Projectwijzer

Software development

**Het project:**

Titel Training Factory Tiel

Datum 16-03-2023

Projectgroep – namen deelnemers

|  |  |  |
| --- | --- | --- |
| **Algemene informatie** | | |
| Kwalificatiedossier en cohort | Software development | 2020 en verder |
| Profiel, niveau en crebocode | P1: Software developer, 4 | 25604 |
| Kerntaak | B1-K1: Realiseert software  B1-K2: Werkt in een ontwikkelteam | |
| Werkprocessen | B1-K1-W1: Plant werkzaamheden en bewaakt de voortgang  B1-K1-W2: Ontwerpt software  B1-K1-W3: Realiseert (onderdelen van) software  B1-K1-W4: Test software  B1-K1-W5: Doet verbetervoorstellen voor de software  B1-K2-W1: Voert overleg  B1-K2-W2: Presenteert het opgeleverde werk  B1-K2-W3: Reflecteert op het werk | |

Inhoud

[1. De productomschrijving 4](#_Toc114826233)

[1.1. De klant, opdrachtgever of doelgroep 4](#_Toc114826234)

[1.2. De probleemstelling 4](#_Toc114826235)

[1.3. De ontwikkeltool, de ontwikkelomgeving 4](#_Toc114826236)

[1.4. De randvoorwaarden 4](#_Toc114826237)

[2. De planning 4](#_Toc114826238)

[2.1. Product backlog 5](#_Toc114826240)

[2.2. Sprint backlog, burn down chart 5](#_Toc114826241)

[2.3. Reflectieverslag 7](#_Toc114826242)

[3. Het ontwerp 7](#_Toc114826243)

[3.1. Een ERD 7](#_Toc114826245)

[3.2. Datadictionairy en relaties 8](#_Toc114826246)

[3.3. Een klassendiagram 8](#_Toc114826247)

[3.4. Wireframe 9](#_Toc114826248)

[3.5. Navigatiestructuur 9](#_Toc114826249)

[3.6. Mock ups 9](#_Toc114826250)

[4. Realisatie van eigen user story’s 10](#_Toc114826251)

[4.1. Programmeren 10](#_Toc114826253)

[4.2. Test scenario’s schrijven en uitvoeren 10](#_Toc114826254)

[4.3. Versiebeheer bijhouden, commits met git 10](#_Toc114826255)

[4.4. Documenteren (in de code of apart) 10](#_Toc114826256)

[4.5. Een database inrichten/aanpassen/aanvullen (indien nodig en afhankelijk van de situatie) 10](#_Toc114826257)

[4.6. Een codereview (onderling) op basis van de code conventies 10](#_Toc114826258)

[4.7. Voortgang in sprint doorspreken (stand up) 10](#_Toc114826259)

[4.8. Een planning bijhouden (werk de burn down chart bij) 10](#_Toc114826260)

[4.9. Opleveren van sprint product 10](#_Toc114826261)

[4.10. Retrospective/ reflectieverslag 10](#_Toc114826262)

[4.11. Stel conclusies en verbetervoorstellen op voor je eigen user story’s. 10](#_Toc114826263)

[4.12. Review met klant 10](#_Toc114826264)

[4.13. Bundel de verbetervoorstellen van de review van de klant van jouw user story’s in een document. 10](#_Toc114826265)

[4.14. Leg de afspraken vast, neem ze mee met de volgende sprint 10](#_Toc114826266)

[4.15. Voeg met het team de verschillende user story’s samen. 10](#_Toc114826267)

[4.16. Stel een scenario (inclusief testdata) op voor de gezamenlijke test. 10](#_Toc114826268)

[4.17. Voer een test uit voor alle user story’s uit de sprint die samen het product maken. 10](#_Toc114826269)

[5. Eindgesprek 11](#_Toc114826270)

[5.1. Presenteer het product 11](#_Toc114826272)

[5.2. Geef feedback op het proces 11](#_Toc114826273)

[5.3. Geef feedback op het product 11](#_Toc114826274)

[5.4. Toon aan wat je hebt gedaan met afspraken 11](#_Toc114826275)

[5.5. Reflecteer op je eigen functioneren 11](#_Toc114826276)

[6. Resultaat 11](#_Toc114826277)

[6.1. Ingevulde projectwijzer 11](#_Toc114826279)

[6.2. Applicatie en gegevensverzameling, via github/gitlab 11](#_Toc114826280)

De productomschrijving

* 1. De klant, opdrachtgever of doelgroep

Naam, of omschrijving, verder aangeduid als de klant.

