**KHÁI NIỆM VÀ NGUYÊN LÝ THIẾT KẾ PHẦN MỀM AIMS**

# I. Đánh giá bản thiết kế và mã nguồn

## 1. Coupling

1.1 Content Coupling

|  |  |  |
| --- | --- | --- |
| Các modules liên quan | Lý do coupling | Giải pháp cải tiến |
| Lớp Order với Lớp Hash Map vs module nào đó trong tương lai | Module quản lý này lấy đối tượng order bằng cách gọi getDeliveryInfo() của lớp Order và sau đó thay đổi deliveryInfo bằng cách gọi put của HashMap. Do đó giá trị delivery Infor bị thay đổi trong khi lớp Order không biết gì |  |

1.2 Common Coupling:

|  |  |  |
| --- | --- | --- |
| Các modules liên quan | Lý do coupling | Sự cải tiến |
| Lớp ViewCartController vs lớp placeOrderContrller với lớp entity Car | Lớp cart có global static member là phương thức getCart, và 2 lớp PlaceOrderController vs phương thức getCartSutotal của viewCartController sử dụng phương thức này. |  |

1.3 Control Coupling

|  |  |  |
| --- | --- | --- |
| Các modules liên quan | Lý do coupling | Sự cải tiến |
| Phương thức calculateShipFee của placeRushOrder | Phương thức calculateShipFee của placeRushOrder có tham số truyền vào cho biết có giao hàng nhanh hay không, từ đó sinh ra 2 cách tính tiền phí ship khác nhau | Tách ra thành các lớp và sử dụng stradegy pattern, chuyển phương thức này cho placeOrderController |

1.4 Stamp Coupling

|  |  |  |
| --- | --- | --- |
| Module | Lý do coupling | Sự cải tiến |
| PHương thức calculateShipFee | Truyền cả 1 đối tượng Order nhưng chỉ dùng 1 vài thuộc tính trong đó | Thay đổi tham số thành 1 Array, nhưng sẽ đánh đổi việc mở rộng nhiều phương thức tính toán sau này. Vậy nên giữ nguyên |
|  |  |  |

1.5 Data Coupling

|  |  |  |
| --- | --- | --- |
| Các modules liên quan | Lý do coupling | Sự cải tiến |
| PaymentController vs Payment Form | PaymentController truyền thông tin invoice vào PayMentForm | Chấp nhận được, tốt |
| PlaceOrder vs PlaceRushOrder | Gọi đến PlaceRushOrderController để kiểm tra khả năng đặt hàng nhanh và thanh toán. |  |

## 2. Cohesion

2.1 Coincidental Cohesion

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| Class utils.Configs | Các thành phần được đặc 1 cách ngẫu nhiên trong module | Tách ra thành các class nhỏ hơn có chung mục đích như DBConfigs, ScreenPathConfigs, UrlConfigs,…v… |
| Class Untils.Utils | Các thành phần được đặc 1 cách ngẫu nhiên trong module | Mỗi phương thức tách ra thành một lớp, như lớp Logger, Formatter, Today…MD5, nhưng có thể dẫn đến bùng nổ lớp nên có thể chấp nhận được |
| Package Utils | Các module được đặc trong cùng 1 module được đặt trong package để cho tiện. | Chấp nhận được vì những module này mang tính thủ tục |

2.2 Logical cohesion

Không

2.3 Temporal cohesion

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| Module validateDeliveryInfo trong placeOrderController | Chỉ validate cùng 1 khoảng thời gian 3 phương thức rồi sau đó không dùng nữa | Tác ra thành các lớp hoặc sử dụng design pattern |
| Modules views | 2 lớp SplashScreenHandler vs lớp HomeScreen cùng hiển thị liên tục khi khởi động app | Chấp nhận được. |

