项目文档

# Functional Requirement

1. Functional Requirements   
1.1 User Registration Function   
Function ID: FR-01   
Description: The system allows a new user to register an account by providing personal information such as name, email, and password. The system validates the input and sends a confirmation email to the user.   
Input: Name, Email, Password, Preferred Category (optional)   
Output: A new UserAccount entity with a unique AccountID, and a new Customer entity with a unique CustomerID. A confirmation email is sent to the customer’s email address.   
  
1.2 User Login Function   
Function ID: FR-02   
Description: The system allows a registered user to log in by providing a valid email and password. The system authenticates the user and initiates a session.   
Input: Email, Password   
Output: Active session for the user, and a logged login event in the system log.   
  
1.3 User Logout Function   
Function ID: FR-03   
Description: The system allows a logged-in user to terminate their session by clicking the "Logout" button.   
Input: User session information   
Output: Termination of the session, redirection to the login or home page, and a logged logout event in the system log.   
  
1.4 Customer Account Creation Function   
Function ID: FR-04   
Description: The system allows a customer to create their account by entering personal information and agreeing to the terms and conditions.   
Input: Name, Email, Password, Terms and Conditions acceptance   
Output: A new Customer entity with a unique CustomerID and a new UserAccount entity linked to the customer. A confirmation email is sent to the customer’s email address.   
  
1.5 Customer Information Update Function   
Function ID: FR-05   
Description: The system allows a logged-in customer to update their personal and contact information in their account settings.   
Input: Updated Name, Address, Phone, Email (if changed), and valid session   
Output: Updated Customer entity in the database and a confirmation message to the customer. A logged update event is recorded in the system log.   
  
1.6 View Purchase History Function   
Function ID: FR-06   
Description: The system allows a logged-in customer or administrator to view the customer’s purchase history, including order details.   
Input: CustomerID and valid session   
Output: A list of PurchaseHistory entities for the customer, including HistoryID, ProductID, PurchaseDate, Quantity, TotalCost. A logged view event is recorded in the system log.   
  
1.7 Manage Purchase History Function   
Function ID: FR-07   
Description: The system allows an administrator to view, edit, or delete a customer’s purchase history.   
Input: CustomerID, selected PurchaseHistory record, and valid administrator session   
Output: Updated or deleted PurchaseHistory records in the database and a confirmation message to the administrator. A logged management event is recorded in the system log.   
  
1.8 Add Product to Cart Function   
Function ID: FR-08   
Description: The system allows a logged-in customer to add a product to their shopping cart, including the product quantity.   
Input: ProductID, Quantity, CustomerID, and valid session   
Output: Updated ShoppingCart entity with the added Product and updated TotalAmount. A logged add event is recorded in the system log.   
  
1.9 Modify Cart Contents Function   
Function ID: FR-09   
Description: The system allows a logged-in customer to modify the quantity of a product or remove it from their shopping cart.   
Input: CartID, ProductID, Quantity (new), and valid session   
Output: Updated ShoppingCart entity in the database and a confirmation message to the customer. A logged modification event is recorded in the system log.   
  
1.10 View Cart Summary Function   
Function ID: FR-10   
Description: The system allows a logged-in customer to view the summary of their shopping cart, including item details, quantities, prices, and total cost.   
Input: CustomerID and valid session   
Output: A list of products in the ShoppingCart, including product names, quantities, prices, and total cost. Estimated shipping cost and final total amount are displayed. A logged view event is recorded in the system log.   
  
1.11 Checkout Order Function   
Function ID: FR-11   
Description: The system allows a logged-in customer to complete the checkout process by entering shipping information and payment details. The system creates an Order, processes the PaymentInformation, and updates the Inventory.   
Input: ShoppingCartID, ShippingAddress, PaymentInformation (CardNumber, ExpiryDate, CVV, Amount), and valid session   
Output: A new Order entity, updated PaymentInformation, updated Inventory, and a confirmation message to the customer. A logged checkout event is recorded in the system log.   
  
1.12 Confirm Order Function   
Function ID: FR-12   
Description: The system confirms the order after the checkout process is completed, updating the order status and inventory, and generating an OrderConfirmationEmail.   
Input: OrderID, ShoppingCartID, Inventory records, and valid session   
Output: Updated Order status to "confirmed," updated Inventory, and a new OrderConfirmationEmail sent to the customer. A logged confirmation event is recorded in the system log.   
  
1.13 Send Order Confirmation Email Function   
Function ID: FR-13   
Description: The system sends an email confirmation to the customer after the order has been successfully created and payment processed.   
Input: OrderID, Customer Email, Order details (items, total amount, estimated delivery date), and valid email server connection   
Output: A new OrderConfirmationEmail entity, including EmailID, OrderID, SentDate, and Content. A logged email sending event is recorded in the system log.   
  
1.14 Product Inventory Management Function   
Function ID: FR-14   
Description: The system allows an administrator to manage inventory, including updating product stock quantities and other related details.   
Input: ProductID, New Inventory Quantity, and valid administrator session   
Output: Updated Inventory entity, including InventoryID, ProductID, Quantity, Location, and LastUpdated. A logged inventory management event is recorded in the system log.   
  
1.15 Add Product Function   
Function ID: FR-15   
Description: The system allows an administrator to add a new product, including product name, description, price, category, and initial inventory quantity.   
Input: Product name, description, price, category ID, initial inventory quantity, and valid administrator session   
Output: A new Product entity, updated Inventory, and a logged product addition event in the system log.   
  
1.16 Update Product Details Function   
Function ID: FR-16   
Description: The system allows an administrator to update a product’s details, including name, description, price, category, and inventory quantity.   
Input: ProductID, Updated name, description, price, category ID, and inventory quantity, and valid administrator session   
Output: Updated Product and Inventory entities in the database. A logged product update event is recorded in the system log.   
  
