

JURNAL MODUL 13



Disusun Oleh :

Nama : Ganes Gemi Putra

Kelas : SE-07-02

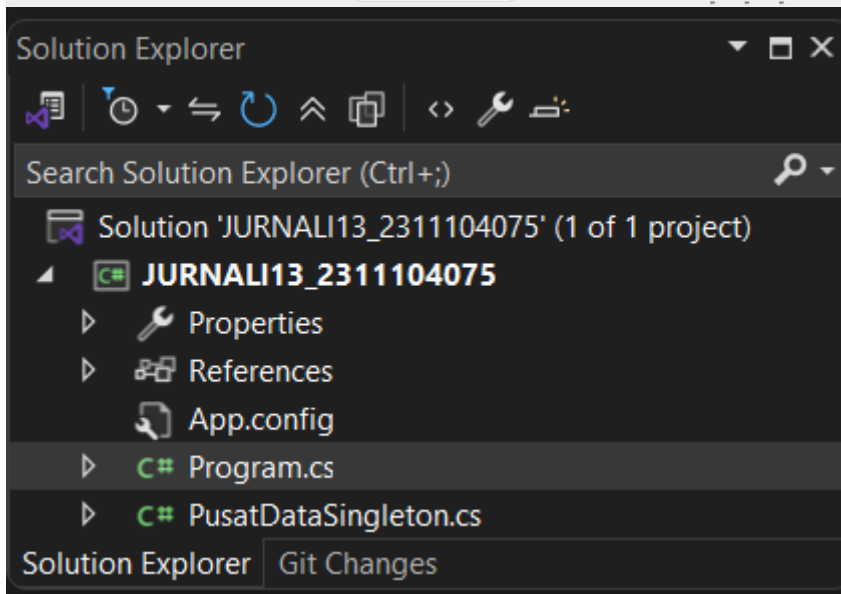
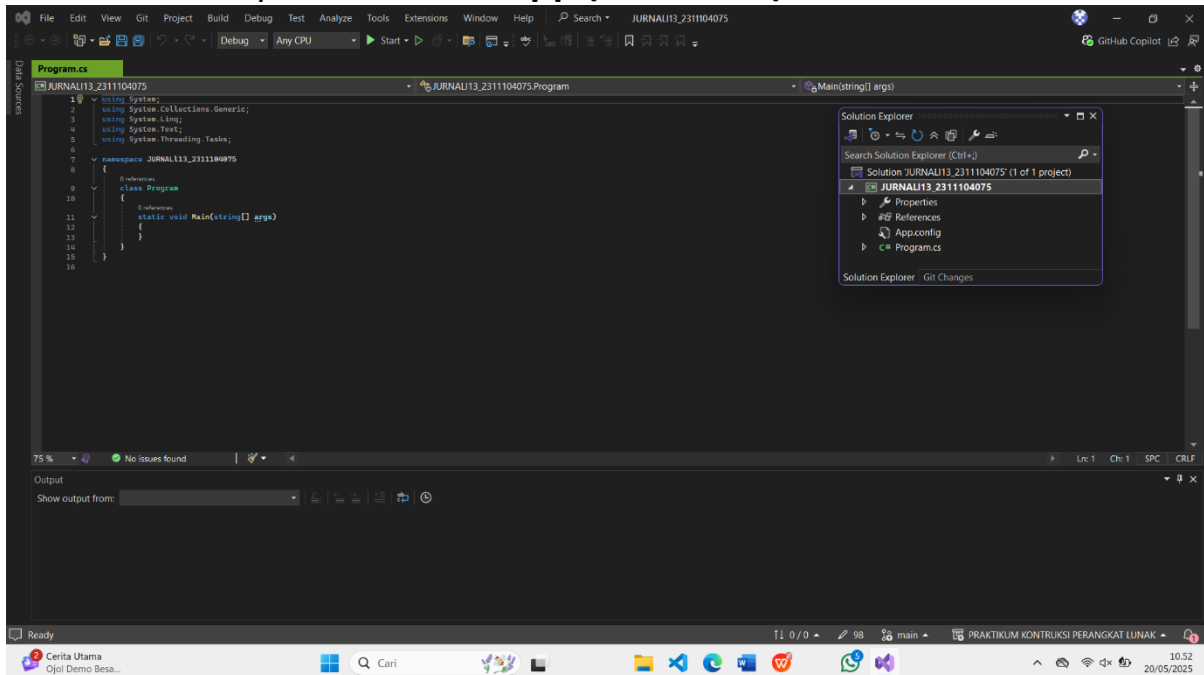
NIM : (2311104075)

Dosen : YUDHA ISLAMI SULISTYA

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2025**

❖ MEMBUAT PROJECT GUI BARU

1. Buka Visual Studio.
2. Buat project baru dengan nama JURNALI13_2311104075
3. Pilih template **Console App (.NET Core)**.



