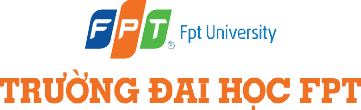
# 



BAKERY MANAGEMENT SYSTEM

The system supports the customers ordering cake and more!

|  |  |
| --- | --- |
| Project Code |  |
| Document Code |  |
| Group Member | Nguyen Tang Tai Phat – SE184119 (Leader)  Luu The Vinh - SE181740  Le Duc Loc - SE181673  Nguyen Nhu Tai - 173015  Nguyen Trung Kien - 181674  Thai Binh Duong - SE182174  Tran Anh Toan – SE173384 |
| Supervisor | ThS.Trinh Huy Hiep |

Ho Chi Minh City, Summer 2024

# TABLE OF CONTENT

[1](#_Toc171982915)

[TABLE OF CONTENT 1](#_Toc171982916)

[1. INTRODUCTION 2](#_Toc171982917)

[1.1. Purpose 3](#_Toc171982918)

[1.2. Scope 3](#_Toc171982919)

[1.3. Stakeholder Profile 4](#_Toc171982920)

[1.4. Project Priorities 5](#_Toc171982921)

[1.5. Overview 5](#_Toc171982922)

[2. OVERALL DESCRIPTION 6](#_Toc171982923)

[2.1. Product Overview 6](#_Toc171982924)

[2.2. Business Rules 6](#_Toc171982925)

[3. FUNCTIONAL REQUIREMENT 8](#_Toc171982926)

[3.1. Use Case Diagram 8](#_Toc171982927)

[3.2. Use Case List 9](#_Toc171982928)

[3.3. Use Case Detail 10](#_Toc171982929)

[UC-01: Sign In 10](#_Toc171982930)

[4. Non-Functional Requirement 11](#_Toc171982931)

[5. Database 11](#_Toc171982932)

[5.1. ERD 11](#_Toc171982933)

[5.2. Relational Schema 11](#_Toc171982934)

# 1. INTRODUCTION

Bakery businesses often face significant challenges in managing their day-to-day operations. These include inefficient inventory management, time-consuming order processing, and inadequate customer relationship management. Currently, many bakeries rely on manual processes or disparate systems to handle tasks such as tracking inventory levels, processing customer orders, and managing financial transactions. These inefficiencies not only reduce operational effectiveness but also affect the bakery’s ability to compete in a market where customer satisfaction and efficient service are crucial. A comprehensive Bakery Management System (BMS) is needed to address these issues by integrating and automating key aspects of bakery operations. The BMS will streamline inventory management, order processing, customer relationship management, and financial tracking. This system aims to enhance operational efficiency, reduce waste, improve customer satisfaction, and ultimately increase profitability. By addressing these operational challenges, the bakery can focus on delivering high-quality products and exceptional customer experiences.

## 1.1. Purpose

The purpose of a bakery system management is to streamline and optimize the various processes involved in the operation of a bakery. This system aims to enhance efficiency, improve customer service, and ensure effective resource utilization.

## 1.2. Scope

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Release 1 | Release 2 | Release 3 |
| FE-1, Order and Payment Processing | Users can order and pay for bakery items from the cafeteria menu for pickup or delivery. Payment by cash or credit card. | Integration with payment gateways for online payments. | Loyalty program integration for customer rewards. |
| FE-2, Menu Management | Admins can create, view, modify, and delete cafeteria menus. | Addition of daily specials and customizable meal options. | Integration with third-party recipe databases. |
| FE-3, Customer Accounts | Users can create and manage accounts to store preferences and order history. | Loyalty program integration for customer rewards. | Enhanced account features and personalized recommendations. |
| FE-4, Basic Reporting | Basic reporting capabilities provide insights into sales trends and popular menu items. | Advanced reporting and analytics for detailed insights. | Predictive analytics for inventory management. |
| FE-5, Multi-Platform Access | Access through corporate intranet, smartphones, tablets, and external internet access for authorized employees. | Development and launch of mobile applications for iOS and Android devices. | Enhancements and updates to mobile apps, including support for additional features. |

## 1.3. Stakeholder Profile

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stakeholder | Major Value | Attitudes | Major Interests | Constraints |
| Bakery Owner/  Manager | Streamlined bakery operations; increased efficiency and profitability | Open to innovation, but cautious about disruptions to existing workflows | Cost-effectiveness; improved inventory management; enhanced customer service | Limited budget for implementation; concerns about staff training and adoption |
| Bakery Staff | Simplified work processes; reduced manual tasks | Open to change, but concerned about job security and workload implications | Job satisfaction; efficient use of time and resources | Training and support needed for transitioning to the new system; potential resistance to technology adoption |
| Customers | Expanded product offerings; convenience | Excited about new offerings, but may have concerns about pricing and quality | Variety of bakery items; ease of ordering and payment | Accessibility issues for customers without internet or mobile devices; concerns about privacy and security of personal information |
| Suppliers | Increased business opportunities; streamlined transactions | Supportive of initiatives that improve efficiency and communication with the bakery | Timely payments; clear order specifications | Integration challenges with existing supplier systems; potential resistance to changes in ordering and invoicing processes |
| IT Department | Enhanced system performance and security | Supportive of technology upgrades, but cautious about system compatibility and maintenance | Scalability; data integration with existing systems | Limited resources for system implementation and maintenance; concerns about data security and compliance with regulations |

## 1.4. Project Priorities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dimension | Constraint | Driver | Degree of Freedom | Dimension |
| Features | All features scheduled for release 1.0 must be fully operational | Fulfillment of bakery operational needs and customer requirements | Flexibility in prioritizing features based on criticality and resource availability | Features |
| Quality | 95% of user acceptance tests must pass; all security tests must pass | Ensuring reliability, usability, and security of the BMS | Margin for addressing any issues identified during testing; continuous improvement | Quality |
| Schedule | Release 1 planned to be available by end of Q1 of next year, release 2 by end of Q2; overrun of up to 2 weeks acceptable without sponsor review | Timely delivery to meet bakery's operational needs and market demands | Buffer for unexpected delays or challenges; proactive monitoring and mitigation strategies | Schedule |
| Cost | Budget overrun up to 15% acceptable without sponsor review | Cost-effectiveness and financial viability | Contingency measures to manage unforeseen expenses; optimization of resource allocation | Cost |
| Staff | Team size is half-time project manager, half-time BA, 3 developers, and 1 tester; additional developer and half-time tester available if necessary | Adequate resource allocation and skill set diversity for project execution | Flexibility to scale team based on project requirements and workload fluctuations | Staff |

