项目文档

# Functional Requirement

# Chapter 1: Functional Requirements   
## 1.1 Customer Registration Function   
\*\*Function ID\*\*: FR-01   
\*\*Description\*\*: Customers can create new accounts by providing personal information. The system validates the input and stores it securely.   
\*\*Input\*\*: Personal information including name, email, password, and optional fields like address and phone number.   
\*\*Output\*\*: A new Customer data entity with the provided details and a confirmation message.   
  
## 1.2 Customer Login Function   
\*\*Function ID\*\*: FR-02   
\*\*Description\*\*: Customers or administrators can log in using their account credentials. The system validates the credentials and creates a session.   
\*\*Input\*\*: Username and password.   
\*\*Output\*\*: A valid LoginSession data entity and redirection to the appropriate dashboard.   
  
## 1.3 Customer Logout Function   
\*\*Function ID\*\*: FR-03   
\*\*Description\*\*: Customers or administrators can terminate their session. The system updates the session status and redirects to the login or home page.   
\*\*Input\*\*: Request to end the session.   
\*\*Output\*\*: A terminated LoginSession data entity and redirection to login/home page.   
  
## 1.4 Customer Profile Management Function   
\*\*Function ID\*\*: FR-04   
\*\*Description\*\*: Customers can view and edit their profile information. The system updates the Customer data entity if changes are made.   
\*\*Input\*\*: Updated profile information such as name, address, or phone number.   
\*\*Output\*\*: Updated Customer data entity and a confirmation message.   
  
## 1.5 Product Search Function   
\*\*Function ID\*\*: FR-05   
\*\*Description\*\*: Customers can search for products using keywords or product names. The system matches the query with the Product data entity and displays relevant results.   
\*\*Input\*\*: Search term or keyword.   
\*\*Output\*\*: A list of matching products from the Product data entity.   
  
## 1.6 Product Filtering Function   
\*\*Function ID\*\*: FR-06   
\*\*Description\*\*: Customers can filter products by category. The system retrieves products from the Product data entity and filters them based on the selected Category.   
\*\*Input\*\*: Selected category ID or name.   
\*\*Output\*\*: A list of filtered products from the Product data entity.   
  
## 1.7 Product View Function   
\*\*Function ID\*\*: FR-07   
\*\*Description\*\*: Customers can view detailed information about a specific product. The system retrieves the product data from the Product data entity and displays it.   
\*\*Input\*\*: Product ID or name.   
\*\*Output\*\*: Displayed product details such as description, price, image, and availability.   
  
## 1.8 Shopping Cart Management Function   
\*\*Function ID\*\*: FR-08   
\*\*Description\*\*: Customers can add, remove, or modify items in their shopping cart. The system updates the ShoppingCart data entity accordingly.   
\*\*Input\*\*: Product ID, quantity, and action (add, remove, or modify).   
\*\*Output\*\*: Updated ShoppingCart data entity and a summary of the cart contents.   
  
## 1.9 Proceed to Checkout Function   
\*\*Function ID\*\*: FR-09   
\*\*Description\*\*: Customers can proceed from the shopping cart to the checkout page. The system prepares the cart for order creation.   
\*\*Input\*\*: Request to proceed to checkout.   
\*\*Output\*\*: Redirected to the payment processing page and preparation of the ShoppingCart for order creation.   
  
## 1.10 Payment Information Management Function   
\*\*Function ID\*\*: FR-10   
\*\*Description\*\*: Customers can enter and save their payment information. The system validates the input and stores it securely in the PaymentInformation data entity.   
\*\*Input\*\*: Card number, expiration date, CVV, and billing address.   
\*\*Output\*\*: Updated PaymentInformation data entity and a confirmation message.   
  
## 1.11 Order Placement Function   
\*\*Function ID\*\*: FR-11   
\*\*Description\*\*: Customers can place an order using the items in their shopping cart and selected payment information. The system creates a new Order and links it to the Customer and PaymentInformation data entities.   
\*\*Input\*\*: ShoppingCart ID and selected PaymentInformation.   
\*\*Output\*\*: New Order data entity and confirmation message.   
  
## 1.12 Order Confirmation Function   
\*\*Function ID\*\*: FR-12   
\*\*Description\*\*: Customers can confirm their order after checkout. The system updates the Order data entity to reflect the confirmed status.   
\*\*Input\*\*: Order ID and confirmation request.   
\*\*Output\*\*: Updated Order data entity and a confirmation message.   
  
## 1.13 Order Cancellation Function   
\*\*Function ID\*\*: FR-13   
\*\*Description\*\*: Customers can cancel an order if it is in a cancellable state. The system updates the Order and ShoppingCart data entities.   
\*\*Input\*\*: Order ID and cancellation request.   
\*\*Output\*\*: Updated Order data entity and, if applicable, updated ShoppingCart data entity.   
  
## 1.14 Order Status Viewing Function   
\*\*Function ID\*\*: FR-14   
\*\*Description\*\*: Customers can view the current status of their orders. The system retrieves the Order data entity and displays the status.   
\*\*Input\*\*: Order ID or list of orders.   
\*\*Output\*\*: Displayed order status (e.g., processing, shipped, delivered, canceled).   
  
## 1.15 Product Purchase History Function   
\*\*Function ID\*\*: FR-15   
\*\*Description\*\*: Customers can view their purchase history. The system retrieves and displays a list of past orders and associated products.   
\*\*Input\*\*: Customer ID and request to view purchase history.   
\*\*Output\*\*: A list of previous orders and their details from the Order and OrderDetail data entities.   
  
## 1.16 Product Addition Function   
\*\*Function ID\*\*: FR-16   
\*\*Description\*\*: Administrators can add new products to the system, including assigning them to a category. The system updates the Product and Category data entities.   
\*\*Input\*\*: Product details such as name, description, price, category, and image.   
\*\*Output\*\*: New Product data entity and updated Category data entity.   
  
## 1.17 Product Update Function   
\*\*Function ID\*\*: FR-17   
\*\*Description\*\*: Administrators can modify existing product information. The system updates the Product and Category data entities.   
\*\*Input\*\*: Product ID and updated details.   
\*\*Output\*\*: Updated Product data entity and, if applicable, updated Category data entity.   
  
## 1.18 Product Deletion Function   
\*\*Function ID\*\*: FR-18   
\*\*Description\*\*: Administrators can delete a product from the system. The system removes the product from the Product and Category data entities.   
\*\*Input\*\*: Product ID and deletion request.   
\*\*Output\*\*: Updated Product and Category data entities and a confirmation message.   
  
## 1.19 Category Management Function   
\*\*Function ID\*\*: FR-19   
\*\*Description\*\*: Administrators can create, update, or delete product categories. The system updates the Category data entity.   
\*\*Input\*\*: Category details such as name and description.   
\*\*Output\*\*: Updated Category data entity and a confirmation message.   
  
## 1.20 Order Review and Management Function   
\*\*Function ID\*\*: FR-20   
\*\*Description\*\*: Administrators can review and manage existing orders. The system updates the Order data entity and may notify the customer of changes.   
\*\*Input\*\*: Order ID and requested action (update, cancel, ship, etc.).   
\*\*Output\*\*: Updated Order data entity and, if applicable, a notification to the customer.   
  
## 1.21 Order Detail Management Function   
\*\*Function ID\*\*: FR-21   
\*\*Description\*\*: Administrators can view and modify details of a specific order, such as product quantity or price. The system updates the OrderDetail data entity.   
\*\*Input\*\*: OrderDetail ID and modified information.   
\*\*Output\*\*: Updated OrderDetail data entity and a confirmation message.   
  
## 1.22 Inventory Management Function   
\*\*Function ID\*\*: FR-22   
\*\*Description\*\*: Administrators can manage product inventory levels. The system updates the Inventory data entity and adjusts the Product data entity if necessary.   
\*\*Input\*\*: Product ID and updated stock quantity.   
\*\*Output\*\*: Updated Inventory data entity and, if applicable, updated Product data entity.   
  
## 1.23 Plugin Installation Function   
\*\*Function ID\*\*: FR-23   
\*\*Description\*\*: Administrators can install new plugins to extend system functionality. The system updates the Plugin data entity.   
\*\*Input\*\*: Plugin ID or name.   
\*\*Output\*\*: Updated Plugin data entity and a confirmation message.   
  
