import java.util.Scanner;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Map;

abstract class Product {

protected int id;

protected String name;

protected double price;

public Product(int id, String name, double price) {

this.id = id;

this.name = name;

this.price = price;

}

public int getId() {

return id;

}

public String getName() {

return name;

}

public double getPrice() {

return price;

}

public abstract void displayInfo();

}

class Electronics extends Product {

private String brand;

public Electronics(int id, String name, double price, String brand) {

super(id, name, price);

this.brand = brand;

}

@Override

public void displayInfo() {

System.out.println("Electronics - " + name + " | Brand: " + brand + " | Price: $" + price);

}

}

class Clothing extends Product {

private String size;

public Clothing(int id, String name, double price, String size) {

super(id, name, price);

this.size = size;

}

@Override

public void displayInfo() {

System.out.println("Clothing - " + name + " | Size: " + size + " | Price: $" + price);

}

}

class Furniture extends Product {

private String material;

public Furniture(int id, String name, double price, String material) {

super(id, name, price);

this.material = material;

}

@Override

public void displayInfo() {

System.out.println("Furniture - " + name + " | Material: " + material + " | Price: $" + price);

}

}

class Customer {

private String username; private String password; private Cart cart;

public Customer(String username, String password) {

this.username = username;

this.password = password;

this.cart = new Cart();

}

public String getUsername() {

return username;

}

public String getPassword() {

return password;

}

public Cart getCart() {

return cart;

}

}

class Cart {

private ArrayList<Product> items = new ArrayList<>();

public void addItem(Product p) {

items.add(p);

System.out.println(p.getName() + " added to cart.");

}

public void removeItem(int productId) {

items.removeIf(p -> p.getId() == productId);

System.out.println("Item with ID " + productId + " removed.");

}

public void displayCart() {

if (items.isEmpty()) {

System.out.println("Cart is empty.");

return;

}

System.out.println("\n--- Cart Items ---");

for (Product p : items) p.displayInfo();

}

public double calculateTotal() {

double total = 0;

for (Product p : items) total += p.getPrice();

return total;

}

public ArrayList<Product> getItems() {

return items;

}

public void clearCart() {

items.clear();

}

}

interface PaymentMethod {

void pay(double amount);

}

class CreditCardPayment implements PaymentMethod {

public void pay(double amount) {

System.out.println("Paid $" + amount + " using Credit Card.");

}

}

class PayPalPayment implements PaymentMethod {

public void pay(double amount) {

System.out.println("Paid $" + amount + " using PayPal.");

}

}

class Order {

private static int counter = 1; private int orderId; private ArrayList<Product> items; private double total;

public Order(ArrayList<Product> items, double total) {

this.orderId = counter++;

this.items = new ArrayList<>(items);

this.total = total;

}

public void displayOrder() {

System.out.println("\n--- Order ID: " + orderId + " ---");

for (Product p : items) p.displayInfo();

System.out.println("Total: $" + total);

}

}

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

Map<String, Customer> users = new HashMap<>();

Customer currentCustomer = null;

while (currentCustomer == null) {

System.out.println("\n==== Welcome to Online Shopping ====");

System.out.println("1. Register");

System.out.println("2. Login");

System.out.print("Choose: ");

int choice = Integer.parseInt(sc.nextLine());

if (choice == 1) {

System.out.print("Choose username: ");

String uname = sc.nextLine();

if (users.containsKey(uname)) {

System.out.println("Username already exists!");

continue;

}

System.out.print("Choose password: ");

String pwd = sc.nextLine();

users.put(uname, new Customer(uname, pwd));

System.out.println("Registration successful!");

} else if (choice == 2) {

System.out.print("Enter username: ");

String uname = sc.nextLine();

System.out.print("Enter password: ");

String pwd = sc.nextLine();

Customer user = users.get(uname);

if (user != null && user.getPassword().equals(pwd)) {

System.out.println("Login successful. Welcome " + uname + "!");

currentCustomer = user;

} else {

System.out.println("Invalid username or password!");

}

} else {

System.out.println("Invalid choice.");

}

}

ArrayList<Product> store = new ArrayList<>();

store.add(new Electronics(1, "Laptop", 1200.0, "Dell"));

store.add(new Electronics(2, "Smartphone", 800.0, "Samsung"));

store.add(new Clothing(3, "T-Shirt", 25.0, "M"));

store.add(new Clothing(4, "Jeans", 50.0, "L"));

store.add(new Furniture(5, "Chair", 70.0, "Wood"));

store.add(new Furniture(6, "Table", 150.0, "Metal"));

ArrayList<Order> orderHistory = new ArrayList<>();

while (true) {

System.out.println("\n==== SHOPPING MENU ====");

System.out.println("1. View Products");

System.out.println("2. Add to Cart");

System.out.println("3. View Cart");

System.out.println("4. Remove from Cart");

System.out.println("5. Checkout");

System.out.println("6. View Order History");

System.out.println("7. Exit");

System.out.print("Enter choice: ");

int choice = Integer.parseInt(sc.nextLine());

switch (choice) {

case 1:

for (Product p : store) p.displayInfo();

break;

case 2:

System.out.print("Enter Product ID to add: ");

int addId = Integer.parseInt(sc.nextLine());

for (Product p : store) {

if (p.getId() == addId) {

currentCustomer.getCart().addItem(p);

break;

}

}

break;

case 3:

currentCustomer.getCart().displayCart();

break;

case 4:

System.out.print("Enter Product ID to remove: ");

int remId = Integer.parseInt(sc.nextLine());

currentCustomer.getCart().removeItem(remId);

break;

case 5:

double total = currentCustomer.getCart().calculateTotal();

if (total == 0) {

System.out.println("Cart is empty. Cannot checkout.");

break;

}

System.out.println("Total amount: $" + total);

System.out.println("Choose payment method: 1. Credit Card 2. PayPal");

int payChoice = Integer.parseInt(sc.nextLine());

PaymentMethod payment = (payChoice == 1) ? new CreditCardPayment() : new PayPalPayment();

payment.pay(total);

Order order = new Order(currentCustomer.getCart().getItems(), total);

orderHistory.add(order);

currentCustomer.getCart().clearCart();

System.out.println("Order placed successfully!");

break;

case 6:

if (orderHistory.isEmpty()) {

System.out.println("No orders found.");

} else {

for (Order o : orderHistory) o.displayOrder();

}

break;

case 7:

System.out.println("Thank you for shopping. Goodbye!");

return;

default:

System.out.println("Invalid choice.");

}

}

}

}