**De doelgroep is volwassenen, jeugd van 12 tot 18 jaar en kinderen van 6 tot 12 jaar.**

* 1. De probleemstelling

Een project wordt uitgevoerd omdat een klant een wens of een noodsituatie heeft. In software development spreek je dan van een probleem (dat opgelost gaat worden).

Beschrijf de bestaande situatie (wat is er nu) en de gewenste situatie (wat moet er komen).

Beschrijf de eisen en wensen om tot de gewenste situatie te kunnen komen.

Beschrijf het product dat volgens jou aan de eisen en wensen kan voldoen.

Leg vast wanneer het product af is (Definition of done). Denk daarbij ook aan code conventies en veiligheid.

* 1. De ontwikkeltool, de ontwikkelomgeving

1. Beschrijf welke programmeer taal je gaat gebruiken

**PHP**

1. Beschrijf welk framework of engine je gaat gebruiken

**CodeIgniter**

1. Beschrijf de ontwikkelomgeving

**Ik gebruik voor deze opdracht VSC**

1. Beschrijf de testomgeving

**Ik test het op school**

* 1. De randvoorwaarden

1. Beschrijf wat buiten je probleemstelling valt
2. Beschrijf wat jouw beperkende grenzen zijn
3. Beschrijf duidelijk waarbinnen je je product gaat opleveren
4. De planning

Nu je duidelijk hebt gemaakt wat je wilt gaan maken, probeer je het ontwerp en de realisatie ervan in te delen in onderdelen. Het project zal bestaan uit een aantal sprints. In elke sprint ga je een deel van de applicatie ontwerpen, realiseren, testen en evt verbetervoorstellen benoemen.

Een manier om de onderdelen van elkaar te scheiden is het formuleren van user story’s (functionaliteiten vanuit de user geformuleerd, wat kan de user doen)

Een totaal overzicht van alle user story’s kun je kwijt onder de product backlog.

1. 1. Product backlog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| product backlog |  |  |  |  |
| user story | eenheden | uitvoerder | prioriteit | taken |
| de gebruiker moet zich kunnen registreren om zich in te schrijven de lessen,  zodat ze kunnen inloggen |  | jay | ja | shield implementeren in mijn bestand voor de login |
| de gebruiker moet kunnen zien of er nog plek is om zich in te schrijven,  zodat hij zich kan inschrijven. |  | jay | ja | een query maken die zegt wanneer het vol is |
| als docent kan ik een maximum aantal opgeven,  zodat de lessen niet te vol worden |  | jay | ja | in de database een x aantal zetten |
| de gebruiker moet kunnen kiezen aan welke activiteiten ze zich willen inschrijven,  zodat ze een keuze hebben |  | jay | ja | alles in de view aangeven |
| de admin moet nieuwe producten in het systeem kunnen zetten,  zodat de leraren een ander product kunnen aanbieden |  | jay | ja | query maken zodat je via de site zelf een les kan toevoegen die naar de database gaat |
| de admin moet leraren kunnen toevoegen, zodat die kunnen inloggen |  | jay | ja | een tabel voor de leraren maken |
| de gebruiker moet een les kunnen annuleren, zodat die van de deelnemerslijst verdwijnt en een plek vrijkomt. |  | jay | ja | ?alleen gedeactiveerd niet uit database |
| als leraar kan ik een les aanmaken, zodat gebruikers zich kunnen inschrijven |  | jay | ja | query maken zodat je via de site zelf een les kan toevoegen die naar de database gaat |
|  |  |  |  |  |
|  |  |  |  |  |

* 1. Sprint backlog, burn down chart

Nu je een overzicht hebt van alle functionaliteit (user story’s) die je wilt realiseren, ga je per sprint een sprint backlog maken en een burn down chart.

Een sprint backlog bestaat uit een aantal user story’s of onderdelen (taken) van een grotere user story die jij tijdens een sprint wilt realiseren (ontwerp, bouw, test, verbeteren). De duur van alle user story’s van die sprint leveren het maximale aantal eenheden op voor de burn down chart. Houd bij je keuze voor de user story’s ook rekening met de prioriteiten, bijvoorbeeld met de MoSCoW-methode.

Wat is de logische volgorde?

Wat zijn de belangrijkste user story’s?

