

**Object Georiënteerd Ontwerp**

**Zelfevaluatie App**

Steven Zegers

Wout De Boeck

Thibault Stroobants

2018 Toegepaste Informatica Reeks 2

[Datum]

University Colleges Leuven-Limburg  
Campus Proximus  
Geldenaaksebaan 335  
3001 Heverlee  
Tel. +32 (0) 16 375 700

Inhoud

[Algemene opmerking 3](#_Toc499494737)

[Vereisten 3](#_Toc499494738)

[Design patterns 5](#_Toc499494739)

[Speciale topics 7](#_Toc499494740)

[Werkverdeling 8](#_Toc499494741)

[Klassen diagram 9](#_Toc499494742)

# Algemene opmerking

Dit verslag gaat over de opdracht ‘Zelfevaluatie app’, dat zal meetellen voor 5 punten van je totaalscore van dit vak. Dit verslag vormt de toelichting bij jouw code.

**Je wordt geacht om je voor het verslag volledig aan deze template te houden. Buiten deze ‘algemene opmerking’ zijn alle paragrafen verplichte onderdelen van het verslag. Je print dit verslag af (dubbelzijdig) – 1 exemplaar per groep en levert dit in bij het einde van het laatste labo OOO.**

**Je maakt een zip file van al je source code (.java bestanden, geen .class bestanden) en van alle bestanden die van belang zijn voor deze OOO opdracht. Je voegt tevens de laatste versie van je verslag (Word document) toe aan deze zip file.**

**Je maakt tevens een jar file van je programma die kan runnen onder Java 8 (of hoger)**

**. Je laadt deze zip file en jar file op via Toledo uiterlijk voor maandag 24/12 23.59 uur. Naam van de zip file: Zelfevaluatie\_ gevolgd door de familienamen van de groepsleden gescheiden door underscore (bv. Zelfevaluatie\_Dox\_Lenaerts\_Jansen.zip). Idem voor jar file (bv. Zelfevaluatie\_Dox\_Lenaerts\_Jansen.jar)**

**Het spreekt voor zich dat de code op Toledo overeenstemt met de code in je repository op GitHub.**

# URL GITHUB repository

**Copy/paste hier de URL van je Github repository met je zelfevaluatie app project**

# URL

https://github.com/steven-zegers/OOO\_projects

# Vereisten

Geef voor het gehele project aan welke vereisten (evt. zelf verder uitgewerkt/opgesplitst aan de hand van de opgave) je succesvol hebben geïmplementeerd, en welke topics niet gelukt zijn. Indien je een deel van de voorziene vereisten niet afgewerkt hebt, geef dan aan waarom niet… De reden kan “tijdsgebrek” zijn, het kan een issue zijn “wist niet hoe, het crashte”, of het kan zijn dat je een zeer goede reden had om het niet te implementeren…

Voeg het definitieve gegenereerde klassendiagramma van je code toe, als afzonderlijke afbeelding, als bijlage bij dit rapport.

Als er zaken zijn uit de opdracht die je niet hebt kunnen uitwerken of die je beter zou willen uitwerken, dan som je deze hier tevens op (in laatste rij van tabel).

|  |  |  |
| --- | --- | --- |
| User story | OK? | indien niet ok – wat lukt niet (zie acceptatie-criteria) en waarom? |
| 1. Overzicht categorieën tonen | OK |  |
| 1. Nieuwe categorie toevoegen | OK |  |
| 1. Overzicht vragen tonen | OK |  |
| 1. Nieuwe vraag toevoegen | OK |  |
| 1. Test afleggen | OK |  |
| 1. Resultaat tonen met punten | OK |  |
| 1. Resultaat tonen met feedback | OK |  |
| 1. Type feedback kiezen | OK |  |
| 1. Resultaat tonen zonder fouten | OK |  |
| 1. Test reeds afgelegd |  |  |
| 1. Test opnieuw afleggen | OK |  |
| 1. Bestaande categorie aanpassen |  |  |
| 1. Bestaande vraag aanpassen |  |  |
| 1. Test selecteren uit lijst van excel bestanden |  |  |

# Design patterns

Geef voor elk gezien patroon aan waar je het gebruikt hebt (mogelijkerwijs meer dan eens). Genereer een klassendiagram voor elke situatie waarin je het patroon gebruikt hebt.

Geef extra informatie (voordelen / waarom / …). Als je een patroon niet toegepast hebt, leg uit waarom niet.

|  |  |  |
| --- | --- | --- |
|  | Toegepast (ja/nee)  In welke stories(nr)  Waarom toegepast(voordeel) | Bijhorend klassendiagram (gegenereerd vanuit je java code) |
| Observer | Ja, om het updaten van de verschillende labels en panels te kunnen realiseren met de “update” methode |  |
| Strategy | Ja, in de database klasses, aangezien het uitlezen van de files afhangt van of het gaat om categories of questions   * Story 1, 2, 3, 4 |  |
| Simple Factory |  |  |
| Façade |  |  |
| Singleton |  |  |
| State |  |  |
| MVC | Ja |  |
| Decorator |  |  |
| Template method |  |  |

# Speciale topics

Geef voor elke “speciale topic” aan of je het gebruikt hebt of niet, en zo ja, waar. Toon eventueel aan met een klassendiagram.

|  |  |  |
| --- | --- | --- |
|  | Toegepast (ja/nee)  In welke stories(nr)  Waarom toegepast(voordeel) | Bijhorend klassendiagram (uit object aid) of extra uitleg |
| Enum |  |  |
| Properties |  |  |
| Reflection |  |  |
| Andere… |  |  |

# Werkverdeling

Geef aan in percentages hoeveel je bij benadering gespendeerd hebt aan deze opdracht.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Steven Zegers | Wout De Boeck | Thibault Stroobants | Totaal |
| Ontwerp |  |  |  | 100% |
| Klassendiagrammen |  |  |  | 100% |
| Implementatie |  |  |  | 100% |
| Verslag |  |  |  | 100% |

# Klassen diagramma’s

Voeg het definitieve gegenereerde klassendiagram van je code toe), als afzonderlijk afbeelding-bestand. **Dit klassendiagram moet goed leesbaar zijn.** Spreid het over meerdere pagina’s (bv. 1 blad per package (MVC) en 1 overzicht klassendiagram (zonder attributen en methoden).