1.17 Delete Product Function   
Function ID: FR-17   
Description: The system allows an administrator to delete a product from the database.   
Input: ProductID and valid administrator session   
Output: Deleted Product and updated Inventory records. A logged product deletion event is recorded in the system log.   
  
1.18 View Product Category Function   
Function ID: FR-18   
Description: The system allows a customer or administrator to view products within a selected category.   
Input: CategoryID and valid session   
Output: A list of Product entities associated with the selected Category. A logged category view event is recorded in the system log.   
  
1.19 Manage Product Category Function   
Function ID: FR-19   
Description: The system allows an administrator to add, update, or delete product categories.   
Input: Category name, description, and valid administrator session   
Output: Updated Category entity in the database. A logged category management event is recorded in the system log.   
  
1.20 Install Plugin Function   
Function ID: FR-20   
Description: The system allows an administrator to install a new plugin, including validating compatibility and updating the plugin status.   
Input: Plugin file or Plugin details (Name, Version, Description, APIEndpoint), and valid administrator session   
Output: A new Plugin entity with a unique PluginID and updated status to "installed." A logged plugin installation event is recorded in the system log.   
  
1.21 Configure Plugin Function   
Function ID: FR-21   
Description: The system allows an administrator to configure the settings of an installed plugin.   
Input: PluginID, Configuration key-value pairs, and valid administrator session   
Output: A new or updated PluginConfiguration entity in the database. A logged plugin configuration event is recorded in the system log.

# External Description

\*\*Chapter 2: External Interfaces\*\*  
  
This chapter outlines the external interfaces that the system must interact with in order to fulfill its functional requirements. These interfaces include user interfaces, hardware interfaces, software interfaces (including databases and third-party systems), and communication interfaces (such as email). Each interface is described in terms of its role, interaction method, and data flow.  
  
---  
  
### \*\*2.1 User Interface Output\*\*  
  
The system provides a set of user-facing interfaces for customers and administrators to perform actions such as registration, login, account management, product browsing, cart manipulation, and order processing. These interfaces are primarily web-based and may also include mobile or desktop applications depending on the deployment environment.  
  
- \*\*User Login/Registration Screen\*\*:   
 A web form where users enter their email and password to log in or provide personal details to register.   
 - \*Input\*: Email, password, name, preferred category (optional)   
 - \*Output\*: Confirmation message, session initiation, or redirect to home page.  
  
- \*\*Customer Account Settings Page\*\*:   
 A web interface allowing logged-in users to update their personal information such as name, address, phone, and email.   
 - \*Input\*: Updated fields and valid session   
 - \*Output\*: Updated profile and confirmation message.  
  
- \*\*Shopping Cart Summary Page\*\*:   
 Displays the current contents of the shopping cart including product names, quantities, prices, and total cost.   
 - \*Input\*: CustomerID and valid session   
 - \*Output\*: List of products in the cart, estimated shipping cost, and final total amount.  
  
- \*\*Checkout Page\*\*:   
 Allows users to enter shipping and payment information to complete an order.   
 - \*Input\*: Shipping address, payment details (card number, expiry, CVV, amount)   
 - \*Output\*: Order confirmation and redirection to order summary page.  
  
- \*\*Order History View Page\*\*:   
 Displays a list of past orders for a customer or administrator.   
 - \*Input\*: CustomerID and valid session   
 - \*Output\*: List of purchase history records with order details.  
  
- \*\*Product Management Dashboard (Admin)\*\*:   
 Provides tools for adding, updating, and deleting products, managing inventory, and handling categories.   
 - \*Input\*: Product details, inventory quantity, category details   
 - \*Output\*: Updated product records and confirmation messages.  
  
- \*\*Plugin Management Interface (Admin)\*\*:   
 Allows administrators to install and configure plugins.   
 - \*Input\*: Plugin file or details, configuration settings   
 - \*Output\*: Installed plugin status and configuration updates.  
  
---  
  
### \*\*2.2 Hardware Interface Output\*\*  
  
The system currently does not require direct interaction with any external hardware devices. However, if future enhancements include integration with IoT devices or point-of-sale (POS) terminals, the following hardware interfaces may be considered:  
  
- \*\*Point-of-Sale (POS) Terminal Integration\*\*:   
 If implemented, the system will communicate via standard POS protocols (e.g., PCI DSS-compliant APIs) to process payments directly from physical terminals.   
 - \*Input\*: Payment transaction data (amount, card type, authorization code)   
 - \*Output\*: Transaction confirmation and order creation in the system.  
  
- \*\*Barcode Scanner Integration\*\*:   
 For inventory management, the system may support barcode scanning through USB or Bluetooth-connected scanners.   
 - \*Input\*: Scanned product ID   
 - \*Output\*: Automatic product lookup and inventory adjustment.  
  
\*Note: These interfaces are optional and depend on future feature development.\*  
  
---  
  
### \*\*2.3 Software Interface Output\*\*  
  
The system interacts with several internal and external software components, including databases, authentication services, and third-party APIs.  
  
