SOLID

Sets of 5 principles, it is a foundation of LLD, it helps/guides developers to write maintainable, extensible, clean code.

S – Single responsibility principle (SRP)

O – Open-Close principle (OCP)

L – Liskov substitution principle (LSP)

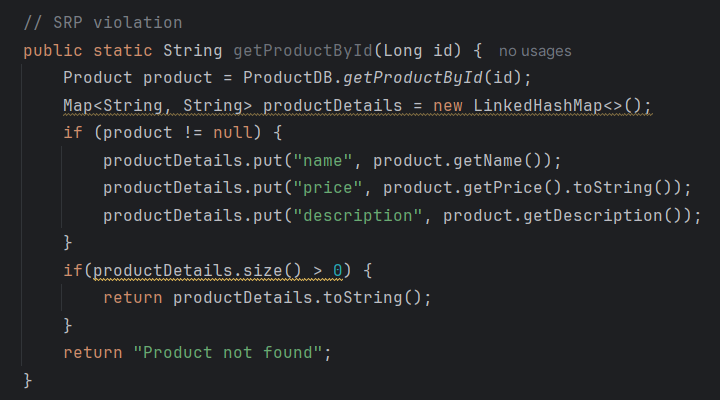
I – Interface segregation principle (ISP)

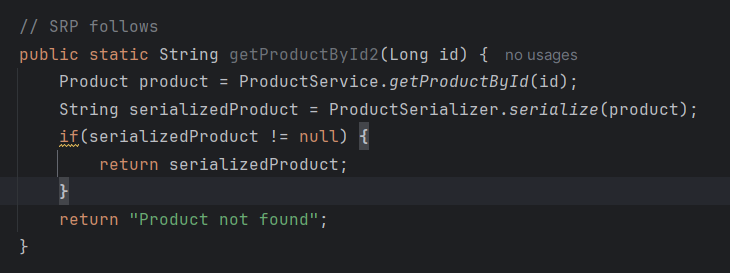
D – Dependency inversion principle (DIP)

Single responsibility principle (SRP)

SRP states that a class or module should have one and only one reason to change.

One should handle single responsibility instead of multiple responsibility.

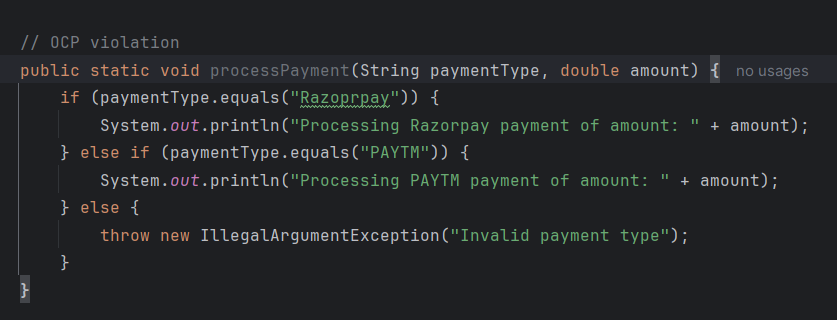


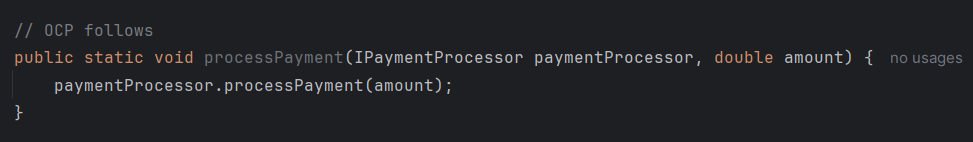


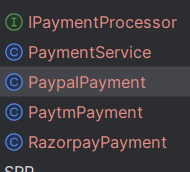
Open-Close principle (OCP)

OCP says that your code should be open for extension but closed for modification.

Adding new feature should fine but making too much changes to the existing code should not be accepted.



