项目文档

# Functional Requirement

Based on the provided use cases and data model, here are the functional requirements for Chapter 1:  
  
### 1.1 Customer Registration Function   
\*\*Function ID:\*\* FR-01   
\*\*Description:\*\* Enables new customers to create accounts by providing personal information. Validates input uniqueness and stores credentials securely.   
\*\*Input:\*\* Customer details (Name, Email, Password, PhoneNumber, Address)   
\*\*Output:\*\* New Customer record with CustomerID, encrypted password, RegistrationDate   
  
### 1.2 Customer Authentication Function   
\*\*Function ID:\*\* FR-02   
\*\*Description:\*\* Verifies customer credentials to establish authenticated sessions and track login activity.   
\*\*Input:\*\* Email, Password   
\*\*Output:\*\* Authenticated session token, updated LastLoginDate in Customer record   
  
### 1.3 Session Termination Function   
\*\*Function ID:\*\* FR-03   
\*\*Description:\*\* Terminates active customer sessions and invalidates authentication tokens.   
\*\*Input:\*\* Session token   
\*\*Output:\*\* Session invalidation confirmation, audit log entry   
  
### 1.4 Customer Profile Management Function   
\*\*Function ID:\*\* FR-04   
\*\*Description:\*\* Allows customers or administrators to update and view customer account information.   
\*\*Input:\*\* CustomerID, updated fields (e.g., Address, PhoneNumber)   
\*\*Output:\*\* Modified Customer record, change audit log   
  
### 1.5 Customer Account Removal Function   
\*\*Function ID:\*\* FR-05   
\*\*Description:\*\* Permits administrators to permanently delete customer accounts while anonymizing personal data.   
\*\*Input:\*\* CustomerID   
\*\*Output:\*\* Account deletion confirmation, anonymized records, disabled authentication tokens   
  
### 1.6 Purchase History Retrieval Function   
\*\*Function ID:\*\* FR-06   
\*\*Description:\*\* Provides customers and administrators with access to historical order data.   
\*\*Input:\*\* CustomerID (for administrators) or session token (for customers)   
\*\*Output:\*\* List of Order records with associated OrderItems   
  
### 1.7 Product Catalog Management Function   
\*\*Function ID:\*\* FR-07   
\*\*Description:\*\* Enables administrators to add, modify, or remove products from the store catalog.   
\*\*Input:\*\* Product details (Name, Description, Price, StockQuantity, Category)   
\*\*Output:\*\* Created/updated Product record or deletion confirmation with cart purges   
  
### 1.8 Inventory Adjustment Function   
\*\*Function ID:\*\* FR-08   
\*\*Description:\*\* Allows administrators to modify product stock levels with audit trails.   
\*\*Input:\*\* ProductID, adjustment quantity (±integer), reason   
\*\*Output:\*\* Updated StockQuantity, inventory change log entry   
  
### 1.9 Product Information Display Function   
\*\*Function ID:\*\* FR-09   
\*\*Description:\*\* Shows product details to users and administrators.   
\*\*Input:\*\* ProductID   
\*\*Output:\*\* Product attributes (Name, Description, Price, StockQuantity, ImageURL, availability status)   
  
### 1.10 Shopping Cart Modification Function   
\*\*Function ID:\*\* FR-10   
\*\*Description:\*\* Manages additions, quantity changes, and removals of items in customer carts.   
\*\*Input:\*\* ProductID, Quantity (or removal flag), session token   
\*\*Output:\*\* Updated ShoppingCartItem records, recalculated cart totals   
  
### 1.11 Cart Summary Display Function   
\*\*Function ID:\*\* FR-11   
\*\*Description:\*\* Preserves real-time view of cart contents with availability verification.   
\*\*Input:\*\* Session token   
\*\*Output:\*\* ShoppingCartItem details with subtotals, availability status, grand total   
  
### 1.12 Order Creation Function   
\*\*Function ID:\*\* FR-12   
\*\*Description:\*\* Converts shopping carts into orders with payment processing initiation.   
\*\*Input:\*\* Session token, shipping address, payment method   
\*\*Output:\*\* New Order record with OrderItems, inventory reservations, "Pending Payment" status   
  
### 1.13 Order Status Management Function   
\*\*Function ID:\*\* FR-13   
\*\*Description:\*\* Updates order progression states with business rule validation.   
\*\*Input:\*\* OrderID, new OrderStatus   
\*\*Output:\*\* Modified Order record, status change notifications, inventory adjustments   
  
### 1.14 Order Item Modification Function   
\*\*Function ID:\*\* FR-14   
\*\*Description:\*\* Allows administrators to alter order compositions before shipment.   
\*\*Input:\*\* OrderID, OrderItemID, new Quantity (or removal flag)   
\*\*Output:\*\* Updated OrderItem records, recalculated order totals   
  
### 1.15 Order Deletion Function   
\*\*Function ID:\*\* FR-15   
\*\*Description:\*\* Removes orders while enforcing status constraints and inventory restocking.   
\*\*Input:\*\* OrderID   
\*\*Output:\*\* Order deletion confirmation, restocked inventory   
  
### 1.16 Payment Method Management Function   
\*\*Function ID:\*\* FR-16   
\*\*Description:\*\* Handles secure storage and maintenance of customer payment instruments.   
\*\*Input:\*\* Payment details (PaymentMethod, CardNumber, ExpiryDate, CVV)   
\*\*Output:\*\* Tokenized Payment record, storage confirmation   
  
### 1.17 Payment Processing Function   
\*\*Function ID:\*\* FR-17   
\*\*Description:\*\* Executes financial transactions through payment gateways.   
\*\*Input:\*\* OrderID, PaymentMethodID, amount   
\*\*Output:\*\* Payment transaction record, "Paid" order status, confirmation emails   
  
### 1.18 Administrator Account Management Function   
\*\*Function ID:\*\* FR-18   
\*\*Description:\*\* Creates and maintains administrator profiles with privilege enforcement.   
\*\*Input:\*\* Administrator details (Username, Password, Role)   
\*\*Output:\*\* New/updated Administrator record with privilege verifications   
  
### 1.19 System Configuration Function   
\*\*Function ID:\*\* FR-19   
\*\*Description:\*\* Modifies global system parameters with validation and service reloads.   
\*\*Input:\*\* SettingID, new Value   
\*\*Output:\*\* Updated SystemSettings record, configuration reload confirmation   
  
### Logic Validation:  
- All inputs/outputs reference defined entities/attributes from the data model  
- Transformation feasibility ensured (e.g., cart items → orders, credentials → sessions)  
- Complete coverage of 34 use cases through atomic functions  
- No undefined terms - all references match provided E-R model  
- Functions derived exclusively from specified use cases without invention

# External Description

### Chapter 2: External Interfaces  
  
This chapter describes the external interfaces of the system, including user interfaces, hardware interfaces, software interfaces, and communication interfaces. Each interface is defined and described based on the functional requirements provided.  
  
---  
  
#### 2.1 User Interface Output  
  
The system provides several user interfaces for customers and administrators to interact with the system. Below are the key user interfaces identified:  
  
1. \*\*Customer Registration Page\*\*   
 - \*\*Description:\*\* A web page where new customers can input their personal information to create an account.   
 - \*\*Inputs:\*\* Name, Email, Password, PhoneNumber, Address.   
 - \*\*Outputs:\*\* Confirmation message, redirect to login page.  
  
2. \*\*Customer Login Page\*\*   
 - \*\*Description:\*\* A web page where customers can authenticate their accounts using their email and password.   
 - \*\*Inputs:\*\* Email, Password.   
 - \*\*Outputs:\*\* Session token, redirect to the dashboard.  
  
3. \*\*Customer Profile Page\*\*   
 - \*\*Description:\*\* A web page where customers can view and update their account information.   
 - \*\*Inputs:\*\* CustomerID, fields to update (e.g., Address, PhoneNumber).   
 - \*\*Outputs:\*\* Updated profile information, change audit log.  
  
4. \*\*Shopping Cart Page\*\*   
 - \*\*Description:\*\* A web page where customers can view and manage items in their shopping cart.   
 - \*\*Inputs:\*\* ProductID, Quantity (or removal flag), session token.   
 - \*\*Outputs:\*\* Updated cart contents, subtotals, and availability status.  
  
5. \*\*Checkout Page\*\*   
 - \*\*Description:\*\* A web page where customers can review their order and complete the payment process.   
 - \*\*Inputs:\*\* Session token, shipping address, payment method.   
 - \*\*Outputs:\*\* Order confirmation, redirect to order tracking page.  
  
6. \*\*Order History Page\*\*   
 - \*\*Description:\*\* A web page where customers can view their purchase history.   
 - \*\*Inputs:\*\* CustomerID or session token.   
 - \*\*Outputs:\*\* List of Order records with associated OrderItems.  
  
7. \*\*Administrator Dashboard\*\*   
 - \*\*Description:\*\* A web page where administrators can manage products, customers, and system settings.   
 - \*\*Inputs:\*\* Administrator credentials, product details, or system settings.   
 - \*\*Outputs:\*\* Updated product records, customer records, or system configuration confirmation.  
  
