 3. A voucher cannot be reused once redeemed. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | Voucher expired | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 5: Purchase Blind Box** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The Customer selects a blind box and proceeds to purchase it. | | |
| **Description:** | This use case allows a Customer to purchase a blind box and complete the transaction. | | |
| **Preconditions:** | 1. The Customer must be logged into the system. 2. The selected blind box must be available in stock. 3. The Customer must have a valid payment method. | | |
| **Postconditions:** | 1. The system deducts the purchase amount from the Customer's wallet or processes an external payment. 2. The order is recorded in the database. 3. The Customer receives a confirmation of the purchase. | | |
| **Normal Flow:** | 1. The Customer browses the available blind boxes. 2. The Customer selects a blind box and clicks "Buy Now." 3. The system displays the order summary, including price, discount (if any), and total amount. 4. The Customer chooses a payment method (wallet balance or external payment). 5. The Customer confirms the purchase. 6. The system processes the payment. 7. If the payment is successful, the system records the transaction and order details. 8. The system updates inventory and marks the blind box as sold. 9. The Customer receives a confirmation notification. | | |
| **Alternative Flows:** | 1. If the blind box is out of stock:  * The system notifies the Customer and prevents the purchase.  1. If the payment fails:  * The system displays an error message and prompts the Customer to retry.  1. If the Customer's wallet balance is insufficient:  * The system suggests using an external payment method or adding funds. | | |
| **Exceptions:** | 1. The Customer cancels the purchase before confirming:  * The system returns to the product selection page.  1. The database connection fails during payment processing:  * The system logs the issue and displays an error message. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily or as needed. | | |
| **Business Rules:** | 1. Customers cannot purchase more than the available quantity. 2. The system must apply any applicable vouchers or discounts before payment | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Customer has a valid payment method linked.The Customer understands that blind boxes are non-refundable once purchased. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 6: Manage Inventory** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Manager | **Secondary Actors:** | None |
| **Trigger:** | The Manager wants to update the inventory details. | | |
| **Description:** | This use case allows the Manager to add, update, or remove inventory items, ensuring the system has accurate stock data. | | |
| **Preconditions:** | 1. The Manager must be logged into the system. 2. The product or blind box must already exist in the database. | | |
| **Postconditions:** | 1. The inventory details are updated in the system. 2. Stock availability is correctly reflected in the database. 3. A success message is displayed to the Manager. | | |
| **Normal Flow:** | 1. The Manager navigates to the "Inventory Management" page. 2. The Manager selects an inventory item to update or chooses to add a new item. 3. The system displays the item details, including quantity, status, and linked products. 4. The Manager updates the necessary fields (e.g., stock quantity, availability status). 5. The Manager clicks the "Save" button. 6. The system validates the input and updates the inventory database. 7. A confirmation message is displayed. | | |
| **Alternative Flows:** | * If the Manager enters invalid details (e.g., negative stock quantity):   + The system highlights errors and prompts for correction. * If the product linked to the inventory item is missing:   + The system prompts the Manager to add or link a valid product. * If the database connection fails:   + The system displays an error message and logs the issue | | |
| **Exceptions:** | * Attempting to delete an inventory item linked to active orders:   + The system prevents deletion and displays a warning message. * Insufficient permissions:   + The system denies access and notifies the user. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily or as needed. | | |
| **Business Rules:** | 1. Stock cannot be set to a negative value. 2. Inventory items linked to active orders cannot be removed. 3. Changes must be logged for auditing purposes. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Manager has the necessary permissions to modify inventory.The system enforces data validation rules. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 7: Apply Voucher to Order** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The Customer applies a voucher during the checkout process. | | |
| **Description:** | This use case allows a Customer to use a valid voucher to get a discount on their order | | |
| **Preconditions:** | 1. The Customer must be logged into the system. 2. The Customer must have at least one valid voucher. 3. The order must meet the voucher’s eligibility criteria. | | |
| **Postconditions:** | 1. The system applies the voucher discount to the total order amount. 2. The voucher usage is recorded and marked as redeemed if applicable. 3. The Customer sees the updated total cost reflecting the applied discount. | | |
| **Normal Flow:** | 1. The Customer proceeds to the checkout page. 2. The Customer selects a voucher from the available list. 3. The system validates the voucher. 4. If valid, the system applies the discount to the order. 5. The system updates the order summary with the new total price. 6. The Customer proceeds with payment. | | |
| **Alternative Flows:** | 1. If the voucher has expired:    * The system displays an error message and prevents its use. 2. If the voucher usage limit has been exceeded:    * The system notifies the Customer that the voucher is no longer valid. 3. If the order does not meet the voucher’s minimum purchase requirement:    * The system informs the Customer and prevents application. | | |
| **Exceptions:** | 1. The Customer removes the voucher before finalizing the payment:    * The system recalculates the order total without the discount. 2. The system experiences a failure during voucher validation:    * The system logs the issue and displays an error message. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed during checkout. | | |
| **Business Rules:** | 1. A voucher can only be used if it meets all applicable criteria. 2. The system must track voucher usage and prevent duplicate applications.. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Customer understands that vouchers may have expiration dates and conditions.The Customer has an active order that meets voucher requirements. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 8: Provide Feedback** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The Customer submits feedback on a purchased blind box.. | | |
| **Description:** | This use case allows a Customer to provide feedback and rate a purchased blind box. | | |
| **Preconditions:** | 1. The Customer must be logged into the system. 2. The Customer must have previously purchased a blind box. 3. The feedback submission page must be accessible. | | |
| **Postconditions:** | 1. The feedback is stored in the database. 2. The blind box rating is updated based on submitted feedback. 3. A confirmation message is displayed to the Customer. | | |
| **Normal Flow:** | 1. The Customer navigates to the "Order History" page. 2. The Customer selects a previously purchased blind box. 3. The system displays the feedback submission form. 4. The Customer enters a title, feedback content, rating, and optionally uploads an image. 5. The Customer clicks the "Submit" button. 6. The system validates the input and saves the feedback. 7. The system updates the blind box's average rating. 8. A success message is displayed to the Customer. | | |
| **Alternative Flows:** | 1. If the Customer submits feedback with missing required fields:    * The system highlights errors and prompts for correction. 2. If the Customer attempts to provide feedback for an unpurchased blind box:    * The system prevents submission and displays an error message. 3. If the database connection fails:    * The system logs the issue and informs the Customer. | | |
| **Exceptions:** | 1. The Customer attempts to submit multiple feedback entries for the same blind box:    * The system prevents duplicate submissions. 2. The Customer uploads an unsupported image format:    * The system rejects the upload and prompts for a valid file type. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed after purchases. | | |
| **Business Rules:** | 1. Customers can only submit feedback for purchased blind boxes. 2. Feedback must adhere to content guidelines (no offensive language, etc.). 3. The system must allow feedback deletion or editing within a certain timeframe. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Customer understands that feedback is visible to other users.The system enforces content moderation to ensure appropriate feedback. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 9: View Order History** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The Customer selects the "Order History" option to view past orders. | | |
| **Description:** | This use case allows a Customer to view details of previous orders, including order status, purchased items, and payment information. | | |
| **Preconditions:** | 1. The Customer must be logged into the system. 2. The Customer must have at least one completed order. 3. The order history page must be available. | | |
| **Postconditions:** | 1. The system displays a list of the Customer's past orders. 2. The Customer can view order details, including purchased items, total price, and status. 3. If applicable, the Customer can request support regarding an order. | | |
| **Normal Flow:** | 1. The Customer navigates to the "Order History" page. 2. The system retrieves and displays a list of past orders. 3. The Customer selects an order to view details. 4. The system displays order details, including order date, items, total cost, payment method, and status. 5. If the order is eligible for a refund or support request, relevant options are provided. | | |
| **Alternative Flows:** | 1. If the Customer has no past orders:    * The system displays a message stating "No order history found." 2. If the database connection fails:    * The system logs the issue and informs the Customer. 3. If the order details cannot be retrieved:    * The system displays an error message and prompts the Customer to try again later. | | |
| **Exceptions:** | The Customer tries to view another user's order history:   * The system denies access and logs the unauthorized attempt. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed to track orders. | | |
| **Business Rules:** | 1. Customers can only view their own order history. 2. Orders older than a certain period (e.g., 12 months) may be archived. 3. If an order is still being processed, certain details may be restricted. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Customer expects real-time updates on order statuses.The system provides options for requesting refunds or support. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 11: Track Order Status** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | The Customer wants to check the current status of their order. | | |
| **Description:** | This use case allows a Customer to view real-time updates on their order status. | | |
| **Preconditions:** | 1. The Customer must be logged into the system. 2. The Customer must have at least one placed order in the system. 3. The system must store order tracking information. | | |
| **Postconditions:** | 1. The order status is displayed to the Customer. 2. The Customer receives real-time updates if the order status changes. | | |
| **Normal Flow:** | 1. The Customer navigates to the "Order History" section. 2. The Customer selects a specific order to view its details. 3. The system retrieves and displays the order status (e.g., Processing, Shipped, Delivered). 4. If the order is in transit, the system displays tracking details if available. 5. The Customer can refresh the page or receive push notifications for status updates. | | |
| **Alternative Flows:** | 1. If the Customer has no orders:    * The system displays a message indicating no available order history. 2. If the order tracking information is temporarily unavailable:    * The system informs the Customer and suggests checking again later. | | |
| **Exceptions:** | 1. The Customer selects an order that no longer exists:    * The system displays an error message and removes the record if necessary. 2. The system experiences a failure in retrieving the order status:    * The system logs the issue and informs the Customer. | | |
| **Priority:** | Medium. | | |
| **Frequency of Use:** | Frequently, depending on order volume. | | |
| **Business Rules:** | 1. Only the Customer who placed the order can track its status. 2. Order status updates must be synchronized with the shipping provider if applicable. 3. Notifications must be sent when the order status changes. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | 1. The Customer understands order tracking functionalities. 2. The system has real-time synchronization for order updates. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 12: Manage Blind Box Listings** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Admin | **Secondary Actors:** | None |
| **Trigger:** | The Admin wants to update the role of a user. | | |
| **Description:** | This use case allows an Admin to manage blind box listings, including adding new boxes, updating details, and removing outdated or discontinued boxes. | | |
| **Preconditions:** | 1. The Admin must be logged into the system. 2. The Admin must have the necessary permissions to manage blind box listings. | | |
| **Postconditions:** | 1. The blind box listing is updated in the database. 2. Changes are reflected in the customer-facing store. 3. A success notification is displayed to the Admin. | | |
| **Normal Flow:** | 1. The Admin navigates to the "Blind Box Management" page. 2. The Admin selects an action: add a new blind box, edit an existing one, or remove a blind box. 3. If adding or editing:    * The system displays a form to input or modify blind box details (name, description, price, availability, images, etc.).    * The Admin enters the necessary details and clicks "Save."    * The system validates the input, updates the database, and confirms the update. 4. If removing:    * The Admin selects a blind box to delete.    * The system checks if the blind box is linked to active orders.    * If not, the system removes it from the database and confirms deletion. | | |
| **Alternative Flows:** | 1. If required fields are missing:    * The system highlights errors and requests corrections. 2. If the blind box is linked to active orders:    * The system prevents deletion and notifies the Admin. 3. If the database update fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The entered blind box name already exists:    * The system prevents duplication and notifies the Admin. 2. If image upload fails:    * The system allows retrying or proceeding without an image. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Regularly, depending on stock updates. | | |
| **Business Rules:** | 1. Blind box names must be unique. 2. A blind box cannot be deleted if it is linked to ongoing transactions. 3. Blind boxes must have a valid price greater than zero. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Admin has permission to manage blind boxes.The system validates blind box details before saving changes. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 13: Manage Customer Support Requests.** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Support Agent | **Secondary Actors:** | Customer |
| **Trigger:** | A customer submits a support request or inquiry. | | |
| **Description:** | This use case allows a Support Agent to manage customer support requests, including responding to inquiries and resolving issues. | | |
| **Preconditions:** | 1. The Support Agent must be logged into the system. 2. The Customer must submit a support request. 3. The system must log support tickets. | | |
| **Postconditions:** | 1. 1The customer's issue is addressed. 2. The support request status is updated. 3. The customer receives a response or resolution. | | |
| **Normal Flow:** | 1. The Customer submits a support request through the system. 2. The system logs the request and assigns it to an available Support Agent. 3. The Support Agent reviews the request details. 4. The Support Agent responds with a resolution or follow-up questions. 5. The system updates the request status and notifies the Customer. 6. If resolved, the request is closed. Otherwise, further actions are taken. | | |
| **Alternative Flows:** | 1. If no Support Agent is available:    * The system queues the request and notifies the Customer of potential delays. 2. If the issue cannot be resolved immediately:    * The Support Agent escalates the request to a higher-level team. | | |
| **Exceptions:** | 1. If the request is unclear:    * The system prompts the Customer for additional information before assigning it. 2. If the system fails to log the request:    * The Customer is notified, and an error is logged. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily, depending on customer inquiries. | | |
| **Business Rules:** | 1. Support Agents must respond within a specified timeframe. 2. Customer requests must be tracked until resolved. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | Customers have access to submit support requests.Support Agents have appropriate permissions to manage requests. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 14: Admin Manage User Accounts** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Admin | **Secondary Actors:** | None |
| **Trigger:** | The Admin wants to manage user accounts (create, update, deactivate users). | | |
| **Description:** | This use case allows the Admin to manage user accounts, including creating new users, updating details, and deactivating accounts when necessary. | | |
| **Preconditions:** | 1. The Admin must be logged into the system. 2. The system must have existing user data. | | |
| **Postconditions:** | 1. The user account information is updated in the database. 2. Deactivated users can no longer access the system. 3. A success message is displayed upon successful changes. | | |
| **Normal Flow:** | 1. The Admin navigates to the "User Management" page. 2. The Admin views the list of registered users. 3. The Admin selects a user to edit or deactivate. 4. The system displays the selected user’s details. 5. The Admin updates the necessary fields (name, role, status, etc.) or chooses to deactivate the account. 6. The Admin clicks the "Save" button. 7. The system validates the input, updates the user details, and displays a success message. | | |
| **Alternative Flows:** | 1. If the Admin tries to update a non-existing user:    * The system displays an error message. 2. If the Admin inputs invalid details (e.g., missing required fields or incorrect formats):    * The system highlights errors and prompts the Admin to correct them. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The Admin tries to deactivate their own account:    * The system prevents this action and displays a warning message. 2. The Admin tries to assign an invalid role:    * The system prevents submission and prompts a valid selection. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Regularly, as needed for user management. | | |
| **Business Rules:** | 1. Only Admin users can manage user accounts. 2. User email addresses must be unique in the system. 3. Deactivated users cannot log in or perform system actions. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Admin has sufficient privileges to modify user accounts.The system provides real-time validation for user management actions. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 15: User View Order History** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | User | **Secondary Actors:** | None |
| **Trigger:** | The User wants to view their past orders. | | |
| **Description:** | This use case allows the User to view their order history, including details such as order status, total amount, and items purchased. | | |
| **Preconditions:** | 1. The User must be logged into the system. 2. The User must have placed at least one order in the system. | | |
| **Postconditions:** | 1. The order history is displayed to the User. 2. The User can view order details, including order items, status, and total cost. | | |
| **Normal Flow:** | 1. The User navigates to the "Order History" page. 2. The system retrieves and displays a list of past orders. 3. The User selects an order to view details. 4. The system displays detailed information about the selected order, including items, status, and total amount. | | |
| **Alternative Flows:** | 1. If no orders exist for the User:    * The system displays a message indicating no past orders. 2. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The User is not logged in:   * The system redirects the User to the login page. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | Frequently, as Users check their order status and history. | | |
| **Business Rules:** | 1. Users can only view their own order history. 2. Order history must be accurate and updated in real time. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | 1. The User has an active account in the system. 2. The system maintains a proper record of past orders. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 16: Staff Manage Categories** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | The Staff wants to manage product categories (create, update, delete categories). | | |
| **Description:** | This use case allows the Staff to manage product categories, including adding new categories, modifying existing ones, and removing categories that are no longer needed. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. The system must have existing category data if updating or deleting. | | |
| **Postconditions:** | 1. The category information is updated in the database. 2. Deleted categories are removed unless they are associated with existing products. 3. A success message is displayed after a successful operation. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Category Management" page. 2. The Staff views the list of available categories. 3. The Staff chooses to create, update, or delete a category. 4. If creating, the Staff enters a new category name and submits the form. 5. If updating, the Staff modifies the category details and saves changes. 6. If deleting, the Staff confirms the deletion action. 7. The system validates the input and updates the database accordingly. 8. The system displays a success message. | | |
| **Alternative Flows:** | 1. If the Staff tries to delete a category linked to existing products:    * The system prevents deletion and notifies the Staff. 2. If the Staff inputs an invalid category name:    * The system displays a validation error message. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The Staff attempts to create a duplicate category:    * The system prevents submission and prompts the Staff to choose a unique name. 2. The Staff tries to modify a category name to an existing one:    * The system prevents duplication and prompts for a different name. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Regularly, as needed for category management. | | |
| **Business Rules:** | 1. Each category name must be unique. 2. Categories linked to active products cannot be deleted. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has sufficient privileges to manage categories.The system provides real-time validation for category management actions. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 17: Staff Process Customer Orders** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | A customer places an order, and the Staff needs to process it. | | |
| **Description:** | This use case allows Staff to process customer orders, including verifying order details, updating order status, and preparing orders for delivery or pickup. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. An order must exist in the system. | | |
| **Postconditions:** | 1. The order status is updated in the database. 2. The customer is notified about order progress. 3. The system logs the order processing action. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Order Management" page. 2. The Staff selects a pending order from the list. 3. The system displays the order details, including customer information, products, and payment status. 4. The Staff verifies the order details. 5. The Staff updates the order status (e.g., "Processing", "Shipped", "Completed"). 6. The system updates the database with the new order status. 7. The system sends a notification to the customer regarding the order update. 8. A success message is displayed to the Staff. | | |
| **Alternative Flows:** | 1. If the order payment is incomplete:    * The system prevents further processing and prompts the Staff to contact the customer. 2. If the system detects an out-of-stock item:    * The system notifies the Staff and suggests a replacement or order cancellation. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The Staff attempts to process an already completed order:    * The system prevents duplication and displays a warning. 2. The Staff tries to ship an order without verifying payment:    * The system prevents the action and prompts for verification. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily, as orders are placed. | | |
| **Business Rules:** | 1. Orders must be processed in chronological order. 2. Only authorized Staff can update order statuses. 3. Orders cannot be completed unless all items are in stock and payment is confirmed. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has permission to process orders.The system provides real-time order status updates. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 18: Staff Handle Customer Complaints** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | A customer submits a complaint regarding a product or service. | | |
| **Description:** | This use case allows Staff to handle customer complaints by reviewing complaints, responding to customers, and taking necessary actions such as issuing refunds or replacements. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. A customer complaint must exist in the system. | | |
| **Postconditions:** | 1. The complaint is marked as resolved or escalated. 2. The customer is notified of the resolution status. 3. If applicable, the system updates order or refund records. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Customer Complaints" page. 2. The Staff selects a pending complaint from the list. 3. The system displays complaint details, including customer information and complaint description. 4. The Staff reviews the complaint and determines an appropriate response. 5. The Staff responds to the customer with a resolution or requests additional information. 6. If applicable, the Staff processes a refund, replacement, or service correction. 7. The system updates the complaint status and notifies the customer. 8. A success message is displayed to the Staff. | | |
| **Alternative Flows:** | 1. If the complaint lacks sufficient details:    * The system prompts the Staff to request more information. 2. If the complaint requires managerial approval:    * The system escalates the issue to an Admin. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The Staff cannot resolve the complaint:    * The system allows escalation to higher management. 2. The complaint is found to be invalid:    * The system marks it as closed and notifies the customer. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed when complaints are received. | | |
| **Business Rules:** | 1. Complaints must be addressed within 48 hours. 2. Refunds must comply with store policies. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has access to customer complaints.The system provides tracking for complaint resolution. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 19: Staff Restock Inventory** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | The Staff identifies a low stock level for a product and decides to restock it. | | |
| **Description:** | This use case allows the Staff to update the inventory by adding new stock for a specific product. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. The product to be restocked must exist in the database. 3. The Staff has the necessary permissions to modify inventory. | | |
| **Postconditions:** | 1. The inventory count for the product is updated in the database. 2. A success notification is displayed to the Staff. 3. The updated stock level is reflected in the inventory list. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Inventory Management" page. 2. The Staff selects the product that needs restocking. 3. The system displays the current stock level and an input field to add new stock. 4. The Staff enters the quantity to be added and clicks the "Update Stock" button. 5. The system validates the input and updates the inventory count in the database. 6. A success message is displayed to the Staff. | | |
| **Alternative Flows:** | 1. If the entered quantity is negative or zero:    * The system displays a validation error message. 2. If the database connection fails:    * The system displays an error message and logs the issue. | | |
| **Exceptions:** | 1. The selected product is discontinued:   * The system prevents restocking and notifies the Staff. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Daily, depending on stock levels. | | |
| **Business Rules:** | 1. The stock quantity must be a positive integer. 2. Staff cannot restock discontinued products. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has access to inventory data.The system has real-time stock level monitoring. | | |

