DESIGN PRINCIPLES

# Single Responsibility Principle

|  |  |  |  |
| --- | --- | --- | --- |
| # | Related modules | Description | Improvement |
| 1.1 | InterbankSubsystemController | It is responsible for 02 tasks: data flow control and data conversion | Divide into 2 classes which are responsible for 2 tasks. |
| 1.2 | PlaceOrderController | It is responsible for 3 tasks: create order, validate delivery information and calculate shipping fee | Divide into:  - 2 interfaces for creating order and validating delivery information  - 1 interface ShippingFeeCalculator |
| 1.3 | PlaceRushOrder  Controller | It is responsible for 3 tasks: create order, validate delivery information and calculate shipping fee | Divide into:  - 2 interfaces for creating order and validating delivery information  - 1 interface ShippingFeeCalculator |

# Open/Closed Principle

|  |  |  |  |
| --- | --- | --- | --- |
| # | Related modules | Description | Improvement |
| 2.1. | PlaceRushOrderController,  calculateShippingFee method | Modify the old codes in PlaceOrderController when changing the formula of shipping fee for Rush Order | create an interface ShippingFeeCalculator and let the controller depends on it |
| 2.2 | PlaceRushOrderController,  processDeliveryInfo method | Modify the old codes in PlaceOrderController when changing the validation of delivery information | Create an interface DeliveryInfoValidation and let controller implements it |

# Liskov Substitution Principle

|  |  |  |  |
| --- | --- | --- | --- |
| # | Related modules | Description | Improvement |
| 3.1. | Method getAllMedia() in Media | Method return List but children classes override and return null | Delete method in children classes |

# Interface Segregation Principle

|  |  |  |  |
| --- | --- | --- | --- |
| # | Related modules | Description | Improvement |
| 4.1 | InterbankSubSystem | payment subsystem for a payment actor is expected to have the two operations payOrder() and refund() | Put 2 methods into the same interface InterbankInterface |

# Dependency Inversion Principle

|  |  |  |  |
| --- | --- | --- | --- |
| # | Related modules | Description | Improvement |
| 5.1. | PaymentTransaction and CreditCard | Cannot adding new type of card without modifying PaymentTransaction | Make an abstract class as parent of all other types of payment card |
| 5.2 | PlaceOrderController, PlaceRushOrderController and ShippingFeeCalculator | Cannot change new formula to calculate shipping fee without modifying codes | Make an abstract class as parent of all types of calculating shipping fee |