## 1.24 Plugin Configuration Function   
\*\*Function ID\*\*: FR-24   
\*\*Description\*\*: Administrators can configure plugin settings to customize their behavior. The system updates the PluginSetting data entity.   
\*\*Input\*\*: Plugin ID and configuration key-value pairs.   
\*\*Output\*\*: Updated PluginSetting data entity and a confirmation message.   
  
## 1.25 Plugin Uninstallation Function   
\*\*Function ID\*\*: FR-25   
\*\*Description\*\*: Administrators can uninstall plugins when they are no longer needed. The system updates the Plugin data entity.   
\*\*Input\*\*: Plugin ID and uninstallation request.   
\*\*Output\*\*: Updated Plugin data entity and a confirmation message.   
  
## 1.26 Email Service Configuration Function   
\*\*Function ID\*\*: FR-26   
\*\*Description\*\*: Administrators can configure the email service used for sending order confirmations. The system updates the EmailService data entity.   
\*\*Input\*\*: Email service settings such as SMTP server, port, and authentication.   
\*\*Output\*\*: Updated EmailService data entity and a confirmation message.   
  
## 1.27 Email Notification Function   
\*\*Function ID\*\*: FR-27   
\*\*Description\*\*: The system sends order confirmation emails to customers after an order is placed and confirmed. The system updates the EmailService data entity and logs the activity.   
\*\*Input\*\*: Order details such as customer name, product list, and total amount.   
\*\*Output\*\*: Sent email notification and updated EmailService data entity.   
  
## 1.28 User Account Management Function   
\*\*Function ID\*\*: FR-28   
\*\*Description\*\*: Administrators can manage customer and administrator accounts, including creation, updates, and deletion. The system updates the Customer or Administrator data entities.   
\*\*Input\*\*: User account details such as name, email, password, and role.   
\*\*Output\*\*: Updated Customer or Administrator data entity and a confirmation message.   
  
## 1.29 System Documentation Access Function   
\*\*Function ID\*\*: FR-29   
\*\*Description\*\*: Users can access and view system documentation for plugins and other features. The system retrieves and displays the DocumentReference data entity.   
\*\*Input\*\*: Request to view documentation.   
\*\*Output\*\*: Displayed documentation content and logged access activity.   
  
## 1.30 Registration Record Management Function   
\*\*Function ID\*\*: FR-30   
\*\*Description\*\*: Administrators can view, update, or delete customer registration records. The system updates the RegistrationRecord data entity.   
\*\*Input\*\*: Registration record details such as email, status, and registration date.   
\*\*Output\*\*: Updated or deleted RegistrationRecord data entity and a confirmation message.   
  
## 1.31 Browser Access Control Function   
\*\*Function ID\*\*: FR-31   
\*\*Description\*\*: Administrators can configure access control settings for supported browsers. The system updates the Browser and SystemAccessControl data entities.   
\*\*Input\*\*: Selected browser types and access rules (allowed or restricted).   
\*\*Output\*\*: Updated Browser and SystemAccessControl data entities and a confirmation message.   
  
## 1.32 Browsing History Tracking Function   
\*\*Function ID\*\*: FR-32   
\*\*Description\*\*: The system tracks customer browsing behavior, including pages visited and time spent. The system updates the BrowsingHistory data entity.   
\*\*Input\*\*: Page URL and timestamp.   
\*\*Output\*\*: Updated BrowsingHistory data entity and logged activity.   
  
## 1.33 Product View Tracking Function   
\*\*Function ID\*\*: FR-33   
\*\*Description\*\*: The system tracks when customers view product details. The system updates the ProductView data entity.   
\*\*Input\*\*: Product ID and timestamp.   
\*\*Output\*\*: Updated ProductView data entity and logged activity.   
  
## 1.34 Search Query Tracking Function   
\*\*Function ID\*\*: FR-34   
\*\*Description\*\*: The system tracks customer search queries for analytics and improvement. The system updates the SearchQuery data entity.   
\*\*Input\*\*: Search term and timestamp.   
\*\*Output\*\*: Updated SearchQuery data entity and logged activity.   
  
## 1.35 Notification Management Function   
\*\*Function ID\*\*: FR-35   
\*\*Description\*\*: The system sends notifications to customers for order status changes. The system updates the NotificationLog data entity.   
\*\*Input\*\*: Message content and timestamp.   
\*\*Output\*\*: Updated NotificationLog data entity and sent notification to the customer.

# External Description

# Chapter 2: External Interfaces  
  
This chapter describes the external interfaces of the system, categorized into \*\*User Interface\*\*, \*\*Hardware Interface\*\*, \*\*Software Interface\*\*, and \*\*Communication Interface\*\*. These interfaces facilitate interaction between the system and its users, external systems, hardware, and communication channels.  
  
---  
  
## 2.1 User Interface Output  
  
### 2.1.1 Web-Based User Interface  
\*\*Description\*\*: The system provides a web-based interface accessible through modern web browsers (e.g., Chrome, Firefox, Safari) for both customers and administrators to interact with the system.  
  
\*\*Interaction Method\*\*:  
- Customers use this interface to register, log in, search products, manage shopping carts, place orders, view order status, and access purchase history.  
- Administrators use this interface to manage products, categories, orders, plugins, email services, user accounts, registration records, browser access control, and documentation.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Form data such as personal information, product details, category names, order actions, plugin settings, etc.  
- \*\*Outputs\*\*: Confirmation messages, dashboards, product listings, order summaries, error messages, and other UI elements.  
  
---  
  
## 2.2 Hardware Interface Output  
  
### 2.2.1 Browser Compatibility Interface  
\*\*Description\*\*: The system supports standard web browsers and adapts to various screen sizes and resolutions.  
  
\*\*Interaction Method\*\*:  
- The system dynamically adjusts layout and functionality based on browser type and device capabilities.  
- Administrators can configure which browsers are allowed or restricted via the Browser Access Control function.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Browser types and versions selected by users.  
- \*\*Outputs\*\*: Updated Browser and SystemAccessControl data entities reflecting allowed/restricted browser configurations.  
  
---  
  
## 2.3 Software Interface Output  
  
### 2.3.1 Database Interface  
\*\*Description\*\*: The system interacts with a relational database to store and retrieve structured data including customer profiles, product inventories, orders, plugins, and logs.  
  
\*\*Data Entities Involved\*\*:  
- Customer  
- LoginSession  
- ShoppingCart  
- Product  
- Category  
- Order  
- PaymentInformation  
- Plugin  
- EmailService  
- NotificationLog  
- BrowsingHistory  
- SearchQuery  
- ProductView  
- RegistrationRecord  
  
\*\*Interaction Method\*\*:  
- The system performs CRUD (Create, Read, Update, Delete) operations on these entities as required by functional requirements.  
- Data is stored securely with appropriate validation and encryption.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Structured data from forms, API calls, or internal processes.  
- \*\*Outputs\*\*: Updated or retrieved database records.  
  
---  
  
### 2.3.2 Plugin Interface  
\*\*Description\*\*: The system supports third-party plugins to extend functionality. It allows installation, configuration, and uninstallation of plugins.  
  
\*\*Interaction Method\*\*:  
- Plugins interact with the system via defined APIs and data structures.  
- Plugin settings are stored in the PluginSetting data entity.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Plugin ID/name, configuration key-value pairs.  
- \*\*Outputs\*\*: Updated Plugin and PluginSetting data entities.  
  
---  
  
### 2.3.3 Documentation Interface  
\*\*Description\*\*: The system provides access to internal documentation resources for plugins and features.  
  
\*\*Interaction Method\*\*:  
- Users request documentation via the UI, and the system retrieves and displays the DocumentReference data entity.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Request to view documentation.  
- \*\*Outputs\*\*: Displayed documentation content and logged access activity.  
  
---  
  
## 2.4 Communication Interface Output  
  
### 2.4.1 Email Service Interface  
\*\*Description\*\*: The system integrates with an email service to send notifications related to orders, account activities, and confirmations.  
  
\*\*Interaction Method\*\*:  
- Administrators configure the email service using SMTP settings.  
- The system sends emails upon specific events like order confirmation and status updates.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Email service settings (SMTP server, port, authentication), order details.  
- \*\*Outputs\*\*: Sent emails, updated EmailService data entity, and activity logs.  
  
---  
  
### 2.4.2 Notification Service Interface  
\*\*Description\*\*: The system sends real-time notifications to users regarding order status changes and administrative actions.  
  
