项目文档

# Functional Requirement

### Chapter 1: Functional Requirements   
  
#### 1.1 Customer Registration   
\*\*Function ID\*\*: FR-01   
\*\*Description\*\*: Enables new customers to create accounts by providing personal information and credentials. Validates input data and stores details in the Customer entity.   
\*\*Input\*\*: Customer details (Name, Email, Password, Address, PhoneNumber)   
\*\*Output\*\*: New record in Customer data entity; confirmation message.   
  
#### 1.2 Update Customer Information   
\*\*Function ID\*\*: FR-02   
\*\*Description\*\*: Allows authenticated customers to modify their stored personal details. Validates inputs before updating the Customer entity.   
\*\*Input\*\*: Edited details (Address, PhoneNumber, Email) from authenticated customer.   
\*\*Output\*\*: Updated Customer data entity record; confirmation message.   
  
#### 1.3 View Customer Profile   
\*\*Function ID\*\*: FR-03   
\*\*Description\*\*: Displays stored customer information (e.g., contact details, order history) to authenticated users.   
\*\*Input\*\*: Authenticated customer session.   
\*\*Output\*\*: Customer details (Name, Email, Address, PhoneNumber, Order history).   
  
#### 1.4 Add Product   
\*\*Function ID\*\*: FR-04   
\*\*Description\*\*: Allows administrators to create new product entries. Links products to categories and updates inventory.   
\*\*Input\*\*: Product details (ProductName, Description, Price, CategoryID, StockQuantity) from administrator.   
\*\*Output\*\*: New record in Product data entity; updated Inventory record; confirmation message.   
  
#### 1.5 Update Product Details   
\*\*Function ID\*\*: FR-05   
\*\*Description\*\*: Enables administrators to modify product information (e.g., price, stock). Syncs changes to Inventory entity.   
\*\*Input\*\*: Modified product details (Price, Description, StockQuantity, CategoryID).   
\*\*Output\*\*: Updated Product and Inventory records; confirmation message.   
  
#### 1.6 View Product Details   
\*\*Function ID\*\*: FR-06   
\*\*Description\*\*: Displays product information (e.g., name, price, description) to customers or administrators.   
\*\*Input\*\*: Product selection from catalog/search.   
\*\*Output\*\*: Product details (ProductName, Description, Price, Image, Category).   
  
#### 1.7 Remove Product   
\*\*Function ID\*\*: FR-07   
\*\*Description\*\*: Allows administrators to delete products not linked to active orders/carts. Removes associated inventory records.   
\*\*Input\*\*: ProductID selection from administrator.   
\*\*Output\*\*: Deleted Product and Inventory records; confirmation message.   
  
#### 1.8 Place Order   
\*\*Function ID\*\*: FR-08   
\*\*Description\*\*: Converts shopping cart items into a formal order. Validates stock, processes payment, and updates inventory.   
\*\*Input\*\*: Customer’s ShoppingCart items, shipping details, payment method.   
\*\*Output\*\*: New Order record; updated Inventory levels; cleared ShoppingCart; confirmation message.   
  
#### 1.9 View Order Details   
\*\*Function ID\*\*: FR-09   
\*\*Description\*\*: Displays order information (e.g., products, status, total cost) to authenticated customers.   
\*\*Input\*\*: OrderID selection from customer.   
\*\*Output\*\*: Order details (OrderDate, TotalAmount, Product list, Status).   
  
#### 1.10 Update Order Status   
\*\*Function ID\*\*: FR-10   
\*\*Description\*\*: Allows administrators to modify order status (e.g., "Shipped"). Triggers customer notifications if applicable.   
\*\*Input\*\*: New status (e.g., "Processing", "Cancelled") from administrator.   
\*\*Output\*\*: Updated Order status; notification to customer; confirmation message.   
  
#### 1.11 Cancel Order   
\*\*Function ID\*\*: FR-11   
\*\*Description\*\*: Enables customers to cancel eligible orders. Restores inventory stock and updates order status.   
\*\*Input\*\*: OrderID selection from authenticated customer.   
\*\*Output\*\*: Updated Order status ("Cancelled"); restored Inventory levels; confirmation message.   
  
#### 1.12 Add to Shopping Cart   
\*\*Function ID\*\*: FR-12   
\*\*Description\*\*: Adds selected products/quantities to the customer’s cart. Validates stock availability.   
\*\*Input\*\*: ProductID and Quantity from customer.   
\*\*Output\*\*: New CartProductItem record; updated Inventory; confirmation message.   
  
#### 1.13 View Shopping Cart   
\*\*Function ID\*\*: FR-13   
\*\*Description\*\*: Displays cart contents (products, quantities, total cost) to authenticated customers.   
\*\*Input\*\*: Authenticated customer session.   
\*\*Output\*\*: ShoppingCart items, quantities, prices, total cost.   
  
#### 1.14 Modify Shopping Cart Item   
\*\*Function ID\*\*: FR-14   
\*\*Description\*\*: Allows customers to adjust quantities or remove items from their cart. Updates inventory accordingly.   
\*\*Input\*\*: Modified Quantity or removal request for CartProductItem.   
\*\*Output\*\*: Updated CartProductItem/Inventory records; recalculated cart total; confirmation message.   
  
#### 1.15 Process Payment   
\*\*Function ID\*\*: FR-15   
\*\*Description\*\*: Handles payment transactions for confirmed orders. Updates order status and payment history.   
\*\*Input\*\*: Payment details (e.g., card number, expiration date) and OrderID.   
\*\*Output\*\*: Payment record; updated Order status ("Paid"); confirmation message.   
  
#### 1.16 View Payment History   
\*\*Function ID\*\*: FR-16   
\*\*Description\*\*: Displays past payment records (e.g., date, amount, status) to authenticated customers.   
\*\*Input\*\*: Authenticated customer session.   
\*\*Output\*\*: Payment history details (PaymentDate, Amount, Status, OrderID).   
  
#### 1.17 Update Inventory Levels   
\*\*Function ID\*\*: FR-17   
\*\*Description\*\*: Allows administrators to manually adjust stock quantities for products.   
\*\*Input\*\*: ProductID and new StockQuantity from administrator.   
\*\*Output\*\*: Updated Inventory record; confirmation message.   
  