❖ B. MENJELASKAN DESIGN PATTERN SINGLETON

A. Dua Contoh Penggunaan Singleton:

1. Manajemen Koneksi Database

Singleton digunakan untuk memastikan hanya ada satu instance koneksi ke database selama aplikasi berjalan. Ini mencegah pembukaan koneksi

berulang yang dapat membebani sumber daya.

2. **Logger Aplikasi**

Singleton memungkinkan satu instance logger untuk mencatat semua aktivitas atau error di seluruh aplikasi, sehingga log tetap konsisten dan terpusat.

B. Langkah Implementasi Singleton:

1. **Buat Konstruktor Private**

Mencegah instansiasi langsung dari luar kelas.

2. **Deklarasi Variabel Static**

Simpan instance tunggal dalam variabel static di dalam kelas.

3. **Buat Method Public Static untuk Akses Instance**

Method ini (misal: `GetInstance()`) memeriksa apakah instance sudah ada. Jika belum, instance baru dibuat.

4. **Thread-Safety (Opsional)**

Tambahkan sinkronisasi jika aplikasi multithread untuk mencegah race condition.

C. Kelebihan dan Kekurangan Singleton:

• **Kelebihan:**

1. Memastikan hanya satu instance yang ada.
2. Akses global ke instance tersebut.
3. Efisiensi sumber daya karena tidak perlu membuat objek berulang.

• **Kekurangan:**

1. Sulit di-test karena ketergantungan pada state global.
2. Melanggar prinsip *Single Responsibility* (kelas mengatur instansiasi dan logika bisnis).
3. Potensi *memory leak* jika instance tidak diatur dengan baik.

✚ **Sumber:** [Refactoring.Guru – Singleton Pattern](#)

➤ **C. IMPLEMENTASI CLASS PusatDataSingleton (C#)**

```
using System;
using System.Collections.Generic;

public sealed class PusatDataSingleton
{
```

```
private static PusatDataSingleton _instance;
public List<string> DataTersimpan { get; private set; }

// Konstruktor private
private PusatDataSingleton()
{
    DataTersimpan = new List<string>();
}

// Method untuk mendapatkan instance Singleton
public static PusatDataSingleton GetDataSingleton()
{
    if (_instance == null)
    {
        _instance = new PusatDataSingleton();
    }
    return _instance;
}

// Method tambah data
public void AddSebuahData(string input)
{
    DataTersimpan.Add(input);
}

// Method hapus data berdasarkan index
public void HapusSebuahData(int index)
{
    if (index >= 0 && index < DataTersimpan.Count)
    {
        DataTersimpan.RemoveAt(index);
    }
}

// Method cetak semua data
public void PrintSemuaData()
{
    foreach (var data in DataTersimpan)
    {
        Console.WriteLine(data);
    }
}

// Method get jumlah data
public int GetSemuaData()
{
    return DataTersimpan.Count;
}
}
```

