

**JURNAL MANDIRI 13**  
**KONSTRUKSI PERANGKAT LUNAK**  
**S1 SOFTWARE ENGINEERING**



**ILHAM LII ASSIDAQ**

**2311104068**

**SE0702**

**DIREKTORAT TELKOM UNIVERSITY PURWOKERTO**

1. Link GitHub : [https://github.com/ilhamlii21/KPL\\_ILHAM\\_LII\\_ASSIDAQ-2311104068\\_S1SE-07-02](https://github.com/ilhamlii21/KPL_ILHAM_LII_ASSIDAQ-2311104068_S1SE-07-02)

2. A. Program.cs

```
1 using JAMODUL13;
2
3 public class Program
4 {
5     public static void Main(string[] args)
6     {
7         var data1 = PusatDataSingleton.GetDataSingleton();
8         var data2 = PusatDataSingleton.GetDataSingleton();
9
10        data1.AddSebuahData("Ilham");
11        data1.AddSebuahData("Christoper");
12        data1.AddSebuahData("William");
13
14        Console.WriteLine("Data di data2:");
15        data2.PrintSemuaData();
16
17        data2.HapusSebuahData(2);
18
19        Console.WriteLine("\nSetelah penghapusan, data di data1:");
20        data1.PrintSemuaData();
21
22        Console.WriteLine($"Jumlah data di data1: {data1.GetSemuaData().Count}");
23        Console.WriteLine($"Jumlah data di data2: {data2.GetSemuaData().Count}");
24    }
25 }
```

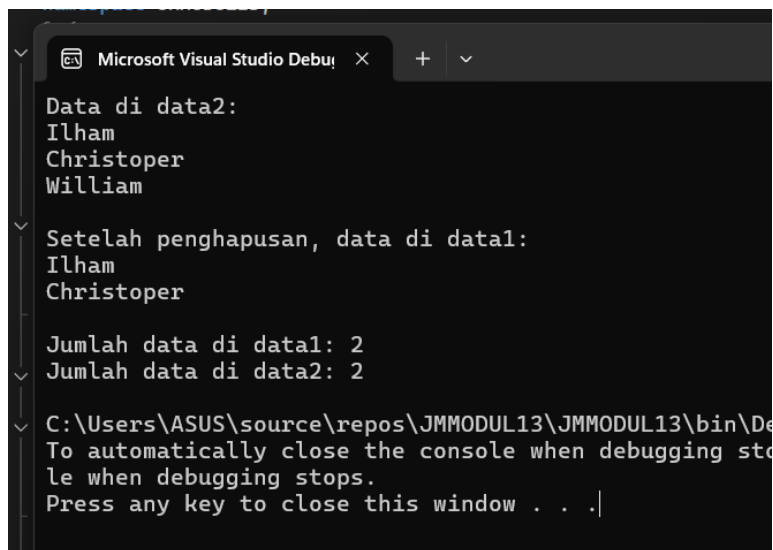
Kode di atas merupakan implementasi dari pola desain Singleton, di mana objek `PusatDataSingleton` hanya dibuat satu kali dan digunakan secara global. Pada `Main`, dua variabel (`data1` dan `data2`) mengambil instance yang sama dari `PusatDataSingleton` melalui metode `GetDataSingleton()`. Data yang ditambahkan melalui `data1` akan muncul juga ketika dipanggil dari `data2`, karena keduanya menunjuk ke objek yang sama. Penghapusan data melalui `data2` pun berdampak langsung pada isi `data1`, yang terlihat dari hasil output jumlah dan isi data. Ini menunjukkan bahwa Singleton menjaga satu-satunya sumber data yang konsisten dalam keseluruhan aplikasi.

B. PusatDataSingleton.cs

```
1 namespace JAMODUL13;
2
3 public class PusatDataSingleton
4 {
5     private static PusatDataSingleton _instance;
6     public List<string> DataTersimpan { get; private set; }
7
8     private PusatDataSingleton()
9     {
10         DataTersimpan = new List<string>();
11     }
12
13     public static PusatDataSingleton GetDataSingleton()
14     {
15         if (_instance == null)
16         {
17             _instance = new PusatDataSingleton();
18         }
19         return _instance;
20     }
21
22     public List<string> GetSemuaData()
23     {
24         return DataTersimpan;
25     }
26
27     public void PrintSemuaData()
28     {
29         foreach (string data in DataTersimpan)
30         {
31             Console.WriteLine(data);
32         }
33     }
34 }
```

Kode tersebut merupakan contoh penggunaan *design pattern* Singleton melalui kelas `PusatDataSingleton` dari namespace `JMMODUL13`. Dalam `Main()`, objek `data1` dan `data2` sama-sama memanggil instance yang sama dari `PusatDataSingleton` menggunakan metode `GetDataSingleton()`. Karena Singleton hanya memiliki satu instance global, maka penambahan data melalui `data1` juga dapat diakses oleh `data2`, dan sebaliknya. Saat data dihapus melalui `data2`, perubahan ini juga terlihat saat mencetak data dari `data1`. Output menunjukkan bahwa kedua variabel mengakses dan memanipulasi data yang sama, menegaskan prinsip utama dari Singleton: satu sumber data yang terpusat dan konsisten di seluruh aplikasi.

#### Running



```
Microsoft Visual Studio Debug Console
Data di data2:
Ilham
Christoper
William

Setelah penghapusan, data di data1:
Ilham
Christoper

Jumlah data di data1: 2
Jumlah data di data2: 2

C:\Users\ASUS\source\repos\JMMODUL13\JMMODUL13\bin\De
To automatically close the console when debugging stops
le when debugging stops.
Press any key to close this window . . .|
```

Output di atas menunjukkan bahwa `data1` dan `data2` merujuk pada instance yang sama dari kelas `PusatDataSingleton`, sehingga semua operasi yang dilakukan melalui salah satu variabel berdampak pada data yang sama. Setelah tiga data ditambahkan melalui `data1`, `data2` menampilkan ketiganya karena mereka berbagi instance. Ketika `data2` menghapus data pada indeks ke-2 (yaitu "William"), hasil penghapusan tersebut juga tercermin saat mencetak isi `data1`, yang kini hanya menampilkan "Ilham" dan "Christoper". Hal ini juga dikonfirmasi oleh jumlah data pada kedua variabel yang sama-sama berjumlah dua, membuktikan bahwa hanya ada satu sumber data aktif, sesuai dengan prinsip dari pola Singleton.