**Advanced Systems Project**

**Use Cases:**

# **Use Case List**

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Primary Actor** | **Use Cases** |
| 001 | User / Admin | Login and Logout |
| 002 | User | Browse Products |
| 003 | User | Add to Cart |
| 004 | User | Apply Promo Code |
| 005 | User | Place an Order |
| 006 | User | Track Orders |
| 007 | User | Cancel Order |
| 008 | Admin | Manage Products |
| 009 | System | Calculate Grand Total |
| 010 | System | Determine Delivery Fee |
| 011 | System | Validate Promo Code |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 001 | | | |
| **Use Case Name:** | Login and Logout | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User /Admin | | |
| **Description:** | | This use case allows users to securely log in to their account and log out once they have completed their session. | | |
| **Trigger:** | | User opens the application or website and wishes to access their account. | | |
| **Preconditions:** | | The user must have a valid username and password. | | |
| **Postconditions:** | | User is authenticated and gains access to their account. After logging out, the session is terminated. | | |
| **Normal Flow:** | | * User navigates to the login page. * User enters the username and password. * System validates the credentials. * Upon successful authentication, the user is granted access to their account. * Once done, the user logs out to terminate the session. | | |
| **Alternative Flows:** | | Invalid login credentials: If the credentials don't match, the user is shown an error message and prompted to try again. | | |
| **Exceptions:** | | System downtime: If the login service is unavailable, the user is informed and advised to try later. | | |
| **Includes:** | | Authentication | | |
| **Frequency of Use:** | | Multiple times per day based on user needs. | | |
| **Special Requirements:** | | * Strong password policies. * Multi-factor authentication (if applicable). | | |
| **Assumptions:** | | The user has an active account. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 002 | | | |
| **Use Case Name:** | Browse Products | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | This use case allows the user to browse and view available products. | | |
| **Trigger:** | | User accesses the product listing page. | | |
| **Preconditions:** | | Products must be listed in the system's catalog. | | |
| **Postconditions:** | | The user can view the product details and decide whether to make a purchase | | |
| **Normal Flow:** | | * User navigates to the product catalog. * User clicks on a product to see its details. * System displays product information (description, price, images). | | |
| **Alternative Flows:** | | No products available: The system shows a "no products available" message. | | |
| **Exceptions:** | | Product data loading error: If there’s an issue fetching product details, the user is shown an error message. | | |
| **Includes:** | | Product catalog display | | |
| **Frequency of Use:** | | As frequently as users need to browse products. | | |
| **Special Requirements:** | | * High-quality product images. * Product filtering and search capabilities. | | |
| **Assumptions:** | | Products are regularly updated in the system. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 003 | | | |
| **Use Case Name:** | Add to Cart | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | This use case allows users to add products to their shopping cart. | | |
| **Trigger:** | | User selects a product and wants to purchase it. | | |
| **Preconditions:** | | The product must be available for sale. | | |
| **Postconditions:** | | The product is added to the shopping cart. | | |
| **Normal Flow:** | | * User selects a product. * User clicks "Add to Cart." * System confirms that the product is added to the cart. * User can view their cart to review the selected items. | | |
| **Alternative Flows:** | | Out of stock: The system notifies the user if the product is out of stock and prevents adding to the cart | | |
| **Exceptions:** | | Product addition error: If there’s an issue adding the product, the user is shown an error message. | | |
| **Includes:** | | Shopping cart management | | |
| **Frequency of Use:** | | Occasional, based on user activity. | | |
| **Special Requirements:** | | Real-time stock level updates. | | |
| **Assumptions:** | | Products are in stock and available for sale. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 004 | | | |
| **Use Case Name:** | Apply Promo code | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User: The customer who is shopping and wishes to apply a | | |
| **Description:** | | This use case allows the user to apply a valid promo code to their shopping cart to receive a discount. The system will verify if the code is applicable, calculate the discount, and update the total price accordingly. | | |
| **Trigger:** | | User reaches the checkout page and opts to apply a promo | | |
| **Preconditions:** | | * The user must have items in the cart and must be at the checkout stage. * A valid promo code must be available for use. * The user is authenticated (if required by the platform). | | |
| **Postconditions:** | | * The cart's total price is updated, including the discount from the promo code. * If the promo code is invalid, the user is notified and the total price remains unchanged. | | |
| **Normal Flow:** | | 1. User navigates to checkout page. 2. User enters a promo code in the designated field. 3. System validates the promo code:    * The system checks if the promo code is valid, active, and applicable to the user’s current cart. 4. System calculates the discount:    * If valid, the system computes the discount based on predefined logic (e.g., percentage discount, fixed amount off, etc.). 5. System updates the total price:    * The new total price (after applying the discount) is displayed to the user. 6. User can proceed to finalize the purchase. | | |
| **Alternative Flows:** | | * Invalid promo code: If the promo code entered is invalid or expired, the system displays an error message and allows the user to try again. * Promo code does not apply: If the promo code does not meet the criteria (e.g., minimum purchase amount), the system will notify the user and keep the price unchanged | | |
| **Exceptions:** | | * Invalid input format: If the user enters an invalid promo code format, the system will reject the input and prompt the user to enter a valid code. * Expired promo code: If the code has expired, the system will notify the user and prevent it from being applied. | | |
| **Includes:** | | Validate Promo Code: This sub-process is used to check if the promo code is valid, expired, or applicable to the user’s cart. | | |
| **Frequency of Use:** | | Frequently used during checkout. This use case may be invoked many times during sales or promotional periods. | | |
| **Special Requirements:** | | * The system must support different types of promo codes, such as percentage-based, fixed discounts, or specific item-based offers. * The system needs to handle various promotional rules, such as minimum purchase thresholds or applicable categories. | | |
| **Assumptions:** | | * The user has a valid promo code. * The system's promo code database is regularly updated to reflect active and valid codes | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 004 | | | |
| **Use Case Name:** | Place an Order | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | This use case allows the user to finalize and place an order. | | |
| **Trigger:** | | The user decides to purchase items from their shopping cart. | | |
| **Preconditions:** | | Items must be added to the cart. | | |
| **Postconditions:** | | The order is placed successfully. | | |
| **Normal Flow:** | | * User reviews the cart. * User proceeds to checkout. * User enters shipping and payment information. * System processes the payment and confirms the order. * User receives an order confirmation. | | |
| **Alternative Flows:** | | Payment failure: The system notifies the user if the payment cannot be processed. | | |
| **Exceptions:** | | Order processing error: The user is informed if there is an issue with placing the order. | | |
| **Includes:** | | Payment processing | | |
| **Frequency of Use:** | | Varies depending on user activity. | | |
| **Special Requirements:** | | Secure payment gateway integration. | | |
| **Assumptions:** | | The user has a valid payment method. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 005 | | | |
| **Use Case Name:** | Track Orders | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | This use case allows users to monitor the status and location of their orders. | | |
| **Trigger:** | | User wants to check the status of a placed order. | | |
| **Preconditions:** | | The user has placed an order. | | |
| **Postconditions:** | | The user can view the order status and estimated delivery time. | | |
| **Normal Flow:** | | * User logs into their account. * User navigates to the "Orders" section. * User selects the desired order to track. * System displays the current status and tracking details. | | |
| **Alternative Flows:** | | No tracking information: The system informs the user if tracking data is not available yet. | | |
| **Exceptions:** | | System error: If the tracking system is down, the user is notified to check back later. | | |
| **Includes:** | | Order tracking | | |
| **Frequency of Use:** | | As often as the user wants to check order status. | | |
| **Special Requirements:** | | Integration with delivery partners. | | |
| **Assumptions:** | | The order has been processed and shipped. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 006 | | | |
| **Use Case Name:** | Cancel Order | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | User | | |
| **Description:** | | This use case allows users to cancel an order before it is fulfilled. | | |
| **Trigger:** | | User decides to cancel a placed order. | | |
| **Preconditions:** | | The order is still in a cancellable state (e.g., not yet shipped). | | |
| **Postconditions:** | | The order is cancelled, and a refund process is initiated (if applicable | | |
| **Normal Flow:** | | * User logs into their account. * User navigates to the "Orders" section. * User selects the order they wish to cancel. * System provides a cancel option. * User confirms the cancellation. * System processes the cancellation and notifies the user. | | |
| **Alternative Flows:** | | Non-cancellable order: The system notifies the user if the order has already been fulfilled and cannot be cancelled. | | |
| **Exceptions:** | | Cancellation failure: The system informs the user if there’s an issue processing the cancellation. | | |
| **Includes:** | | Order management | | |
| **Frequency of Use:** | | Occasional, based on user decision. | | |
| **Special Requirements:** | | Clear communication of cancellation policies. | | |
| **Assumptions:** | | Orders can be cancelled within a certain time frame. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 008 | | | |
| **Use Case Name:** | Manage Products | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | Admin | | |
| **Description:** | | This use case allows the admin to add, edit, or remove products from the catalog. | | |
| **Trigger:** | | Admin needs to update product information in the system. | | |
| **Preconditions:** | | Admin has access to the product catalog. | | |
| **Postconditions:** | | The product catalog is updated with the changes. | | |
| **Normal Flow:** | | * Admin logs into the admin dashboard. * Admin navigates to the "Manage Products" section. * Admin adds, edits, or removes products. * System updates the product catalog accordingly. | | |
| **Alternative Flows:** | | Product update failure: If there's an issue updating the product, the system displays an error message. | | |
| **Exceptions:** | | Unauthorized action: Admins without proper permissions cannot perform product updates. | | |
| **Includes:** | | Product management | | |
| **Frequency of Use:** | | Frequently, based on business needs. | | |
| **Special Requirements:** | | Efficient product search and filter options for admin. | | |
| **Assumptions:** | | The admin has the necessary permissions. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 009 | | | |
| **Use Case Name:** | Calculate Grand Total | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | System | | |
| **Description:** | | This use case calculates the grand total of all items in the shopping cart, considering discounts, taxes, and additional charges. | | |
| **Trigger:** | | User adds products to the cart and proceeds to checkout. | | |
| **Preconditions:** | | The user must have items in their cart and proceed to checkout. | | |
| **Postconditions:** | | The system computes and displays the final total cost, including any adjustments. | | |
| **Normal Flow:** | | 1. User proceeds to checkout. 2. System retrieves cart items and their prices. 3. System applies discounts and taxes. 4. System calculates and displays the grand total. | | |
| **Alternative Flows:** | | Invalid discount: System notifies the user if a discount is invalid and reverts to the original price. | | |
| **Exceptions:** | | Calculation error: System notifies the user if there is an error in calculating prices. | | |
| **Includes:** | | Price calculation, tax and discount application | | |
| **Frequency of Use:** | | Frequently during checkout | | |
| **Special Requirements:** | | Real-time updates of prices based on cart and location. Accurate application of taxes and discounts. | | |
| **Assumptions:** | | The system has access to updated product prices and taxes. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 010 | | | |
| **Use Case Name:** | Determine Delivery Fee | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | System | | |
| **Description:** | | This use case calculates the shipping fee based on the delivery address, item weight, and shipping method selected. | | |
| **Trigger:** | | User proceeds to checkout and selects a delivery option. | | |
| **Preconditions:** | | The user must provide a shipping address. | | |
| **Postconditions:** | | The system computes and displays the delivery fee. | | |
| **Normal Flow:** | | 1. User enters the delivery address and selects a shipping method. 2. System calculates the shipping cost based on the distance, weight, and selected method. 3. System displays the calculated fee | | |
| **Alternative Flows:** | | Free shipping promotion: System applies free shipping if the user qualifies. | | |
| **Exceptions:** | | Invalid address: If the address is incorrect, the system prompts the user to correct it. | | |
| **Includes:** | | Shipping fee calculation | | |
| **Frequency of Use:** | | Frequently during checkout | | |
| **Special Requirements:** | | Delivery fee based on region, weight, and shipping method. | | |
| **Assumptions:** | | The system has current shipping rates and delivery information. | | |

## Use Case(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID:** | 011 | | | |
| **Use Case Name:** | Validate Promo Code | | | |
| **Created By:** | Seethu Maila | | **Last Updated By:** |  |
| **Date Created:** |  | | **Last Revision Date:** |  |
| **Actors:** | | System | | |
| **Description:** | | This use case validates the promo code entered by the user, checking if it's active, meets conditions, and applies the discount. | | |
| **Trigger:** | | User enters a promo code during checkout. | | |
| **Preconditions:** | | The user must have items in the cart and be logged in (if required). | | |
| **Postconditions:** | | The system confirms the promo code is valid and applies the discount to the cart. | | |
| **Normal Flow:** | | 1. User enters a promo code. 2. System checks if the code is valid, active, and applicable. 3. System calculates and applies the discount to the total price. 4. User can proceed to complete the purchase. | | |
| **Alternative Flows:** | | Invalid code: If the code is expired or doesn’t meet conditions, the system displays an error. | | |
| **Exceptions:** | | Invalid format: System prompts the user to enter a valid code. | | |
| **Includes:** | | Promo code validation, discount calculation | | |
| **Frequency of Use:** | | Frequently during sales or promotions | | |
| **Special Requirements:** | | System supports multiple types of promo codes (e.g., percentage or fixed discounts). | | |
| **Assumptions:** | | The promo code database is up-to-date with active, valid codes. | | |