➤ D. IMPLEMENTASI PROGRAM UTAMA

```
using System;

class Program
{
```

```
static void Main(string[] args)
{
    // Langkah D.A dan D.B: Inisialisasi data1 dan data2 dengan instance Singleton
    PusatDataSingleton data1 = PusatDataSingleton.GetDataSingleton();
    PusatDataSingleton data2 = PusatDataSingleton.GetDataSingleton();

    // Langkah D.C: Tambah data ke data1
    data1.AddSebuahData("Ganes Gemi Putra (Team 8)");
    data1.AddSebuahData("AHMAD AL-FARIZI (Team 8)");
    data1.AddSebuahData("KAFKA PUTRA RIYADI (Team 8)");
    data1.AddSebuahData("NAURA AISHA ZAHIRA (Team 8)");
    data1.AddSebuahData("FAISHAL ARIF SETIAWAN (Team 8)");
    data1.AddSebuahData("Asisten Praktikum: Gideon Tonarawa Ladiyo"); // Index 5

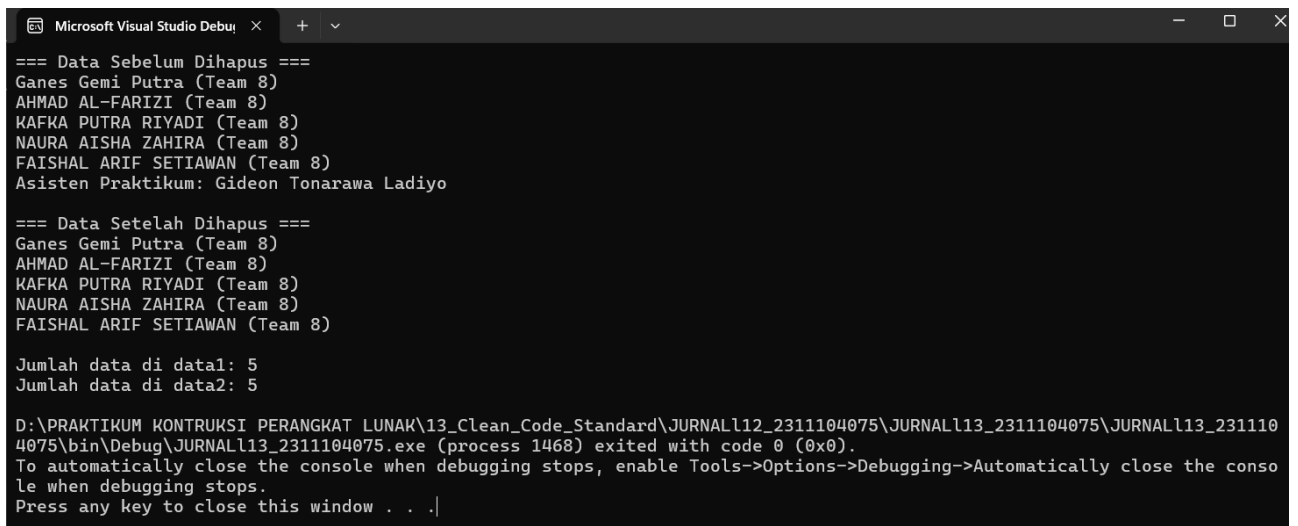
    // Langkah D.D: Cetak data dari data2
    Console.WriteLine("=== Data Sebelum Dihapus ===");
    data2.PrintSemuaData();

    // Langkah D.E: Hapus data asisten (index 5)
    data2.HapusSebuahData(5); // Perbaiki indeks

    // Langkah D.F: Cetak data dari data1
    Console.WriteLine("\n=== Data Setelah Dihapus ===");
    data1.PrintSemuaData();

    // Langkah D.G: Cetak jumlah data di data1 dan data2
    Console.WriteLine($"Jumlah data di data1: {data1.GetSemuaData()}");
    Console.WriteLine($"Jumlah data di data2: {data2.GetSemuaData()}");
}
}
```

OUTPUT :



```
Microsoft Visual Studio Debug Console
=== Data Sebelum Dihapus ===
Ganes Gemi Putra (Team 8)
AHMAD AL-FARIZI (Team 8)
KAFKA PUTRA RIYADI (Team 8)
NAURA AISHA ZAHIRA (Team 8)
FAISHAL ARIF SETIAWAN (Team 8)
Asisten Praktikum: Gideon Tonarawa Ladiyo

=== Data Setelah Dihapus ===
Ganes Gemi Putra (Team 8)
AHMAD AL-FARIZI (Team 8)
KAFKA PUTRA RIYADI (Team 8)
NAURA AISHA ZAHIRA (Team 8)
FAISHAL ARIF SETIAWAN (Team 8)

Jumlah data di data1: 5
Jumlah data di data2: 5

D:\PRAKTIKUM KONTRUKSI PERANGKAT LUNAK\13_Clean_Code_Standard\JURNAL12_2311104075\JURNAL13_2311104075\JURNAL13_2311104075\bin\Debug\JURNAL13_2311104075.exe (process 1468) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Penjelasan Kode

Kode yang Anda tulis sudah benar dan lengkap. Berikut detailnya:

1. Langkah D.D (Cetak Data Sebelum Dihapus)

```
Console.WriteLine("=== Data Sebelum Dihapus ===");  
data2.PrintSemuaData();
```

✓ **Sudah diisi:** Menampilkan semua data sebelum penghapusan.

2. Langkah D.E (Hapus Data Asisten)

```
data2.HapusSebuahData(5);
```

✓ **Sudah diisi:** Menghapus data asisten di indeks 5 (indeks valid karena list memiliki 6 elemen).

3. Langkah D.F (Cetak Data Setelah Dihapus)

```
Console.WriteLine("\n=== Data Setelah Dihapus ===");  
data1.PrintSemuaData();
```

✓ **Sudah diisi:** Menampilkan data setelah penghapusan.

4. Langkah D.G (Cetak Jumlah Data)

```
Console.WriteLine($"Jumlah data di data1: {data1.GetSemuaData()}");  
Console.WriteLine($"Jumlah data di data2: {data2.GetSemuaData()}");
```

✓ **Sudah diisi:** Menampilkan jumlah data yang konsisten di data1 dan data2.

