**EINDOPLEVERING**

|  |  |
| --- | --- |
| **Naam** | Jay Bhageloe |
| **Studentnummer** | 22072039 |
| **Klas** | SE.H5 |

**OPDRACHT**

**Geef hieronder een korte beschrijving van je opdracht (een samenvatting van je productvisie) zodat je docent OPT3 ook begrijpt met welke opdracht jij bezig bent (neem dit onderdeel over uit je laatst ingeleverde opdracht en pas het aan aan de hand van je eigen bevindingen en aan de hand van feedback van medestudenten en je docent aan):**

Mijn opdracht helpt de productgever met het plannen dat een andere manier gebruikt dan wat de school aanbeveelt aan de studenten. Een applicatie waar je invoert hoeveel lessen, projecten, deadlines of toetsen je hebt. Integendeel tot een normale kalender kan je hier toevoegen hoeveel keer je nodig hebt voor een toets voordat je het lesmateriaal onder de knie hebt. Voor lessen krijg je herinneringen dat je je werkcollege of lesmateriaal voorbereiding moet doen ook al is het maar voor 15 minuten, het is handiger dat je een herinnering krijgt en een paar minuten besteed dat helemaal niets doet.

Voor projecten en deadlines krijg je ook een countdown hoeveel dagen je nog overhebt want een visuele representatie van wanneer iets af moet is veel motiverend voor de klant dan een normale kalender bericht. Je kan ook tijden aangeven wanneer je meestal dingen voor school aan het doen bent bijvoorbeeld laat in de avond of juist tussen 5 en 8 voor werk. De app kan in die tijd aangeven wat er gedaan kan worden in die tijd en of er iets urgents afgemaakt moet worden

**GITHUB**

Maak je GitHub-repo Private en geef je docent voor het werkcollege toegang tot je repo (zie paragraaf 3 in [de uitleg bij deze opdracht op BrightSpace](https://dehaagsehogeschool.sharepoint.com/:b:/r/sites/AgileOOProgrammingSemesterSE-S2_groups-Docenten/Gedeelde%20documenten/Docenten/H-SE-S2OPT3/Lesmateriaal%20(publiek)/OPT3%2014.1%20-%20Realisatie%20zonder%20code%20smells%20-%20Toelichting.pdf?csf=1&web=1&e=MHo6uM) hoe je de repo *clean* houdt en hoe je zorgt dat je een project Maven-*aware* maakt). Stuur niet steeds een nieuwe link/uitnodiging, maar bouw in feature-branches verder aan de applicatie die je eerder hebt opgeleverd:

<https://github.com/SchoolAccount22072/ProjectCodeSmells>

**UML-ontwerp voor OPT2**

Bij deze eindoplevering ben ik uitgegaan van de volgende User Stories (gebruik de ‘+’ rechts van de tabel om een User Story toe te voegen):

|  |
| --- |
| **Als** eigenaar  **wil ik** een interface waar ik gemakkelijk mijn informatie kan invoeren en kan opslaan en terugzien **zodat ik** de deadlines, taken en toetsen kan inzien en later kan zien of de vakken, projecten en deadlines zijn voldaan of niet**.** |
| Alseigenaar wil ik interface waar ik op een gesorteerde manier de informatie kan inzien zodat ik precies weet hoeveel tijd er is voor elk vak, deadline, toets, etc. |
| Alseigenaar wil ik specifieke informatie (datum, tijd, vak) kan invoeren voor elke deadline zodat ik een duidelijk overzicht heb van mijn verplichtingen. |
| Alseigenaar wil ik dat er alleen relevante informatie in de applicatie komt en niet informatie zoals individuele lessen, vrije dagen, bijlessen, etc. zodat ik alleen de belangrijke informatie kan inzien en niet overbelast raak met onnodige informatie. |
| Alseigenaar wil ik een overzichtelijke visualisatie van de progressie op basis van gehaald of niet gehaald toetsen en of projecten zodat ik kan inzien of de applicatie effectief is en of alle taken zoals deadlines zijn voldaan.. |

**EINDONTWERP**

Voeg hieronder het eindontwerp toe dat je als resultaat hebt opgeleverd bij de vorige opdracht (met eventuele wijzigingen als je dat op basis van feedback van je docent en/of medestudenten wijs leek):

<Copy-Paste hier een kopie van je eindontwerp (opdracht 2 voor OPT3 met patterns en SOLID)>

\* Controleer of het plaatje goed leesbaar is. Als je wilt dat je docent de eerste opdracht voor OPT3 opnieuw beoordeelt, lever je het ingevulde template voor opdracht 1 in via [de inleverbox voor opdracht 1 in BrightSpace](https://brightspace.hhs.nl/d2l/le/lessons/56739/topics/736366).

