COGNIZANT

Digital Nurture 4.0

Deep Skilling - Java FSE

WEEK-2 HANDS ON

By Kaviya P

**LOGGING USING SLF4J**

**Exercise 1: Logging Error Messages and Warning Levels Task: Write a Java application that demonstrates logging error messages and warning levels using SLF4J.**

**Pom.xml**

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<scope>test</scope>

</dependency>

<!-- Optionally: parameterized tests support -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-params</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.30</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

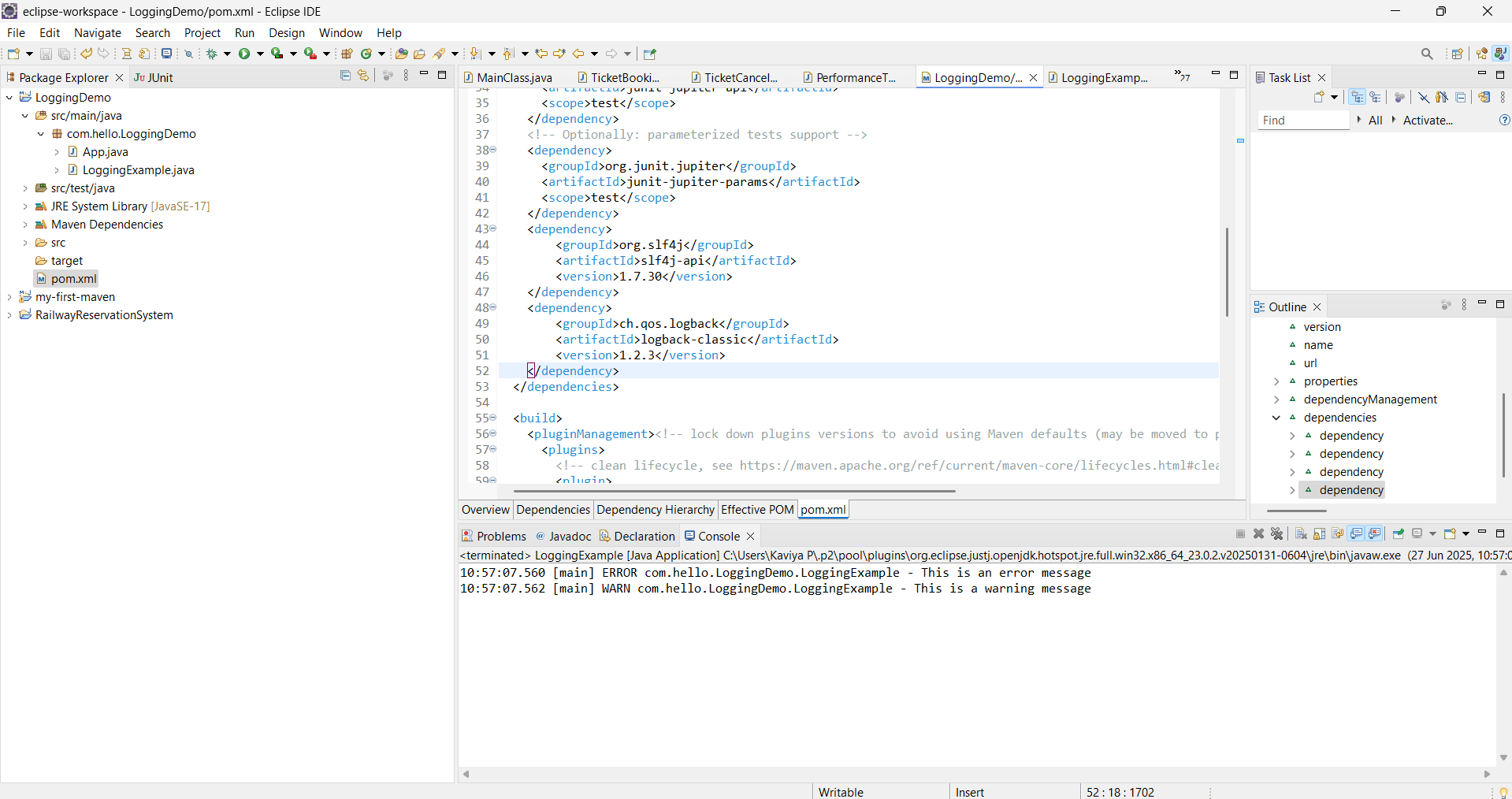
<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

</dependencies>

**Output of dependencies added in pom.xml**

****

**LoggingExample.java**

**package** com.hello.LoggingDemo;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** LoggingExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(LoggingExample.**class**);

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

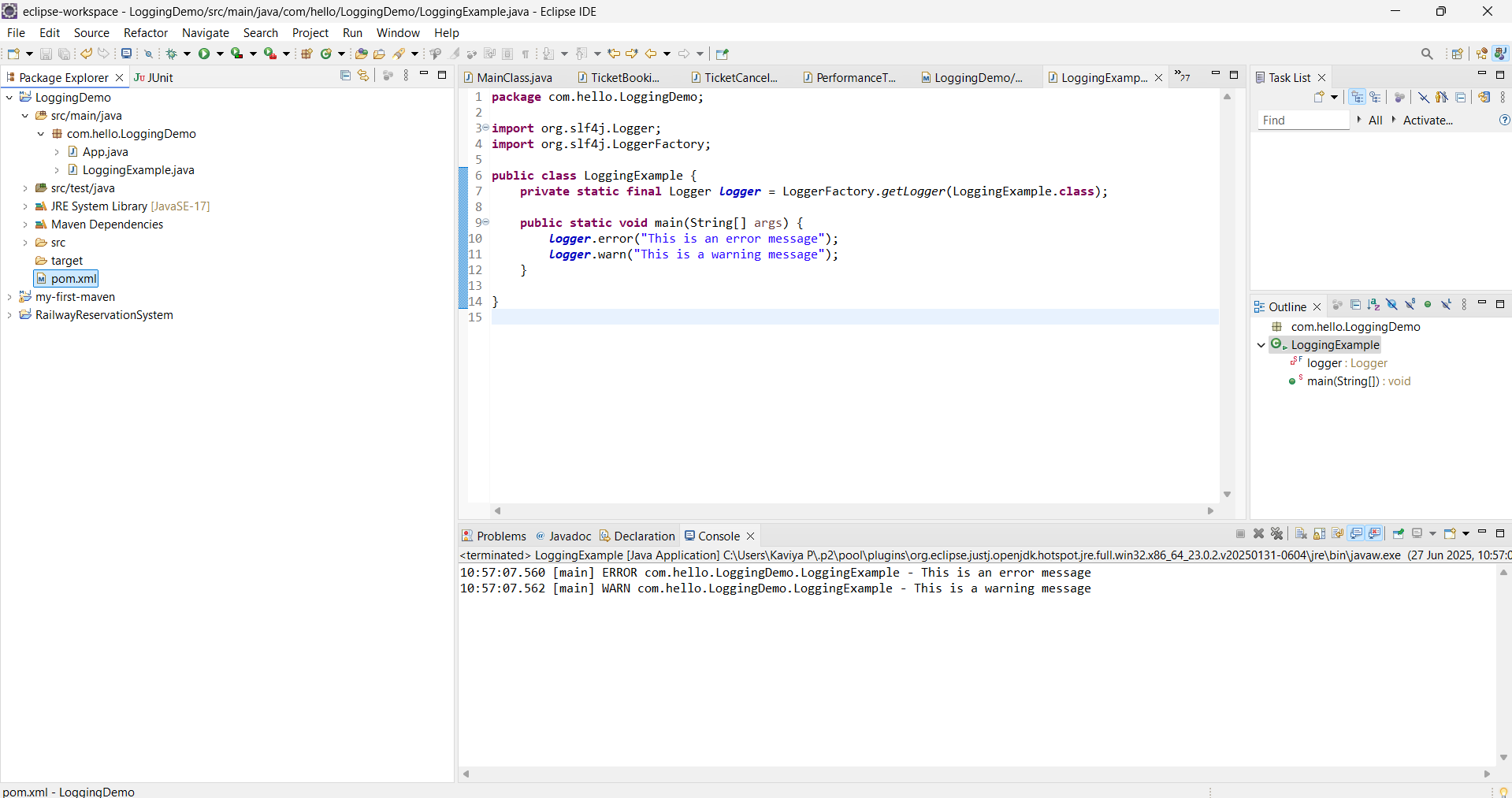
***logger***.error("This is an error message");

***logger***.warn("This is a warning message");

}

}

**OUTPUT**

****

**Exercise 2: Parameterized Logging**

**Task: Write a Java application that demonstrates parameterized logging using SLF4J.**

**ParameterizedLoggingExample.java**

**package** com.hello.LoggingDemo;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** ParameterizedLoggingExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(ParameterizedLoggingExample.**class**);