**Unit test script:**

**Unit Name**: ShopeEase User Operations  
**Module Name**: E-Commerce Application (Frontend)  
**Use case**: **User Login and Logout (UC-1)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step No.** | **Description** | **Test data (if any)** | **Expected result** | **Observed result** | **Pass / Fail** |
| 1 | User should be able to access the login page from the homepage | None | Navigation to login page should be successful | Navigation successful | Y |
| 2 | User must enter valid credentials and submit the form | username: user1, password: pass123 | User should be authenticated and redirected to the homepage | Login successful | Y |
| 3 | Test invalid credentials | username: user1, password: wrongpass | An error message should be displayed stating invalid login credentials | Error message displayed ("Invalid credentials, please try again") | Y |
| 4 | Test logout functionality | None | User should be logged out and redirected to the login page | Logout successful | Y |

**Unit Name**: ShopeEase User Operations  
**Module Name**: E-Commerce Application (Frontend)  
**Use case**: **Browse Products (UC-2)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step No.** | **Description** | **Test data (if any)** | **Expected result** | **Observed result** | **Pass / Fail** |
| 1 | User should be able to navigate to the product catalog | None | Navigation to product listing page should be successful | Navigation successful | Y |
| 2 | User should be able to view product details by clicking on a product | Product ID: 123 | Product details should display correctly | Product details displayed | Y |

**Unit Name**: ShopeEase User Operations  
**Module Name**: E-Commerce Application (Frontend)  
**Use case**: **Add to Cart (UC-3)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step No.** | **Description** | **Test data (if any)** | **Expected result** | **Observed result** | **Pass / Fail** |
|  |  |  |  |  |  |
| 1 | User should be able to select a product to add to the shopping cart | Product ID: 123 | Product should be added to the cart successfully | Product added successfully | Y |
| 2 | Verify shopping cart contents after adding a product | None | Shopping cart should show the correct product and quantity | Cart shows correct product | Y |

