Core Java Case Study:

Objective: Design and implement the system using Java that demonstrates Object-Oriented Programming (OOP) concepts, including inheritance, polymorphism, and collections.

E-Commerce Platform – Online Shopping Application

Design online shopping application which allows customers to browse products, manage their shopping carts, and place orders, while admins manage product listings and update order statuses. The application features a menu-driven console interface for both customers and admins, sharing a common service layer to ensure data consistency across operations. Customers can view and order products, and admins can add or remove products and update order statuses to "Completed," "Delivered," or "Cancelled," reflecting real-time stock adjustments and order management.

Entities:

- Product: Represents a product in the catalog.
- User: Base class for different types of users.
- Customer: Inherits from User, can place orders.
- Admin: Inherits from User, can manage the product catalog.
- Order: Represents an order placed by a customer.
- ShoppingCart: Represents a shopping cart for a customer.

Functionalities:

- Admin can add, update, and remove products.
- Customer can browse products.
- Customer can add products to the shopping cart.
- Customer can place an order.
- List all orders for a customer.

Entities and Attributes:

1. Product

Attributes:

productId: Unique identifier for the product.

name: Name of the product.

price: Price of the product.

stockQuantity: Available stock quantity of the product.

Relationships:

Many-to-Many with Order: A Product can be part of many Orders, and an Order can include many Products.

Many-to-Many with ShoppingCart: A Product can be in many ShoppingCarts, and a ShoppingCart can contain many Products.

2. User (Base Class)

Attributes:

userId: Unique identifier for the user.

username: Username of the user.

email: Email address of the user.

Relationships:

Inherits to Customer: Customer extends User.

Inherits to Admin: Admin extends User.

3. Customer (Inherits from User)

Attributes:

address: Address of the customer.

shoppingCart: The customer's shopping cart (one-to-one relationship).

orders: List of orders placed by the customer (one-to-many relationship).

Relationships:

One-to-One with ShoppingCart: Each Customer has exactly one ShoppingCart.

One-to-Many with Order: Each Customer can place multiple Orders.

4. Admin (Inherits from User)

Relationships:

Manages Products: An Admin can manage (add/remove) multiple Products.

5. Order

Attributes:

orderId: Unique identifier for the order.

customer: The customer who placed the order (many-to-one relationship).

products: List of ProductQuantityPair representing the products and their quantities in the order (many-to-many relationship with Product).

status: Status of the order (e.g., "Pending", "Completed").

Relationships:

Many-to-One with Customer: Each Order is placed by one Customer.

Many-to-Many with Product: An Order can include multiple Products, and a Product can be included in multiple Orders.

6. ShoppingCart

Attributes:

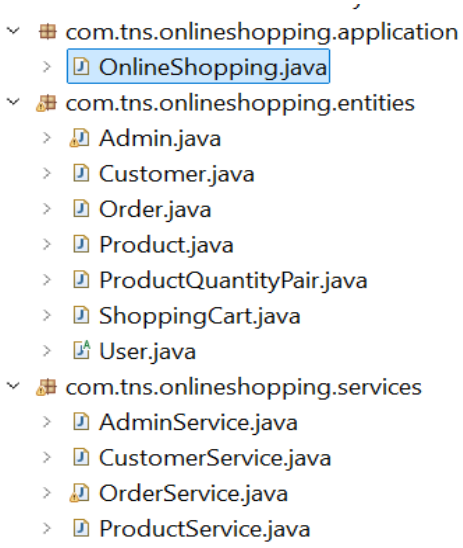
items: Map of Product to quantity, representing the items in the cart (many-to-many relationship with Product).