8. \*\*Product Catalog Page\*\*   
 - \*\*Description:\*\* A web page where customers and administrators can view product details.   
 - \*\*Inputs:\*\* ProductID.   
 - \*\*Outputs:\*\* Product attributes (Name, Description, Price, StockQuantity, ImageURL, availability status).  
  
9. \*\*Payment Method Management Page\*\*   
 - \*\*Description:\*\* A web page where customers can add or update their payment methods.   
 - \*\*Inputs:\*\* Payment details (PaymentMethod, CardNumber, ExpiryDate, CVV).   
 - \*\*Outputs:\*\* Tokenized Payment record, storage confirmation.  
  
10. \*\*Email Confirmation/Notification\*\*   
 - \*\*Description:\*\* An email sent to customers and administrators for account verification, order confirmation, or password reset.   
 - \*\*Inputs:\*\* Recipient email, event trigger (e.g., registration, order completion).   
 - \*\*Outputs:\*\* Email message with confirmation link or notification details.  
  
---  
  
#### 2.2 Hardware Interface Output  
  
The system does not directly interact with external hardware devices. However, the following hardware interfaces may be involved:  
  
1. \*\*Payment Terminal\*\*   
 - \*\*Description:\*\* A hardware device used for card-based payments.   
 - \*\*Inputs:\*\* Card swipe or insertion.   
 - \*\*Outputs:\*\* Card information for payment processing.  
  
2. \*\*Barcode/QR Code Scanner\*\*   
 - \*\*Description:\*\* A hardware device used for scanning product barcodes or QR codes.   
 - \*\*Inputs:\*\* Barcode/QR code data.   
 - \*\*Outputs:\*\* Product details for inventory management.  
  
---  
  
#### 2.3 Software Interface Output  
  
The system interacts with several software components and external services. Below are the key software interfaces identified:  
  
1. \*\*Customer Database\*\*   
 - \*\*Description:\*\* A database used to store customer information, including personal details and account credentials.   
 - \*\*Inputs:\*\* Customer details (Name, Email, Password, PhoneNumber, Address).   
 - \*\*Outputs:\*\* Customer record with CustomerID, encrypted password, and RegistrationDate.  
  
2. \*\*Product Database\*\*   
 - \*\*Description:\*\* A database used to store product information, including inventory details.   
 - \*\*Inputs:\*\* Product details (Name, Description, Price, StockQuantity, Category).   
 - \*\*Outputs:\*\* Product record with updated StockQuantity and inventory change log entry.  
  
3. \*\*Order Database\*\*   
 - \*\*Description:\*\* A database used to store order information, including order status and items.   
 - \*\*Inputs:\*\* OrderID, new OrderStatus, payment method details.   
 - \*\*Outputs:\*\* Updated Order record with status change notifications and inventory adjustments.  
  
4. \*\*Payment Gateway\*\*   
 - \*\*Description:\*\* An external service used to process payments securely.   
 - \*\*Inputs:\*\* OrderID, PaymentMethodID, amount.   
 - \*\*Outputs:\*\* Payment transaction record, "Paid" order status, confirmation emails.  
  
5. \*\*System Configuration Database\*\*   
 - \*\*Description:\*\* A database used to store global system parameters.   
 - \*\*Inputs:\*\* SettingID, new Value.   
 - \*\*Outputs:\*\* Updated SystemSettings record, configuration reload confirmation.  
  
---  
  
#### 2.4 Communication Interface Output  
  
The system communicates with external entities through various means. Below are the key communication interfaces identified:  
  
1. \*\*Email Notifications\*\*   
 - \*\*Description:\*\* Communication via email for account verification, order confirmation, and password reset.   
 - \*\*Inputs:\*\* Recipient email, event trigger (e.g., registration, order completion).   
 - \*\*Outputs:\*\* Email message with confirmation link or notification details.  
  
2. \*\*SMS Notifications\*\*   
 - \*\*Description:\*\* Communication via SMS for order status updates and password reset.   
 - \*\*Inputs:\*\* Recipient phone number, event trigger (e.g., order status change).   
 - \*\*Outputs:\*\* SMS message with status update or reset link.  
  
3. \*\*API Interfaces\*\*   
 - \*\*Description:\*\* Communication with external systems via APIs for payment processing, inventory management, or customer authentication.   
 - \*\*Inputs:\*\* API request parameters (e.g., ProductID, adjustment quantity).   
 - \*\*Outputs:\*\* API response with confirmation or error messages.  
  
---  
  
### Summary  
  
The system's external interfaces are categorized into user interfaces, hardware interfaces, software interfaces, and communication interfaces. Each interface is defined with its role, inputs, and outputs to ensure clear interaction with external entities. This section ensures that all referenced external data sources in the functional requirements are covered and clearly defined.

# Use Case

### Use Case Description   
\*\*Use Case Name:\*\* Manage Customer   
\*\*Use Case ID:\*\* UC-01   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is logged into the system.   
2. Customer data repository is accessible.   
3. Administrator has sufficient privileges to modify customer records.   
  
\*\*Postconditions:\*\*   
1. Customer data is created, updated, or deactivated in the system.   
2. Changes are audited and persisted in the database.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Customer" function.   
2. System displays search interface for customer records.   
3. Administrator enters customer ID/name and initiates search.   
4. System retrieves and displays customer details (e.g., contact info, order history).   
5. Administrator edits required fields (e.g., address, email) and confirms changes.   
6. System validates input and updates customer record.   
7. System displays "Update successful" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: Create New Customer\*   
1. At step 2, Administrator selects "Create New" instead of searching.   
2. System displays empty customer registration form.   
3. Administrator inputs mandatory details (name, email, contact) and submits.   
4. System generates new customer ID and saves record.   
  
\*B: Deactivate Customer\*   
1. At step 4, Administrator selects "Deactivate Account" option.   
2. System prompts confirmation.   
3. Administrator confirms deactivation.   
4. System flags customer as inactive and disables login access.   
  
\*C: Invalid Data Entry\*   
1. At step 6 (Main Flow) or step 3 (Alternative Flow A), validation fails.   
2. System highlights erroneous fields with error messages.   
3. Administrator corrects data and resubmits.   
  
---  
  
> 🔍 \*Note: Use case focuses on core CRUD operations. Exception flows (e.g., system errors, permission denials) omitted per length constraints. Consistent with typical e-commerce system entities provided.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Register Customer   
\*\*Use Case ID:\*\* UC-02   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is not logged into the system.   
2. Registration functionality is available and accessible.   
3. Customer data repository is operational.   
  
\*\*Postconditions:\*\*   
1. New customer record is created with unique ID.   
2. Customer credentials are stored securely.   
3. System sends registration confirmation (email/SMS).   
  
\*\*Main Flow:\*\*   
1. Customer selects "Register" option.   
2. System displays registration form with mandatory fields (name, email, password).   
3. Customer enters valid information and submits form.   
4. System verifies data uniqueness (e.g., non-duplicate email).   
5. System generates customer ID and stores new record.   
6. System sends confirmation notification to customer.   
7. System redirects to login page with success message.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Input Data\*   
1. At steps 3-4, validation fails (e.g., invalid email format).   
2. System highlights errors without clearing form.   
3. Customer corrects data and resubmits.   
  
\*B: Duplicate Account\*   
1. At step 4, system detects existing email.   
2. System displays "Email already registered" alert.   
3. Customer uses password recovery or new email.   
  
\*C: Registration Abandonment\*   
1. At any step before submission, customer closes window.   
2. System discards partial data without persistence.   
  
---  
  
> 🔍 \*Note: Core flows focus on self-registration by end-user. Excluded: Payment integration, third-party logins, or admin approvals per scope constraints. Aligns with entity model including Customer and authentication components.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Login Customer   
\*\*Use Case ID:\*\* UC-03   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer has completed registration (valid account exists).   
2. Login functionality is available and accessible.   
3. Authentication service is operational.   
  
\*\*Postconditions:\*\*   
1. Customer is authenticated and granted system access.   
2. Customer session is established with appropriate permissions.   
3. Login event is recorded in audit logs.   
  
\*\*Main Flow:\*\*   
1. Customer navigates to login page.   
2. System displays login form with email and password fields.   
3. Customer enters registered credentials and submits.   
4. System verifies credentials against stored records.   
5. System creates authenticated session and redirects to customer dashboard.   
6. System displays personalized welcome message.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Credentials\*   
1. At step 4, credential verification fails.   
2. System displays "Invalid email or password" error.   
3. System retains email field but clears password.   
4. Customer re-attempts login or selects password recovery.   
  
\*B: Account Locked\*   
1. At step 4, system detects multiple failed login attempts.   
2. System temporarily locks account and displays lockout message.   
3. Customer must wait for cooldown period or contact support.   
  
\*C: Inactive Account\*   
1. At step 4, system detects deactivated account status.   
2. System displays "Account disabled" alert with reactivation instructions.   
3. Customer follows account recovery process or contacts administrator.   
  
---   
> 🔍 \*Note: Focuses on core authentication flow. Excludes: Social login integrations, multi-factor authentication, and session timeout handling per scope limitations. Aligns with Customer entity and security requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Logout Customer   
\*\*Use Case ID:\*\* UC-04   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is logged into the system with an active session.   
2. Logout functionality is available and operational.   
3. Session management service is accessible.   
  
