Rachit Manglik • 2nd
Software engineer 2 | C# | .NET 8 | MVC | MySQL | REST API | jQuery | Apache Kafka | RabbitM...
3d •

Follow

🚀 **What is Strategy design Pattern** 🤯 (**Behavioral** Design Pattern)

"Instead of putting all the logic in one class, **put each behavior** (**or algorithm**) **in its own class and use a interface shared among all classes to switch between them easily**."

🚀**Why do we need Strategy design Pattern** 😎

```
if (paymentType == "CreditCard")
{
 // credit card logic
}
else if (paymentType == "Cash")
{
 // cash logic
}
else if (paymentType == "UPI")
{
 // upi logic
}
```

Looks okay at first. But **what happens when we want to add Net Banking**, **Wallet**, **or other types**?🤔🤯

• It's common to put all logic in **one big class**. You often use **lots of** `if-else` or `switch` statements to handle different behaviors.

• But as the project **grows** and you keep adding more conditions as a result the code becomes **messy**, **hard to read**, **and difficult to maintain**.

🚀 **How does the strategy pattern solves this**

• 📦 **Wrap each algorithm in its own class**

• 🔁 **Swap behaviors at runtime** Need different logic on the fly? Just inject a new strategy, no code rewrite needed.

• 🔒 **Keeps the algorithm details separate** Your core code stays focused as strategies can handle their own logic.

• 📦 **Prefer composition over inheritance** No rigid parent- child class structure, making the whole structure loosely coupled.

• 🚪 **Open/Closed Principle in action** Add new strategies by extending the code instead of modifying it.

What's Next?

In the next step, I will explain the **application**, **implementation**, **pros**, and **cons** of the **Strategy Pattern**.

Stay tuned! 🚀

Follow me for more useful insights on design patterns.

🚀 **Check out my previous posts on design patterns**:

• **Singleton Pattern**: https://lnkd.in/gcqa6fHt

• **Factory Pattern**: https://lnkd.in/gp4PtxsD

🚀🚀 #StrategyPattern #DesignPatterns #SOLIDPrinciples #SoftwareEngineering #CSharp #DotNet #CleanCode #OOP #Coding #Developers #SoftwareDevelopment