Cognizant Digital Nurture 4.0

Name: Siri Chandana Chittipolu

Email: sirichittipolu11@gmail.com

Superset ID: 6386277

Mandatory Hands-On Exercises

Design principles & Patterns

Exercise 1: Implementing the Singleton Pattern

Solution:

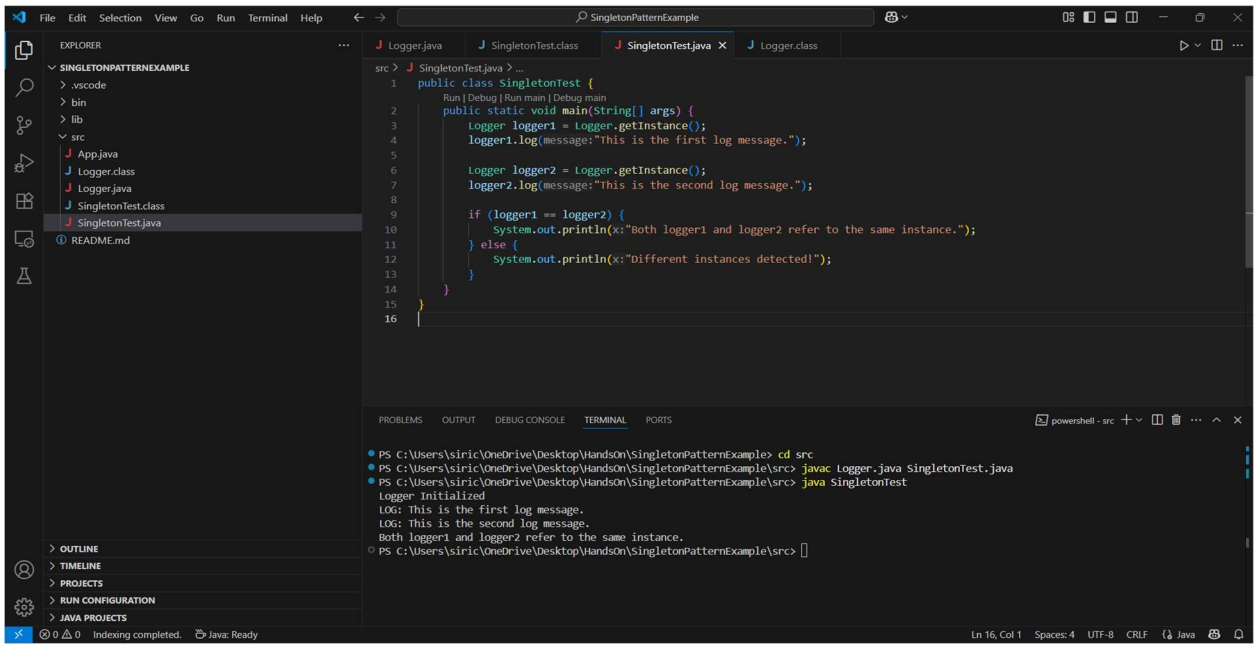
Logger.java:

```
public class Logger {
  private static Logger instance;
   private Logger() {
      System.out.println("Logger Initialized");
   }
   public static Logger getInstance() {
      if (instance == null) {
         instance = new Logger();
      }
      return instance;
   }
   public void log(String message) {
      System.out.println("LOG: " + message);
   }
}
```

SingletonTest.java :

```
public class SingletonTest {
    public static void main(String[] args) {
        Logger logger1 = Logger.getInstance();
        logger1.log("This is the first log message.");
        Logger logger2 = Logger.getInstance();
        logger2.log("This is the second log message.");
        if (logger1 == logger2) {
            System.out.println("Both logger1 and logger2 refer to the same instance.");
        } else {
            System.out.println("Different instances detected!");
        }
    }
}
```

Output:



Exercise 2: Implementing the Factory Method Pattern:

Solution:

Document.java :

```
public interface Document {
   void open();
}
```

WordDocument.java :

```
public class WordDocument implements Document {
   public void open() {
      System.out.println("Opening a Word document.");
   }
}
```

PdfDocument.java :

```
public class PdfDocument implements Document {
   public void open() {
      System.out.println("Opening a PDF document.");
   }
}
```

ExcelDocument.java :

```
public class ExcelDocument implements Document {
   public void open() {
      System.out.println("Opening an Excel document.");
   }
}
```

WordDocumentFactory.java :

```
public class WordDocumentFactory extends DocumentFactory {
   public Document createDocument() {
      return new WordDocument();
   }
}
```

PdfDocumentFactory.java :

```
public class PdfDocumentFactory extends DocumentFactory {
   public Document createDocument() {
      return new PdfDocument();
   }
}
```

ExceldocumentFactory.java :

```
public class ExcelDocumentFactory extends DocumentFactory {
   public Document createDocument() {
      return new ExcelDocument();
   }
}
```

TestFactoryMethod.java :

```
public class TestFactoryMethod {
    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();
    }
}
```

Output: