项目文档

# Functional Requirement

1.1 Customer Registration Function   
Function ID: FR-01   
Description: Customers can register by providing personal information, including name, email, password, and contact details. The system validates the email, checks for duplicates, and sends a confirmation email.   
Input: Customer name, email, password, and contact information   
Output: A new Customer record in the database with hashed password and active status after confirmation   
  
1.2 Customer Login Function   
Function ID: FR-02   
Description: Customers can log in to the system by entering their registered email and password. The system verifies the credentials and generates a session token for authentication.   
Input: Registered email and password   
Output: Valid session token and redirection to the customer dashboard   
  
1.3 Customer Logout Function   
Function ID: FR-03   
Description: Customers can log out of the system by selecting the "Logout" option. The system invalidates the session token and clears session data.   
Input: Valid session token   
Output: Invalidated session token and redirection to the login or home page   
  
1.4 View Product Details Function   
Function ID: FR-04   
Description: Customers can view detailed information for a selected product, including name, description, price, category, and inventory status.   
Input: Product ID or product name   
Output: Display of product details, including name, description, price, category, and current stock level   
  
1.5 Search Products by Category Function   
Function ID: FR-05   
Description: Customers can search and view a list of products within a selected category. The system retrieves active products from the database and displays them with relevant details.   
Input: Category ID or category name   
Output: List of active products in the selected category, including name, description, price, and inventory status   
  
1.6 Add Product to Cart Function   
Function ID: FR-06   
Description: Customers can add a product to their shopping cart after selecting a valid quantity. The system checks inventory availability and updates the cart accordingly.   
Input: Product ID, quantity, and valid session token   
Output: Updated cart with the added product and confirmation message   
  
1.7 Remove Product from Cart Function   
Function ID: FR-07   
Description: Customers can remove a product from their shopping cart. The system verifies the product's presence in the cart, removes it, and updates the inventory if applicable.   
Input: Product ID, valid session token, and confirmation of removal   
Output: Updated cart with the removed product and confirmation message   
  
1.8 View Cart Function   
Function ID: FR-08   
Description: Customers can view the contents of their shopping cart, including product names, prices, quantities, and total cost. The system retrieves cart data and displays it.   
Input: Valid session token   
Output: Display of cart items, total quantity, total price, and inventory status for each item   
  
1.9 Checkout and Place Order Function   
Function ID: FR-09   
Description: Customers can complete the checkout process by entering shipping and billing information and selecting a payment method. The system creates an order, processes the payment, and sends an order confirmation.   
Input: Cart items, shipping address, billing address, and selected payment method   
Output: New Order record, updated Inventory, processed Payment, and confirmation Email   
  
1.10 Receive Order Confirmation Email Function   
Function ID: FR-10   
Description: After a successful checkout, the system sends a confirmation email to the customer's registered email address, including order details and expected delivery.   
Input: Order details and customer email   
Output: Sent confirmation Email with order information   
  
1.11 Update Customer Information Function   
Function ID: FR-11   
Description: Customers can update their personal information, such as name, email, password, or contact details. The system validates the new data and updates the customer record.   
Input: Customer ID, updated personal information (e.g., new email or password), and valid session token   
Output: Updated Customer record in the database and, if applicable, a confirmation Email for email changes   
  
1.12 Delete Customer Account Function   
Function ID: FR-12   
Description: Customers can delete their account after confirming their identity. The system removes or anonymizes the customer's data and sends a confirmation email.   
Input: Customer ID, valid session token, and confirmation of deletion   
Output: Deleted or anonymized Customer record, confirmation Email, and updated Customer database   
  
1.13 Administrator Login Function   
Function ID: FR-13   
Description: Administrators can log in to the system by entering their email and password. The system verifies the credentials and generates a session token for access.   
Input: Administrator email and password   
Output: Valid session token and redirection to the administrative dashboard   
  
1.14 Administrator Logout Function   
Function ID: FR-14   
Description: Administrators can log out of the system by selecting the "Logout" option. The system invalidates the session token and clears session data.   
Input: Valid session token   
Output: Invalidated session token and redirection to the login or home page   
  
1.15 Add New Product Function   
Function ID: FR-15   
Description: Administrators can add new products to the system by entering product details, assigning a category, and setting inventory levels.   
Input: Product name, description, price, category, and initial stock quantity   
Output: New Product record in the database, updated Inventory, and association with the correct Category   
  
1.16 Update Product Details Function   
Function ID: FR-16   
Description: Administrators can modify existing product details, including name, description, price, category, and stock quantity.   
Input: Product ID, updated product information, and valid session token   
Output: Updated Product record, updated Inventory, and, if changed, updated Category association   
  
1.17 Delete Product Function   
Function ID: FR-17   
Description: Administrators can delete a product from the system after confirming its existence and ensuring no active orders reference it.   
Input: Product ID, valid session token, and confirmation of deletion   
Output: Deleted or marked Product record, updated Inventory, and updated Category association   
  
1.18 Manage Inventory Levels Function   
Function ID: FR-18   
Description: Administrators can update the inventory levels for a specific product, ensuring stock availability is maintained.   
Input: Product ID, new stock quantity, and valid session token   
Output: Updated Inventory record and adjusted product availability status   
  
1.19 View Inventory Report Function   
Function ID: FR-19   
Description: Administrators can generate and view an inventory report, which includes product names, stock levels, and category information.   
Input: Valid session token and optional filters (category, product, date range)   
Output: Inventory report with product details, stock levels, and category information   
  
1.20 Create and Update Product Categories Function   
Function ID: FR-20   
Description: Administrators can create new product categories or update existing ones, ensuring products are properly categorized.   
Input: Category name, description, and optional parent category (for hierarchical structure)   
Output: New or updated Category record in the database and association with relevant Products   
  
1.21 Delete Product Category Function   
Function ID: FR-21   
Description: Administrators can delete an existing product category after confirming it is not in use by any active products.   
Input: Category ID, valid session token, and confirmation of deletion   
Output: Deleted or marked Category record and reassigned or uncategorized Products   
  
1.22 View Category List Function   
Function ID: FR-22   
Description: Users (customers or administrators) can view a list of active product categories.   
Input: Valid session token (optional for public access)   
Output: Displayed list of active categories, including names and descriptions   
  
1.23 Process Payment Function   
Function ID: FR-23   
Description: The system processes customer payments during checkout using the selected payment method and updates the order status accordingly.   
Input: Order details, payment method, and transaction request   
Output: Updated Order status as "Paid", recorded Payment details, and updated Inventory   
  
1.24 View Payment History Function   
Function ID: FR-24   
Description: Customers can view their payment history, including transaction dates, amounts, methods, and associated order references.   
Input: Customer ID and valid session token   
Output: Displayed list of Payment records with transaction details and filtering options   
  
1.25 Manage Plugin Installation Function   
Function ID: FR-25   
Description: Administrators can install or uninstall plugins, ensuring compatibility and updating API integrations as needed.   
Input: Plugin name, version, and configuration details   
Output: Updated Plugin status in the database, API integration status, and confirmation message   
  
1.26 Develop New Plugin Function   
Function ID: FR-26   
Description: Administrators can develop new plugins for the system, ensuring they comply with API standards and are registered for use.   
Input: Plugin name, version, code, and documentation   
Output: Registered Plugin in the database, updated API integration, and confirmation message   
  
1.27 Update Plugin Function   
Function ID: FR-27   
Description: Administrators can update existing plugins, including their configuration or version, and reconfigure API connections if necessary.   
Input: Plugin ID, new configuration, or updated version   
Output: Updated Plugin in the database, updated API integration, and confirmation message   
  
1.28 Delete Plugin Function   
Function ID: FR-28   
Description: Administrators can delete a plugin after ensuring it is not being used by any active system functionality.   
Input: Plugin ID, valid session token, and confirmation of deletion   
Output: Deleted Plugin record, updated API connection status, and confirmation message   
  
1.29 View API Documentation Function   
Function ID: FR-29   
Description: Administrators can view the API documentation for registered plugins or system APIs.   
Input: API ID and valid session token   
Output: Displayed PluginDocumentation or API documentation with detailed specifications   
  
1.30 Access System Documentation Function   
Function ID: FR-30   
Description: Administrators can access system documentation, including plugin APIs and user manuals, for reference and support.   
Input: Valid session token   
Output: Displayed documentation files in a structured and searchable format

# External Description

\*\*Chapter 2: External Interfaces\*\*   
  
This chapter outlines the external interfaces of the system, including user interfaces, hardware interfaces, software interfaces, and communication interfaces. These interfaces describe how the system interacts with users, hardware components, other software systems, and communication protocols. The interfaces are derived from the functional requirements and include all referenced external data sources such as databases, APIs, and email notifications.   
  