\*\*Postconditions:\*\*   
1. Customer session is terminated and authentication token invalidated.   
2. System records logout event in audit logs.   
3. Customer is redirected to login page or landing page.   
  
\*\*Main Flow:\*\*   
1. Customer selects "Logout" option from navigation menu.   
2. System confirms logout intent via pop-up dialog.   
3. Customer confirms logout action.   
4. System terminates active session and invalidates session token.   
5. System redirects customer to login page with "Logout successful" message.   
  
\*\*Alternative Flow:\*\*   
\*A: Session Already Expired\*   
1. At step 2, system detects inactive session (timeout).   
2. System automatically invalidates residual session token.   
3. System redirects to login page with "Session expired" notification.   
  
\*B: Aborted Logout\*   
1. At step 3, customer cancels logout confirmation.   
2. System retains active session and returns to previous interface.   
3. Session timer resets for continued activity.   
  
---  
  
> 🔍 \*Note: Focuses on explicit and implicit session termination. Excludes: Concurrent device sessions, forced logouts by administrators, or token refresh mechanisms per scope constraints. Aligns with Customer entity and security protocols.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Update Customer Information   
\*\*Use Case ID:\*\* UC-05   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Customer record exists in the database.   
3. Data repository is accessible and operational.   
4. Actor has required permissions (Customer can edit own profile; Administrator can edit any profile).   
  
\*\*Postconditions:\*\*   
1. Customer record is updated in the database.   
2. Change audit log is generated.   
3. Confirmation notification is displayed to the actor.   
  
\*\*Main Flow:\*\*   
1. Actor selects "Customer Profile" or "Manage Customers" menu.   
2. System displays customer search interface (Administrator) or current profile (Customer).   
3. Actor locates target customer record (Administrator enters search criteria; Customer views own profile).   
4. Actor initiates edit mode and modifies fields (e.g., address, contact number).   
5. System validates updated data format and constraints.   
6. Actor confirms changes.   
7. System persists updates and displays "Information updated successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Search Failure (Administrator)\*   
1. At step 3, no matching customer found.   
2. System displays "No records found" and retains search criteria.   
3. Administrator refines search or cancels operation.   
  
\*B: Validation Failure\*   
1. At step 5, data violates rules (e.g., invalid phone format).   
2. System highlights errors without clearing inputs.   
3. Actor corrects data and resubmits.   
  
\*C: Unauthorized Modification\*   
1. At step 4, Administrator attempts restricted changes (e.g., system-generated ID).   
2. System disables immutable fields and displays "Field not editable" warning.   
  
---  
  
> 🔍 \*Note: Core flow covers both self-service and administrative updates. Excludes: Password/Payment method changes (handled separately), bulk updates, or historical versioning per scope constraints. Aligns with Customer entity and audit requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* View Customer Purchase History   
\*\*Use Case ID:\*\* UC-06   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Order repository is accessible and operational.   
3. Customer records exist in the database.   
4. Administrator has privileges to view any customer's history (if applicable).   
  
\*\*Postconditions:\*\*   
1. Purchase history for target customer is displayed.   
2. Audit log records history access event (for Administrator actions).   
3. System maintains data integrity without modifications.   
  
\*\*Main Flow:\*\*   
1. Actor selects "View Purchase History" option.   
2. System presents:   
 - Customer: Direct interface to own history.   
 - Administrator: Customer search interface.   
3. Administrator enters customer search criteria (e.g., ID/name) and initiates search (not required for Customer).   
4. System retrieves target customer's order history:   
 - Selected customer (Administrator)   
 - Current session customer (Customer).   
5. System displays purchase history summary (Order ID, date, total, status).   
  
\*\*Alternative Flow:\*\*   
\*A: No Matching Customer (Administrator)\*   
1. At step 3, search yields no results.   
2. System displays "No customers found" alert.   
3. Administrator refines criteria or cancels operation.   
  
\*B: Empty Purchase History\*   
1. At step 4, target customer has no orders.   
2. System displays "No purchase records available" message.   
  
\*C: Session Timeout\*   
1. At any step, actor session expires.   
2. System invalidates access and redirects to login page.   
  
---  
  
> 🔍 \*Note: Focuses on read-only access to historical data. Excludes: Order modification flows, payment details, or bulk export features per scope constraints. Aligns with Order/Customer entities and audit requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Delete Customer Account   
\*\*Use Case ID:\*\* UC-07   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is logged into the system.   
2. Customer data repository is accessible.   
3. Target customer account exists in the system.   
4. Administrator has deletion privileges.   
  
\*\*Postconditions:\*\*   
1. Customer account is permanently removed from the system.   
2. Associated personal data is anonymized (orders preserved without identifiers).   
3. Audit log records deletion event.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Customers" function.   
2. System displays customer search interface.   
3. Administrator enters customer ID/email and initiates search.   
4. System retrieves and displays target customer profile.   
5. Administrator selects "Delete Account" option.   
6. System displays confirmation dialog with deletion consequences.   
7. Administrator confirms permanent deletion.   
8. System removes customer record and anonymizes personal data.   
9. System disables all authentication tokens for the account.   
10. System displays "Account deleted successfully" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: No Matching Account\*   
1. At step 4, no customer matches search criteria.   
2. System displays "No records found" alert.   
3. Administrator refines search or cancels operation.   
  
\*B: Deletion Cancelled\*   
1. At step 6 or 7, Administrator cancels action.   
2. System returns to customer profile view without changes.   
  
\*C: Order Dependency Detected\*   
1. At step 6, system identifies active orders.   
2. System displays "Cannot delete: X pending orders exist" warning.   
3. Flow returns to step 5 requiring order resolution first.   
  
---  
  
> 🔍 \*Note: Focuses on GDPR-compliant account erasure. Excludes: Self-service deletion, bulk processing, or payment data retention per scope constraints. Aligns with Customer/Order entities and data protection requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Add Product   
\*\*Use Case ID:\*\* UC-08   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated in the system.   
2. Product repository is accessible and operational.   
3. Administrator has privileges to create product records.   
  
\*\*Postconditions:\*\*   
1. New product record is created with unique identifier.   
2. Product is available in the catalog for customer orders.   
3. Creation event is logged in audit trail.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Product Management" function.   
2. System displays product catalog interface with "Add New" option.   
3. Administrator initiates product creation.   
4. System displays blank product form with mandatory fields (name, price, category).   
5. Administrator enters product details (name, description, price, stock quantity) and submits.   
6. System validates data integrity and uniqueness.   
7. System generates unique SKU and persists product record.   
8. System displays "Product added successfully" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Data Entry\*   
1. At step 6, validation fails (e.g., negative price, missing name).   
2. System highlights erroneous fields with error messages.   
3. Administrator corrects data and resubmits.   
  
\*B: Duplicate Product\*   
1. At step 6, system detects identical product name/SKU exists.   
2. System displays "Duplicate product detected" alert.   
3. Administrator modifies product attributes and resubmits.   
  
\*C: Stock Handling Exception\*   
1. At step 5, administrator enters stock level ≤0.   
2. System auto-flags product as "Out of Stock" during persistence.   
3. Record creation proceeds with inventory status notification.   
  
---  
  
> 🔍 \*Note: Focuses on core product creation workflow. Excludes: Media uploads, variant management, or supplier integration per scope constraints. Aligns with Product entity and catalog requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Update Product Details   
\*\*Use Case ID:\*\* UC-09   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with product management privileges.   
2. Product repository is accessible and operational.   
3. Target product exists in the system catalog.   
4. Required data fields (name, price, stock) are modifiable.   
  
\*\*Postconditions:\*\*   
1. Product details are updated in the database.   
2. Catalog reflects real-time changes for customer visibility.   
3. Audit log records modification event.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Product Management" function.   
2. System displays product search interface with filter options.   
3. Administrator enters product ID/SKU and initiates search.   
4. System retrieves and displays current product details.   
5. Administrator edits modifiable fields (price, description, stock level).   
6. System validates updated data for format and constraints.   
7. Administrator confirms changes.   
8. System persists updates and displays "Product updated successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Data Entry\*   
1. At step 6, validation fails (e.g., negative price).   
2. System highlights errors and retains entered values.   
3. Administrator corrects data and resubmits.   
  
\*B: Product Not Found\*   
1. At step 4, no matching product exists.   
2. System displays "Product not found" alert.   
3. Administrator refines search criteria.   
  
\*C: Concurrent Modification\*   
1. At step 8, system detects conflicting updates.   
2. System displays "Version conflict" warning with latest data.   
3. Administrator reviews changes and reapplies edits.   
  
---  
  
> 🔍 \*Note: Focuses on atomic product updates. Excludes: Batch editing, media asset management, or supplier sync per scope constraints. Aligns with Product entity and catalog integrity rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Delete Product   
\*\*Use Case ID:\*\* UC-10   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with product management privileges.   
2. Product repository and order/cart systems are accessible.   
3. Target product exists in the catalog.   
  
