

QUESTION:

29	<p>Design and implement a console-based Marketplace system to onboard sellers, manage catalogs, and publish products with pricing rules using OOP in Java.</p> <p>Requirements:</p> <ol style="list-style-type: none">1. Create at least 4 classes:<ul style="list-style-type: none">○ Seller – sellerId, name, email, rating, catalog.○ Product – sku, title, basePrice, category, stock.○ Catalog – list of products, category filters, bulk ops.○ MarketplaceService – onboarding, listing, pricing, search.2. Each class must include:<ul style="list-style-type: none">○ ≥4 instance/static variables.○ A constructor to initialize values.○ ≥5 methods (getters/setters, addProduct(), updatePrice(), publish(), search()).3. Demonstrate OOPS Concepts:<ul style="list-style-type: none">○ Inheritance → ApparelProduct/ElectronicProduct extend Product with rules.○ Method Overloading → search() by title/category/price range.○ Method Overriding → finalPrice() differs by product type (GST, warranty).○ Polymorphism → compute cart totals from List<Product>.○ Encapsulation → protect stock and pricing updates.4. Write a Main class (MarketplaceAppMain) to test:<ul style="list-style-type: none">○ Onboard sellers, create catalogs, add products.○ Publish listings, update stock/price.○ Run searches and print category-wise price lists.
----	--

SOURCE CODE:

```
package javaassignment1;
import java.util.*;

//Base Product Class
class Product {
    protected String sku;
    protected String title;
    protected double basePrice;
    protected String category;
    protected int stock;

    public Product(String sku, String title, double basePrice, String category, int
stock) {
        this.sku = sku;
        this.title = title;
        this.basePrice = basePrice;
        this.category = category;
        this.stock = stock;
    }

    // Getter and Setter
    public String getSku() { return sku; }
    public String getTitle() { return title; }
    public double getBasePrice() { return basePrice; }
    public String getCategory() { return category; }
    public int getStock() { return stock; }

    public void setStock(int stock) { this.stock = stock; }

    // Overridden in subclasses
    public double finalPrice() {
        return basePrice;
    }

    @Override
    public String toString() {
        return String.format("[%s] %s | Category: %s | Price: %.2f | Stock: %d",
            sku, title, category, finalPrice(), stock);
    }
}

//Inheritance: ApparelProduct
class ApparelProduct extends Product {
    private double gst = 0.05; // 5% GST

    public ApparelProduct(String sku, String title, double basePrice, String
category, int stock) {
        super(sku, title, basePrice, category, stock);
    }

    @Override
    public double finalPrice() {
        return basePrice + (basePrice * gst);
    }
}
```

```

//Inheritance: ElectronicProduct
class ElectronicProduct extends Product {
    private double warrantyCharge = 200; // flat warranty fee

    public ElectronicProduct(String sku, String title, double basePrice, String
category, int stock) {
        super(sku, title, basePrice, category, stock);
    }

    @Override
    public double finalPrice() {
        return basePrice + warrantyCharge;
    }
}

//Seller Class
class Seller {
    private String sellerId;
    private String name;
    private String email;
    private double rating;
    private List<Product> catalog;

    public Seller(String sellerId, String name, String email) {
        this.sellerId = sellerId;
        this.name = name;
        this.email = email;
        this.rating = 5.0; // default
        this.catalog = new ArrayList<>();
    }

    public String getSellerId() { return sellerId; }
    public String getName() { return name; }
    public List<Product> getCatalog() { return catalog; }

    public void addProduct(Product p) {
        catalog.add(p);
    }

    @Override
    public String toString() {
        return sellerId + " - " + name + " (" + email + "), Rating: " + rating;
    }
}

//Marketplace Service
class MarketplaceService {
    List<Seller> sellers = new ArrayList<>();

    // Onboard seller
    public void onboardSeller(Seller s) {
        sellers.add(s);
    }

    // Search Overloaded
    public List<Product> search(String title) {
        List<Product> results = new ArrayList<>();
        for (Seller s : sellers) {

```

```

        for (Product p : s.getCatalog()) {
            if (p.getTitle().toLowerCase().contains(title.toLowerCase())) {
                results.add(p);
            }
        }
    }
    return results;
}

public List<Product> searchByCategory(String category) {
    List<Product> results = new ArrayList<>();
    for (Seller s : sellers) {
        for (Product p : s.getCatalog()) {
            if (p.getCategory().equalsIgnoreCase(category)) {
                results.add(p);
            }
        }
    }
    return results;
}

public List<Product> searchByPrice(double min, double max) {
    List<Product> results = new ArrayList<>();
    for (Seller s : sellers) {
        for (Product p : s.getCatalog()) {
            if (p.finalPrice() >= min && p.finalPrice() <= max) {
                results.add(p);
            }
        }
    }
    return results;
}

public void listAllProducts() {
    for (Seller s : sellers) {
        System.out.println("\nSeller: " + s.getName());
        for (Product p : s.getCatalog()) {
            System.out.println("    " + p);
        }
    }
}

//Main App
public class MarketplaceAppMain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        MarketplaceService service = new MarketplaceService();

        while (true) {
            System.out.println("\n===== Marketplace Menu =====");
            System.out.println("1. Onboard Seller");
            System.out.println("2. Add Product");
            System.out.println("3. List All Products");
            System.out.println("4. Search by Title");
            System.out.println("5. Search by Category");
            System.out.println("6. Search by Price Range");
            System.out.println("7. Exit");
            System.out.print("Choose option: ");