#### 1.18 View Inventory Levels   
\*\*Function ID\*\*: FR-18   
\*\*Description\*\*: Displays current stock levels for all products to administrators.   
\*\*Input\*\*: Administrator request.   
\*\*Output\*\*: Product list with StockLevels.   
  
#### 1.19 Manage Users   
\*\*Function ID\*\*: FR-19   
\*\*Description\*\*: Enables administrators to add, edit, or delete customer accounts.   
\*\*Input\*\*: Customer details (for add/edit) or CustomerID (for delete).   
\*\*Output\*\*: Added/updated/deleted Customer record; confirmation message.   
  
#### 1.20 Install Plugin   
\*\*Function ID\*\*: FR-20   
\*\*Description\*\*: Adds new plugins to the system after validating compatibility and dependencies.   
\*\*Input\*\*: Plugin file or repository selection from administrator.   
\*\*Output\*\*: New Plugin record; system integration; confirmation message.   
  
#### 1.21 Update Plugin   
\*\*Function ID\*\*: FR-21   
\*\*Description\*\*: Modifies existing plugins to new versions. Validates compatibility before updating.   
\*\*Input\*\*: Updated plugin file/version from administrator.   
\*\*Output\*\*: Updated Plugin record; confirmation message.   
  
#### 1.22 Uninstall Plugin   
\*\*Function ID\*\*: FR-22   
\*\*Description\*\*: Removes plugins not in use by active processes.   
\*\*Input\*\*: PluginID selection from administrator.   
\*\*Output\*\*: Removed Plugin record; confirmation message.   
  
#### 1.23 Create Product Category   
\*\*Function ID\*\*: FR-23   
\*\*Description\*\*: Allows administrators to define new product categories.   
\*\*Input\*\*: Category details (CategoryName, Description) from administrator.   
\*\*Output\*\*: New ProductCategory record; confirmation message.   
  
#### 1.24 Update Product Category   
\*\*Function ID\*\*: FR-24   
\*\*Description\*\*: Modifies existing product category details (e.g., name, description).   
\*\*Input\*\*: Modified category details (CategoryName, Description).   
\*\*Output\*\*: Updated ProductCategory record; confirmation message.   
  
#### 1.25 Delete Product Category   
\*\*Function ID\*\*: FR-25   
\*\*Description\*\*: Removes unused product categories not linked to active products.   
\*\*Input\*\*: CategoryID selection from administrator.   
\*\*Output\*\*: Deleted ProductCategory record; confirmation message.   
  
#### 1.26 Manage Payment History   
\*\*Function ID\*\*: FR-26   
\*\*Description\*\*: Allows administrators to filter, view, and export payment records.   
\*\*Input\*\*: Filters (e.g., date range, status); export format (e.g., CSV).   
\*\*Output\*\*: Filtered PaymentHistory data; exported file.   
  
#### 1.27 Manage Order Status History   
\*\*Function ID\*\*: FR-27   
\*\*Description\*\*: Enables administrators to view/export historical status changes for orders.   
\*\*Input\*\*: Filters (e.g., OrderID, date range); export format.   
\*\*Output\*\*: Filtered OrderStatusHistory data; exported file.

# External Description

### Chapter 2: External Interfaces  
  
#### 2.1 User Interface  
  
The system provides a graphical user interface (GUI) through which users interact with the system. This interface includes:  
  
1. \*\*Web-based Graphical User Interface (Web GUI):\*\*  
 - \*\*Description:\*\* The Web GUI is the primary interface for users to interact with the system. It includes forms for input, buttons for actions, and screens to display data.  
 - \*\*Interaction Method:\*\* Users interact with the Web GUI using standard web browsers. The interface includes input forms, navigation menus, and display screens.  
 - \*\*Inputs:\*\* User inputs such as customer details, product information, payment details, and search queries.  
 - \*\*Outputs:\*\* Display of customer profiles, product details, order summaries, payment confirmations, and other relevant information.  
  
2. \*\*Mobile Application Interface:\*\*  
 - \*\*Description:\*\* The system also provides a mobile application interface for users to access the system on mobile devices.  
 - \*\*Interaction Method:\*\* Users interact with the mobile application using touch inputs, similar to the Web GUI but optimized for mobile devices.  
 - \*\*Inputs:\*\* Same as the Web GUI, but optimized for mobile input methods.  
 - \*\*Outputs:\*\* Same as the Web GUI, but optimized for mobile display.  
  
#### 2.2 Hardware Interface  
  
The system interacts with external hardware devices primarily through standard communication protocols. The key hardware interfaces include:  
  
1. \*\*Payment Terminal:\*\*  
 - \*\*Description:\*\* The system integrates with payment terminals to process physical payments in-store.  
 - \*\*Interaction Method:\*\* Communication via industry-standard payment processing protocols (e.g., ISO 8583).  
 - \*\*Inputs:\*\* Payment card details, PIN entries, and transaction amounts.  
 - \*\*Outputs:\*\* Payment confirmation, transaction receipts, and error messages.  
  
#### 2.3 Software Interface  
  
The system interacts with several external software systems to provide its functionality. These interfaces include:  
  
1. \*\*Database Management System (DBMS):\*\*  
 - \*\*Description:\*\* The system uses a relational database management system to store and retrieve data.  
 - \*\*Interaction Method:\*\* SQL queries and transactions.  
 - \*\*Inputs:\*\* Data to be stored (e.g., customer details, product information, orders).  
 - \*\*Outputs:\*\* Retrieved data (e.g., customer profiles, product listings, order history).  
  
2. \*\*Third-Party Payment Gateway:\*\*  
 - \*\*Description:\*\* The system integrates with a third-party payment gateway to process online payments.  
 - \*\*Interaction Method:\*\* RESTful API calls over HTTPS.  
 - \*\*Inputs:\*\* Payment details (e.g., credit card information, transaction amount).  
 - \*\*Outputs:\*\* Payment confirmation, transaction ID, and error codes.  
  
3. \*\*Plugin Management System:\*\*  
 - \*\*Description:\*\* The system allows administrators to install, update, and uninstall plugins.  
 - \*\*Interaction Method:\*\* Plugin API provided by the system.  
 - \*\*Inputs:\*\* Plugin files or repository references.  
 - \*\*Outputs:\*\* Plugin installation, update, or removal confirmation.  
  