\*\*Interaction Method\*\*:  
- Notifications are generated internally and pushed to users via supported notification channels (e.g., in-app alerts, SMS, or push notifications if integrated).  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Message content and timestamp.  
- \*\*Outputs\*\*: Sent notification, updated NotificationLog data entity.  
  
---  
  
### 2.4.3 Analytics and Tracking Interface  
\*\*Description\*\*: The system tracks user behavior for analytics purposes, including browsing history, product views, and search queries.  
  
\*\*Interaction Method\*\*:  
- The system logs interactions and stores them in respective tracking entities for later analysis.  
  
\*\*Inputs/Outputs\*\*:  
- \*\*Inputs\*\*: Page URL, product ID, search term, and timestamp.  
- \*\*Outputs\*\*: Updated BrowsingHistory, ProductView, and SearchQuery data entities.  
  
---  
  
## Summary Table of External Interfaces  
  
| Interface Type | Interface Name | Description |  
|-----------------------|-------------------------------------|-----------------------------------------------------------------------------|  
| User Interface | Web-Based User Interface | Primary interface for customer and admin interaction |  
| Hardware Interface | Browser Compatibility Interface | Supports multiple browsers and device types |  
| Software Interface | Database Interface | Manages persistent storage and retrieval of all system data |  
| Software Interface | Plugin Interface | Enables extension of system functionality via third-party modules |  
| Software Interface | Documentation Interface | Provides access to internal documentation |  
| Communication Interface | Email Service Interface | Sends transactional and informational emails |  
| Communication Interface | Notification Service Interface | Delivers real-time updates to users |  
| Communication Interface | Analytics and Tracking Interface | Logs user behavior for monitoring and improvement |  
  
---  
  
All external interfaces have been mapped directly from the functional requirements to ensure alignment and completeness. Developers should refer to this section for implementation guidance on integrating with external components and managing system interactions.

# Use Case

Use Case Name: Register Customer Account   
Use Case ID: UC-01   
Actors: Customer, Browser, Plugin, Administrator   
Preconditions:   
1. The customer has access to the system via the browser.   
2. The system is operational and accessible.   
3. The customer does not have an existing account.   
  
Postconditions:   
1. A new customer account is successfully created in the system.   
2. The customer is notified of successful registration.   
3. The system updates the Customer data entity with the new account details.   
  
Main Flow:   
1. The customer opens the browser and navigates to the registration page using a plugin.   
2. The system displays the registration form.   
3. The customer enters their personal information (e.g., name, email, password).   
4. The customer submits the registration form.   
5. The system validates the input data (e.g., email format, password strength).   
6. The system checks if the email is already registered.   
7. If the email is not registered, the system creates a new Customer data entity.   
8. The system sends a confirmation message to the customer.   
9. The customer receives the confirmation and is redirected to the login page.   
  
Alternative Flow:   
1. If the email is already registered, the system displays an error message and prompts the customer to use a different email.   
2. If the input data is invalid, the system displays an error message and prompts the customer to correct the information.   
3. If the system is under maintenance, the administrator blocks the registration process, and the customer receives a message indicating that registration is temporarily unavailable.  
  
Use Case Name: Manage Customer Profile   
Use Case ID: UC-02   
Actors: Customer, Administrator, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has navigated to the profile management section.   
  
Postconditions:   
1. The customer's profile information is updated or viewed in the system.   
2. The system logs the activity for audit purposes.   
3. If changes are made, the system updates the Customer data entity accordingly.   
  
Main Flow:   
1. The customer logs in and accesses their profile through the browser.   
2. The system displays the customer's current profile information.   
3. The customer selects the "Edit Profile" option using the browser.   
4. The system loads the editable profile form.   
5. The customer modifies the desired information (e.g., address, phone number, preferences).   
6. The customer submits the updated profile.   
7. The system validates the updated data (e.g., phone number format).   
8. The system updates the Customer data entity with the new information.   
9. The system provides a confirmation message that the profile has been successfully updated.   
  
Alternative Flow:   
1. If the customer does not have permission to edit their profile, the system displays an access denied message.   
2. If the input data is invalid, the system displays an error message and prompts the customer to correct the information.   
3. If the system is under maintenance, the administrator blocks profile modifications, and the customer receives a message indicating that profile updates are temporarily unavailable.  
  
Use Case Name: View Purchase History   
Use Case ID: UC-03   
Actors: Customer, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has navigated to the purchase history section.   
  
Postconditions:   
1. The customer's purchase history is displayed in the system.   
2. The system logs the activity for audit purposes.   
3. The system retrieves and presents relevant Order and Product data entities to the customer.   
  
Main Flow:   
1. The customer logs in and accesses the purchase history section through the browser.   
2. The system displays a list of previous orders.   
3. The customer selects a specific order to view more details.   
4. The system retrieves the selected order information from the Order data entity.   
5. The system displays the order details, including products, quantities, and prices.   
6. The system provides a summary of total purchases and other relevant statistics.   
  
Alternative Flow:   
1. If the customer has no purchase history, the system displays a message indicating that no orders have been placed.   
2. If the system cannot retrieve the order information, it displays an error message and logs the issue for administrator review.   
3. If the system is under maintenance, the administrator blocks access to the purchase history section, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Save Payment Information   
Use Case ID: UC-04   
Actors: Customer, Browser, Plugin, Payment Information, Shopping Cart   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The customer has added at least one product to the shopping cart.   
3. The system is accessible via the browser.   
4. The customer has navigated to the checkout or payment information section.   
  
Postconditions:   
1. The customer's payment information is stored securely in the system.   
2. The system updates the Payment Information data entity with the new details.   
3. The customer is notified that the payment information has been successfully saved.   
4. The shopping cart remains intact for future use or is cleared after order completion.   
  
Main Flow:   
1. The customer accesses the checkout page through the browser.   
2. The system displays the payment information form.   
3. The customer enters their payment details (e.g., card number, expiration date, CVV).   
4. The customer selects the "Save Payment Information" option.   
5. The system validates the entered payment data.   
6. The system securely saves the information in the Payment Information data entity.   
7. The system confirms the successful saving of payment information and displays it for future reference.   
  
Alternative Flow:   
1. If the entered payment information is invalid, the system displays an error message and prompts the customer to correct the data.   
2. If the system detects duplicate payment information, it displays a message asking the customer if they want to update the existing entry or add a new one.   
3. If the system is under maintenance, the payment information saving process is temporarily blocked, and the customer is informed accordingly.  
  
Use Case Name: Login to System   
Use Case ID: UC-05   
Actors: Customer, Administrator, Browser, Plugin   
  
Preconditions:   
1. The customer or administrator has access to the system via the browser.   
2. The system is operational and accessible.   
3. The customer or administrator has an existing account.   
  
Postconditions:   
1. The customer or administrator is successfully logged into the system.   
2. The system updates the session status for the user.   
3. The user is redirected to their respective dashboard or homepage.   
  
Main Flow:   
1. The user opens the browser and navigates to the login page using a plugin.   
2. The system displays the login form.   
3. The user enters their username and password.   
4. The user submits the login form.   
5. The system validates the entered credentials against the stored information in the Customer or Administrator data entity.   
6. If the credentials are valid, the system creates a session and logs the login activity.   
7. The system redirects the user to their dashboard or homepage.   
  
Alternative Flow:   
1. If the entered credentials are invalid, the system displays an error message and prompts the user to re-enter their username and password.   
2. If the user account is locked or disabled, the system displays a message indicating that the account is not active.   
3. If the system is under maintenance, the administrator blocks the login process, and the user receives a message indicating that the system is temporarily unavailable.  
  
Use Case Name: Logout from System   
Use Case ID: UC-06   
Actors: Customer, Administrator, Browser, Plugin   
  
Preconditions:   
1. The customer or administrator has an active session in the system.   
2. The system is accessible via the browser.   
3. The user has navigated to the logout option or clicked on the logout button.   
  
Postconditions:   
1. The user's session is terminated.   
2. The system logs the logout activity for audit purposes.   
3. The user is redirected to the login or homepage with no active session.   
  
Main Flow:   
1. The customer or administrator clicks on the "Logout" option in the system interface through the browser.   
2. The system prompts for confirmation to proceed with logout.   
3. The user confirms the logout action.   
4. The system terminates the current session.   
5. The system logs the logout activity.   
6. The user is redirected to the login page or homepage.   
  
