**DESIGN PATTERNS AND PRINCIPLES**

Exercise 1: Implementing the singleton pattern

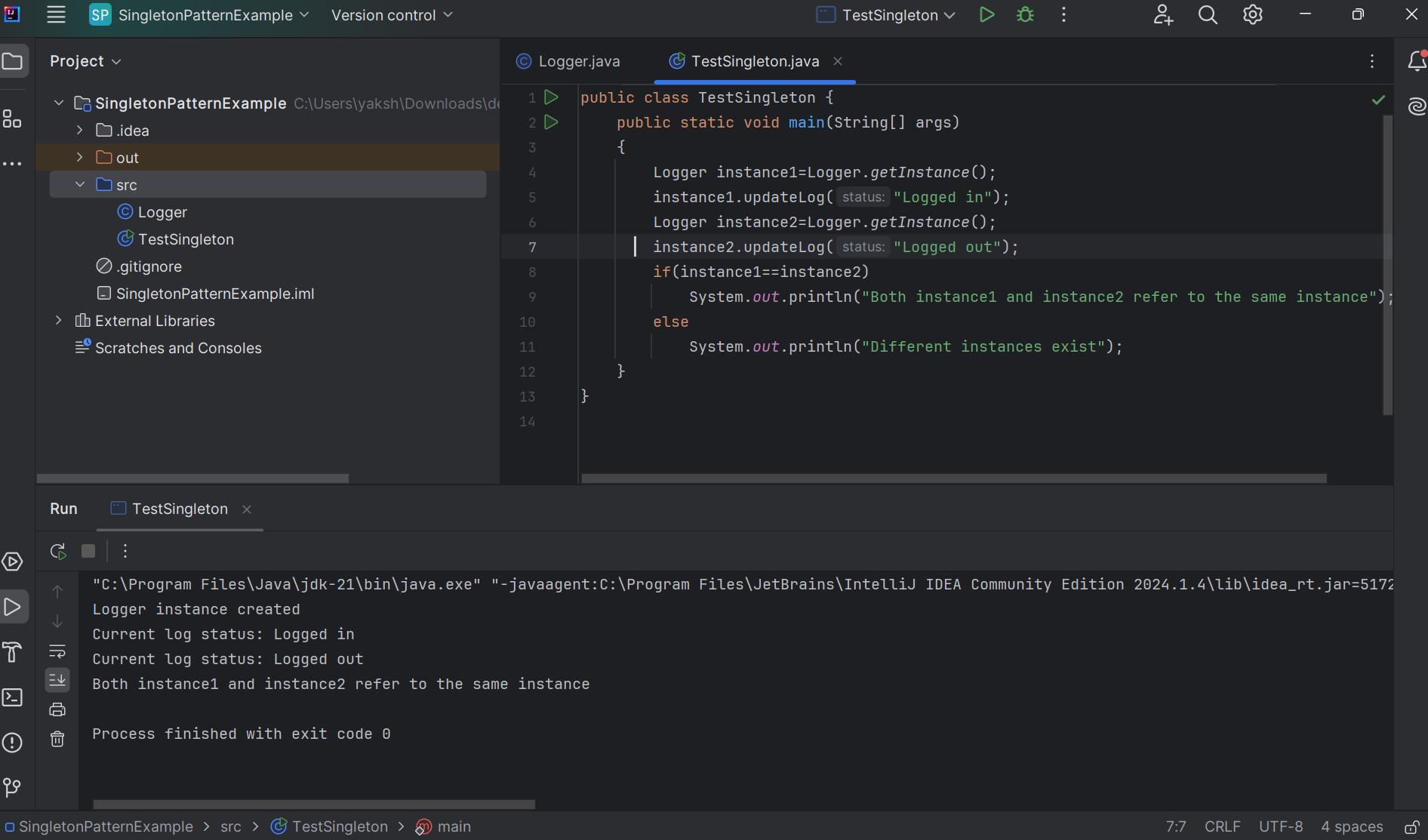
**Code:**

**File : *Logger.class***  
public class Logger {  
 private static Logger *logger*;  
 private Logger()  
 {  
 System.*out*.println("Logger instance created");  
 }  
 public static Logger getInstance()  
 {  
 if(*logger*==null)  
 {  
 *logger*=new Logger();  
 }  
 return *logger*;  
 }  
 public void updateLog(String status)  
 {  
 System.*out*.println("Current log status: "+ status);  
 }  
}

**File:** ***TestSimpleton.class***

public class TestSingleton {  
 public static void main(String[] args)  
 {  
 Logger instance1=Logger.*getInstance*();  
 instance1.updateLog("Logged in");  
 Logger instance2=Logger.*getInstance*();  
 instance2.updateLog("Logged out");  
 if(instance1==instance2)  
 System.*out*.println("Both instance1 and instance2 refer to the same instance");  
 else  
 System.*out*.println("Different instances exist");  
 }  
}

**Output Screenshot :**

****

Thus , a logging utility class has been implemented using the simpleton pattern.