My Solution

TO ensure that only **one instance** of the Logger class is ever created and used throughout the entire application. This is done to maintain consistency — especially for shared resources like logging, configuration, or database connections

The Logger class contains

| **Code Component** | **Purpose** |
| --- | --- |
| private static Logger instance | Holds the one and only instance of the Logger class |
| private Logger() | Private constructor prevents outside classes from creating new objects |
| public static Logger getInstance() | Provides a global access point; creates instance if it doesn't exist |
| log(String message) | Example method to simulate logging functionality |

.

The LoggerTest class is used to test the logger class, in this class

* We request the logger twice (logger1 and logger2) using Logger.getInstance().
* Since Logger is a singleton:
  + Both variables point to the **same object** in memory.
  + logger1 == logger2 returns true.
* This confirms that only **one logger** instance is used throughout the application.