Alternative Flow:   
1. If the user cancels the logout confirmation, the system retains the current session and does not perform any logout action.   
2. If the system is under maintenance, the administrator may disable the logout function temporarily, and the user receives a message indicating that the feature is unavailable.  
  
Use Case Name: Manage Products   
Use Case ID: UC-07   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the product management section.   
  
Postconditions:   
1. The product information is updated, added, or removed in the system.   
2. The system updates the Product and Category data entities accordingly.   
3. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the product management interface through the browser.   
2. The system displays a list of existing products and categories.   
3. The administrator selects an option to add, edit, or delete a product.   
4. The system loads the appropriate form based on the selected option.   
5. The administrator enters or modifies product details (e.g., name, description, price, category, image).   
6. The administrator submits the form.   
7. The system validates the input data (e.g., price format, required fields).   
8. The system updates the Product and Category data entities with the new or modified information.   
9. The system provides a confirmation message that the product has been successfully managed.   
  
Alternative Flow:   
1. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the administrator attempts to delete a product that is linked to existing orders, the system displays a warning message and prompts for confirmation.   
3. If the system is under maintenance, the administrator is informed that product management is temporarily unavailable.  
  
Use Case Name: Add New Product   
Use Case ID: UC-08   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the product management section and selected the "Add New Product" option.   
4. The necessary categories are already defined in the system.   
  
Postconditions:   
1. A new product is successfully added to the system.   
2. The system updates the Product and Category data entities with the new product information.   
3. The administrator is notified of the successful addition.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the "Add New Product" interface through the browser.   
2. The system displays the product creation form with fields for product details (e.g., name, price, description, category, image).   
3. The administrator fills in the product information and selects the appropriate category.   
4. The administrator submits the form.   
5. The system validates the input data (e.g., required fields, correct data types).   
6. The system checks if the selected category exists.   
7. The system creates a new Product data entity and links it to the selected Category data entity.   
8. The system confirms the successful addition of the product and displays it in the product list.   
  
Alternative Flow:   
1. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the selected category does not exist, the system displays an error and prompts the administrator to choose a valid category.   
3. If the system is under maintenance, the administrator is informed that adding new products is temporarily unavailable.  
  
Use Case Name: Update Product Details   
Use Case ID: UC-09   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the product management section and selected a specific product to update.   
4. The selected product exists in the system.   
  
Postconditions:   
1. The product details are successfully updated in the system.   
2. The system updates the Product and Category data entities with the new information.   
3. The administrator is notified of the successful update.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the product management section through the browser.   
2. The system displays a list of products.   
3. The administrator selects a product to update.   
4. The system loads the product edit form with current details.   
5. The administrator modifies product information (e.g., name, price, description, category, image).   
6. The administrator submits the updated product form.   
7. The system validates the updated data (e.g., required fields, correct data types).   
8. The system updates the Product and Category data entities with the new information.   
9. The system confirms the successful update of the product details.   
  
Alternative Flow:   
1. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the selected category does not exist, the system displays an error and prompts the administrator to choose a valid category.   
3. If the system is under maintenance, the administrator is informed that product updates are temporarily unavailable.  
  
Use Case Name: Delete Product   
Use Case ID: UC-10   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the product management section.   
4. A product to be deleted is already present in the system.   
  
Postconditions:   
1. The selected product is successfully deleted from the system.   
2. The system updates the Product and Category data entities to reflect the deletion.   
3. The administrator is notified of the successful deletion.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the product management section through the browser.   
2. The system displays a list of products.   
3. The administrator selects a product to delete.   
4. The system prompts for confirmation to proceed with the deletion.   
5. The administrator confirms the deletion.   
6. The system checks if the product is associated with any existing orders.   
7. If no association is found, the system removes the Product data entity and updates the Category data entity if necessary.   
8. The system provides a confirmation message that the product has been successfully deleted.   
  
Alternative Flow:   
1. If the product is linked to existing orders, the system displays a warning message and prompts the administrator to confirm the deletion despite the association.   
2. If the administrator cancels the deletion confirmation, the system retains the product and does not perform any deletion action.   
3. If the system is under maintenance, the administrator is informed that product deletion is temporarily unavailable.  
  
Use Case Name: Assign Product to Category   
Use Case ID: UC-11   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the product or category management section.   
4. The product to be assigned is already present in the system.   
5. At least one category is already defined in the system.   
  
Postconditions:   
1. The product is successfully assigned to the selected category.   
2. The system updates the Product and Category data entities to reflect the new relationship.   
3. The administrator is notified of the successful assignment.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the product management section through the browser.   
2. The system displays a list of products with their current category assignments.   
3. The administrator selects a product to assign to a new or additional category.   
4. The system displays a form for selecting the category or categories for the product.   
5. The administrator chooses the appropriate category from the dropdown or search list.   
6. The administrator submits the form.   
7. The system validates the selected category and confirms the product exists.   
8. The system updates the Product data entity to link it with the selected Category data entity.   
9. The system provides a confirmation message that the product has been successfully assigned to the category.   
  
Alternative Flow:   
1. If the selected category does not exist, the system displays an error and prompts the administrator to choose a valid category.   
2. If the system detects duplicate assignments to the same category, it displays a message and allows the administrator to confirm or cancel the action.   
3. If the system is under maintenance, the administrator is informed that assigning products to categories is temporarily unavailable.  
  
Use Case Name: Create Product Category   
Use Case ID: UC-12   
Actors: Administrator, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the category management section.   
4. No category with the same name already exists in the system.   
  
Postconditions:   
1. A new product category is successfully created in the system.   
2. The system updates the Category data entity with the new category details.   
3. The administrator is notified of the successful creation.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the "Create Product Category" section through the browser.   
2. The system displays the category creation form with fields such as category name, description, and parent category (if applicable).   
3. The administrator enters the category details.   
4. The administrator submits the form.   
5. The system validates the input data (e.g., required fields, unique name).   
6. The system creates a new Category data entity with the provided information.   
7. The system provides a confirmation message that the category has been successfully created.   
  
Alternative Flow:   
1. If the category name already exists, the system displays an error message and prompts the administrator to choose a different name.   
2. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
3. If the system is under maintenance, the administrator is informed that category creation is temporarily unavailable.  
  
Use Case Name: Manage Product Categories   
Use Case ID: UC-13   
Actors: Administrator, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the category management section.   
  
Postconditions:   
1. Product categories are successfully created, updated, or deleted in the system.   
2. The system updates the Category data entity accordingly.   
3. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the category management section through the browser.   
2. The system displays a list of existing product categories.   
3. The administrator selects an action (e.g., create, update, delete) for a category.   
4. The system loads the appropriate form or confirmation prompt based on the selected action.   
5. If creating, the administrator enters the category name, description, and parent category (if applicable).   
6. If updating, the administrator modifies the selected category's details.   
7. If deleting, the administrator confirms the deletion of the selected category.   
8. The system validates the input data (e.g., required fields, unique name for creation).   
9. The system updates the Category data entity based on the selected action.   
10. The system provides a confirmation message that the category has been successfully managed.   
  
Alternative Flow:   
1. If the category name already exists when attempting to create a new one, the system displays an error and prompts the administrator to choose a different name.   
2. If the input data is invalid during creation or update, the system displays an error message and prompts the administrator to correct the information.   
3. If the administrator attempts to delete a category with associated products, the system displays a warning and prompts for confirmation.   
4. If the system is under maintenance, the administrator is informed that category management is temporarily unavailable.  
  
Use Case Name: Delete Category   
Use Case ID: UC-14   
Actors: Administrator, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the category management section.   
4. A category to be deleted is already present in the system.   
  
Postconditions:   
1. The selected category is successfully deleted from the system.   
2. The system updates the Category data entity to reflect the deletion.   
3. The system removes the association of the category with any products (if applicable).   
4. The administrator is notified of the successful deletion.   
5. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the category management section through the browser.   
2. The system displays a list of existing categories.   
3. The administrator selects a category to delete.   
4. The system prompts for confirmation to proceed with the deletion.   
5. The administrator confirms the deletion.   
6. The system checks if the category is associated with any products.   
7. If no association is found, the system removes the Category data entity.   
8. The system provides a confirmation message that the category has been successfully deleted.   
  
