DESIGN PRINCIPLES

1. 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.

2. Open/Closed Principle

#	Related modules	Description	Improvement
2.1.	ReturnBikeController, calculateAmount method	Modify the old codes in ReturnBikeController when changing the formula of rental pricing	create an interface CalculateRentalPricing and let the controller depends on it
2.2	TransactionController, validateCard method	Modify the old codes in TransactionController when changing the card information validation	Create an interface PaymentInfoValidate and let controller implements it
.3	ViewBikeController, setBike method	Modify the old codes in ViewBikeController when having a new type of bike	Make an abstract class as parent of all other types of bike

3. Liskov Substitution Principle

#	Related modules	Description	Improvement
3.1.	Method getAllBike() in Bike	Method return List but children classes override and return null	Delete method in children classes

4. 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

5. Dependency Inversion Principle

#	Related modules	Description	Improvement
5.1.	TransactionInfo and Card	Cannot adding new type of card without modifying PaymentTransaction	Make an abstract class as parent of all other types of payment card
5.2	ReturnBikeController and calculateAmount method	Cannot change new formula to calculate rental pricing without modifying codes	Make an abstract class as parent of all types of calculating rental pricing