**SOLID PRINCIPLES REVIEW**

1. **Single responsibility principle**

The class Utils violated this principle since it handle variety of responsible: Convert integer into currency string, get the current date, encrypting message…

1. **Open/closed principle**

New screen can be added by inherit the class BaseScreenHandler, new controller can inherit BaseController, new type of making payment transaction can be added by building a new subsystem for it, new type of bike can inherit the abstract class Bike => The code can be extend without modify the old ones.

1. **Liskov substitution principle**

The child class can replace the parent class without any unwanted behavior that violate the design of the parent class. => The principle is not violated.

1. **Interface segregation principle**

Currently, the only interface in the project only have one method so any class that implement this interface should have no side effects or forced implementation => The principle is not violated.

1. **Dependency inversion principle**

Some class has been made into abstract or interface class for further use and extension. The dependency of these class are only stop at the abstract/interface level so it’s safe to say that this principle is not violated.