Relationships:

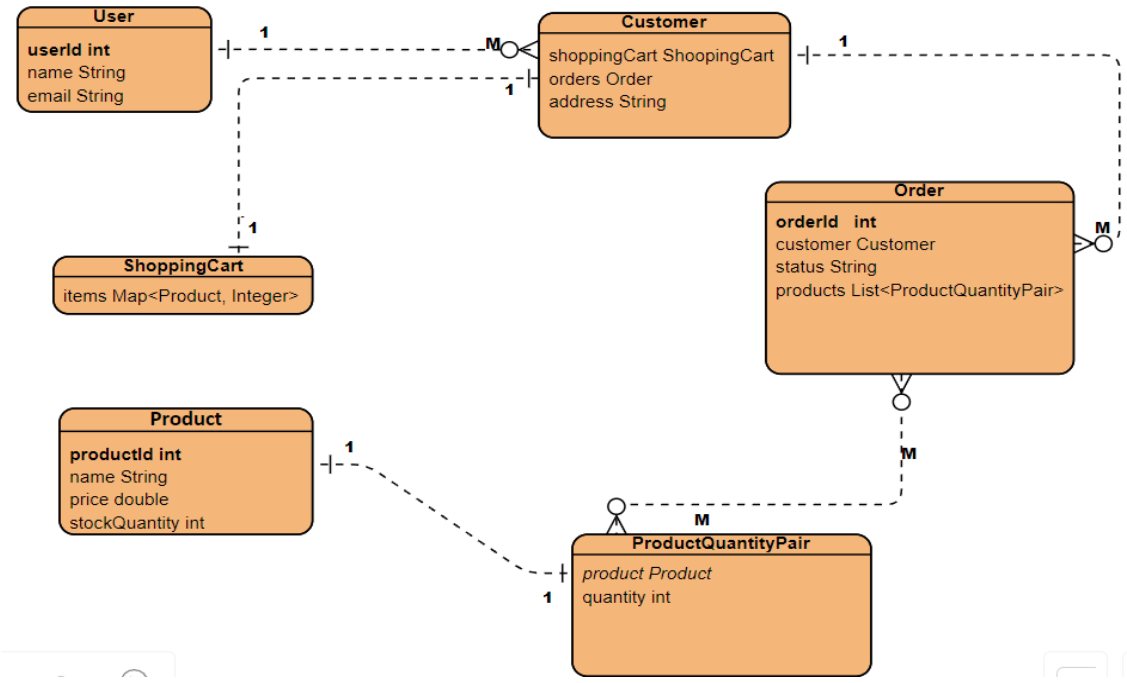
Many-to-One with Customer: Each ShoppingCart belongs to one Customer.

Many-to-Many with Product: A ShoppingCart can contain multiple Products, and a Product can be in multiple ShoppingCarts.

Design the application for the same.



E – R Diagram



Design the following:

1. Entities

User

OnlineShopping.java | User.java

```java
//Base class for Users
package com.tns.onlineshopping.entities;
public abstract class User {
    private int userId;
    private String username;
    private String email;

    public User(int userId, String username, String email) {
        this.userId = userId;
        this.username = username;
        this.email = email;
    }

    // Getters and setters
    public int getUserId() { return userId;     }

    public void setUserId(int userId) {     this.userId = userId;    }

    public String getUsername() {       return username;     }

    public void setUsername(String username) {  this.username = username;    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    @Override
    public String toString() {
        return "User [userId=" + userId + ", username=" + username + ", email=" + email + "]";
    }
}
```

Admin

```java
//Program to define Admin entity
package com.tns.onlineshopping.entities;

public class Admin extends User {
    public Admin(int userId, String username, String email) {
        super(userId, username, email);
    }
}
```

Customer

OnlineShopping.java | Customer.java

```java
//Program to define Customer entity
package com.tns.onlineshopping.entities;

import java.util.ArrayList;

public class Customer extends User {
    private String address;
    private ShoppingCart shoppingCart;
    private List<Order> orders;

    public Customer(int userId, String username, String email, String address) {
        super(userId, username, email);
        this.address = address;
        this.shoppingCart = new ShoppingCart();
        this.orders = new ArrayList<>();
    }

    // Getters and setters
    public String getAddress() { return address; }
    public void setAddress(String address) { this.address = address; }
    public ShoppingCart getShoppingCart() { return shoppingCart; }
    public List<Order> getOrders() { return orders; }

    public void addOrder(Order order) {
        orders.add(order);
    }
}
```