2.4 Procedural cohesion

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| Module validateDeliveryInfo | Gồm 3 thành phần thực hiện theo thứ tự các trường là validatePhoneNumber, validateName, validateAddress | Tách ra chuyển thành các lớp, đưa vào utils. |
| Module PlaceOrderController | Module validateDeliveryInfo và module calculateShipFee thực hiện theo thứ tự | Tách ra thành 1 lớp khác |
| Module PlaceRushOrder | Thực hiển việc validateExpectedDeliveryTime vs việc tính phí ship | Chuyển việc tính phí ship cho placeOrderController |

2.5 Communicational cohesion

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| InterbankSubSystemController | Gồm 2 lớp payOrder vs lớp refund hoạt động trên cùng một đầu vào input và cùng trả về 1 giao dịch. | Có thể tách ra thành 2 lớp, có thể làm bùng nổ lớp, do đó nên giữ nguyên |
|  |  |  |

2.6 Sequencial Cohesion

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| Class API | Đầu ra phương thức setUPConnection() là đầu vào của phương thức get() | Chấp nhận được |
| Module Views | CÓ input từ package Home là đầu vào package cart.  Có input từ package cart là đầu vào package shipping.  Có input từ package **invoice** là đầu vào package Payment | Chấp nhận được |

2.7 Functional Cohesion.

|  |  |  |
| --- | --- | --- |
| Module | Lý do cohesion | Sự cải tiến |
| Class API | 2 phương thức get() và setUpConnection đều liên hệ chặt chẽ với nhau và hướng đến mục đích chung | Tốt nên chấp nhận được. |
|  |  |  |

II. Nguyên lý thiết kế SOLID

1. Single Responsibility

|  |  |  |
| --- | --- | --- |
| Những Modules liên quan | Lý do không thỏa mãn | Sự cải tiến |
| InterbankSubsystemContrller | Chịu 2 trách nhiệm: Điều khiển luồng dữ liệu và chuyển đổi luồng dữ liệu. | Tách chuyển đổi luồng dữ liệu ra thành 1 lớp khác |
| PlaceOrderController | Làm thêm nhiệm vụ validate | Tách ra thành 1 class khác |

2. Open/Closed

|  |  |  |
| --- | --- | --- |
| Những Modules liên quan | Lý do không thỏa mãn | Sự cải tiến |
| PlaceOrdercontroller vs placeRushOrderController | Phương thức caculateSHipFee, khi ta muốn tính phí ships khác thì sẽ phải sửa đoạn code trên bằng một đoạn code khác | Sử dụng Stradegy Pattern |
| CreditCard vs PaymentTransaction | Vì paymentTransaction phụ thuộc vào CreditCard nên nếu ta muốn sử dụng 1 phương thức khác để thanh toán thì sẽ phải viết lại code. | Sử dụng Stradegy Pattern |

3. Liskov Substitution

|  |  |  |
| --- | --- | --- |
| Những Modules liên quan | Lý do không thỏa mãn | Sự cải tiến |
| Gồm 3 lớp CD, DVD, BOOK kế thừa lớp Media | PHương thức getAllMedia trong abstract mong muốn trả về 1 list, nhưng các lớp con của nó lại trả về NULL( Wrong abstracution) | Implement phương thức getAllMedia của lớp Media và xóa bỏ phương thức Override của lớp con |
|  |  |  |

4. Interface Segregation

|  |  |  |
| --- | --- | --- |
| Những Modules liên quan | Lý do không thỏa mãn | Sự cải tiến |
| Subsystem Interface | Có cả 2 phương thức refund vs payOrder | Tách ra thành.2 giao diện nhưng sẽ khiến tạo ra nhiều module |
|  |  |  |

5. Dependency Inversion

|  |  |  |
| --- | --- | --- |
| Những Modules liên quan | Lý do không thỏa mãn | Sự cải tiến |
| Lớp CreditCard vs Lớp PaymentTransaction | Lớp PaymentTransaction phụ thuộc vào lớp ko phải abstract là lớp creditcard (và rất chặt chẽ) | Sử dụng Stradegy Pattern tạo ra lớp abstract là PaymentCard vs lớp PaymentTransaction. Các loại thẻ sẽ kế thửa lớp PaymentCard. |