**DIGITAL\_NURTURE\_4.0**

**Exercise 1 & 2**

**Superset ID - 6402411**

**Exercise 1: Implementing the Singleton Pattern**

**Code**

public class Main {

    public static void main(String[] args) {

        Logger logger1 = Logger.getInstance();

        logger1.log("First message");

        Logger logger2 = Logger.getInstance();

        logger2.log("Second message");

        if (logger1 == logger2) {

            System.out.println("Both logger instances are the same (Singleton confirmed).");

        } else {

            System.out.println("Different logger instances (Singleton failed).");

        }

    }

}

class Logger {

    private static Logger instance;

    private Logger() {

M

    }

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

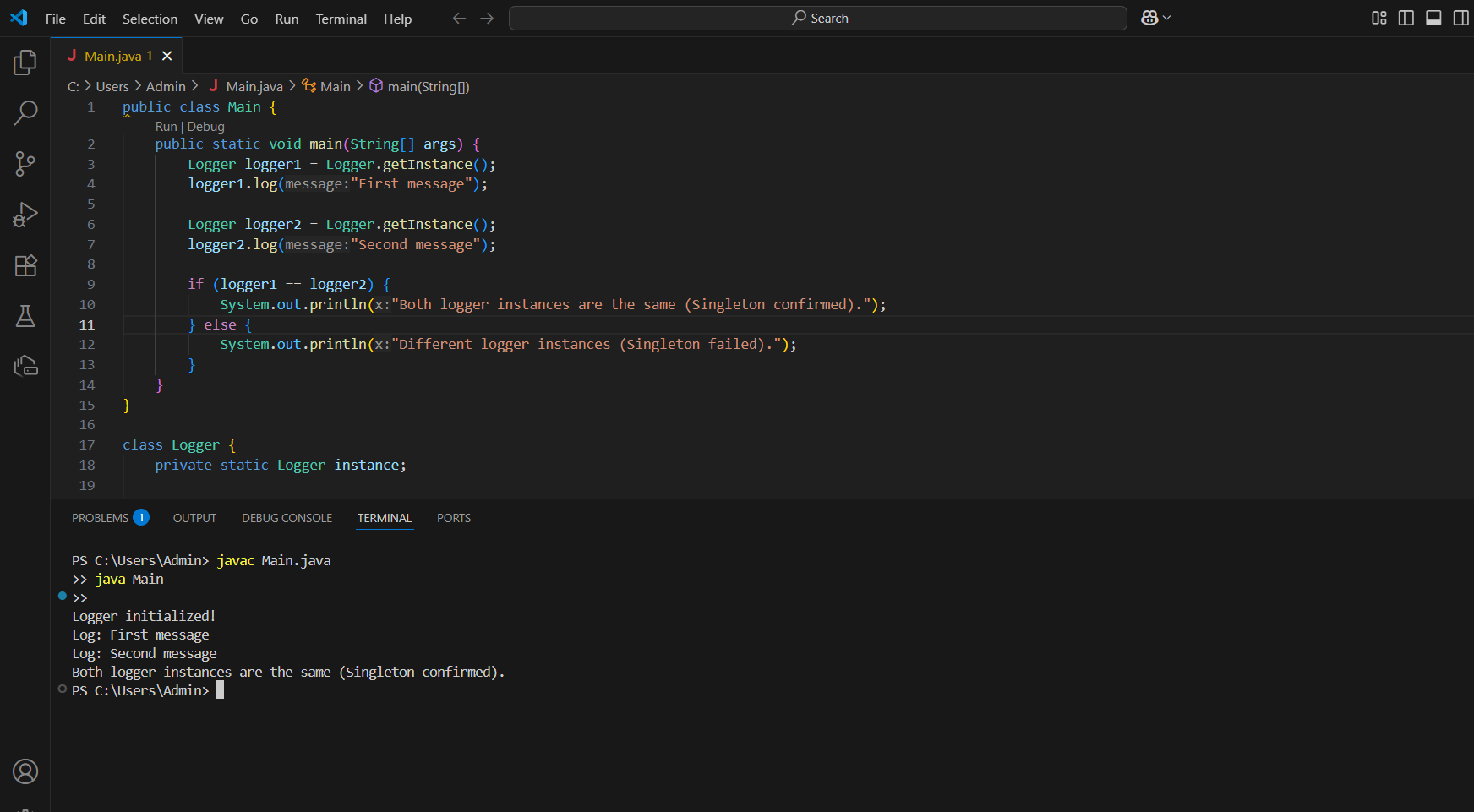
    public void log(String message) {

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

    }

}

**CODE OUTPUT**



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

**CODE**

public class factory {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document wordDoc = wordFactory.createDocument();