Alternative Flow:   
1. If the category is associated with products, the system displays a warning and asks if the administrator wants to proceed with deletion.   
2. If the administrator cancels the deletion confirmation, the system retains the category and does not perform any deletion action.   
3. If the system is under maintenance, the administrator is informed that category deletion is temporarily unavailable.  
  
Use Case Name: Place Order   
Use Case ID: UC-15   
Actors: Customer, Product, Order, Shopping Cart, Payment Information, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The customer has added at least one product to the shopping cart.   
3. The system is accessible via the browser.   
4. The customer has navigated to the checkout or order placement section.   
  
Postconditions:   
1. A new Order data entity is created in the system.   
2. The shopping cart is cleared or updated.   
3. The customer is notified of the successful order placement.   
4. The system records the associated Payment Information and links it to the order.   
5. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The customer logs in and accesses the shopping cart through the browser.   
2. The system displays the cart with the selected products and their details.   
3. The customer proceeds to checkout by clicking the "Place Order" button.   
4. The system prompts the customer to enter or select payment information.   
5. The customer selects or enters the payment method.   
6. The system validates the payment information and confirms the product availability.   
7. The system creates a new Order data entity and links it to the Customer and Payment Information data entities.   
8. The system updates the Shopping Cart data entity by removing the ordered items.   
9. The system sends a confirmation message to the customer.   
  
Alternative Flow:   
1. If the payment information is invalid, the system displays an error message and prompts the customer to correct the data.   
2. If the selected product is out of stock, the system displays a warning and allows the customer to modify the order.   
3. If the system is under maintenance, the administrator blocks the order placement process, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Confirm Order   
Use Case ID: UC-16   
Actors: Customer, Order, Shopping Cart, Payment Information, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The customer has placed an order (UC-15).   
3. The system is accessible via the browser.   
4. The customer has navigated to the order confirmation section.   
  
Postconditions:   
1. The order is confirmed and finalized in the system.   
2. The Order data entity is updated to reflect the confirmed status.   
3. The Shopping Cart data entity is cleared of the confirmed items.   
4. The customer is notified of the successful confirmation.   
5. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The customer logs in and accesses the order confirmation page through the browser.   
2. The system displays the order summary, including items, total price, and selected payment method.   
3. The customer reviews the order details.   
4. The customer clicks the "Confirm Order" button.   
5. The system validates the order and payment information.   
6. The system updates the Order data entity to mark it as confirmed.   
7. The system clears the Shopping Cart of the items included in the order.   
8. The system sends a confirmation message to the customer via email or in-app notification.   
  
Alternative Flow:   
1. If the order is already confirmed, the system displays a message indicating that the order cannot be confirmed again.   
2. If the payment information is invalid or incomplete, the system displays an error and prompts the customer to correct it.   
3. If the system is under maintenance, the administrator blocks the confirmation process, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Cancel Order   
Use Case ID: UC-17   
Actors: Customer, Order, Browser, Plugin, Administrator   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has placed an order (UC-15).   
4. The order is in a cancellable state (e.g., pending, not yet shipped).   
  
Postconditions:   
1. The selected order is successfully canceled in the system.   
2. The Order data entity is updated to reflect the canceled status.   
3. The Shopping Cart data entity is updated if necessary (e.g., items are returned to cart).   
4. The customer is notified of the successful cancellation.   
5. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The customer logs in and navigates to the order management section through the browser.   
2. The system displays a list of the customer's orders.   
3. The customer selects an order to cancel.   
4. The system displays the order details and confirms whether it is in a cancellable state.   
5. The customer clicks the "Cancel Order" button.   
6. The system validates the order status and confirms the cancellation.   
7. The system updates the Order data entity to mark it as canceled.   
8. If applicable, the system returns the items to the Shopping Cart data entity.   
9. The system sends a confirmation message to the customer.   
  
Alternative Flow:   
1. If the order is not in a cancellable state, the system displays an error message and does not allow the cancellation.   
2. If the system cannot validate the order, it displays an error and logs the issue for administrator review.   
3. If the system is under maintenance, the administrator blocks the cancellation process, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: View Order Status   
Use Case ID: UC-18   
Actors: Customer, Order, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has placed at least one order.   
4. The customer has navigated to the order status section.   
  
Postconditions:   
1. The current status of the selected order is displayed to the customer.   
2. The system logs the activity for audit purposes.   
3. The customer is informed of any relevant updates or changes to the order status.   
  
Main Flow:   
1. The customer logs in and accesses the order status section through the browser.   
2. The system displays a list of the customer's previous orders.   
3. The customer selects a specific order to view its status.   
4. The system retrieves the status information from the Order data entity.   
5. The system displays the order status (e.g., processing, shipped, delivered, canceled).   
6. The system may also show additional details, such as tracking information or expected delivery date.   
  
Alternative Flow:   
1. If the order does not exist or cannot be retrieved, the system displays an error message and logs the issue for administrator review.   
2. If the order status has not been updated, the system informs the customer that the latest status is not available.   
3. If the system is under maintenance, the administrator blocks access to the order status section, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Send Order Confirmation Email   
Use Case ID: UC-19   
Actors: Customer, Order, Browser, Plugin, Administrator   
  
Preconditions:   
1. The customer has placed an order (UC-15).   
2. The order is confirmed (UC-16).   
3. The system is accessible via the browser.   
4. The customer has provided a valid email address during registration or checkout.   
5. The system is not under maintenance.   
  
Postconditions:   
1. An order confirmation email is sent to the customer.   
2. The email contains the order details, including product list, total amount, and payment method.   
3. The system logs the email sending activity for audit purposes.   
4. The customer receives the email and is aware of the order confirmation.   
  
Main Flow:   
1. After the order is confirmed (UC-16), the system automatically triggers the "Send Order Confirmation Email" process.   
2. The system retrieves the order details from the Order data entity.   
3. The system compiles the email content, including the customer's name, order summary, and shipping/payment details.   
4. The system sends the confirmation email to the customer's registered email address.   
5. The system logs the successful email sending activity.   
  
Alternative Flow:   
1. If the customer's email address is invalid or missing, the system displays an error and logs the issue for administrator review.   
2. If the system is unable to send the email due to technical issues, it logs the error and retries sending the email after a predefined interval.   
3. If the system is under maintenance, the administrator may disable the email sending feature, and the system logs the skipped action for later processing.  
  
Use Case Name: Add to Shopping Cart   
Use Case ID: UC-14   
Actors: Customer, Product, Shopping Cart, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has navigated to the product page or search results.   
4. The selected product is available in the system.   
  
Postconditions:   
1. The selected product is added to the customer's Shopping Cart data entity.   
2. The system updates the Shopping Cart with the new item and quantity.   
3. The customer is notified that the product has been successfully added.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The customer browses the product page or search results through the browser.   
2. The system displays product details, including price, availability, and description.   
3. The customer selects the desired product and specifies the quantity.   
4. The customer clicks the "Add to Cart" button via the browser.   
5. The system validates the product availability and quantity.   
6. The system updates the Shopping Cart data entity by adding the product with the specified quantity.   
7. The system provides a confirmation message that the product has been added to the cart.   
  
Alternative Flow:   
1. If the selected product is out of stock, the system displays a warning message and does not add it to the cart.   
2. If the quantity exceeds the available stock, the system prompts the customer to reduce the quantity.   
3. If the system is under maintenance, the administrator blocks the cart addition process, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Modify Shopping Cart Items   
Use Case ID: UC-14   
Actors: Customer, Shopping Cart, Product, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has at least one item in their shopping cart.   
4. The customer has navigated to the shopping cart page.   
  
Postconditions:   
1. The shopping cart items are successfully modified (e.g., quantity updated or item removed).   
2. The Shopping Cart data entity is updated to reflect the changes.   
3. The system logs the activity for audit purposes.   
4. The customer is notified of the successful modification.   
  
Main Flow:   
1. The customer logs in and navigates to the shopping cart page through the browser.   
2. The system displays the list of items currently in the shopping cart.   
3. The customer selects an item to modify (e.g., change quantity or remove it).   
4. The customer updates the quantity or clicks the "Remove" button for the selected item.   
5. The system validates the requested change (e.g., ensures the quantity is within available stock).   
6. The system updates the Shopping Cart data entity with the new quantity or removes the item.   
7. The system recalculates the total price and displays the updated cart summary.   
8. The system provides a confirmation message that the cart has been successfully modified.   
  
