## Saveetha School of Engineering

**Saveetha Institute of Medical and Technical Sciences Department of Computer Science Engineering**

#### Assignment on topic: Unit 1

##### CO2: [Develop object-oriented programs using](https://docs.google.com/document/d/10f_xjkVwmQZAh87hr4x56QfoXzI4cEdQdbaNibtDYBo/edit#heading%3Dh.pi90ampbgfqv) Operators, Inheritance and Overloading.

How would you design the purchasing product class and implement the described functionality for this online E-commerce system? Include the code and expected output based on the given scenario.

#### Scenario:

You are a software developer working for an e-commerce company called **ShopEase**, which operates an online marketplace where customers can purchase a wide variety of products ranging from electronics, clothing, home goods, and more. ShopEase aims to provide a smooth and secure shopping experience, with features that allow customers to browse, purchase, and track their orders online.

The company wants to enhance its customer experience by implementing an online e-commerce system that will enable:

* Allow customers to view product details and availability.
* Provide an option for customers to add products to the shopping cart and proceed with the checkout process.
* Ensure customers can apply discount codes and promotions during checkout.
* Track the status of orders from placement to delivery.
* Enable users to cancel or modify their orders before shipping.
* Prevent double bookings for limited-stock items by checking real-time inventory availability.
* Provide an administrative interface for inventory management.

**Tasks:**

#### Define the Product class:

* + Create attributes to store the productName, description, price, stockQuantity, isAvailable.

#### 2. Implement Methods in the Product Class:

#### displayProductInfo(): This method should display the details of the bus ticket, including:

* + productName

#### description

#### price

#### stockQuantity

#### isAvailable

#### checkAvailability(): This method should return true if the product is available, and false if the product is unavailable to the customer's location.

#### addToCart(): This method should:

#### save the loved and liked product to the cart.

#### Update the isAvailable attribute to false.

#### If the product is ordered, inform the customer that the product is ordered.

#### Simulate the ordering process:

* + **Create Objects for Different product purchases**: Create two instances of the product class for different productName, description, price, stockQuantity, and isAvailable.
  + **Display the Product Details**: Display the information of the product, showing the details and ordering status.
  + **Attempt to Purchase a product from the site**: Try to order a product on one of the E-commerce sites. After ordering, it displays the product information with its payment methods and track details .
  + **Try purchasing the next product**: After ordering a product, try to order the same product again or a different product to test how the system handles an attempt to purchase multiple products.

#### Steps:

* Display the details for all products.
* Check and purchase the product in the sites.
* Display updated and retailed product details.
* Try booking the same seat again to verify that it cannot be double-booked.

**Test Scenarios:** To validate the system, create two instances of ordering products:

1. **Laptop, High-performance laptop, Rs.50000.0. 10, Available.**
2. **Smartphone, latest Android smartphone, Rs.20000.0, 5, Available.**

#### Deliverables:

The system should show the correct product details.

The product should be successfully purchased.

Any subsequent attempts to order the same product will indicate its quantity and delivery time.

**Grading Rubrics**

| **Criterion** | **Needs Improvement (0-2)** | **Satisfactory(3-5)** | **Good(6-8)** | **Excellent(9-10)** |
| --- | --- | --- | --- | --- |
| Class Design | * Missing class definition. * Class lacks required attributes.. | - Basic class structure with some attributes but may be incomplete or inaccurate. | - Class structure is mostly correct, but may miss one or two attributes or minor issues in naming conventions. | -Complete class with all required attributes  - Proper naming conventions and encapsulation. |
| Constructor Implementation | * Constructor is missing or incomplete. * Fails to initialize attributes properly. | - Constructor is present but may miss initializing some attributes correctly. | - Constructor initializes most attributes correctly, minor issues like missing default values. | * Properly initializes all attributes in the constructor. * Clear and logical parameterized constructor. |
| Object Creation and Testing | * Missing or incorrect object creation. * Fails to create and test ticket instances as required. | - Objects are created, but testing may be incomplete or incorrect | - Objects are created and tested but may miss edge cases | - Objects are correctly created and thoroughly tested |
| Code Quality and Testing | Poor code quality with many errors or lack of testing.  Doesn’t follow naming conventions or Java best practices. | Code works but lacks consistency Some parts of the code are untested or don’t handle all edge cases. | Code is clean and follows Java best practices  Code is tested, but some edge cases are missed. | High-quality code that adheres to Java best practices, well-documented, and fully tested.  Handles edge cases and input validation thoroughly. |
| **Submission Deadline** | Frequently misses deadlines, often requiring significant extensions | Submits work within a short grace period after deadlines, with some minor delays | Deadlines are met in most cases, with only occasional minor delays | Always meets or beats submission deadlines with no delays |

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**CSA09 – PROGRAMMING IN JAVA**

# ASSIGNMENT REPORT

**REGISTER NUMBER: 192373014**

**NAME: Sowmiyan.I**

# SUBMISSION DATE: 19-11-2024

## Online E-commerce System

### Objective

The objective of this project is to design and implement a basic ecommerce system. The system should allow customers to view product details, check product availability, add products to their shopping cart, and proceed to checkout. The functionality will be implemented using a Product class with relevant attributes and methods to manage product details and the shopping cart.

### Design and Implementation

The solution was designed based on the requirements provided in the scenario. The key aspects of the implementation are as follows:

##### Class Design

##### The class Product was created to represent an individual product in the ecommerce system with relevant attributes such as product name, description, price, stock quantity, and availability status.

Attributes include:

* **productName**: The name of the product.
* **description**: A brief description of the product.
* **price**: The price of the product.
* **stockQuantity**: The number of items available in stock.
* **isAvailable**: A boolean value indicating whether the product is available for purchase.