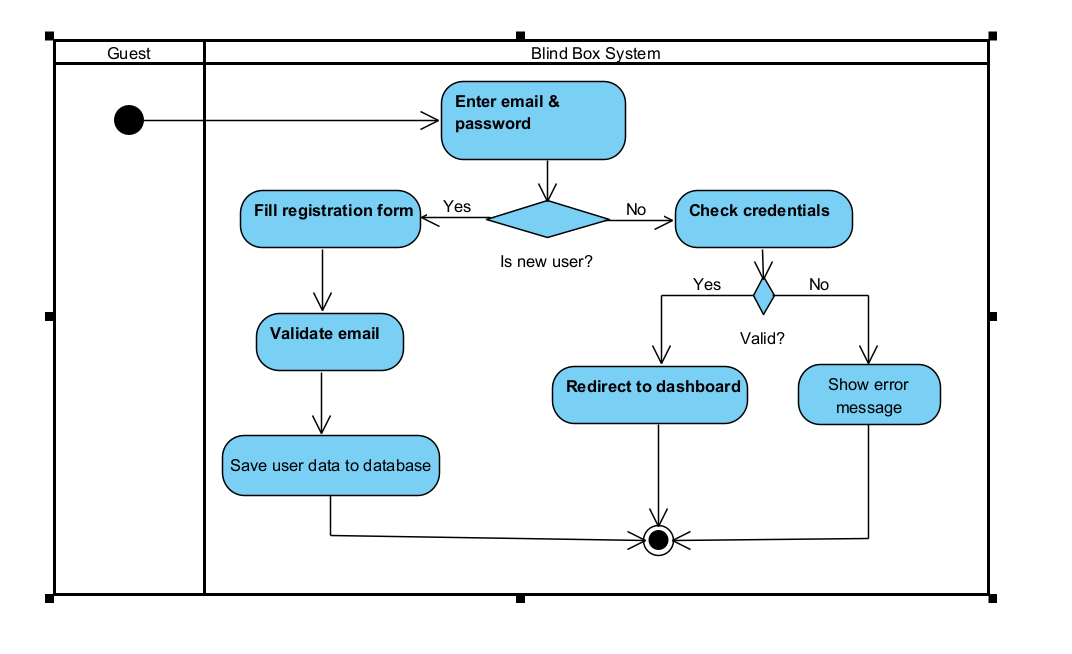
|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 20: Staff Generate Sales Report** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | The Staff wants to generate a sales report for analysis. | | |
| **Description:** | This use case allows the Staff to generate a detailed sales report based on specific time frames and filters. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. The system must have recorded sales data. | | |
| **Postconditions:** | 1. A sales report is generated and displayed. 2. The Staff can download or print the report. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Reports" section. 2. The Staff selects the "Sales Report" option. 3. The system presents filter options (date range, product category, total revenue, etc.). 4. The Staff applies the desired filters and clicks "Generate Report." 5. The system fetches the relevant sales data and compiles the report. 6. The system displays the sales report. 7. The Staff can choose to download or print the report. | | |
| **Alternative Flows:** | 1. If no sales data is available for the selected filters:    * The system displays a message stating that no records were found. 2. If the database connection fails:    * The system displays an error message and logs the issue. | | |
| **Exceptions:** | 1. The Staff does not have permission to access sales reports:   * The system denies access and notifies the Staff. | | |
| **Priority:** | High | | |
| **Frequency of Use:** | Weekly or monthly, depending on reporting needs. | | |
| **Business Rules:** | 1. Only authorized Staff members can generate reports. 2. Reports must include accurate, up-to-date sales data. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has reporting access.The system integrates with real-time sales data tracking. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 21: Staff Manage Discount Codes** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Staff | **Secondary Actors:** | None |
| **Trigger:** | The Staff needs to create, update, or delete discount codes. | | |
| **Description:** | This use case allows the Staff to manage discount codes, including adding new codes, modifying existing ones, and removing expired or unnecessary codes. | | |
| **Preconditions:** | 1. The Staff must be logged into the system. 2. The system must have existing discount codes if updating or deleting. | | |
| **Postconditions:** | 1. The discount code information is updated in the database. 2. Expired discount codes are removed. 3. A success message is displayed after a successful operation. | | |
| **Normal Flow:** | 1. The Staff navigates to the "Discount Management" page. 2. The Staff views the list of available discount codes. 3. The Staff chooses to create, update, or delete a discount code. 4. If creating, the Staff enters the new discount code details (code, discount percentage, validity dates) and submits the form. 5. If updating, the Staff modifies the discount code details and saves changes. 6. If deleting, the Staff confirms the deletion action. 7. The system validates the input and updates the database accordingly. 8. The system displays a success message. | | |
| **Alternative Flows:** | 1. If the Staff tries to delete a discount code currently in use:    * The system prevents deletion and notifies the Staff. 2. If the Staff inputs an invalid discount value:    * The system displays a validation error message. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The Staff attempts to create a duplicate discount code:    * The system prevents submission and prompts the Staff to choose a unique code. 2. The Staff tries to modify an expired discount code:    * The system prevents changes and suggests creating a new code. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed for promotions. | | |
| **Business Rules:** | 1. Each discount code must be unique. 2. Discount codes must have a validity period. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The Staff has permission to manage discount codes.The system provides real-time validation for discount management actions. | | |

