Factory Method Pattern – Java

Exercise Title:Implementing the Factory Method Pattern
Module:Design Patterns and Principles
Track:DN 4.0 DotNet FSE Deep Skilling Program

Objective
To design a flexible document creation system that can generate different document types (e.g., Word, PDF, Excel) using the *Factory Method Pattern, promoting **loose coupling* and *scalability* in code.

Concepts Applied
* Object-Oriented Programming (OOP)
* Design Patterns: Factory Method Pattern
* Polymorphism and Inheritance
* Abstraction for flexible object creation

Problem Summary
Design a document management system where:
* A common interface or abstract class defines the structure of a document.
* Concrete classes implement the structure for specific types of documents (e.g., Word, PDF, Excel).
* A factory method determines which document object to instantiate at runtime.
* This supports *open/closed principle*—easy to add new document types without modifying existing code.

Deliverables
* An abstract class or interface Document
* Concrete classes: WordDocument, PdfDocument, ExcelDocument implementing Document
* Abstract factory: DocumentFactory with method createDocument()
* Concrete factories: WordFactory, PdfFactory, ExcelFactory
* A test class that creates and demonstrates use of each document via their factory

Tools & Technologies
* Java
* IDE (IntelliJ, Eclipse, or VS Code)

Evaluation Criteria
* Correct use of the Factory Method Pattern
* Clear separation of interface and implementation
* Scalable design supporting new document types easily
* Code structure is modular and maintainable
* Test class validates the creation and use of all document types
* Project structured in week-wise folders and committed to GitHub

Self-Evaluation Checklist
✔ Interface or abstract base class for documents implemented
✔ Concrete classes for Word, PDF, and Excel extend the base type
✔ Factory method defined in an abstract class and implemented in concrete factories

✔ Test class successfully demonstrates polymorphic creation of documents
✔ Folder structure is organized (Week1/FactoryMethodPatternExample)
✔ Code compiled and tested successfully
✔ Project uploaded to public GitHub repository