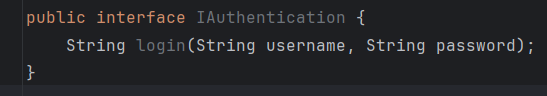


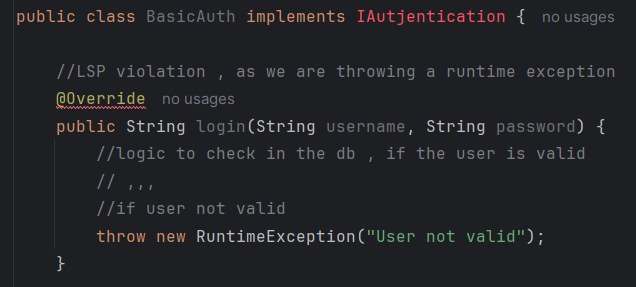


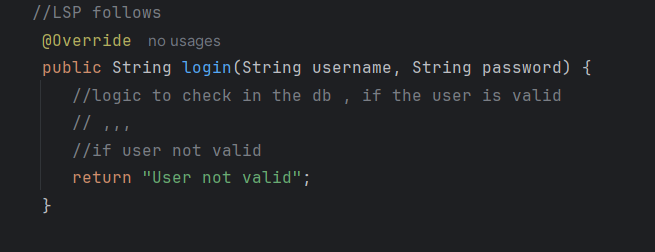
Liskov substitution principle (LSP)

LSP says that derived class should be substitutable for their base class without altering the correctness of the program.

Or child class object should be substitutable on the parent class object without making any changes/correction.





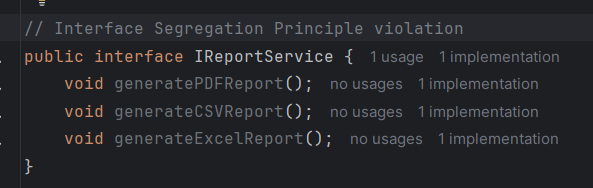


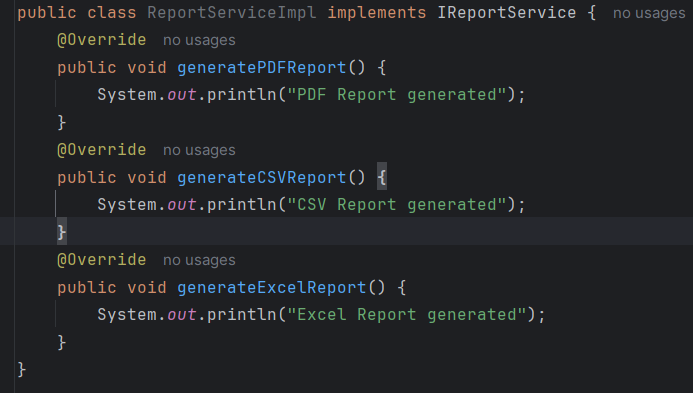
Interface segregation principle (ISP)

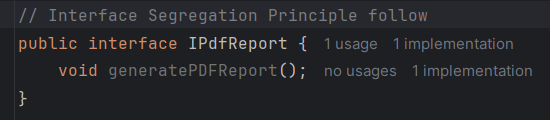
ISP states that a class should not be forced to implement interfaces it does not use.

Means, multiple smaller, specific interfaces are better than a single large, general-purpose one.

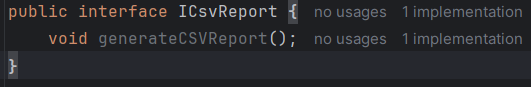
So, it tells interface should have minimum numbers of methods.













Dependency inversion principle (DIP)

DIP says that the high-level-modules should not depends on low-level-modules, Instead both should depends on abstraction.

Means, no two concrete implementation should be directly dependent on each other, Instead they should be depend on each other via a Abstract class or Interface.