4. \*\*Email Service Provider:\*\*  
 - \*\*Description:\*\* The system uses an email service provider to send notifications and confirmations.  
 - \*\*Interaction Method:\*\* SMTP or RESTful API.  
 - \*\*Inputs:\*\* Email content, recipient addresses, and subject lines.  
 - \*\*Outputs:\*\* Email delivery confirmations or error messages.  
  
#### 2.4 Communication Interface  
  
The system communicates with external systems and users through various communication channels. These include:  
  
1. \*\*Web Browsing:\*\*  
 - \*\*Description:\*\* Users access the system through web browsers using HTTP/HTTPS protocols.  
 - \*\*Interaction Method:\*\* Standard web browsing over TCP/IP.  
 - \*\*Inputs:\*\* User requests (e.g., URL navigation, form submissions).  
 - \*\*Outputs:\*\* Web pages, responses to form submissions, and AJAX responses.  
  
2. \*\*Email Notifications:\*\*  
 - \*\*Description:\*\* The system sends email notifications for order confirmations, payment receipts, and other events.  
 - \*\*Interaction Method:\*\* SMTP or email API.  
 - \*\*Inputs:\*\* Email content and recipient information.  
 - \*\*Outputs:\*\* Email delivery status.  
  
3. \*\*SMS Notifications:\*\*  
 - \*\*Description:\*\* The system sends SMS notifications for certain events (e.g., order status updates).  
 - \*\*Interaction Method:\*\* SMS gateway API.  
 - \*\*Inputs:\*\* Message content and recipient phone numbers.  
 - \*\*Outputs:\*\* SMS delivery status.  
  
### Summary  
  
This chapter has outlined the external interfaces of the system, including user interfaces, hardware interfaces, software interfaces, and communication interfaces. Each interface has been described in terms of its interaction method, inputs, and outputs. These interfaces are critical to the system's functionality and must be implemented according to the specifications provided.

# Use Case

Use Case Name: Register Customer   
Use Case ID: UC-01   
Actors: Customer, System   
Preconditions:   
- The system is operational and accessible.   
- The customer is not yet registered in the system.   
  
Postconditions:   
- The customer is successfully registered in the system.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the registration page of the system.   
2. The system displays a registration form with fields for customer details (e.g., name, email, password).   
3. The customer fills in the required information and submits the form.   
4. The system validates the input data (e.g., checks for valid email format and password strength).   
5. The system creates a new customer account in the Customer data entity.   
6. The system sends a confirmation message to the customer.   
7. The use case ends with the customer registered and informed.   
  
Alternative Flow:   
1. If the customer enters invalid or incomplete information (e.g., duplicate email, missing required field), the system displays an error message.   
2. The customer corrects the information and resubmits the form.   
3. If the customer cancels the registration process, the system returns to the home page.  
  
Use Case Name: Update Customer Information   
Use Case ID: UC-02   
Actors: Customer, System   
Preconditions:   
- The system is operational and accessible.   
- The customer is already registered in the system.   
- The customer is logged in.   
  
Postconditions:   
- The customer's information is updated in the Customer data entity.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the account settings or profile page.   
2. The system displays the current customer information in an editable form.   
3. The customer modifies the necessary details (e.g., address, phone number, email).   
4. The customer submits the updated form.   
5. The system validates the input data (e.g., checks for valid email format).   
6. The system updates the customer's information in the Customer data entity.   
7. The system displays a confirmation message indicating the successful update.   
8. The use case ends with the customer's information updated.   
  
Alternative Flow:   
1. If the customer enters invalid or incomplete information (e.g., duplicate email, missing required field), the system displays an error message.   
2. The customer corrects the information and resubmits the form.   
3. If the customer cancels the update process, the system returns to the home page or previous screen.  
  
Use Case Name: View Customer Profile   
Use Case ID: UC-03   
Actors: Customer, System   
Preconditions:   
- The system is operational and accessible.   
- The customer is already registered in the system.   
- The customer is logged in.   
  
Postconditions:   
- The customer's profile information is displayed on the screen.   
- The system remains in a stable state.   
  
Main Flow:   
1. The customer navigates to the profile page from the account menu.   
2. The system retrieves the customer's information from the Customer data entity.   
3. The system displays the customer's profile details (e.g., name, email, address, order history).   
4. The use case ends with the customer viewing their profile.   
  
Alternative Flow:   
1. If the system fails to retrieve the customer's information, it displays an error message.   
2. The customer may choose to refresh the page or contact support for assistance.   
3. If the customer navigates away from the profile page, the system returns to the home page or previous screen.  
  
Use Case Name: Add Product   
Use Case ID: UC-04   
Actors: Administrator, System, Product, ShoppingCart, Inventory, Plugin   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The product to be added is not already in the system.   
- The inventory has sufficient capacity to add the new product.   
  
Postconditions:   
- The product is successfully added to the system.   
- The product information is stored in the Product data entity.   
- Inventory records are updated with the new product.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The system displays a form for adding new product information (e.g., product name, description, price, category, inventory quantity).   
3. The administrator fills in the required product details and selects a category from the Product Category data entity.   
4. The administrator submits the form.   
5. The system validates the input data (e.g., checks for valid price format, available category).   
6. The system creates a new product entry in the Product data entity.   
7. The system updates the Inventory data entity with the new product and its quantity.   
8. The system displays a confirmation message indicating the product was added successfully.   
9. The use case ends with the product added and inventory updated.   
  
Alternative Flow:   
1. If the administrator enters invalid or incomplete information (e.g., duplicate product ID, missing required fields), the system displays an error message.   
2. The administrator corrects the information and resubmits the form.   
3. If the inventory is at capacity or the selected category does not exist, the system displays an error message.   
4. The administrator may cancel the add product process, and the system returns to the product management page or home screen.  
  
Use Case Name: Update Product Details   
Use Case ID: UC-05   
Actors: Administrator, System, Product, Product Category, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The product to be updated already exists in the system.   
  
Postconditions:   
- The product details are successfully updated in the Product data entity.   
- The Inventory data entity is updated if necessary (e.g., stock quantity).   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The system displays a list of existing products and allows the administrator to select one for editing.   
3. The administrator selects the product and opens the product details form.   
4. The system retrieves and displays the product information (e.g., name, price, description, category, stock quantity).   
5. The administrator modifies the necessary product details (e.g., price, description, category, stock quantity).   
6. The administrator submits the updated form.   
7. The system validates the input data (e.g., checks for valid price format, existing category).   
8. The system updates the Product data entity with the new information.   
9. If the stock quantity was modified, the system updates the Inventory data entity accordingly.   
10. The system displays a confirmation message indicating the product details were updated successfully.   
11. The use case ends with the product updated and the administrator informed.   
  
Alternative Flow:   
1. If the administrator enters invalid or incomplete information (e.g., negative stock quantity, invalid price), the system displays an error message.   
2. The administrator corrects the information and resubmits the form.   
3. If the selected category does not exist, the system displays an error message.   
4. The administrator may cancel the update process, and the system returns to the product management page or home screen.  
  
Use Case Name: View Product Details   
Use Case ID: UC-06   
Actors: Customer, System, Product   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is browsing the product catalog or has searched for a specific product.   
- The product exists in the Product data entity.   
  
Postconditions:   
- The product details are displayed to the customer.   
- The system remains in a stable state.   
  
Main Flow:   
1. The customer selects a product from the catalog or search results.   
2. The system retrieves the product information from the Product data entity.   
3. The system displays the product details (e.g., name, description, price, image, category).   
4. The customer may add the product to the Shopping Cart or continue browsing.   
5. The use case ends with the product details viewed by the customer.   
  
Alternative Flow:   
1. If the product does not exist or the system fails to retrieve its details, an error message is displayed.   
2. The customer may choose to return to the catalog or contact support for assistance.   
3. If the customer navigates away from the product details page, the system returns to the previous screen.  
  
Use Case Name: Remove Product   
Use Case ID: UC-07   
Actors: Administrator, System, Product, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The product to be removed exists in the Product data entity.   
- The product is not currently in any active Order or Shopping Cart.   
  
Postconditions:   
- The product is removed from the Product data entity.   
- The Inventory data entity is updated to reflect the removal.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the product management section of the system.   
2. The system displays a list of existing products with options to remove them.   
3. The administrator selects the product to be removed.   
4. The system verifies that the product is not associated with any active orders or shopping carts.   
5. The administrator confirms the removal of the product.   
6. The system deletes the product entry from the Product data entity.   
7. The system updates the Inventory data entity to remove the product's stock information.   
8. The system displays a confirmation message indicating the product was successfully removed.   
9. The use case ends with the product removed and the administrator informed.   
  
Alternative Flow:   
1. If the product is associated with an active order or shopping cart, the system displays an error message.   
2. The administrator may choose to proceed if the product is no longer in use.   
3. If the administrator cancels the removal process, the system returns to the product management page or home screen.  
  
Use Case Name: Place Order   
Use Case ID: UC-08   
Actors: Customer, System, Shopping Cart, Product, Inventory, Payment   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has added at least one product to the Shopping Cart.   
- The inventory contains sufficient stock for the selected products.   
- The payment method is available and configured.   
  
Postconditions:   
- The order is successfully created in the Order data entity.   
- The inventory quantities are updated to reflect the purchased items.   
- The Shopping Cart is cleared.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the Shopping Cart and reviews the selected items.   
2. The customer proceeds to checkout.   
3. The system displays the order summary and prompts for shipping details and payment method.   
4. The customer confirms the order details and selects a payment method.   
5. The system validates the inventory for each product in the cart.   
6. The system processes the payment through the Payment data entity.   
7. If the payment is successful, the system creates a new order in the Order data entity.   
8. The system updates the Inventory data entity by reducing the stock of each ordered product.   
9. The system clears the Shopping Cart.   
10. The system displays a confirmation message to the customer with the order details.   
11. The use case ends with the order placed and the customer informed.   
  
Alternative Flow:   
1. If the inventory does not have sufficient stock for one or more items, the system displays an error message.   
2. The customer may adjust the cart or remove the out-of-stock item.   
3. If the payment fails, the system displays an error message and prompts the customer to try again or select another method.   
4. The customer may cancel the order process, and the system returns to the Shopping Cart or home page.  
  
Use Case Name: View Order Details   
Use Case ID: UC-09   
Actors: Customer, System, Order, Product, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is already registered and logged in.   
- The customer has at least one order in the Order data entity.   
  
Postconditions:   
- The order details are displayed to the customer.   
- The system remains in a stable state.   
  
Main Flow:   
1. The customer navigates to the "My Orders" section from the account menu.   
2. The system retrieves a list of the customer's orders from the Order data entity.   
3. The customer selects a specific order to view its details.   
4. The system retrieves and displays the order information (e.g., order date, total amount, product list, shipping status).   
5. The system also shows the details of each product in the order from the Product data entity.   
6. The use case ends with the customer viewing the selected order's information.   
  
Alternative Flow:   
1. If the system fails to retrieve the order details, it displays an error message.   
2. The customer may choose to refresh the page or contact support for assistance.   
3. If the customer navigates away from the order details page, the system returns to the "My Orders" list or home screen.  
  
Use Case Name: Update Order Status   
Use Case ID: UC-10   
Actors: Administrator, System, Order   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The order to be updated exists in the Order data entity.   
- The order is not yet finalized or completed.   
  
Postconditions:   
- The order status is updated in the Order data entity.   
- The customer is notified of the status change (if applicable).   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the order management section of the system.   
2. The system displays a list of orders with their current statuses.   
3. The administrator selects the specific order to update.   
4. The system retrieves and displays the order details, including the current status.   
5. The administrator chooses a new status (e.g., "Processing," "Shipped," "Cancelled").   
6. The administrator confirms the status update.   
7. The system updates the Order data entity with the new status.   
8. If applicable, the system sends a notification to the customer about the status change.   
9. The system displays a confirmation message to the administrator.   
10. The use case ends with the order status updated.   
  
Alternative Flow:   
1. If the administrator selects an invalid or unsupported status, the system displays an error message.   
2. The administrator can cancel the update process, and the system returns to the order management page or home screen.  
  
Use Case Name: Cancel Order   
Use Case ID: UC-11   
Actors: Customer, System, Order, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is already registered and logged in.   
- The customer has an active order in the Order data entity.   
- The order is not yet completed, shipped, or finalized.   
  
Postconditions:   
- The order is marked as "Cancelled" in the Order data entity.   
- The inventory quantities for the ordered products are restored.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the "My Orders" section from the account menu.   
2. The system displays a list of the customer's orders, including those that are active or pending.   
3. The customer selects an order to cancel.   
4. The system verifies the order status and confirms it is eligible for cancellation.   
5. The customer confirms the cancellation request.   
6. The system updates the Order data entity to mark the order as "Cancelled."   
7. The system restores the inventory stock for the products in the cancelled order.   
8. The system displays a confirmation message to the customer.   
9. The use case ends with the order cancelled and the customer informed.   
  