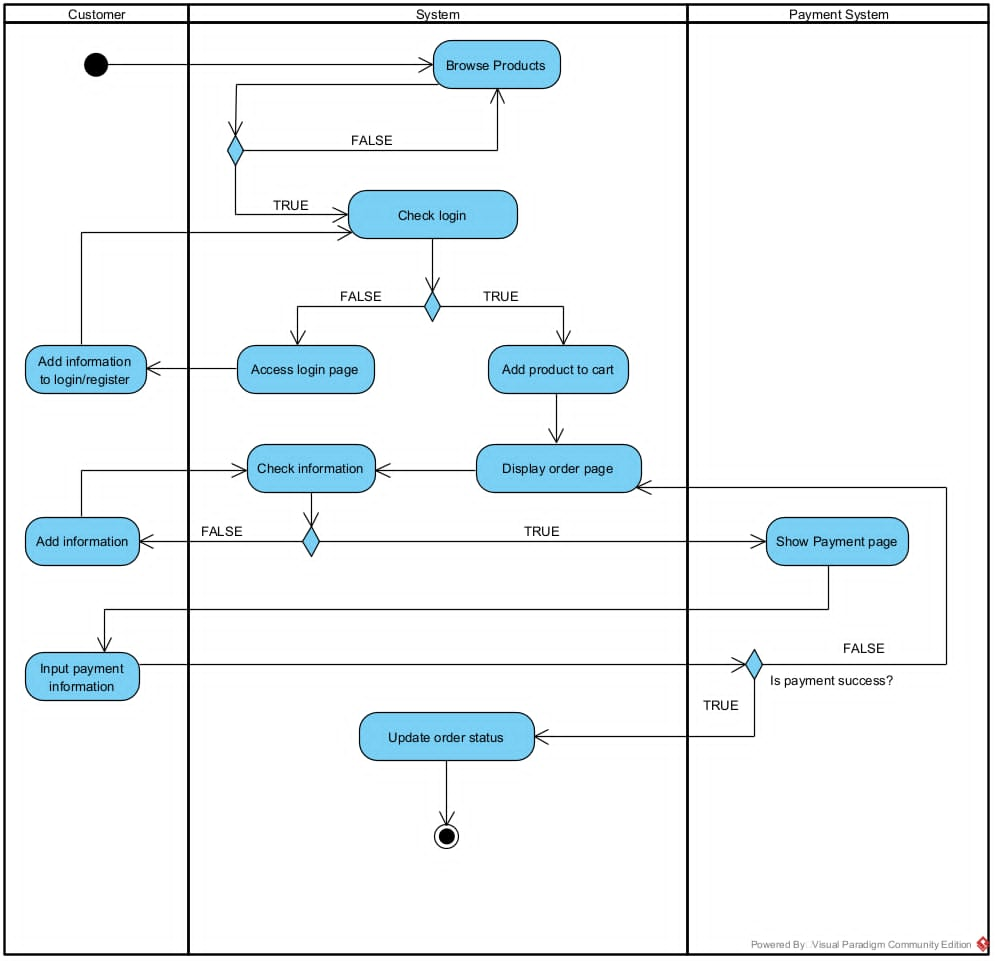
|  |  |  |  |
| --- | --- | --- | --- |
| **UC ID and Name:** | **the UC – 22: Customer Submit Product Review** | | |
| **Created By:** | Khánh Linh | **Date Created:** | 13/01/2025 |
| **Primary Actor:** | Customer | **Secondary Actors:** | None |
| **Trigger:** | A customer wants to submit a review for a purchased product. | | |
| **Description:** | This use case allows a customer to submit a review and rating for a product they have purchased. | | |
| **Preconditions:** | 1. The customer must be logged into the system. 2. The customer must have previously purchased the product. | | |
| **Postconditions:** | 1. The review is stored in the database and displayed on the product page. 2. The product's overall rating is updated. 3. The customer receives a confirmation message. | | |
| **Normal Flow:** | 1. The customer navigates to their order history. 2. The customer selects a product they have purchased. 3. The system displays an option to leave a review. 4. The customer enters a star rating and writes a review. 5. The customer submits the review. 6. The system validates the input and saves the review in the database. 7. The system updates the product’s rating based on the new review. 8. The system displays a success message. | | |
| **Alternative Flows:** | 1. If the customer tries to review a product they haven't purchased:    * The system prevents submission and displays an error message. 2. If the review is missing required fields:    * The system prompts the customer to complete all fields. 3. If the database connection fails:    * The system logs the issue and displays an error message. | | |
| **Exceptions:** | 1. The review contains inappropriate content:    * The system flags the review for moderation. 2. The customer tries to submit multiple reviews for the same product:    * The system prevents duplicate submissions. | | |
| **Priority:** | Medium | | |
| **Frequency of Use:** | As needed, when customers leave reviews. | | |
| **Business Rules:** | 1. Customers can only review products they have purchased. 2. Reviews must be moderated before being publicly visible. | | |
| **Other Information:** | N/A | | |
| **Assumptions:** | The customer has an active account and has completed at least one purchase.The system supports review moderation. | | |