---  
  
### 2.1 User Interface Output   
  
The system provides several user interfaces for interaction with customers and administrators. These interfaces are designed to be intuitive and accessible through a web browser.   
  
- \*\*Customer Registration Interface\*\*:   
 - \*Description\*: A form-based interface where customers provide personal information such as name, email, password, and contact details.   
 - \*Interaction Method\*: Web-based form submission.   
 - \*Inputs\*: Name, email, password, and contact information.   
 - \*Outputs\*: Confirmation message and redirection to the login page.   
  
- \*\*Customer Login Interface\*\*:   
 - \*Description\*: A web-based form for customers to enter their email and password to authenticate and gain access to the system.   
 - \*Interaction Method\*: Web-based form submission with HTTP POST.   
 - \*Inputs\*: Registered email and password.   
 - \*Outputs\*: Session token generation and redirection to the customer dashboard.   
  
- \*\*Customer Dashboard Interface\*\*:   
 - \*Description\*: A personalized interface for customers to manage their account, view cart, and access order history.   
 - \*Interaction Method\*: Web-based dynamic loading using session token authentication.   
 - \*Inputs\*: Session token.   
 - \*Outputs\*: Display of cart items, order history, and personal information.   
  
- \*\*Product Detail Interface\*\*:   
 - \*Description\*: A web-based interface to display detailed product information such as name, description, price, category, and inventory status.   
 - \*Interaction Method\*: Dynamic loading based on product ID or name.   
 - \*Inputs\*: Product ID or name.   
 - \*Outputs\*: Display of product details and current stock level.   
  
- \*\*Product Search Interface\*\*:   
 - \*Description\*: An interface where customers can search for products by category.   
 - \*Interaction Method\*: Web-based form or dropdown selection.   
 - \*Inputs\*: Category ID or name.   
 - \*Outputs\*: Display of active products in the selected category with details.   
  
- \*\*Shopping Cart Interface\*\*:   
 - \*Description\*: A dynamic interface where customers can view and manage their cart items.   
 - \*Interaction Method\*: Web-based interface with real-time updates.   
 - \*Inputs\*: Product ID, quantity, and session token.   
 - \*Outputs\*: Display of cart items, total price, and inventory status for each item.   
  
- \*\*Checkout Interface\*\*:   
 - \*Description\*: A multi-step form for customers to enter shipping and billing information and select a payment method.   
 - \*Interaction Method\*: Web-based form submission with session token authentication.   
 - \*Inputs\*: Cart items, shipping address, billing address, and selected payment method.   
 - \*Outputs\*: Order confirmation and redirection to the order history.   
  
- \*\*Administrator Login Interface\*\*:   
 - \*Description\*: A secure web-based form for administrators to authenticate using email and password.   
 - \*Interaction Method\*: HTTP POST with validation.   
 - \*Inputs\*: Administrator email and password.   
 - \*Outputs\*: Session token generation and redirection to the administrative dashboard.   
  
- \*\*Administrative Dashboard Interface\*\*:   
 - \*Description\*: A comprehensive interface for administrators to manage products, inventory, categories, and plugins.   
 - \*Interaction Method\*: Web-based dynamic content loading.   
 - \*Inputs\*: Session token and optional filters (e.g., category, product, date range).   
 - \*Outputs\*: Inventory reports, product lists, plugin status, and category information.   
  
- \*\*Payment History Interface\*\*:   
 - \*Description\*: An interface for customers to view their payment history, including transaction dates, amounts, and associated orders.   
 - \*Interaction Method\*: Web-based interface with filtering options.   
 - \*Inputs\*: Customer ID and session token.   
 - \*Outputs\*: Display of payment records and transaction details.   
  
- \*\*System Documentation Interface\*\*:   
 - \*Description\*: A web-based interface for administrators to access system documentation, user manuals, and plugin APIs.   
 - \*Interaction Method\*: Web-based search and display.   
 - \*Inputs\*: Session token.   
 - \*Outputs\*: Display of structured and searchable documentation.   
  
---  
  
### 2.2 Hardware Interface Output   
  
The system does not directly interface with any hardware devices. All user interactions are performed through standard web browsers and servers. Therefore, there are no hardware interfaces defined for this system.   
  
---  
  
### 2.3 Software Interface Output   
  
The system interacts with several software components, including databases, APIs, and external tools.   
  
- \*\*Customer Database\*\*:   
 - \*Description\*: A database that stores customer information such as name, email, hashed password, and contact details.   
 - \*Interaction Method\*: SQL queries or ORM operations.   
 - \*Inputs\*: Personal information (name, email, password, etc.), session token.   
 - \*Outputs\*: Creation, update, or deletion of customer records.   
  
- \*\*Product Database\*\*:   
 - \*Description\*: A database that stores product details including name, description, price, category, and inventory status.   
 - \*Interaction Method\*: SQL queries or ORM operations.   
 - \*Inputs\*: Product details (name, price, category, etc.), session token.   
 - \*Outputs\*: Creation, update, or deletion of product records.   
  
- \*\*Inventory Database\*\*:   
 - \*Description\*: A database that tracks the stock levels of products.   
 - \*Interaction Method\*: SQL queries or ORM operations.   
 - \*Inputs\*: Product ID and stock quantity.   
 - \*Outputs\*: Updated inventory records and product availability status.   
  
- \*\*Order Database\*\*:   
 - \*Description\*: A database to store order details including customer ID, product list, shipping address, billing address, payment method, and status.   
 - \*Interaction Method\*: SQL queries or ORM operations.   
 - \*Inputs\*: Cart items, shipping and billing addresses, payment method.   
 - \*Outputs\*: Creation of order records and status updates.   
  
- \*\*Plugin Management Interface\*\*:   
 - \*Description\*: An interface for administrators to manage plugins, including installation, updates, and deletion.   
 - \*Interaction Method\*: Web-based form and API calls to register or update plugins.   
 - \*Inputs\*: Plugin name, version, configuration, or code.   
 - \*Outputs\*: Updated plugin status, API integration status, and confirmation message.   
  
- \*\*Category Management Interface\*\*:   
 - \*Description\*: An interface for administrators to manage product categories, including creation, updates, and deletion.   
 - \*Interaction Method\*: Web-based form and database operations.   
 - \*Inputs\*: Category name, description, and parent category.   
 - \*Outputs\*: Creation, update, or deletion of category records and product associations.   
  
- \*\*API Documentation Interface\*\*:   
 - \*Description\*: An interface to display API documentation for plugins and system APIs.   
 - \*Interaction Method\*: Web-based interface with search and display functionality.   
 - \*Inputs\*: API ID and session token.   
 - \*Outputs\*: Display of detailed API specifications and usage instructions.   
  
- \*\*Payment Gateway API\*\*:   
 - \*Description\*: An external API used to process customer payments during checkout.   
 - \*Interaction Method\*: HTTP requests (POST/GET) to the payment gateway.   
 - \*Inputs\*: Order details, payment method, and transaction request.   
 - \*Outputs\*: Payment confirmation, updated order status as "Paid", and recorded transaction details.   
  
---  
  
### 2.4 Communication Interface Output   
  
The system uses communication interfaces to send and receive data via network protocols.   
  
- \*\*Email Notification Service\*\*:   
 - \*Description\*: A service used to send confirmation emails to customers after registration, account deletion, or order placement.   
 - \*Interaction Method\*: SMTP or REST API-based email sending.   
 - \*Inputs\*: Customer email and order details.   
 - \*Outputs\*: Sent confirmation email with order information or account status.   
  