Product

OnlineShopping.java | Product.java

```java
//Program to define Product class
package com.tns.onlineshopping.entities;

public class Product {
    private int productId;
    private String name;
    private double price;
    private int stockQuantity;

    public Product(int productId, String name, double price, int stockQuantity) {
        this.productId = productId;
        this.name = name;
        this.price = price;
        this.stockQuantity = stockQuantity;
    }

    // Getters and setters
    public int getProductId() { return productId; }
    public void setProductId(int productId) { this.productId = productId; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public double getPrice() { return price; }
    public void setPrice(double price) { this.price = price; }
    public int getStockQuantity() { return stockQuantity; }
    public void setStockQuantity(int stockQuantity) { this.stockQuantity = stockQuantity; }

    @Override
    public String toString() {
        return "Product [productId=" + productId + ", name=" + name + ", price=" + price + ", stockQuantity=" + stockQuantity + "]";
    }
}

```

ProductQuantityPair

OnlineShopping.java | ProductQuantityPair.java

```java
//Program to define ProductQuantityPair class
package com.tns.onlineshopping.entities;

public class ProductQuantityPair {
     private Product product;
     private int quantity;

     public ProductQuantityPair(Product product, int quantity) {
          this.product = product;
          this.quantity = quantity;
     }

     // Getters and setters
     public Product getProduct() {
          return product;
     }

     public void setProduct(Product product) {
          this.product = product;
     }

     public int getQuantity() {
          return quantity;
     }

     public void setQuantity(int quantity) {
          this.quantity = quantity;
     }
}

```

Shopping Cart

OnlineShopping.java | ShoppingCart.java

```java
//Program to define ShoppingCart class
package com.tns.onlineshopping.entities;

import java.util.HashMap;

public class ShoppingCart {
     private Map<Product, Integer> items;

     public ShoppingCart() {
          this.items = new HashMap<>();
     }

     // Getters and setters
     public Map<Product, Integer> getItems() {
          return items;
     }

     public void addItem(Product product, int quantity) {
          items.put(product, quantity);
     }

     public void removeItem(Product product) {
          items.remove(product);
     }

     @Override
     public String toString() {
          return "ShoppingCart [items=" + items + "]";
     }
}
```

2. Services

AdminService

OnlineShopping.java | AdminService.java

```java
//Program to define AdminService class
package com.tns.onlineshopping.services;

import java.util.ArrayList;

public class AdminService {


        private List<Admin> adminList = new ArrayList<>();

        public void addAdmin(Admin admin) {
            adminList.add(admin);
        }

        public List<Admin> getAdmins() {
            return adminList;
        }
    }

```

CustomerService

OnlineShopping.java | CustomerService.java

```java
//Program to define CustomerService class
package com.tns.onlineshopping.services;

import java.util.ArrayList;

public class CustomerService {

        private List<Customer> customerList = new ArrayList<>();

        public void addCustomer(Customer customer) {
            customerList.add(customer);
        }
        //retrieve Customer by ID
        public Customer getCustomer(int userId) {
            return customerList.stream()
                    .filter(customer -> customer.getUserId() == userId)
                    .findFirst()
                    .orElse(null);
        }

        public List<Customer> getCustomers() {
            return customerList;
        }
    }

```

ProductService

OnlineShopping.java | OrderService.java | ProductService.java

```java
//Program to define Product Service - add, remove, retrieve products
package com.tns.onlineshopping.services;

import java.util.ArrayList;

public class ProductService {

    private List<Product> productList = new ArrayList<>();

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

    public void removeProduct(int productId) {
        productList.removeIf(product -> product.getProductId() == productId);
    }

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

     public Product getProductById(int productId) {
            return productList.stream()
                    .filter(product -> product.getProductId() == productId)
                    .findFirst()
                    .orElse(null);
        }
}
```

OrderService

OnlineShopping.java | OrderService.java

```java
//Program to define Order Service - place the order, update the status, retrieve order by ID and List all orders
package com.tns.onlineshopping.services;

import java.util.ArrayList;

public class OrderService {
    private List<Order> orderList = new ArrayList<>();

    public OrderService() {
    }

    public void placeOrder(Order order) {
        orderList.add(order);
    }

    public void updateOrderStatus(int orderId, String status) {
        Order order = getOrder(orderId);

        if (order != null) {
            if ("Completed".equalsIgnoreCase(status) && "Pending".equalsIgnoreCase(order.getStatus())) {
                for (ProductQuantityPair pair : order.getProducts()) {
                    Product product = pair.getProduct();
                    int quantity = pair.getQuantity();

                    if (product.getStockQuantity() >= quantity) {
                        product.setStockQuantity(product.getStockQuantity() - quantity);
                    } else {
                        System.out.println("Insufficient stock for product: " + product.getName());
                        return;
                    }
                }
            } else if ("Cancelled".equalsIgnoreCase(status)) {
                if ("Completed".equalsIgnoreCase(order.getStatus()) || "Pending".equalsIgnoreCase(order.getStatus())) {
                    for (ProductQuantityPair pair : order.getProducts()) {
                        Product product = pair.getProduct();
                        int quantity = pair.getQuantity();
```