Alternative Flow:   
1. If the selected order is not eligible for cancellation (e.g., already shipped or completed), the system displays an error message.   
2. If the customer cancels the cancellation process, the system returns to the "My Orders" list or home screen.   
3. If the system fails to update the order status or restore inventory, it displays an error message.   
4. The customer may choose to refresh the page or contact support for assistance.  
  
Use Case Name: Add to Shopping Cart   
Use Case ID: UC-12   
Actors: Customer, System, Product, Shopping Cart, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The product to be added exists in the Product data entity.   
- The inventory has sufficient stock for the product.   
  
Postconditions:   
- The product is added to the customer's Shopping Cart.   
- The inventory quantity for the product is updated to reflect the addition.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer views a product on the product details page.   
2. The customer selects the quantity of the product they wish to add to the cart.   
3. The customer clicks the "Add to Cart" button.   
4. The system checks the inventory to confirm sufficient stock is available.   
5. The system adds the product and its quantity to the Shopping Cart.   
6. The system updates the inventory by reducing the stock of the product.   
7. The system displays a confirmation message that the product was successfully added.   
8. The use case ends with the product added to the cart and inventory updated.   
  
Alternative Flow:   
1. If the inventory does not have sufficient stock for the selected quantity, the system displays an error message.   
2. The customer may choose to adjust the quantity or proceed without adding the product.   
3. If the customer cancels the "Add to Cart" action, the system returns to the product details page.  
  
Use Case Name: View Shopping Cart   
Use Case ID: UC-13   
Actors: Customer, System, Shopping Cart, Product, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has at least one product in their Shopping Cart.   
- The Shopping Cart data entity is associated with the customer.   
  
Postconditions:   
- The contents of the Shopping Cart are displayed to the customer.   
- The product and inventory information remains unchanged.   
- The use case ends with the customer viewing their cart.   
  
Main Flow:   
1. The customer navigates to the Shopping Cart section from the main menu or a product page.   
2. The system retrieves the customer's Shopping Cart from the Shopping Cart data entity.   
3. The system displays the list of products in the cart along with their quantities, prices, and total cost.   
4. The system also shows the estimated shipping cost and final total.   
5. The use case ends with the customer viewing their cart.   
  
Alternative Flow:   
1. If the Shopping Cart is empty, the system displays a message indicating that there are no items in the cart.   
2. The customer may choose to add products or return to the home page.   
3. If the system fails to retrieve the cart contents, it displays an error message.   
4. The customer may choose to refresh the page or contact support for assistance.   
5. If the customer navigates away from the cart page, the system returns to the previous screen.  
  
Use Case Name: Modify Shopping Cart   
Use Case ID: UC-14   
Actors: Customer, System, Product, Shopping Cart, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has at least one product in their Shopping Cart.   
- The Shopping Cart is associated with the customer.   
- The inventory contains the product(s) in the cart.   
  
Postconditions:   
- The Shopping Cart is updated to reflect the modifications made by the customer.   
- The inventory quantities are adjusted accordingly if items are removed or quantities changed.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the Shopping Cart section.   
2. The system displays the list of products currently in the cart with their quantities and prices.   
3. The customer selects a product to modify (e.g., change quantity, remove item).   
4. The system updates the Shopping Cart data entity with the new quantity or removes the product.   
5. If the product quantity is reduced, the system increases the inventory stock for that product.   
6. If the product is removed, the system fully restores its inventory stock.   
7. The system recalculates the total cost and updates the cart display.   
8. The system displays a confirmation message indicating the cart was successfully modified.   
9. The use case ends with the cart updated and the customer informed.   
  
Alternative Flow:   
1. If the customer attempts to increase the quantity beyond the available inventory, the system displays an error message.   
2. The customer adjusts the quantity to a valid value or removes the item.   
3. If the system fails to update the cart or inventory, it displays an error message.   
4. The customer may refresh the page or contact support for assistance.   
5. If the customer cancels the modification, the system reverts to the previous cart state or returns to the home page.  
  
Use Case Name: Process Payment   
Use Case ID: UC-15   
Actors: Customer, System, Payment, Order, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has a confirmed order in the Order data entity.   
- The payment method is valid and available for processing.   
  
Postconditions:   
- The payment is successfully processed and recorded in the Payment data entity.   
- The order status is updated to "Paid" in the Order data entity.   
- The inventory is updated to reflect the completed transaction.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer confirms the order at checkout.   
2. The system prompts the customer to select a payment method.   
3. The customer selects a payment option and enters the required payment details (e.g., card number, expiration date).   
4. The system verifies the selected payment method and processes the transaction via the Payment data entity.   
5. If the payment is approved, the system updates the order status to "Paid" in the Order data entity.   
6. The system reduces the inventory stock for the purchased items.   
7. The system displays a payment confirmation message to the customer.   
8. The use case ends with the payment processed and the customer informed.   
  
Alternative Flow:   
1. If the payment is declined or fails, the system displays an error message.   
2. The customer may try again with the same or different payment method.   
3. If the customer cancels the payment process, the system returns to the Shopping Cart or order summary page.   
4. If the system encounters an error in updating the order or inventory, it displays an error message and stops the process.   
5. The customer may refresh the page or contact support for assistance.  
  
Use Case Name: View Payment History   
Use Case ID: UC-16   
Actors: Customer, System, Payment   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is already registered and logged in.   
- The customer has at least one completed payment in the Payment data entity.   
  
Postconditions:   
- The customer's payment history is displayed on the screen.   
- The system remains in a stable state.   
  
Main Flow:   
1. The customer navigates to the "Payment History" section from the account menu.   
2. The system retrieves the customer's payment records from the Payment data entity.   
3. The system displays a list of past payments, including details such as payment date, amount, status, and associated order.   
4. The customer can view specific payment details by selecting a record.   
5. The system shows additional information such as transaction ID, payment method, and confirmation status.   
6. The use case ends with the customer viewing their payment history.   
  
Alternative Flow:   
1. If the customer has no payment history, the system displays a message indicating this.   
2. The customer may choose to return to the account menu or browse orders.   
3. If the system fails to retrieve payment records, it displays an error message.   
4. The customer may refresh the page or contact support for assistance.   
5. If the customer navigates away from the payment history page, the system returns to the account menu or home screen.  
  