- \*\*Web Browsing Interface\*\*:   
 - \*Description\*: The system is accessed through standard web browsers using HTTP/HTTPS protocols.   
 - \*Interaction Method\*: Web-based user interface with session token authentication.   
 - \*Inputs\*: Session token, user input (e.g., form data).   
 - \*Outputs\*: Dynamic content loading and real-time updates.   
  
- \*\*Plugin API Integration\*\*:   
 - \*Description\*: The system supports third-party plugins via API integrations, allowing administrators to extend system functionality.   
 - \*Interaction Method\*: RESTful API calls with JSON data exchange.   
 - \*Inputs\*: Plugin configuration or transaction data.   
 - \*Outputs\*: Updated plugin status, processed transactions, or system updates.   
  
- \*\*System API Communication\*\*:   
 - \*Description\*: The system exposes internal APIs for administrators to access documentation, manage plugins, and retrieve reports.   
 - \*Interaction Method\*: RESTful API with session token authentication.   
 - \*Inputs\*: Session token, API ID, or plugin ID.   
 - \*Outputs\*: Display of API documentation, plugin status, or inventory reports.   
  
---  
  
\*\*Summary of External Data Sources and Interfaces\*\*   
  
All external data sources referenced in the functional requirements are covered in the above interfaces. The system interacts with:   
  
- \*Databases\*: Customer database, product database, inventory database, order database.   
- \*APIs\*: Payment gateway API, plugin APIs, system APIs.   
- \*Communication Protocols\*: Email notifications via SMTP or REST API, web browsing via HTTP/HTTPS.   
- \*User Interfaces\*: Web-based forms and dynamic dashboards for both customers and administrators.   
  
These interfaces ensure that the system can effectively manage user data, product information, inventory, and external integrations while maintaining secure and efficient communication.

# Use Case

Use Case Name: Customer Registration   
Use Case ID: UC-01   
Actors: Customer, Administrator, System   
Preconditions:   
1. The customer is not yet registered in the system.   
2. The system is operational and accessible to the customer.   
3. The administrator has configured the registration module and ensured that it is active.   
  
Postconditions:   
1. A new customer account is successfully created in the system.   
2. The customer receives a confirmation email.   
3. The system updates the customer database with the new information.   
  
Main Flow:   
1. The customer accesses the registration page of the system.   
2. The customer provides personal information, including name, email, password, and contact details.   
3. The system validates the email format and checks for existing customers with the same email.   
4. The system hashes the password and stores it securely.   
5. The customer submits the registration form.   
6. The system creates a new customer record in the database.   
7. The system sends a confirmation email to the provided email address.   
8. The customer clicks the confirmation link in the email.   
9. The system verifies the confirmation token and marks the account as active.   
10. The customer is redirected to the login page.   
  
Alternative Flow:   
1. If the email is already registered, the system displays an error message and does not proceed with registration.   
2. If the password does not meet the security requirements (e.g., length, complexity), the system prompts the customer to choose a stronger password.   
3. If the confirmation email fails to send, the system logs the error and displays a message to the customer indicating the issue.   
4. If the customer does not click the confirmation link within the expiration time, the registration is considered invalid, and the customer must re-initiate the registration process.  
  
Use Case Name: Customer Login   
Use Case ID: UC-02   
Actors: Customer, System, Email   
Preconditions:   
1. The customer has a valid and activated account in the system.   
2. The system is operational and accessible to the customer.   
3. The login page is available and functional.   
  
Postconditions:   
1. The customer is successfully authenticated and logged into the system.   
2. The system records the login activity.   
3. The customer is redirected to the dashboard or the last visited page.   
  
Main Flow:   
1. The customer navigates to the login page of the system.   
2. The customer enters their registered email and password.   
3. The system validates the input fields for completeness and correct format.   
4. The system verifies the customer's credentials against the stored information in the customer database.   
5. If the credentials are valid, the system logs the customer in and generates a session token.   
6. The system redirects the customer to the appropriate interface based on their account type (e.g., customer dashboard).   
  
Alternative Flow:   
1. If the email is not registered, the system displays an error message indicating that the email is invalid.   
2. If the password is incorrect, the system prompts the customer to re-enter the correct password.   
3. If the system detects multiple failed login attempts, it temporarily locks the account and sends a notification email to the customer.   
4. If the session token cannot be generated, the system logs the error and displays a message to the customer to try again later.  
  
Use Case Name: Customer Logout   
Use Case ID: UC-03   
Actors: Customer, System   
Preconditions:   
1. The customer is logged into the system with a valid session.   
2. The system has an active logout functionality.   
  
Postconditions:   
1. The customer is successfully logged out of the system.   
2. The session token is invalidated.   
3. The customer is redirected to the login or home page.   
  
Main Flow:   
1. The customer navigates to the account settings or user profile page.   
2. The customer selects the "Logout" option.   
3. The system verifies the session and initiates the logout process.   
4. The system invalidates the session token and clears the session data.   
5. The system redirects the customer to the login or home page.   
  
Alternative Flow:   
1. If the session token is already expired or invalid, the system displays a message indicating that the customer is not logged in.   
2. If the logout process fails due to a technical error, the system logs the error and displays a message to the customer to try again later.  
  
Use Case Name: View Product Details   
Use Case ID: UC-04   
Actors: Customer, System, Product, Administrator   
Preconditions:   
1. The customer is logged into the system.   
2. The product exists in the system's database and is active.   
3. The system is operational and the product details page is accessible.   
4. The administrator has ensured the product display functionality is enabled.   
  
Postconditions:   
1. The customer views the detailed information of the selected product.   
2. The system logs the customer's request to view product details.   
3. If the product has an associated plugin or API, the system fetches and displays additional data as needed.   
  
Main Flow:   
1. The customer browses the product catalog or searches for a specific product.   
2. The customer selects a product to view its details.   
3. The system retrieves the product information from the database, including name, description, price, and category.   
4. The system displays the product details on a dedicated page or modal.   
5. If the product has inventory information, the system shows the current stock level.   
6. The customer may choose to add the product to the cart or return to the catalog.   
  
Alternative Flow:   
1. If the product is not found or is inactive, the system displays an error message and suggests similar products.   
2. If the system cannot retrieve inventory information, it displays a message indicating that stock status is temporarily unavailable.   
3. If the product details page encounters an error, the system logs the issue and displays a generic error message to the customer.   
4. If the product requires an API or plugin to display additional data (e.g., reviews, specifications), and the API is unavailable, the system informs the customer that some details cannot be displayed at this time.  
  
Use Case Name: Search Products by Category   
Use Case ID: UC-05   
Actors: Customer, System, Product, Category, Administrator   
Preconditions:   
1. The customer is logged into the system.   
2. The product database contains products organized under one or more categories.   
3. The category functionality is enabled by the administrator.   
4. The system is operational and accessible to the customer.   
  
Postconditions:   
1. The customer views a list of products within the selected category.   
2. The system logs the customer's search activity.   
3. If an API or plugin is used, the system fetches and displays additional product information as required.   
  
Main Flow:   
1. The customer navigates to the product catalog page.   
2. The customer selects a category from the list of available categories.   
3. The system retrieves all active products assigned to the selected category from the database.   
4. The system displays the list of products, including their names, descriptions, prices, and images.   
5. If applicable, the system shows the current inventory status for each product.   
6. The customer can refine the search further or select a product to view its details.   
  
Alternative Flow:   
1. If no products are found in the selected category, the system displays a message stating that there are no products available.   
2. If the category is inactive or deleted, the system does not display it to the customer.   
3. If the system encounters an error while fetching product data, it logs the error and displays a message to the customer indicating the issue.   
4. If the inventory data cannot be retrieved due to a technical problem, the system displays a message stating that stock status is temporarily unavailable.   
5. If the selected category requires an external API or plugin to display additional data and the API is unavailable, the system shows a message that some information cannot be displayed at this time.  
  
Use Case Name: Add Product to Cart   
Use Case ID: UC-06   
Actors: Customer, System, Product, Inventory, Order   
  
Preconditions:   
1. The customer is logged into the system.   
2. The product exists in the database and is active.   
3. The inventory for the product has sufficient stock.   
4. The cart functionality is enabled in the system.   
  