OnlineShopping.java | OrderService.java

```java
                for (ProductQuantityPair pair : order.getProducts()) {
                    Product product = pair.getProduct();
                    int quantity = pair.getQuantity();

                    if (product.getStockQuantity() >= quantity) {
                        product.setStockQuantity(product.getStockQuantity() - quantity);
                    } else {
                        System.out.println("Insufficient stock for product: " + product.getName());
                        return;
                    }
                }
            } else if ("Cancelled".equalsIgnoreCase(status)) {
                if ("Completed".equalsIgnoreCase(order.getStatus()) || "Pending".equalsIgnoreCase(order.getStatus())) {
                    for (ProductQuantityPair pair : order.getProducts()) {
                        Product product = pair.getProduct();
                        int quantity = pair.getQuantity();
                        product.setStockQuantity(product.getStockQuantity() + quantity);
                    }
                }
            } else if ("Delivered".equalsIgnoreCase(status) && "Completed".equalsIgnoreCase(order.getStatus())) {
                // Only update the status to Delivered, no stock adjustment needed
            }
            order.setStatus(status);
        } else
            System.out.println("Invalid Order");
    }

    public Order getOrder(int orderId) {
        return orderList.stream().filter(order -> order.getOrderId() == orderId).findFirst().orElse(null);
    }

    public List<Order> getOrders() {
        return orderList;
    }
}
```

3. Menu driven Driver Class for Admin and Customer module

OnlineShopping.java

```java
//Program to demonstrate Online Shopping Application
package com.tns.onlineshopping.application;

import java.util.Scanner;

public class OnlineShopping {
    private static ProductService productService = new ProductService();
    private static CustomerService customerService = new CustomerService();
    private static OrderService orderService = new OrderService();
    private static AdminService adminService = new AdminService();

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("1. Admin Menu");
            System.out.println("2. Customer Menu");
            System.out.println("3. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();

            switch (choice) {
            case 1: //Admin Module
                int adminChoice;
                do {
                    System.out.println("\nAdmin Menu:");
                    System.out.println("1. Add Product");
                    System.out.println("2. Remove Product");
                    System.out.println("3. View Products");
                    System.out.println("4. Create Admin");
                    System.out.println("5. View Admins");
                    System.out.println("6. Update Order Status");
                    System.out.println("7. View Orders");
                    System.out.println("8. Return");
                    System.out.print("Choose an option: ");
                    adminChoice = scanner.nextInt();
```

OnlineShopping.java

```java
                    switch (adminChoice) {
                    case 1:
                        addProduct(scanner);
                        break;
                    case 2:
                        removeProduct(scanner);
                        break;
                    case 3:
                        viewProducts();
                        break;
                    case 4:
                        createAdmin(scanner);
                        break;
                    case 5:
                        viewAdmins();
                        break;
                    case 6:
                        updateOrderStatus(scanner);
                        break;
                    case 7:
                        viewOrders();
                        break;
                    case 8:
                        System.out.println("Exiting Admin...");
                        break;
                    default:
                        System.out.println("Invalid choice! Please try again.");
                    }
                } while (adminChoice != 8);
```

OnlineShopping.java

```java
            case 2: //Customer Module
                int customerChoice;
                do {
                    System.out.println("\nCustomer Menu:");
                    System.out.println("1. Create Customer");
                    System.out.println("2. View Customers");
                    System.out.println("3. Place Order");
                    System.out.println("4. View Orders");
                    System.out.println("5. View Products");
                    System.out.println("6. Return");
                    System.out.print("Choose an option: ");
                    customerChoice = scanner.nextInt();
                    switch (customerChoice) {
                    case 1:
                        createCustomer(scanner);
                        break;
                    case 2:
                        viewCustomers();
                        break;
                    case 3:
                        placeOrder(scanner);
                        break;
                    case 4:
                        viewOrders(scanner);
                        break;
                    case 5:
                        viewProducts();
                        break;
                    case 6:
                        System.out.println("Exiting Customer Menu...");
                        break;
                    default:
                        System.out.println("Invalid choice! Please try again.");
                    }
                } while (customerChoice != 6);
                break;
```

OnlineShopping.java

```java
                } while (customerChoice != 6);
                break;
            case 3:
                System.out.println("Exiting...");
                scanner.close();
                return;
            default:
                System.out.println("Invalid choice! Please try again.");
            }
        }

    }

    private static void addProduct(Scanner scanner) {
        System.out.print("Enter Product ID: ");
        int productId = scanner.nextInt();
        System.out.print("Enter Product Name: ");
        String name = scanner.next();
        System.out.print("Enter Product Price: ");
        double price = scanner.nextDouble();
        System.out.print("Enter Stock Quantity: ");
        int stockQuantity = scanner.nextInt();

        Product product = new Product(productId, name, price, stockQuantity);
        productService.addProduct(product);
        System.out.println("Product added successfully!");
    }

    private static void removeProduct(Scanner scanner) {
        System.out.print("Enter Product ID: ");
        int productId = scanner.nextInt();

        productService.removeProduct(productId);
        System.out.println("Product removed successfully!");
    }
```

Output

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 1

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 1

Enter Product ID: 101

Enter Product Name: T-Shirt

Enter Product Price: 560

Enter Stock Quantity: 100

Product added successfully!

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 1

Enter Product ID: 102

Enter Product Name: Trouser

Enter Product Price: 1400

Enter Stock Quantity: 50

Product added successfully!

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 3

Products:

Product [productId=101, name=T-Shirt, price=560.0, stockQuantity=100]

Product [productId=102, name=Trouser, price=1400.0, stockQuantity=50]

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 8

Exiting Admin...

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 2

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 1

Enter User ID: 1001

Enter Username: Aniket

Enter Email: aniket@gmail.com

Enter Address: Nashik

Customer created successfully!

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 1

Enter User ID: 1002

Enter Username: Pooja

Enter Email: pooja@gmail.com

Enter Address: Pune

Customer created successfully!

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 2

Customers:

User ID: 1001, Username: Aniket, Email: aniket@gmail.com, Address: Nashik

User ID: 1002, Username: Pooja, Email: pooja@gmail.com, Address: Pune

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 5

Products:

Product [productId=101, name=T-Shirt, price=560.0, stockQuantity=100]

Product [productId=102, name=Trouser, price=1400.0, stockQuantity=50]

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 3

Enter Customer ID: 1001

Enter Product ID to add to order (or -1 to complete): 101

Enter quantity: 20

Enter Product ID to add to order (or -1 to complete): -1

Order placed successfully!

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 3

Enter Customer ID: 1002

Enter Product ID to add to order (or -1 to complete): 101

Enter quantity: 50

Enter Product ID to add to order (or -1 to complete): 102

Enter quantity: 20

Enter Product ID to add to order (or -1 to complete): -1

Order placed successfully!

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 4

Enter Customer ID: 1001

Orders:

Order ID: 1, Status: Pending

Product: T-Shirt, Quantity: 20

Customer Menu:

1. Create Customer

2. View Customers

3. Place Order

4. View Orders

5. View Products

6. Return

Choose an option: 6

Exiting Customer Menu...

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 1

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 6

Enter Order ID: 1

Enter new status (Completed/Delivered/Cancelled): Completed

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 7

Orders:

Order ID: 1, Customer: Aniket, Status: Completed

  Product: T-Shirt, Quantity: 20

Order ID: 2, Customer: Pooja, Status: Pending

  Product: T-Shirt, Quantity: 50

  Product: Trouser, Quantity: 20

Admin Menu:

1. Add Product

2. Remove Product

3. View Products

4. Create Admin

5. View Admins

6. Update Order Status

7. View Orders

8. Return

Choose an option: 8

Exiting Admin...

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 3

Exiting...