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

String user = "Kaviya P";

**int** attempts = 5;

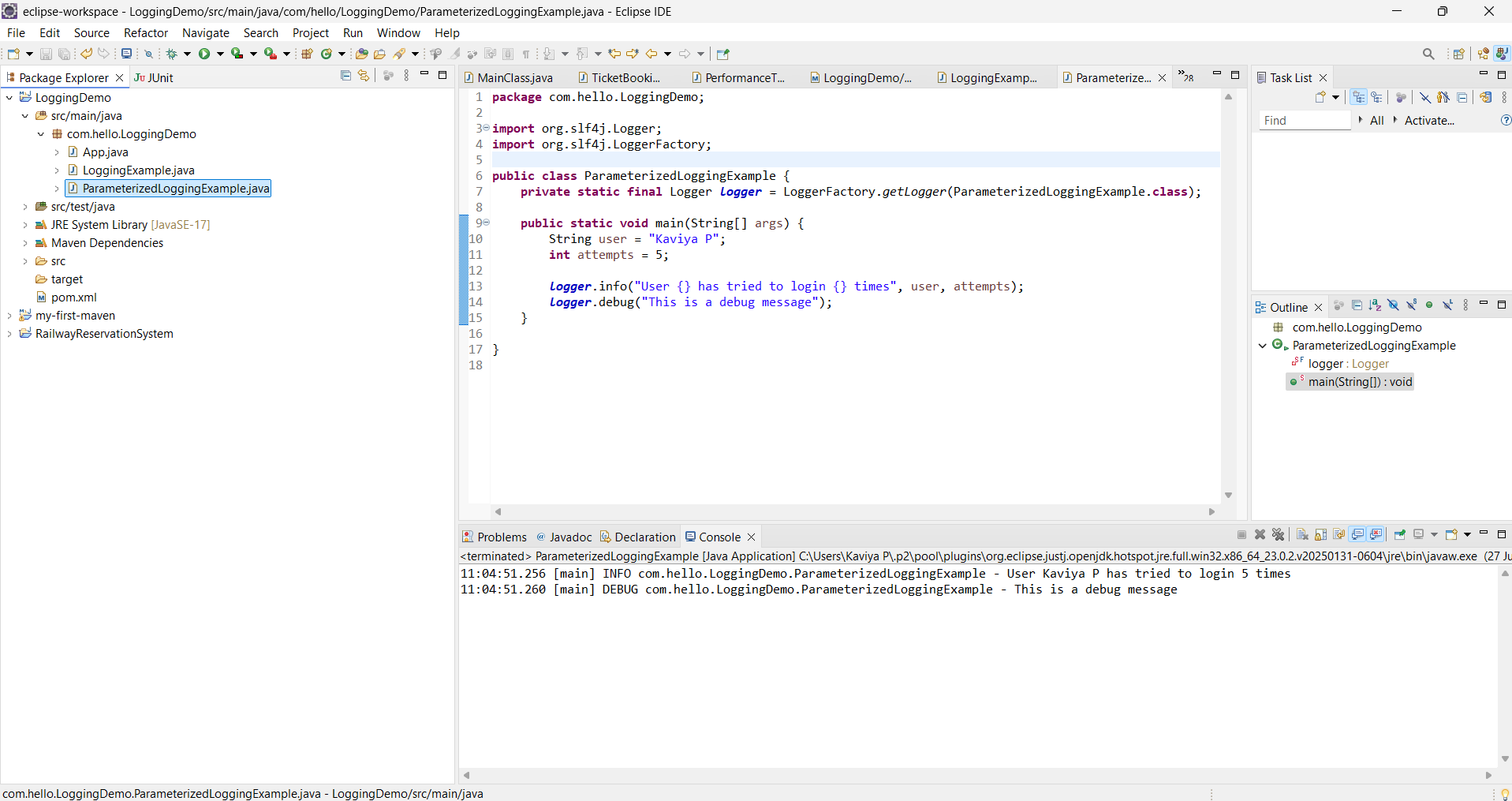
***logger***.info("User {} has tried to login {} times", user, attempts);

***logger***.debug("This is a debug message");

}

}

**OUTPUT**

****

**Exercise 3: Using Different Appenders**

**Task: Write a Java application that demonstrates using different appenders with SLF4J.**

**logback.xml**

<configuration>

    <appender name="console" class="ch.qos.logback.core.ConsoleAppender">

        <encoder>

            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>

        </encoder>

    </appender>

    <appender name="file" class="ch.qos.logback.core.FileAppender">

        <file>app.log</file>

        <append>true</append>

        <encoder>

            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>

        </encoder>

    </appender>

    <root level="debug">

        <appender-ref ref="console"/>

        <appender-ref ref="file"/>

    </root>

</configuration>

**AppenderExample.java**

**package** com.hello.LoggingDemo;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** AppenderExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(AppenderExample.**class**);

