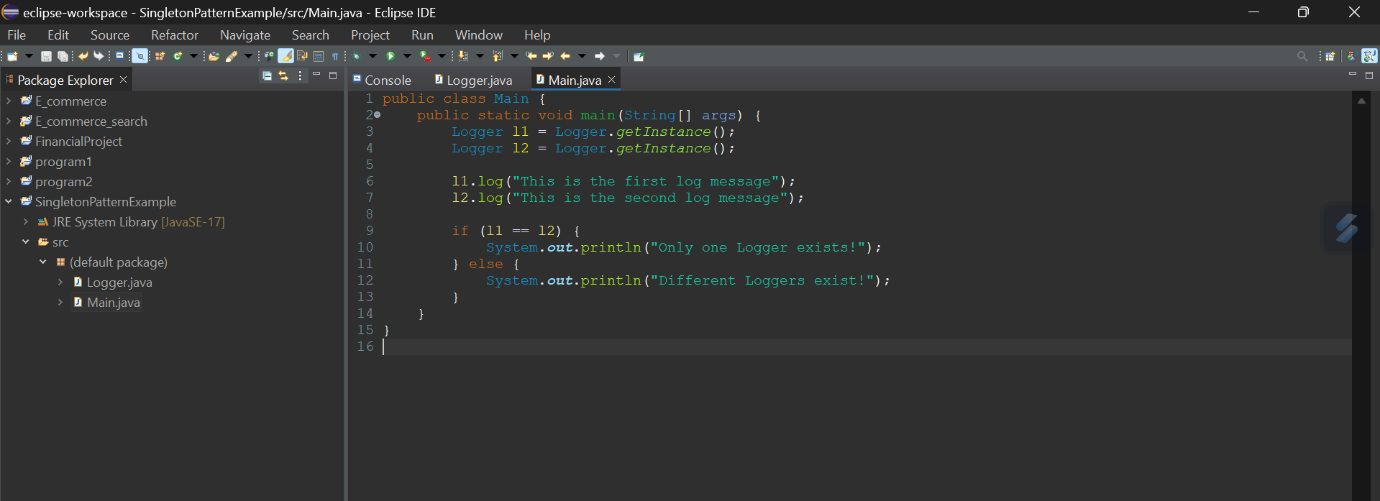
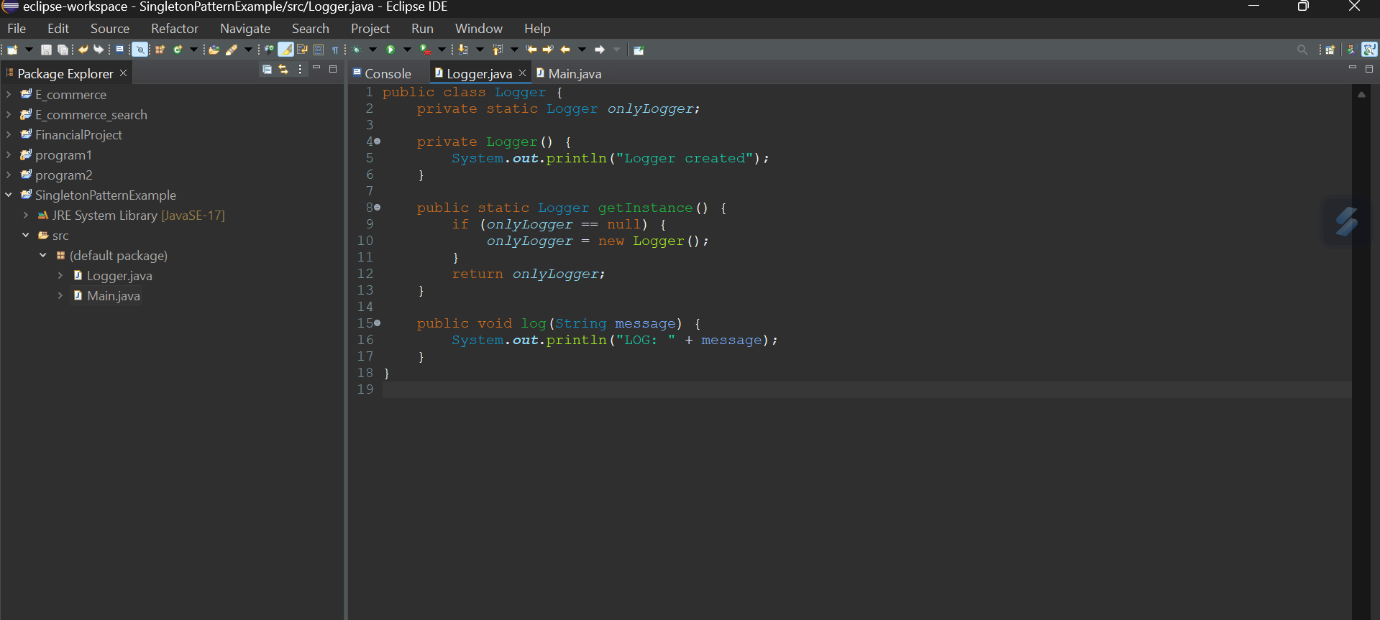