#### \*\*2.3.1 Database Interfaces\*\*  
  
- \*\*UserAccount Table\*\*:   
 Stores user credentials and session-related data.   
 - \*Inputs\*: Name, Email, Password, SessionToken   
 - \*Outputs\*: UserAccount entity with unique AccountID  
  
- \*\*Customer Table\*\*:   
 Contains personal and contact information for registered customers.   
 - \*Inputs\*: Name, Email, Address, Phone   
 - \*Outputs\*: Customer entity with unique CustomerID  
  
- \*\*Product Table\*\*:   
 Stores product details such as name, description, price, and category.   
 - \*Inputs\*: Name, Description, Price, CategoryID   
 - \*Outputs\*: Product entity with unique ProductID  
  
- \*\*Inventory Table\*\*:   
 Tracks stock levels and location for each product.   
 - \*Inputs\*: ProductID, Quantity, Location   
 - \*Outputs\*: Inventory entity with unique InventoryID  
  
- \*\*Order Table\*\*:   
 Records all customer orders, including status, total cost, and shipping details.   
 - \*Inputs\*: CustomerID, ShoppingCartID, ShippingAddress, TotalAmount   
 - \*Outputs\*: Order entity with unique OrderID  
  
- \*\*PurchaseHistory Table\*\*:   
 Logs historical purchases for audit and reporting purposes.   
 - \*Inputs\*: CustomerID, ProductID, PurchaseDate, Quantity, TotalCost   
 - \*Outputs\*: PurchaseHistory entity with unique HistoryID  
  
- \*\*ShoppingCart Table\*\*:   
 Manages items added by customers before checkout.   
 - \*Inputs\*: ProductID, Quantity, CustomerID   
 - \*Outputs\*: ShoppingCart entity with unique CartID  
  
- \*\*Category Table\*\*:   
 Defines product categories and allows filtering of products.   
 - \*Inputs\*: CategoryName, Description   
 - \*Outputs\*: Category entity with unique CategoryID  
  
- \*\*Plugin Table\*\*:   
 Tracks installed plugins and their configurations.   
 - \*Inputs\*: PluginName, Version, Description, APIEndpoint   
 - \*Outputs\*: Plugin entity with unique PluginID  
  
- \*\*PluginConfiguration Table\*\*:   
 Stores configuration settings for each plugin.   
 - \*Inputs\*: PluginID, ConfigurationKey, ConfigurationValue   
 - \*Outputs\*: PluginConfiguration entity with unique ConfigurationID  
  
- \*\*EmailLog Table\*\*:   
 Logs all outgoing emails sent by the system (e.g., confirmation, order status).   
 - \*Inputs\*: EmailContent, SentDate, RecipientEmail, OrderID   
 - \*Outputs\*: EmailLog entity with unique EmailID  
  
#### \*\*2.3.2 Third-Party Software Interfaces\*\*  
  
- \*\*Email Service Provider (ESP) API\*\*:   
 Used to send confirmation and order-related emails to customers.   
 - \*Input\*: RecipientEmail, Subject, Content, OrderID   
 - \*Output\*: Email sent successfully or error response  
  
- \*\*Payment Gateway API (e.g., Stripe, PayPal)\*\*:   
 Processes credit card payments during checkout.   
 - \*Input\*: CardNumber, ExpiryDate, CVV, Amount   
 - \*Output\*: Payment success/failure status, transaction ID  
  
- \*\*Authentication Service (e.g., OAuth2, JWT)\*\*:   
 Validates user sessions and authenticates login requests.   
 - \*Input\*: Email, Password, Token   
 - \*Output\*: Authentication result (success/failure), session token  
  
---  
  
### \*\*2.4 Communication Interface Output\*\*  
  
The system communicates with external entities using various network-based methods, including email, API calls, and logging events.  
  
#### \*\*2.4.1 Email Communication Interface\*\*  
  
- \*\*Order Confirmation Email\*\*:   
 Sent to the customer after successful checkout and payment processing.   
 - \*Input\*: OrderID, CustomerEmail, OrderDetails (items, total, delivery date)   
 - \*Output\*: Email message with content, sent date, and associated EmailID in the EmailLog table.  
  
- \*\*Account Registration Confirmation Email\*\*:   
 Sent upon successful user registration.   
 - \*Input\*: CustomerEmail, VerificationLink   
 - \*Output\*: Email message with verification link and timestamp.  
  
#### \*\*2.4.2 Logging Interface\*\*  
  
- \*\*System Log Interface\*\*:   
 All user actions (login, logout, cart modification, order creation, etc.) are logged into a centralized system log for auditing and monitoring.   
 - \*Input\*: EventType (e.g., "Login", "Logout", "Order Created"), UserID, Timestamp, Details   
 - \*Output\*: Log entry stored in a database with unique LogID  
  
#### \*\*2.4.3 API Communication Interfaces\*\*  
  
- \*\*External Plugin API Endpoint\*\*:   
 When a plugin is installed, it may expose an API endpoint for integration with the system.   
 - \*Input\*: PluginID, RequestType (GET, POST, PUT, DELETE), Payload   
 - \*Output\*: Plugin response data or error message  
  
- \*\*Third-Party Product API (Optional)\*\*:   
 If integrated, used to fetch additional product details from external marketplaces or suppliers.   
 - \*Input\*: ProductSearchQuery, CategoryFilter   
 - \*Output\*: List of products from external source  
  
---  
  
### \*\*Summary of External Interfaces Covered\*\*  
  
| Interface Type | Description |  
|----------------|-------------|  
| \*\*User Interface\*\* | Web-based forms and dashboards for user interaction, including login, registration, account settings, product browsing, and order management. |  
| \*\*Database Interfaces\*\* | Internal tables for storing user accounts, customer data, products, orders, inventory, logs, and plugin configurations. |  
| \*\*Software Interfaces\*\* | Integration with third-party authentication, payment gateways, and email service providers. |  
| \*\*Communication Interfaces\*\* | Network-based interactions such as sending emails, logging system events, and calling external plugin APIs. |  
  
All external data sources referenced in the functional requirements have been accounted for and clearly defined in this section.

# Use Case

Use Case Name: Register User   
Use Case ID: UC-01   
Actors: User, Administrator   
Preconditions:   
1. The system is accessible to the user.   
2. The user does not have an existing account in the system.   
3. The user has a valid email address.   
  
Postconditions:   
1. A new user account is successfully created in the system.   
2. The user receives a confirmation email to validate their account.   
3. The system logs the registration event.   
  