Use Case Name: Update Inventory Levels   
Use Case ID: UC-17   
Actors: Administrator, System, Inventory, Product   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The product exists in the Product data entity.   
- The inventory contains records for the product.   
  
Postconditions:   
- The inventory levels for the selected product are updated in the Inventory data entity.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the inventory management section of the system.   
2. The system displays a list of products and their current inventory levels.   
3. The administrator selects a product to update its inventory level.   
4. The system retrieves the current inventory quantity for the product.   
5. The administrator enters the new inventory quantity.   
6. The administrator submits the updated inventory data.   
7. The system validates the input (e.g., ensures the quantity is a positive number).   
8. The system updates the Inventory data entity with the new quantity for the product.   
9. The system displays a confirmation message indicating the inventory level was successfully updated.   
10. The use case ends with the inventory level modified and the administrator informed.   
  
Alternative Flow:   
1. If the administrator enters an invalid quantity (e.g., negative or non-numeric), the system displays an error message.   
2. The administrator corrects the input and resubmits the update.   
3. If the system fails to update the inventory, it displays an error message.   
4. The administrator may choose to retry the update or cancel the process.   
5. If the administrator cancels the update, the system returns to the inventory management page or home screen.  
  
Use Case Name: View Inventory Levels   
Use Case ID: UC-18   
Actors: Administrator, System, Inventory, Product   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The inventory contains records for at least one product.   
  
Postconditions:   
- The current inventory levels for all products are displayed to the administrator.   
- The system remains in a stable state.   
  
Main Flow:   
1. The administrator navigates to the inventory management section of the system.   
2. The system retrieves the current inventory records from the Inventory data entity.   
3. The system displays a list of products along with their current stock levels.   
4. The administrator reviews the displayed inventory information.   
5. The use case ends with the administrator viewing the inventory levels.   
  
Alternative Flow:   
1. If the system fails to retrieve inventory data, it displays an error message.   
2. The administrator may choose to refresh the page or contact support for assistance.   
3. If the administrator navigates away from the inventory page, the system returns to the home screen or previous page.  
  
Use Case Name: Manage Users   
Use Case ID: UC-19   
Actors: Administrator, System, Customer   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Customer data entity exists in the system.   
  
Postconditions:   
- The user (customer) is either added, updated, or removed from the Customer data entity.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the user management section of the system.   
2. The system displays a list of existing customers along with options to add, edit, or remove a user.   
3. The administrator selects the appropriate action (add, update, or remove) for a user.   
4. If adding a user, the administrator fills in the required details (e.g., name, email, password, address).   
5. If updating a user, the administrator selects a customer and modifies their details.   
6. If removing a user, the administrator selects a customer and confirms the deletion.   
7. The system validates the input data (e.g., checks for valid email format, duplicate entries).   
8. The system updates the Customer data entity accordingly.   
9. The system displays a confirmation message to the administrator.   
10. The use case ends with the user successfully managed.   
  
Alternative Flow:   
1. If the administrator enters invalid or duplicate information during adding or updating, the system displays an error message.   
2. The administrator corrects the information and resubmits the request.   
3. If the selected user is part of an active order, the system displays a warning before deletion.   
4. If the administrator cancels the action, the system returns to the user management page or home screen.   
5. If the system fails to process the request, it displays an error message.   
6. The administrator may choose to refresh the page or contact support for assistance.  
  
Use Case Name: Manage Plugins   
Use Case ID: UC-20   
Actors: Administrator, System, Plugin   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Plugin data entity exists in the system.   
  
Postconditions:   
- The plugin is either added, activated, deactivated, or removed from the Plugin data entity.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The system displays a list of available plugins along with their current status (e.g., active, inactive).   
3. The administrator selects an action (e.g., add, activate, deactivate, remove) for a specific plugin.   
4. If adding a new plugin, the administrator uploads or selects the plugin from a repository.   
5. The system validates the plugin (e.g., checks compatibility, required dependencies).   
6. The system updates the Plugin data entity with the new or modified plugin status.   
7. The system displays a confirmation message to the administrator.   
8. The use case ends with the plugin successfully managed.   
  
Alternative Flow:   
1. If the plugin is incompatible or fails validation, the system displays an error message.   
2. The administrator may choose to correct the plugin or select another.   
3. If the administrator cancels the action, the system returns to the plugin management page or home screen.   
4. If the system fails to update the plugin status, it displays an error message.   
5. The administrator may retry the action or contact support for assistance.  
  
Use Case Name: Create Product Category   
Use Case ID: UC-21   
Actors: Administrator, System, Product Category   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Product Category data entity exists in the system.   
- The category to be created does not already exist.   
  
Postconditions:   
- The new product category is successfully created and stored in the Product Category data entity.   
- A confirmation message is displayed to the administrator.   
- The system remains in a stable state.   
  
Main Flow:   
1. The administrator navigates to the product category management section of the system.   
2. The system displays a form for creating a new product category (e.g., category name, description, parent category).   
3. The administrator fills in the required information for the new category.   
4. The administrator submits the form.   
5. The system validates the input data (e.g., checks for duplicate category names, valid description).   
6. The system creates a new entry in the Product Category data entity.   
7. The system displays a confirmation message indicating the category was successfully created.   
8. The use case ends with the new product category added to the system.   
  
Alternative Flow:   
1. If the administrator enters a duplicate category name or invalid data, the system displays an error message.   
2. The administrator corrects the information and resubmits the form.   
3. If the administrator cancels the creation process, the system returns to the product category management page or home screen.   
4. If the system fails to create the category, it displays an error message.   
5. The administrator may choose to refresh the page or contact support for assistance.  
  
Use Case Name: Update Product Category   
Use Case ID: UC-22   
Actors: Administrator, System, Product Category   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Product Category to be updated already exists in the Product Category data entity.   
  
Postconditions:   
- The product category is successfully updated in the Product Category data entity.   
- A confirmation message is displayed to the administrator.   
- The system remains in a stable state.   
  