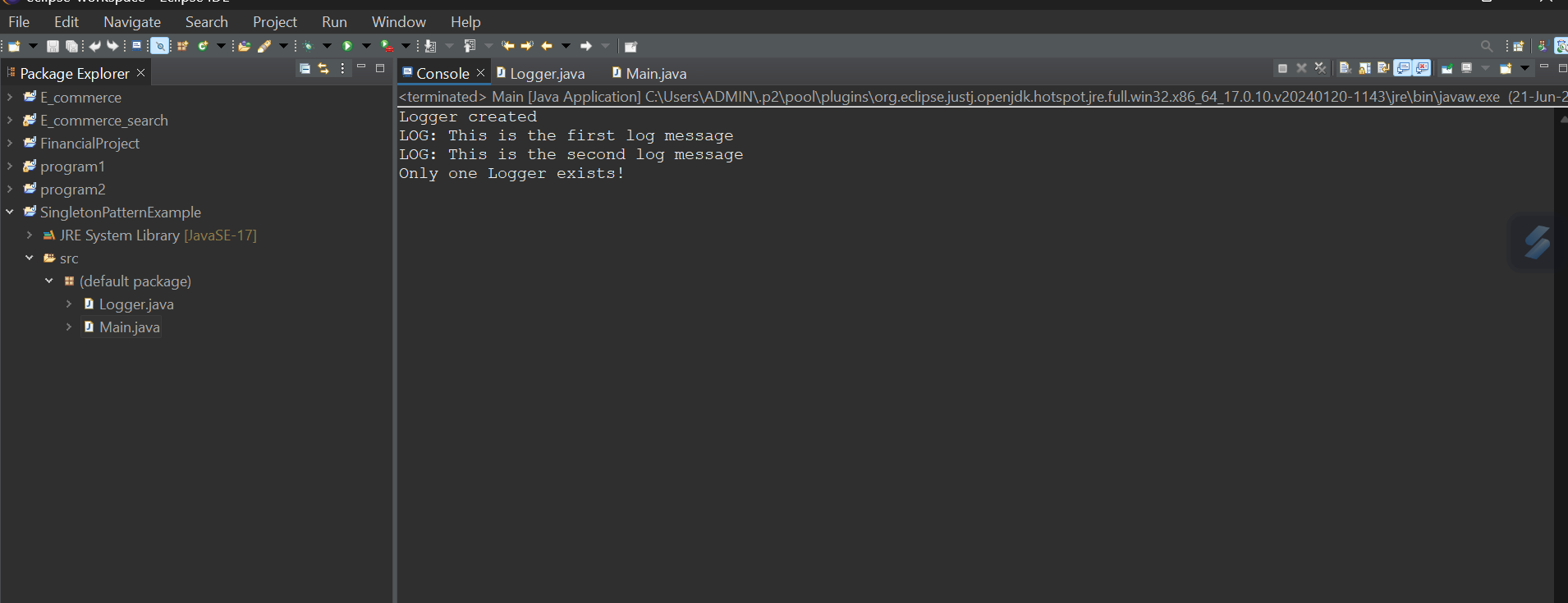
*DESIGN PATTERNS AND PRINCIPLES*