        wordDoc.open();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdfDoc = pdfFactory.createDocument();

        pdfDoc.open();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excelDoc = excelFactory.createDocument();

        excelDoc.open();

    }

}

// Interface

interface Document {

    void open();

}

//Concrete Document Classes

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 Factory

abstract class DocumentFactory {

    public abstract Document createDocument();

}

// Concrete Factories

class WordDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }

}

class PdfDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

class ExcelDocumentFactory extends DocumentFactory {

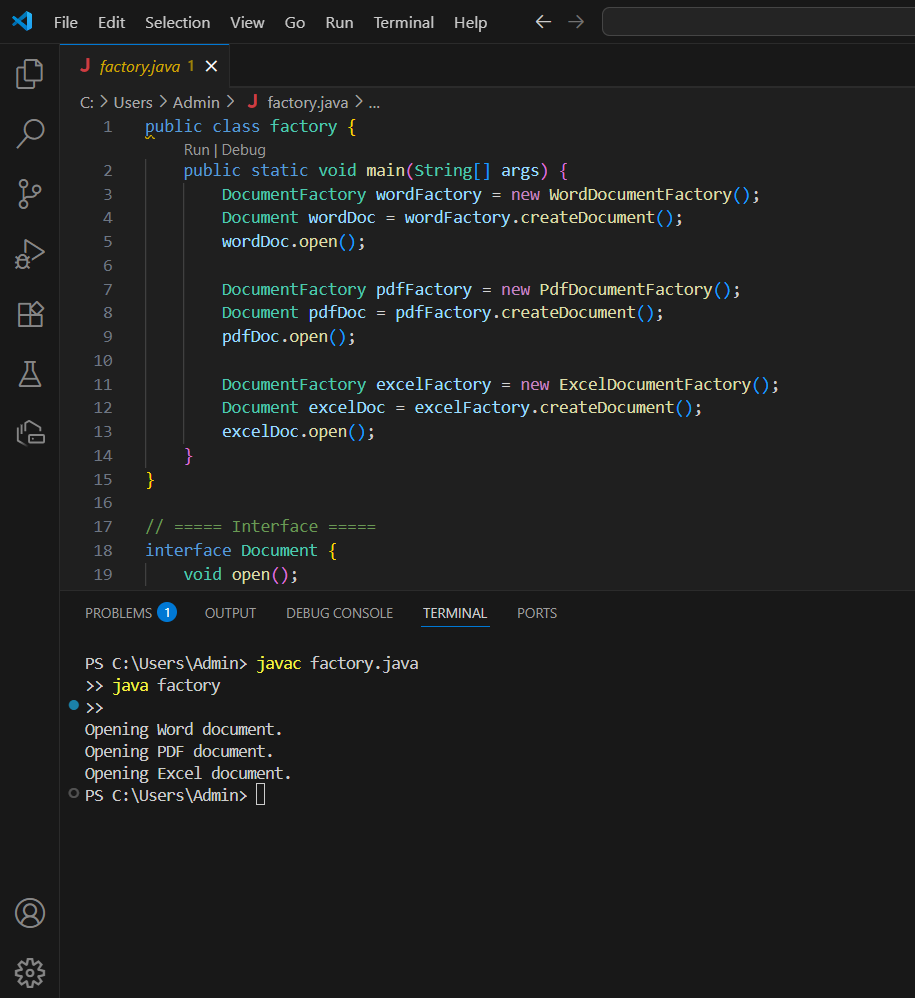
    public Document createDocument() {

        return new ExcelDocument();

    }

}

**OUTPUT**

****