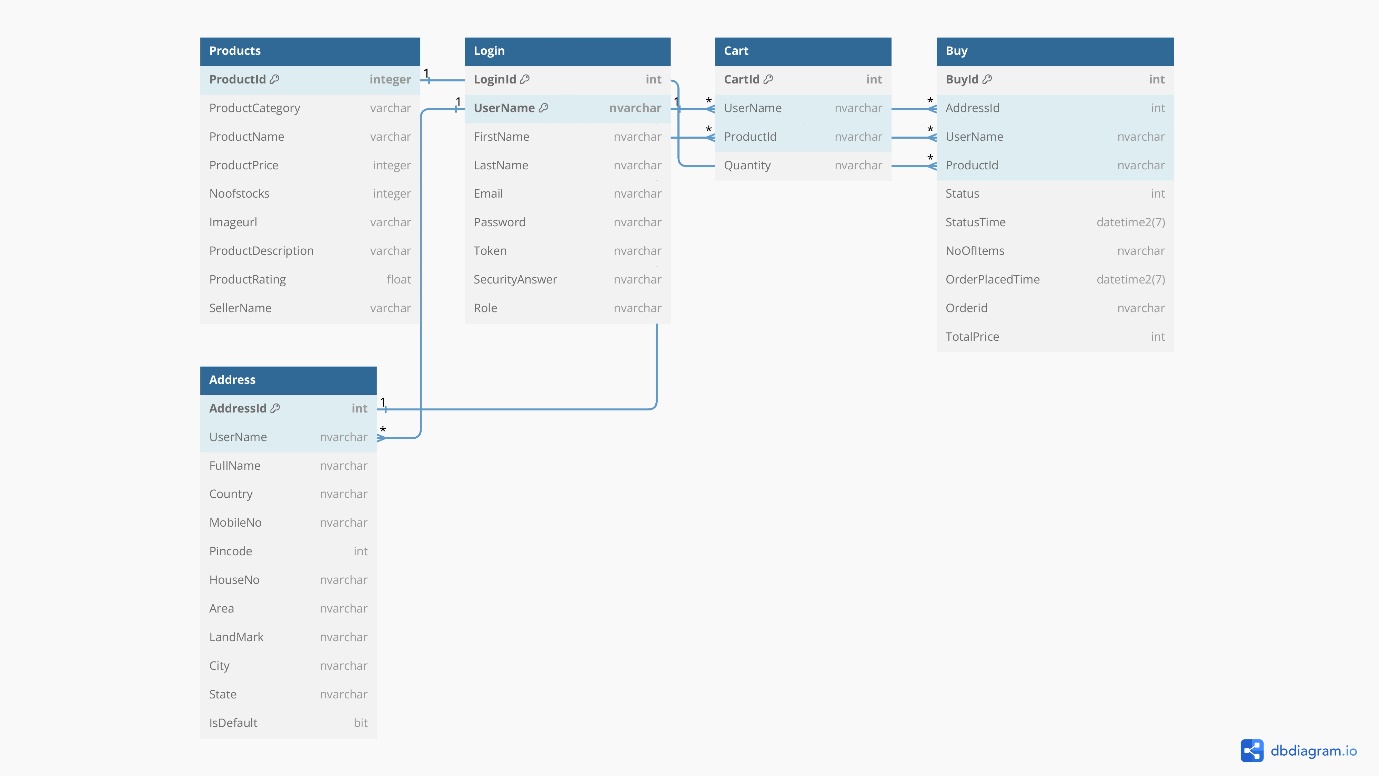
**Unit Name**: ShopeEase User Operations  
**Module Name**: E-Commerce Application (Frontend)  
**Use case**: **Place an Order (UC-4)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step No.** | **Description** | **Test data (if any)** | **Expected result** | **Observed result** | **Pass / Fail** |
| 1 | User should be able to view cart and proceed to checkout | None | User navigates to checkout page | Navigation successful | Y |
| 2 | User enters delivery and payment information | Name, Address, Card No | User should be able to input valid delivery and payment details | Input successful | Y |
| 3 | User clicks on "Place Order" to complete the purchase | None | Order should be placed successfully, and confirmation message displayed | Order placed successfully | Y |

**Unit Name**: ShopeEase Admin Operations  
**Module Name**: E-Commerce Application (Admin Panel)  
**Use case**: **Manage Products (UC-5)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step No.** | **Description** | **Test data (if any)** | **Expected result** | **Observed result** | **Pass / Fail** |
| 1 | Admin should be able to log in with admin credentials | username: admin, password: adminpass | Admin should access the admin dashboard | Admin login successful | Y |
| 2 | Admin should be able to add a new product to the product catalog | Product details | Product should be added successfully | Product added successfully | Y |
| 3 | Admin should be able to edit product details | Updated product details | Product details should be updated successfully | Product updated successfully | Y |
| 4 | Admin should be able to delete a product from the catalog | Product ID: 123 | Product should be removed successfully | Product removed successfully | Y |

**Database schema for the ShopeEase:**



The provided e-commerce database schema consists of five main tables (Products, Login, Cart, Buy, and Address) that work together to manage users, products, and transactions. Here’s an alternate explanation of the same schema:

### 1. ****Products Table****:

* **ProductId**: Unique identifier for each product.
* **ProductCategory**: Categorizes the products (e.g., Electronics, Clothing).
* **ProductName**: The name of the product.
* **ProductPrice**: The price of the product.
* **NoOfStocks**: Number of items available in stock.
* **ImageUrl**: Link to the product image.
* **ProductDescription**: Short description of the product.
* **ProductRating**: Average user rating (float value).
* **SellerName**: Name of the seller listing the product.