**1.4 Activity Diagram**

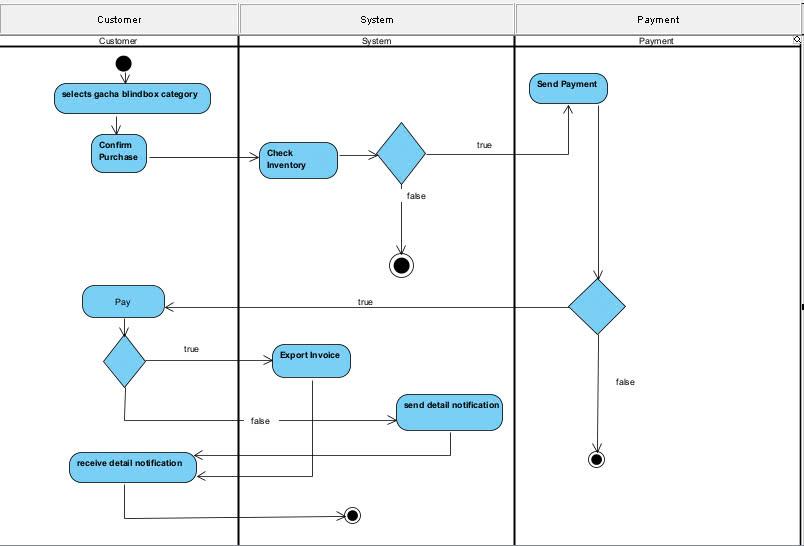
**1.4.1 Login**

**

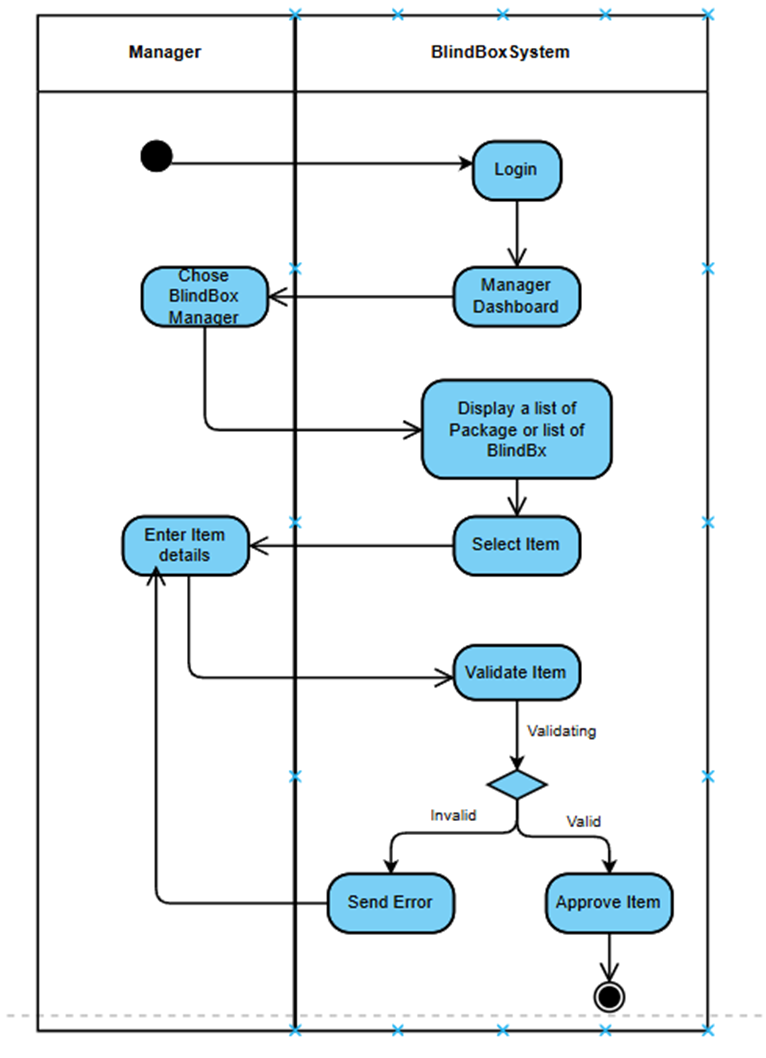
**1.4.2 Order Products(Blindbox/Package)**