Main Flow:   
1. The user navigates to the registration page of the system.   
2. The user enters their personal information, including name, email address, and password.   
3. The user selects a preferred category for product browsing.   
4. The system validates the user's input (e.g., checks for valid email format and password strength).   
5. The system generates a unique user ID and stores the user information in the database.   
6. The system sends a confirmation email to the user's provided email address.   
7. The user clicks the confirmation link in the email to verify their account.   
8. The system confirms the user's account and displays a welcome message.   
  
Alternative Flow:   
1. If the email address is already registered, the system displays an error message and does not proceed with registration.   
2. If the user input is invalid (e.g., password does not meet requirements), the system displays a specific error message and prompts the user to re-enter the information.   
3. If the confirmation email fails to send, the system logs the error and displays a message to the user indicating the issue. The user may need to contact support.   
4. If the user does not click the confirmation link within a specified time frame, the registration request is automatically deleted, and the user must re-register.  
  
Use Case Name: Login User   
Use Case ID: UC-02   
Actors: User, Administrator   
Preconditions:   
1. The system is accessible to the user.   
2. The user has an existing and validated account in the system.   
3. The user has entered their registered email address and password.   
  
Postconditions:   
1. The user is successfully authenticated and logged into the system.   
2. The user's session is initiated.   
3. The system logs the login event.   
  
Main Flow:   
1. The user navigates to the login page of the system.   
2. The user enters their registered email address and password.   
3. The user clicks the "Login" button.   
4. The system verifies the email and password against the stored user account information.   
5. If the credentials are valid, the system authenticates the user and redirects them to their dashboard or home page.   
6. The system logs the login event for audit purposes.   
  
Alternative Flow:   
1. If the email address is not registered, the system displays an error message indicating that the email is invalid.   
2. If the password is incorrect, the system displays an error message and prompts the user to re-enter the password.   
3. If the user enters incorrect credentials multiple times, the system may lock the account temporarily and notify the user via email.   
4. If the system detects suspicious login activity, it may require additional authentication (e.g., two-factor authentication) before granting access.  
  
Use Case Name: Logout User   
Use Case ID: UC-03   
Actors: User, Administrator   
Preconditions:   
1. The system is accessible to the user.   
2. The user is currently logged into the system.   
3. The user has navigated to the account or session management section.   
  
Postconditions:   
1. The user's session is terminated.   
2. The user is redirected to the login or home page.   
3. The system logs the logout event for audit purposes.   
  
Main Flow:   
1. The user accesses the account settings or profile page.   
2. The user clicks the "Logout" button.   
3. The system verifies the user's session and initiates the logout process.   
4. The system terminates the session and clears session-related data.   
5. The user is redirected to the login page or home page.   
6. The system logs the logout event in the system log.   
  
Alternative Flow:   
1. If the user is not authenticated, the system redirects them to the login page without logging an event.   
2. If the system fails to terminate the session due to an error, it displays a message to the user and logs the error.   
3. If the user closes the browser without explicitly logging out, the session may expire after a set period of inactivity.   
4. If the system detects an invalid session during logout, it invalidates the session and displays a message to the user.  
  
Use Case Name: Create Customer Account   
Use Case ID: UC-04   
Actors: Customer, Administrator   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer does not have an existing account in the system.   
3. The customer provides valid personal and contact information.   
  
Postconditions:   
1. A new customer account is successfully created in the system.   
2. The system sends a confirmation email to the customer's email address.   
3. The customer receives and validates their account via the confirmation link.   
4. The system logs the account creation event.   
  
Main Flow:   
1. The customer navigates to the account creation page of the system.   
2. The customer enters their personal information, including name, email address, and password.   
3. The customer agrees to the terms and conditions of the system.   
4. The system validates the customer's input (e.g., checks for valid email format and password strength).   
5. The system generates a unique customer ID and stores the customer information in the database.   
6. The system sends a confirmation email to the customer's provided email address.   
7. The customer clicks the confirmation link in the email to validate their account.   
8. The system confirms the customer's account and displays a welcome message.   
  
Alternative Flow:   
1. If the email address is already registered, the system displays an error message and does not proceed with account creation.   
2. If the customer input is invalid (e.g., password does not meet requirements), the system displays a specific error message and prompts the customer to re-enter the information.   
3. If the confirmation email fails to send, the system logs the error and displays a message to the customer indicating the issue. The customer may need to contact support.   
4. If the customer does not click the confirmation link within a specified time frame, the account creation request is automatically deleted, and the customer must re-register.  
  
Use Case Name: Update Customer Information   
Use Case ID: UC-05   
Actors: Customer, Administrator   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has an existing account in the system.   
4. The system allows customers to update their account information.   
  
Postconditions:   
1. The customer's information is successfully updated in the system.   
2. The system logs the update event.   
3. The customer receives a confirmation message indicating the update was successful.   
  
Main Flow:   
1. The customer navigates to the account settings or profile page.   
2. The customer selects the option to update their information.   
3. The system displays the current customer information in a form.   
4. The customer modifies the desired fields (e.g., name, address, contact details).   
5. The customer clicks the "Save Changes" button.   
6. The system validates the updated information (e.g., checks for valid email format if email is modified).   
7. The system updates the customer's information in the database.   
8. The system displays a confirmation message to the customer.   
9. The system logs the update event for audit purposes.   
  
Alternative Flow:   
1. If the customer is not authenticated, the system redirects them to the login page.   
2. If the updated email address is already registered, the system displays an error message and does not update the information.   
3. If the updated information is invalid (e.g., missing required fields or incorrect format), the system displays specific error messages and prompts the customer to correct the data.   
4. If the system fails to update the information due to a database error, it displays an error message to the customer and logs the error for further investigation.  
  
Use Case Name: View Purchase History   
Use Case ID: UC-06   
Actors: Customer, Administrator   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has at least one completed order in the system.   
  