## 1.5. Overview

Bakery businesses often encounter challenges in managing daily operations, such as inefficient inventory management, time-consuming order processing, and inadequate customer relationship management. Many bakeries still use manual processes or disparate systems for tasks like tracking inventory, processing orders, and managing finances. These inefficiencies hinder operational effectiveness and competitiveness, affecting customer satisfaction and service efficiency. A comprehensive Bakery Management System (BMS) is proposed to address these issues by integrating and automating essential operations. The BMS will streamline inventory management, order processing, customer relationship management, and financial tracking, enhancing operational efficiency, reducing waste, improving customer satisfaction, and increasing profitability. This system will enable bakeries to focus on delivering high-quality products and exceptional customer experiences.

# 2. OVERALL DESCRIPTION

## 2.1. Product Overview

A **Bakery Management System (BMS)** is a software solution designed to streamline and automate all aspects of running your bakery, from ingredient control to customer service. Imagine a system that helps you manage inventory, create baking schedules, process sales, and analyse data.

## 2.2. Business Rules

|  |  |
| --- | --- |
| **ID** | **Rule Definition** |
| |  | | --- | | BR-01 |  |  | | --- | |  | | Some features such as viewing order history and browsing the product catalog should be available even without an internet connection. |
| |  | | --- | | BR-02 |  |  | | --- | |  | | Customers must provide valid payment information at the time of order placement.   |  | | --- | |  | |
| |  | | --- | | BR-03 |  |  | | --- | |  | | Orders cannot be cancelled or modified once they reach the "Baking" status.   |  | | --- | |  | |
| |  | | --- | | BR-04 |  |  | | --- | |  | | Inventory levels must be updated in real-time as orders are placed and processed   |  | | --- | |  | |
| |  | | --- | | BR-05 |  |  | | --- | |  | | Discount codes must be applied at the time of checkout, and only one discount code can be used per order.   |  | | --- | |  | |
| |  | | --- | | BR-06 |  |  | | --- | |  | | Refunds are only available for orders that have not yet been started ("Pending" status).   |  | | --- | |  | |
| |  | | --- | | BR-07 |  |  | | --- | |  | | Delivery options must be chosen at the time of order placement, with delivery times being estimated and not guaranteed.   |  | | --- | |  | |
| |  | | --- | | BR-08 |  |  | | --- | |  | | Customers have the right to request deletion of their personal data from the system in compliance with data protection regulations. |

# 3. FUNCTIONAL REQUIREMENT

## 3.1. Use Case Diagram

A diagram of a bakery management system

Description automatically generated

Full Resolution: https://drive.google.com/file/d/1oWOwajUPNKhp\_Ss\_NY-93UNa-7qRLXFg/view

## 3.2. Use Case List

|  |  |  |
| --- | --- | --- |
| ID | Usecase | Primary Actor |
| UC-01 | Sign In | Guest |
| UC-02 | Sign Up | Guest |
| UC-03 | View Near-Expired Food list | Guest |
| UC-04 | View Near-expired Food Detail | Guest |
| UC-05 | Search near-Expired Foods | Guest |
| UC-06 | Add Food To Cart | Guest |
| UC-07 | View Cart | Customer |
| UC-08 | Remove Item(s) | Customer |
| UC-09 | Check Out | Customer |
| UC-10 | View Order List | Customer |
| UC-11 | View Order Detail | Customer |
| UC-12 | Cancel Order | Customer |
| UC-13 | Refund | Customer |
| UC-14 | Contact via Built-in Chat System | Customer |
| UC-15 | Calling | Customer |
| UC-16 | View Profile | Customer |
| UC-17 | Edit Profile | Customer |
| UC-18 | View Accounts List | Admin |
| UC-19 | View Accounts Detail | Admin |
| UC-20 | Create Accounts | Admin |
| UC-21 | Modify Accounts | Admin |
| UC-22 | Disable Account | Admin |
| UC-23 | Remove Sale Product | Admin |
| UC-24 | Modify Sale Product | Admin |
| UC-25 | Verify New Sale Request | Admin |
| UC-26 | View own Sale Request | Distributor |
| UC-27 | Create new Sale Product | Distributor |
| UC-28 | Modify own Sale Product | Distributor |
| UC-29 | Delete own Sale Product | Distributor |
| UC-30 | View Related Order List | Distributor |
| UC-31 | View Related Order Detail | Distributor |

## 3.3. Use Case Detail

### UC-01: Register

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-01: Register** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guest wants to register an account. | | |
| Description: | This feature allows customer to create new account to become candidate or enterprise in the system. | | |
| Preconditions: | Guest has email or Google Account that can be used to receive OTP number. | | |
| Post-conditions: | **Success:** Redirect to member profile page  **Fail:** System shows error message corresponding to error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | * At least one input field doesn’t match the constrain * Email existed in the system * Username existed in the system * OTP number is expired * OTP number is not correct | | |
| Business Rules: | * OTP input consists of 6 positive numbers and is valid in 10 minutes. * If actors input the wrong OTP 5 times, their email will be locked in 10 minutes. * After signing up success, status of account is “inactive” and need update profile to become “active”. | | |
| Assumptions: | * **Users already have valid accounts:** It is assumed that all users attempting to log in to the system have previously registered accounts. * **Stable Internet Connection:** Assume that the user has a stable Internet connection to send login requests to the server and receive responses. * **Database is available and accessible:** Assume that the database that stores user information is functioning properly and is accessible for login authentication. * **Secure environment:** Assume that secure protocols (such as HTTPS) are used to protect credentials from leaks.   Compatible browser: It is assumed that the user uses a modern web browser and is compatible with the login system. | | |

