**DESIGN PATTERNS AND PRINCIPLES**

**Exercise 1: Implementing the Singleton Pattern**

**Logger.java**

public class Logger {

    private static Logger singleInstance;

    private Logger() {

        System.out.println("Logger instance has been created.");

    }

    public static Logger getInstance() {

        if (singleInstance == null) {

            singleInstance = new Logger();

        }

        return singleInstance;

    }

    public void writeLog(String logMessage) {

        System.out.println("[SYSTEM LOG]: " + logMessage);

    }

}

**TestSingleton.java**

public class TestSingleton {

    public static void main(String[] args) {

        Logger firstLogger = Logger.getInstance();

        firstLogger.writeLog("Application started successfully");

        Logger secondLogger = Logger.getInstance();

        secondLogger.writeLog("User authentication completed");

        if (firstLogger == secondLogger) {

            System.out.println("SUCCESS: Both references point to the same Logger instance");

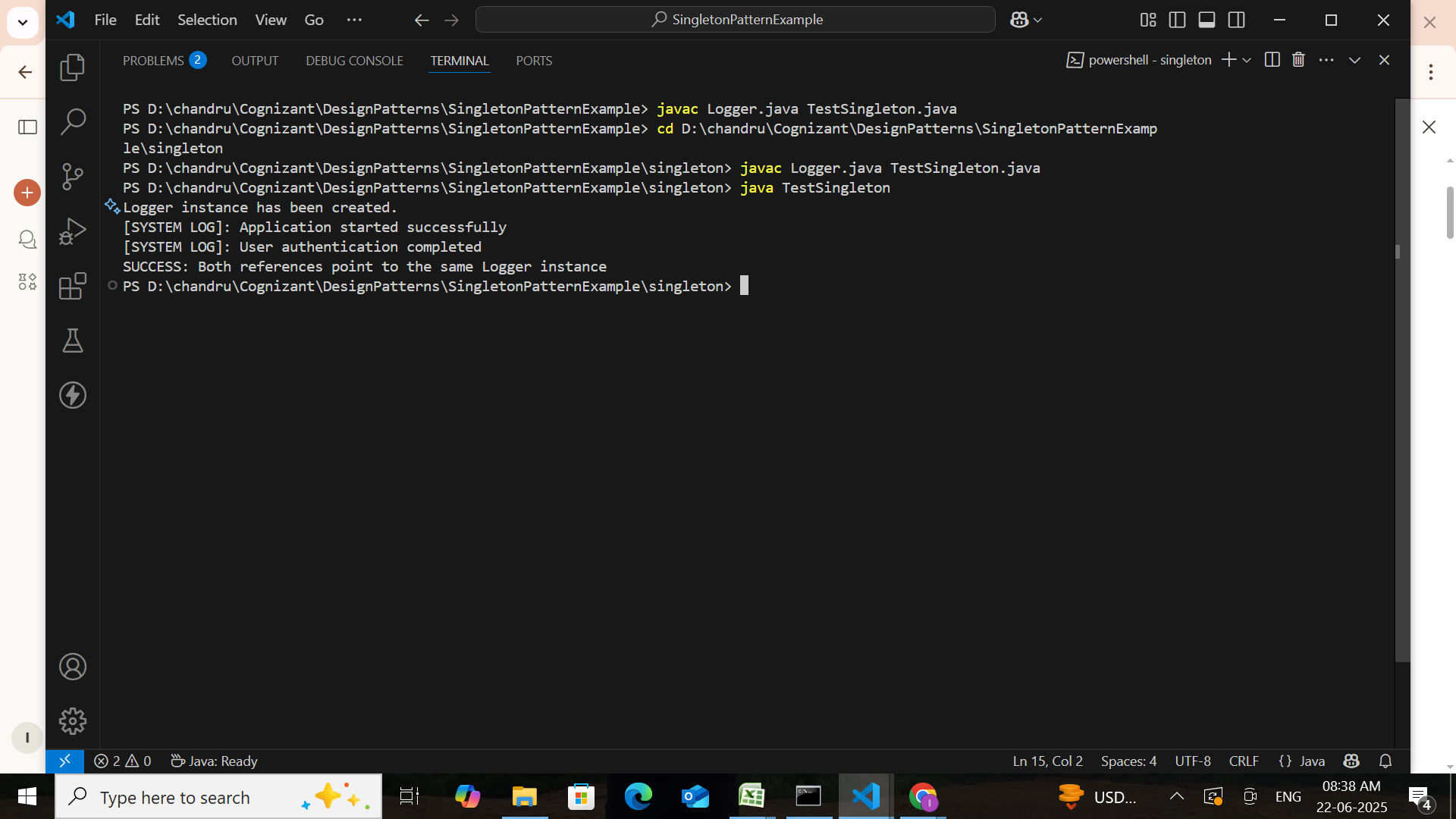
        } else {

            System.out.println("ERROR: Multiple Logger instances detected");

        }

    }

}

**Output:**

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

**Document.java**

public interface Document {

    void open();

}

**ExcelDocument.java**

public class ExcelDocument implements Document {

    public void openDocument() {

        System.out.println("Starting Excel spreadsheet application...");

    }

}

**PdfDocument.java**

public class PdfDocument implements Document {

    public void openDocument() {

        System.out.println("Loading PDF reader with security features...");

    }

}

**WordDocument.java**

public class WordDocument implements Document {

    public void openDocument() {

        System.out.println("Initializing Microsoft Word document editor...");

    }

}

**DocumentFactory.java**

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

**ExcelDocumentFactory.java**

public class ExcelDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new ExcelDocument();

    }

}

**PdfDocumentFactory.java**

public class PdfDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

**WordDocumentFactory.java**

public class WordDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }

}

**DocumentFactoryTest.java**

public class DocumentFactoryTest {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document wordDocument = wordFactory.createDocument();

        wordDocument.openDocument();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdfDocument = pdfFactory.createDocument();

        pdfDocument.openDocument();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excelDocument = excelFactory.createDocument();

        excelDocument.openDocument();

    }

}

**Output:**

