**Exercise – 1 Logging Error Messages and Warning Levels**

**SLF4J stands for:**

Simple Logging Facade for Java

It is a common interface for various Java logging frameworks like:

- Log4j

- Logback

- java.util.logging (JUL)

- tinylog etc.

**Example**

import org.apache.log4j.Logger;

Logger logger = Logger.getLogger(MyClass.class);

Here, you are tightly coupled to Log4j. If someday you want to switch to Logback or any other, you have to rewrite all your code.

Instead

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

Logger logger = LoggerFactory.getLogger(MyClass.class);

logger.info("Hello!");

Here, SLF4J acts as a middle layer, then at runtime, you plug in whichever logging backend you want. Hence, we can say that SLF4J is like a “Universal adapter”.

**LoggingExample.java**

package com.example;

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");

logger.info("This is an info message");

logger.debug("This is a debug message (may not be shown by default)");

}

}

**LogBack.xml**

<configuration>

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

<file>logs/app.log</file>

<append>true</append>

<encoder>

<pattern>%d{yyyy-MM-dd HH:mm:ss} %-5level %logger{36} - %msg%n</pattern>

</encoder>

</appender>

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

<encoder>

<pattern>%d{HH:mm:ss} %-5level %logger - %msg%n</pattern>

</encoder>

</appender>

<root level="debug">

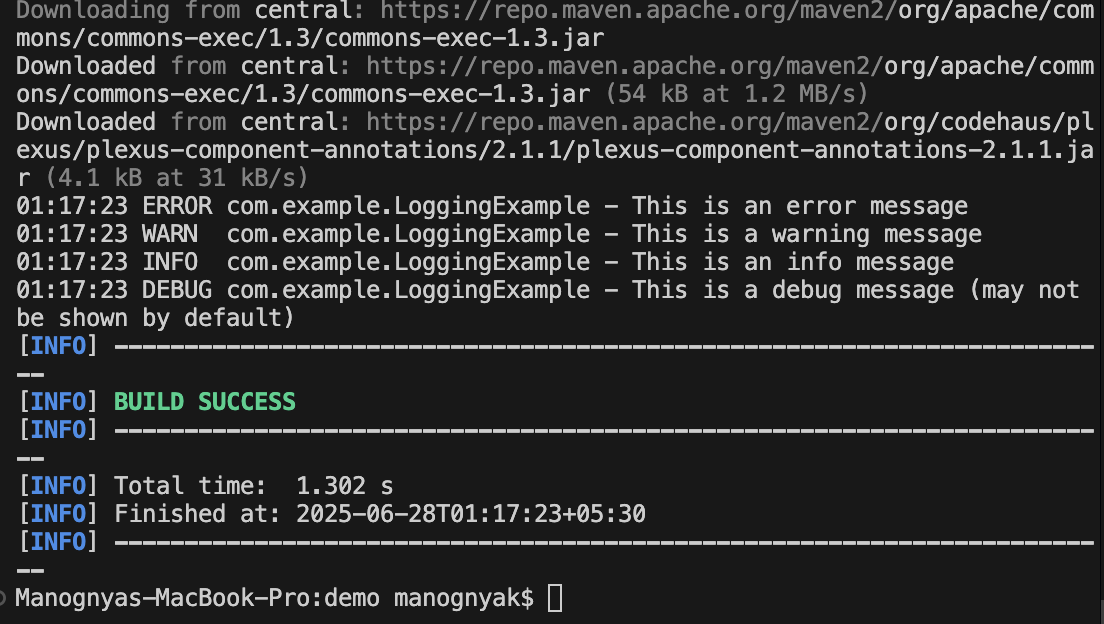
<appender-ref ref="STDOUT" />

<appender-ref ref="FILE" />

</root>

</configuration>

**Output**

****