Đánh giá Cohesion, Coupling và SOLID của PlaceOrderController

Cohesion

Phương thức/Lớp	Cohesion	Giải thích	Khuyến nghị
Constructor (PlaceOrderController)	Functional Cohesion	Dùng để khởi tạo danh sách cartMediaList.	Duy trì trạng thái hiện tại.
getCartMediaList, getDeliveryInfo	Functional Cohesion	Các phương thức này chỉ lấy thông tin thuộc tính.	Duy trì trạng thái hiện tại.
calculateShippingFee	Logical Cohesion	Tính phí vận chuyển dựa trên trọng lượng và số tiền trong giỏ hàng với nhiều logic khác nhau (khu vực, khuyến mãi).	Tách thành các phương thức nhỏ hoặc sử dụng Strategy Pattern để tính phí vận chuyển dựa trên loại địa phương.
processDeliveryInfo	Logical Cohesion	Kết hợp xác thực thông tin người dùng và khởi tạo DeliveryInfo.	Tách logic xác thực người dùng sang Utils hoặc một dịch vụ riêng.
checkRushOrderAddress	Functional Cohesion	Kiểm tra điều kiện đơn hàng nhanh (rush order) dựa trên địa chỉ giao hàng.	Duy trì trạng thái hiện tại.
calculateSubtotal	Functional Cohesion	Tính tổng giá trị đơn hàng dựa trên số lượng và giá của các sản phẩm trong giỏ hàng.	Cân nhắc chuyển logic này sang CartMedia hoặc một dịch vụ tính toán tổng hợp riêng.

Coupling

Lớp	Coupling với	Loại Coupling	Giải thích	Khuyến nghị
PlaceOrderController	CartMedia	Data Coupling	Gọi trực tiếp các phương thức của CartMedia để tính toán giá trị và trọng lượng.	Duy trì trạng thái hiện tại, hoặc tạo abstraction nếu cần tái sử dụng logic tính toán trong các ngữ cảnh khác.
PlaceOrderController	DeliveryInfo	Data Coupling	Sử dụng DeliveryInfo để lưu trữ và xử lý thông tin giao hàng.	Duy trì trạng thái hiện tại.

Lớp	Coupling với	Loại Coupling	Giải thích	Khuyến nghị
PlaceOrderController	Utils	Control Coupling	Gọi trực tiếp phương thức Utils.processUserInfo để xác thực thông tin giao hàng.	Tách chức năng xử lý thông tin giao hàng sang một dịch vụ chuyên trách thay vì dùng Utils.
calculateShippingFee	cartMediaList	Data Coupling	Phụ thuộc mạnh vào cấu trúc của cartMediaList để tính phí vận chuyển và kiểm tra các điều kiện khác nhau.	Tách logic này ra một service chuyên biệt hoặc sử dụng các object nhỏ hơn để đại diện cho phí vận chuyển.

SOLID

	Tuân thủ/Vi phạm	Giải thích	Khuyến nghị
SRP	Vi phạm	PlaceOrderController xử lý nhiều nhiệm vụ khác nhau: quản lý giỏ hàng, tính phí vận chuyển, kiểm tra địa chỉ rush order.	Tách lớp này thành nhiều lớp nhỏ hơn, ví dụ: ShippingCalculator, DeliveryInfoProcessor, OrderValidator.
OCP	Vi phạm	Nếu thêm các địa phương mới hoặc chính sách tính phí vận chuyển mới, cần thay đổi code trong calculateShippingFee.	Áp dụng Strategy Pattern hoặc Factory để mở rộng logic tính phí vận chuyển mà không cần chỉnh sửa phương thức hiện có.
LSP	Tuân thủ	Các lớp con của CartMedia hoặc DeliveryInfo (nếu tồn tại) có thể thay thế mà không phá vỡ chức năng.	Duy trì trạng thái hiện tại.
ISP	Không áp dụng	PlaceOrderController không dựa trên interface nào cụ thể, do đó không có dấu hiệu vi phạm.	Nên xem xét tách chức năng qua các interface chuyên biệt, ví dụ: OrderProcessor, ShippingService.
DIP	Vi phạm	PlaceOrderController phụ thuộc trực tiếp vào Utils và không sử dụng abstraction.	Sử dụng Dependency Injection để inject UserValidationService hoặc ShippingService thay vì gọi trực tiếp từ Utils.

Code mới:

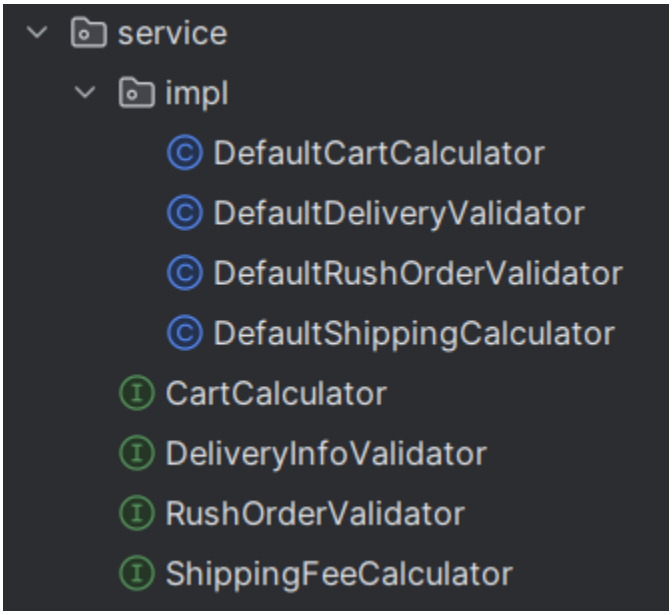
PlaceOrderController

```java
package com.example.aims.controller;

import com.example.aims.entity.cart.CartMedia;
import com.example.aims.entity.delivery.DeliveryInfo;
import com.example.aims.exception.RuntimeException;
import com.example.aims.service.ShippingFeeCalculator;
import com.example.aims.service.DeliveryInfoValidator;
import com.example.aims.service.RushOrderValidator;
import com.example.aims.service.CartCalculator;
import com.example.aims.service.impl.DefaultShippingCalculator;
import com.example.aims.service.impl.DefaultDeliveryValidator;
import com.example.aims.service.impl.DefaultRushOrderValidator;
import com.example.aims.service.impl.DefaultCartCalculator;

import java.util.HashMap;
import java.util.List;

public class PlaceOrderController {  8 usages  JVCP223
    private List<CartMedia> cartMediaList;  4 usages
    private DeliveryInfo deliveryInfo;  3 usages
    private final ShippingFeeCalculator shippingCalculator;  2 usages
    private final DeliveryInfoValidator deliveryValidator;  2 usages
    private final RushOrderValidator rushOrderValidator;  2 usages
    private final CartCalculator cartCalculator;  2 usages

    public PlaceOrderController(List<CartMedia> cartMediaList) {  1 usage  JVCP223
        this.cartMediaList = cartMediaList;
        this.shippingCalculator = new DefaultShippingCalculator();
        this.deliveryValidator = new DefaultDeliveryValidator();
        this.rushOrderValidator = new DefaultRushOrderValidator();
        this.cartCalculator = new DefaultCartCalculator();
    }
```

```java
    public List<CartMedia> getCartMediaList() {  2 usages  JVCP223
        return cartMediaList;
    }

    public DeliveryInfo getDeliveryInfo() {  3 usages  JVCP223
        return deliveryInfo;
    }

    public int calculateShippingFee(String province) {  2 usages  JVCP223
        return shippingCalculator.calculateFee(cartMediaList, province);
    }

    public void processDeliveryInfo(HashMap<String, String> info) throws RuntimeException {  1 usage  JVCP223
        deliveryValidator.validateInfo(
                info.get("name"),
                info.get("email"),
                info.get("address"),
                info.get("phone")
        );

        deliveryInfo = DeliveryInfo.builder()
                .name(info.get("name"))
                .province(info.get("province"))
                .address(info.get("address"))
                .phone(info.get("phone"))
                .email(info.get("email"))
                .build();
    }
```

```java
    public boolean checkRushOrderAddress() {  1 usage  JVCP223
        return rushOrderValidator.isEligible(deliveryInfo.getProvince());
    }

    public int calculateSubtotal() {  2 usages  JVCP223
        return cartCalculator.calculateSubtotal(cartMediaList);
    }
}
```

Thêm service và implementation:



Code cũ:

PlaceOrderController:

```java
package com.example.aims.controller;

import com.example.aims.entity.cart.CartMedia;
import com.example.aims.entity.delivery.DeliveryInfo;
import com.example.aims.exception.RuntimeException;
import com.example.aims.utils.Utils;

import java.util.HashMap;
import java.util.List;

public class PlaceOrderController {  8 usages  JVCP223 +1

    private List<CartMedia> cartMediaList;  4 usages

    private DeliveryInfo deliveryInfo;  3 usages

    public PlaceOrderController (List<CartMedia> cartMediaList) { this.cartMediaList = cartMediaList; }

    public List<CartMedia> getCartMediaList() {  2 usages  JVCP223
        return cartMediaList;
    }

    public DeliveryInfo getDeliveryInfo() { return deliveryInfo; }

    public int calculateShippingFee(String provinceInput) {  2 usages  JVCP223 +1
        double totalWeight = 0;
        int amount = 0;
        int shippingfee = 0;

        for (CartMedia cm : cartMediaList) {
            amount += cm.getPrice() * cm.getQuantity();
            totalWeight += cm.getMedia().getWeight() * cm.getQuantity();
```

```
        }

        if (provinceInput.equals("Ho Chi Minh (TP)") || provinceInput.equals("Ha Noi (TP)")) {
            totalWeight -= 3;
            shippingfee += 22000;
        } else {
            totalWeight -= 0.5;
            shippingfee += 30000;
        }

        if (totalWeight > 0) {
            totalWeight = (int) Math.ceil(totalWeight * 2);

            shippingfee += 2500 * totalWeight;
        }

        if (amount > 1000000) {
            shippingfee = Math.max(shippingfee - 25000, 0);
        }

        return shippingfee;
    }

    public void processDeliveryInfo(HashMap<String, String> info) throws RuntimeException {  1 usage  JVCP223
        Utils.processUserInfo(info.get("name"), info.get("email"),info.get("address"),info.get("phone"));
        deliveryInfo = DeliveryInfo.builder()
                .name(info.get("name"))
                .province(info.get("province"))
                .address(info.get("address"))
                .phone(info.get("phone"))
                .email(info.get("email"))
                .build();
    }

    public boolean checkRushOrderAddress() { return deliveryInfo.getProvince().equals("Ha Noi (TP)"); }

    public int calculateSubtotal() {  2 usages  JVCP223
        return cartMediaList.stream()  Stream<CartMedia>
                .mapToInt(cartMedia -> cartMedia.getMedia().getPrice() * cartMedia.getQuantity())  IntStream
                .sum();
    }
}
```