****

**1.4.3 Lucky wheel**

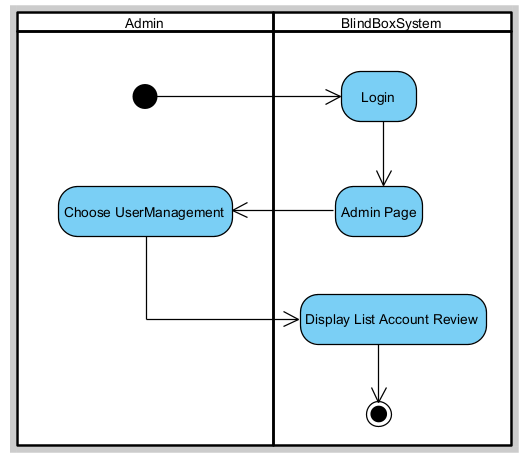
****

**1.4.4 Manage Product(Blindbox/Package)**

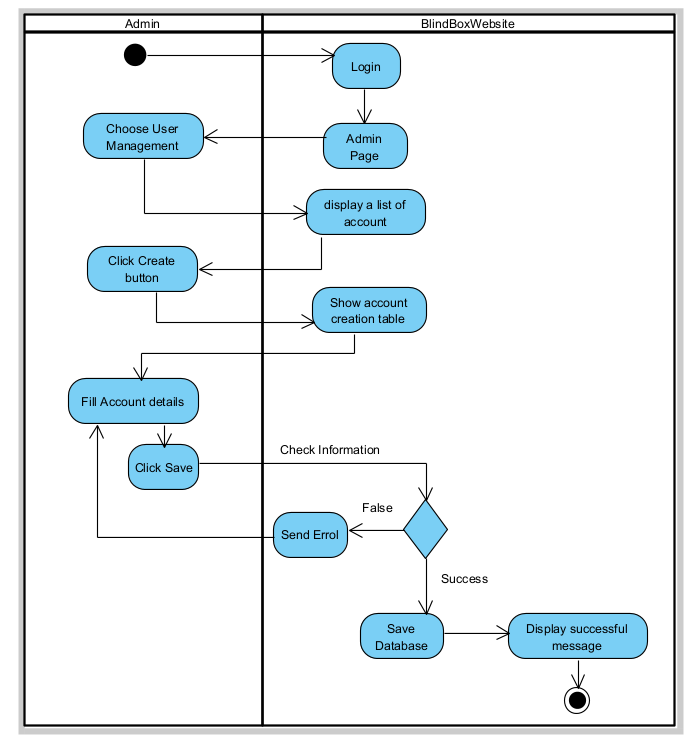
****

**1.4.5 Manage Account**

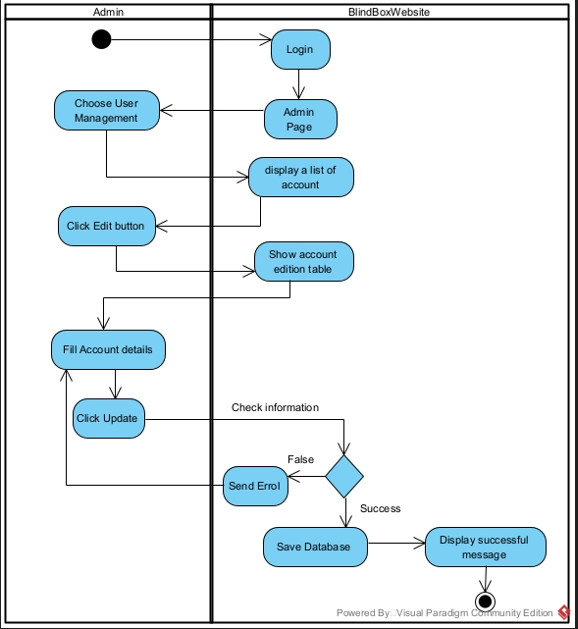
**<View account list>**



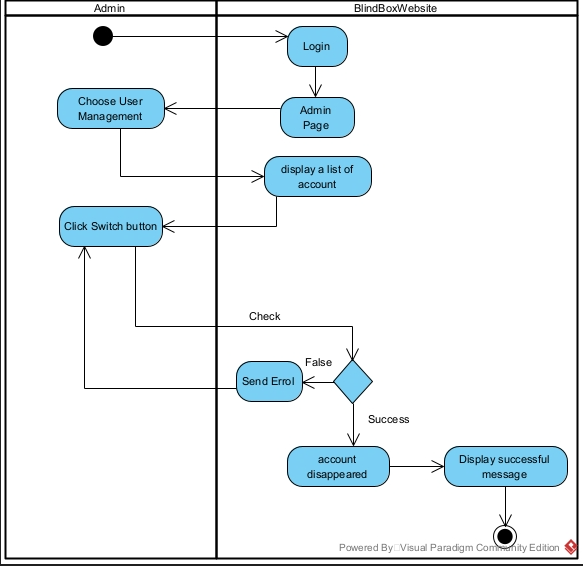
**<Add account>**



**<Edit account>**

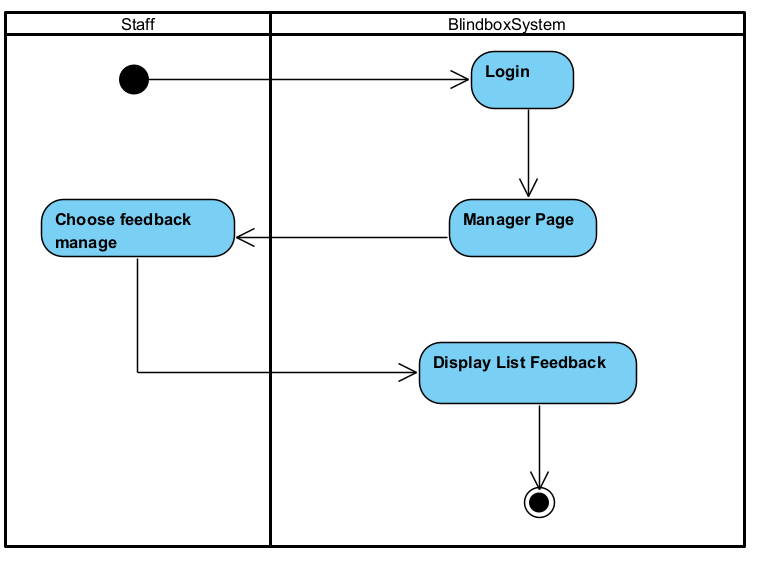


**<Delete Account>**

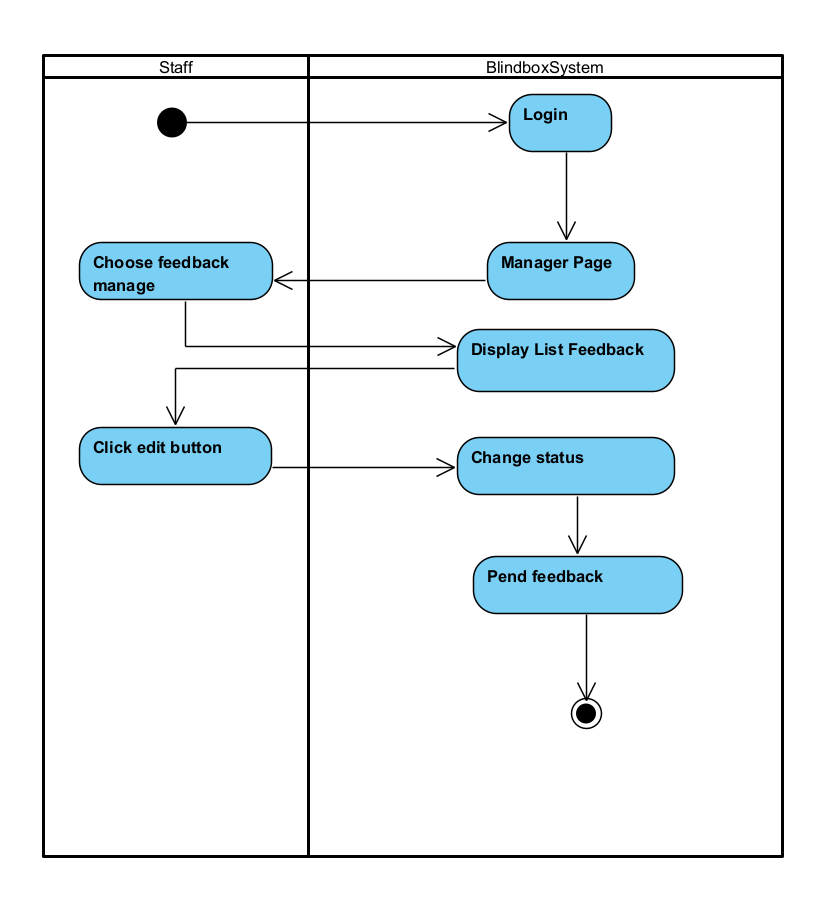


**1.4.6 Manage Feedback**

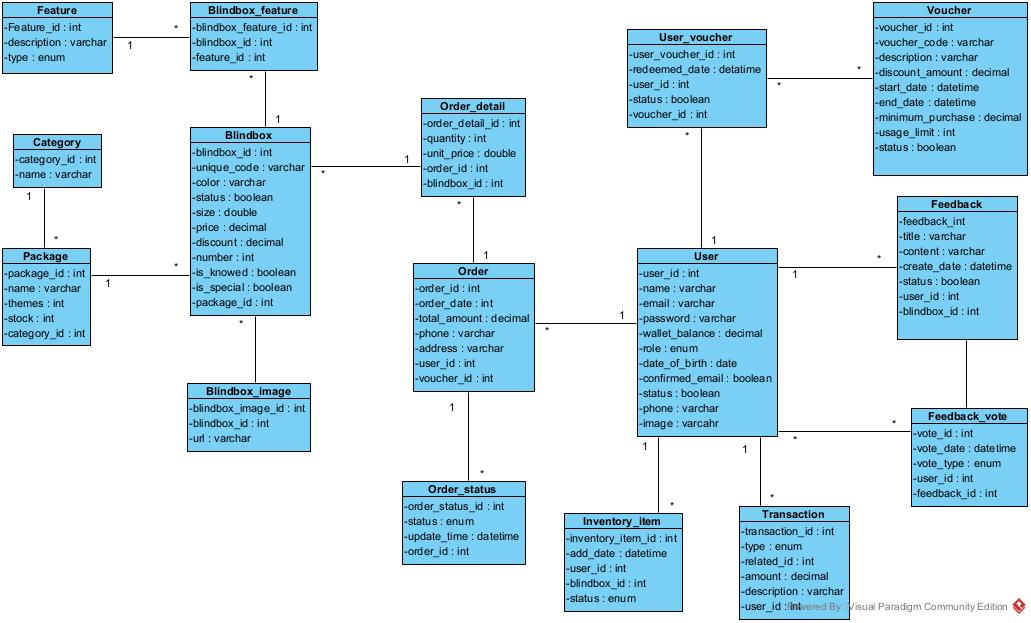
**<View list feedback>**

****

**<Edit feedback status>**

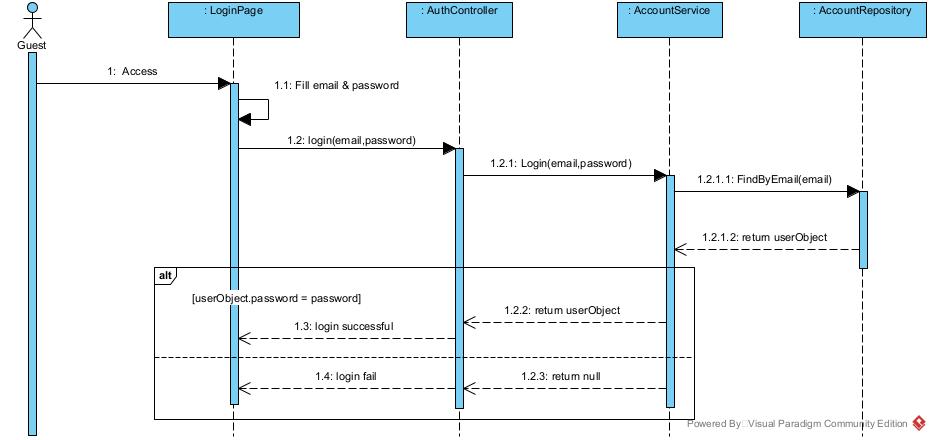