Alternative Flow:   
1. If the requested quantity exceeds the available stock, the system displays an error message and prompts the customer to adjust the quantity.   
2. If the customer attempts to modify an item that is no longer available, the system removes the item from the cart and displays a notification.   
3. If the system is under maintenance, the administrator blocks cart modifications, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Remove from Shopping Cart   
Use Case ID: UC-13   
Actors: Customer, Shopping Cart, Product, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has at least one item in their shopping cart.   
4. The customer has navigated to the shopping cart page.   
  
Postconditions:   
1. The selected item is successfully removed from the shopping cart.   
2. The Shopping Cart data entity is updated to reflect the removal.   
3. The system logs the activity for audit purposes.   
4. The customer is notified of the successful removal.   
  
Main Flow:   
1. The customer logs in and navigates to the shopping cart page through the browser.   
2. The system displays the list of items currently in the shopping cart.   
3. The customer selects an item to remove.   
4. The customer clicks the "Remove" button for the selected item.   
5. The system validates the item exists in the shopping cart.   
6. The system updates the Shopping Cart data entity by removing the selected item.   
7. The system recalculates the total price and displays the updated cart summary.   
8. The system provides a confirmation message that the item has been successfully removed from the cart.   
  
Alternative Flow:   
1. If the selected item is no longer available in the system, the system removes it from the cart and displays a notification.   
2. If the system is under maintenance, the administrator blocks cart modifications, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Proceed to Checkout   
Use Case ID: UC-17   
Actors: Customer, Shopping Cart, Order, Payment Information, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has at least one item in their shopping cart.   
4. The customer has navigated to the checkout section.   
  
Postconditions:   
1. The customer is directed to the payment information section.   
2. The system updates the Order data entity with the customer's cart items.   
3. The system logs the activity for audit purposes.   
4. The shopping cart is prepared for order completion.   
  
Main Flow:   
1. The customer logs in and navigates to the shopping cart through the browser.   
2. The system displays the cart with the selected items.   
3. The customer clicks the "Proceed to Checkout" button.   
4. The system verifies the cart items and their availability.   
5. The system redirects the customer to the payment information section.   
  
Alternative Flow:   
1. If the cart is empty, the system displays an error and prompts the customer to add items.   
2. If an item is out of stock, the system removes it from the cart and displays a notification.   
3. If the system is under maintenance, the administrator blocks the checkout process, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Process Payment   
Use Case ID: UC-18   
Actors: Customer, Order, Payment Information, Browser, Plugin   
  
Preconditions:   
1. The customer has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The customer has navigated to the payment processing section.   
4. The customer has placed an order (UC-15) and is ready to complete the transaction.   
  
Postconditions:   
1. The payment is successfully processed, and the order is marked as paid.   
2. The system updates the Order and Payment Information data entities with the transaction details.   
3. The customer is notified of the successful payment.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The customer logs in and navigates to the payment processing page through the browser.   
2. The system displays the selected order and payment method for review.   
3. The customer confirms the payment details and clicks the "Process Payment" button.   
4. The system verifies the payment method and initiates the transaction.   
5. The system processes the payment through the payment gateway.   
6. If the payment is successful, the system updates the Order data entity with the payment confirmation.   
7. The system updates the Payment Information data entity with the transaction details.   
8. The system sends a confirmation message to the customer.   
9. The customer is redirected to an order confirmation page.   
  
Alternative Flow:   
1. If the payment fails due to insufficient funds or invalid information, the system displays an error and prompts the customer to try a different payment method.   
2. If the customer cancels the payment process, the system retains the order in a pending state for future processing.   
3. If the system is under maintenance, the administrator blocks payment processing, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Manage Inventory   
Use Case ID: UC-19   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the inventory management section.   
4. The system contains at least one product and one category.   
  
Postconditions:   
1. Inventory data is successfully updated, added, or removed in the system.   
2. The system updates the Product and Category data entities accordingly.   
3. The administrator is notified of the successful management action.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and accesses the inventory management section through the browser.   
2. The system displays the inventory dashboard with product listings and stock levels.   
3. The administrator selects an option to add, update, or delete inventory items.   
4. The system loads the appropriate form or confirmation prompt based on the selected option.   
5. If adding, the administrator enters product details and stock quantity.   
6. If updating, the administrator modifies the stock quantity or other relevant details.   
7. If deleting, the administrator confirms the removal of the product from inventory.   
8. The system validates the input data and confirms the product exists in the system.   
9. The system updates the Product and Category data entities with the new inventory information.   
10. The system provides a confirmation message that the inventory has been successfully managed.   
  
Alternative Flow:   
1. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the administrator attempts to delete a product that is associated with an active order, the system displays a warning and asks for confirmation.   
3. If the system is under maintenance, the administrator is informed that inventory management is temporarily unavailable.  
  
Use Case Name: Install Plugin   
Use Case ID: UC-14   
Actors: Administrator, Plugin, Browser   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the plugin management section.   
4. The plugin to be installed is available in the system.   
  
Postconditions:   
1. The selected plugin is successfully installed in the system.   
2. The system updates the Plugin data entity to reflect the installation.   
3. The administrator is notified of the successful installation.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the plugin management section through the browser.   
2. The system displays a list of available plugins.   
3. The administrator selects a plugin to install.   
4. The system confirms the selection and prompts for installation.   
5. The administrator confirms the installation.   
6. The system installs the plugin and updates the Plugin data entity.   
7. The system provides a confirmation message that the plugin has been successfully installed.   
  
Alternative Flow:   
1. If the plugin is already installed, the system displays a message indicating that the plugin cannot be installed again.   
2. If the system is under maintenance, the administrator is informed that plugin installation is temporarily unavailable.  
  
Use Case Name: Configure Plugin Settings   
Use Case ID: UC-15   
Actors: Administrator, Plugin, Browser   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the plugin settings section.   
4. The selected plugin is already installed in the system.   
  
Postconditions:   
1. The plugin settings are successfully configured in the system.   
2. The system updates the Plugin data entity with the new settings.   
3. The administrator is notified of the successful configuration.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the plugin settings section through the browser.   
2. The system displays a list of installed plugins with options to configure each.   
3. The administrator selects a plugin to configure.   
4. The system loads the plugin configuration form with current settings.   
5. The administrator modifies the plugin settings as needed (e.g., enable/disable features, set preferences).   
6. The administrator submits the configuration form.   
7. The system validates the configuration data.   
8. The system updates the Plugin data entity with the new settings.   
9. The system provides a confirmation message that the plugin settings have been successfully configured.   
  
Alternative Flow:   
1. If the configuration data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the plugin is not installed, the system displays an error and prompts the administrator to install it first.   
3. If the system is under maintenance, the administrator is informed that plugin configuration is temporarily unavailable.  
  
Use Case Name: Uninstall Plugin   
Use Case ID: UC-16   
Actors: Administrator, Plugin, Browser   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the plugin management section.   
4. The plugin to be uninstalled is currently installed in the system.   
  
Postconditions:   
1. The selected plugin is successfully uninstalled from the system.   
2. The system updates the Plugin data entity to reflect the uninstallation.   
3. The administrator is notified of the successful uninstallation.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the plugin management section through the browser.   
2. The system displays a list of installed plugins with their details.   
3. The administrator selects a plugin to uninstall.   
4. The system prompts for confirmation to proceed with the uninstallation.   
5. The administrator confirms the uninstallation.   
6. The system checks for any dependencies or active configurations related to the plugin.   
7. The system uninstalls the plugin and updates the Plugin data entity.   
8. The system provides a confirmation message that the plugin has been successfully uninstalled.   
  
Alternative Flow:   
1. If the plugin is in use or has active dependencies, the system displays a warning and asks the administrator to confirm the uninstallation despite potential issues.   
2. If the administrator cancels the uninstallation confirmation, the system retains the plugin and does not perform any uninstallation action.   
3. If the system is under maintenance, the administrator is informed that plugin uninstallation is temporarily unavailable.  
  
Use Case Name: Browse Storefront   
Use Case ID: UC-01   
Actors: Customer, Browser, Plugin, Administrator   
Preconditions:   
1. The customer has access to the system via the browser.   
2. The system is operational and accessible.   
3. The storefront is populated with products and categories.   
  
Postconditions:   
1. The customer is able to view products and categories on the storefront.   
2. The system logs the customer's browsing activity for analytics purposes.   
3. The customer may add products to their shopping cart or continue browsing.   
  
