**Exercise 2: Implementing the Factory Method Pattern**

**Scenario:**

You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

**What is a Factory Method Pattern?**

Factory Method Design Pattern define an interface for creating an object, but let subclass decide which class to instantiate. Factory Method lets a class defer instantiation to subclass.

It’s a design pattern used to create objects without specifying the exact class name.

**When to use Factory Method Design Pattern?**

Factory method design pattern can be used in java in following cases:

* A class cannot predict the type of objects it needs to create.
* A class wants its subclasses to specify the objects it creates.
* Classes delegate responsibility to one of multiple helper subclasses, and you aim to keep the information about which helper subclass is the delegate within a specific scope or location.

**Real-Life Analogy**

Imagine a **document creation kiosk**:

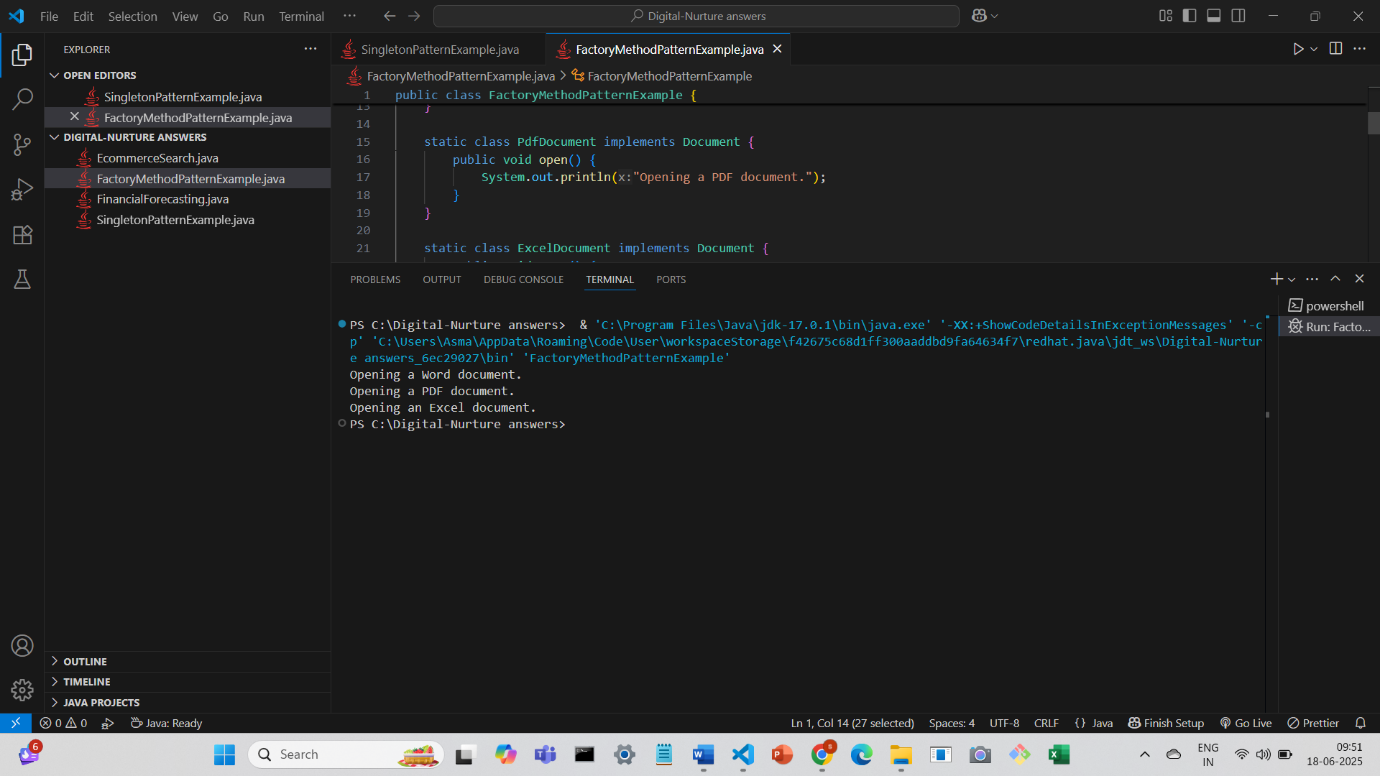
* If the user presses "Create Word", the machine (factory) creates a Word document.
* If they press "Create PDF", the machine creates a PDF.

You’re not telling the machine how to make it — you just tell it what to make.

**Why This is Useful**

* Your code doesn't depend on concrete classes like new WordDocument().
* You can switch from PDF to Excel just by changing the factory, without changing logic.

**Output Screenshot:**

****

**Conclusion:**

Instead of doing new PdfDocument(),You let the factory method decide what object to create. The "factory method" is the method createDocument() in your factory classes — it's responsible for creating the right object, and follows the Factory Method Pattern.