****

## 2. Class Diagram

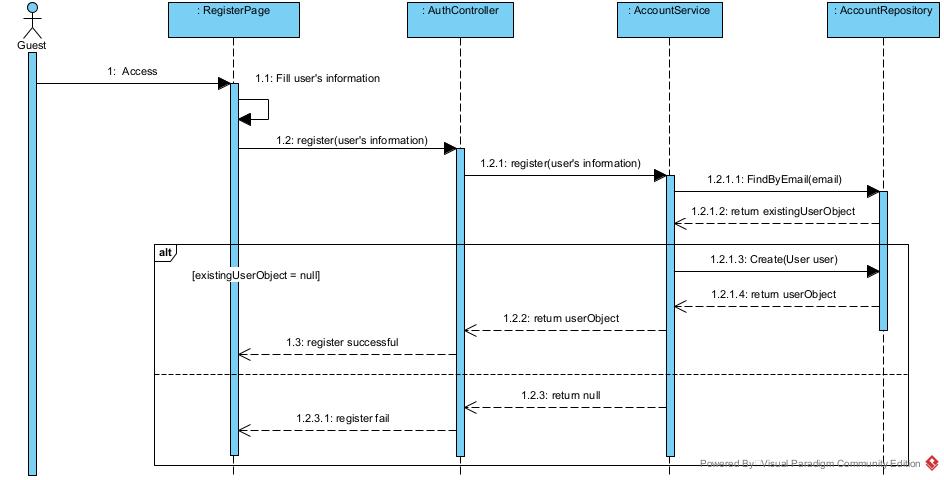


## 3. Sequence Diagram

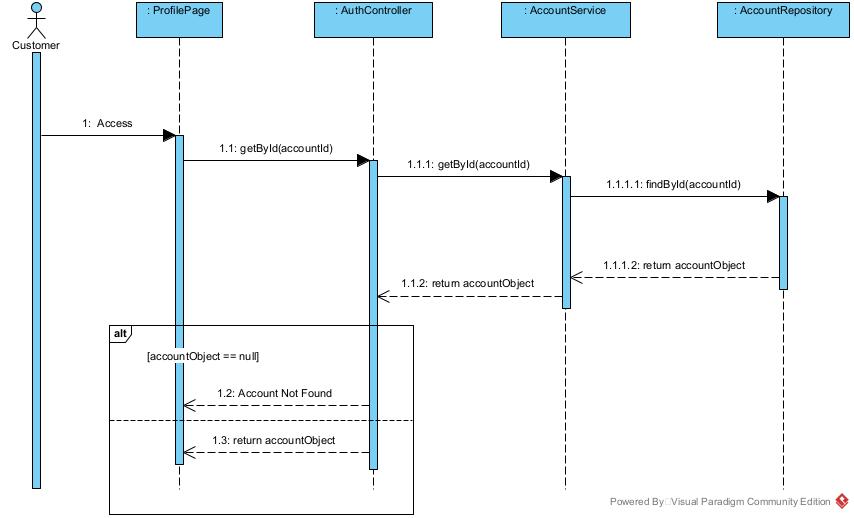
#### 3.1 Login

**

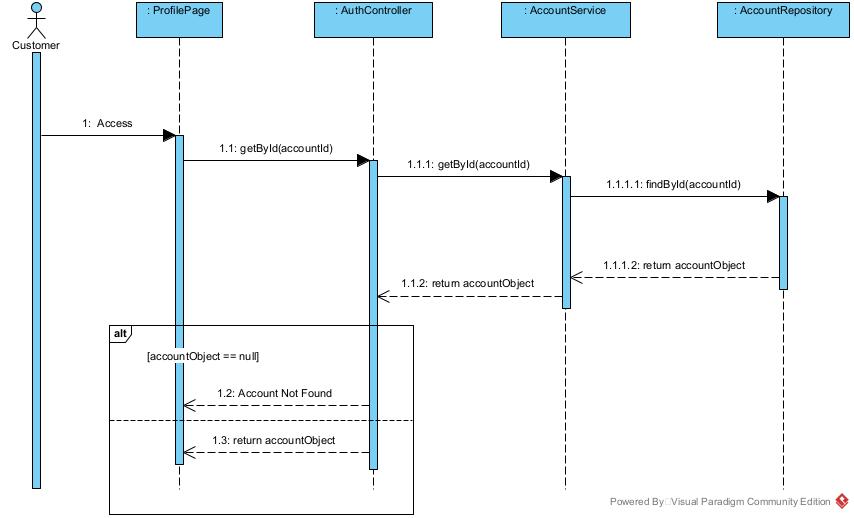
3.2 Register

******

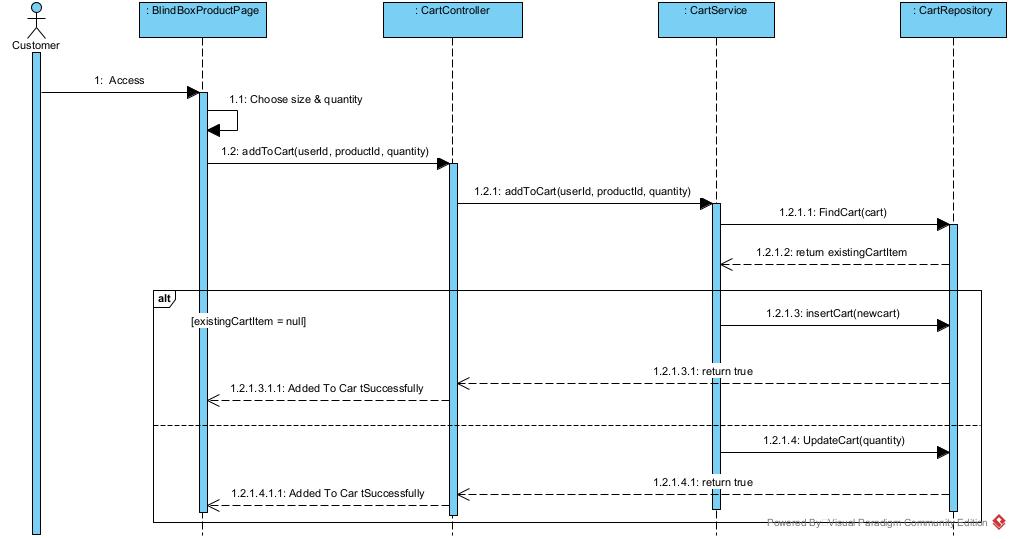
3.3 View Profile

******

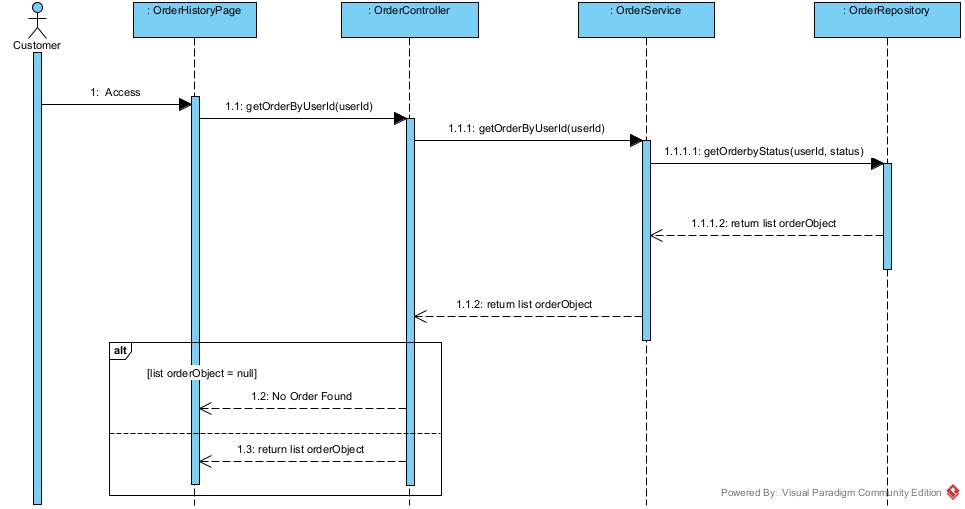
3.4 Edit Profile

******

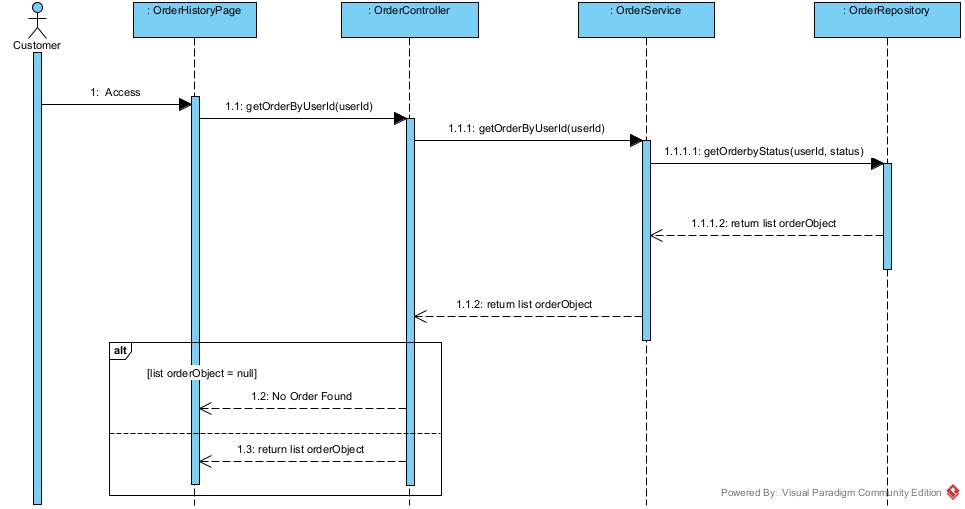
3.5 Change cart<Add to cart>

******

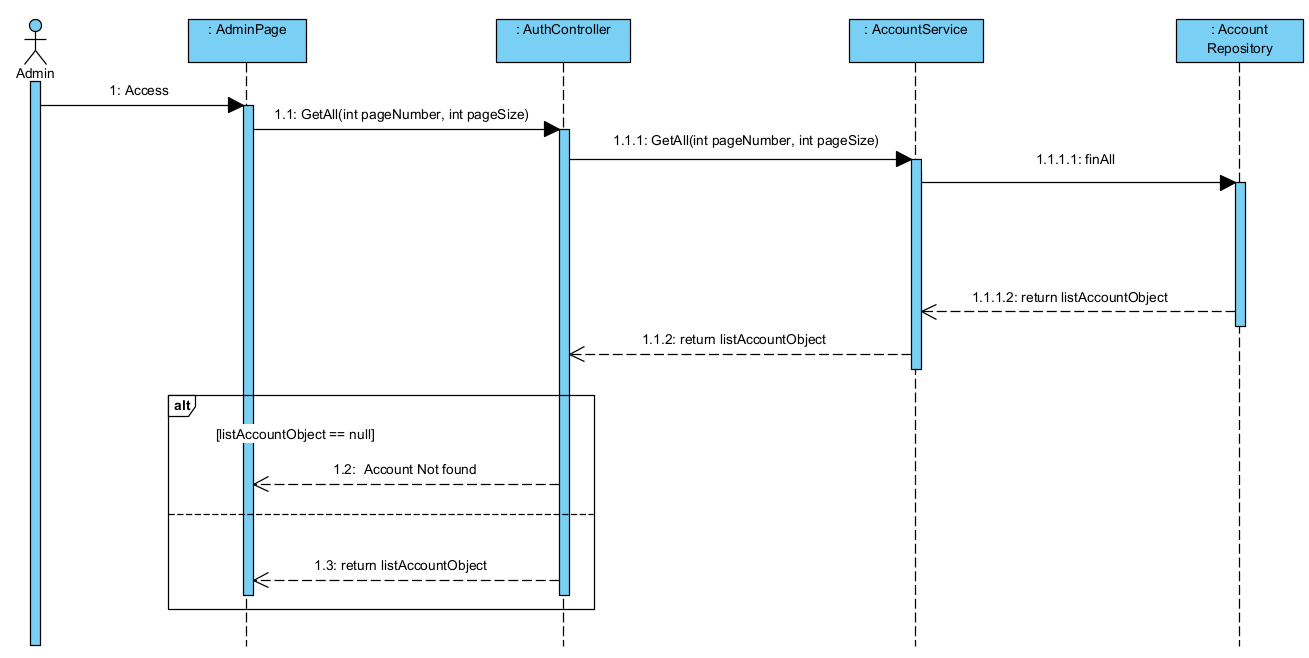
3.6 Change cart<Update cart>

******

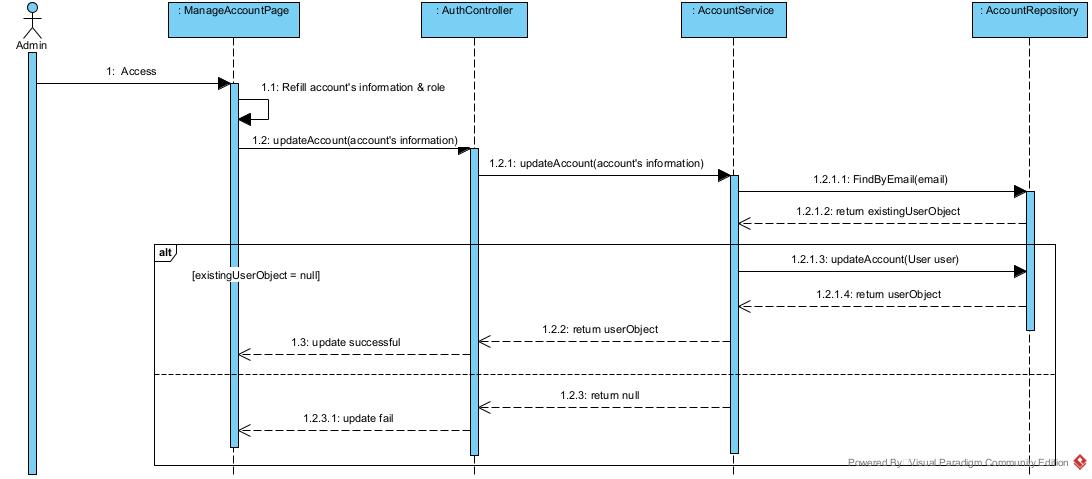
3.7 View order history

