TUGAS PENDAHULUAN 13

KONSTRUKSI PERANGKAT LUNAK

S1 SOFTWARE ENGINEERING

ILHAM LII ASSIDAQ

2311104068

SE0702

DIREKTORAT TELKOM UNIVERSITY PURWOKERTO

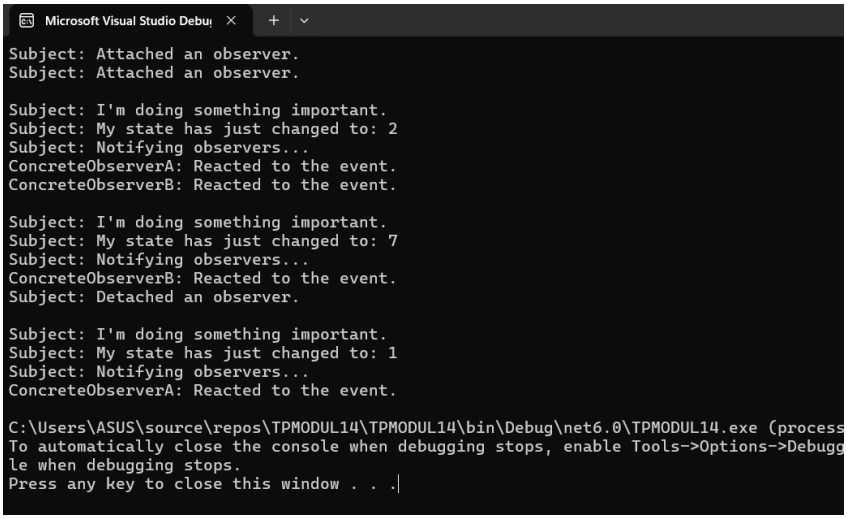
1. Link GitHub : https://github.com/ilhamlii21/KPL_ILHAM_LII_ASSIDAQ-2311104068_S1SE-07-02

2. A. Program.cs

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

namespace RefactoringGuru.DesignPatterns.Observer.Conceptual
{
    8 references
    public interface IObserver
    {
        3 references
        void Update(ISubject subject);
    }

    4 references
    public interface ISubject
    {
        3 references
        void Attach(IObserver observer);

        // Detach an observer from the subject.
        2 references
        void Detach(IObserver observer);

        // Notify all observers about an event.
        2 references
        void Notify();
    }

    // The Subject owns some important state and notifies observers when the
    // state changes.
    4 references
    public class Subject : ISubject
    {
        // For the sake of simplicity, the Subject's state, essential to all
        // subscribers, is stored in this variable.
        5 references
```

Kode tersebut merupakan implementasi dari design pattern Observer dalam bahasa C#. Pola ini memungkinkan objek Subject untuk memberi tahu sejumlah Observer ketika terjadi perubahan status internalnya. Kelas Subject menyimpan status dalam properti State dan daftar observer-nya dalam _observers. Saat SomeBusinessLogic() dipanggil, status akan berubah secara acak, dan semua observer yang terdaftar akan diberi tahu melalui metode Notify(). Dua observer konkrit (ConcreteObserverA dan ConcreteObserverB) bereaksi terhadap perubahan status berdasarkan kondisi tertentu. Program ini mencerminkan prinsip pemisahan antara sumber data (subject) dan para pendengar (observers), memungkinkan sistem lebih fleksibel dan modular.

Running



Output tersebut menunjukkan proses kerja pola Observer saat objek Subject menjalankan logika bisnisnya dan memberitahu para observer. Pertama, dua observer (ConcreteObserverA dan ConcreteObserverB) didaftarkan ke Subject. Saat status berubah menjadi 2, keduanya bereaksi karena memenuhi kondisi masing-masing. Pada perubahan status kedua menjadi 7, hanya ConcreteObserverB yang bereaksi karena ConcreteObserverA hanya merespons jika status kurang dari 3. Setelah ConcreteObserverB dilepas, perubahan status terakhir menjadi 1 hanya memicu reaksi dari ConcreteObserverA. Ini menunjukkan bagaimana observer hanya merespons jika terdaftar dan jika kondisi mereka terpenuhi, mencerminkan fleksibilitas dan efisiensi pola desain Observer.