\*\*Postconditions:\*\*   
1. Product record is permanently removed from the catalog.   
2. Product is purged from all active shopping carts.   
3. Audit log records deletion event with timestamp and administrator ID.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Product Management" function.   
2. System displays product search interface with filter options.   
3. Administrator enters product SKU/name and initiates search.   
4. System retrieves and displays product details with "Delete" option.   
5. Administrator selects "Delete Product".   
6. System verifies no active orders contain the product (see Alternative Flow C).   
7. System displays deletion confirmation dialog with consequences.   
8. Administrator confirms permanent deletion.   
9. System removes product record and purges all cart references.   
10. System displays "Product deleted successfully" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: Product Not Found\*   
1. At step 4, search yields no results.   
2. System displays "Product not found" error with search criteria preserved.   
3. Administrator modifies criteria or cancels operation.   
  
\*B: Deletion Aborted\*   
1. At step 7 or 8, administrator cancels action.   
2. System returns to product details view without changes.   
  
\*C: Active Order Dependency\*   
1. At step 6, system detects product in pending/active orders.   
2. System displays "Cannot delete: X active orders reference this product".   
3. Flow returns to step 5 with resolution instructions.   
  
---  
  
> 🔍 \*Note: Focuses on atomic product deletion with dependency checks. Excludes: Archive functionality, batch deletions, or supplier notifications per scope constraints. Aligns with Product/Order/Cart entities and referential integrity rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* View Product Information   
\*\*Use Case ID:\*\* UC-11   
\*\*Actors:\*\* Customer (unauthenticated or authenticated), Administrator   
\*\*Preconditions:\*\*   
1. Product repository is accessible and operational.   
2. For Administrator: authenticated with product viewing privileges.   
3. Product catalog display functionality is enabled.   
  
\*\*Postconditions:\*\*   
1. Requested product details are displayed.   
2. System audit log records administrator access (if applicable).   
3. No data modifications occur (read-only operation).   
  
\*\*Main Flow:\*\*   
1. Actor initiates product viewing:   
 - Customer: Navigates to product catalog or uses search function.   
 - Administrator: Selects "View Products" in management interface.   
2. System displays appropriate interface:   
 - Customer: Public product browsing with search/categories.   
 - Administrator: Product search form with filter options.   
3. Actor identifies target product:   
 - Customer: Browses categories or searches, then selects product.   
 - Administrator: Enters search criteria (e.g., SKU/name), executes search, selects product from results.   
4. System retrieves product details from repository.   
5. System displays product information page with name, description, price, and availability status.   
  
\*\*Alternative Flow:\*\*   
\*A: Product Not Found on Retrieval\*   
1. At step 4, system fails to locate product (invalid ID or removed).   
2. System displays "Product unavailable" error.   
3. Actor returned to step 2 interface.   
  
\*B: No Matching Products (Administrator)\*   
1. At step 3 (search execution), system finds no matches.   
2. System displays "No products found" in results list.   
3. Administrator refines criteria or cancels operation.   
  
\*C: Restricted Access Attempt\*   
1. At step 1, unauthenticated administrator attempts management interface.   
2. System redirects to login page with "Authentication required" alert.   
  
---   
> 🔍 \*Note: Focuses on core product viewing flow. Excludes: Add-to-cart actions, inventory-level visibility rules, or third-party data integrations per scope constraints. Aligns with Product entity and catalog requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage Inventory   
\*\*Use Case ID:\*\* UC-12   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with inventory management privileges.   
2. Product repository and inventory tracking system are accessible.   
3. At least one product record exists in the catalog.   
  
\*\*Postconditions:\*\*   
1. Product stock levels are updated in the system.   
2. Inventory change records are logged with timestamp and administrator ID.   
3. Product availability status is synchronized across the catalog.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Inventory" function.   
2. System displays inventory dashboard with product search filters.   
3. Administrator enters product SKU/name and initiates search.   
4. System retrieves and displays current stock level and inventory history.   
5. Administrator inputs stock adjustment quantity (+/-) and reason.   
6. System validates new quantity (non-negative value).   
7. Administrator confirms inventory update.   
8. System persists stock change and logs adjustment details.   
9. System updates product availability status in catalog.   
  
\*\*Alternative Flow:\*\*   
\*A: Bulk Inventory Update\*   
1. At step 2, Administrator selects "Bulk Update" option.   
2. System displays CSV template download and upload interface.   
3. Administrator uploads adjusted inventory file.   
4. System validates all entries and shows preview.   
5. Administrator confirms batch update.   
  
\*B: Stock Threshold Alert\*   
1. At step 4, system detects stock below minimum threshold.   
2. System highlights product with "Low Stock" warning.   
3. Administrator initiates restock workflow from alert interface.   
  
\*C: Invalid Adjustment Quantity\*   
1. At step 6, validation fails (e.g., negative final quantity).   
2. System displays "Invalid stock level" error with max reduction limit.   
3. Administrator corrects quantity and resubmits.   
  
---  
  
> 🔍 \*Note: Focuses on core inventory adjustment workflows. Excludes: Supplier integration, automatic reordering, or multi-warehouse management per scope constraints. Aligns with Product entity and inventory control requirements.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Create Order   
\*\*Use Case ID:\*\* UC-13   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is logged into the system.   
2. Shopping cart contains at least one product.   
3. Product inventory is available for all cart items.   
4. Order management system is accessible.   
  
\*\*Postconditions:\*\*   
1. New order record is created with unique ID.   
2. Shopping cart is cleared.   
3. Product inventory is decremented for purchased items.   
4. Order status is set to "Pending Payment".   
  
\*\*Main Flow:\*\*   
1. Customer selects "Proceed to Checkout" from cart page.   
2. System displays order summary with cart items, prices, and total.   
3. Customer confirms shipping address and contact details.   
4. Customer selects payment method and enters required details.   
5. System validates payment information format.   
6. System creates order record and generates unique order ID.   
7. System reserves inventory for all ordered products.   
8. System clears shopping cart contents.   
9. System displays order confirmation with order ID.   
  
\*\*Alternative Flow:\*\*   
\*A: Insufficient Inventory\*   
1. At step 6, system detects low stock for any product.   
2. System displays "Insufficient stock" alert with affected items.   
3. Customer removes out-of-stock items or cancels order.   
  
\*B: Payment Validation Failure\*   
1. At step 5, payment details format is invalid.   
2. System highlights errors without clearing entered data.   
3. Customer corrects information and resubmits.   
  
\*C: Address Validation Error\*   
1. At step 3, shipping address fails format check.   
2. System flags invalid fields with correction guidance.   
3. Customer updates address details and continues.   
  
---   
> 🔍 \*Note: Focuses on core order creation workflow. Excludes: Tax calculations, coupon applications, split payments, and real-time payment processing per scope constraints. Aligns with Order, Cart, and Product entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* View Order Details   
\*\*Use Case ID:\*\* UC-14   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Order repository is accessible and operational.   
3. Target order exists in the database.   
4. Customer can only view own orders; Administrator can view any order.   
  
\*\*Postconditions:\*\*   
1. Detailed order information is displayed.   
2. Audit log records access event (for Administrator actions).   
3. No data modifications occur (read-only operation).   
  
\*\*Main Flow:\*\*   
1. Actor navigates to order management interface:   
 - Customer: Selects "My Orders"   
 - Administrator: Selects "Manage Orders"   
2. System presents order search/list interface:   
 - Customer: Displays personal order history with summaries   
 - Administrator: Provides search filters (order ID, date range, customer)   
3. Actor identifies target order:   
 - Customer: Selects order from list   
 - Administrator: Enters search criteria and selects order from results   
4. System retrieves full order details from repository.   
5. System displays comprehensive order view:   
 - Products (names, quantities, prices)   
 - Payment status and shipping information   
 - Order timestamp and current status   
  
\*\*Alternative Flow:\*\*   
\*A: Order Not Found\*   
1. At step 3 (Administrator search), no matching orders exist.   
2. System displays "No orders found" alert with preserved criteria.   
3. Administrator refines search parameters.   
  
\*B: Unauthorized Access\*   
1. At step 4, Customer attempts to view non-owned order.   
2. System blocks access and displays "Order unavailable" message.   
3. Customer returned to personal order history.   
  
\*C: Session Expiration\*   
1. At any step, actor session times out.   
2. System invalidates access and redirects to login page.   
  
---   
> 🔍 \*Note: Focuses on core order viewing functionality. Excludes: Order modification, invoice generation, or return processing per scope constraints. Aligns with Order, Product, and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Update Order Status   
\*\*Use Case ID:\*\* UC-15   
\*\*Actors:\*\* Administrator, System (automated triggers)   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with order management privileges.   
2. Order repository is accessible and operational.   
3. Target order exists in the database.   
4. Order status transition complies with business rules (e.g., "Shipped" cannot revert to "Processing").   
  
\*\*Postconditions:\*\*   
1. Order status is updated in the database.   
2. Status change event is recorded in audit logs.   
3. Dependent systems are notified (e.g., inventory restocked for cancellations).   
4. Customer receives status update notification (if applicable).   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Orders" function.   
2. System displays order search interface with filters (ID, date range, status).   
3. Administrator enters target Order ID and initiates search.   
4. System retrieves and displays current order details (products, customer, payment status).   
5. Administrator selects "Update Status" and chooses new status (e.g., Shipped, Cancelled).   
6. System validates status transition rules (e.g., "Delivered" orders cannot be modified).   
7. Administrator confirms status change with optional notes.   
8. System persists status update and triggers notifications.   
9. System displays "Status updated successfully" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Status Transition\*   
1. At step 6, system detects invalid status change (e.g., "Cancelled" → "Shipped").   
2. System displays "Invalid status transition" error with allowed options.   
3. Administrator selects valid status or cancels operation.   
  