Postconditions:   
1. The customer's purchase history is displayed in a clear and organized format.   
2. The system logs the access to the purchase history.   
3. The customer can view order details, including products, quantities, prices, and order dates.   
  
Main Flow:   
1. The customer navigates to the "Purchase History" section of their account.   
2. The system retrieves the customer's order data from the database.   
3. The system displays a list of all completed orders, including order ID, date, and total amount.   
4. The customer selects a specific order to view more details.   
5. The system displays the order details, including the products in the shopping cart, payment information, and order confirmation status.   
6. The customer can return to the list or navigate to other sections of the account.   
7. The system logs the view event for audit purposes.   
  
Alternative Flow:   
1. If the customer has no completed orders, the system displays a message indicating that there is no purchase history available.   
2. If the system fails to retrieve the purchase history due to a database error, it displays an error message and logs the issue for further investigation.   
3. If the customer is not authenticated, the system redirects them to the login page before allowing access to the purchase history.   
4. If the customer attempts to view another user's purchase history, the system denies access and displays an error message.  
  
Use Case Name: Add Product to Cart   
Use Case ID: UC-07   
Actors: Customer   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The product is available in the system's inventory.   
4. The product details page is displayed to the customer.   
  
Postconditions:   
1. The selected product is added to the customer's shopping cart.   
2. The shopping cart is updated in the system.   
3. The customer receives a confirmation message that the product was added.   
4. The system logs the addition of the product to the cart.   
  
Main Flow:   
1. The customer browses the product catalog and selects a product to view.   
2. The system displays the product details, including price, description, and inventory status.   
3. The customer selects the desired quantity of the product.   
4. The customer clicks the "Add to Cart" button.   
5. The system verifies the product's availability in the inventory.   
6. The system adds the product with the specified quantity to the customer's shopping cart.   
7. The system updates the cart in the database and displays a confirmation message.   
  
Alternative Flow:   
1. If the product is out of stock, the system displays a message indicating that the product is unavailable and does not add it to the cart.   
2. If the selected quantity exceeds the available inventory, the system displays an error message and allows the customer to adjust the quantity.   
3. If the customer is not logged in, the system prompts them to log in or register before adding the product to the cart.   
4. If the system fails to update the cart due to a database error, it displays an error message to the customer and logs the issue for further investigation.  
  
Use Case Name: Modify Cart Contents   
Use Case ID: UC-08   
Actors: Customer   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has at least one item in their shopping cart.   
4. The shopping cart is displayed to the customer.   
  
Postconditions:   
1. The cart contents are updated according to the customer's modifications.   
2. The system logs the cart modification event.   
3. The customer receives a confirmation message that the cart has been updated.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The system displays the current cart contents, including product names, quantities, and prices.   
3. The customer selects an item to modify (e.g., change quantity or remove item).   
4. The customer updates the quantity or clicks the "Remove" button for the selected item.   
5. The system validates the requested changes (e.g., ensures the quantity is within available inventory).   
6. The system updates the cart in the database.   
7. The system displays a confirmation message showing the updated cart.   
  
Alternative Flow:   
1. If the requested quantity is greater than the available inventory, the system displays an error message and allows the customer to adjust the quantity.   
2. If the customer tries to modify a cart item that no longer exists in the inventory, the system removes the item from the cart and displays a message.   
3. If the customer is not logged in, the system prompts them to log in or register before allowing cart modifications.   
4. If the system fails to update the cart due to a database error, it displays an error message to the customer and logs the issue for further investigation.  
  
Use Case Name: View Cart Summary   
Use Case ID: UC-09   
Actors: Customer   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has at least one item in their shopping cart.   
4. The system allows customers to view their cart summary.   
  
Postconditions:   
1. The customer can view a summary of their shopping cart, including product names, quantities, prices, and total amount.   
2. The system logs the cart summary view event.   
3. The cart summary is displayed clearly and accurately.   
  
Main Flow:   
1. The customer navigates to the shopping cart page.   
2. The system retrieves the items currently in the customer's shopping cart from the database.   
3. The system displays the cart summary, including product details, quantity, price, and total cost.   
4. The system also shows the estimated shipping cost and the final total amount.   
5. The customer can proceed to checkout or continue shopping.   
6. The system logs the view event for audit purposes.   
  
Alternative Flow:   
1. If the customer has no items in their cart, the system displays a message indicating that the cart is empty.   
2. If the system fails to retrieve the cart items due to a database error, it displays an error message and logs the issue.   
3. If the inventory status of an item in the cart changes (e.g., out of stock), the system updates the cart summary and displays a message to the customer.   
4. If the customer is not logged in, the system prompts them to log in or register to view the cart summary.  
  
Use Case Name: Checkout Order   
Use Case ID: UC-10   
Actors: Customer, Administrator, Payment Information, Shopping Cart, Order, Inventory   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has at least one item in their shopping cart.   
4. The shopping cart is displayed to the customer.   
5. The customer has selected the items to purchase.   
  
Postconditions:   
1. The order is successfully created in the system.   
2. The payment is processed, and the payment information is updated.   
3. The inventory is updated to reflect the deduction of purchased items.   
4. The system logs the checkout event.   
5. The customer receives an order confirmation email.   
  
Main Flow:   
1. The customer navigates to the checkout page from the shopping cart.   
2. The system displays the cart summary, including product names, quantities, prices, and total amount.   
3. The customer enters or confirms their shipping address and contact information.   
4. The customer selects a preferred payment method and enters the required payment information.   
5. The system verifies the availability of all items in the cart.   
6. The system processes the payment and confirms the transaction.   
7. The system creates a new order with the selected items and updates the order status to "confirmed."   
8. The system deducts the purchased quantities from the inventory.   
9. The system sends an order confirmation email to the customer.   
10. The system logs the checkout event for audit purposes.   
  