**Role**: The Products table holds information about all products available for purchase on the platform. It includes details like price, stock, ratings, and a description for users to view.

### 2. ****Login Table****:

* **LoginId**: Unique identifier for each login session.
* **UserName**: Username for user identification; links with other tables.
* **FirstName / LastName**: Personal details of the user.
* **Email**: Email address for communication.
* **Password**: Encrypted password for login authentication.
* **Token**: Session management or security token.
* **SecurityAnswer**: Security question answer for account recovery.
* **Role**: Defines user type (e.g., Admin or Customer).

**Role**: This table manages user authentication and details for both customers and administrators. It ensures secure login and user role management for the platform.

### 3. ****Cart Table****:

* **CartId**: Unique identifier for the cart.
* **UserName**: Links the cart to the specific user (from the Login table).
* **ProductId**: Tracks the product added to the cart (from Products table).
* **Quantity**: Number of units added to the cart for a specific product.

**Role**: This table records all items users add to their shopping cart while browsing. Each user has their own cart, and the selected products are stored with quantities.

### 4. ****Buy Table****:

* **BuyId**: Unique identifier for the purchase transaction.
* **AddressId**: Links the order to the delivery address (from Address table).
* **UserName**: Links the order to the specific user (from Login table).
* **ProductId**: Refers to the product being purchased (from Products table).
* **Status**: Tracks the order status (e.g., Pending, Shipped, Delivered).
* **StatusTime**: Timestamp when the status was updated.
* **NoOfItems**: Number of items purchased.
* **OrderPlacedTime**: Time when the order was placed.
* **OrderId**: A unique identifier to track the order.
* **TotalPrice**: Total cost of the purchase.

**Role**: The Buy table records details of all completed orders. It links the user, their selected address, and the products being purchased, while also tracking the order status and timing.

### 5. ****Address Table****:

* **AddressId**: Unique identifier for the address.
* **UserName**: Links the address to a specific user (from the Login table).
* **FullName**: The recipient's name for shipping purposes.
* **Country, State, City, Pincode, Area, Landmark, HouseNo**: Complete address details for delivery.
* **MobileNo**: User's contact number for communication.
* **IsDefault**: Boolean flag indicating if this is the user’s default address.

**Role**: This table stores delivery addresses associated with users. A user may have multiple addresses, and this information is linked to the Buy table for order shipments.

### How It Works Together:

1. **User Login/Registration**: A user signs up or logs in. Their credentials are stored in the **Login** table, defining whether they are an admin or a regular customer.
2. **Browsing Products**: Users can browse the items listed in the **Products** table. They see details like price, stock, and descriptions.
3. **Adding to Cart**: Users select products and add them to their cart. The **Cart** table tracks the user’s selections and quantities for each product.
4. **Placing an Order**: When users proceed to checkout, the system transfers the cart contents to the **Buy** table. It records purchase information, links the user’s address from the **Address** table, and updates product quantities.
5. **Order Tracking**: Admins and users can track order status (e.g., pending, shipped) through the **Buy** table. The **StatusTime** field keeps a log of when the status changed.

**Design details:**

The detailed **design** based on the use cases for the **ShopeEase E-Commerce Application**:

### 1. Database Schema Design:

#### 1.1. ****Login and Logout (Use Case 001)****

* **Database Tables Involved**: Login, Session
* **Table Design**:
  + **Login Table**:
    - Fields: LoginId, UserName, FirstName, LastName, Email, Password, Role, Token, SecurityAnswer
    - Relationships: Links with other tables like Cart, Orders, and Address via UserName.
  + **Session Table** (for session management):
    - Fields: SessionId, UserId, StartTime, EndTime
    - Function: Track active sessions and logout status.
* **Process**:
* On login, validate credentials against Login table.
  + Generate a Token for session management in the Session table.
  + Upon logout, mark session as ended by updating the EndTime in Session.

