**1. Implementing the Singleton Pattern**

**Project : SingletonPatternExample**

**Code:**

**Main.java**

public class Main {

    public static void main(String[] args) {

        Logger logger1 = Logger.getInstance();

        Logger logger2 = Logger.getInstance();

        logger1.log("Logging first message");

        logger2.log("Logging second message");

        if (logger1 == logger2) {

            System.out.println("Same logger instance is used.");

        } else {

            System.out.println("Different instances detected.");

        }

    }

}

class Logger {

    private static Logger instance;

    private Logger() {

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

    }

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

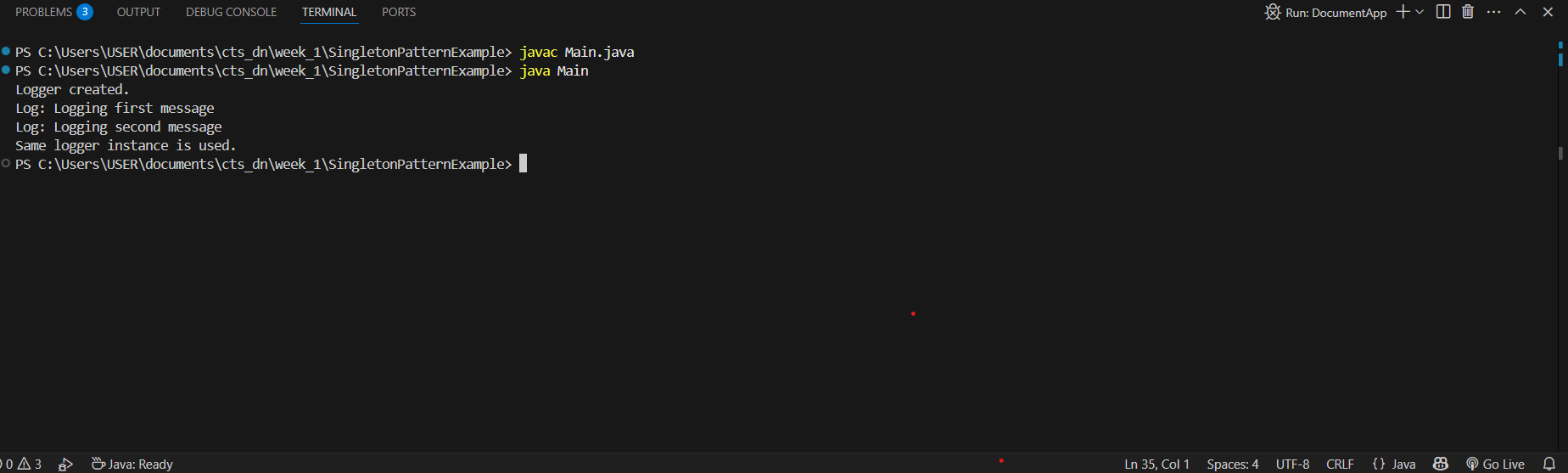
    public void log(String message) {

        System.out.println("Log: " + message);

    }

}

**Output:**

****

**2.** **Implementing the Factory Method Pattern**

**Project : FactoryMethodPatternExample**

**Code:**

**DocumentApp.java**

public class DocumentApp {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document word = wordFactory.createDocument();

        word.open();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdf = pdfFactory.createDocument();

        pdf.open();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excel = excelFactory.createDocument();

        excel.open();

    }

}

interface Document {

    void open();

}

class WordDocument implements Document {

    public void open() {

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

    }

}

class PdfDocument implements Document {

    public void open() {

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

    }

}

class ExcelDocument implements Document {

    public void open() {

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

    }

}

abstract class DocumentFactory {

    public abstract Document createDocument();

}

class WordDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }

}

class PdfDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

class ExcelDocumentFactory extends DocumentFactory {

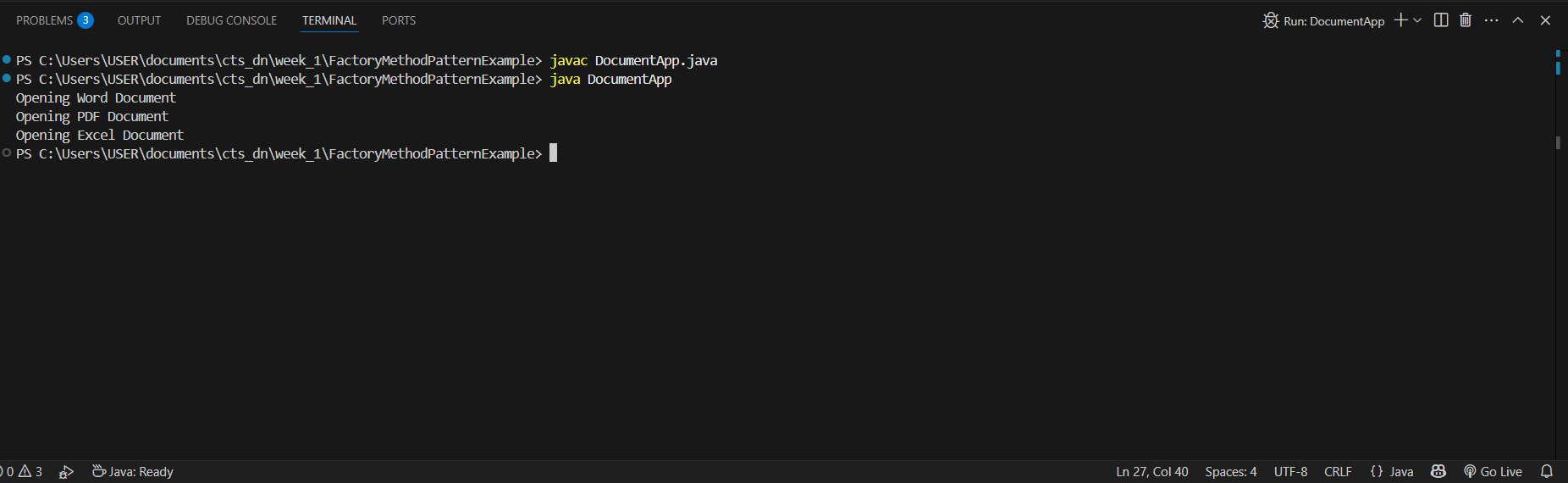
    public Document createDocument() {

        return new ExcelDocument();

    }

}

**Output:**

****