**Project: Implementing the Singleton Pattern**

**✅Code Implementation:**

using system;

namespace SingletonPattern

{

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 DisplayMsg(String msg)

{

Console.WriteLine("Msg:"+msg);

}

}

public class Test

{

static void main(String args[])

{

Logger obj1=Logger.GetInstance();

Logger obj2=Logger.GetInstance();

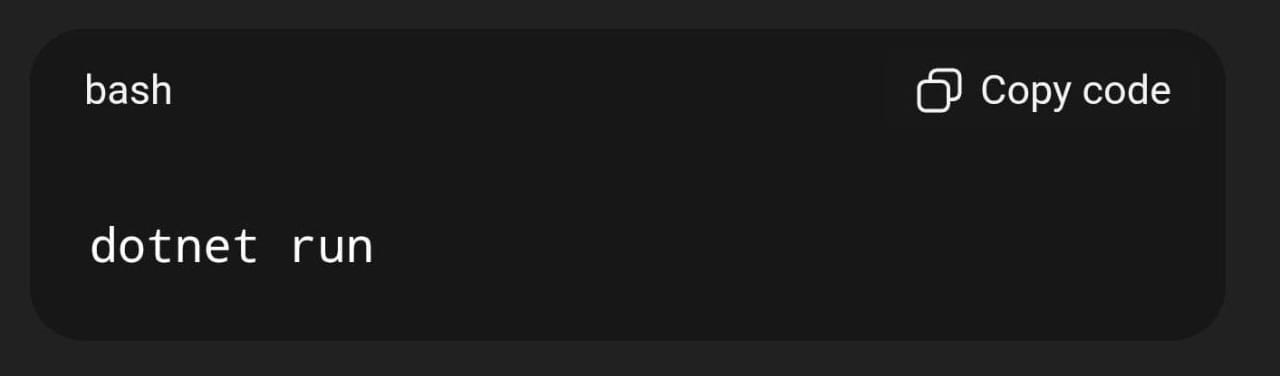
obj1.DisplayMsg("this is first message");

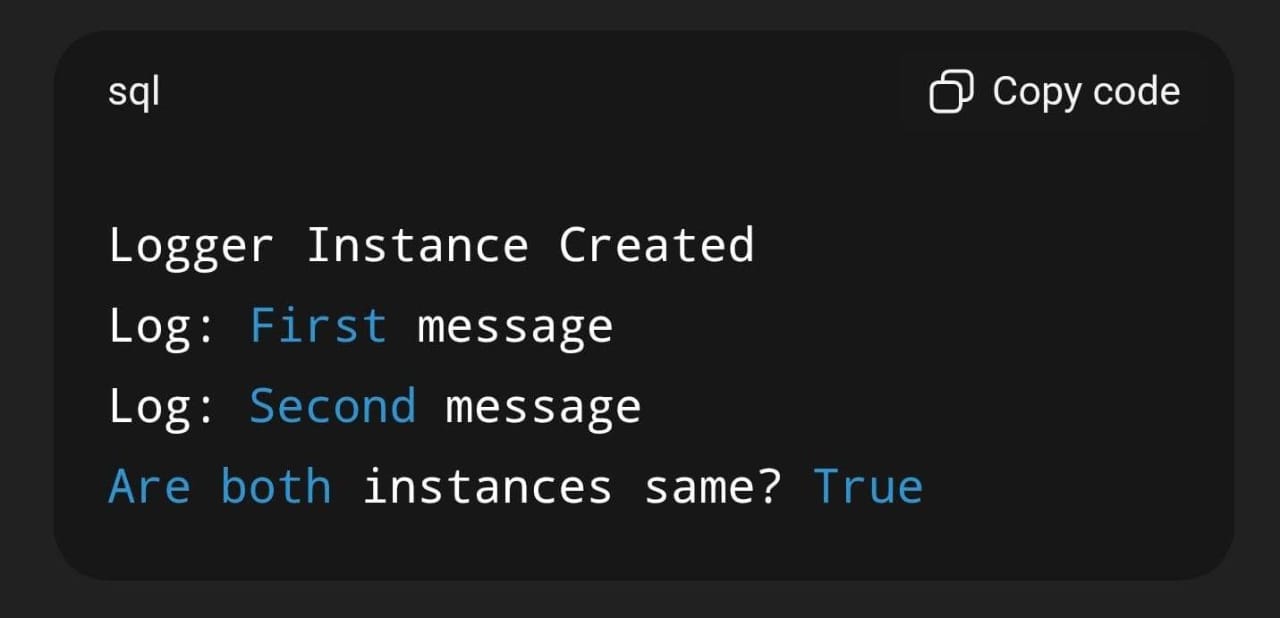
obj1.DisplayMsg("this is second message");

}

}

}





**Project: Implementing the Factory Method Pattern**

**✅Code Implementation:**

using System;

public interface IProduct

{

void Display();

}

public class ProductA : IProduct

{

public void Display() => Console.WriteLine("Product A created.");

}

public class ProductB : IProduct

{

public void Display() => Console.WriteLine("Product B created.");

}

public abstract class Creator

{

public abstract IProduct FactoryMethod();

}

public class ConcreteCreatorA : Creator

{

public override IProduct FactoryMethod() => new ProductA();

}

public class ConcreteCreatorB : Creator

{

public override IProduct FactoryMethod() => new ProductB();

}

class Program

{

static void Main(string[] args)

{

Creator creator = new ConcreteCreatorA();

IProduct product = creator.FactoryMethod();

product.Display();

creator = new ConcreteCreatorB();

