

## CASE STUDY: Online Shopping Cart System

M.Polireddy

192325056

### 1.Implement the Product Class

Define the 'Product' class with attributes and methods to handle product details and stock quantity updates.

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import java.time.LocalDateTime;
```

```
public class Product {
```

```
    private String productId;
```

```
    private String name;
```

```
    private double price;
```

```
    private int stockQuantity;
```

```
    public Product(String productId, String name, double price, int  
stockQuantity) {
```

```
        this.productId = productId;
```

```
        this.name = name;
```

```
        this.price = price;
```

```
        this.stockQuantity = stockQuantity;
```

```
    }
```

```
public String getProductId() {  
    return productId;  
}
```

```
public String getName() {  
    return name;  
}
```

```
public double getPrice() {  
    return price;  
}
```

```
public int getStockQuantity() {  
    return stockQuantity;  
}
```

```
public void updateStockQuantity(int quantity) {  
    this.stockQuantity += quantity;  
}  
}
```

## 2. Implement the Customer Class

Define the Customer class with attributes and methods to manage the shopping cart.

```
public class Customer {  
    private String customerId;
```

```
private String name;  
private String email;  
private List<Product> cart;
```

```
public Customer(String customerId, String name, String email) {  
    this.customerId = customerId;  
    this.name = name;  
    this.email = email;  
    this.cart = new ArrayList<>();  
}
```

```
public String getCustomerId() {  
    return customerId;  
}
```

```
public String getName() {  
    return name;  
}
```

```
public String getEmail() {  
    return email;  
}
```

```
public void addToCart(Product product) {  
    cart.add(product);  
}
```

```
}
```

```
public void removeFromCart(Product product) {  
    cart.remove(product);  
}
```

```
public void viewCart() {  
    System.out.println("Cart Contents:");  
    for (Product product : cart) {  
        System.out.println(product.getName() + " - $" +  
product.getPrice());  
    }  
}
```

```
public Order checkout() {  
    double totalAmount = 0;  
    for (Product product : cart) {  
        totalAmount += product.getPrice();  
    }  
    Order order = new Order("ORDER" +  
System.currentTimeMillis(), this, new ArrayList<>(cart),  
totalAmount, LocalDateTime.now());  
    cart.clear();  
    return order;  
}  
}
```

### 3. Implement the Order Class

Define the Order class with attributes and methods to handle order details and calculate the total amount.

```
public class Order {  
    private String orderId;  
    private Customer customer;  
    private List<Product> products;  
    private double totalAmount;  
    private LocalDateTime orderDate;  
  
    public Order(String orderId, Customer customer, List<Product>  
products, double totalAmount, LocalDateTime orderDate) {  
        this.orderId = orderId;  
        this.customer = customer;  
        this.products = products;  
        this.totalAmount = totalAmount;  
        this.orderDate = orderDate;  
    }  
  
    public String getOrderId() {  
        return orderId;  
    }  
  
    public Customer getCustomer() {  
        return customer;  
    }  
}
```

```
}
```

```
public List<Product> getProducts() {  
    return products;  
}
```

```
public double getTotalAmount() {  
    return totalAmount;  
}
```

```
public LocalDateTime getOrderDate() {  
    return orderDate;  
}
```

```
public void calculateTotalAmount() {  
    totalAmount = 0;  
    for (Product product : products) {  
        totalAmount += product.getPrice();  
    }  
}
```

#### 4. Implement the Inventory Class

Define the Inventory class with attributes and methods to manage product inventory.

```
public class Inventory {
```

```
private List<Product> products;
```

```
public Inventory() {  
    this.products = new ArrayList<>();  
}
```

```
public void addProduct(Product product) {  
    products.add(product);  
}
```

```
public Product getProductById(String productId) {  
    for (Product product : products) {  
        if (product.getId().equals(productId)) {  
            return product;  
        }  
    }  
    return null;  
}
```

```
public void updateProductStock(String productId, int quantity) {  
    Product product = getProductById(productId);  
    if (product != null) {  
        product.updateStockQuantity(quantity);  
    }  
}
```

```
}
```

## 5. Develop a Main Class to Test the System

Create instances of Product, Customer, and Inventory. Add products to the inventory, simulate adding products to the customer's cart, view the cart, and checkout.

```
public class Main {  
    public static void main(String[] args) {  
        // Create Inventory  
        Inventory inventory = new Inventory();  
  
        // Create Products  
        Product product1 = new Product("P001", "Laptop", 1000.00,  
10);  
        Product product2 = new Product("P002", "Smartphone", 500.00,  
20);  
  
        // Add Products to Inventory  
        inventory.addProduct(product1);  
        inventory.addProduct(product2);  
  
        // Create Customer  
        Customer customer = new Customer("C001", "John Doe",  
"john.doe@example.com");  
  
        // Add Products to Customer's Cart  
        customer.addToCart(product1);
```



```
customer.addToCart(product2);

// View Cart Contents
customer.viewCart();

// Checkout
Order order = customer.checkout();