Main Flow:   
1. The customer opens the browser and navigates to the storefront using a plugin.   
2. The system loads the storefront homepage, displaying featured products and categories.   
3. The customer selects a category to filter products or searches for a specific product.   
4. The system displays the filtered or searched product listings.   
5. The customer views product details, including images, descriptions, and pricing.   
6. The customer may add a product to their shopping cart.   
7. The system updates the shopping cart with the added product.   
8. The customer continues browsing or proceeds to checkout.   
  
Alternative Flow:   
1. If no products match the search or category, the system displays a message indicating that no results were found.   
2. If the system is under maintenance, the administrator blocks storefront access, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Search Products   
Use Case ID: UC-04   
Actors: Customer, Product, Category, Browser, Plugin   
Preconditions:   
1. The customer has access to the system via the browser.   
2. The system is operational and accessible.   
3. The storefront contains products and categories.   
4. The customer is either logged in or not logged in.   
  
Postconditions:   
1. The customer receives a list of products matching their search criteria.   
2. The system logs the search activity for analytics purposes.   
3. The customer can view product details or add products to their shopping cart.   
  
Main Flow:   
1. The customer navigates to the storefront via the browser.   
2. The customer uses the search bar to enter a product name, keyword, or category.   
3. The system processes the search query and matches it against the Product and Category data entities.   
4. The system displays a list of relevant products with their details (e.g., name, price, image).   
5. The customer selects a product to view more details or adds it to the shopping cart.   
6. The system updates the shopping cart if a product is added.   
7. The system provides a summary of the search results for the customer.   
  
Alternative Flow:   
1. If no products match the search query, the system displays a message indicating that no results were found.   
2. If the search query is invalid or incomplete, the system displays an error and prompts the customer to refine their search.   
3. If the system is under maintenance, the administrator blocks the search functionality, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Filter Products by Category   
Use Case ID: UC-02   
Actors: Customer, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The customer has access to the system via the browser.   
2. The system is operational and accessible.   
3. The storefront is populated with products and categories.   
4. The customer is either logged in or not logged in.   
  
Postconditions:   
1. The customer receives a list of products filtered by the selected category.   
2. The system logs the filtering activity for analytics purposes.   
3. The customer can view the filtered product details or add products to their shopping cart.   
  
Main Flow:   
1. The customer navigates to the storefront using the browser.   
2. The system displays a list of available categories.   
3. The customer selects a category to filter products.   
4. The system processes the category selection and retrieves all products in that category.   
5. The system displays the filtered product list with their details (e.g., name, price, image).   
6. The customer may add a product to their shopping cart.   
7. The system updates the shopping cart with the added product.   
8. The customer continues browsing or proceeds to checkout.   
  
Alternative Flow:   
1. If no products are found in the selected category, the system displays a message indicating that there are no products available.   
2. If the selected category does not exist or is invalid, the system displays an error message and prompts the customer to select a valid category.   
3. If the system is under maintenance, the administrator blocks the filtering functionality, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: View Product Details   
Use Case ID: UC-03   
Actors: Customer, Product, Browser, Plugin   
Preconditions:   
1. The customer has access to the system via the browser.   
2. The system is operational and accessible.   
3. A product exists in the system that the customer wants to view.   
  
Postconditions:   
1. The product details are displayed to the customer.   
2. The system logs the customer's view activity for analytics purposes.   
3. The customer may add the product to their shopping cart or continue browsing.   
  
Main Flow:   
1. The customer navigates to the product listing page or search results via the browser.   
2. The system displays a list of products with basic information (e.g., name, price, image).   
3. The customer selects a specific product to view its details.   
4. The system retrieves the product's full information from the Product data entity.   
5. The system displays the product details, including description, price, availability, and images.   
6. The customer reviews the product information.   
7. The customer may choose to add the product to their shopping cart or return to browsing.   
  
Alternative Flow:   
1. If the product is not found or no longer exists, the system displays an error message and suggests related products or categories.   
2. If the system is under maintenance, the administrator blocks the product details view, and the customer receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Admin Login   
Use Case ID: UC-07   
Actors: Administrator, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is not currently logged in.   
2. The system is accessible via the browser.   
3. The login page is available and not blocked due to maintenance.   
  
Postconditions:   
1. The administrator is successfully logged into the system.   
2. The system updates the session status for the administrator.   
3. The administrator is redirected to the admin dashboard.   
4. The system logs the login activity for audit purposes.   
  
Main Flow:   
1. The administrator opens the browser and navigates to the admin login page via a plugin.   
2. The system displays the admin login form.   
3. The administrator enters their username and password.   
4. The administrator submits the login form.   
5. The system validates the credentials against the Administrator data entity.   
6. If the credentials are valid, the system creates an admin session and logs the login.   
7. The system redirects the administrator to the admin dashboard.   
  
Alternative Flow:   
1. If the entered credentials are invalid, the system displays an error message and prompts the administrator to re-enter their username and password.   
2. If the administrator account is locked or disabled, the system displays a message indicating that the account is not active.   
3. If the system is under maintenance, the administrator is informed that login is temporarily unavailable.  
  
Use Case Name: Manage User Accounts   
Use Case ID: UC-19   
Actors: Administrator, Customer, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the user account management section.   
  
Postconditions:   
1. User accounts are successfully created, updated, or deleted in the system.   
2. The system updates the Customer and Administrator data entities accordingly.   
3. The system logs the activity for audit purposes.   
4. The user is notified of any changes to their account.   
  
Main Flow:   
1. The administrator logs in and navigates to the user account management section through the browser.   
2. The system displays a list of user accounts with options to create, edit, or delete.   
3. The administrator selects an action (create, update, delete) for a specific user.   
4. The system loads the appropriate form or confirmation prompt based on the selected action.   
5. If creating, the administrator enters the new user's details (e.g., name, email, password).   
6. If updating, the administrator modifies the selected user's information.   
7. If deleting, the administrator confirms the deletion of the selected user.   
8. The system validates the input data and checks for any constraints (e.g., duplicate email for creation).   
9. The system updates the Customer or Administrator data entity based on the selected action.   
10. The system provides a confirmation message that the account has been successfully managed.   
  
Alternative Flow:   
1. If the email already exists when creating a new user, the system displays an error message and prompts the administrator to choose a different email.   
2. If the input data is invalid (e.g., incorrect password format), the system displays an error message and prompts the administrator to correct the information.   
3. If the administrator attempts to delete a user with active orders, the system displays a warning and asks for confirmation.   
4. If the system is under maintenance, the administrator is informed that user account management is temporarily unavailable.  
  
Use Case Name: Restrict Access by Browser Type   
Use Case ID: UC-14   
Actors: Administrator, Browser, Plugin, Customer   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the browser access restriction settings.   
4. A browser type is defined in the system.   
  
Postconditions:   
1. Access is restricted based on the browser type configuration.   
2. The system updates the Browser data entity with the restriction settings.   
3. The administrator is notified of the successful restriction configuration.   
4. The system logs the activity for audit purposes.   
5. Unauthorized browser users are blocked or redirected.   
  
Main Flow:   
1. The administrator logs in and navigates to the browser restriction configuration page through the browser.   
2. The system displays a list of supported browser types and current restriction settings.   
3. The administrator selects specific browser types to allow or restrict.   
4. The administrator submits the configuration.   
5. The system validates the selected browser types and updates the Browser data entity.   
6. The system applies the restriction rules and confirms the update to the administrator.   
  
Alternative Flow:   
1. If the system detects an invalid or unsupported browser type, it displays an error and prompts the administrator to choose a valid browser.   
2. If the administrator attempts to restrict a browser type that is currently in use, the system displays a warning and asks for confirmation.   
3. If the system is under maintenance, the administrator is informed that browser restriction configuration is temporarily unavailable.  
  
Use Case Name: View System Documentation   
Use Case ID: UC-14   
Actors: Customer, Administrator, Browser, Plugin   
Preconditions:   
1. The customer or administrator has access to the system via the browser.   
2. The system is operational and accessible.   
3. The customer or administrator has navigated to the documentation section.   
4. The system documentation is available and up to date.   
  
Postconditions:   
1. The customer or administrator is able to view the system documentation.   
2. The system logs the activity for audit purposes.   
3. The user is informed of the documentation content and can navigate through it.   
  
