**COGNIZANT DIGI NURTURE 4.0**

**WEEK 1 MANDATORY HANDS ON BY (6377625)**

**DESIGN PRINCIPLES AND PATTERNS:**

**Exercise 1: Implementing Singleton Pattern**

**Program:**

I created a project named SimpleLogger and a separate Main class to verify that only one instance of Logger is created and used across the application.

**Code:**

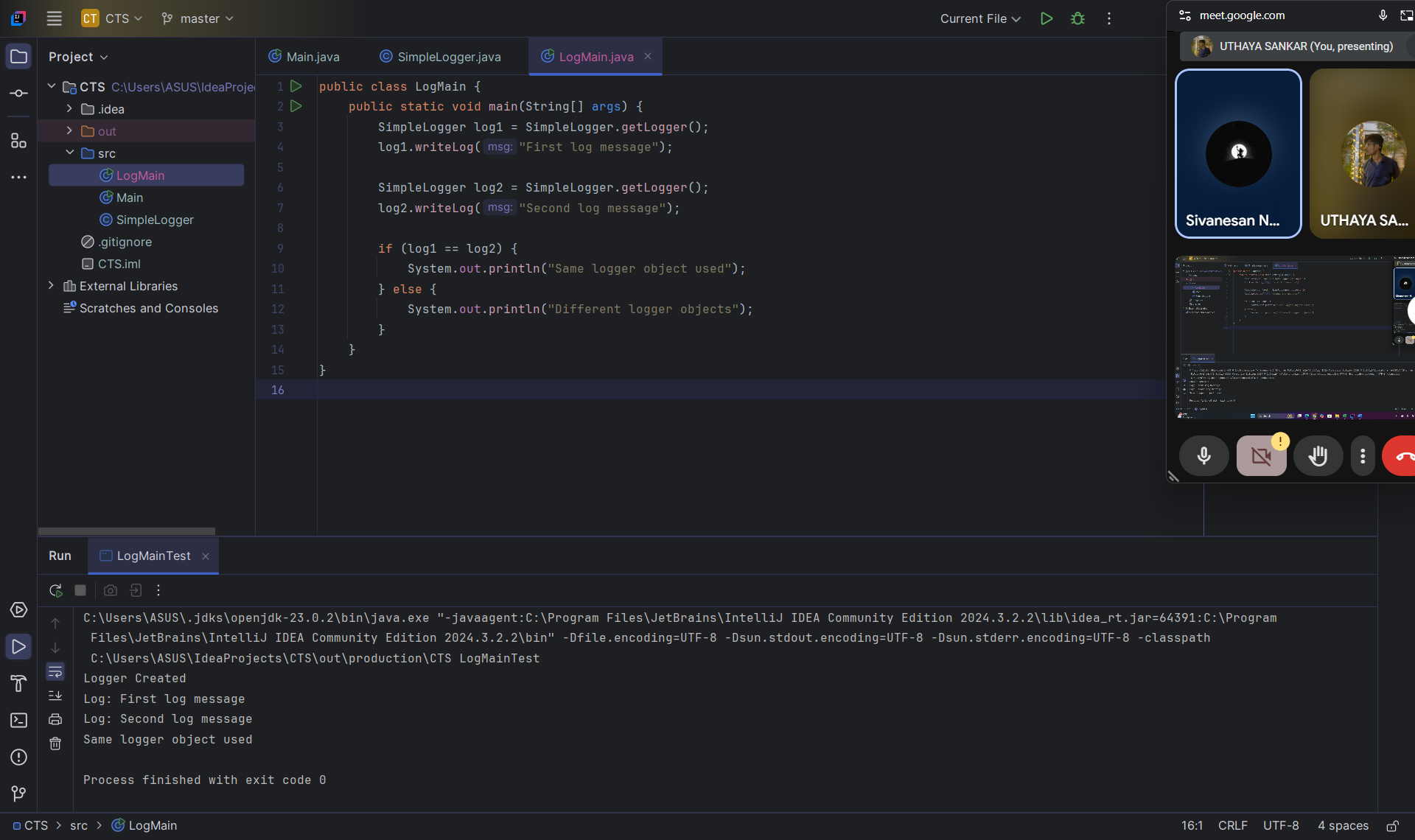
**SimpleLogger.java**

public class SimpleLogger {  
  
 private static SimpleLogger instance;  
  
 private SimpleLogger() {  
 System.out.println("Logger Created");  
 }  
  
 public static SimpleLogger getLogger() {  
 if (instance == null) {  
 instance = new SimpleLogger();  
 }  
 return instance;  
 }  
  
 public void writeLog(String msg) {  
 System.out.println("Log: " + msg);  
 }  
}

LogMain.java

public class LogMain {  
 public static void main(String[] args) {  
 SimpleLogger log1 = SimpleLogger.getLogger();  
 log1.writeLog("First log message");  
  
 SimpleLogger log2 = SimpleLogger.getLogger();  
 log2.writeLog("Second log message");  
  
 if (log1 == log2) {  
 System.out.println("Same logger object used");  
 } else {  
 System.out.println("Different logger objects");  
 }  
 }  
}

**OUTPUT :**

****

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

I developed a Document Management System that needs to support the creation of different types of documents like Word, PDF, and Excel. To create these documents without tightly coupling the code to specific classes, I used the Factory Method Pattern.

