**NAME: SULTAN**

**SAP ID : 56189**

**DSA SEMESTER**

**PROJECT**

**E-COMMERCE SYSTEM FOR**

**A MOBILE SHOP**

**PROJECT DOCUMENTATION:**

**Documentation for Mobile Shop Management System**

**Mobile Shop Management System**

**Overview**

The Mobile Shop Management System provides separate functionalities for administrators (owners) and customers. It uses linked list data structures for handling mobile inventories and shopping carts dynamically.

**Features**

**1. Admin Features**

* **Add Mobile**  
  Admins can add new mobiles to the catalog by specifying the name and price. Duplicate entries are not allowed.
* **Remove Mobile**  
  Admins can delete mobiles from the catalog by providing the name.
* **Update Mobile Price**  
  Modify the price of a mobile in the catalog.
* **Search Mobile**  
  Search for a specific mobile in the catalog and view its details.
* **View All Mobiles**  
  Display a complete list of mobiles available in the shop.

**2. Customer Features**

* **View All Mobiles**  
  Customers can browse the catalog to see all available mobiles and their prices.
* **Search Mobile**  
  Customers can search for specific mobiles by name.
* **Add to Cart**  
  Add selected mobiles to the cart with a specified quantity.
* **View Cart**  
  Display all items in the shopping cart along with the total price.
* **Remove from Cart**  
  Remove a specific item from the cart. Updates the cart total automatically.

**3. Menu System**

* **Admin Menu**  
  Accessed via secure login (username: admin, password: 1234). Allows full control over the catalog management.
* **Customer Menu**  
  Provides options to view mobiles, add items to the cart, and manage the cart.
* **Main Menu**  
  Offers a gateway for selecting between admin and customer operations or exiting the system.

**Class Structure**

**1. Mobile Class**

* Represents a mobile phone in the catalog.
* **Attributes**:
  + name: The name of the mobile.
  + price: The price of the mobile.
  + next: Pointer to the next mobile (for doubly linked list).
  + previous: Pointer to the previous mobile.
* **Methods**:
  + Getters and setters for all attributes.

**2. Cart Class**

* Represents an item in the shopping cart.
* **Attributes**:
  + item: Pointer to a Mobile object.
  + quantity: Quantity of the mobile in the cart.
  + next: Pointer to the next cart item (single-linked list).
* **Methods**:
  + Getters and setters for all attributes.

**3. Implementation Class**

* Manages the system's core functionality for both admin and customer operations.
* **Attributes**:
  + head: Points to the head of the mobile list.
  + cartHead: Points to the head of the shopping cart.
  + total: Tracks the total price of items in the cart.
* **Methods**:
  + **Admin Operations**:
    - Add, remove, update, and search mobiles.
    - View all mobiles.
  + **Customer Operations**:
    - View mobiles, add to cart, view cart, and remove from the cart.
  + **Menu Navigation**:
    - Separate menus for admin and customers.

**System Workflow**

**1. Admin Workflow**

1. Log in with admin credentials (username: admin, password: 1234).
2. Perform any of the following operations:
   * Add mobiles to the catalog.
   * Remove mobiles by name.
   * Update the price of an existing mobile.
   * Search for mobiles by name.
   * View all available mobiles.
3. Exit the admin menu after completing tasks.

**2. Customer Workflow**

1. Select the customer menu from the main menu.
2. Browse the catalog or search for a specific mobile.
3. Add desired mobiles to the cart by specifying the quantity.
4. View the cart to see the selected items and total price.
5. Remove items from the cart if needed.
6. Exit the customer menu.

**Sample Code Usage**

**Main Menu**

int main() {

Implementation imp;

int choice;

do {

cout << " Mobile Shop \n";

cout << " 1. Owner \n";

cout << " 2. Customer \n";

cout << " 3. Exit \n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

imp.LogIn();

break;

case 2:

imp.Customer\_Menu();

break;

case 3:

cout << "Exiting the shop.\n";

break;

default:

cout << "Invalid choice.\n";

break;

}

} while (choice != 3);

return 0;

}

**Key Advantages**

1. **Dynamic Data Management**  
   Linked lists dynamically handle inventory and cart operations without size limitations.
2. **Encapsulation and Reusability**  
   Each class has a focused responsibility, making the code modular and easier to maintain.
3. **User-Friendly Menus**  
   Separate menus for admins and customers make the system intuitive.
4. **Scalable Architecture**  
   The code can be easily extended to include more features, such as payment integration or data persistence.

**Future Enhancements**

* **Data Persistence**: Save the catalog and cart data to files for persistence across sessions.
* **Enhanced Search**: Use hash tables or maps for faster searches.
* **Graphical User Interface**: Replace the console interface with a GUI for better user experience.
* **Advanced Authentication**: Add roles and permissions for multiple users.

**Conclusion**

The Mobile Shop Management System is a robust and scalable solution for managing a mobile shop's inventory and customer orders. Its object-oriented design ensures maintainability, and the linked list structure efficiently handles dynamic data. The system is suitable for small to medium-scale retail operations and provides a foundation for further development.