**WEEK – 2 SLF4J LOGGING FRAMEWORK**

**EXERCISE 1:** Logging Error Messages and Warning Levels

A Java application that demonstrates logging error messages and warning levels using SLF4J.

**DEPENDENCY:**

<dependencies> <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>

These dependencies do two things:

* The **SLF4J API** allows you to write generic logging code without binding it to a specific implementation.
* The **Logback Classic** dependency is the actual logging engine that takes care of formatting and writing the log messages.

**LOGGINGEXAMPLE.JAVA**

package org.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"); }}

It uses SLF4J to get a logger instance tied to the current class (LoggingExample).

It logs two types of messages:

* An **error message**, which is the highest level, used for serious issues.
* A **warning message**, which indicates a potential problem or abnormal situation.

**Output:**

12:00:01.123 [main] ERROR org.example.LoggingExample - This is an error message

12:00:01.126 [main] WARN org.example.LoggingExample - This is a warning message

**SUMMARY:**

This code is a basic but solid example of structured logging in Java using SLF4J + Logback. It introduces a reliable way to report errors and warnings in the application, which is essential for any serious development or production deployment. This pattern is considered best practice in modern Java development.