Postconditions:   
1. The selected product is added to the customer's shopping cart.   
2. The cart is updated to reflect the new item and quantity.   
3. The system logs the addition of the product to the cart.   
4. If inventory is affected, the system updates the inventory database accordingly.   
  
Main Flow:   
1. The customer views a product page or browses the catalog.   
2. The customer selects a product and specifies the quantity.   
3. The customer clicks the "Add to Cart" button.   
4. The system checks the product's availability in the inventory.   
5. The system adds the product and quantity to the customer's cart.   
6. The system updates the cart display to show the added item and total quantity.   
7. The system provides a confirmation message that the product was successfully added.   
  
Alternative Flow:   
1. If the requested quantity exceeds the available stock, the system displays an error message and prevents the addition.   
2. If the product is inactive or deleted, the system shows a message that the product is unavailable.   
3. If the cart functionality is disabled by the administrator, the system informs the customer and suggests an alternative action.   
4. If the system fails to update the cart due to a technical issue, it logs the error and displays a message to the customer to try again.   
5. If inventory data is temporarily unavailable, the system displays a message that stock status cannot be checked at this time.  
  
Use Case Name: Remove Product from Cart   
Use Case ID: UC-07   
Actors: Customer, System, Product, Order, Inventory   
  
Preconditions:   
1. The customer is logged into the system.   
2. The customer has at least one product in their shopping cart.   
3. The cart functionality is enabled in the system.   
4. The system has access to the product and inventory databases.   
  
Postconditions:   
1. The selected product is successfully removed from the customer's cart.   
2. The cart is updated to reflect the removal.   
3. The system logs the removal of the product from the cart.   
4. If inventory is affected, the system updates the inventory database accordingly.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The customer selects the product they wish to remove.   
3. The customer clicks the "Remove" or "Delete" button for the selected product.   
4. The system verifies the product's presence in the cart.   
5. The system removes the product and its quantity from the cart.   
6. The system updates the cart display to reflect the changes.   
7. The system provides a confirmation message that the product was successfully removed.   
  
Alternative Flow:   
1. If the product is not in the cart, the system displays an error message and does not proceed.   
2. If the cart is empty, the system displays a message indicating that no items are available to remove.   
3. If the system fails to update the cart due to a technical issue, it logs the error and displays a message to the customer to try again.   
4. If the inventory update fails, the system logs the issue but still removes the product from the cart, with a message that inventory update is pending.  
  
Use Case Name: Checkout and Place Order   
Use Case ID: UC-08   
Actors: Customer, System, Order, Product, Inventory, Payment, Email   
  
Preconditions:   
1. The customer is logged into the system.   
2. The customer's cart contains at least one product.   
3. The product is active and available in the inventory.   
4. The payment gateway and order placement functionality are enabled.   
5. The system has access to the inventory, order, and payment databases.   
  
Postconditions:   
1. A new order is successfully created in the system.   
2. The inventory is updated to reflect the reduction in stock.   
3. The payment is processed and recorded.   
4. The customer receives an order confirmation email.   
5. The cart is cleared of the purchased items.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The customer reviews the items in the cart and proceeds to checkout.   
3. The system displays the checkout form, requiring shipping address, billing address, and payment method.   
4. The customer fills in the required information and selects a payment method.   
5. The system validates the entered information and confirms the total order amount.   
6. The system checks the inventory to ensure all items are still in stock.   
7. The customer confirms the order and clicks the "Place Order" button.   
8. The system initiates the payment process through the selected payment method.   
9. The system creates a new order record with the customer details, product list, and payment information.   
10. The system updates the inventory by reducing the stock for each ordered product.   
11. The system sends an order confirmation email to the customer.   
12. The system clears the cart and displays a success message.   
  
Alternative Flow:   
1. If the inventory is insufficient for any product, the system displays an error message and prevents order placement.   
2. If the customer enters an invalid shipping or billing address, the system prompts the customer to correct it.   
3. If the payment method fails or is declined, the system displays an error message and allows the customer to try an alternative method.   
4. If the payment gateway is unavailable, the system logs the error and displays a message to the customer to try again later.   
5. If the system fails to create an order due to a technical error, it logs the issue and displays a message to the customer to retry the checkout.   
6. If the confirmation email fails to send, the system logs the error and displays a message to the customer indicating the issue.  
  
Use Case Name: Receive Order Confirmation Email   
Use Case ID: UC-09   
Actors: Customer, System, Email, Order, Payment   
Preconditions:   
1. The customer has successfully placed an order (UC-08).   
2. The system has processed the payment and created the order record.   
3. The email service is configured and operational.   
4. The customer provided a valid email address during registration or checkout.   
  
Postconditions:   
1. The customer receives an email confirming the order details.   
2. The system logs the email delivery attempt.   
3. The customer is aware of the order status and expected delivery.   
  
Main Flow:   
1. The system generates an order confirmation message after the order is successfully placed.   
2. The system compiles the order details, including product names, quantities, prices, total amount, and delivery information.   
3. The system sends the confirmation email to the customer's registered email address.   
4. The email is delivered to the customer's inbox.   
5. The system logs the successful delivery of the email.   
  
Alternative Flow:   
1. If the email address is invalid or delivery fails, the system logs the error and displays a message to the customer that the confirmation email could not be sent.   
2. If the email service is temporarily unavailable, the system logs the error and displays a message to the customer indicating the issue.   
3. If the system fails to generate the email content due to a technical error, it logs the issue and displays a message to the customer that the confirmation email could not be sent.  
  
Use Case Name: Update Customer Information   
Use Case ID: UC-10   
Actors: Customer, System, Administrator, Email   
Preconditions:   
1. The customer is logged into the system with a valid session.   
2. The system has an active customer information update functionality.   
3. The administrator has enabled the update feature and configured necessary settings.   
4. The email service is configured and operational for sending confirmation emails if needed.   
  
Postconditions:   
1. The customer's account information is updated in the database.   
2. The system logs the update activity.   
3. The customer receives a confirmation email if the update involves critical information such as email or password.   
4. The updated information is reflected in the customer's profile.   
  
Main Flow:   
1. The customer navigates to the account settings or profile page.   
2. The customer selects the "Edit Profile" option.   
3. The system displays the current information, allowing the customer to modify fields such as name, email, password, or contact details.   
4. The customer updates the desired information and submits the changes.   
5. The system validates the new data (e.g., email format, password complexity).   
6. If the email is changed, the system sends a confirmation email to the new address.   
7. If the password is changed, the system hashes the new password and updates the record.   
8. The system updates the customer's information in the database.   
9. The system provides a confirmation message that the profile has been successfully updated.   
  
Alternative Flow:   
1. If the new email is already registered, the system displays an error message and does not proceed with the update.   
2. If the password does not meet the security requirements, the system prompts the customer to choose a stronger password.   
3. If the confirmation email fails to send, the system logs the error and displays a message to the customer.   
4. If the system cannot update the customer information due to a technical error, it logs the issue and displays a message to the customer to try again later.  
  
Use Case Name: Delete Customer Account   
Use Case ID: UC-11   
Actors: Customer, System, Administrator, Email   
  
Preconditions:   
1. The customer is logged into the system with a valid session.   
2. The account deletion functionality is enabled and configured by the administrator.   
3. The system is operational and accessible to the customer.   
4. The email service is configured and operational for sending confirmation emails.   
  
Postconditions:   
1. The customer's account is permanently deleted from the system.   
2. All associated data (e.g., orders, preferences) are removed or anonymized based on system policy.   
3. The system logs the account deletion activity.   
4. The customer receives a confirmation email that their account has been deleted.   
5. The system updates the customer database to reflect the deletion.   
  
Main Flow:   
1. The customer navigates to the account settings or profile page.   
2. The customer selects the "Delete Account" option.   
3. The system prompts the customer to confirm the deletion and provide a reason (optional).   
4. The customer confirms the deletion and submits the request.   
5. The system verifies the customer's identity (e.g., re-entering password).   
6. The system initiates the deletion process, removing the account and associated data.   
7. The system sends a confirmation email to the customer's registered email address.   
8. The system logs the deletion and displays a success message to the customer.   
9. The customer is redirected to the login or home page.   
  