### UC-02: Sign in

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-01: Sign in** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guest wants to authorize to the system so that he could access more functions as a member. | | |
| Description: | This feature allows customer to login to website. | | |
| Preconditions: | Guest has created an account on the website. | | |
| Post-conditions: | **Success**: System redirects customer to dashboard home page  **Fail**: System informs the message corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | At least one input field doesn’t match the constrain  Username not existed in the system  Incorrect password | | |
| Business Rules: | If actors input wrong password 5 times for 1 username, their account will be locked in 10 minutes. | | |
| Assumptions: | * **Registered Users** : It is assumed that all users attempting to log in have previously registered accounts in the system. * **Valid Login Information** : The user's username and password are entered in the correct format (e.g., do not contain invalid special characters). * **Stable Internet Connection** : The system requires a stable internet connection to authenticate login information through the server. * **Response Time** : The system must respond to login requests within an acceptable amount of time (usually a few seconds). | | |

### UC-03: View Near-Expired Food list

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-03: View Near-Expired Food list** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guests want to see lists of nearly expired foods so they can easily manage their shopping carts. | | |
| Description: | This feature allows Guests to quickly see expired foods. | | |
| Preconditions: | Customers must have a system account. | | |
| Post-conditions: | **Success**: The system redirects Guest to the nearly expired product category  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no expired products | | |
| Business Rules: | If there are no products that are nearly expired, the system will display a corresponding message | | |
| Assumptions: | * **Accurate Expiration Dates**: The expiration\_date field in the foods table is accurate and updated timely to reflect the true expiration dates of all food items. * **Consistent Date Format**: The expiration\_date is stored in a consistent and standard DATE format, ensuring accurate and reliable date comparisons. * **Sufficient Data for Filtering**: Each food item in the foodstablefood\_id,food\_name,expiration\_date,quantity. * **Regular Updates and Maintenance**: The NearExpiredFood view is regularly updated through triggers or scheduled jobs to reflect the most current data on near-expired food items. | | |

### UC-04: View Near-expired Food Detail

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-04: View Near-expired Food Detail** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guest wants to see details of any product in the list of expired products | | |
| Description: | This function allows Guests to view detailed information about any expired product | | |
| Preconditions: | Customers must have a system account. | | |
| Post-conditions: | **Success**: The system redirects Guest to the nearly expired product category  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no expired products | | |
| Business Rules: | If there are no products that are nearly expired, the system will display a corresponding message | | |
| Assumptions: | * **Accurate Expiration Dates**: The expiration\_date field in the foods table is accurate and updated timely to reflect the true expiration dates of all food items. * **Consistent Date Format**: The expiration\_date is stored in a consistent and standard DATE format, ensuring accurate and reliable date comparisons. * **Sufficient Data for Filtering**: Each food item in the foodstablefood\_id,food\_name,expiration\_date,quantity. * **Regular Updates and Maintenance**: The NearExpiredFood view is regularly updated through triggers or scheduled jobs to reflect the most current data on near-expired food items. | | |

### UC-05: Search near-Expired Foods

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-05: Search near-Expired Foods** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guests can search for any expired product name | | |
| Description: | This function allows Guests to search for any type of product in the list of expired products | | |
| Preconditions: | Guests need to go to the list of nearly expired products | | |
| Post-conditions: | **Success**: The system will display the product Guest is looking for  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no expired products | | |
| Business Rules: | The search character must match the system format | | |
| Assumptions: | * **Exact Expiration Date Data** : The fields expiration\_datein the table foodsare accurate and updated promptly to reflect the correct expiration dates of all food items. * **Uniform Date Format** : Expiration dates ( expiration\_date) are stored in the standard DATE format, ensuring accurate date comparisons and queries. * **Full Data** : Each entry in the table foodshas complete data including food\_id, food\_name, expiration\_date, and quantity. * **Expiration Time Clearly Defined** : The time considered "near expiration" is clearly defined (e.g., within the next 7 days) and applied consistently in the search logic. | | |

### UC-06: Add Food To Cart

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-06: Add Food To Cart** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Guest | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Guests can add the food they want to buy to their shopping cart | | |
| Description: | This function allows Guests to add any type of food they want to buy to the cart | | |
| Preconditions: | Guest must successfully log in and access the system's home page | | |
| Post-conditions: | **Success**: The food Guest wants to buy will be displayed in the shopping cart  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The food Guest wants to buy is not in the system list | | |
| Business Rules: | The search character must match the system format | | |
| Assumptions: | * **User Authentication**: Users must be authenticated (logged in) to add items to their cart. * **Valid Food Items**: The food items being added to the cart exist in the foods table and have valid food\_id,food\_name,price, and quantity fields. * **Data Integrity**: The system ensures that data integrity is maintained, meaning that no corrupted or invalid data is present in the food stable * **Session Persistence**: The cart’s state persists across sessions if the user logs out and logs back in, assuming the system supports persistent carts. | | |

### UC-07: View Cart

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-07: View Cart** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Guest |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can view products that have been added to the cart | | |
| Description: | This function allows customers to view the foods they have added to the cart | | |
| Preconditions: | The customer must have successfully added the product | | |
| Post-conditions: | **Success**: The system displays a list of products added to the cart  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not added any products to the cart | | |
| Business Rules: | Only logged in users can view their shopping cart.  The shopping cart will display detailed information about each product, including product name, quantity, unit price, and total price of each product.  Update Product Information In Real Time. | | |
| Assumptions: | * **Logged In User** : Assume that the user is logged in to the system to be able to view his shopping cart. * **Active Session** : It is assumed that the user's session is active and their shopping cart has been initialized. * **Accurate Product Information** : It is assumed that all product information in the shopping cart such as product name, price, and quantity is correct and updated in a timely manner. * **Cart Data Stored Properly** : It is assumed that the shopping cart data is stored in a database or temporary memory without errors or data loss. | | |

