LLD Day-07

Statergy Design Pattern : It is a Behavioural Design Pattern that enables you to define a family of algorithms, encapsulate each one, and make them interchangeble. The algorithm can vary independently from the clients that use it. Use cases involve

- Payment Methods (Credit Card, UPI, Paypal)
- Sorting Algorithms (Quick sort, Bubble Sort, ----)

Driver Management / Allocation

Use Case

↳ please find in next page

Tier-1 City Tier-2 Tier-3

- ▨ – person
- o – options

Driver Matching :

(Tier-1) ⇒ Rating of Driver, No of Rides, Rating of Rider.

(Tier-3) ⇒ Location.

Some times in Tier-1 also if it is raining, only location will be taken care of. So, they are so many options. So, our program should allocate ~~some~~ correct statergy for driver matching in run time.

Rating Based Algo
Locatn Based Algo
No of Rides Based Algo

Run Time

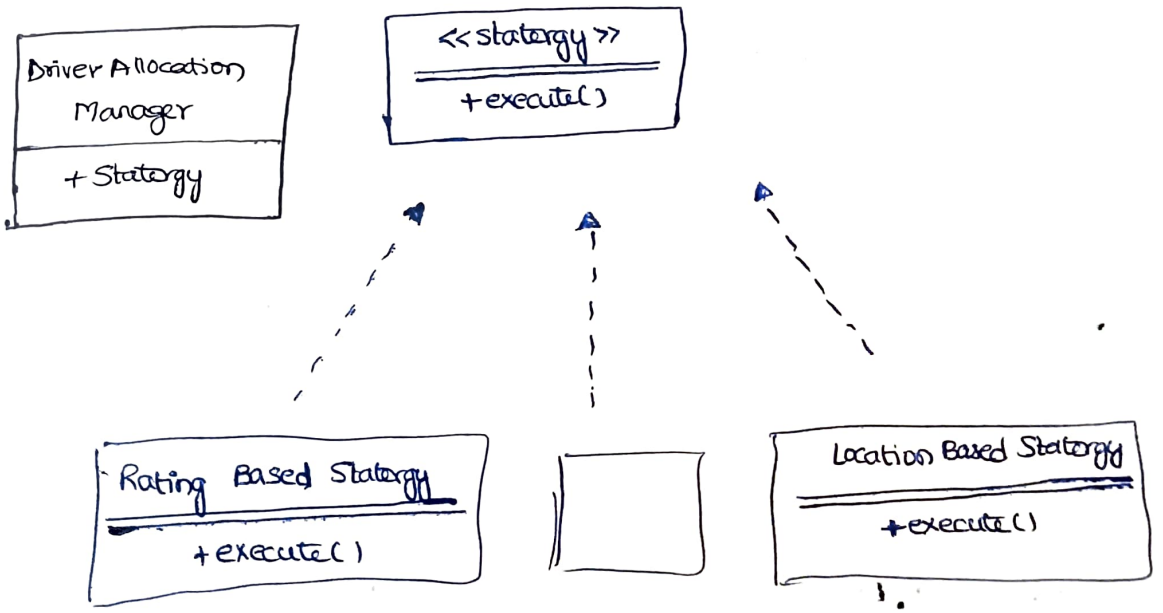
Condrions
+
weather conditions
rating conditions

- Initial Pseudo Code :

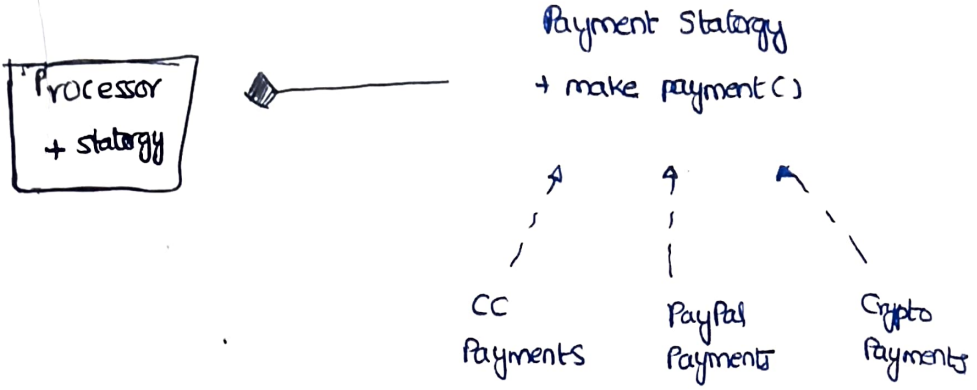
```
Driver Allocation Manager {

    Driver Match Drivers ( _____ ) {

        if (location == Tier 1) {

            return ____ //Algo

        }

        else-if ( _____ ) {

        }

        .
        .
        .
```

But Violates Basic SOLID Principals.

So now atleast to maintain some of SOLID, we redesign it as below :



- check payment ~~statergy~~ code in github.



- (3) Components of Strategy Design Pattern
  - Context, Strategy interface, concrete strategy
    - Context → obj (behaviour) → dynamic / runtime
    - Strategy interface → common methods
    - concrete strategy → Actual Implementation of Algo

Strategy + Factory Design Pattern :

problem	strategy	Strategy + Factory
client knows concrete classes	⊗ Yes	☑ No
object creation logic	⊗ Client	☑ Factory
Easy to Add Payment Method	⊗ Hard	☑ Easy
Open-Close Principal.	⊗ No	☑ Yes

Statergy Design Pattern defines how an ~~a+~~ action is performed, while Factory Pattern decides which statagy to instantiate. Using both together results in a clean, extensible, and loosely coupled design

- See code in github (S+F) DP.