Alternative Flow:   
1. If the customer fails to verify their identity, the system displays an error message and does not proceed with deletion.   
2. If the account deletion process fails due to a technical error, the system logs the issue and displays a message to the customer to try again later.   
3. If the confirmation email fails to send, the system logs the error and displays a message to the customer indicating the issue.   
4. If the customer cancels the deletion process, the system returns them to their profile page without making any changes.  
  
Use Case Name: Administrator Login   
Use Case ID: UC-12   
Actors: Administrator, System, Email   
  
Preconditions:   
1. The administrator has a valid account in the system.   
2. The system is operational and accessible to the administrator.   
3. The login page for administrators is available and functional.   
  
Postconditions:   
1. The administrator is successfully authenticated and logged into the system.   
2. The system records the login activity.   
3. The administrator is redirected to the administrative dashboard.   
  
Main Flow:   
1. The administrator navigates to the administrator login page.   
2. The administrator enters their registered email and password.   
3. The system validates the input fields for completeness and correct format.   
4. The system verifies the administrator's credentials against the stored information in the administrator database.   
5. If the credentials are valid, the system logs the administrator in and generates a session token.   
6. The system redirects the administrator to the administrative dashboard.   
  
Alternative Flow:   
1. If the email is not registered for an administrator, the system displays an error message indicating that the email is invalid.   
2. If the password is incorrect, the system prompts the administrator to re-enter the correct password.   
3. If the system detects multiple failed login attempts, it temporarily locks the account and sends a notification email to the administrator.   
4. If the session token cannot be generated, the system logs the error and displays a message to the administrator to try again later.  
  
Use Case Name: Administrator Logout   
Use Case ID: UC-13   
Actors: Administrator, System   
Preconditions:   
1. The administrator is logged into the system with a valid session.   
2. The system has an active logout functionality for administrators.   
  
Postconditions:   
1. The administrator is successfully logged out of the system.   
2. The session token is invalidated.   
3. The administrator is redirected to the login or home page.   
  
Main Flow:   
1. The administrator navigates to the account settings or user profile page within the administrative interface.   
2. The administrator selects the "Logout" option.   
3. The system verifies the session and initiates the logout process.   
4. The system invalidates the session token and clears the session data.   
5. The system redirects the administrator to the login or home page.   
  
Alternative Flow:   
1. If the session token is already expired or invalid, the system displays a message indicating that the administrator is not logged in.   
2. If the logout process fails due to a technical error, the system logs the error and displays a message to the administrator to try again later.  
  
Use Case Name: Add New Product   
Use Case ID: UC-14   
Actors: Administrator, System, Product, Inventory, Category, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The system has access to the product, inventory, and category databases.   
3. The product addition functionality is enabled and configured.   
4. The system supports integration with an API or plugin for additional product data (if required).   
  
Postconditions:   
1. A new product is successfully added to the system's database.   
2. The inventory is updated with the initial stock quantity.   
3. The product is assigned to the correct category.   
4. If an API or plugin is used, additional product data is fetched and stored.   
5. The system logs the product addition activity.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The administrator selects the "Add New Product" option.   
3. The system displays a form for entering product details such as name, description, price, category, and stock quantity.   
4. The administrator fills in the required product information and uploads product images if applicable.   
5. The system validates the input data (e.g., price format, valid category, stock quantity).   
6. The system checks for any required API or plugin integration and fetches additional data if needed.   
7. The system creates a new product record in the database and assigns it to the specified category.   
8. The system updates the inventory with the initial stock quantity.   
9. The system provides a confirmation message that the product was successfully added.   
  
Alternative Flow:   
1. If the product name or SKU already exists, the system displays an error message and prevents the addition.   
2. If the category is invalid or not selected, the system prompts the administrator to choose a valid category.   
3. If the stock quantity is less than or equal to zero, the system displays an error message and prevents the addition.   
4. If the system encounters an error while saving the product record, it logs the error and displays a message to the administrator to retry the operation.   
5. If the API or plugin is unavailable, the system logs the issue and displays a message that some data could not be retrieved.  
  
Use Case Name: Update Product Details   
Use Case ID: UC-15   
Actors: Administrator, System, Product, Inventory, Category, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The product exists in the database and is active.   
3. The system has access to the product, inventory, and category databases.   
4. The product update functionality is enabled and configured.   
5. The system supports integration with an API or plugin for additional product data (if required).   
  
Postconditions:   
1. The product's details in the system are successfully updated.   
2. The inventory is updated to reflect any changes in stock quantity.   
3. The product is assigned to the correct category if modified.   
4. If an API or plugin is used, additional product data is fetched and updated.   
5. The system logs the product update activity.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The administrator selects an existing product to update.   
3. The system displays a form with the current product details, including name, description, price, category, and stock quantity.   
4. The administrator modifies the necessary fields and uploads updated product images if applicable.   
5. The system validates the updated data (e.g., price format, valid category, stock quantity).   
6. The system checks for any required API or plugin integration and fetches additional data if needed.   
7. The system updates the product record in the database and adjusts the category if changed.   
8. The system updates the inventory with the new stock quantity.   
9. The system provides a confirmation message that the product was successfully updated.   
  
Alternative Flow:   
1. If the product name or SKU is changed to a duplicate entry, the system displays an error message and prevents the update.   
2. If the category is invalid or not selected, the system prompts the administrator to choose a valid category.   
3. If the stock quantity is less than or equal to zero, the system displays an error message and prevents the update.   
4. If the system encounters an error while saving the updated product record, it logs the error and displays a message to the administrator to retry the operation.   
5. If the API or plugin is unavailable, the system logs the issue and displays a message that some data could not be retrieved or updated.  
  
Use Case Name: Delete Product   
Use Case ID: UC-16   
Actors: Administrator, System, Product, Inventory, Category, API, Documentation   
  
Preconditions:   
1. The administrator is logged into the system.   
2. The product exists in the database and is active.   
3. The system has access to the product, inventory, and category databases.   
4. The product deletion functionality is enabled and configured.   
5. The system supports integration with an API or plugin for additional product data (if required).   
  
Postconditions:   
1. The product is marked as deleted or removed from the database.   
2. The inventory is updated to reflect the removal of the product.   
3. The product is no longer visible to customers in the catalog or search results.   
4. If an API or plugin is used, any related product data is also updated or removed.   
5. The system logs the product deletion activity.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The administrator selects the product to be deleted.   
3. The system displays the product details and prompts the administrator to confirm the deletion.   
4. The administrator confirms the deletion.   
5. The system verifies the product’s existence in the database.   
6. The system marks the product as deleted or removes it from the database.   
7. The system updates the inventory to remove any stock associated with the product.   
8. The system updates the category database to remove the product from its assigned category.   
9. The system provides a confirmation message that the product was successfully deleted.   
  
Alternative Flow:   
1. If the product is not found in the database, the system displays an error message and does not proceed with deletion.   
2. If the product is referenced in an active order, the system displays a warning and prevents deletion.   
3. If the system encounters an error while deleting the product record, it logs the error and displays a message to the administrator to retry the operation.   
4. If the inventory update fails, the system logs the issue but still marks the product as deleted, with a message that inventory update is pending.   
5. If the API or plugin is unavailable, the system logs the issue and displays a message that some data could not be updated.  
  
Use Case Name: Manage Inventory Levels   
Use Case ID: UC-17   
Actors: Administrator, System, Inventory, Product, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The inventory management functionality is enabled and configured.   
3. The product exists in the system and is associated with an inventory record.   
4. The system has access to the inventory and product databases.   
5. The system supports integration with an API or plugin for external inventory updates (if applicable).   
  
Postconditions:   
1. The inventory levels for the selected product are updated in the system.   
2. The system logs the inventory update activity.   
3. If an API or plugin is used, the external inventory system is also updated.   
4. The product's availability is adjusted based on the updated inventory.   
5. The system displays a confirmation message to the administrator.   
  