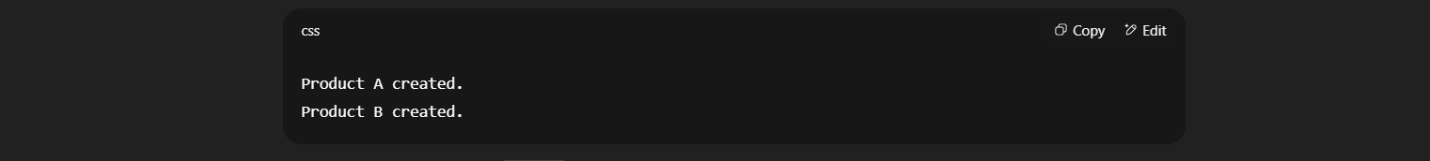
product = creator.FactoryMethod();

product.Display();

}

}

**✅Output:**



**Project: E-Commerce Product Search Function**

**✅Code Implementation**:

using System;

using System.Collections.Generic;

using System.Linq;

class Product

{

public int Id;

public string Name;

public double Price;

public Product(int id, string name, double price)

{

Id = id;

Name = name;

Price = price;

}

public void Display()

{

Console.WriteLine($"ID: {Id} | Name: {Name} | Price: ₹{Price}");

}

}

class ECommerceSearch

{

static void Main()

{

// Sample Product List

List<Product> productList = new List<Product>()

{

new Product(101, "Apple iPhone 15", 79999),

new Product(102, "Samsung Galaxy S24", 72999),

new Product(103, "OnePlus Nord CE", 24999),

new Product(104, "Apple AirPods Pro", 24999),

new Product(105, "Samsung Smart Watch", 19999),

new Product(106, "Sony Bluetooth Speaker", 7999)

};

Console.WriteLine("🔍 Enter keyword to search products:");

string keyword = Console.ReadLine().ToLower();

var results = productList

.Where(p => p.Name.ToLower().Contains(keyword))

.ToList();

Console.WriteLine("\n📦 Search Results:");

if (results.Count == 0)

{

Console.WriteLine("No matching products found.");

}

else

{

foreach (var product in results)

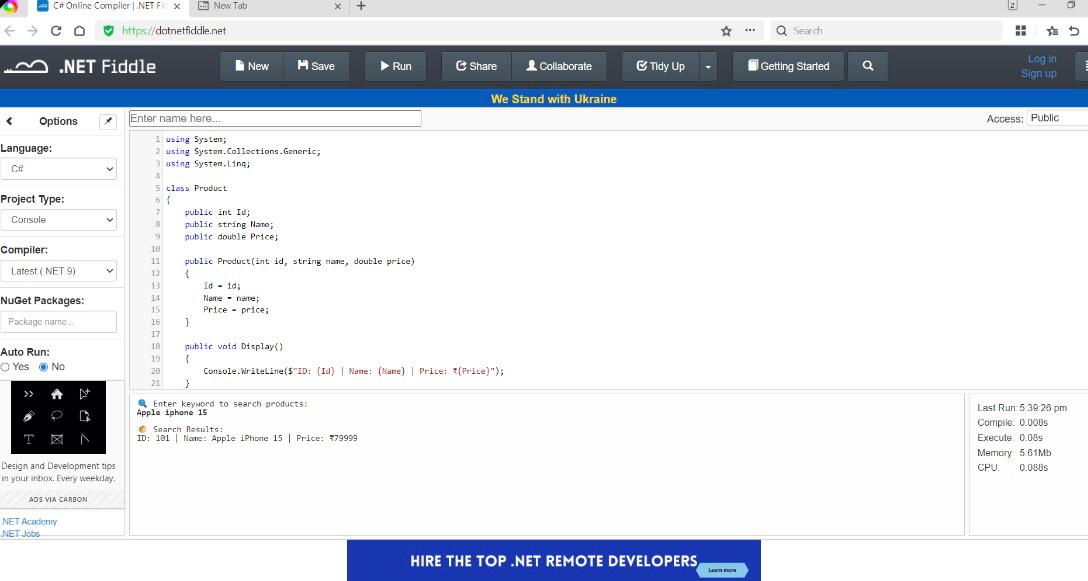
product.Display();

}

}

}

**✅Output:**



**Project: Financial Forecasting**

**✅Code Implementation:**

using System;

using System.Collections.Generic;

using System.Linq;

class Program

{

static void PrintBar(string label, double value)

{

int barLength = (int)(value);

Console.Write(label.PadRight(12) + " ₹" + value.ToString("0.00") + " | ");

for (int i = 0; i < barLength; i++)

Console.Write("█");

Console.WriteLine();

}

static void Main()

{

List<double> revenues = new List<double> { 10.5, 12.2, 11.8, 13.0, 14.5, 15.1, 14.0, 16.2, 17.8, 18.5, 19.0, 20.1 };

List<string> months = new List<string>();

for (int i = 1; i <= 12; i++) months.Add("Month " + i);

Console.WriteLine("📈 Historical Revenue Data (in Lakhs)\n");

for (int i = 0; i < revenues.Count; i++)

{

PrintBar(months[i], revenues[i]);

}

Console.WriteLine("\n📊 Forecasting next 3 months using 3-month Moving Average...\n");

for (int i = 0; i < 3; i++)

{

double avg = revenues.Skip(revenues.Count - 3).Take(3).Average();

revenues.Add(avg);

months.Add("Forecast " + (i + 1));

}

for (int i = 12; i < revenues.Count; i++)

{

PrintBar(months[i], revenues[i]);

}

}

}

**✅Output:**