\*B: Order Not Found\*   
1. At step 4, search yields no matching orders.   
2. System displays "Order not found" alert with preserved search criteria.   
3. Administrator corrects Order ID or cancels.   
  
\*C: Automated Status Update\*   
1. System detects external trigger (e.g., payment gateway confirmation).   
2. System auto-updates status (e.g., "Pending Payment" → "Processing").   
3. Audit log records automated update with trigger source.   
  
\*D: Cancellation with Restocking\*   
1. At step 5, Administrator selects "Cancelled" status.   
2. System verifies no shipment occurred.   
3. System restocks reserved inventory and updates product records.   
  
---   
> 🔍 \*Note: Focuses on atomic status transitions. Excludes: Bulk updates, return management, or shipment tracking integrations per scope constraints. Aligns with Order, Product, and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Delete Order   
\*\*Use Case ID:\*\* UC-16   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with order management privileges.   
2. Order repository is accessible and operational.   
3. Target order exists in the database.   
4. Order status permits deletion (e.g., "Pending" or "Cancelled" status).   
  
\*\*Postconditions:\*\*   
1. Order record is permanently removed from the system.   
2. Reserved inventory is restocked (if applicable).   
3. Audit log records deletion event with timestamp.   
4. Associated payment records are archived per compliance rules.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Orders" function.   
2. System displays order search interface with filters (ID, status).   
3. Administrator enters target Order ID and initiates search.   
4. System retrieves and displays order details (products, status, customer).   
5. Administrator selects "Delete Order" option.   
6. System verifies deletable status (see Alternative Flow C).   
7. System displays deletion confirmation with impact summary.   
8. Administrator confirms permanent deletion.   
9. System removes order record and restocks reserved inventory.   
10. System displays "Order deleted successfully" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: Order Not Found\*   
1. At step 4, search yields no matching orders.   
2. System displays "Order not found" alert.   
3. Administrator corrects Order ID or cancels.   
  
\*B: Deletion Aborted\*   
1. At step 7 or 8, administrator cancels action.   
2. System returns to order details view without changes.   
  
\*C: Invalid Status for Deletion\*   
1. At step 6, order status prohibits deletion (e.g., "Shipped").   
2. System displays "Cannot delete: Order status is [status]" alert.   
3. Flow returns to step 5 requiring status change first.   
  
---  
  
> 🔍 \*Note: Focuses on atomic order deletion with dependency checks. Excludes: Bulk deletions, partial cancellations, or shipment reversal workflows per scope constraints. Aligns with Order/Product entities and inventory rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Add to Cart   
\*\*Use Case ID:\*\* UC-17   
\*\*Actors:\*\* Customer (authenticated or unauthenticated)   
\*\*Preconditions:\*\*   
1. Product catalog is accessible and functional.   
2. Target product exists in the catalog and is active.   
3. Shopping cart service is operational.   
  
\*\*Postconditions:\*\*   
1. Selected product and quantity are added to the customer's cart.   
2. Cart total and item count are updated in real-time.   
3. Product availability status remains unchanged (inventory reserved at checkout).   
  
\*\*Main Flow:\*\*   
1. Customer views product details on catalog page.   
2. Customer selects desired quantity (default: 1) and clicks "Add to Cart".   
3. System validates:   
 - Product is active   
 - Quantity is positive integer   
 - Total quantity (current cart + new) ≤ available stock   
4. System adds product with quantity to cart (creates new entry or increments existing).   
5. System updates cart summary (total items/cost).   
6. System displays "[Product] added to cart" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Product Unavailable\*   
1. At step 3, product is inactive or stock=0.   
2. System displays "Product unavailable" error.   
3. Cart remains unchanged.   
  
\*B: Invalid Quantity\*   
1. At step 3, quantity ≤0 or non-integer.   
2. System highlights field with "Enter positive quantity" error.   
3. Customer corrects input and resubmits.   
  
\*C: Insufficient Stock\*   
1. At step 3, (cart quantity + new) > available stock.   
2. System calculates max addable (stock - current cart).   
3. System displays "Only [max] more available" alert.   
4. Customer adjusts quantity or cancels action.   
  
---  
  
> 🔍 \*Note: Focuses on core cart addition workflow. Excludes: Cart management UI interactions, inventory reservation, or guest-to-member cart conversion. Aligns with Product and Cart entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Modify Cart Contents   
\*\*Use Case ID:\*\* UC-18   
\*\*Actors:\*\* Customer (authenticated or unauthenticated)   
\*\*Preconditions:\*\*   
1. Shopping cart exists (initialized during session or created via UC-17).   
2. Cart contains at least one modifiable item.   
3. Product repository is accessible.   
4. Cart management functionality is operational.   
  
\*\*Postconditions:\*\*   
1. Cart contents are updated per modification request.   
2. Cart totals and item counts are recalculated.   
3. Changes are persisted (saved cart for authenticated users; session storage for guests).   
  
\*\*Main Flow:\*\*   
1. Customer navigates to cart summary page.   
2. System displays current cart contents with product details (name, unit price), quantities, and subtotals.   
3. Customer selects modification action for specific item:   
 - Increase/decrease quantity input field   
 - "Remove" button   
4. Customer confirms modification.   
5. System validates request:   
 - For quantity change: new quantity ≤ available stock   
 - For removal: item exists in cart   
6. System updates cart:   
 - Adjusts item quantity/removes item   
 - Recalculates cart totals   
7. System displays updated cart with "Cart updated" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Quantity Request\*   
1. At step 5, requested quantity exceeds stock limit.   
2. System displays "Maximum available: [stock]" alert near item.   
3. Auto-sets quantity to maximum available.   
4. Proceed to step 6.   
  
\*B: Product Unavailable During Edit\*   
1. At step 5, system detects product deactivation/stock=0.   
2. System removes item automatically with "[Product] discontinued" notice.   
3. Recalculate totals (step 6).   
  
\*C: Clear Entire Cart\*   
1. At step 3, customer selects "Empty Cart" option.   
2. System prompts for confirmation.   
3. Upon confirmation, system removes all items.   
4. Display "Cart is empty" state.   
  
---  
  
> 🔍 \*Note: Focuses on atomic cart modifications. Excludes: Cart-to-wishlist transfers, bulk operations, or coupon applications per scope constraints. Aligns with Cart/Product entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Process Checkout   
\*\*Use Case ID:\*\* UC-19   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is logged into the system.   
2. Shopping cart contains at least one product.   
3. All cart items have sufficient inventory.   
4. Payment gateway connectivity is operational.   
  
\*\*Postconditions:\*\*   
1. New order record is created with status "Paid".   
2. Shopping cart is cleared and inventory decremented.   
3. Payment transaction is recorded and validated.   
4. Order confirmation is sent to customer.   
  
\*\*Main Flow:\*\*   
1. Customer selects "Checkout" from cart page.   
2. System displays order summary (items, subtotal, taxes, shipping cost).   
3. Customer confirms default shipping address or enters new valid address.   
4. Customer selects payment method and enters valid payment details.   
5. System validates payment information format and fund availability.   
6. System creates order record with unique ID and reserves inventory.   
7. Payment gateway processes transaction successfully.   
8. System updates order status to "Paid" and clears cart.   
9. System sends email confirmation with order details.   
10. System displays "Order completed" page with tracking information.   
  
\*\*Alternative Flow:\*\*   
\*A: Address Validation Failure\*   
1. At step 3, address format violates postal rules.   
2. System highlights invalid fields with correction guide.   
3. Customer updates address and continues checkout.   
  
\*B: Payment Authorization Decline\*   
1. At step 5/7, payment gateway declines transaction.   
2. System displays "Payment declined: [reason]" alert.   
3. Customer updates payment details or selects alternative method.   
  
\*C: Inventory Shortfall During Checkout\*   
1. At step 6, real-time inventory check shows insufficient stock.   
2. System removes out-of-stock items automatically.   
3. System displays updated cart with available items only.   
4. Customer confirms revised order or cancels checkout.   
  
\*D: Session Timeout\*   
1. At any step, customer session expires.   
2. System saves cart state and redirects to login page.   
3. After re-authentication, system restores checkout progress.   
  
---  
  
> 🔍 \*Note: Focuses on end-to-end transaction completion. Excludes: Split payments, gift wrapping options, or loyalty point redemption per scope constraints. Aligns with Order, Cart, Payment, and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* View Cart Summary   
\*\*Use Case ID:\*\* UC-20   
\*\*Actors:\*\* Customer (authenticated or unauthenticated)   
\*\*Preconditions:\*\*   
1. Shopping cart service is accessible and operational.   
2. Customer has an active session with cart data initialized.   
3. Product repository is available for real-time validation.   
  
\*\*Postconditions:\*\*   
1. Current cart contents and summary are displayed.   
2. Cart state remains unchanged (read-only operation).   
3. Product availability status is verified but not modified.   
  