Main Flow:   
1. The administrator navigates to the inventory management section of the system.   
2. The administrator selects the product for which inventory levels need to be updated.   
3. The system displays the current inventory level for the selected product.   
4. The administrator enters the new stock quantity.   
5. The system validates the input (e.g., numeric value, non-negative quantity).   
6. The system updates the inventory record in the database with the new quantity.   
7. If the product is linked to an API or plugin, the system sends the updated quantity to the external inventory system.   
8. The system updates the product's availability status based on the new inventory level.   
9. The system provides a confirmation message that the inventory was successfully updated.   
  
Alternative Flow:   
1. If the entered quantity is negative or non-numeric, the system displays an error message and prevents the update.   
2. If the inventory update fails due to a database error, the system logs the issue and displays a message to the administrator to retry the operation.   
3. If the API or plugin is unavailable, the system logs the issue and displays a message that the external inventory system could not be updated.   
4. If the product does not exist in the inventory database, the system displays an error message and does not proceed with the update.  
  
Use Case Name: View Inventory Report   
Use Case ID: UC-18   
Actors: Administrator, System, Inventory, Product, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The inventory report functionality is enabled and configured.   
3. The system has access to the inventory and product databases.   
4. The system supports integration with an API or plugin for external inventory data (if applicable).   
  
Postconditions:   
1. The administrator receives a detailed inventory report.   
2. The report includes product names, stock levels, and category information.   
3. If an API or plugin is used, the report also includes external inventory data.   
4. The system logs the report generation activity.   
5. The administrator can export or view the report as needed.   
  
Main Flow:   
1. The administrator navigates to the inventory management section of the system.   
2. The administrator selects the "Generate Inventory Report" option.   
3. The system retrieves inventory data from the database, including product stock levels and status.   
4. The system compiles the data into a structured report.   
5. The system displays the report to the administrator, including filters for category, product, or date range.   
6. The administrator can choose to export the report in a desired format (e.g., PDF, Excel).   
7. The system provides a confirmation message that the report is ready for viewing or export.   
  
Alternative Flow:   
1. If the inventory database is inaccessible, the system logs the error and displays a message to the administrator.   
2. If the API or plugin is unavailable, the system logs the issue and displays a message that external inventory data could not be retrieved.   
3. If no data is found based on the selected filters, the system displays a message that no inventory items match the criteria.   
4. If the system fails to generate the report due to a technical error, it logs the issue and displays a message to the administrator to try again later.  
  
Use Case Name: Create and Update Product Categories   
Use Case ID: UC-19   
Actors: Administrator, System, Category, Product, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The category management functionality is enabled and configured.   
3. The system has access to the category and product databases.   
4. The system supports integration with an API or plugin for external category data (if applicable).   
  
Postconditions:   
1. A new product category is successfully created or an existing category is updated.   
2. The category is stored in the system's database.   
3. The system logs the category creation or update activity.   
4. If an API or plugin is used, the external system is also updated.   
5. Products can be assigned or reassigned to the updated or new category.   
  
Main Flow:   
1. The administrator navigates to the category management section of the system.   
2. The administrator selects either "Create New Category" or "Edit Existing Category."   
3. The system displays a form with fields for category name, description, and parent category (if hierarchical).   
4. The administrator fills in the necessary information for the new or updated category.   
5. The system validates the input data (e.g., unique category name, proper formatting).   
6. The system checks for any required API or plugin integration and fetches additional data if needed.   
7. The system creates or updates the category record in the database.   
8. The system provides a confirmation message that the category was successfully created or updated.   
  
Alternative Flow:   
1. If the category name is already in use, the system displays an error message and prevents the creation or update.   
2. If the system encounters an error while saving the category, it logs the error and displays a message to the administrator to retry the operation.   
3. If the API or plugin is unavailable, the system logs the issue and displays a message that some data could not be retrieved or updated.   
4. If the category is referenced by active products and the administrator attempts to delete it, the system displays a warning and prevents the deletion.  
  
Use Case Name: Delete Product Category   
Use Case ID: UC-20   
Actors: Administrator, System, Category, Product, API, Documentation   
  
Preconditions:   
1. The administrator is logged into the system.   
2. The category management functionality is enabled and configured.   
3. The product category exists in the database and is active.   
4. The system has access to the category and product databases.   
5. The system supports integration with an API or plugin for external category data (if applicable).   
  
Postconditions:   
1. The product category is marked as deleted or removed from the database.   
2. All products previously assigned to the category are either reassigned or marked as uncategorized.   
3. The system logs the category deletion activity.   
4. If an API or plugin is used, the external system is also updated.   
5. The category is no longer visible in the system for customers or administrators.   
  
Main Flow:   
1. The administrator navigates to the category management section of the system.   
2. The administrator selects the category to be deleted.   
3. The system displays the category details and prompts the administrator to confirm the deletion.   
4. The system checks if any products are still assigned to the category.   
5. The administrator confirms the deletion and selects an option to reassign products or mark them as uncategorized.   
6. The system verifies the category’s existence in the database.   
7. The system marks the category as deleted or removes it from the database.   
8. The system updates the product database to reassign or unassign products as specified.   
9. The system provides a confirmation message that the category was successfully deleted.   
  
Alternative Flow:   
1. If the category is not found in the database, the system displays an error message and does not proceed with deletion.   
2. If the category is referenced by active products and the administrator does not choose to reassign them, the system displays a warning and prevents deletion.   
3. If the system encounters an error while deleting the category, it logs the error and displays a message to the administrator to retry the operation.   
4. If the product reassignment fails due to a technical error, the system logs the issue but still deletes the category, with a message that the product update is pending.   
5. If the API or plugin is unavailable, the system logs the issue and displays a message that some data could not be updated.  
  
Use Case Name: View Category List   
Use Case ID: UC-21   
Actors: Customer, Administrator, System, Category   
Preconditions:   
1. The system is operational and accessible to the user.   
2. The category database contains at least one active category.   
3. The category list functionality is enabled by the administrator.   
4. The user (customer or administrator) is either logged in or has access to the public catalog.   
  
Postconditions:   
1. The user is presented with a list of active categories.   
2. The system logs the user's access to the category list.   
3. If an API or plugin is used, the system fetches and displays additional category information as needed.   
  
Main Flow:   
1. The user navigates to the product catalog or category section of the system.   
2. The system retrieves the list of active categories from the database.   
3. The system displays the category list to the user, including category names and descriptions.   
4. The user may select a category to view its associated products or navigate further.   
  
Alternative Flow:   
1. If no active categories are found, the system displays a message indicating that no categories are available.   
2. If the category list functionality is disabled, the system informs the user and suggests an alternative action.   
3. If the system encounters an error while fetching the category list, it logs the issue and displays a message to the user to try again later.   
4. If an API or plugin is required to display additional data and is unavailable, the system displays a message that some information cannot be displayed at this time.  
  
Use Case Name: Process Payment   
Use Case ID: UC-22   
Actors: Customer, System, Payment, Order, Inventory, Email   
  
Preconditions:   
1. The customer has selected products and completed the checkout process (UC-08).   
2. The system has verified the customer's shipping and billing information.   
3. The inventory has confirmed the availability of all ordered items.   
4. The payment gateway is configured and operational.   
5. The system has access to the payment and order databases.   
  
Postconditions:   
1. The payment is successfully processed and recorded in the system.   
2. The order status is updated to reflect the payment completion.   
3. The inventory is updated to reflect the reduction in stock for the purchased items.   
4. The customer receives a payment confirmation and order confirmation via email.   
5. The system logs the payment transaction for auditing and reporting purposes.   
  
Main Flow:   
1. The customer confirms the order and selects a payment method during checkout.   
2. The system initiates the payment process by sending the transaction request to the selected payment gateway.   
3. The system receives a confirmation from the payment gateway that the transaction is successful.   
4. The system updates the order status to "Paid" and records the payment details.   
5. The system updates the inventory by reducing the stock for each purchased product.   
6. The system generates and sends a payment and order confirmation email to the customer.   
7. The system logs the transaction details and displays a success message to the customer.   
  
