1. What is your understanding of the term “Design Patterns”?
Provide a description in your own words.

Design patterns are ways of solving common problems that are faced when writing software. Instead of solving problems from scratch every time, we can use design patterns — smart “blueprints” that solve them in a clean and organized way. Design patterns are like guidelines or structures that can help to write cleaner code.

2. Explain the MVC Pattern

1) What does MVC stand for?
2) Explain the pattern in detail.
3) What are some use cases for this framework?

1)
MVC stands for:
M - Model, V - View, C - Controller

2)
The MVC pattern is a way to organize code by dividing it into three parts. Each part has its own job, and they work together to make the application run smoothly.

The Model handles the data and the rules about how that data works.
The View is what the user sees — it shows the data to the user.
The Controller is in charge of reacting to user actions and updating the Model or View accordingly.

3)
anywhere we want to separate UI, logic, and data clearly.

3. List three other design patterns

Provide names and details for three additional design patterns.

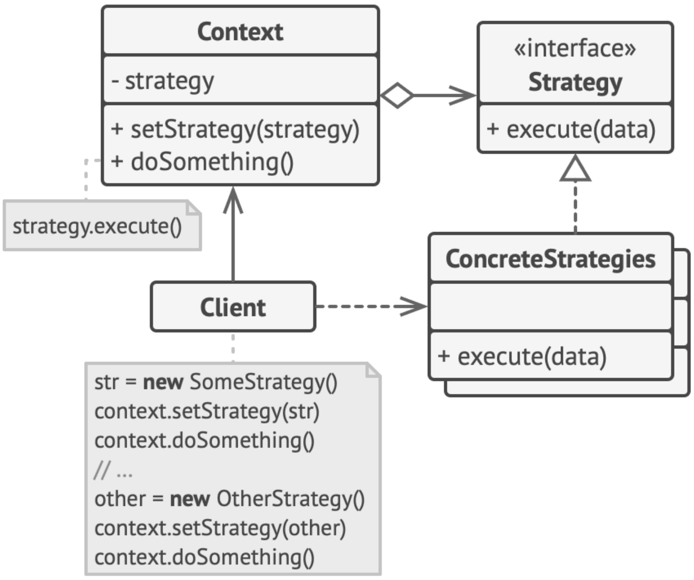
Explain how you have used those patterns in the past and how they have solved your problem

Use diagrams to explain the design patterns.

Strategy pattern:

It lets us define different ways of doing something, and switch between them without changing the rest of the code.

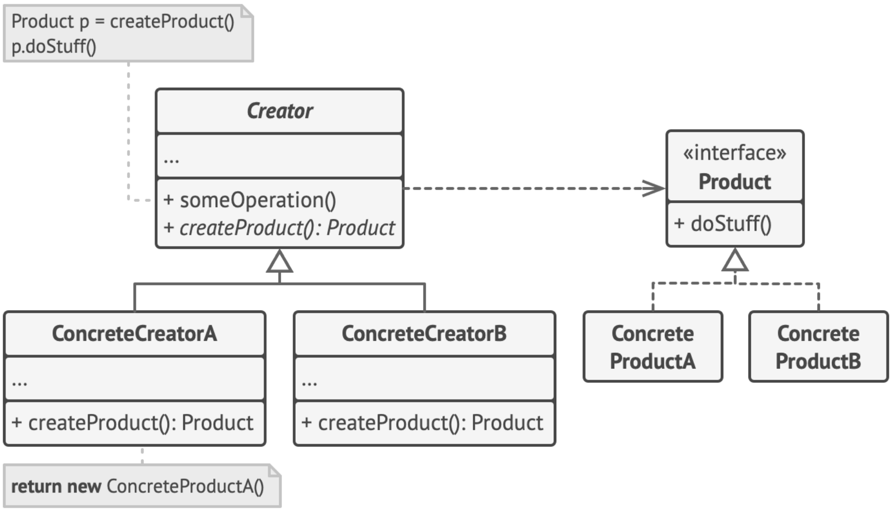
It helped me when I wanted my objects to have different behaviors depending on their properties.



Factory method:

The design pattern used to create objects without specifying the exact class of object that will be created.

During my studies in uni I have had multiple tasks for creating objects like Circle, Square, or Rectangle from a menu. Instead of writing long if/else or switch statements throughout the program, I used a single factory method that returned the right shape based on a name.



Template method:

It is used to define the basic steps of an algorithm in a base class, but let subclasses fill in some specific details.

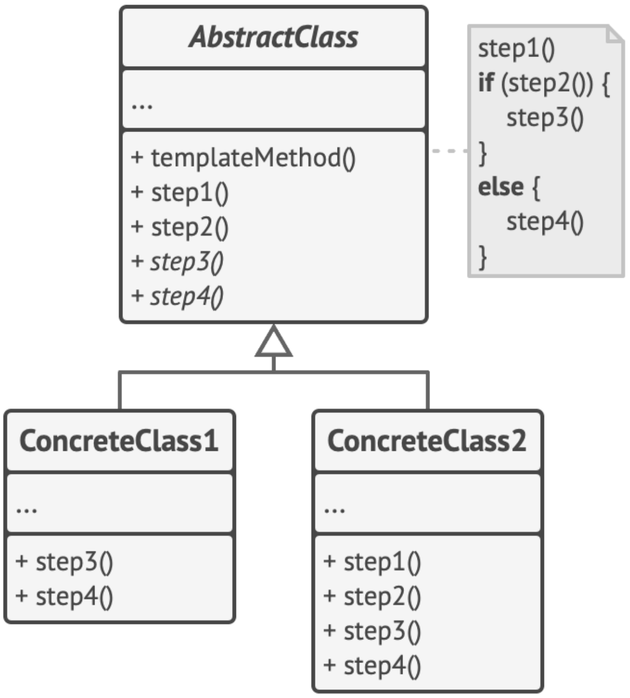
I used the Template Method to define a general process:

Load data

Process it

Export results

Different types of data processors inherited the same base class but provided their own versions of loadData() and exportResults().


AbstractClass
...
+ templateMethod()
+ step1()
+ step2()
+ step3()
+ step4()
step1()
if (step2()) {
step3()
}
else {
step4()
}
ConcreteClass1
...
+ step3()
+ step4()
ConcreteClass2
...
+ step1()
+ step2()
+ step3()
+ step4()