1. **Coupling và Cohesion**
2. Coupling
   1. Content coupling

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm  Truy xuất data qua các getter và setter |  |

* 1. Common coupling

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm  Sử dụng ngôn ngữ Java không có biến global |  |

* 1. Control coupling

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm  Các funtion không có tham số điều khiển |  |

* 1. Stamp coupling

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Có thể vi phạm  Các tham số trong funtion là HashMap<string, string> có thể bao gồm những dữ liệu thừa trong tham số truyền vào | Có thể chấp nhận được |

* 1. Data coupling

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Chưa đạt đến mức này |  |

1. Cohesion
   1. Coincidental cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm  Các elenments trong component có liên quan đến nhau |  |

* 1. Logical cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Vi phạm  Các funtion validate liên quan đến nhau về mặt logic nhưng không liên quan đến nhau về mặt chức năng | Chuyển sang một class khác là class Validate trong package Utility |

* 1. Temporal cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm |  |

* 1. Procedural cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm |  |

* 1. Communicational cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm |  |

* 1. Sequential Cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm |  |

* 1. Functional cohesion

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm |  |

1. **SOLID**
2. Single Responsibility Principle

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Vi phạm  PlaceRushOrder chỉ nên có các funtion liên quan đến việc tạo và thanh toán. | Các funtion liên quan đến validate hay tính phí ship nên ở một class khác. |

1. Open/Closed Principle

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Vi phạm  Khi tính phí ship mà muốn thay đổi phí ship cho từng sản phẩm thì phải sửa code. | Config phí ship ở appsettings có thể thay đổi trong runtime, hoặc thêm tham số khi truyền vào funtion. |

1. Liskov Substitution Principe

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Vi phạm  Class PlaceRushOrderController không kế thừa bất kì một class cha nào cả. Gây khó khăn cho việc mở rộng. | Implement một interface hoặc một abtract có các funtion cần thiết để class khác có thể giao tiếp. |

1. Interface Segregation Principle

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Không vi phạm  Không implement bất kỳ interface nào. |  |

1. Dependency Inversion Principle

|  |  |  |
| --- | --- | --- |
| Related modules | Description | Improvement |
| PlaceRushOrder | Vi phạm  Vì không implement bất cứ interface nào nên nếu class khác muốn sử dụng các funtion class này sẽ bị phụ thuộc trực tiếp vào class này. | Implement một interface IPlaceRushOrderController để dễ dàng cho các class khác giao tiếp. |

1. **Yêu cầu bổ sung**
2. Thay đổi phí ship

Phải sửa lại funtion caculateShippingFee(…), vi phạm nguyên lý Open/Closed principle.

Cải thiện bằng cách tách ra một class riêng implement một interface có funtion là caculateShippingFee(…), sau đó class PlaceRushOrderController sẽ được injection interface đó vào. Nếu muốn thay đổi cách tính phí ship thì sẽ thêm một class khác và config lại dependency injection.

1. Thêm một cách tính tiền mới

Thêm funtion mới vào class cũ, vi phạm nguyên lý Open/Closed principle.

Thêm một class mới extend lại class tính tiền cũ. Thêm các phương thức cho cách tính tiền mới vào class mới và chỉnh sửa lại dependency injection từ class cha sang class con.

1. Sử dụng interbank khác

Thay đổi class API và class Interbank, vi phạm nguyên lý Dependency inversion principle vì class Interbank phụ thuộc trực tiếp vào class API, vi phạm nguyên lý Open/Closed principle vì phải sửa code class API.

Để class API implement một interface có các phương thức post, get. Interbank phụ thuộc vào interface vừa rồi. Khi đổi cách thức gọi API thì sẽ thay đổi class implement interface vừa rồi.

Vì lớp Interbank đang implement InterbankInterface nên chỉ cần tạo lớp Interbank mới implement lại InterbankInterface và thay đổi dependency injection.