Alternative Flow:   
1. If any item in the cart is out of stock, the system displays an error message and allows the customer to adjust the cart.   
2. If the payment information is invalid or the transaction fails, the system displays an error message and allows the customer to re-enter the payment details.   
3. If the customer is not logged in, the system prompts them to log in or register before proceeding to checkout.   
4. If the system fails to update the inventory due to a database error, it displays an error message and logs the issue. The customer may need to contact support.   
5. If the system fails to send the order confirmation email, it logs the error and displays a message to the customer. The customer may need to contact support.  
  
Use Case Name: Confirm Order   
Use Case ID: UC-11   
Actors: Customer, Administrator, Order, Order Confirmation Email, Payment Information, Shopping Cart, Inventory   
Preconditions:   
1. The system is accessible to the customer.   
2. The customer is logged into their account.   
3. The customer has completed the checkout process and the order is created.   
4. The payment has been successfully processed.   
5. The order confirmation email is generated by the system.   
  
Postconditions:   
1. The order confirmation email is successfully sent to the customer.   
2. The order status is updated to "confirmed" in the system.   
3. The inventory is updated to reflect the deduction of purchased items.   
4. The system logs the order confirmation event.   
5. The customer receives a confirmation message and can view their confirmed order in the purchase history.   
  
Main Flow:   
1. After the customer completes the checkout process, the system generates an order confirmation email.   
2. The system updates the order status to "confirmed" in the database.   
3. The system deducts the purchased quantities from the inventory.   
4. The system sends the order confirmation email to the customer's registered email address.   
5. The customer receives the email with the order details, including order ID, items purchased, total amount, and estimated delivery date.   
6. The system logs the order confirmation and email sending event.   
7. The customer is redirected to a confirmation page or receives a message indicating the order has been successfully confirmed.   
  
Alternative Flow:   
1. If the system fails to send the order confirmation email, it logs the error and displays a message to the customer. The customer may need to contact support.   
2. If the inventory update fails due to a database error, the system rolls back the order and displays an error message to the customer. The customer may need to contact support.   
3. If the customer's email address is invalid or no longer registered, the system logs the issue and displays a message to the customer. The customer may need to update their email address in their account settings.   
4. If the customer attempts to confirm an order that has already been confirmed, the system displays a message indicating that the order is already confirmed and provides the order details.  
  
Use Case Name: Send Order Confirmation Email   
Use Case ID: UC-12   
Actors: Customer, Administrator, Order, Order Confirmation Email, Payment Information, Shopping Cart, Inventory   
Preconditions:   
1. The system is accessible to the customer.   
2. The order has been successfully created and the payment has been processed.   
3. The customer has a valid and registered email address.   
4. The system has access to the customer's order details.   
5. The system is connected to an email server or plugin to send emails.   
  
Postconditions:   
1. The order confirmation email is successfully sent to the customer's email address.   
2. The email contains the order details, including order ID, items, total amount, and estimated delivery date.   
3. The system logs the event of sending the confirmation email.   
4. The customer is notified of the successful order confirmation via the email.   
  
Main Flow:   
1. The system generates the order confirmation email content after the order is created and the payment is processed.   
2. The system retrieves the customer's registered email address from the user account.   
3. The system sends the order confirmation email using the email plugin or server.   
4. The customer receives the email and views the order confirmation details.   
5. The system logs the email sending event for audit purposes.   
  
Alternative Flow:   
1. If the system fails to connect to the email server, it logs the error and displays a message to the customer indicating the issue. The customer may need to contact support.   
2. If the customer's email address is invalid or cannot be verified, the system logs the error and displays a message to the customer. The customer may need to update their email in their account settings.   
3. If the email plugin fails to send the confirmation email, the system logs the error and displays a message to the customer. The customer may need to contact support or check their spam folder.   
4. If the customer does not receive the confirmation email within a reasonable time, the system allows the customer to request a re-sending of the email through the account settings or contact support.  
  
Use Case Name: Manage Inventory   
Use Case ID: UC-13   
Actors: Administrator, Inventory, Product, Category, Plugin   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to manage inventory.   
4. The inventory database is available and functioning.   
5. The product and category information is up to date.   
  
Postconditions:   
1. The inventory levels for products are updated based on the administrator's actions.   
2. The system logs the inventory management event.   
3. The product information is updated in the database if necessary.   
4. The inventory status is reflected accurately on the product pages.   
  
Main Flow:   
1. The administrator navigates to the inventory management section of the system.   
2. The system displays a list of products along with their current inventory levels and categories.   
3. The administrator selects a product to update its inventory.   
4. The administrator enters the new inventory quantity or modifies product details (e.g., price, category).   
5. The system validates the input (e.g., ensures quantity is a non-negative number).   
6. The system updates the inventory and product information in the database.   
7. The system logs the update event for audit purposes.   
8. The system displays a confirmation message to the administrator that the inventory has been successfully managed.   
  
Alternative Flow:   
1. If the administrator does not have permission to manage inventory, the system displays an error message and denies access.   
2. If the inventory quantity is invalid (e.g., negative number), the system displays an error message and prompts the administrator to re-enter the correct quantity.   
3. If the system fails to update the inventory due to a database error, it displays an error message to the administrator and logs the issue for further investigation.   
4. If the selected product is no longer available in the system, the system displays a message indicating the product is not found and prompts the administrator to select a valid product.  
  
Use Case Name: Add Product   
Use Case ID: UC-04   
Actors: Administrator, Product, Inventory, Category   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to add products.   
4. The inventory and category databases are available and functioning.   
5. The product details are prepared for entry.   
  