\*\*Main Flow:\*\*   
1. Customer selects "View Cart" option.   
2. System retrieves cart contents (product IDs, quantities) from session storage.   
3. System fetches current product details (name, price, availability status) for each item.   
4. System calculates:   
 - Line-item subtotals (unit price × quantity)   
 - Cart subtotal (sum of all line-item values)   
5. System displays cart summary including:   
 - Product names with thumbnails   
 - Unit prices and quantities   
 - Availability indicators   
 - Cart subtotal   
  
\*\*Alternative Flow:\*\*   
\*A: Empty Cart\*   
1. At step 2, system detects zero items in cart.   
2. System displays "Your cart is empty" message with product browsing suggestions.   
  
\*B: Product Unavailable\*   
1. At step 3, system detects discontinued/out-of-stock products.   
2. System flags affected items with "No longer available" warnings.   
3. Excludes unavailable items from subtotal calculation.   
  
\*C: Price Discrepancy\*   
1. At step 3, system detects price changes since item addition.   
2. System displays both original and current prices with change indicators.   
3. Uses current price for subtotal calculations.   
  
---  
  
> 🔍 \*Note: Focuses on real-time cart snapshot. Excludes: Cart modification features, tax/shipping calculations, and checkout initiation per scope constraints. Aligns with Cart and Product entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Create Payment Method   
\*\*Use Case ID:\*\* UC-21   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is logged into the system.   
2. Payment management functionality is accessible.   
3. Customer has not exceeded maximum saved payment methods (if applicable).   
4. Payment gateway integration is operational.   
  
\*\*Postconditions:\*\*   
1. New payment method record is securely stored in the system.   
2. Payment method is associated with the customer's account.   
3. Payment method becomes available for future transactions.   
  
\*\*Main Flow:\*\*   
1. Customer selects "Payment Methods" in account settings.   
2. System displays existing payment methods with "Add New" option.   
3. Customer initiates creation of new payment method.   
4. System displays form with required fields (e.g., card type, number, expiration date).   
5. Customer enters valid payment details and confirms submission.   
6. System validates input format and performs Luhn check (for cards).   
7. System tokenizes sensitive data via payment gateway.   
8. System saves payment method record linked to customer account.   
9. System displays "Payment method added" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Input Format\*   
1. At step 6, validation fails (e.g., expired date, incorrect CVV).   
2. System highlights erroneous fields with specific error messages.   
3. Customer corrects data and resubmits.   
  
\*B: Duplicate Payment Method\*   
1. At step 6, system detects identical payment details.   
2. System displays "Payment method already exists" alert.   
3. Customer selects existing method or provides new details.   
  
\*C: Payment Gateway Failure\*   
1. At step 7, tokenization request fails.   
2. System displays "Temporary service issue" error with retry option.   
3. Customer retries operation or aborts process.   
  
---   
> 🔍 \*Note: Focuses on core payment method creation. Excludes: Billing address management, payment verification charges, or third-party wallets per scope constraints. Aligns with Payment and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Process Payment   
\*\*Use Case ID:\*\* UC-22   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Order exists with status "Pending Payment".   
2. Actor is authenticated and authorized to process payment for the order.   
3. Payment gateway connectivity is operational.   
4. Sufficient funds/credit available for the transaction (if applicable).   
  
\*\*Postconditions:\*\*   
1. Payment transaction is recorded and linked to the order.   
2. Order status is updated to "Paid".   
3. Inventory is decremented for all order items (if not previously reserved).   
4. Payment confirmation is sent to customer and recorded in audit logs.   
  
\*\*Main Flow:\*\*   
1. Actor accesses target order details (Customer via "My Orders", Administrator via "Manage Orders").   
2. Actor selects "Process Payment" option.   
3. System retrieves order amount and displays available payment methods.   
4. Actor selects payment method:   
 - Saved payment method (Customer)   
 - Manual entry (card/transfer details)   
 - Offline payment (Administrator)   
5. System validates payment details format and fund availability.   
6. Payment gateway processes transaction and returns authorization.   
7. System records payment transaction ID, amount, and timestamp.   
8. System updates order status to "Paid".   
9. System sends payment confirmation email to customer.   
  
\*\*Alternative Flow:\*\*   
\*A: Payment Authorization Decline\*   
1. At step 6, payment gateway declines transaction.   
2. System displays "Payment failed: [reason]" alert.   
3. Actor selects alternative payment method or cancels operation.   
  
\*B: Invalid Payment Details\*   
1. At step 5, validation fails (e.g., expired card, invalid CVV).   
2. System highlights erroneous fields with correction guidelines.   
3. Actor corrects details and resubmits.   
  
\*C: Offline Payment Processing\*   
1. At step 4, Administrator selects "Mark as Paid Offline".   
2. System prompts for payment evidence (receipt/transaction ID).   
3. Administrator uploads proof and confirms manual completion.   
4. System updates status without gateway interaction.   
  
\*D: Partial Payment\*   
1. At step 4, actor specifies partial payment amount.   
2. System verifies minimum payment threshold compliance.   
3. Payment processes for partial amount; order status updates to "Partially Paid".   
  
---  
  
> 🔍 \*Note: Focuses on atomic payment processing. Excludes: Refund workflows, payment plan setups, or multi-currency conversions per scope constraints. Aligns with Order and Payment entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* View Payment Details   
\*\*Use Case ID:\*\* UC-23   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Payment repository is accessible and operational.   
3. Target order exists in the database.   
4. Payment record is associated with the order.   
5. Customer can only view own payments; Administrator can view any payment.   
  
\*\*Postconditions:\*\*   
1. Payment details for the target order are displayed.   
2. Audit log records access event (for Administrator actions).   
3. No data modifications occur (read-only operation).   
  
\*\*Main Flow:\*\*   
1. Actor navigates to order/payment management interface:   
 - Customer: Selects "My Orders" → chooses target order   
 - Administrator: Selects "Manage Payments" → enters order ID/customer details   
2. Actor selects "View Payment Details" option for target order.   
3. System retrieves payment record(s) linked to the order.   
4. System displays payment details:   
 - Payment method and transaction ID   
 - Amount paid and currency   
 - Payment status and timestamp   
 - Billing address (if applicable)   
  
\*\*Alternative Flow:\*\*   
\*A: Payment Record Not Found\*   
1. At step 3, no payment record exists for the order.   
2. System displays "No payment details available" alert.   
3. Flow returns to order view interface.   
  
\*B: Unauthorized Access (Customer)\*   
1. At step 3, Customer attempts to view non-owned payment.   
2. System blocks access and displays "Payment details unavailable".   
3. Redirects to customer's order history.   
  
\*C: Order-Payment Mismatch\*   
1. At step 3, payment amount ≠ order total.   
2. System displays payment discrepancy warning with variance amount.   
3. Details view includes reconciliation notes section.   
  
---  
  
> 🔍 \*Note: Focuses on core payment information retrieval. Excludes: Payment modification, refund processing, or gateway error diagnostics per scope constraints. Aligns with Payment and Order entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Update Payment Information   
\*\*Use Case ID:\*\* UC-24   
\*\*Actors:\*\* Customer, Administrator   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Payment repository is accessible and operational.   
3. Target payment method exists in the system.   
4. Customer can update only own payment methods; Administrator can update any.   
  
\*\*Postconditions:\*\*   
1. Payment method details are modified and persisted.   
2. Updated information is synchronized with payment gateway.   
3. Audit log records update event with timestamp and actor ID.   
4. Confirmation notification is displayed to the actor.   
  
\*\*Main Flow:\*\*   
1. Actor navigates to payment management interface:   
 - Customer: Selects "My Payment Methods"   
 - Administrator: Selects "Manage Payments" → searches customer account   
2. Actor selects target payment method for update.   
3. System displays current payment details (e.g., card type, last digits, expiration).   
4. Actor edits modifiable fields (e.g., expiration date, billing address).   
5. System validates new data format and constraints.   
6. Actor confirms changes.   
7. System tokenizes sensitive data via payment gateway.   
8. System persists updates and displays "Payment updated successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Data Format\*   
1. At step 5, validation fails (e.g., expired date, invalid CVV).   
2. System highlights errors with correction guidance.   
3. Actor corrects fields and resubmits.   
  
\*B: Payment Method Not Found\*   
1. At step 3, selected payment record is invalid or deleted.   
2. System displays "Payment method unavailable" alert.   
3. Actor returns to payment management interface.   
  
\*C: Gateway Tokenization Failure\*   
1. At step 7, payment gateway rejects tokenization.   
2. System displays "Update failed: Payment service error".   
3. Actor retries operation or contacts support.   
  
---  
  
> 🔍 \*Note: Focuses on core payment data modification. Excludes: Primary method selection, fraud detection workflows, or bulk updates per scope constraints. Aligns with Payment and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Delete Payment Method   
\*\*Use Case ID:\*\* UC-25   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer is logged into the system.   
2. Payment repository is accessible and operational.   
3. Target payment method exists and is associated with the customer’s account.   
4. Payment method is not actively used in pending orders or subscriptions.   
  
\*\*Postconditions:\*\*   
1. Payment method is permanently removed from the system.   
2. Payment method is dissociated from the customer account.   
3. Deletion event is recorded in audit logs.   
  