CODE:

**Document.java:**

public interface Document {  
 void open();  
}

**WordDocument.java:**

public class WordDocument implements Document {

public void open() {

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

}

}

**PdfDocument.java:**

public class PdfDocument implements Document {

public void open() {

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

}

}

**ExcelDocument.java:**

public class ExcelDocument implements Document {

public void open() {

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

}

}

**DocumentFactory.java:**

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**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();

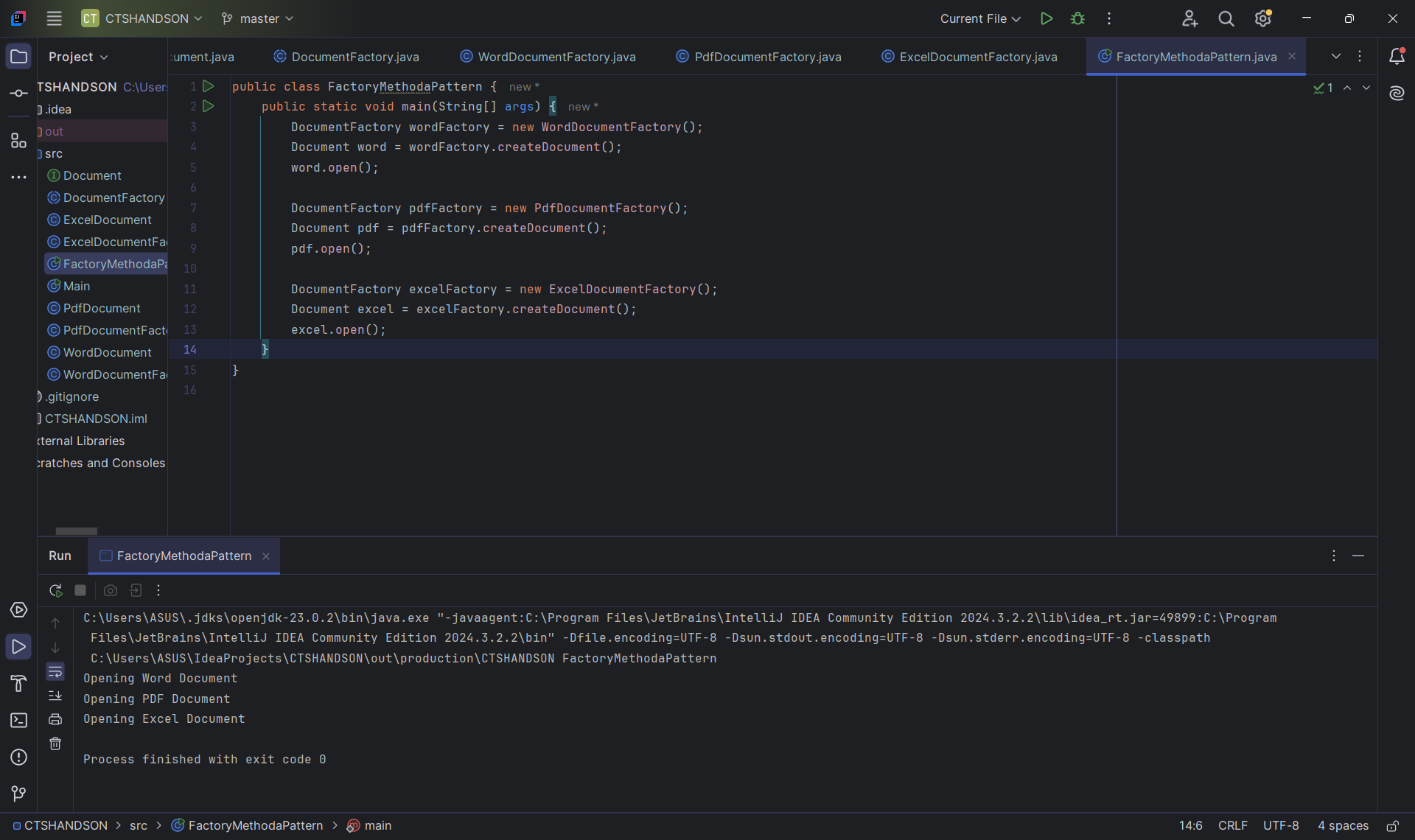
}

}

**FactoryMethodPattern.java:**

public class FactoryMethodPattern {  
 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();  
 }  
}

**OUTPUT :**

****