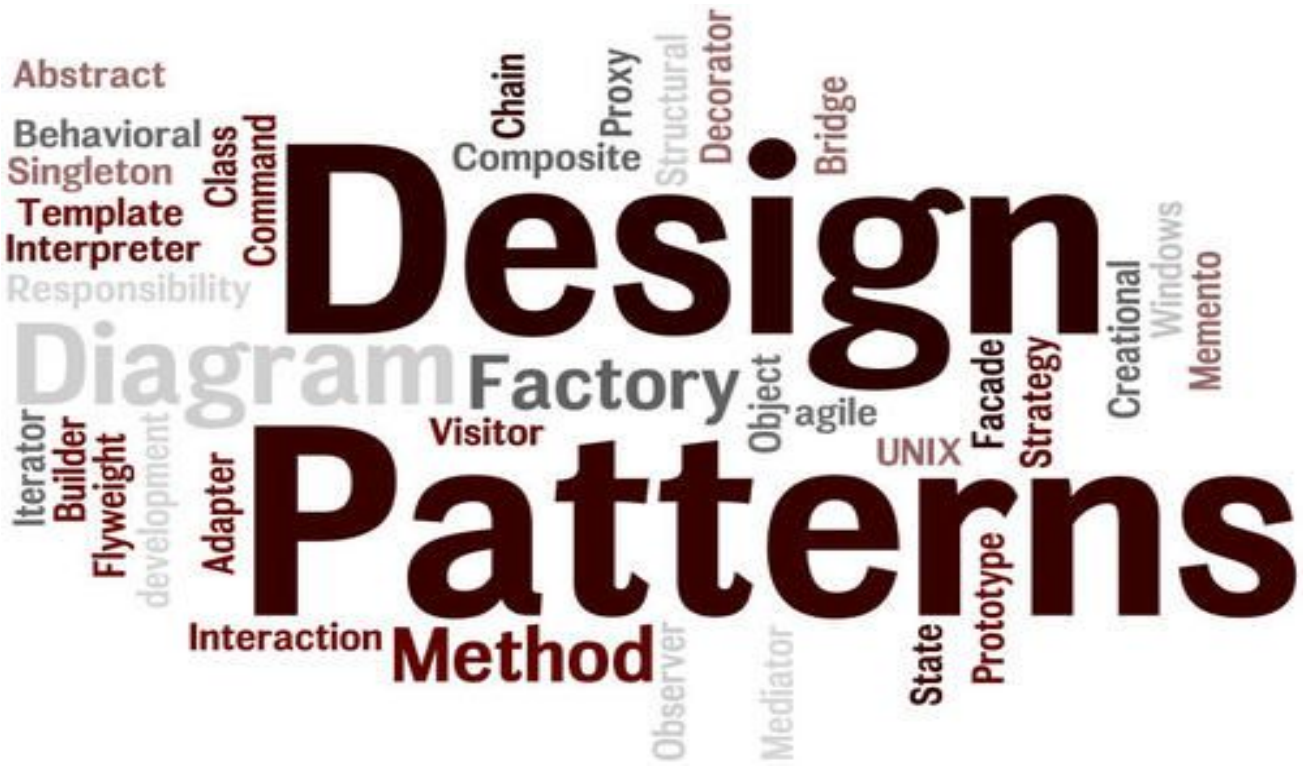
Restaurant bestelpaal

Oplevering Design Patterns



Versie 1.0
03 April 2020

Docenten:
Gerjan van Oenen

Groep:
Yaron Lambers - 1569488
Thomas Christerus - 1550965
Lennart Pikijn - 1576565
Michiel Jansen - 1571997

Inhoudsopgave

Planning

Om het vak “Design patterns” succesvol als groep te kunnen voltooien, hebben wij een planning gemaakt die ons hierbij ondersteunt. Tevens hebben wij voor het maken van het programma tickets gebruikt, die vertelde welke taak uitgevoerd moest worden.

Weeknummer	**Planning design patterns**
Week 1	Brainstormen over het idee
Week 2	Brainstormen over het idee en logische patterns bedenken
Week 3	Brainstormen over het idee en logische patterns bedenken
Week 4	Klassendiagram realiseren
Week 5	Inlevering startdocument 1e keer
Week 6	Verbetering startdocument en klassendiagram
Week 7	Inlevering startdocument 2e keer Project uitwerken
Week 8	Project uitwerken Project voltooien

Code of conduct

De class opbouw moet worden gehanteerd zoals is aangegeven in dit document. Aanvullende informatie is te vinden via deze link.

Class opbouw

```
    /// <summary>
    ///
    /// </summary>
public class TestClass
{
    private int _count;                 // Description variable
    private String _name            // Description variable

    /// <summary>
    ///
    /// </summary>

    public TestClass(String name)
    {
        // Setting class attributes
        _name = name;
        _count = 0;
    }

    public String Name
    {
        get { return _name; }
        set { _name = value; }
    }
```

```
    /// <summary>
    ///
    /// </summary>
    /// <param name="name">param description</param>
    public void SetName(String name)
    {
        // Add 'is my name!' to the name variable
        name += " is my name!";

        // Set the name variable
        _name = name;
    }

     /// <summary>
    ///
    /// </summary>
    public void PublicTestMethod()
    {

    }



    /// <summary>
    ///
    /// </summary>
    protected private void ProtectedTestMethod()
    {

    }

     /// <summary>
     /// Description method
     /// </summary>
     /// <param name="count">param description</param>
    private String privateTestMethod(int count)
    {
        return "";
    }
}
```

Versiebeheer

Wij hebben gebruik gemaakt van de Git Flow waar twee beschermde branches zijn aangemaakt: master en develop. Vanuit de develop zijn telkens nieuwe featurebranches aangemaakt op basis van vooraf bepaalde tickets. Na het afhandelen van een issue werd een pull request aangemaakt naar de develop branch. Voordat een pull request gemerged werd in de develop branch, werd er eerste een code review uitgevoerd door een ander groepslid. De contributions zijn geen goede peilers omdat voor complexe problemen gebruik is gemaakt van screenshare.

UML

Design Patterns UML

L. Pikijn, Y. Lambers, T. Christerus, M. Jansen | April 3, 2020