******

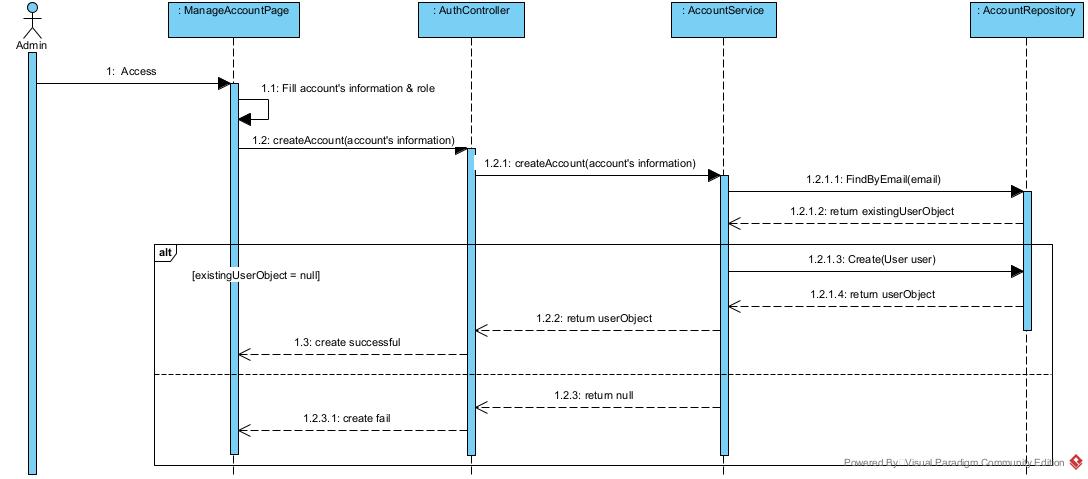
3.8 View Account



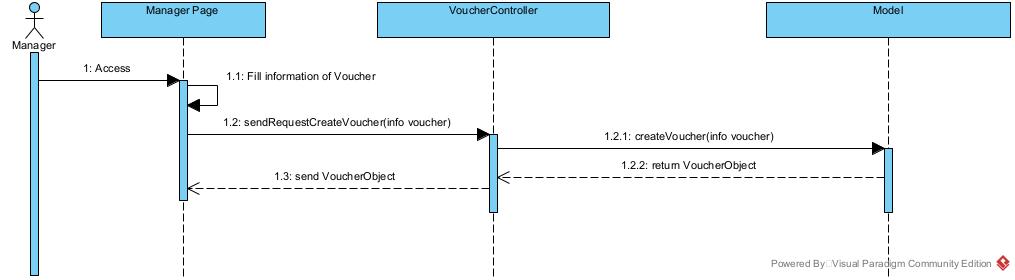
3.9 Change Account <Update Account>

******

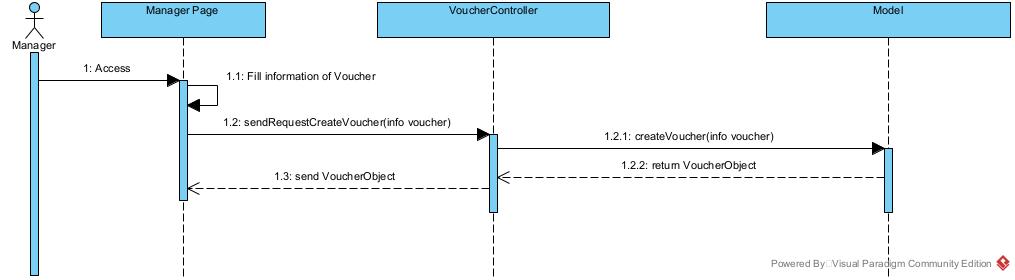
3.10 Change Account <Create Account>

******

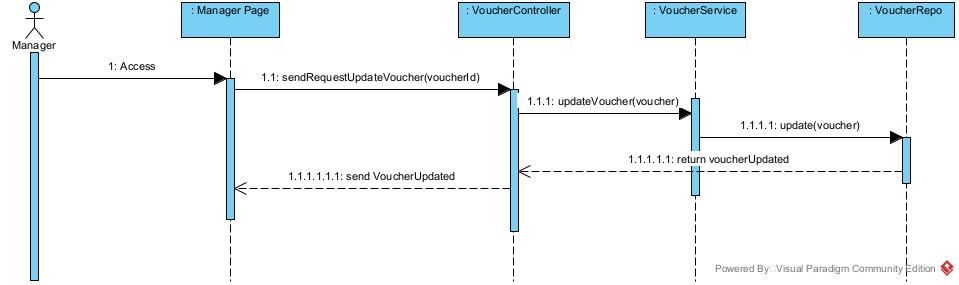
3.12 Change Voucher <Update voucher>



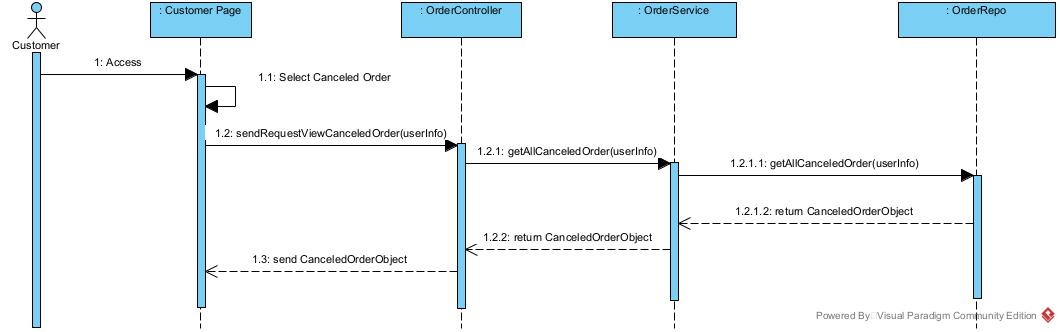
3.13 Change Voucher <Create voucher>



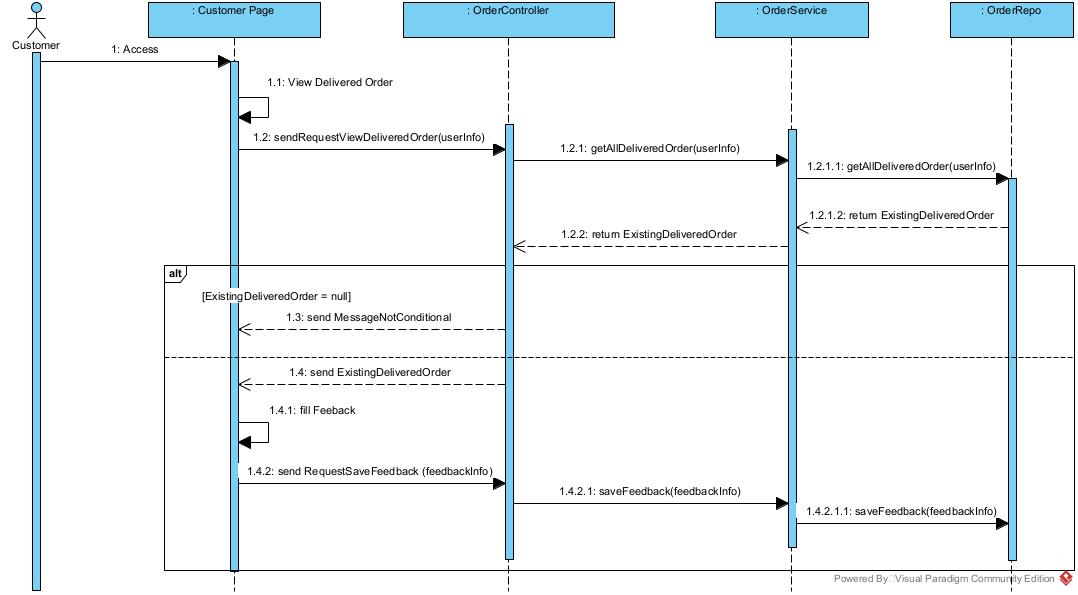
3.14 Change Voucher <Delete voucher>



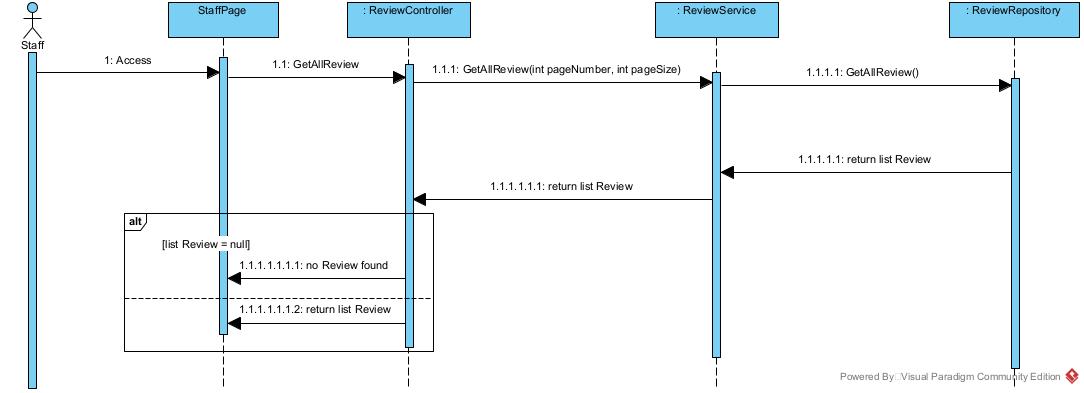
3.15 Feedback<Give feedback – Customer need order success>



