# **Submitted By :** 6364888

**Design Patterns and Principles.**

# **Exercise 1: Implementing the Singleton Pattern**

# **C# Code :**

Logger.cs

public class Logger

{

    private static Logger? instance;

    private Logger()

    {

        Console.WriteLine("Logger instance created");

    }

    public static Logger GetInstance()

    {

        if (instance == null)

        {

            instance = new Logger();

        }

        return instance;

    }

    public void Log(string message)

    {

        Console.WriteLine("Log: " + message);

    }

}

Program.cs

class Program

{

    static void Main(string[] args)

    {

        Logger logger1 = Logger.GetInstance();

        Logger logger2 = Logger.GetInstance();

        logger1.Log("First log message.");

        logger2.Log("Second log message.");

if (logger1 == logger2)

        {

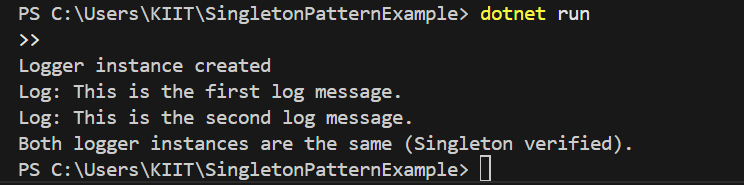
            Console.WriteLine("Singleton verified: Both logger instances are the same.");

        }

}

}

## Output Screenshot :



**Exercise 2: Implementing the Factory Method Pattern**

**C# Code :**

i. Document Types;

IDocument.cs

public interface IDocument

{

    void Open();

}

DocumentTypes.cs

public class WordDocument : IDocument

{

public void Open()

    {

        Console.WriteLine("Opening a Word document.");

    }

}

public class PdfDocument : IDocument

{

    public void Open()

    {

        Console.WriteLine("Opening a PDF document.");

    }

}

public class ExcelDocument : IDocument

{

public void Open()

    {

        Console.WriteLine("Opening an Excel document.");

    }

}

## ii. Factory Implementation;

DocumentFactory.cs

public abstract class DocumentFactory

{

    public abstract IDocument CreateDocument();

}

FactoryTypes.cs

public class WordFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new WordDocument();

    }

}

public class PdfFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new PdfDocument();

    }

}

public class ExcelFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new ExcelDocument();

    }

}

## **iii. Program.cs;**

class Program

{

    static void Main(string[] args)

    {

        DocumentFactory wordFactory = new WordFactory();

        IDocument wordDoc = wordFactory.CreateDocument();

        wordDoc.Open();

        DocumentFactory pdfFactory = new PdfFactory();

        IDocument pdfDoc = pdfFactory.CreateDocument();

        pdfDoc.Open();

DocumentFactory excelFactory = new ExcelFactory();

        IDocument excelDoc = excelFactory.CreateDocument();

        excelDoc.Open();

    }

}

## **Screenshot 2025-06-21 174259iv. Output Screenshot:**