Alternative Flow:   
1. If the payment is declined by the gateway, the system displays an error message and allows the customer to try a different payment method.   
2. If the payment gateway is unavailable, the system logs the error and displays a message to the customer to try again later.   
3. If the system fails to update the order status, it logs the error and displays a message to the customer to contact support.   
4. If the inventory update fails, the system logs the issue but still confirms the payment, with a message that inventory update is pending.   
5. If the confirmation email fails to send, the system logs the error and displays a message to the customer indicating the issue.  
  
Use Case Name: View Payment History   
Use Case ID: UC-23   
Actors: Customer, System, Payment, Order, Email   
  
Preconditions:   
1. The customer is logged into the system with a valid session.   
2. The customer has at least one completed order with associated payment records.   
3. The payment history functionality is enabled and configured.   
4. The system has access to the payment and order databases.   
5. The email service is operational for sending notifications if needed.   
  
Postconditions:   
1. The customer views a detailed list of their previous payments.   
2. The system logs the customer's access to the payment history.   
3. If an API or plugin is used, the system fetches and displays additional payment information as needed.   
4. The customer can filter, sort, or export the payment history.   
  
Main Flow:   
1. The customer navigates to the "Payment History" section in their account dashboard.   
2. The system retrieves all payment records associated with the customer's account from the database.   
3. The system displays the payment history, including transaction date, amount, payment method, and order reference.   
4. The customer can filter the payment history by date range, payment method, or order status.   
5. The system updates the displayed payment history based on the selected filters.   
6. The customer can view detailed information for a specific payment by selecting it.   
7. The system provides a summary of the selected payment, including any relevant transaction IDs or receipts.   
8. The customer can export the payment history in a desired format (e.g., PDF, CSV).   
9. The system confirms the export and provides a download link or file.   
  
Alternative Flow:   
1. If no payment records are found, the system displays a message indicating that there is no payment history available.   
2. If the payment history functionality is disabled, the system informs the customer and suggests an alternative method for retrieving payment information.   
3. If the system encounters an error while fetching payment data, it logs the issue and displays a message to the customer to try again later.   
4. If an API or plugin is required to display additional payment details and is unavailable, the system shows a message that some information cannot be displayed at this time.   
5. If the export process fails, the system logs the error and displays a message to the customer to retry the export.  
  
Use Case Name: Manage Plugin Installation   
Use Case ID: UC-24   
Actors: Administrator, System, Plugin, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The plugin management functionality is enabled and accessible.   
3. The system has access to the plugin database and external API resources.   
4. The plugin documentation is available for reference.   
5. The system is connected to a plugin repository or source.   
  
Postconditions:   
1. The plugin is successfully installed or uninstalled in the system.   
2. The system logs the plugin installation or removal activity.   
3. The plugin status is updated in the plugin database.   
4. If an API is involved, the system updates or removes the API integration.   
5. The administrator receives a confirmation message of the action taken.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The administrator selects either "Install Plugin" or "Uninstall Plugin."   
3. If installing, the system displays a list of available plugins from the repository.   
4. The administrator selects a plugin and reviews its description and documentation.   
5. The administrator confirms the installation or uninstallation request.   
6. The system validates the plugin's compatibility and dependencies.   
7. The system performs the installation or uninstallation process.   
8. If the plugin involves an API, the system configures or removes the API connection.   
9. The system updates the plugin status in the database.   
10. The system provides a confirmation message to the administrator.   
  
Alternative Flow:   
1. If the plugin is already installed, the system displays a message and does not proceed with installation.   
2. If the plugin is not compatible with the system, the system logs the issue and displays a message to the administrator.   
3. If the plugin repository is unavailable, the system logs the error and displays a message that plugins cannot be fetched at this time.   
4. If the API integration fails during installation, the system logs the issue and displays a message to the administrator.   
5. If the system encounters a technical error during the process, it logs the error and displays a message to the administrator to retry the operation.  
  
Use Case Name: Develop New Plugin   
Use Case ID: UC-25   
Actors: Administrator, System, Plugin, API, Documentation   
Preconditions:   
1. The administrator is logged into the system.   
2. The plugin development functionality is enabled and accessible.   
3. The system has access to the plugin database and external API resources.   
4. The plugin documentation is available for reference.   
5. The system is connected to a plugin development framework or repository.   
  
Postconditions:   
1. A new plugin is successfully developed and registered in the system.   
2. The plugin status is updated in the plugin database.   
3. The system logs the plugin development activity.   
4. If the plugin integrates with an API, the API connection is configured.   
5. The administrator receives a confirmation message that the plugin is ready for testing or deployment.   
  
Main Flow:   
1. The administrator navigates to the plugin development section of the system.   
2. The administrator selects the "Develop New Plugin" option.   
3. The system provides a development interface or tools for creating the plugin.   
4. The administrator writes the plugin code and configures its settings.   
5. The administrator reviews the plugin documentation to ensure compliance with system standards.   
6. The administrator selects a category or functionality area where the plugin will be used.   
7. The administrator tests the plugin for functionality and compatibility.   
8. The system validates the plugin against predefined criteria (e.g., security, syntax, dependencies).   
9. The system registers the plugin in the database and marks it as "in development" or "ready for deployment."   
10. The system provides a confirmation message to the administrator.   
  
Alternative Flow:   
1. If the plugin code fails validation (e.g., syntax errors, missing dependencies), the system displays an error message and prompts the administrator to fix the issue.   
2. If the plugin is not compatible with the system framework, the system logs the issue and displays a message to the administrator.   
3. If the documentation is incomplete or not provided, the system displays a warning and prevents registration until documentation is added.   
4. If the system encounters an error during the plugin registration process, it logs the error and displays a message to the administrator to retry the operation.   
5. If the API integration is required but not properly configured, the system logs the issue and displays a message to the administrator to resolve the API setup.  
  
Use Case Name: Update Plugin   
Use Case ID: UC-26   
Actors: Administrator, System, Plugin, API, Documentation   
  
Preconditions:   
1. The administrator is logged into the system.   
2. The plugin exists in the system and is currently active or inactive.   
3. The plugin management functionality is enabled and accessible.   
4. The system has access to the plugin database and external API resources.   
5. The plugin documentation is available for reference.   
  
Postconditions:   
1. The plugin's configuration or version is successfully updated in the system.   
2. The system logs the update activity.   
3. If an API is involved, the API integration is updated accordingly.   
4. The plugin status is updated in the plugin database.   
5. The administrator receives a confirmation message of the update.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The administrator selects an existing plugin to update.   
3. The system displays the plugin's current settings and version.   
4. The administrator modifies the plugin configuration or uploads a new version.   
5. The system validates the new settings or version for compatibility and correctness.   
6. The system updates the plugin in the database and applies the new configuration.   
7. If the plugin uses an API, the system updates the API integration as needed.   
8. The system restarts or reloads the plugin to apply changes.   
9. The system provides a confirmation message that the plugin was successfully updated.   
  
Alternative Flow:   
1. If the new plugin version is not compatible with the system, the system logs the issue and displays a message to the administrator.   
2. If the plugin configuration is invalid, the system prompts the administrator to correct it before proceeding.   
3. If the API integration fails after the update, the system logs the issue and displays a message to the administrator.   
4. If the system fails to restart or reload the plugin, it logs the error and displays a message to the administrator to retry the update.   
5. If the system encounters a technical error during the update, it logs the issue and displays a message to the administrator to try again later.  
  
Use Case Name: Delete Plugin   
Use Case ID: UC-27   
Actors: Administrator, System, Plugin, API, Documentation   
  
Preconditions:   
1. The administrator is logged into the system with a valid session.   
2. The plugin exists in the system and is registered in the plugin database.   
3. The plugin management functionality is enabled and accessible.   
4. The system has access to the plugin database and related API configurations.   
5. The plugin is not currently in use by any active system functionality.   
  