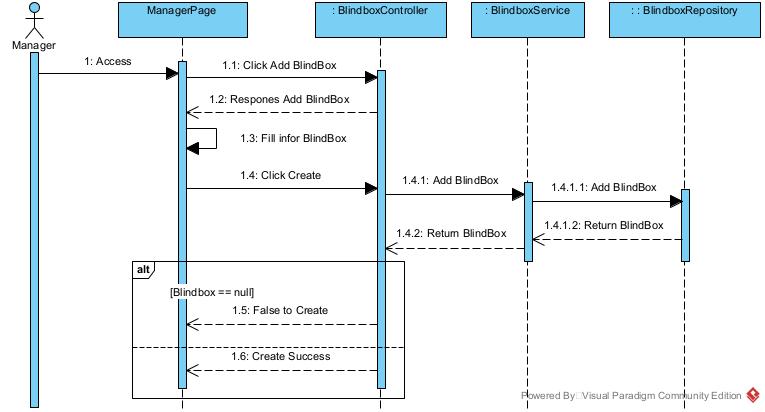
3.16 Feedback<View feedback list >



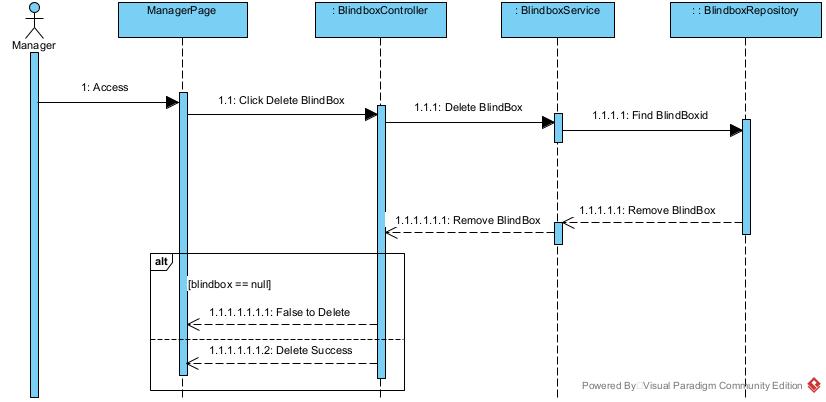
3.17 Feedback<Update feedback status>

******

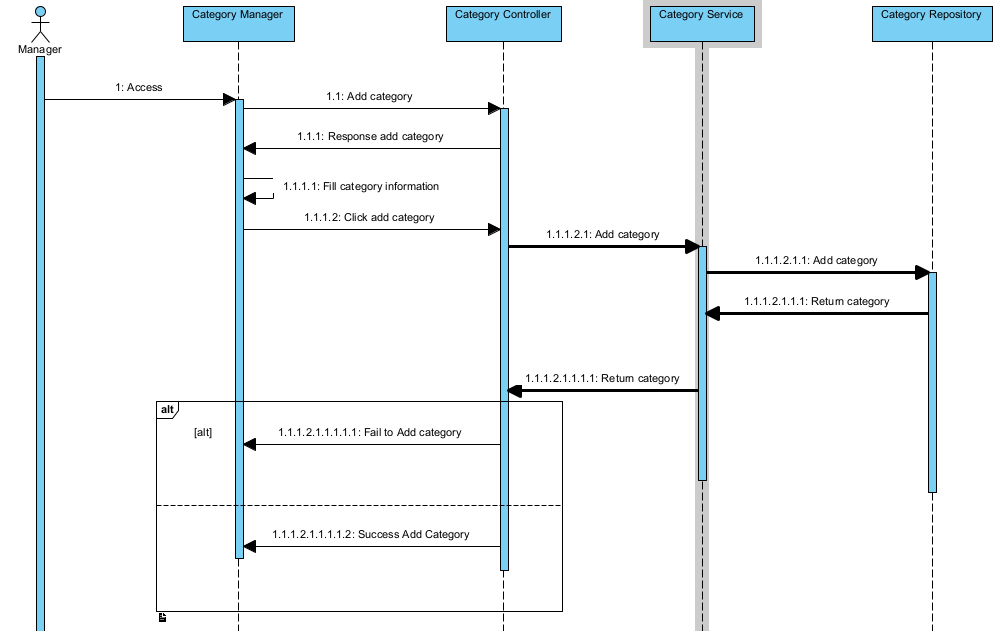
3.18 Update Blindbox

******

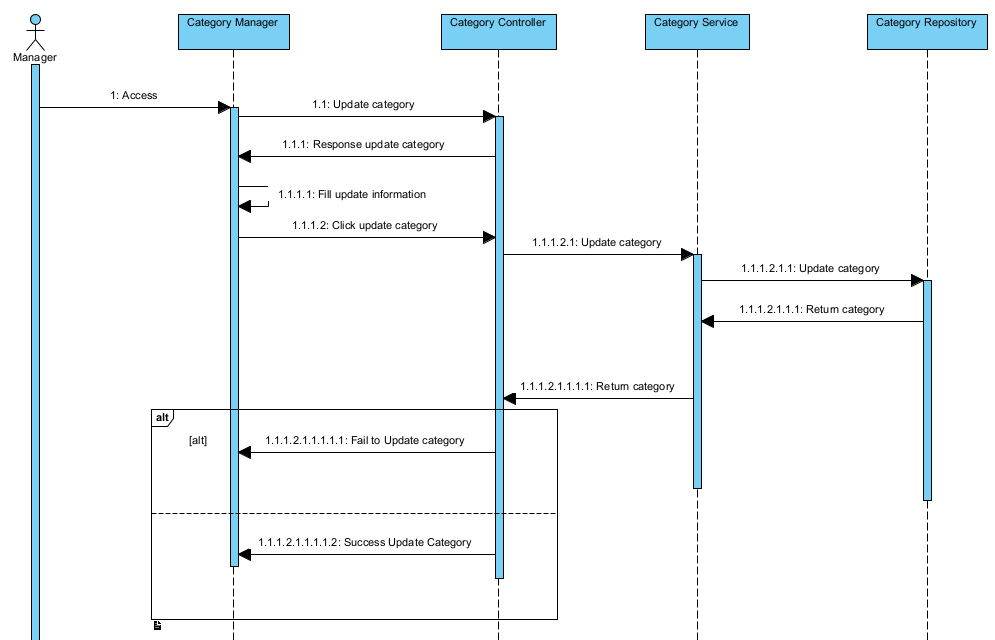
3.19 Delete Blindbox

******

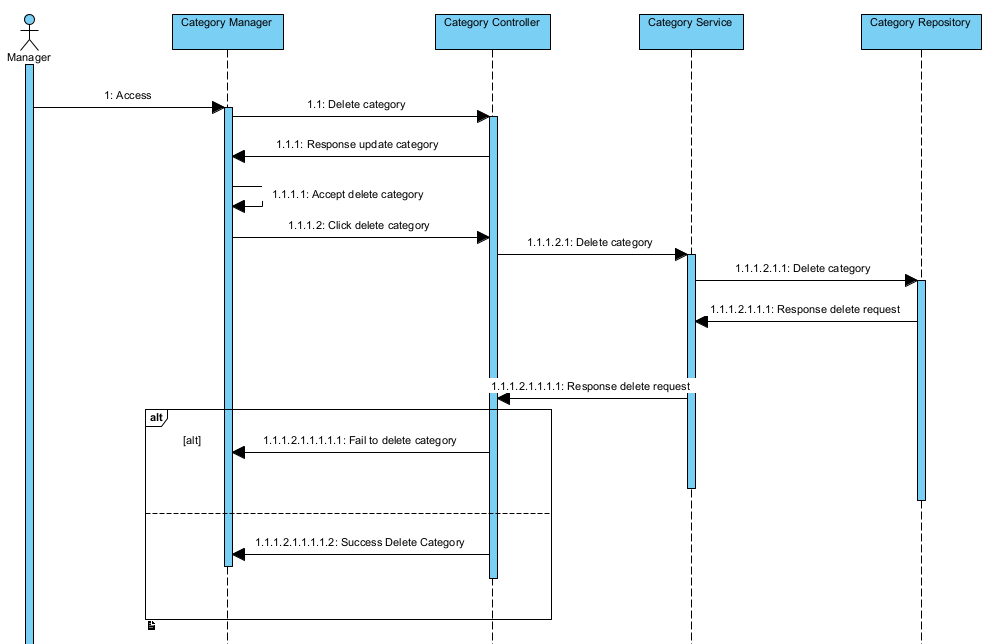
3.20 Category<Add category>

******

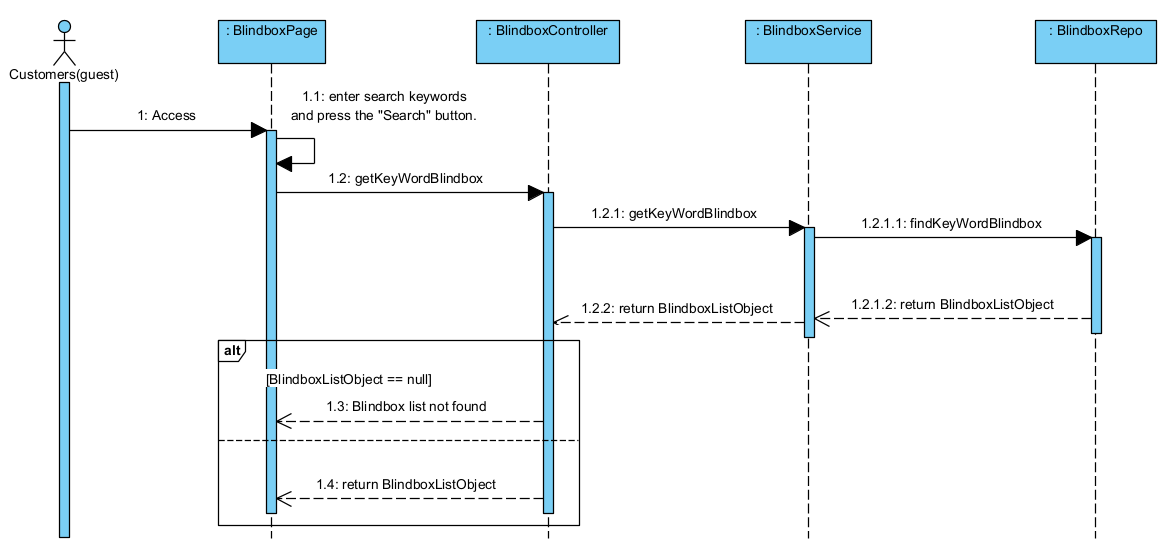
3.21 Category<Update Category >

******

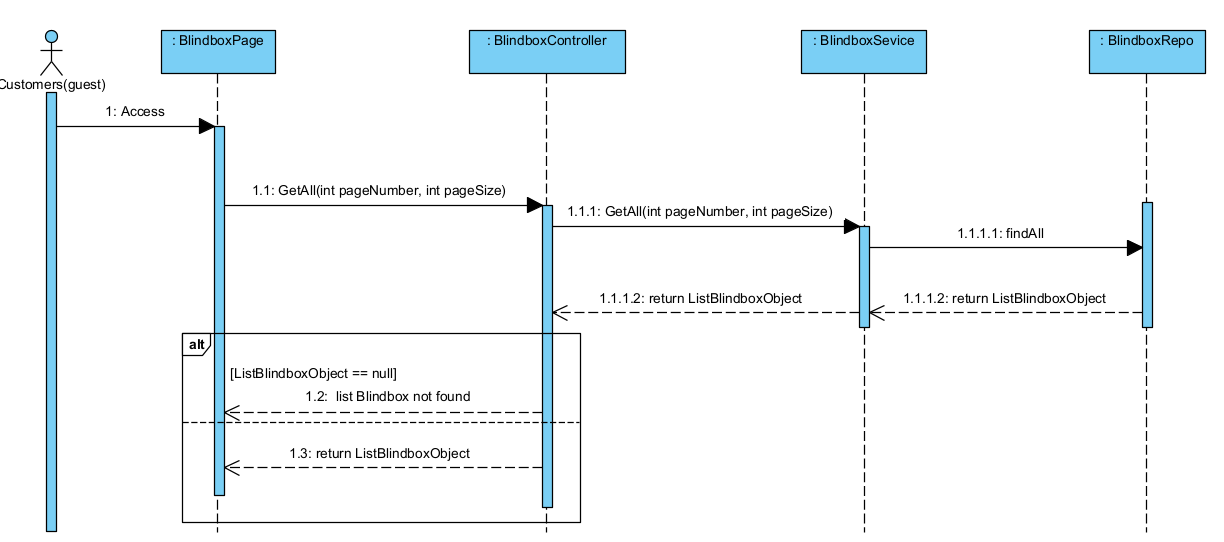
3.22 Category<Delete Category>

******

3.23 Search Product



3.24 View list product

******

3.25 View product detail

******

1. **State diagram**

**Order**