### UC-08: Remove Item(s)

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-08: Remove Item(s)** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can replace products in the shopping cart | | |
| Description: | This function allows customers to change any food in the cart to another food | | |
| Preconditions: | The customer must successfully add the product to the cart | | |
| Post-conditions: | **Success**: Display the newly replaced product in the customer's shopping cart  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not added any products to the cart | | |
| Business Rules: | Only logged in users can view their shopping cart.  The shopping cart will display detailed information about each product, including product name, quantity, unit price, and total price of each product.  Update Product Information In Real Time. | | |
| Assumptions: | * **User Authentication**: Users must be authenticated (logged in) to remove items from their cart. * **Active Session**: Users have an active session that maintains their cart state throughout their interaction with the system. * **Existing Cart Items**: The cart contains items, and the item(s) the user wishes to remove are already present in the cart. * **Accurate Item Identification**: Each item in the cart has a unique identifier (e.g., food\_id) that allows for accurate identification and removal. | | |

### UC-09: Check Out

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-09: Check Out** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can pay for selected products | | |
| Description: | This function allows customers to proceed with payment for products added to the cart | | |
| Preconditions: | The customer must successfully add the product to the cart | | |
| Post-conditions: | **Success**: The system will display a notification of successful payment  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not added any products to the cart | | |
| Business Rules: | The system must verify that the user’s cart is not empty before allowing them to proceed with checkout.  The system must check that all items in the cart are available in the required quantities before completing the checkout process.  Users must provide and confirm accurate shipping information before proceeding to payment.  The system must accurately calculate the total amount for the items in the cart, including taxes, shipping fees (if applicable), and any discounts or promotional codes. | | |
| Assumptions: | * **Accurate Cart Contents:** The cart accurately reflects the items that the user intends to purchase, with correct quantities and prices. * **Stock Availability:** The system assumes that the stock levels in the database are up-to-date and reflect real-time inventory levels. * **Valid Shipping Information:** Users will provide accurate and valid shipping information necessary for order delivery. * **Payment Method Availability:** The system supports multiple payment methods (credit card, debit card, e-wallet, etc.) and they are operational at the time of checkout. | | |

### UC-10: View Order List

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-10: View Order List** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can view the order list | | |
| Description: | This function allows customers to view product orders that have been paid | | |
| Preconditions: | Customers must pay for the order successfully | | |
| Post-conditions: | **Success**: Displays a list of orders that have been successfully paid  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not paid for the product in the shopping cart | | |
| Business Rules: | Users must select a payment method (e.g., credit card, debit card, e-wallet, cash on delivery) and provide the necessary payment details.  The system must authenticate and validate the payment information, ensuring that the payment transaction is successful before processing the order.  The system must update the stock levels for the purchased items after a successful checkout to reflect the new inventory status. | | |
| Assumptions: | * **System Performance and Reliability:** The system is capable of handling multiple concurrent checkout processes without performance degradation or failures. * **Real-Time Order Confirmation:** The system is capable of sending real-time order confirmations to users via email or in-app notifications. * **Session Persistence:** User session data is persisted properly, ensuring that the checkout process can continue smoothly even if interrupted. | | |

### UC-11: View Order Detail

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-11: View Order Detail** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can view details of orders that have been paid | | |
| Description: | Customers can view detailed information about the paid product | | |
| Preconditions: | Customers must pay for the order successfully | | |
| Post-conditions: | **Success**: Displays a list of orders that have been successfully paid  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not paid for the product in the shopping cart | | |
| Business Rules: | Users must select a payment method (e.g., credit card, debit card, e-wallet, cash on delivery) and provide the necessary payment details.  The system must authenticate and validate the payment information, ensuring that the payment transaction is successful before processing the order.  The system must update the stock levels for the purchased items after a successful checkout to reflect the new inventory status. | | |
| Assumptions: | * **System Performance and Reliability:** The system is capable of handling multiple concurrent checkout processes without performance degradation or failures. * **Real-Time Order Confirmation:** The system is capable of sending real-time order confirmations to users via email or in-app notifications. * **Session Persistence:** User session data is persisted properly, ensuring that the checkout process can continue smoothly even if interrupted. | | |

### UC-12: Cancel Order

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-12: Cancel Order** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can cancel paid orders | | |
| Description: | This function allows customers to cancel payment for products that have already been paid | | |
| Preconditions: | The customer must successfully purchase the product | | |
| Post-conditions: | **Success**: Display a notification that the order has been successfully canceled  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not paid for the product in the shopping cart | | |
| Business Rules: | Orders can only be canceled if they are in a cancellable status (e.g., pending, processing). Orders that are already shipped, delivered, or completed cannot be canceled.  Users must cancel their orders within a specific timeframe from the order placement time (e.g., within 24 hours) as defined by the bakery's cancellation policy.  The system must update the order status to "canceled" in real-time and ensure that the inventory is adjusted accordingly if items were reserved or deducted from stock. | | |
| Assumptions: | * **Payment and Refund Processing Capability**: The system has the capability to process payments and initiate refunds according to the bakery's policies and the user's payment method. * **Inventory Management**: The system can adjust inventory levels accurately and in real-time, reflecting the items being returned to stock following the order cancellation. * **Notification System**: The system has a reliable notification system to inform users about the successful cancellation of their order and any subsequent steps, such as refund processing. | | |

### UC-13: Refund

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-13: Refund** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers are compensated for the product if there is an error from the store | | |
| Description: | This function allows customers to be compensated for defective products caused by the store | | |
| Preconditions: | The customer received the order successfully | | |
| Post-conditions: | **Success**: Displays a compensation confirmation message  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not received the order | | |
| Business Rules: | Refunds are only processed for orders that meet the bakery's refund policy criteria, such as cancellations within the allowed timeframe, incorrect items delivered, or items that are damaged upon delivery.  Users must initiate a refund request through the system, either by submitting a form or contacting customer support. The request must include the order number, reason for the refund, and any supporting evidence (e.g., photos of damaged items).  All refund requests must be reviewed and approved by the bakery’s customer service team. The team verifies the validity of the request based on the provided information and the bakery’s refund policy. | | |
| Assumptions: | * **Supporting Evidence**: Users provide necessary supporting evidence (e.g., photos of damaged items) when submitting a refund request. * **Customer Service Availability**: The bakery’s customer service team is available to review and process refund requests within a reasonable timeframe. * **Accurate Refund Calculation**: The system can accurately calculate the refund amount, including any applicable taxes, shipping fees, and deductions such as cancellation fees if applicable. | | |

### UC-14: Contact via Built-in Chat System

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-14: Contact via Built-in Chat System** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can chat with the system admin | | |
| Description: | This function allows customers to chat with the admin about product questions | | |
| Preconditions: | Customers must successfully log in to the system | | |
| Post-conditions: | **Success**: Display chat frame in customer chat section  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has no information to discuss with the admin | | |
| Business Rules: | The chat system should display the availability status of customer support agents. If no agents are available, users should be notified and provided with alternative contact methods (e.g., email, phone).  Users can initiate a chat session from the designated section within the Bakery Management System interface, such as the help or support section.  Upon initiating a chat, users should receive an automatic welcome message informing them that a support agent will be with them shortly. | | |
| Assumptions: | * **Stable Internet Connection**: Users and support agents have a stable internet connection to ensure continuous and uninterrupted communication during chat sessions. * **System Uptime**: The chat system is operational and has minimal downtime, ensuring availability for users to contact support at any time. * **Privacy and Security Compliance**: The system complies with data protection and privacy regulations, ensuring that all chat communications are secure and confidential. | | |

### UC-15: Calling

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-15: Calling** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers call the store directly | | |
| Description: | Customers can talk directly with the store via the hotline | | |
| Preconditions: | Customers must have the store's phone number | | |
| Post-conditions: | **Success**: The customer successfully called the store  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has no information to discuss with the store | | |
| Business Rules: | Users can initiate a call through the Bakery Management System interface by clicking a "Call Support" button or similar functionality.  Calls must be routed to the appropriate support agent based on the nature of the inquiry (e.g., orders, technical support, general inquiries).  The system should offer users the option to request a callback if the wait time exceeds a certain threshold. Users can specify a preferred callback time. | | |
| Assumptions: | * **Stable Internet and Phone Connection**: Users and support agents have stable internet and phone connections to ensure continuous and uninterrupted communication during calls. * **Privacy and Security Compliance**: The system complies with data protection and privacy regulations, ensuring that all call communications are secure and confidential. * **Call Recording Capability**: The system has the capability to record calls securely for quality assurance and training purposes, and users are informed about call recording. | | |

### UC-16: View Profile

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-16: View Profile** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can view their own information | | |
| Description: | Customers can view their own details in the system | | |
| Preconditions: | Customers must update information while registering for an account | | |
| Post-conditions: | **Success**: Display complete customer information  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Customers have not fully updated their information on the system | | |
| Business Rules: | Users can view their profile information, including personal details (name, email, phone number), account details, order history, and saved addresses.  Profile information is displayed in read-only mode by default. Users must switch to edit mode to make any changes.  The system must display accurate and up-to-date profile information based on the latest data provided by the user. | | |
| Assumptions: | * **Accurate Data Entry**: Users have entered their profile information accurately during registration or profile updates, ensuring the data displayed is correct. * **User Familiarity with Interface**: Users have a basic understanding of how to navigate the profile section within the Bakery Management System software. * **Read-Only Default View**: Profile information is displayed in a read-only format by default, preventing accidental changes. * **Profile Picture Availability**: If users have uploaded a profile picture, it will be displayed. If not, a default image will be shown. | | |

### UC-17: Edit Profile

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-17: Edit Profile** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Customer | Secondary Actors: | Member |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Customers can edit personal information | | |
| Description: | This function allows customers to edit their personal information | | |
| Preconditions: | Customers must update information while registering for an account | | |
| Post-conditions: | **Success**: Display update completed customer information  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Customers do not have any personal information that needs to be edited | | |
| Business Rules: | Users can edit specific fields within their profile, such as name, email, phone number, address, and profile picture. Sensitive fields like user ID or email verification status should not be editable by users directly.  Users must explicitly save their changes by clicking a "Save" or "Update" button. Changes should not be saved automatically to prevent accidental updates.  Users should be able to upload a new profile picture. The system must validate the file type (e.g., JPG, PNG) and file size before accepting the upload. | | |
| Assumptions: | * **Accurate Initial Data**: The profile data initially displayed to users is accurate and up-to-date, reflecting the latest information provided by the user. * **User Familiarity with Interface**: Users have a basic understanding of how to navigate the profile editing interface within the Bakery Management System software. * **User Consent**: Users have agreed to the terms and conditions of using the profile editing feature, understanding the implications of changing their personal information. * **File Upload Capability**: The system supports file uploads for profile pictures, including validation for file type (e.g., JPG, PNG) and file size. | | |

### UC-18: View Accounts List

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-18: View Accounts List** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can view the list of customers on the system | | |
| Description: | This function allows Admin to see all customer accounts currently on the system | | |
| Preconditions: | The system must have customer registered account information | | |
| Post-conditions: | **Success**: Displays a list of customer accounts  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | No customers have successfully registered for a system account | | |
| Business Rules: | The accounts list should include relevant details such as user ID, username, email address, role, account status (active/inactive), and date of creation.  The system must ensure that the accounts list displayed is accurate and up-to-date, reflecting the latest changes made to user accounts.  Users should be able to search for specific accounts and filter the list based on criteria such as role, account status, or creation date.  Users should be able to sort the accounts list by different fields such as username, email address, role, and date of creation. | | |
| Assumptions: | * **User Roles and Permissions:**   + The system has multiple user roles such as Admin, Manager, Accountant, and Employee.  + Only authorized users (e.g., Admins and Accountants) can access the View Accounts List feature.   * **Account Types:** The system categorizes accounts into different types such as Customer Accounts, Vendor Accounts, and Employee Payroll Accounts. * **Data Storage:** Account data is stored in a secure database with proper indexing to allow efficient querying and retrieval. * **Search and Filtering:** Users can search for accounts using various criteria such as account name, account ID, contact information, and balance range. | | |