Postconditions:   
1. The new product is successfully added to the system's database.   
2. The inventory is updated to reflect the added product.   
3. The product is associated with the correct category.   
4. The system logs the product addition event.   
5. The product becomes visible in the catalog for browsing and purchasing.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The administrator selects the option to add a new product.   
3. The system displays a product form with fields for product name, description, price, category, and inventory quantity.   
4. The administrator fills in the required product information.   
5. The administrator selects a category for the product from the category list.   
6. The system validates the input (e.g., ensures price is a valid number and inventory quantity is non-negative).   
7. The system saves the product information to the database and updates the inventory.   
8. The system logs the event and displays a confirmation message to the administrator.   
  
Alternative Flow:   
1. If the administrator does not have permission to add products, the system displays an error message and denies access.   
2. If the product name is missing or invalid, the system displays an error message and prompts the administrator to provide valid information.   
3. If the system fails to save the product due to a database error, it displays an error message and logs the issue for further investigation.   
4. If the selected category does not exist, the system displays an error message and prompts the administrator to select a valid category.  
  
Use Case Name: Update Product Details   
Use Case ID: UC-14   
Actors: Administrator, Product, Inventory, Category   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to update product details.   
4. The product to be updated exists in the system's database.   
5. The inventory and category databases are available and functioning.   
  
Postconditions:   
1. The product details are successfully updated in the database.   
2. The inventory and category information are updated accordingly.   
3. The system logs the product update event.   
4. The updated product information is visible in the catalog for customers to browse.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The system displays a list of products with their current details, including name, description, price, category, and inventory.   
3. The administrator selects a specific product to update.   
4. The system displays a form with the product's current information.   
5. The administrator modifies the product details as needed (e.g., price, description, category, or inventory quantity).   
6. The administrator clicks the "Save Changes" button.   
7. The system validates the updated information (e.g., checks for valid price format and non-negative inventory quantity).   
8. The system updates the product details, inventory, and category associations in the database.   
9. The system logs the update event for audit purposes.   
10. The system displays a confirmation message to the administrator that the product has been successfully updated.   
  
Alternative Flow:   
1. If the administrator does not have permission to update product details, the system displays an error message and denies access.   
2. If the product name is missing or invalid, the system displays an error message and prompts the administrator to provide valid information.   
3. If the updated inventory quantity is negative, the system displays an error message and prompts the administrator to re-enter a valid quantity.   
4. If the system fails to update the product due to a database error, it displays an error message and logs the issue for further investigation.   
5. If the selected category no longer exists, the system displays an error message and prompts the administrator to select a valid category.  
  
Use Case Name: Delete Product   
Use Case ID: UC-15   
Actors: Administrator, Product, Inventory   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to delete products.   
4. The product to be deleted exists in the system's database.   
5. The inventory database is available and functioning.   
  
Postconditions:   
1. The product is successfully removed from the system's database.   
2. The inventory is updated to remove the product's stock.   
3. The system logs the product deletion event.   
4. The product is no longer visible in the catalog for customers to browse.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The system displays a list of products with their details.   
3. The administrator selects a specific product to delete.   
4. The system prompts the administrator to confirm the deletion.   
5. The administrator confirms the deletion.   
6. The system removes the product from the database and updates the inventory.   
7. The system logs the deletion event for audit purposes.   
8. The system displays a confirmation message to the administrator that the product has been successfully deleted.   
  
Alternative Flow:   
1. If the administrator does not have permission to delete products, the system displays an error message and denies access.   
2. If the product is currently in the shopping cart of any customer, the system displays a warning and prevents the deletion until the product is no longer in any cart.   
3. If the system fails to delete the product due to a database error, it displays an error message and logs the issue for further investigation.   
4. If the administrator cancels the deletion, the system retains the product in the database and does not make any changes.  
  
Use Case Name: View Product Category   
Use Case ID: UC-16   
Actors: Customer, Administrator, Category   
Preconditions:   
1. The system is accessible to the customer or administrator.   
2. The customer or administrator is logged into the system.   
3. The product category exists in the system's database.   
4. The category database is available and functioning.   
  
Postconditions:   
1. The customer or administrator can view the products in the selected category.   
2. The system logs the category view event.   
3. The displayed products are accurate and up to date.   
  
Main Flow:   
1. The customer or administrator navigates to the product catalog or category section of the system.   
2. The system displays a list of available product categories.   
3. The customer or administrator selects a specific category to view.   
4. The system retrieves the products associated with the selected category from the database.   
5. The system displays the list of products in the selected category, including product names, descriptions, and prices.   
6. The customer or administrator can further filter or sort the products within the category.   
7. The system logs the category view event for audit purposes.   
  
Alternative Flow:   
1. If the selected category does not exist in the system, the system displays an error message and prompts the user to select a valid category.   
2. If the system fails to retrieve the products due to a database error, it displays an error message and logs the issue for further investigation.   
3. If the customer or administrator is not authenticated, the system redirects them to the login page before allowing access to the category.   
4. If no products are associated with the selected category, the system displays a message indicating that there are no products available in this category.  
  
Use Case Name: Manage Product Category   
Use Case ID: UC-17   
Actors: Administrator, Category, Product   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to manage product categories.   
4. The category database is available and functioning.   
5. The product database is available and functioning.   
  
Postconditions:   
1. The product category is successfully added, updated, or deleted in the system.   
2. The system logs the category management event.   
3. The changes to the category are reflected in the product catalog.   
  
Main Flow:   
1. The administrator navigates to the category management section of the system.   
2. The system displays a list of existing product categories.   
3. The administrator selects an action: "Add New Category," "Edit Category," or "Delete Category."   
4. If "Add New Category" is selected, the administrator enters the new category name and description.   
5. If "Edit Category" is selected, the administrator chooses a category and modifies its name or description.   
6. If "Delete Category" is selected, the system prompts the administrator to confirm the deletion.   
7. The administrator confirms the action.   
8. The system validates the input (e.g., ensures the category name is not empty).   
9. The system updates the category database accordingly.   
10. The system logs the category management event for audit purposes.   
11. The system displays a confirmation message to the administrator.   
  