// Print Order Details
System.out.println("Order ID: " + order.getId());
System.out.println("Customer: " +
order.getCustomer().getName());
System.out.println("Order Date: " + order.getOrderDate());
System.out.println("Total Amount: $" +
order.getTotalAmount());

// Update Inventory Stock
inventory.updateProductStock("P001", -1);
inventory.updateProductStock("P002", -1);

// Print Updated Stock
System.out.println("Updated Stock for P001: " +
product1.getStockQuantity());
System.out.println("Updated Stock for P002: " +
product2.getStockQuantity());
}
```

```
}
```

## FULL JAVA CODE

```
import java.time.LocalDateTime;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
// Product Class
```

```
class Product {
```

```
    private String productId;
```

```
    private String name;
```

```
    private double price;
```

```
    private int stockQuantity;
```

```
    public Product(String productId, String name, double price, int  
stockQuantity) {
```

```
        this.productId = productId;
```

```
        this.name = name;
```

```
        this.price = price;
```

```
        this.stockQuantity = stockQuantity;
```

```
    }
```

```
    public String getProductId() {
```

```
        return productId;
```

```
    }
```

```
public String getName() {  
    return name;  
}
```

```
public double getPrice() {  
    return price;  
}
```

```
public int getStockQuantity() {  
    return stockQuantity;  
}
```

```
public void updateStockQuantity(int quantity) {  
    this.stockQuantity += quantity;  
}  
}
```

// Customer Class

```
class Customer {  
    private String customerId;  
    private String name;  
    private String email;  
    private List<Product> cart;  
  
    public Customer(String customerId, String name, String email) {
```

```
    this.customerId = customerId;
    this.name = name;
    this.email = email;
    this.cart = new ArrayList<>();
}
```

```
public void addToCart(Product product) {
    cart.add(product);
}
```

```
public void removeFromCart(Product product) {
    cart.remove(product);
}
```

```
public List<Product> viewCart() {
    return cart;
}
```

```
public Order checkout() {
    double totalAmount = 0;
    for (Product product : cart) {
        totalAmount += product.getPrice();
    }

    Order order = new Order("ORD" + System.currentTimeMillis(),
this, new ArrayList<>(cart), totalAmount);
```

```
        cart.clear(); // Clear cart after checkout
        return order;
    }
}
```

// Order Class

```
class Order {
    private String orderId;
    private Customer customer;
    private List<Product> products;
    private double totalAmount;
    private LocalDateTime orderDate;

    public Order(String orderId, Customer customer, List<Product>
products, double totalAmount) {
        this.orderId = orderId;
        this.customer = customer;
        this.products = products;
        this.totalAmount = totalAmount;
        this.orderDate = LocalDateTime.now();
    }

    public String getOrderId() {
        return orderId;
    }
}
```

```
public Customer getCustomer() {  
    return customer;  
}
```

```
public List<Product> getProducts() {  
    return products;  
}
```

```
public double getTotalAmount() {  
    return totalAmount;  
}
```

```
public LocalDateTime getOrderDate() {  
    return orderDate;  
}
```

```
public double calculateTotalAmount() {  
    return totalAmount;  
}  
}
```

// Inventory Class

```
class Inventory {  
    private List<Product> products;
```

```
public Inventory() {  
    this.products = new ArrayList<>();  
}
```

```
public void addProduct(Product product) {  
    products.add(product);  
}
```

```
public Product getProductById(String productId) {  
    for (Product product : products) {  
        if (product.getId().equals(productId)) {  
            return product;  
        }  
    }  
    return null;  
}
```

```
public void updateProductStock(String productId, int quantity) {  
    Product product = getProductById(productId);  
    if (product != null) {  
        product.updateStockQuantity(quantity);  
    }  
}
```

// Main Class

public class Main {

public static void main(String[] args) {

// Create Inventory

Inventory inventory = new Inventory();

// Add Products to Inventory

Product product1 = new Product("P001", "Laptop", 999.99, 10);

Product product2 = new Product("P002", "Headphones", 29.99,  
50);

inventory.addProduct(product1);

inventory.addProduct(product2);

// Create Customer

Customer customer = new Customer("C001", "Alice",  
"alice@example.com");

// Add Products to Cart

customer.addToCart(product1);

customer.addToCart(product2);

// View Cart

System.out.println("Cart Contents:");

for (Product product : customer.viewCart()) {



```
        System.out.println(product.getName() + " - $" +  
product.getPrice());  
    }  
  
    // Checkout  
    Order order = customer.checkout();  
    System.out.println("\nOrder Summary:");  
    System.out.println("Order ID: " + order.getId());  
    System.out.println("Total Amount: $" +  
order.calculateTotalAmount());  
    System.out.println("Order Date: " + order.getOrderDate());  
}  
}
```

## OUTPUT

### Output

```
java -cp /tmp/Cmwz6Mrasl/Main  
Cart Contents:  
Laptop - $999.99  
Headphones - $29.99  
  
Order Summary:  
Order ID: ORD1721664882882  
Total Amount: $1029.98  
Order Date: 2024-07-22T16:14:42.917588  
  
=== Code Execution Successful ===
```