### UC-19: View Accounts Detail

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-19: View Accounts Detail** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can view customer account details | | |
| Description: | This function allows Admin to view personal information in customer accounts on the system | | |
| Preconditions: | The system must have customer registered account information | | |
| Post-conditions: | **Success**: Display detailed information of any customer account  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer has not filled in all personal information in the profile section | | |
| Business Rules: | Only users with appropriate roles (e.g., Admin, Accountant) can view detailed account information.  Access to sensitive account details (e.g., transaction history, balances) is restricted based on user permissions.  All account details must be validated before being displayed, ensuring data consistency and accuracy.  Any updates to account details are subjected to validation checks to prevent data corruption. | | |
| Assumptions: | * **User Roles and Permissions:**   + The system has multiple user roles such as Admin, Manager, Accountant, and Employee.  + Only authorized users (e.g., Admins and Accountants) can access the View Accounts List feature.   * **Account Types:** The system categorizes accounts into different types such as Customer Accounts, Vendor Accounts, and Employee Payroll Accounts. * **Data Storage:** Account data is stored in a secure database with proper indexing to allow efficient querying and retrieval.   **Search and Filtering:** Users can search for accounts using various criteria such as account name, account ID, contact information, and balance range. | | |

### UC-20: Create Accounts

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-20: Create Accounts** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin creates accounts for customers | | |
| Description: | This function allows Admin to create accounts for customers | | |
| Preconditions: | The customer does not have an account on the system | | |
| Post-conditions: | **Success**: Display notification of successful account registration  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | The customer already has an account on the system | | |
| Business Rules: | * Only authorized users (e.g., Admins, Accountants) can create new accounts. * User roles and permissions are managed through the system's role management module. * Users must specify the account type (e.g., Customer, Vendor, Employee) during account creation. * Each account type has mandatory fields that must be completed before the account can be created. | | |
| Assumptions: | * **User Roles and Permissions:** The system has predefined user roles such as Admin, Manager, Accountant, and Employee. * **Account Types:** The system supports multiple account types, including Customer Accounts, Vendor Accounts, and Employee Accounts. * **Mandatory and Optional Fields:** There are mandatory fields for all account types, such as Account Name, Contact Information, and Account ID. * **Unique Identifiers:** Each account must have a unique Account ID, automatically generated by the system to prevent duplication. | | |

### UC-21: Modify Accounts

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-21: Modify Accounts** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can change customer accounts | | |
| Description: | This function allows Admin to modify information according to customer requests | | |
| Preconditions: | Customers must successfully register an account on the system | | |
| Post-conditions: | **Success**: Display notification of successful account information modification  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Customers do not send requests to modify information to Admin | | |
| Business Rules: | * Only authorized users (e.g., Admins, Accountants) can modify account information. * User roles and permissions are managed through the system’s role management module. * Users must provide a unique identifier (e.g., Account ID) to select the account to be modified. * Mandatory fields such as Account Name, Contact Information, and key identifiers must be correctly filled out before changes are saved. | | |
| Assumptions: | * **User Roles and Permissions:** The system has predefined user roles such as Admin, Manager, Accountant, and Employee. * **Audit Trail:** Audit logs include details of the user who made the change, the changes made, and the time of modification. * **Approval Workflow:** Certain modifications, such as changes to credit limits or account status, may require approval from higher authorities. * **Notification and Confirmation:** Upon successful modification of an account, the system sends a confirmation notification to the user. | | |

### UC-22: Disable Account

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-22: Disable Account** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can disable any customer account | | |
| Description: | This function allows Admin to disable any customer account that has stopped being used on the system | | |
| Preconditions: | The customer already has a system account | | |
| Post-conditions: | **Success**: The system will display a notification that the customer account has been successfully added  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Customers still use accounts on the system | | |
| Business Rules: | * User roles and permissions are managed through the system’s role management module. * Users must provide a unique identifier (e.g., Account ID) to select the account to be disabled. * Common reasons include inactivity, violation of terms, fraud detection, or account closure upon request. * The system sends a notification to the account holder informing them that their account has been disabled, including the reason for the action. | | |
| Assumptions: | * **User Roles and Permissions:** Only authorized users (e.g., Admins and Managers) have the permission to disable accounts. * **Account Identification:** Each account has a unique identifier (e.g., Account ID) that is used to select and disable the account. * **Reason for Disabling:** Common reasons include inactivity, violation of terms, fraud detection, or account closure upon request. * **Reporting:** Users with appropriate permissions can generate reports on disabled accounts, including details such as the number of disabled accounts, reasons for disabling, and the duration of the disabled status. | | |

### UC-23: Remove Sale Product

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-23: Remove Sale Product** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin will delete discounted products when the promotion period expires | | |
| Description: | This function allows Admin to delete discounted products from the system when the promotion period has expired | | |
| Preconditions: | There must be discounted products on the system | | |
| Post-conditions: | **Success**: Displays a notification that the sale product has been successfully removed  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no sale products on the system | | |
| Business Rules: | * User roles and permissions are managed through the system’s role management module. * Users must provide a unique identifier (e.g., Product ID) to select the product to be removed from the sale. * Common reasons include product discontinuation, stock issues, pricing errors, or promotional end. * Removing a product from sale updates the inventory system to reflect the change in availability or status. | | |
| Assumptions: | * **User Roles and Permissions:** The system has predefined user roles such as Admin, Manager, Salesperson, and Inventory Clerk. * **Product Identification:** Each product has a unique identifier (e.g., Product ID) that is used to select the product for removal from sale. * **Reason for Removal:** Common reasons include product discontinuation, stock issues, pricing errors, or end of a promotional period. * **Customer Orders and Reservations:** Customers are notified of the product's removal, and appropriate actions (e.g., substitutions, refunds) are taken. | | |