Postconditions:   
1. The plugin is successfully deleted from the system.   
2. The plugin status is updated in the plugin database to reflect deletion.   
3. The system logs the plugin deletion activity.   
4. If the plugin involves an API, the API integration is removed or disabled.   
5. The administrator receives a confirmation message that the plugin has been deleted.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The administrator selects the plugin to be deleted.   
3. The system displays a confirmation dialog and warns about the impact of deletion.   
4. The administrator confirms the deletion request.   
5. The system verifies the plugin's existence in the database.   
6. The system checks if the plugin is in use by any active feature or API.   
7. If the plugin is not in use, the system proceeds to delete it from the database and removes associated configurations.   
8. The system disables or removes any API connections related to the plugin.   
9. The system provides a confirmation message that the plugin has been successfully deleted.   
  
Alternative Flow:   
1. If the plugin is currently in use by a feature or API, the system displays a warning and prevents deletion.   
2. If the system encounters an error while deleting the plugin, it logs the issue and displays a message to the administrator to retry the operation.   
3. If the API connection cannot be removed due to a dependency or error, the system logs the issue but proceeds with the plugin deletion, with a message that API cleanup is pending.   
4. If the plugin is not found in the database, the system displays an error message and does not proceed with deletion.   
5. If the administrator cancels the deletion process, the system returns to the plugin management page without making any changes.  
  
Use Case Name: View API Documentation   
Use Case ID: UC-28   
Actors: Administrator, System, API, Documentation   
Preconditions:   
1. The administrator is logged into the system with a valid session.   
2. The API documentation feature is enabled and configured in the system.   
3. The system has access to the API documentation database or files.   
4. The relevant API is registered and available in the system.   
  
Postconditions:   
1. The administrator is able to view the API documentation for the selected API.   
2. The system logs the access to the API documentation.   
3. The administrator receives a confirmation message or visual feedback that the documentation is displayed.   
4. If the documentation is stored in an external format, the system retrieves and presents it correctly.   
  
Main Flow:   
1. The administrator navigates to the API management section of the system.   
2. The administrator selects an API from the list of registered APIs.   
3. The system retrieves the associated documentation for the selected API.   
4. The system displays the API documentation in a user-friendly format.   
5. The administrator can navigate through the documentation, search for specific content, or export it.   
6. The system records the administrator's access to the documentation.   
  
Alternative Flow:   
1. If the selected API does not have associated documentation, the system displays a message indicating that no documentation is available.   
2. If the documentation is stored externally and the system cannot access it, the system logs the error and displays a message to the administrator.   
3. If the API is not registered in the system, the system displays an error message and prevents the view.   
4. If the documentation retrieval fails due to a technical error, the system logs the issue and displays a message to the administrator to try again later.  
  
Use Case Name: Access System Documentation   
Use Case ID: UC-29   
Actors: Administrator, System, Documentation   
  
Preconditions:   
1. The administrator is logged into the system with a valid session.   
2. The system has an accessible documentation section or module.   
3. The system documentation is available and up to date.   
4. The documentation feature is enabled by the administrator.   
  
Postconditions:   
1. The administrator views the system documentation.   
2. The system logs the access to the documentation.   
3. The administrator is provided with a structured and searchable interface for the documentation.   
4. The system ensures that the documentation is displayed in the correct format (e.g., text, HTML, PDF).   
  
Main Flow:   
1. The administrator navigates to the system documentation section from the administrative interface.   
2. The system retrieves the documentation files or content from the documentation database or storage.   
3. The system displays the documentation in a user-friendly format, including a table of contents and search functionality.   
4. The administrator can browse, search, or download the documentation as needed.   
5. The system logs the access to the documentation.   
  
Alternative Flow:   
1. If the system documentation is not available, the system displays a message that no documentation is accessible at this time.   
2. If the documentation feature is disabled, the system shows an error message and suggests contacting the administrator to enable it.   
3. If the system encounters an error while retrieving documentation, it logs the issue and displays a message to the administrator to try again later.   
4. If the documentation file is corrupted or inaccessible, the system logs the error and displays a message that the documentation could not be displayed.  
  
Use Case Name: View Cart   
Use Case ID: UC-30   
Actors: Customer, System, Product, Cart, Inventory   
  
Preconditions:   
1. The customer is logged into the system.   
2. The cart functionality is enabled and configured.   
3. The system has access to the cart and inventory databases.   
4. The customer has at least one item in their cart (optional).   
  
Postconditions:   
1. The customer is presented with a list of items currently in their shopping cart.   
2. The system displays the total quantity and price of items in the cart.   
3. The system logs the customer's access to the cart.   
4. If an API or plugin is used, the system fetches and displays additional cart-related information as needed.   
5. The inventory status for each item is shown to the customer.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The system retrieves the customer's cart from the database, including all items and their quantities.   
3. The system displays the cart items, including product names, prices, images, and inventory status.   
4. The system calculates and displays the total price of all items in the cart.   
5. The customer can view, edit, or remove items from the cart.   
6. The system provides navigation options to continue shopping or proceed to checkout.   
  
Alternative Flow:   
1. If the cart is empty, the system displays a message indicating that no items are currently in the cart.   
2. If the system cannot retrieve the cart data due to a technical issue, it logs the error and displays a message to the customer to try again later.   
3. If the inventory status for any item cannot be retrieved, the system displays a message that stock status is temporarily unavailable for that item.   
4. If an API or plugin is required to display additional cart details and is unavailable, the system shows a message that some information cannot be displayed at this time.  
  
Use Case Name: Add Product to Cart   
Use Case ID: UC-06   
Actors: Customer, System, Product, Inventory, Order   
  
Preconditions:   
1. The customer is logged into the system.   
2. The product exists in the database and is active.   
3. The inventory for the product has sufficient stock.   
4. The cart functionality is enabled in the system.   
  
Postconditions:   
1. The selected product is added to the customer's shopping cart.   
2. The cart is updated to reflect the new item and quantity.   
3. The system logs the addition of the product to the cart.   
4. If inventory is affected, the system updates the inventory database accordingly.   
  
Main Flow:   
1. The customer views a product page or browses the catalog.   
2. The customer selects a product and specifies the quantity.   
3. The customer clicks the "Add to Cart" button.   
4. The system checks the product's availability in the inventory.   
5. The system adds the product and quantity to the customer's cart.   
6. The system updates the cart display to show the added item and total quantity.   
7. The system provides a confirmation message that the product was successfully added.   
  
Alternative Flow:   
1. If the requested quantity exceeds the available stock, the system displays an error message and prevents the addition.   
2. If the product is inactive or deleted, the system shows a message that the product is unavailable.   
3. If the cart functionality is disabled by the administrator, the system informs the customer and suggests an alternative action.   
4. If the system fails to update the cart due to a technical issue, it logs the error and displays a message to the customer to try again.   
5. If inventory data is temporarily unavailable, the system displays a message that stock status cannot be checked at this time.  
  
Use Case Name: Remove Product from Cart   
Use Case ID: UC-31   
Actors: Customer, System, Product, Cart, Inventory   
  
Preconditions:   
1. The customer is logged into the system.   
2. The customer has at least one product in their shopping cart.   
3. The cart functionality is enabled and accessible.   
4. The system has access to the cart and inventory databases.   
5. The selected product exists in the cart.   
  
Postconditions:   
1. The selected product is removed from the customer's cart.   
2. The cart is updated to reflect the removal, including total price and quantity.   
3. The system logs the removal activity.   
4. If the inventory is impacted (e.g., stock is updated due to removal), the inventory database is updated accordingly.   
5. The customer is informed that the removal was successful.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The customer selects the product they wish to remove from the cart.   
3. The system displays a confirmation dialog or prompt to confirm the removal.   
4. The customer confirms the removal request.   
5. The system verifies the presence of the product in the cart.   
6. The system removes the product and its associated quantity from the cart.   
7. The system recalculates the total cart price and updates the cart display.   
8. The system updates the inventory to increase the stock quantity for the removed product.   
9. The system provides a confirmation message that the product was successfully removed from the cart.   
  
Alternative Flow:   
1. If the product is not found in the cart, the system displays an error message and does not proceed with removal.   
2. If the customer cancels the removal request, the system returns to the cart view without making any changes.   
3. If the cart is empty, the system displays a message that no items are available to remove.   
4. If the system fails to update the cart due to a technical issue, it logs the error and displays a message to the customer to try again.   
5. If the inventory update fails, the system logs the issue but still removes the product from the cart, with a message that inventory update is pending.