**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**CODING:**

**Logger.java**

package SingletonPattern;

public class Logger {

private static Logger *instance*;

String name="Ahamed saliha";

int age=20;

private Logger(){}

public static Logger getInstance()

{

if(*instance*==null)

{

*instance*=new Logger();

}

return *instance*;

}

public String getName()

{

return name;

}

}

**Test.java**

**package** SingletonPattern;

**public** **class** Test {

**public** **static** **void** main(String args[])

{

Logger obj2=Logger.*getInstance*();

String user="Ahamed saliha";

String user2="Ahamed saliha";

String name=obj2.getName();

**if**(user.equals(name))

{

System.***out***.println("We have the authorized user");

}

**if**(user2.equals(name))

{

System.***out***.println("We have the authorized user");

}

**else**

{

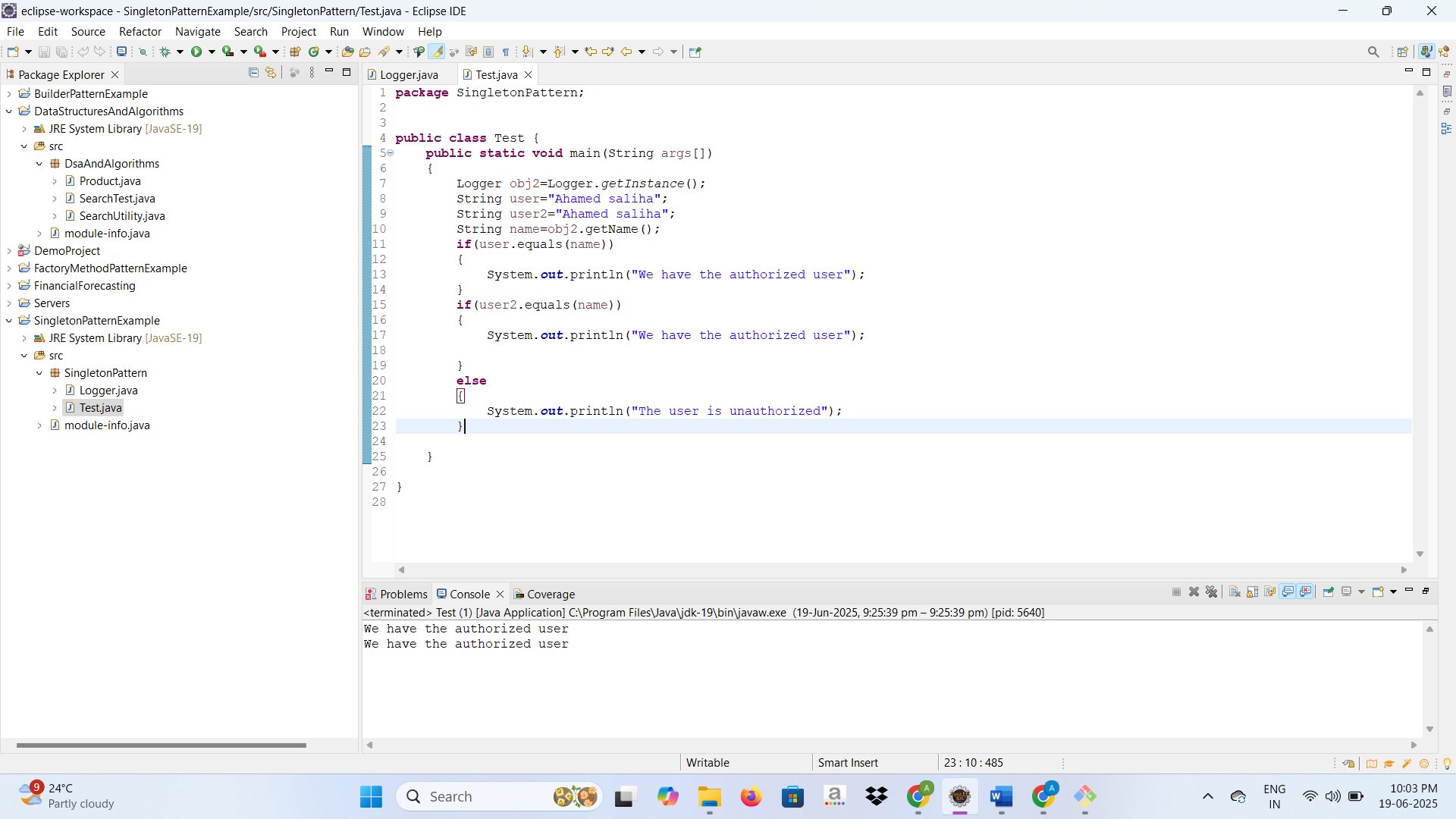
System.***out***.println("The user is unauthorized");

}

}

}

OUTPUT



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

**Scenario:**

You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

**CODING:**

**Document.java**

package FactoryMethod;

public interface Document {

void open();

}

**ExcelDocument.java**

package FactoryMethod;

public class ExcelDocument implements Document{

public void open()

{

System.*out*.println("Opening excel document");

}

}

**PdfDocument.java**

package FactoryMethod;

public class PdfDocument implements Document{

public void open()

{

System.*out*.println("Opening pdf document");

}}

**WordDocument.java**

package FactoryMethod;

public class WordDocument implements Document{

public void open()

{

System.*out*.println("Opening word document");

}

}

**DocumentFactory.java**

package FactoryMethod;

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**ExcelDocumentFactory.java**

package FactoryMethod;

public class ExcelDocumentFactory extends DocumentFactory{

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**PdfDocumentFactory.java**

package FactoryMethod;

public class PdfDocumentFactory extends DocumentFactory{

@Override

public Document createDocument() {

return new PdfDocument();

}

}

**WordDocumentFactory.java**

package FactoryMethod;

public class WordDocumentFactory extends DocumentFactory{

public Document createDocument() {

return new WordDocument();

}

}

**FactoryTest.java**

package FactoryMethod;

public class FactoryTest {

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**