\*\*Main Flow:\*\*   
1. Customer selects "Payment Methods" in account settings.   
2. System displays saved payment methods with "Delete" options.   
3. Customer selects target payment method and chooses "Delete".   
4. System displays confirmation dialog with payment method details (e.g., card type, last digits).   
5. Customer confirms deletion.   
6. System verifies no active dependencies (e.g., pending orders).   
7. System permanently deletes payment method record.   
8. System displays "Payment method deleted successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Deletion Cancelled\*   
1. At step 5, customer cancels deletion.   
2. System returns to payment methods list without changes.   
  
\*B: Active Dependency Detected\*   
1. At step 6, system detects payment method linked to pending orders/subscriptions.   
2. System displays "Cannot delete: Payment method in use" alert.   
3. Flow returns to step 2 requiring dependency resolution.   
  
\*C: Payment Method Not Found\*   
1. At step 3, system fails to locate selected payment method (e.g., concurrent deletion).   
2. System displays "Payment method unavailable" error.   
3. Customer returned to payment methods list.   
  
---   
> 🔍 \*Note: Focuses on atomic payment method removal. Excludes: Payment gateway sync, bulk deletions, or archival workflows per scope constraints. Aligns with Payment and Customer entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Register Administrator   
\*\*Use Case ID:\*\* UC-26   
\*\*Actors:\*\* Super Administrator   
\*\*Preconditions:\*\*   
1. Super Administrator is authenticated in the system.   
2. Administrator repository is accessible and operational.   
3. Super Administrator has user management privileges.   
4. Target email domain complies with organizational policies.   
  
\*\*Postconditions:\*\*   
1. New administrator record is created with unique ID.   
2. Administrator credentials are stored securely.   
3. System sends account activation instructions to the new administrator.   
4. Audit log records creation event.   
  
\*\*Main Flow:\*\*   
1. Super Administrator selects "Manage Administrators" in system settings.   
2. System displays administrator list with "Register New" option.   
3. Super Administrator initiates registration.   
4. System displays form with mandatory fields (name, email, role).   
5. Super Administrator enters valid details and sets initial password.   
6. System validates email uniqueness and password strength.   
7. System generates administrator ID and stores encrypted credentials.   
8. System sends activation email to the new administrator.   
9. System displays "Administrator registered successfully" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Input Data\*   
1. At step 6, validation fails (e.g., invalid email format).   
2. System highlights errors without clearing form.   
3. Super Administrator corrects data and resubmits.   
  
\*B: Duplicate Administrator\*   
1. At step 6, system detects existing administrator email.   
2. System displays "Email already registered" alert.   
3. Super Administrator modifies email or cancels operation.   
  
\*C: Privilege Escalation Attempt\*   
1. At step 5, Super Administrator assigns higher role than permitted.   
2. System auto-downgrades to maximum allowed role.   
3. System notifies "Role adjusted per permissions" with audit trail.   
  
---  
  
> 🔍 \*Note: Focuses on administrative account creation by privileged users. Excludes: Self-registration, multi-factor setup, or bulk imports per scope constraints. Aligns with Administrator entity and security protocols.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Update Administrator Information   
\*\*Use Case ID:\*\* UC-27   
\*\*Actors:\*\* Administrator (self-update), Super Administrator (update others)   
\*\*Preconditions:\*\*   
1. Actor is authenticated in the system.   
2. Administrator repository is accessible and operational.   
3. Target administrator record exists in the database.   
4. Actor has required permissions:   
 - Administrator can update own non-privileged fields (e.g., contact info)   
 - Super Administrator can update any administrator's information.   
  
\*\*Postconditions:\*\*   
1. Administrator record is updated in the database.   
2. Change audit log is generated with timestamp and actor ID.   
3. Confirmation notification is displayed to the actor.   
  
\*\*Main Flow:\*\*   
1. Actor navigates to administrator management interface:   
 - Administrator: Selects "My Profile"   
 - Super Administrator: Selects "Manage Administrators"   
2. System displays:   
 - Administrator: Current profile details (non-editable view)   
 - Super Administrator: Administrator search interface   
3. Super Administrator enters target admin ID/email and executes search (skip for self-update).   
4. System retrieves and displays target administrator details.   
5. Actor selects "Edit" option and modifies allowed fields (e.g., name, email, phone number).   
6. System validates input format and business rules.   
7. Actor confirms changes.   
8. System persists updates and displays "Administrator updated successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Administrator Not Found (Super Admin)\*   
1. At step 3, search yields no results.   
2. System displays "No matching administrator found".   
3. Super Administrator refines criteria or cancels.   
  
\*B: Validation Failure\*   
1. At step 6, data violates rules (e.g., invalid email format).   
2. System highlights errors and retains entered values.   
3. Actor corrects data and resubmits.   
  
\*C: Unauthorized Field Modification\*   
1. At step 5, Administrator attempts restricted change (e.g., role permissions).   
2. System disables restricted fields and displays "Field not editable" warning.   
  
---  
  
> 🔍 \*Note: Focuses on atomic administrator updates. Excludes: Password changes (handled separately), bulk updates, or role permission overrides. Aligns with Administrator entity and privilege segregation rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Delete Administrator Account   
\*\*Use Case ID:\*\* UC-28   
\*\*Actors:\*\* Super Administrator   
\*\*Preconditions:\*\*   
1. Super Administrator is authenticated in the system.   
2. Administrator repository is accessible and operational.   
3. Target administrator account exists in the database.   
4. Super Administrator has higher privilege level than target administrator.   
  
\*\*Postconditions:\*\*   
1. Target administrator account is permanently removed from the system.   
2. All associated permissions and sessions are revoked.   
3. Audit log records deletion event with Super Administrator ID and timestamp.   
  
\*\*Main Flow:\*\*   
1. Super Administrator selects "Manage Administrators" in system settings.   
2. System displays administrator list with search filters (ID, email, role).   
3. Super Administrator enters target administrator ID/email and executes search.   
4. System retrieves and displays administrator details (role, last login, assigned permissions).   
5. Super Administrator selects "Delete Account" option.   
6. System verifies privilege hierarchy (target ≠ current user and lower privilege).   
7. System displays confirmation dialog with deletion consequences.   
8. Super Administrator confirms permanent deletion.   
9. System purges administrator record and associated credentials.   
10. System displays "Administrator deleted successfully" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: Administrator Not Found\*   
1. At step 4, search yields no matching records.   
2. System displays "No administrator found" alert with preserved criteria.   
3. Super Administrator refines search or cancels operation.   
  
\*B: Privilege Violation Attempt\*   
1. At step 6, target has equal/higher privilege than Super Administrator.   
2. System blocks deletion and displays "Insufficient privileges" error.   
3. Flow returns to administrator list view.   
  
\*C: Active Session Dependency\*   
1. At step 6, system detects active sessions for target account.   
2. System forces session terminations before proceeding to step 7.   
3. Audit logs record automatic session invalidation.   
  
---  
  
> 🔍 \*Note: Focuses on atomic administrator deletion with privilege enforcement. Excludes: Account deactivation/recovery, bulk deletions, or permission reassignment workflows per scope constraints. Aligns with Administrator entity and security hierarchy rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage Users   
\*\*Use Case ID:\*\* UC-29   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated in the system.   
2. User repository (Customer/Administrator) is accessible.   
3. Administrator has user management privileges.   
4. Target user record exists (for modification/deletion).   
  
\*\*Postconditions:\*\*   
1. User records are created, updated, or deactivated.   
2. Changes are audited and persisted in the database.   
3. Authentication access is modified for deactivated accounts.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Users" function.   
2. System displays user type selection (Customer/Administrator).   
3. Administrator selects user type and enters search criteria (ID/email).   
4. System retrieves and displays user details (contact info, status, roles).   
5. Administrator chooses action:   
 a. \*\*Edit\*\*: Updates fields (e.g., email, permissions) and confirms.   
 b. \*\*Create New\*\*: Enters mandatory details (name, email, role) and submits.   
 c. \*\*Deactivate\*\*: Confirms account suspension.   
6. System validates inputs and executes changes.   
7. System displays "Operation successful" notification.   
  
\*\*Alternative Flow:\*\*   
\*A: Create New User\*   
1. At step 2, Administrator selects "Create New" before searching.   
2. System displays blank form based on user type selection.   
3. Proceed to step 5b in Main Flow.   
  
\*B: Invalid Data Entry\*   
1. At step 6, validation fails (e.g., duplicate email, invalid role).   
2. System highlights errors without clearing form.   
3. Administrator corrects data and resubmits.   
  
\*C: Deactivation Conflict\*   
1. At step 5c, system detects active orders (customers) or privileges (admins).   
2. System displays "Resolution required" alert (e.g., transfer orders).   
3. Administrator resolves dependencies before retrying.   
  
\*D: Unauthorized Modification\*   
1. At step 5, Administrator attempts restricted changes (e.g., own privileges).   
2. System blocks action and displays "Permission denied".   
  
---   
> 🔍 \*Note: Unified user management for Customer/Administrator entities. Excludes: Password resets, bulk operations, or third-party integrations. Consistent with UC-01 and UC-26 formats.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage Products   
\*\*Use Case ID:\*\* UC-30   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with product management privileges.   
2. Product repository is accessible and operational.   
3. Catalog management functionality is enabled.   
  
\*\*Postconditions:\*\*   
1. New products are added to the catalog, existing products are updated or deleted.   
2. Inventory status and pricing are synchronized system-wide.   
3. Audit logs record all modifications with timestamp and administrator ID.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Products" in the control panel.   
2. System displays product management dashboard with options: Create/Edit/Delete.   
3. Administrator chooses operation:   
 - \*\*Create\*\*: Proceeds to blank product form   
 - \*\*Edit\*\*: Enters product SKU/name and initiates search   
 - \*\*Delete\*\*: Enters product SKU/name and initiates search   
4. For \*\*Edit/Delete\*\*:   
 - System retrieves product details   
 - Administrator modifies fields (Edit) or selects "Delete"   
5. System validates inputs against business rules.   
6. Administrator confirms action.   
7. System persists changes and updates catalog.   
8. System displays "Operation completed successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Create New Product\*   
1. At step 3, Administrator selects "Create".   
2. System displays form with mandatory fields (name, price, category).   
3. Administrator enters details and submits.   
4. System generates unique SKU and saves record.   
  
\*B: Edit Product\*   
1. At step 4 (Edit path), Administrator modifies fields.   
2. System validates data integrity (e.g., non-negative price).   
3. On validation failure, highlights errors for correction.   
  
\*C: Delete Product\*   
1. At step 4 (Delete path), system checks for active orders.   
2. If dependencies exist, displays "Cannot delete: X active orders".   
3. Administrator resolves dependencies or cancels.   
  
\*D: Invalid Search\*   
1. At step 4, no matching product found.   
2. System displays "Product not found" with preserved search criteria.   
3. Administrator refines search or cancels.   
  
---  
  
> 🔍 \*Note: Unified CRUD workflow for product lifecycle management. Excludes: Batch operations, media asset handling, and supplier synchronization per scope. Aligns with Product entity and catalog integrity rules.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage Orders   
\*\*Use Case ID:\*\* UC-31   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with order management privileges.   
2. Order repository is accessible and operational.   
3. Customer and product data repositories are available.   
  
\*\*Postconditions:\*\*   
1. Orders are created, updated, or deleted as requested.   
2. Order status changes are recorded in audit logs.   
3. Inventory adjustments are synchronized for modifications/cancellations.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "Manage Orders" in the control panel.   
2. System displays order dashboard with search filters (ID, status, date range).   
3. Administrator chooses operation:   
 - \*\*Create\*\*: Enters customer ID and product details   
 - \*\*Edit\*\*: Searches for target order via ID   
 - \*\*Delete\*\*: Searches for target order via ID   
4. For \*\*Edit/Delete\*\*: System retrieves order details and verifies mutable status (e.g., "Pending").   
5. Administrator confirms action:   
 - \*\*Create\*\*: Submits new order with customer/items   
 - \*\*Edit\*\*: Modifies shipping/items and saves   
 - \*\*Delete\*\*: Confirms permanent removal   
6. System processes changes:   
 - Creates order with "Processing" status   
 - Updates order records and inventory   
 - Deletes order and restocks items   
7. System displays operation confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Create New Order\*   
1. At step 3a, Administrator selects customer from search results.   
2. System displays product catalog; Administrator adds items to cart.   
3. Administrator sets shipping/payment method and submits.   
  
\*B: Edit Order Items\*   
1. At step 5b, Administrator modifies product quantities.   
2. System validates real-time inventory for new quantities.   
3. On validation failure, highlights items with stock shortages.   
  
\*C: Delete Restricted Order\*   
1. At step 4c, system detects non-deletable status (e.g., "Shipped").   
2. System blocks deletion and displays status-based error.   
3. Administrator updates status first via UC-15.   
  
\*D: Customer/Product Not Found\*   
1. During creation/edit, referenced customer/product is invalid.   
2. System displays "Invalid reference" error with ID.   
3. Administrator corrects references or cancels operation.   
  
---  
  
> 🔍 \*Note: Unified workflow for order lifecycle management. Excludes: Payment processing (UC-22), status-only updates (UC-15), and bulk operations. Aligns with Order, Customer, and Product entities.\*  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage System Settings   
\*\*Use Case ID:\*\* UC-32   
\*\*Actors:\*\* Administrator (System-level privileges)   
\*\*Preconditions:\*\*   
1. Administrator is authenticated in the system.   
2. System settings module is accessible and operational.   
3. Administrator has "System Configuration" privileges.   
  
\*\*Postconditions:\*\*   
1. System configuration parameters are updated and persisted.   
2. Audit log records all setting modifications.   
3. Dependent services reload configurations dynamically.   
  
\*\*Main Flow:\*\*   
1. Administrator selects "System Settings" in control panel.   
2. System displays configuration categories (e.g., Payment, Notifications, Security).   
3. Administrator selects target category.   
4. System displays current settings with editable fields (e.g., tax rate, session timeout).   
5. Administrator modifies values and confirms changes.   
6. System validates input formats and value ranges.   
7. System persists updated settings to configuration repository.   
8. System reloads dependent services with new configurations.   
9. System displays "Settings updated successfully".   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Configuration Value\*   
1. At step 6, validation fails (e.g., negative tax rate).   
2. System highlights invalid fields with allowed range.   
3. Administrator corrects values and resubmits.   
  
\*B: Settings Rollback\*   
1. After step 7, critical service fails to initialize.   
2. System auto-reverts to last stable configuration.   
3. Administrator notified via system alert with error details.   
  
\*C: Permission Denied\*   
1. At step 1, Administrator lacks required privileges.   
2. System blocks access and displays "Insufficient permissions".  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage ShoppingCartItem   
\*\*Use Case ID:\*\* UC-33   
\*\*Actors:\*\* Customer   
\*\*Preconditions:\*\*   
1. Customer has an active session with initialized shopping cart.   
2. Target ShoppingCartItem exists in the cart.   
3. Product repository is accessible for real-time validation.   
  
\*\*Postconditions:\*\*   
1. ShoppingCartItem quantity is updated or item is removed from cart.   
2. Cart totals and item counts are recalculated.   
3. Changes are persisted (saved cart for authenticated users; session storage for guests).   
  
\*\*Main Flow:\*\*   
1. Customer navigates to cart summary page.   
2. System displays cart contents with item-level action controls.   
3. Customer selects target ShoppingCartItem and chooses "Edit Quantity" or "Remove".   
4. For quantity update:   
 a. Customer enters new positive integer quantity.   
 b. System validates quantity ≤ available stock.   
 c. System updates ShoppingCartItem quantity.   
5. For removal:   
 a. System removes ShoppingCartItem from cart.   
6. System recalculates cart totals and item counts.   
7. System displays updated cart with "Cart updated" confirmation.   
  
\*\*Alternative Flow:\*\*   
\*A: Invalid Quantity Entry\*   
1. At step 4b, quantity exceeds stock limit.   
2. System auto-sets quantity to maximum available stock.   
3. Displays "[Product] quantity adjusted to [max]" notification.   
  
\*B: Product Unavailable During Edit\*   
1. At step 4b, system detects product deactivation/out-of-stock.   
2. System automatically removes item with "[Product] discontinued" notice.   
3. Recalculates cart totals excluding removed item.   
  
\*C: Concurrent Modification Conflict\*   
1. During step 4-6, system detects cart changes from another session.   
2. System displays "Your cart has changed" warning with refreshed data.   
3. Customer reapplies modifications to updated cart state.   
  
---  
  
### Use Case Description   
\*\*Use Case Name:\*\* Manage OrderItem   
\*\*Use Case ID:\*\* UC-34   
\*\*Actors:\*\* Administrator   
\*\*Preconditions:\*\*   
1. Administrator is authenticated with order management privileges.   
2. Target order exists with status "Processing" or "Pending Shipment".   
3. OrderItem exists in the target order.   
4. Product repository and inventory system are accessible.   
  
\*\*Postconditions:\*\*   
1. OrderItem quantity is modified or item is removed from order.   
2. Order totals are recalculated and inventory levels adjusted.   
3. Order audit log records item-level changes.   
  
\*\*Main Flow:\*\*   
1. Administrator accesses target order via "Manage Orders" interface.   
2. System displays order details with item-level edit options.   
3. Administrator selects target OrderItem and chooses "Modify Quantity" or "Remove Item".   
4. For quantity modification:   
 a. Administrator enters new positive integer quantity.   
 b. System validates inventory availability for new quantity.   
 c. System updates OrderItem quantity and adjusts reserved inventory.   
5. For removal:   
 a. System deletes OrderItem from order and releases reserved inventory.   
6. System recalculates order subtotal, taxes, and grand total.   
7. System displays "Order items updated" confirmation with revised totals.   
  
\*\*Alternative Flow:\*\*   
\*A: Insufficient Inventory for Quantity Increase\*   
1. At step 4b, new quantity exceeds available stock.   
2. System displays "Only [max] units available" alert with stock count.   
3. Administrator adjusts quantity or cancels modification.   
  
\*B: Post-Modification Price Discrepancy\*   
1. At step 6, system detects product price change since order creation.   
2. System applies current price to modified OrderItem with change notice.   
3. Audit log records price override with original vs. current values.   
  
\*C: Invalid Order Status for Modification\*   
1. At step 2, order status prohibits changes (e.g., "Shipped").   
2. System blocks edits and displays "Order status: [status] - modifications disabled".   
3. Administrator updates order status via UC-15 before retrying.   
  
---