Scenario:

You are tasked with developing an online shopping cart system for an e-commerce website. The

system should handle products, customers, and orders, allowing customers to add products to

their cart, view the cart contents, and proceed to checkout.

Requirements:

1. Product Class:

o Attributes: productId (String), name (String), price (double), and

stockQuantity (int).

o Methods: updateStockQuantity(int quantity) to adjust stock levels when a

product is purchased.

2. Customer Class:

o Attributes: customerId (String), name (String), email (String), and cart

(List&lt;Product&gt;).

o Methods: addToCart(Product product), removeFromCart(Product

product), viewCart(), and checkout().

3. Order Class:

o Attributes: orderId (String), customer (Customer), products (List&lt;Product&gt;),

totalAmount (double), and orderDate (LocalDateTime).

o Methods: calculateTotalAmount() to compute the total cost of the order.

4. Inventory Class:

o Attributes: products (List&lt;Product&gt;).

o Methods: addProduct(Product product), getProductById(String

productId), and updateProductStock(String productId, int quantity).

Tasks:

1. Implement the Product Class:

o Define the class with appropriate attributes and methods.

o Implement logic to update the stock quantity when products are purchased.

2. Implement the Customer Class:

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

o Implement methods to add products to the cart, remove products from the cart,

view the cart contents, and proceed to checkout.

3. Implement the Order Class:

o Define the class with attributes and methods to handle order details.

o Implement the calculateTotalAmount() method to compute the total cost of the

order.

4. Implement the Inventory Class:

o Define the class to manage the product inventory.

o Implement methods to add products, retrieve a product by its ID, and update stock

levels.

5. Develop a Main Class to Test the System:

o Create instances of Product, Customer, and Inventory.

o Add products to the inventory.

o Simulate adding products to the customer&#39;s cart, viewing the cart, and checking

out.

Program:

import java.util.ArrayList;

import java.util.List;

import java.util.UUID;

import java.time.LocalDateTime;

class Product {

private String productId;

private String name;

private double price;

private int stockQuantity;

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

this.productId = UUID.randomUUID().toString();

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;

}

}

class Customer {

private String customerId;

private String name;

private String email;

private List<Product> cart;

public Customer(String name, String email) {

this.customerId = UUID.randomUUID().toString();

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() {

for (Product product : cart) {

System.out.println(product.getName() + " - $" + product.getPrice());

}

}

public void checkout(Inventory inventory) {

double totalAmount = 0;

for (Product product : cart) {

totalAmount += product.getPrice();

inventory.updateProductStock(product.getProductId(), -1);

}

cart.clear();

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

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

}

}

class Order {

private String orderId;

private Customer customer;

private List<Product> products;

private double totalAmount;

private LocalDateTime orderDate;

public Order(Customer customer, List<Product> products) {

this.orderId = UUID.randomUUID().toString();

this.customer = customer;

this.products = products;

this.orderDate = LocalDateTime.now();

this.totalAmount = calculateTotalAmount();

}

public double calculateTotalAmount() {

double total = 0;

for (Product product : products) {

total += product.getPrice();

}

return total;

}

}

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.getProductId().equals(productId)) {

return product;

}

}

return null;

}

public void updateProductStock(String productId, int quantity) {

Product product = getProductById(productId);

if (product != null) {

product.updateStockQuantity(quantity);

}

}

}

public class Main {

public static void main(String[] args) {

// Create Inventory

Inventory inventory = new Inventory();

// Add Products to Inventory

Product product1 = new Product("Laptop", 1200.99, 10);

Product product2 = new Product("Smartphone", 799.99, 20);

inventory.addProduct(product1);

inventory.addProduct(product2);

// Create Customer

Customer customer = new Customer("John Doe", "john@example.com");

// Add Products to Cart

customer.addToCart(product1);

customer.addToCart(product2);

// View Cart

customer.viewCart();

// Checkout

customer.checkout(inventory);

}

}