Main Flow:   
1. The administrator navigates to the product category management section of the system.   
2. The system displays a list of existing product categories with options to edit them.   
3. The administrator selects a product category to update.   
4. The system retrieves and displays the current details of the selected product category (e.g., category name, description, parent category).   
5. The administrator modifies the category details as needed (e.g., name, description).   
6. The administrator submits the updated form.   
7. The system validates the input data (e.g., checks for duplicate category names, valid description format).   
8. The system updates the Product Category data entity with the new information.   
9. The system displays a confirmation message indicating the category was successfully updated.   
10. The use case ends with the product category updated and the administrator informed.   
  
Alternative Flow:   
1. If the administrator enters a duplicate category name or invalid data, the system displays an error message.   
2. The administrator corrects the information and resubmits the form.   
3. If the system fails to update the category, it displays an error message.   
4. The administrator may choose to retry the update or cancel the process.   
5. If the administrator cancels the update, the system returns to the product category management page or home screen.  
  
Use Case Name: Delete Product Category   
Use Case ID: UC-23   
Actors: Administrator, System, Product Category   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Product Category to be deleted already exists in the Product Category data entity.   
- The category is not associated with any active Product entries.   
  
Postconditions:   
- The product category is successfully deleted from the Product Category data entity.   
- A confirmation message is displayed to the administrator.   
- The system remains in a stable state.   
  
Main Flow:   
1. The administrator navigates to the product category management section of the system.   
2. The system displays a list of existing product categories with options to delete them.   
3. The administrator selects the product category to delete.   
4. The system verifies that the category is not currently associated with any active products.   
5. The administrator confirms the deletion of the category.   
6. The system deletes the selected category from the Product Category data entity.   
7. The system displays a confirmation message indicating the category was successfully deleted.   
8. The use case ends with the category removed and the administrator informed.   
  
Alternative Flow:   
1. If the category is associated with one or more active products, the system displays an error message.   
2. The administrator may choose to first remove or reassign the associated products before attempting deletion again.   
3. If the administrator cancels the deletion process, the system returns to the product category management page or home screen.   
4. If the system fails to delete the category, it displays an error message.   
5. The administrator may choose to retry the deletion or contact support for assistance.  
  
Use Case Name: Install Plugin   
Use Case ID: UC-24   
Actors: Administrator, System, Plugin   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Plugin data entity exists in the system.   
- The plugin to be installed is available (either uploaded or accessible via a repository).   
  
Postconditions:   
- The plugin is successfully installed and stored in the Plugin data entity.   
- The system recognizes and integrates the plugin functionality.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The system displays a list of available plugins and an option to install a new plugin.   
3. The administrator selects the "Install Plugin" option and uploads or selects the plugin.   
4. The system validates the plugin (e.g., checks file integrity, compatibility, and dependencies).   
5. The system installs the plugin and registers it in the Plugin data entity.   
6. The system updates the plugin status to "Installed" in the plugin management interface.   
7. The system displays a confirmation message to the administrator.   
8. The use case ends with the plugin installed and the administrator informed.   
  
Alternative Flow:   
1. If the plugin fails validation (e.g., incompatible version, missing dependencies), the system displays an error message.   
2. The administrator may choose to correct the issue or select a different plugin.   
3. If the administrator cancels the installation process, the system returns to the plugin management page or home screen.   
4. If the system fails to install the plugin, it displays an error message.   
5. The administrator may retry the installation or contact support for assistance.  
  
Use Case Name: Update Plugin   
Use Case ID: UC-25   
Actors: Administrator, System, Plugin   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The plugin to be updated already exists in the Plugin data entity.   
- The updated plugin is available (either uploaded or accessible via a repository).   
  
Postconditions:   
- The plugin is successfully updated in the Plugin data entity.   
- The system recognizes and integrates the updated plugin functionality.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The system displays a list of installed plugins along with options to update them.   
3. The administrator selects the plugin to be updated.   
4. The system retrieves the current plugin details and displays an option to upload the updated version.   
5. The administrator uploads or selects the updated plugin from a repository.   
6. The system validates the updated plugin (e.g., checks compatibility, file integrity, and dependencies).   
7. The system updates the Plugin data entity with the new plugin version.   
8. The system applies the updated plugin and modifies its status to "Updated."   
9. The system displays a confirmation message to the administrator.   
10. The use case ends with the plugin updated and the administrator informed.   
  
Alternative Flow:   
1. If the updated plugin fails validation (e.g., incompatible with the system, missing dependencies), the system displays an error message.   
2. The administrator may choose to correct the issue or select a different version of the plugin.   
3. If the administrator cancels the update process, the system returns to the plugin management page or home screen.   
4. If the system fails to update the plugin, it displays an error message.   
5. The administrator may retry the update or contact support for assistance.  
  
Use Case Name: Uninstall Plugin   
Use Case ID: UC-26   
Actors: Administrator, System, Plugin   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The plugin to be uninstalled is already installed in the system.   
- The plugin is not currently in use by any active Order, Product, or other system processes.   
  
Postconditions:   
- The plugin is successfully uninstalled from the system.   
- The Plugin data entity is updated to reflect the uninstallation.   
- A confirmation message is displayed to the administrator.   
  
Main Flow:   
1. The administrator navigates to the plugin management section of the system.   
2. The system displays a list of installed plugins with options to uninstall.   
3. The administrator selects the plugin to be uninstalled.   
4. The system verifies that the plugin is not being used in any active processes.   
5. The administrator confirms the uninstallation.   
6. The system uninstalls the plugin and removes it from the Plugin data entity.   
7. The system updates the plugin status to "Uninstalled" in the plugin management interface.   
8. The system displays a confirmation message to the administrator.   
9. The use case ends with the plugin uninstalled and the administrator informed.   
  
Alternative Flow:   
1. If the plugin is currently in use by an active process (e.g., tied to an Order or Product), the system displays an error message.   
2. The administrator may choose to first disable the plugin or ensure no dependencies are active.   
3. If the administrator cancels the uninstallation, the system returns to the plugin management page or home screen.   
4. If the system fails to uninstall the plugin, it displays an error message.   
5. The administrator may retry the uninstallation or contact support for assistance.  
  
Use Case Name: Manage Payment History   
Use Case ID: UC-27   
Actors: Administrator, System, Payment History, Payment, Order   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Payment History data entity exists in the system and contains at least one record.   
  
Postconditions:   
- The payment history records are either viewed, filtered, or exported by the administrator.   
- The system remains in a stable state.   
- A confirmation or output (e.g., exported file) is provided to the administrator.   
  