### UC-24: Modify Sale Product

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-24: Modify Sale Product** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can modify sale products | | |
| Description: | This function allows Admin to modify sale products on the system | | |
| Preconditions: | There must be discounted products on the system | | |
| Post-conditions: | **Success**: Displays a notification that the sale product has been successfully modified  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no sale products on the system that need to be modified | | |
| Business Rules: | * User roles and permissions are managed through the system’s role management module. * Users must provide a unique identifier (e.g., Product ID) to select the product to be removed from the sale. * Common reasons include product discontinuation, stock issues, pricing errors, or promotional end. * Modifying a product from sale updates the inventory system to reflect the change in availability or status. | | |
| Assumptions: | * **User Roles and Permissions:** Only authorized users (e.g., Admins and Managers) have the permission to modify sale product details. * **Product Identification:** Each product has a unique identifier (e.g., Product ID) that is used to select the product for modification. * **Editable Fields:** Common editable fields include product name, description, price, discount, stock quantity, and sale status. * **Pricing and Discounts:** Modifications to pricing and discounts comply with predefined business rules and promotional guidelines. | | |

### UC-25: Verify New Sale Request

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-25: Verify New Sale Request** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Admin | Secondary Actors: | Storekeeper |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Admin can verify new sales requests on the system | | |
| Description: | This function allows Admin to verify any new item to be allowed to post on the system | | |
| Preconditions: | Admin must receive the verification request | | |
| Post-conditions: | **Success**: Displays the request to verify the successfully sold product  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no new product verification requirements | | |
| Business Rules: | * Only authorized users (e.g., Managers, Salespersons) can verify new sale requests. * The system verifies that all products in the sale request are available in the inventory. * All fields in the sale request (e.g., product ID, quantity, price) must pass validation checks for accuracy and completeness. * The system verifies that the prices and discounts applied to the sale request comply with the bakery’s pricing policies and promotional guidelines. | | |
| Assumptions: | * **User Roles and Permissions:** Only authorized users (e.g., Managers and Salespersons) have the permission to verify new sale requests. * **Product Availability:** Products included in the sale request are currently available in the inventory. * **Data Accuracy:** The information provided in the sale request (e.g., product ID, quantity, price) is accurate and complete. * **Customer Information:** Customer information provided in the sale request (e.g., name, contact details) is accurate and complete. | | |

### UC-26: View own Sale Request

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-26: View own Sale Request** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Distributor allows sellers to review their own selling requests | | |
| Description: | This function allows Distributors to review their sales requests on the system | | |
| Preconditions: | Distributors have their sales requests on the system | | |
| Post-conditions: | **Success**: Displays the request to verify the successfully sold product  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Distributors do not have their sales requirements on the system | | |
| Business Rules: | * User roles and permissions are managed through the system’s role management module. * Users must be authenticated (e.g., login with valid credentials) to access their own sale requests. * Users can only view sale requests that they have personally created or submitted. * Users can view the status of their sale requests (e.g., pending, verified, processed, completed, cancelled). | | |
| Assumptions: | * **User Authentication:** Users are required to log in with valid credentials to access their sale requests. * **User Roles and Permissions:** Users have permissions set according to their roles, restricting access to their own sale requests only. * **Data Integrity:** The system performs validation checks to ensure data integrity before storing sale requests. * **Product Availability:** The system ensures that stock levels are current and available quantities are up-to-date. | | |

### UC-27: Create new Sale Product

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-27: Create new Sale Product** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | Hjgh | Frequency of Use: | High |
| Trigger: | Distributor can create a new sale product | | |
| Description: | This function allows Distributor to create a new sale product on the system | | |
| Preconditions: | Distributor has a new sale product that needs to be created on the system | | |
| Post-conditions: | **Success**: Display notification that a new sale product has been created  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Distributor has no sales products to create | | |
| Business Rules: | * User roles and permissions are managed through the system’s role management module. * Each new sale product must have a unique identifier (e.g., Product ID) generated by the system. * Users must complete all mandatory fields when creating a new sale product, including product name, description, category, price, stock quantity, and SKU. * The price of the new sale product must comply with the bakery’s pricing policies and guidelines. | | |
| Assumptions: | * **User Authentication:** Users creating new sale products are authenticated with valid credentials. * **User Roles and Permissions:** Only users with appropriate permissions (e.g., Admins, Managers) can create new sale products. * **Product Information:** Users have access to accurate and complete information about the new sale products they are creating. * **Unique Product Identification:** The system generates a unique identifier (e.g., Product ID) for each new sale product. | | |

### UC-28: Modify own Sale Product

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-28: Modify own Sale Product** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Distributor can Modify own Sale Product | | |
| Description: | This function allows Distributor can Modify own Sale Product on the system | | |
| Preconditions: | Distributor has its own sale product | | |
| Post-conditions: | **Success**: Display notification that a own sale product has been modified  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Distributor haven’t Modify own Sale Product | | |
| Business Rules: | * The system supports an approval process for modifications that impact operational or financial aspects. * Modifications to sale products are logged in an audit trail, including details of the user who made the changes and the timestamp. * Optional notifications can be sent to customers or stakeholders affected by product updates. * Regular security assessments and updates are performed to protect product data from unauthorized access and breaches. | | |
| Assumptions: | * **User Authentication:** Users modifying their own sale products are authenticated with valid credentials. * **User Roles and Permissions:** Users have appropriate permissions to modify their own sale products. * **Product Identification:** The system provides a clear interface for users to locate and access their own sale products. * **Product Information:** Users have access to accurate and up-to-date information about their sale products. | | |

### UC-29: Delete own Sale Product

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-29: Delete own Sale Product** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Distributor can Delete own Sale Product | | |
| Description: | This function allows Distributor can Delete own Sale Product | | |
| Preconditions: | Distributor must have own sale product | | |
| Post-conditions: | **Success**: Display notification that a own sale product has been deleted  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | Distributor haven’t Delete own Sale Product | | |
| Business Rules: | * Only authorized users with appropriate permissions (e.g., Admins, Managers) can delete their own sale products. * Users can select and delete their own sale products from a list of products they are authorized to manage. * Deleting a sale product requires user confirmation to prevent accidental deletion. * The system validates the deletion request to ensure it meets predefined criteria (e.g., user authorization, product status). | | |
| Assumptions: | * **User Authentication:** Users attempting to delete their own sale products are authenticated with valid credentials. * **User Authorization:** Admins, Managers, or designated users with deletion rights can initiate product deletions. * **Product Identification:** The system provides a clear interface for users to locate and select the product for deletion. * **Confirmation:** The system prompts users to confirm their intention to delete the product before proceeding. | | |