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

***logger***.info("Logging to both console and file");

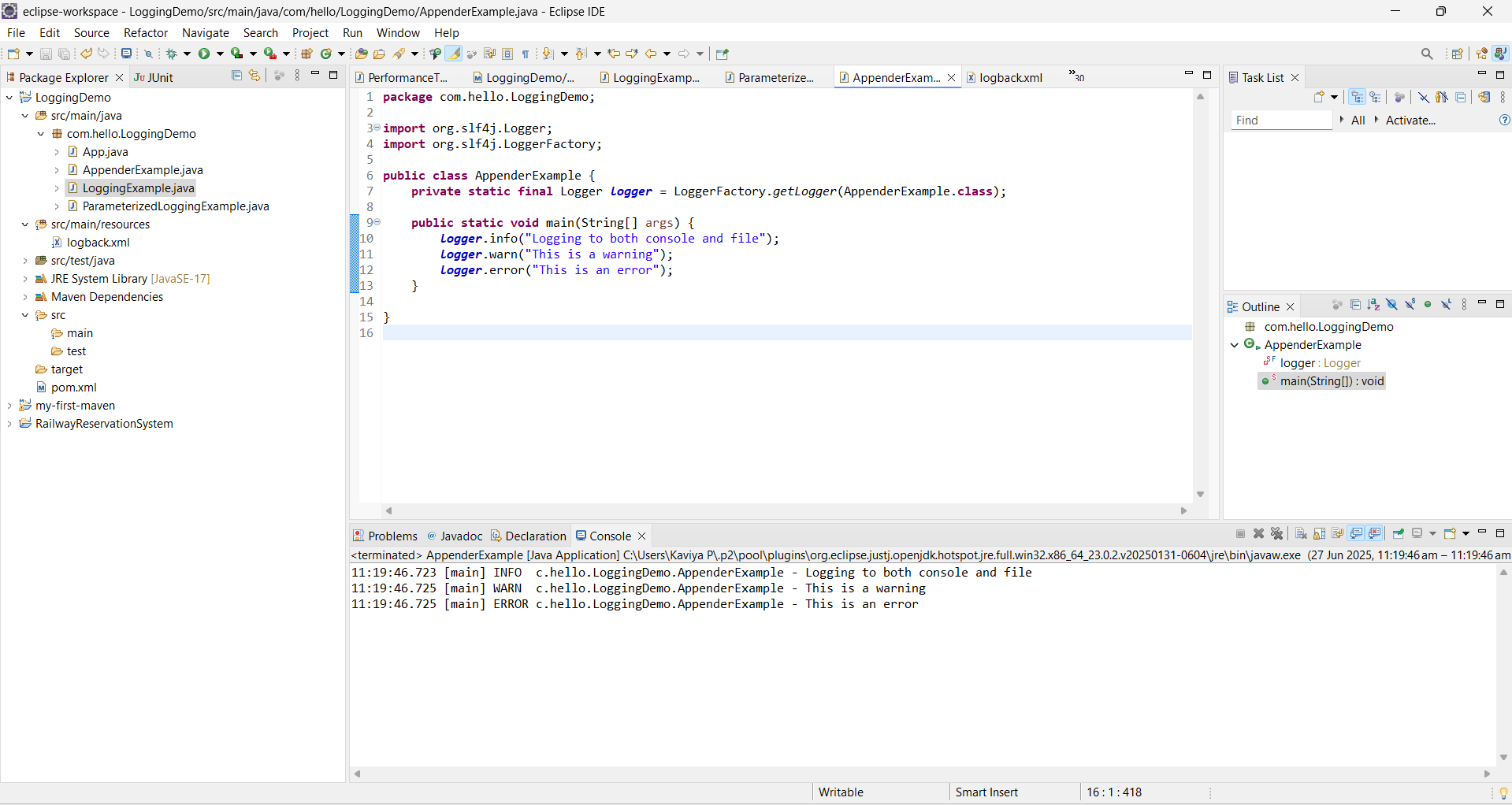
***logger***.warn("This is a warning");

***logger***.error("This is an error");

}

}

**OUTPUT**

****