# Đánh giá vi phạm SOLID và đề xuất cải thiện

Đối với **PlaceOrderController**:

|  |  |  |  |
| --- | --- | --- | --- |
| Modules | Nguyên lý vi phạm | Lý do vi phạm | Đề xuất cải thiện |
| PlaceOrderController | SRP (Single Responsibility Principle) | Lớp PlaceOrderController có nhiều trách nhiệm: kiểm tra tính khả dụng của sản phẩm, tạo đơn hàng, tạo hóa đơn, xử lý thông tin giao hàng, tính phí vận chuyển... Điều này khiến lớp quá phức tạp và khó bảo trì. | Tách lớp thành nhiều lớp nhỏ hơn, mỗi lớp có một trách nhiệm riêng biệt. Ví dụ: tách OrderService để tạo đơn hàng, ShippingService để tính phí vận chuyển, ValidationService để xác thực thông tin. |
| placeOrder() | OCP (Open/Closed Principle) | Phương thức placeOrder xử lý trực tiếp việc kiểm tra tính khả dụng của sản phẩm, nếu thay đổi quy trình kiểm tra này, cần thay đổi mã nguồn. Điều này vi phạm nguyên lý mở/đóng. | Tạo một interface (ví dụ: ProductAvailabilityChecker) và để PlaceOrderController phụ thuộc vào interface này thay vì phụ thuộc trực tiếp vào Cart. |
| createOrder() | OCP (Open/Closed Principle) | Phương thức createOrder tạo đơn hàng trực tiếp từ Cart, nếu cách lưu trữ hoặc xử lý các sản phẩm trong giỏ thay đổi, phải thay đổi trực tiếp trong lớp này. | Sử dụng interface hoặc service để trừu tượng hóa quá trình tạo đơn hàng, giúp dễ dàng thay đổi cách xử lý mà không thay đổi mã nguồn của PlaceOrderController. |
| createInvoice() | OCP (Open/Closed Principle) | Phương thức createInvoice trực tiếp sử dụng Order để tạo hóa đơn. Nếu quy trình tính toán hóa đơn thay đổi, sẽ phải sửa mã nguồn này. | Đưa logic tạo hóa đơn vào một service riêng biệt, giúp có thể thay đổi quy trình mà không ảnh hưởng đến PlaceOrderController. |

Đối với **DeliveryForm:**

|  |  |  |  |
| --- | --- | --- | --- |
| Modules | Nguyên lý vi phạm | Lý do vi phạm | Đề xuất cải thiện |
| DeliveryForm | Single Responsibility Principle (SRP) | Lớp DeliveryForm thực hiện nhiều công việc như xử lý giao diện, kiểm tra tính hợp lệ dữ liệu, và thao tác với cơ sở dữ liệu, làm lớp trở nên phức tạp. | Tách riêng các nhiệm vụ ra thành các lớp độc lập. Ví dụ: tách phần xử lý dữ liệu từ cơ sở dữ liệu thành lớp OrderService, và phần xử lý giao diện (UI) có thể giữ nguyên trong lớp DeliveryForm. |
| DeliveryForm | Open/Closed Principle (OCP) | Lớp DeliveryForm không dễ dàng mở rộng mà không thay đổi mã nguồn hiện tại (ví dụ: thay đổi logic tính phí hoặc kiểm tra tính hợp lệ của thông tin). | Tách logic nghiệp vụ (như tính phí giao hàng) ra khỏi lớp DeliveryForm. Sử dụng các dịch vụ hoặc chiến lược để mở rộng mà không thay đổi mã hiện tại, giúp lớp này dễ dàng mở rộng mà không cần sửa lại mã cũ. |
| DeliveryForm | Liskov Substitution Principle (LSP) | Không có vi phạm rõ ràng đối với nguyên lý này trong mã nguồn hiện tại vì không sử dụng kế thừa, do đó không có vấn đề thay thế lớp con với lớp cha. | Không có cải tiến cần thiết cho nguyên lý này vì mã nguồn không sử dụng kế thừa. |
| DeliveryForm | Interface Segregation Principle (ISP) | Lớp DeliveryForm kết hợp quá nhiều giao diện như xử lý nhập liệu từ người dùng và thao tác với cơ sở dữ liệu. Điều này khiến lớp này không tuân thủ ISP. | Tách giao diện thành các phần nhỏ hơn. Ví dụ, có thể có các giao diện khác nhau như ShippingInfoProcessor hoặc RushInfoHandler để xử lý các chức năng cụ thể mà không làm lớp DeliveryForm bị phình to. |

Phân tích **PaymentController:**

|  |  |  |  |
| --- | --- | --- | --- |
| Modules | Nguyên lý vi phạm | Lý do vi phạm | Đề xuất cải thiện |
| PaymentController | Single Responsibility Principle (SRP) | Lớp PaymentController thực hiện nhiều nhiệm vụ như điều khiển quá trình thanh toán, xử lý kết quả giao dịch và quản lý giỏ hàng. Điều này làm lớp phức tạp và khó duy trì. | Tách lớp PaymentController thành các lớp con chuyên trách hơn: một lớp chỉ xử lý thanh toán (ví dụ: PaymentProcessingService), và một lớp riêng xử lý việc cập nhật giỏ hàng (ví dụ: CartManager). |
| PaymentController | Open/Closed Principle (OCP) | Lớp PaymentController chưa dễ dàng mở rộng mà không phải thay đổi mã nguồn hiện tại. Nếu thêm dịch vụ thanh toán khác (ví dụ: PayPal), cần phải thay đổi mã hiện tại. | Đảm bảo tính mở rộng cho các dịch vụ thanh toán bằng cách sử dụng các lớp trừu tượng hoặc giao diện, cho phép thêm dịch vụ thanh toán mới mà không thay đổi mã trong PaymentController. |
| PaymentController | Liskov Substitution Principle (LSP) | Không có vi phạm rõ ràng đối với nguyên lý này trong mã nguồn hiện tại, vì lớp PaymentController không sử dụng kế thừa. | Không có cải tiến cần thiết vì mã nguồn không sử dụng kế thừa. |
| PaymentController | Interface Segregation Principle (ISP) | Lớp PaymentController có thể vi phạm ISP khi thực hiện các công việc không liên quan như xử lý kết quả giao dịch và xử lý giỏ hàng. Điều này khiến lớp trở nên nặng nề. | Tách giao diện thành các phần nhỏ hơn để lớp chỉ tập trung vào một nhiệm vụ cụ thể. Ví dụ: có thể tách việc xử lý kết quả thanh toán thành một giao diện riêng, giúp lớp PaymentController chỉ tập trung vào việc điều khiển quy trình thanh toán. |