

**Object Georiënteerd Ontwerp**

**Bestel App Broodjeszaak**

Milo Claes

Sander Doggen

Devy Leskens

Jelle Michiels

Greg François Reynders

2TI reeks 5/6

29/12/2021

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

# Algemene opmerking

Dit verslag gaat over de opdracht ‘Bestel app broodjeszaak’, 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 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 laadt deze zip file op via Toledo uiterlijk op woensdag 29/12/2021 - 23.59 uur. Naam van de zip file: volgnr\_familienaam1\_familienaam2\_familienaam3\_familienaam4\_Broodjeszaak\_2021\_2022 De familienamen zijn oplopend alfabetisch geordend (vb 17\_Cox\_Dox\_Jansen\_Lenaerts\_ Broodjeszaak\_2021\_2022).**

**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 bestel app project**

# URL

https://github.com/DevyLeskens/1\_Claes\_Doggen\_Leskens\_Michiels\_Reynders\_Broodjeszaak\_2021\_2022

# 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…

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 en waarom? |
| 1. Overzicht broodjes/beleg tonen | OK |  |
| 1. Overzicht broodjes/beleg tonen met keuze uit tekst of excel | OK |  |
| 1. Een nieuwe bestelling opstarten | OK |  |
| 1. Broodjes/beleg toevoegen | OK |  |
| 1. Identieke bestellijnen toevoegen | OK |  |
| 1. Bestellijn Wissen   Bestelling annuleren | OK |  |
| 1. Bestelling afsluiten | OK |  |
| 1. Bestelling betalen en doorsturen naar keuken | OK |  |
| 1. Bestelling opvragen en afwerken in keuken | OK |  |
| 1. Aanpassen settings | OK |  |

# 

# 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 | Het observer patroon werd gebruikt zodat we bij de start van een nieuw event de ingeschreven observers konden aanpassen.  Stories: 2 |  |
| Strategy | Strategy werd gebruikt voor onze verschillende discounts die er mogelijk zijn. Omdat deze allemaal enkel nodig zijn voor het berekenen van de korting. Dit deden we ook voor het loaden en saven  Story: 2 (LoadSafe), 7 (Discounts) |  |
| Simple Factory | Voor het maken van discounts hebben we een discountFactory aangemaakt zodat deze de discount aanmaakte. Dit is ook zo voor onze loadSaveStrategy.  Stories: 2, 7 |  |
| Façade | We hebben gebruikt gemaakt van een façade zodat onze applicatie via deze klasse aan alle andere methodes en klassen geraakt.  Stories: 1 |  |
| Singleton | Dit werd toegepast in onze topping- en sandwichDatabase en onze orderFacade zodat er maar één instantie aangemaakt kan worden en er dus altijd in de juist database gewerkt word.  Stories: 1, 2 |  |
| State | Voor onze orders hebben we state pattern gebruikt zodat we aan de hand van de states waarin een order zich bevind het toegang geven tot bepaalde methodes zoals bijvoorbeeld pay.  Stories: 1, 3, 4, 5, 6, 7, 8, 9 |  |
| MVC | Het MVC pattern is gebruikt bij onze verschillende views (admin, kitchen en order). Deze hebben ook elk een controller. Deze controllers kunnen via de orderFaçade aan alle andere methodes en klassen die zich ook bevinden in het model package.  Stories: 1 |  |
| Template method | Tijdens story één hebben we dit gebruikt in onze textLoadSaveTemplate klasse zodat het lezen en schrijven telkens volgens een vast stappenplan gebeurd.  Stories: 1 |  |

# 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 of extra uitleg |
| Enum | We hebben dit toegepast voor al onze loadSaveStrategies, discountStrategies en onze odersEvents. De orderEvents werden als enum gebruikt zodat wanneer we onze observers wouden notifiëren we op basis van onze orderEvents de juiste observers notifieerde.  Dit was tijdens volgende stories: 7 |  |
| Properties | Dit werd toegepast tijdens story tien zodat men in het settings tab van de admin view kon kiezen hoe alles bewaard zou worden.  Stories: 10 |  |
| Reflection | Refelection wordt gebruikt voor het aanmaken aanmaken van de loadsave strategies, en de discount strategies. Zo hoefde we niets meer te hardcode! |  |

|  |  |  |
| --- | --- | --- |
| Andere… | We hebben gebruikt gemaakt van het adapter pattern. Dit deden we zodat we via excel konden inlezen.  Stories: 2 |  |

# Werkverdeling

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Claes | Doggen | Leskens | Michiels | Reynders | Totaal |
| Ontwerp | 25% | 45% | 10% | 10% | 10% | 100% |
| Implementatie | 10% | 15% | 15% | 10% | 50% | 100% |
| Verslag | 10% | 10% | 30% | 40% | 10% | 100% |

# Klassen diagramma’s

Voeg het definitieve gegenereerde klassendiagram van je code toe.

**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). GEEN KLASSENDIAGRAM MET ZWARTE ACHTERGRONKLEUR AFPRINTEN!!!!!!!!!!

Afbeelding met tekst, schermafbeelding, binnen, computer

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermafbeelding, binnen, verschillende

Automatisch gegenereerde beschrijving