Main Flow:   
1. The customer or administrator logs in and navigates to the system documentation page through the browser.   
2. The system displays a table of contents or search functionality for the documentation.   
3. The user selects a topic or section to view.   
4. The system retrieves and displays the selected documentation content.   
5. The user may navigate to other sections or search for additional information.   
6. The system logs the documentation access activity.   
  
Alternative Flow:   
1. If the documentation is not available or outdated, the system displays a message informing the user and prompts them to check back later.   
2. If the user attempts to access a non-existent documentation section, the system displays an error and provides a list of valid sections.   
3. If the system is under maintenance, the administrator blocks access to the documentation, and the user receives a message indicating that the feature is temporarily unavailable.  
  
Use Case Name: Manage Order Details   
Use Case ID: UC-20   
Actors: Administrator, Order, Product, Customer, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the order details management section.   
4. A specific order exists in the system and is selected for modification.   
5. The system has access to the OrderDetail data entity.   
  
Postconditions:   
1. The order details (e.g., product quantity, price, or notes) are successfully updated or reviewed.   
2. The system updates the OrderDetail data entity and may update the Order and Product data entities if necessary.   
3. The administrator is notified of the successful management of order details.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the order management section through the browser.   
2. The system displays a list of orders with options to view or edit details.   
3. The administrator selects a specific order to manage its details.   
4. The system retrieves the associated OrderDetail data entities and displays them in an editable format.   
5. The administrator modifies specific order details (e.g., updates the quantity of a product, adjusts the price, or adds a note).   
6. The administrator submits the updated details.   
7. The system validates the changes (e.g., ensures the new quantity is within available stock or matches pricing rules).   
8. The system updates the OrderDetail data entity with the modified information.   
9. If necessary, the system updates the Order data entity to reflect any overall changes (e.g., total order amount).   
10. The system provides a confirmation message that the order details have been successfully managed.   
  
Alternative Flow:   
1. If the modified quantity exceeds available stock, the system displays an error message and prompts the administrator to adjust the value.   
2. If the administrator attempts to modify an OrderDetail that is linked to a completed or shipped order, the system displays a warning and may block the change or require additional authorization.   
3. If the system is under maintenance, the administrator is informed that managing order details is temporarily unavailable.  
  
Use Case Name: Manage Inventory   
Use Case ID: UC-21   
Actors: Administrator, Product, Category, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the inventory management section.   
4. The system contains at least one product and one category.   
  
Postconditions:   
1. Inventory data is successfully updated, added, or removed in the system.   
2. The system updates the Product and Category data entities accordingly.   
3. The administrator is notified of the successful management action.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and accesses the inventory management section through the browser.   
2. The system displays an overview of inventory levels, including product names, quantities, and associated categories.   
3. The administrator selects an action to perform: add new inventory, update existing inventory, or remove inventory.   
4. The system presents a relevant form or interface based on the selected action.   
5. If adding new inventory, the administrator enters product details and specifies the quantity to add.   
6. If updating inventory, the administrator modifies the quantity or other relevant inventory attributes (e.g., reorder level, stock status).   
7. If removing inventory, the administrator confirms the reduction in stock or full removal of the product.   
8. The system validates the input data (e.g., non-negative quantity, valid product selection).   
9. The system updates the Product data entity with the new inventory levels and modifies the Category data entity if necessary.   
10. The system provides a confirmation message that the inventory has been successfully managed.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., negative quantity, missing product), the system displays an error message and prompts the administrator to correct the information.   
2. If the administrator attempts to remove inventory for a product that is associated with an active or pending order, the system displays a warning and asks for confirmation before proceeding.   
3. If the system is under maintenance, the administrator is informed that inventory management is temporarily unavailable.  
  
Use Case Name: Manage Email Service   
Use Case ID: UC-22   
Actors: Administrator, Email Service, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the email service management section.   
4. The system has access to the Email Service configuration.   
  
Postconditions:   
1. The email service is configured, updated, or disabled in the system.   
2. The system updates the Email Service data entity with the new settings.   
3. The administrator is notified of the successful management action.   
4. The system logs the activity for audit purposes.   
5. Email-related features (e.g., notifications, confirmations) are affected according to the configuration.   
  
Main Flow:   
1. The administrator logs in and navigates to the email service configuration section through the browser.   
2. The system displays the current email service settings, including server details, port, protocol, authentication, and email templates.   
3. The administrator selects an option to configure, update, or disable the email service.   
4. The system loads the appropriate form or interface based on the selected action.   
5. The administrator modifies the email service settings (e.g., updates SMTP credentials, changes email templates, or disables the service).   
6. The administrator submits the updated configuration.   
7. The system validates the input data (e.g., correct server format, valid port number).   
8. The system updates the Email Service data entity with the new settings.   
9. The system confirms the successful configuration of the email service.   
10. If the service is disabled, the system logs the deactivation and updates email-related functionality accordingly.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., incorrect server address, invalid credentials), the system displays an error message and prompts the administrator to correct the information.   
2. If the administrator attempts to disable the email service while there are pending email notifications, the system displays a warning and suggests rescheduling or completing the notifications before disabling.   
3. If the system is under maintenance, the administrator is informed that managing the email service is temporarily unavailable.  
  
Use Case Name: Manage Registration Record   
Use Case ID: UC-23   
Actors: Administrator, Customer, Registration Record, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the registration record management section.   
4. The system contains at least one customer registration record.   
  
Postconditions:   
1. Registration records are successfully viewed, updated, or deleted in the system.   
2. The system updates the Registration Record data entity accordingly.   
3. The administrator is notified of the successful management action.   
4. The system logs the activity for audit purposes.   
  
Main Flow:   
1. The administrator logs in and navigates to the registration record management section through the browser.   
2. The system displays a list of customer registration records, including details such as registration date, email, status (e.g., confirmed/unconfirmed), and other relevant information.   
3. The administrator selects a specific registration record to view, update, or delete.   
4. If viewing, the system displays the full details of the selected registration record.   
5. If updating, the system loads the editable form for the selected registration record, allowing the administrator to modify fields such as confirmation status or registration date.   
6. If deleting, the system prompts the administrator to confirm the deletion of the selected registration record.   
7. The administrator submits the changes or confirms the deletion.   
8. The system validates the requested action (e.g., ensures the record exists, validates status changes).   
9. The system updates the Registration Record data entity with the changes or removes the record if deletion is confirmed.   
10. The system provides a confirmation message that the registration record has been successfully managed.   
  
Alternative Flow:   
1. If the selected registration record does not exist, the system displays an error message and prompts the administrator to select a valid record.   
2. If the administrator attempts to delete a registration record that is linked to an active customer account, the system displays a warning and asks for confirmation before proceeding.   
3. If the system is under maintenance, the administrator is informed that managing registration records is temporarily unavailable.  
  
Use Case Name: Manage Order   
Use Case ID: UC-24   
Actors: Administrator, Order, Product, Customer, Browser, Plugin   
  
Preconditions:   
1. The administrator has an existing account and is logged in.   
2. The system is accessible via the browser.   
3. The administrator has navigated to the order management section.   
4. At least one order exists in the system.   
5. The system has access to the Order data entity.   
  
Postconditions:   
1. The order is successfully managed (e.g., updated, canceled, or shipped).   
2. The system updates the Order data entity with the new status or modifications.   
3. The administrator is notified of the successful management of the order.   
4. The system logs the activity for audit purposes.   
5. The customer may receive notifications regarding the order status changes.   
  
Main Flow:   
1. The administrator logs in and navigates to the order management section through the browser.   
2. The system displays a list of orders with their current status (e.g., pending, confirmed, shipped, delivered, canceled).   
3. The administrator selects a specific order to manage.   
4. The system loads the order details, including customer information, product list, and payment status.   
5. The administrator chooses an action to perform (e.g., update the order status, cancel the order, or modify the order details).   
6. The system validates the selected action against the current order status.   
7. If valid, the system updates the Order data entity to reflect the new status or changes.   
8. The system logs the order management activity.   
9. The system sends a notification to the customer if the order status has changed.   
10. The system provides a confirmation message that the order has been successfully managed.   
  
Alternative Flow:   
1. If the selected action is not valid for the current order status (e.g., attempting to cancel a delivered order), the system displays an error message and does not apply the action.   
2. If the system cannot locate the selected order, it displays an error message and logs the issue for administrator review.   
3. If the system is under maintenance, the administrator is informed that order management is temporarily unavailable.