### UC-30: View Related Order List

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-30: View Related Order List** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Distributor can View Related Order List | | |
| Description: | This function allows Distributors to view their related orders | | |
| Preconditions: | The system must have orders related to the distributor's products | | |
| Post-conditions: | **Success**: The system will display orders related to the distributor's products  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no orders related to the distributor's products | | |
| Business Rules: | * Access to order lists is based on user roles and permissions (e.g., Admins, Managers, Salespersons). * The system allows users to search and filter order lists based on various parameters:   + Customer name or account ID.   + Order status (e.g., pending, fulfilled, cancelled).   + Order date or time frame. * Users can view detailed information for each order in the list, including: * Order ID and reference number. * Customer details (name, contact information). * Products ordered (names, quantities, prices). * Order status and fulfillment details (e.g., delivery schedule, pickup location). * The system ensures that sensitive customer information is protected in compliance with data privacy regulations. | | |
| Assumptions: | * **User Authentication:** Users accessing related order lists are authenticated with valid credentials. * **User Roles and Permissions:** Access to order lists is based on user roles and permissions (e.g., Admins, Managers, Salespersons). * **Order Filtering and Search:** The system allows users to filter and search order lists based on various criteria * **Data Security:** Sensitive customer information is protected in compliance with data privacy regulations. | | |

### UC-31: View Related Order Detail

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and Name:** | **UC-31: View Related Order Detail** | | |
| Created By: | Tran Anh Toan | Date Created: | 06/29/2024 |
| Primary Actor: | Distributor | Secondary Actors: | Admin |
| Priority: | High | Frequency of Use: | High |
| Trigger: | Distributor can view related order detail | | |
| Description: | This function allows distributors to view detailed information about orders in the list | | |
| Preconditions: | The system must have orders related to the distributor's products | | |
| Post-conditions: | **Success**: The system will display specific information about that order  **Fail**: The system notifies the corresponding error code | | |
| Normal Flow: | N/A | | |
| Alternative Flow: | N/A | | |
| Exceptions: | There are no orders related to the distributor's products | | |
| Business Rules: | * Access to order details is restricted based on user roles and permissions (e.g., Admins, Managers, Salespersons). * Personal customer information and payment details are safeguarded in compliance with data privacy regulations. * Order details are integrated with other modules such as Inventory Management and Customer Relationship Management (CRM). * Audit logs capture details such as the user who accessed the order detail and the timestamp of access. | | |

# 4. Non-Functional Requirement

**4.1. Usability**

- All text, messages should be written in English.

- The mobile app Interface should follow the Figma prototypes’ design.

- Each perspective of the app must load within 2 seconds.

- The user doesn’t need specialized skills to use the app.

- The error rate of users submitting their payment details at the checkout page mustn’t exceed 10 percent

- The app can be downloaded and used as long as there is internet connection.

- The user should be able to understand all the functions within 1hour of self-experiencing.

**4.2. Reliability**

- The availability of the system is over 85%.

- Mean Time Between Failures (MTBF) should be more than 125 hours.

- Mean Time To Repair (MTTR) should be less than 12 hours .

- Precision in the system’s output must be 100%.

- Response time for a transaction must be 5 seconds or less

- Capacity for the number of users the system can accommodate is at least 500 users.

- Resource utilization, such as memory, disk, communications, and so forth must be smoothie and with minimal error.

**4.3. Performance**

- System should return result for each request within 5 seconds

- Processing time is from 03 to 05 seconds for each frame.

# 5. Database

### 5.1. ERD

A diagram of a computer

Description automatically generated

### 5.2. Relational Schema

*A diagram of a product

Description automatically generated*

*-- Create Users Table*

*CREATE TABLE Users (*

*UserID INT PRIMARY KEY AUTO\_INCREMENT,*

*Username VARCHAR(50) NOT NULL UNIQUE,*

*Password VARCHAR(255) NOT NULL,*

*Role VARCHAR(50) NOT NULL*

*);*

*-- Create Products Table*

*CREATE TABLE Products (*

*ProductID INT PRIMARY KEY AUTO\_INCREMENT,*

*Name VARCHAR(100) NOT NULL,*

*Price DECIMAL(10, 2) NOT NULL*

*);*

*-- Create Orders Table*

*CREATE TABLE Orders (*

*OrderID INT PRIMARY KEY AUTO\_INCREMENT,*

*UserID INT,*

*OrderDate DATE NOT NULL,*

*FOREIGN KEY (UserID) REFERENCES Users(UserID)*

*);*

*-- Create OrderItems Table*

*CREATE TABLE OrderItems (*

*OrderItemID INT PRIMARY KEY AUTO\_INCREMENT,*

*OrderID INT,*

*ProductID INT,*

*Quantity INT NOT NULL,*

*FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),*

*FOREIGN KEY (ProductID) REFERENCES Products(ProductID)*

*);*

*-- Create Feedback Table*

*CREATE TABLE Feedback (*

*FeedbackID INT PRIMARY KEY AUTO\_INCREMENT,*

*UserID INT,*

*ProductID INT,*

*Rating INT CHECK (Rating BETWEEN 1 AND 5),*

*FOREIGN KEY (UserID) REFERENCES Users(UserID),*

*FOREIGN KEY (ProductID) REFERENCES Products(ProductID)*

*);*

*-- Create Transactions Table*

*CREATE TABLE Transactions (*

*TransactionID INT PRIMARY KEY AUTO\_INCREMENT,*

*OrderID INT,*

*PaymentMethod VARCHAR(50) NOT NULL,*

*FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)*

*);*