***********Exercise 01:* *Implementing the Singleton Pattern***



***CODE:***

***Logger.java***

*public class Logger {*

*private static Logger onlyLogger;*

*private Logger() {*

*System.****out****.println("Logger created");*

*}*

*public static Logger getInstance() {*

*if (onlyLogger == null) {*

*onlyLogger = new Logger();*

*}*

*return onlyLogger;*

*}*

*public void log(String message) {*

*System.****out****.println("LOG: " + message);*

*}*

*}*

***Main.java***

*public class Main {*

*public static void main(String[] args) {*

*Logger l1 = Logger.getInstance();*

*Logger l2 = Logger.getInstance();*

*l1.log("This is the first log message");*

*l2.log("This is the second log message");*

*if (l1 == l2) {*

*System.****out****.println("Only one Logger exists!");*

*} else {*

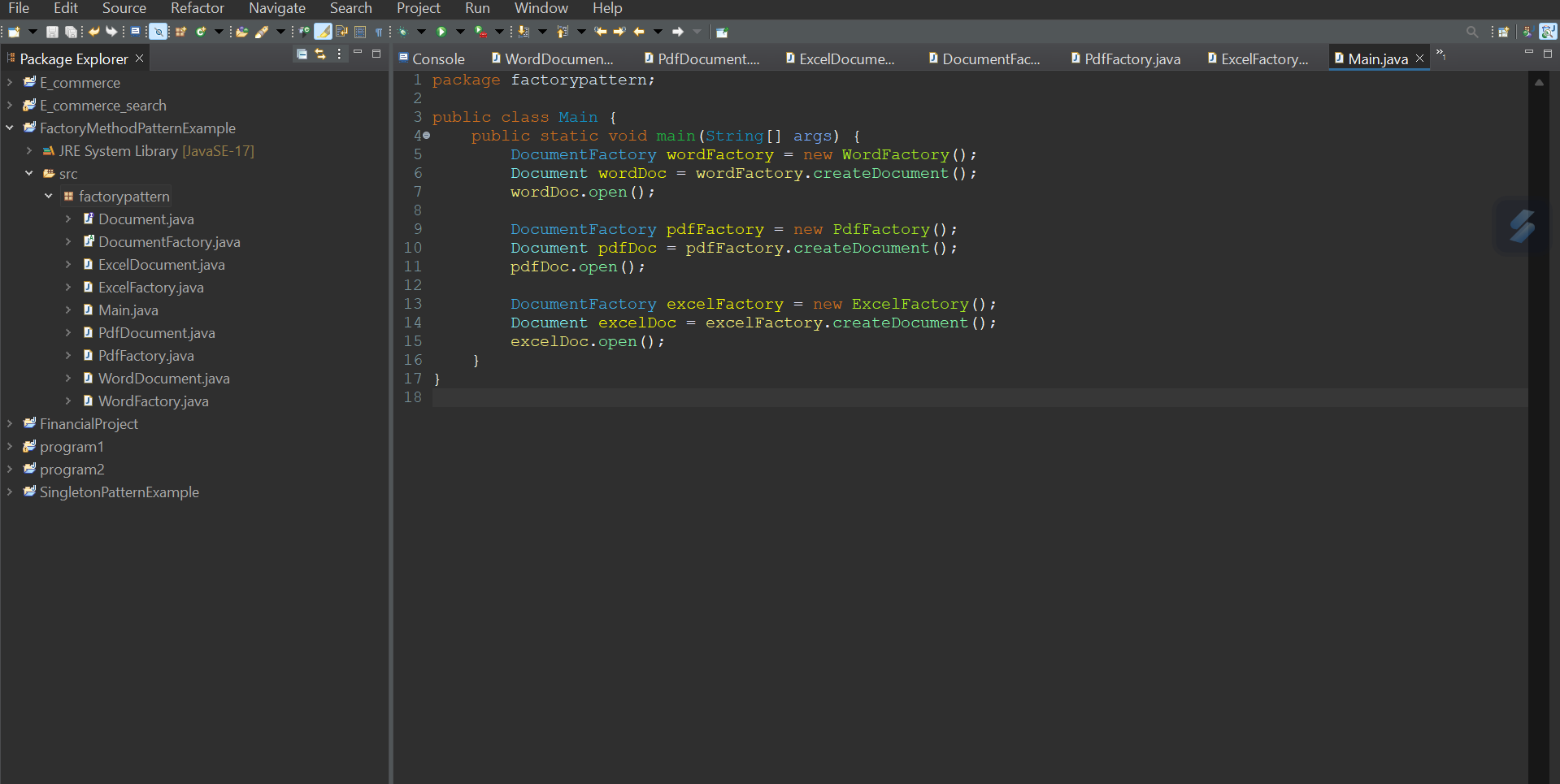
*System.****out****.println("Different Loggers exist!");*