```

```

    int choice = sc.nextInt();
    sc.nextLine(); // consume newline

    switch (choice) {
        case 1:
            System.out.print("Enter Seller ID: ");
            String sid = sc.nextLine();
            System.out.print("Enter Name: ");
            String name = sc.nextLine();
            System.out.print("Enter Email: ");
            String email = sc.nextLine();
            service.onboardSeller(new Seller(sid, name, email));
            System.out.println("✔ Seller onboarded successfully!");
            break;

        case 2:
            System.out.print("Enter Seller ID to add product: ");
            String sellerId = sc.nextLine();
            Seller selectedSeller = null;
            for (Seller s : service.sellers) {
                if (s.getSellerId().equals(sellerId)) {
                    selectedSeller = s;
                    break;
                }
            }
            if (selectedSeller == null) {
                System.out.println("✗ Seller not found!");
                break;
            }
            System.out.print("Enter SKU: ");
            String sku = sc.nextLine();
            System.out.print("Enter Title: ");
            String title = sc.nextLine();
            System.out.print("Enter Base Price: ");
            double price = sc.nextDouble();
            sc.nextLine();
            System.out.print("Enter Category (Apparel/Electronics/Other): ");
            String category = sc.nextLine();
            System.out.print("Enter Stock: ");
            int stock = sc.nextInt();
            sc.nextLine();

            Product p;
            if (category.equalsIgnoreCase("Apparel")) {
                p = new ApparelProduct(sku, title, price, category, stock);
            } else if (category.equalsIgnoreCase("Electronics")) {
                p = new ElectronicProduct(sku, title, price, category,
stock);
            } else {
                p = new Product(sku, title, price, category, stock);
            }
            selectedSeller.addProduct(p);
            System.out.println("✔ Product added successfully!");
            break;

        case 3:
            service.listAllProducts();
            break;
    }
}

```

```

        case 4:
            System.out.print("Enter Title Keyword: ");
            String keyword = sc.nextLine();
            List<Product> results1 = service.search(keyword);
            results1.forEach(System.out::println);
            break;

        case 5:
            System.out.print("Enter Category: ");
            String cat = sc.nextLine();
            List<Product> results2 = service.searchByCategory(cat);
            results2.forEach(System.out::println);
            break;

        case 6:
            System.out.print("Enter Min Price: ");
            double min = sc.nextDouble();
            System.out.print("Enter Max Price: ");
            double max = sc.nextDouble();
            List<Product> results3 = service.searchByPrice(min, max);
            results3.forEach(System.out::println);
            break;

        case 7:
            System.out.println("👋 Exiting... Thank you!");
            sc.close();
            return;

        default:
            System.out.println("❌ Invalid choice!");
    }
}
}
}

```

OUTPUT:

```
717824F229 - javaprogram/src/javaassignment1/MarketplaceAppMain.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Javadoc Declaration Console X
<terminated> MarketplaceAppMain [Java Application] C:\Users\HP\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_6
===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 1
Enter Seller ID: 101
Enter Name: Danu
Enter Email: danu04@gmail.com
☑ Seller onboarded successfully!

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 2
Enter Seller ID to add product: 102
X Seller not found!

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 2
Enter Seller ID to add product: 101
Enter SKU: P101
Enter Title: Laptop
Enter Base Price: 40000
Enter Category (Apparel/Electronics/Other): Electronics
Enter Stock: 20
☑ Product added successfully!

===== Marketplace Menu =====
```

```
717824F229 - javaprogram/src/javaassignment1/MarketplaceAppMain.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Javadoc Declaration Console X
<terminated> MarketplaceAppMain [Java Application] C:\Users\HP\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_6
===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 1
Enter Seller ID: 101
Enter Name: Danu
Enter Email: danu04@gmail.com
☑ Seller onboarded successfully!

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 2
Enter Seller ID to add product: 102
X Seller not found!

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 2
Enter Seller ID to add product: 101
Enter SKU: P101
Enter Title: Laptop
Enter Base Price: 40000
Enter Category (Apparel/Electronics/Other): Electronics
Enter Stock: 20
☑ Product added successfully!

===== Marketplace Menu =====
```

```
717824f229 - javaprogram/src/javaassignment1/MarketplaceAppMain.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help

@ Javadoc Declaration Console X
<terminated> MarketplaceAppMain [Java Application] C:\Users\HP\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_2:
6. Search by Price Range
7. Exit
Choose option: 4
Enter Title Keyword: Laptop
[P101] Laptop | Category: Electronics | Price: 40200.00 | Stock: 20

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 5
Enter Category: Electronics
[P101] Laptop | Category: Electronics | Price: 40200.00 | Stock: 20

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 6
Enter Min Price: 30000
Enter Max Price: 50000
[P101] Laptop | Category: Electronics | Price: 40200.00 | Stock: 20

===== Marketplace Menu =====
1. Onboard Seller
2. Add Product
3. List All Products
4. Search by Title
5. Search by Category
6. Search by Price Range
7. Exit
Choose option: 7
Exiting... Thank you!
```

GITHUB REPOSITORY LINK:

<https://github.com/717824f229-design/Marketplace-App-.git>