#### 1.2. ****Browse Products (Use Case 002)****

* **Database Tables Involved**: Products
* **Table Design**:
  + **Products Table**:
    - Fields: ProductId, ProductCategory, ProductName, ProductPrice, NoOfStocks, ImageUrl, ProductDescription, ProductRating, SellerName
* **Process**:
  + Retrieve product list from the Products table based on the selected category.
  + Display relevant product details such as price, description, and stock availability.

#### 1.3. ****Add to Cart (Use Case 003)****

* **Database Tables Involved**: Cart
* **Table Design**:
  + **Cart Table**:
    - Fields: CartId, UserName, ProductId, Quantity
    - Relationships: Links with Login (via UserName) and Products (via ProductId).
* **Process**:
  + When a user adds a product, insert the ProductId, UserName, and Quantity into the Cart table.
  + Ensure the system checks for stock availability before adding the product.

#### 1.4. ****Place an Order (Use Case 004)****

* **Database Tables Involved**: Buy, Cart, Products, Address
* **Table Design**:
  + **Buy Table**:
    - Fields: BuyId, AddressId, UserName, ProductId, Status, NoOfItems, OrderPlacedTime, OrderId, TotalPrice
  + **Address Table**:
    - Fields: AddressId, UserName, FullName, Country, State, City, Pincode, MobileNo, IsDefault
    - Relationships: Links with Login via UserName.
* **Process**:
  + Retrieve cart items from Cart for the user.
  + Allow the user to input an address from the Address table.
  + Process the payment and update the Buy table with order details.
  + Clear the Cart table after a successful transaction.

#### 1.5. ****Track Orders (Use Case 005)****

* **Database Tables Involved**: Buy, Products
* **Table Design**:
  + **Buy Table**:
    - Fields: BuyId, OrderId, Status, StatusTime, UserName
* **Process**:
  + Retrieve order details from Buy based on the user’s UserName.
  + Provide real-time updates on the order’s Status (e.g., Pending, Shipped, Delivered).

#### 1.6. ****Cancel Order (Use Case 006)****

* **Database Tables Involved**: Buy
* **Table Design**:
  + **Buy Table**:
    - Fields: BuyId, OrderId, Status, UserName, OrderPlacedTime
* **Process**:
  + If the order status is Pending, allow the user to cancel the order by updating the Status field.
  + Update the product stock in Products to reflect the cancellation.

#### 1.7. ****Admin Login (Use Case 007)****

* **Database Tables Involved**: Login
* **Table Design**:
  + **Login Table**:
    - Fields: LoginId, UserName, Password, Role
* **Process**:
  + Admin credentials are validated using the Login table, checking for the Role as Admin.
  + Admin dashboard is only accessible after successful authentication as Admin.

#### 1.8. ****Manage Products (Use Case 008)****

* **Database Tables Involved**: Products
* **Table Design**:
  + **Products Table**:
    - Fields: ProductId, ProductCategory, ProductName, ProductPrice, NoOfStocks, ProductDescription, ProductRating, SellerName
* **Process**:
  + Admin can perform INSERT, UPDATE, or DELETE operations on the Products table.
  + Product information updates in real-time for users to browse.

#### 1.9. ****View Orders (Use Case 009)****

* **Database Tables Involved**: Buy
* **Table Design**:
  + **Buy Table**:
    - Fields: BuyId, OrderId, UserName, TotalPrice, Status, StatusTime
* **Process**:
  + Admin views orders placed by users.
  + Admin can filter and sort orders based on Status and UserName.

#### 1.10. ****Update Delivery Status (Use Case 010)****

* **Database Tables Involved**: Buy
* **Table Design**:
  + **Buy Table**:
    - Fields: BuyId, OrderId, Status, StatusTime
* **Process**:
  + Admin updates the delivery status in the Buy table.
  + Users can view updated statuses when tracking their orders.

### 2. System Design Considerations:

* **Session Management**: The session token in the Login and Session tables ensures secure user authentication and timeout policies.
* **Security Measures**:
  + Passwords are stored securely (hashed) in the Login table.
  + Multi-factor authentication for admins and users is recommended, enhancing security.
  + Secure Payment Gateway: Payments are processed through a secure API, and transaction details are logged in the Buy table.
* **Real-Time Updates**: For inventory and orders, real-time updates are managed through triggers or background services, ensuring consistent data between users, products, and admins.