*}*

*}*

*}*

***Exercise 02: Implementing the Factory Method Pattern***

**

***Main.java***

*package factorypattern;*

*public class Main {*

*public static void main(String[] args) {*

*DocumentFactory wordFactory = new WordFactory();*

*Document wordDoc = wordFactory.createDocument();*

*wordDoc.open();*

*DocumentFactory pdfFactory = new PdfFactory();*

*Document pdfDoc = pdfFactory.createDocument();*

*pdfDoc.open();*

*DocumentFactory excelFactory = new ExcelFactory();*

*Document excelDoc = excelFactory.createDocument();*

*excelDoc.open();*

*}*

*}*

***Document.java***

*package factorypattern;*

*public interface Document {*

*void open();*

*}*

***WordDocument.java***

*package factorypattern;*

*public class WordDocument implements Document {*

*public void open() {*

*System.out.println("Opening Word Document");*

*}*

*}*

***PdfDocument.java***

*package factorypattern;*

*public class PdfDocument implements Document {*

*public void open() {*

*System.out.println("Opening PDF Document");*

*}*

*}*

***ExcelDocument.java***

*package factorypattern;*

*public class ExcelDocument implements Document {*

*public void open() {*

*System.out.println("Opening Excel Document");*

*}*

*}*

***DocumentFactory.java***

*package factorypattern;*

*public abstract class DocumentFactory {*

