**Name: Amogh Girish Nagarkar**

**Superset ID: 6403503**

**DN-4.0 - Java FSE - Deep Skilling**

**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.

**Steps:**

1. **Create a New Java Project:** o Create a new Java project named SingletonPatternExample.
2. **Define a Singleton Class:** 
   * Create a class named Logger that has a private static instance of itself.
   * Ensure the constructor of Logger is private.
   * Provide a public static method to get the instance of the Logger class.
3. **Implement the Singleton Pattern:** 
   * Write code to ensure that the Logger class follows the Singleton design pattern.
4. **Test the Singleton Implementation:**

Create a test class to verify that only one instance of Logger is created and used across the application.

Code: Logger.java package singleton;

public class Logger {

private static Logger *instance*;

private Logger() {

System.*out*.println("Logger Initialized");

}

public static Logger getInstance() {

if (*instance* == null) { *instance* = new Logger();

}

return *instance*;

}

public void log(String message) {

System.*out*.println("Log: " + message);

**}**

**}**

**Main.java** package singleton;

public class Main { public static void main(String[] args) {

Logger logger1 = Logger.*getInstance*();

logger1.log("Application Starting...");

Logger logger2 = Logger.*getInstance*(); logger2.log("Another log message...");

if (logger1 == logger2) {

System.*out*.println("Both logger1 and logger2 are the same instance.");

} else {

System.*out*.println("Error! Different instances");

}

}

}

**Output:**