Alternative Flow:   
1. If the administrator does not have permission to manage categories, the system displays an error message and denies access.   
2. If the category name is missing or invalid, the system displays an error message and prompts the administrator to provide valid information.   
3. If the system fails to update the category due to a database error, it displays an error message and logs the issue for further investigation.   
4. If the administrator cancels the deletion, the system retains the category and does not make any changes.   
5. If the selected category contains products, the system displays a warning and prevents deletion until all products are reassigned or removed.  
  
Use Case Name: Install Plugin   
Use Case ID: UC-18   
Actors: Administrator, Plugin   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to install plugins.   
4. The plugin to be installed is available and compatible with the system.   
5. The system has the required infrastructure (e.g., email server, API gateway) to support the plugin.   
  
Postconditions:   
1. The plugin is successfully installed and activated in the system.   
2. The system logs the plugin installation event.   
3. The plugin's functionality is available to the relevant system components.   
4. The administrator receives a confirmation message that the plugin has been installed.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The system displays a list of available plugins that can be installed.   
3. The administrator selects the plugin to install and confirms the action.   
4. The system validates the plugin (e.g., checks compatibility and dependencies).   
5. The system proceeds to install the plugin using the appropriate installation process.   
6. The system activates the plugin and updates the plugin status in the database.   
7. The system logs the installation event for audit purposes.   
8. The system displays a confirmation message to the administrator indicating the plugin has been successfully installed.   
  
Alternative Flow:   
1. If the administrator does not have permission to install plugins, the system displays an error message and denies access.   
2. If the plugin is incompatible or has missing dependencies, the system displays an error message and prevents installation.   
3. If the plugin installation fails due to a system error, the system logs the error and displays a message to the administrator. The administrator may need to troubleshoot or contact support.   
4. If the administrator cancels the installation, the system does not proceed with the installation and retains the existing plugin configuration.  
  
Use Case Name: Configure Plugin   
Use Case ID: UC-19   
Actors: Administrator, Plugin   
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to configure plugins.   
4. The plugin is installed and available in the system.   
  
Postconditions:   
1. The plugin is successfully configured with the specified settings.   
2. The system logs the plugin configuration event.   
3. The plugin's functionality is updated based on the new configuration.   
4. The administrator receives a confirmation message that the configuration is complete.   
  
Main Flow:   
1. The administrator navigates to the plugin configuration section of the system.   
2. The system displays a list of installed plugins along with their current configuration status.   
3. The administrator selects a plugin to configure and opens the configuration form.   
4. The administrator modifies the plugin settings (e.g., API keys, integration details, or feature toggles).   
5. The administrator clicks the "Save Configuration" button.   
6. The system validates the configuration inputs (e.g., checks for valid API key format or required fields).   
7. The system saves the updated configuration to the database and applies the changes.   
8. The system displays a confirmation message to the administrator that the plugin is configured.   
9. The system logs the configuration event for audit purposes.   
  
Alternative Flow:   
1. If the administrator does not have permission to configure plugins, the system displays an error message and denies access.   
2. If the plugin is not installed, the system displays an error message and prompts the administrator to install the plugin first.   
3. If the configuration input is invalid (e.g., missing required fields or incorrect format), the system displays specific error messages and prompts the administrator to correct the data.   
4. If the system fails to save the configuration due to a database error, it displays an error message to the administrator and logs the issue for further investigation.   
5. If the administrator cancels the configuration, the system retains the plugin's previous settings and does not make any changes.  
  
Use Case Name: Manage Purchase History   
Use Case ID: UC-20   
Actors: Administrator, Customer, Purchase History, User Account   
  
Preconditions:   
1. The system is accessible to the administrator.   
2. The administrator is logged into the system.   
3. The administrator has the necessary permissions to manage purchase history.   
4. The purchase history database is available and functioning.   
5. The customer has at least one order in the system.   
  
Postconditions:   
1. The purchase history for a specific customer is successfully viewed, edited, or deleted by the administrator.   
2. The system logs the event of managing the purchase history.   
3. The updated or deleted purchase history is reflected in the database.   
4. The administrator receives a confirmation message based on the action performed.   
  
Main Flow:   
1. The administrator navigates to the customer management section of the system.   
2. The system displays a list of customers along with basic account information.   
3. The administrator selects a specific customer to manage their purchase history.   
4. The system retrieves and displays the customer's purchase history, including order IDs, dates, product details, and total amounts.   
5. The administrator selects an action: "View Purchase History," "Edit Purchase History," or "Delete Purchase History."   
6. If "View Purchase History" is selected, the system displays the details of the selected order.   
7. If "Edit Purchase History" is selected, the administrator can modify specific details (e.g., order status or notes).   
8. If "Delete Purchase History" is selected, the system prompts the administrator to confirm the deletion.   
9. The administrator confirms the action.   
10. The system updates the database to reflect the changes made.   
11. The system logs the management event for audit purposes.   
12. The system displays a confirmation message to the administrator indicating the action was completed.   
  
Alternative Flow:   
1. If the administrator does not have permission to manage purchase history, the system displays an error message and denies access.   
2. If the selected customer has no purchase history, the system displays a message indicating no orders are available.   
3. If the system fails to retrieve the purchase history due to a database error, it displays an error message and logs the issue for further investigation.   
4. If the administrator cancels the deletion, the system retains the purchase history and does not make any changes.   
5. If the system fails to update the purchase history due to a database error, it displays an error message and logs the issue for further investigation.   
6. If the selected order is invalid or no longer exists, the system displays an error message and prompts the administrator to select a valid order.