**Exercise 1: Implementing the Singleton Pattern**

**Logger.java**

public class Logger {

    // Private static instance of Logger

    private static Logger instance;

    // Private constructor to prevent instantiation

    private Logger() {

    }

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

    public void log(String message) {

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

    }

}

LoggerTest.java

public class LoggerTest {

    public static void main(String[] args) {

        Logger logger1 = Logger.getInstance();

        Logger logger2 = Logger.getInstance();

        // Test if both references point to the same instance

        if (logger1 == logger2) {

            System.out.println("Logger is a singleton. Both references are the same instance.");

        } else {

            System.out.println("Logger is NOT a singleton. Different instances exist.");

        }

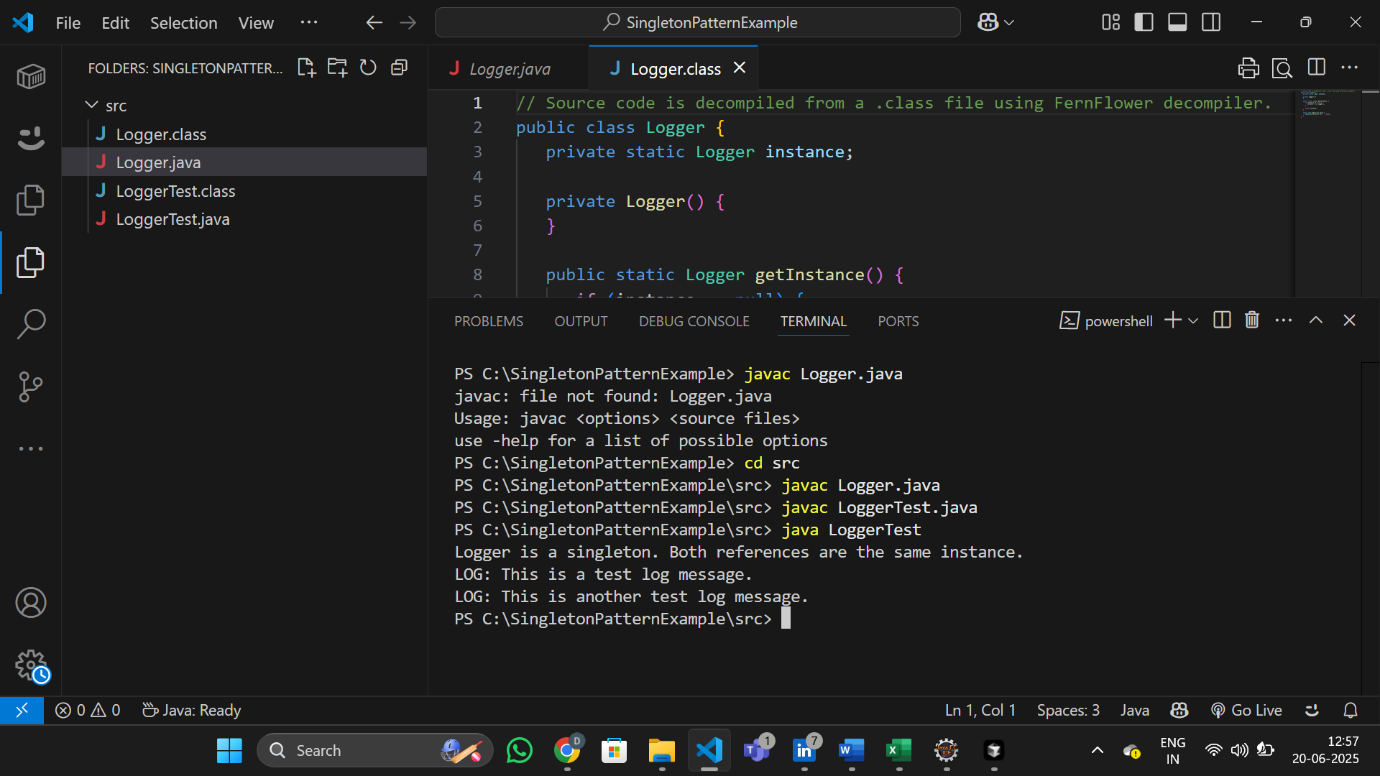
        // Test logging functionality

        logger1.log("This is a test log message.");

        logger2.log("This is another test log message.");

    }

}



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

**Document.java**

public interface Document {

    void open();

    void save();

}

**DocumentFactory.java**

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

**ExcelDocument.java**

public class ExcelDocument implements Document {

    @Override

    public void open() {

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

    }

    @Override

    public void save() {

        System.out.println("Saving Excel document...");

    }

}

**ExcelDocumentFactory.java**

public class ExcelDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new ExcelDocument();

    }

}

**FactoryMethodPatternTest.java**

public class FactoryMethodPatternTest {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document wordDoc = wordFactory.createDocument();

        wordDoc.open();

        wordDoc.save();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdfDoc = pdfFactory.createDocument();

        pdfDoc.open();

        pdfDoc.save();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excelDoc = excelFactory.createDocument();

        excelDoc.open();

        excelDoc.save();

    }

}

**PdfDocument.java**

public class PdfDocument implements Document {

    @Override

    public void open() {

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

    }

    @Override

    public void save() {

        System.out.println("Saving PDF document...");

    }

}

**PdfDocumentFactory.java**

public class PdfDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new PdfDocument();

    }

}

**WordDocument.java**

public class WordDocument implements Document {

    @Override

    public void open() {

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

    }

    @Override

    public void save() {

        System.out.println("Saving Word document...");

    }

}

**WordDocumentFactory.java**

public class WordDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new WordDocument();

    }

}