*public abstract Document createDocument();*

*}*

***WordFactory.java***

*package factorypattern;*

*public class WordFactory extends DocumentFactory {*

*public Document createDocument() {*

*return new WordDocument();*

*}*

***PdfFactory.java***

*package factorypattern;*

*public class PdfFactory extends DocumentFactory {*

*public Document createDocument() {*

*return new PdfDocument();*

*}*

*}*

***ExcelFactory.java***

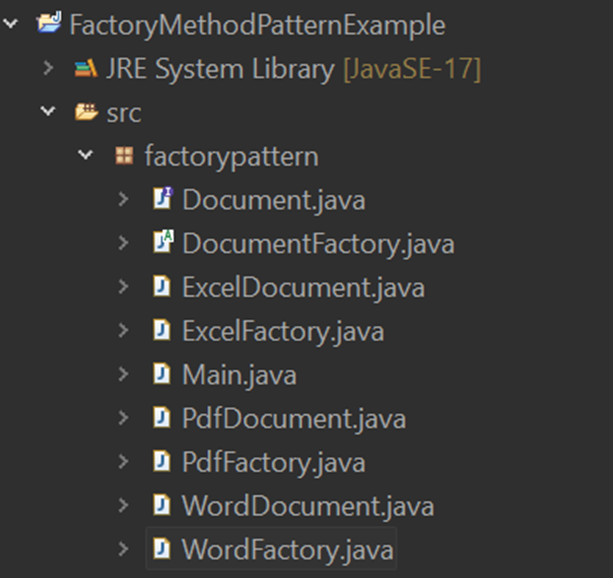
*package factorypattern;*

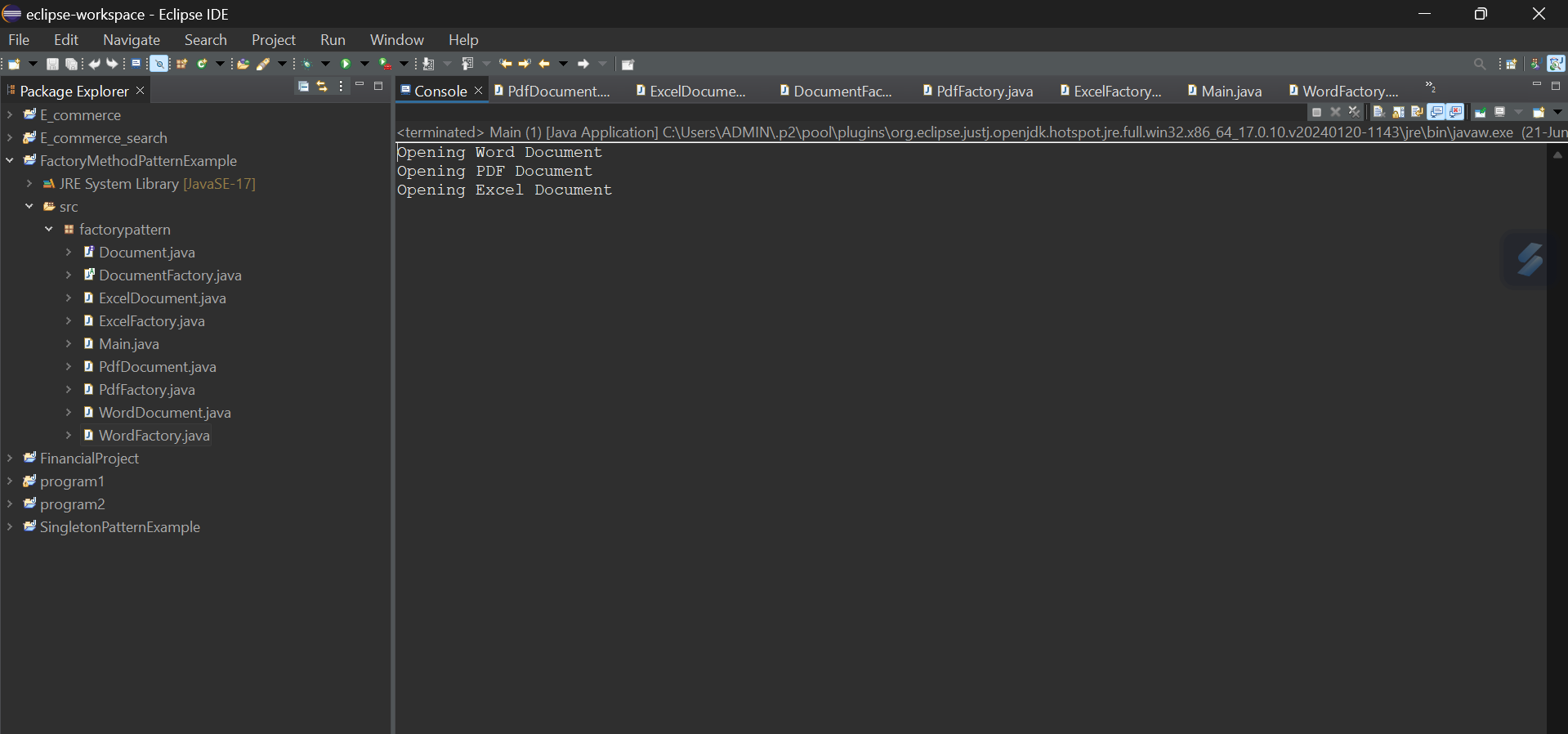
*public class ExcelFactory extends DocumentFactory {*

*public Document createDocument() {*

*return new ExcelDocument();*

*}}*

**



*In this project, I implemented the Factory Method Design Pattern to create a simple Document Management System that supports Word, PDF, and Excel files. I created a common interface Document, with each document type providing its own open() method. Then I defined an abstract class DocumentFactory and extended it through WordFactory, PdfFactory, and ExcelFactory to generate specific document types. In the Main class, I used each factory to create and open the corresponding document. This approach made the code more flexible and easier to extend, while following the structure expected by the Factory Method pattern.*