##### Methods Implemented

* + displayProductInfo(): This method displays the details of the product, including its name, description, price, stock quantity, and availability status.
  + checkAvailability(): This method checks if the product is available for purchase (i.e., stock quantity > 0).
  + addToCart(): This method adds a product to the shopping cart if it is available, and updates the stock quantity.
  + removeFromCart(): This method removes a product from the shopping cart and restores the stock quantity.
  + checkout(): This method simulates the checkout process and confirms the order.

The program performs the following actions:

1. Displaying ticket details for buses.
2. Booking a seat for a bus.
3. Re-attempting to book the same seat to test if the system handles double booking.
4. Re-displaying ticket details after booking or cancellation to confirm the seat’s status.

##### 3. CODING

// Product Class: Represents a product in the ecommerce system

class Product {

private String productName;

private String description;

private double price;

private int stockQuantity;

private boolean isAvailable;

// Constructor to initialize product details

public Product(String productName, String description, double price, int stockQuantity) {

this.productName = productName;

this.description = description;

this.price = price;

this.stockQuantity = stockQuantity;

this.isAvailable = stockQuantity > 0; // Initially available if stock > 0

}

// Method to display product information

public void displayProductInfo() {

System.out.println("Product Name: " + productName);

System.out.println("Description: " + description);

System.out.println("Price: Rs." + price);

System.out.println("Stock Quantity: " + stockQuantity);

System.out.println("Availability: " + (isAvailable ? "Available" : "Out of Stock"));

}

// Method to check if the product is available

public boolean checkAvailability() {

return isAvailable;

}

// Method to add the product to the shopping cart

public void addToCart(int quantity) {

if (stockQuantity >= quantity) {

stockQuantity -= quantity;

isAvailable = stockQuantity > 0;

System.out.println(quantity + " " + productName + "(s) added to cart.");

} else {

System.out.println("Not enough stock available for " + productName);

}

}

// Method to remove the product from the cart

public void removeFromCart(int quantity) {

stockQuantity += quantity;

isAvailable = stockQuantity > 0;

System.out.println(quantity + " " + productName + "(s) removed from cart.");

}

// Method to checkout the cart

public void checkout() {

if (isAvailable) {

System.out.println("Checkout successful for " + productName + ". Total amount: Rs." + (price \* stockQuantity));

} else {

System.out.println("Checkout failed. Product is out of stock.");

}

}

}

// Main Class: Ecommerce System

public class EcommerceSystem {

public static void main(String[] args) {

// Create product objects

Product product1 = new Product("Laptop", "High-performance laptop", 50000, 10);

Product product2 = new Product("Smartphone", "Latest Android smartphone", 20000, 5);

// Display product information

System.out.println("Product 1 Info:");

product1.displayProductInfo();

System.out.println("\nProduct 2 Info:");

product2.displayProductInfo();

// Check availability and add products to cart

System.out.println("\nAdding 3 Laptops to Cart:");

product1.addToCart(3); // Adds 3 laptops to the cart

// Attempt to add more products than available

System.out.println("\nAttempting to add 10 Smartphones to Cart:");

product2.addToCart(10); // This will fail due to insufficient stock

// Display updated product info after adding products to the cart

System.out.println("\nUpdated Product 1 Info:");

product1.displayProductInfo();

System.out.println("\nUpdated Product 2 Info:");

product2.displayProductInfo();

// Checkout

System.out.println("\nProceeding to Checkout for Laptops:");

product1.checkout();

// Remove products from cart

System.out.println("\nRemoving 2 Laptops from Cart:");

product1.removeFromCart(2); // Removes 2 laptops from the cart

// Display final product information

System.out.println("\nFinal Product 1 Info After Removal:");

product1.displayProductInfo();

}

}

**4. OUTPUT:**

**Product 1 Info:**

**Product Name: Laptop**

**Description: High-performance laptop**

**Price: Rs.50000.0**

**Stock Quantity: 10**

**Availability: Available**

**Product 2 Info:**

**Product Name: Smartphone**

**Description: Latest Android smartphone**

**Price: Rs.20000.0**

**Stock Quantity: 5**

**Availability: Available**

**Adding 3 Laptops to Cart:**

**3 Laptop(s) added to cart.**

**Attempting to add 10 Smartphones to Cart:**

**Not enough stock available for Smartphone**

**Updated Product 1 Info:**

**Product Name: Laptop**

**Description: High-performance laptop**

**Price: Rs.50000.0**

**Stock Quantity: 7**

**Availability: Available**

**Updated Product 2 Info:**

**Product Name: Smartphone**

**Description: Latest Android smartphone**

**Price: Rs.20000.0**

**Stock Quantity: 5**

**Availability: Available**

**Proceeding to Checkout for Laptops:**

**Checkout successful for Laptop. Total amount: Rs.150000.0**

**Removing 2 Laptops from Cart:**

**2 Laptop(s) removed from cart.**

**Final Product 1 Info After Removal:**

**Product Name: Laptop**

**Description: High-performance laptop**

**Price: Rs.50000.0**

**Stock Quantity: 9**

**Availability: Available**

### Conclusion

The ecommerce system was successfully designed and implemented. It allows users to:

* View product information, including description, price, stock quantity, and availability.
* Add products to their shopping cart if the product is available.
* Attempt to add more products than available and handle such situations.
* Proceed to checkout, calculating the total cost for the selected quantity of items.
* Remove products from the cart and update stock quantities accordingly.

The system operates as expected and fulfills all requirements outlined in the scenario, providing a simple yet effective way to manage product inventory and handle online purchases.