A diagram of a company

Description automatically generated

ZOOM IN VOOR UML DIAGRAM

**EINDOPLEVERING**

Voor de huidige opdracht (de realisatie zonder *code* smells) heb je het eindontwerp uit opdracht 1 (het ontwerp met *patterns* & SOLID) nu omgezet naar werkende code en heb je daaruit de *code smells* verwijderd. Dat laatste is het doel van deze opdracht en daar ligt dus ook het zwaartepunt van de beoordeling van de geschreven code. Hieronder kun je in een checklist nagaan of je alle relevante *Code Smells* uit je code hebt verwijderd en kun je toelichten welke techniek je hebt gebruikt om specifieke *Code Smells* op te ruimen (dat kan dus ook door te verwijzen naar de toepassing van *patterns* en SOLID-principes waarmee je hebt voorkomen dat de *code smells* in je code binnensluipen).

Zie het volgende document voor voorbeelduitwerking en aanwijzingen:

‘[OPT3 14.1 - Realisatie zonder code smells - Toelichting.pdf](https://dehaagsehogeschool.sharepoint.com/:b:/r/sites/AgileOOProgrammingSemesterSE-S2_groups-Docenten/Gedeelde%20documenten/Docenten/H-SE-S2OPT3/Lesmateriaal%20(publiek)/OPT3%2014.1%20-%20Realisatie%20zonder%20code%20smells%20-%20Toelichting.pdf?csf=1&web=1&e=hLzEIv)’

**CHECKLIST *CODE SMELLS***

Long Method

Large Class

Temporary Field

Long Parameter List

Duplicate Code

Switch-statements

Primitive Obsession

Divergent Change

Shotgun Surgery

**VERANTWOORDING *CODE SMELLS***

|  |  |  |
| --- | --- | --- |
| ***Code Smell*** | **Oude situatie** | ***Vervanging*** |
| Long Method | Eerst was het zo dat de beheerKeuze in de Menu klasse te lang was en te veel responsibilities had wat tegen de SOLID principes ingaan | Daarom is de code zo veranderd dat er nu meerdere kleinere methodes zijn voor het uitvoeren van opties |
| Dit komt overeen met de SRP omdat nu elke methode een responsibility heeft ipv meedere. | |
| Switch Statements | In de beheerKeuze menu was het zo dat er meerdere switch statements waren waardoor de code te gecompliceerd werd en dat het moeilijk zou zijn om het te extenden zonder de code te veranderen wat tegen de Open Closed principle ingaat | In de nieuwe situatie is er een factory pattern gebruikt dat gebruik maakt van polymorfisme zodat bij verschillende acties er verschillende reacties komen. |
| De code komt overeen met OCP omdat je de code nu kan extenden zodat dat je de code zelf moet veranderen. | |
| Duplicate Code | In de voerInfoIn was het zo dat in verschillende klassen er duplicate code was in verschillende plekken waardoor je als er bij 1 een probleem is zijn er bij de andere ook een probleem dus moet je alles steeds opnieuw veranderen. | Nu is het zo dat er een centrale class is voor de logica voor de deadline functie wat dan geïmplementeerd wordt door de project en toets class. |
| De code komt overeen met de SRP want nu heeft de nieuwe klasse alleen 1 responsibility en dat is het uitvoeren van deadlines | |
| Primitive Obsession | Eerst wat het zo dat de code alleen maar strings en integers ging gebruiken maar dit werd gedaan in meerdere klassen ipv 1 centrale plek. | Dus nu voor de Datum is het zo dat we i.p.v. een String of Int voor de data een DateInfo klasse hebben dat de datum op een centrale plek verwerkt i.p.v. meerdere klassen met aparte logica per klassen. |
| Dit komt weer terug op het SRP waar de dataInfo een responsibility waar het de logica voor de data gebruikt en dat weer wordt geïmplementeerd in verschillende klassen. | |