Main Flow:   
1. The administrator navigates to the "Manage Payment History" section in the system.   
2. The system retrieves all records from the Payment History data entity.   
3. The system displays the payment history records, including details such as transaction ID, customer ID, order ID, payment date, amount, and status.   
4. The administrator can apply filters (e.g., by date range, payment status, customer, or order).   
5. The system updates the displayed payment history based on the applied filters.   
6. The administrator can export the filtered or full payment history data in a supported format (e.g., CSV, Excel).   
7. The system generates and provides the exported file to the administrator.   
8. The system displays a confirmation message that the payment history was managed successfully.   
9. The use case ends with the administrator having accessed or exported the payment history.   
  
Alternative Flow:   
1. If no payment history records exist or the filters return no results, the system displays a message indicating this.   
2. The administrator may choose to clear the filters or return to the main management page.   
3. If the system fails to retrieve or export the payment history data, it displays an error message.   
4. The administrator may choose to refresh the page or contact support for assistance.   
5. If the administrator cancels the management process, the system returns to the home screen or previous page.  
  
Use Case Name: Manage Order Status History   
Use Case ID: UC-28   
Actors: Administrator, System, Order, OrderStatusHistory   
  
Preconditions:   
- The system is operational and accessible.   
- The administrator is logged in and has appropriate permissions.   
- The Order data entity contains at least one order with associated status changes.   
- The OrderStatusHistory data entity exists in the system and is linked to the relevant order.   
  
Postconditions:   
- The administrator can view, filter, or export the order status history records.   
- The system remains in a stable state.   
- A confirmation or output (e.g., exported file) is provided to the administrator.   
  
Main Flow:   
1. The administrator navigates to the "Order Status History" section in the system.   
2. The system retrieves all records from the OrderStatusHistory data entity.   
3. The system displays a list of orders along with their historical status changes, including timestamps and status details.   
4. The administrator selects a specific order to view its full status history.   
5. The system retrieves and displays detailed status transitions (e.g., "Created", "Processing", "Shipped", "Delivered") for the selected order.   
6. The administrator can apply filters (e.g., by date range, specific statuses, or order ID).   
7. The system updates the displayed status history based on the applied filters.   
8. The administrator can export the filtered or full status history data in a supported format (e.g., CSV, Excel).   
9. The system generates and provides the exported file to the administrator.   
10. The system displays a confirmation message that the order status history was managed successfully.   
11. The use case ends with the administrator having accessed or exported the order status history.   
  
Alternative Flow:   
1. If no status history records exist for the selected order or filters return no results, the system displays a message indicating this.   
2. The administrator may choose to clear the filters or return to the main order management page.   
3. If the system fails to retrieve or export the status history data, it displays an error message.   
4. The administrator may choose to refresh the page or contact support for assistance.   
5. If the administrator cancels the management process, the system returns to the home screen or previous page.  
  
Use Case Name: Manage Cart Product Item   
Use Case ID: UC-29   
Actors: Customer, System, ShoppingCart, CartProductItem, Product   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has at least one product in their Shopping Cart.   
- The CartProductItem data entity exists in the system.   
  
Postconditions:   
- The customer can view, modify, or remove specific items in the Shopping Cart through the CartProductItem data entity.   
- The inventory quantities are adjusted accordingly if items are removed or quantities changed.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the Shopping Cart section of the system.   
2. The system retrieves the customer’s Shopping Cart and displays the list of CartProductItems.   
3. The customer selects a specific CartProductItem for management (e.g., view details, change quantity, or remove item).   
4. The system retrieves and displays the details of the selected CartProductItem (e.g., product name, current quantity, price).   
5. If the customer modifies the quantity, the system validates the requested quantity against the available inventory.   
6. If the quantity is valid, the system updates the CartProductItem in the Shopping Cart data entity.   
7. The system adjusts the inventory to reflect any changes in quantity (e.g., increases stock if the quantity is reduced).   
8. If the customer chooses to remove the CartProductItem, the system deletes the item from the Shopping Cart and restores the inventory stock for the product.   
9. The system recalculates the total cost of the cart and updates the display.   
10. The system displays a confirmation message indicating the CartProductItem was successfully managed.   
11. The use case ends with the cart updated and the customer informed.   
  
Alternative Flow:   
1. If the customer requests a quantity greater than the available inventory, the system displays an error message.   
2. The customer may choose to adjust the quantity or remove the item.   
3. If the system fails to update or remove the CartProductItem, it displays an error message.   
4. The customer may choose to refresh the page or contact support for assistance.   
5. If the customer cancels the management action, the system returns to the Shopping Cart or home screen.  
  
Use Case Name: Manage Shopping Cart   
Use Case ID: UC-30   
Actors: Customer, System, Shopping Cart, Product, Inventory   
  
Preconditions:   
- The system is operational and accessible.   
- The customer is registered and logged in.   
- The customer has at least one product in their Shopping Cart.   
- The Shopping Cart data entity is associated with the customer.   
  
Postconditions:   
- The Shopping Cart is successfully managed (e.g., items added, removed, or quantities adjusted).   
- Inventory quantities are updated accordingly if items are removed or quantities changed.   
- A confirmation message is displayed to the customer.   
  
Main Flow:   
1. The customer navigates to the Shopping Cart section from the main menu or a product page.   
2. The system retrieves the customer's Shopping Cart from the Shopping Cart data entity.   
3. The system displays the list of CartProductItems, including product details, quantities, and prices.   
4. The customer selects an action to manage the cart (e.g., add another product, remove an item, or adjust quantity).   
5. The system processes the selected action (e.g., adds a new product, removes an item, or updates quantity).   
6. The system validates the inventory for any changes in quantity (e.g., ensures sufficient stock is available for additions).   
7. If inventory is updated (e.g., product is removed or quantity is adjusted), the system reflects the changes in the Inventory data entity.   
8. The system recalculates the total cost of the cart and updates the display.   
9. The system displays a confirmation message indicating the cart was successfully managed.   
10. The use case ends with the Shopping Cart updated and the customer informed.   
  
Alternative Flow:   
1. If the customer attempts to add a product with a quantity exceeding the available inventory, the system displays an error message.   
2. The customer may choose to reduce the quantity or cancel the addition.   
3. If the system fails to update the cart or inventory, it displays an error message.   
4. The customer may choose to refresh the page or contact support for assistance.   
5. If the customer cancels the management action, the system reverts to the previous cart state or returns to the home page.