Tijdens de sprint hou je bij hoeveel eenheden je hebt afgemaakt. Die breng je op mindering op de burn down chart. Aan het eind van de sprint, als je alles af hebt gemaakt, is de burn down chart op de nullijn.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| sprint backlog |  |  |  |  |  |  |
| user story | eenheden gepland | meetmoment 1 | meetmoment 2 | uitvoerder | prioriteit | taken |
| gebruiker kan inloggen, zodat hij kan boeken | 8 | 4 | 4 | HK | H | inlog scherm, selectie, controle toegang, menubalk, landingspagina |
| gebruiker kan boeking wijzigen, zodat hij foutjes kan herstellen | 9 | 6 | 2 | HK | M | update scherm, query, zoekfunctie, check datum, controle toegang, redirect |
| score |  | 10 | 6 |  |  |  |
| doel | 17 | 17 | 7 |  |  |  |
| rest na meetmoment 1 | 8,5 | 7 | 1 |  |  |  |
| rest na meetmoment 2 | 0 |  |  |  |  |  |
|  |  |  |  |  |  |  |
| |  | | --- | |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

* 1. Reflectieverslag

Aan het eind van elke sprint maak je een overzicht van wat je goed vond gaan, wat je minder goed vond gaan, wat je hebt geleerd en wat je de volgende keer anders zou doen.

1. Het ontwerp
2. 1. Een ERD

Een ERD of entiteiten – relaties diagram is een geordend overzicht van de gegevens in de applicatie. De gegevens die nodig zijn bij het product zijn af te leiden uit de user story’s. Breng alle gegevens in beeld en order ze in een logisch samenhangend geheel van entiteiten. Gebruik eventueel normaliseren als techniek om de gegevens te ordenen.

Je geeft een entiteit weer met een soft box

Artikel

id

naam

prijs

Een entiteit heeft als naam een zelfstandig naamwoord en een aantal attributen (eigenschappen).

De naam van de entiteit is in het enkelvoud.

Tussen entiteiten bestaan vaak relaties. Het is niet verplicht dat alle entiteiten met relaties verbonden zijn.

De mogelijke relaties zijn

1. één op veel
2. één op één
3. Veel op veel

Zorg dat bij het uiteindelijk ontwerp geen veel op veel relaties meer voorkomen, maar los deze op met tussen-entiteiten.

De lijnen zijn on onderbroken als de verplicht is.

De lijn is onderbroken als de relatie optioneel is. Afbeelding met diagram

Automatisch gegenereerde beschrijving

* 1. Datadictionairy en relaties

1. **1.Een navigatie hoort bij 1 of meer boekingen**
2. **Een boeking bevat 1 en alleen 1 navigatie**

**Afbeelding met diagram

Automatisch gegenereerde beschrijving**

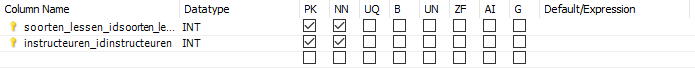


Afbeelding met tafel

Automatisch gegenereerde beschrijving

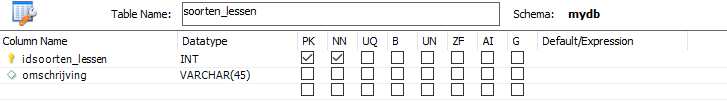
Afbeelding met diagram

Automatisch gegenereerde beschrijving



Afbeelding met diagram

Automatisch gegenereerde beschrijving



Een klassendiagram

Een klassendiagram heeft de klassen en latere objecten aan die in de applicatie aanwezig zijn. Het klassendiagram is in principe onafhankelijk van de ERD, maar kan overeenkomsten vertonen. Het klassendiagram geeft aan welke eigenschappen een object heeft en welke methoden. De eigenschappen en methoden zijn af te leiden uit de user story’s.

Lamp

* type: string

+ getType: string

+ setType(type: string): void

Artikel

# Id: integer

- naam: string

- prijs: decimal

- voorraad: integer

+ getNaam():string

+ getPrijs(): decimal

+ getVoorraad(): integer

+ setNaam(naam: string): void

+ setPrijs(prijs: decimal): void

+ setVoorraad(stuks:integer): void

1

n

Factuurregel

# artikel\_id: integer

# factuur\_id:integer

- aantal: integer

- geleverd: boolean

+ getAantal(): integer

+ setAantal( aantal:integer): void

+ getGeleverd():boolean

+ setGeleverd(geleverd: boolean): void

* 1. Wireframe

Een snelle schets met de indeling van de schermen

* 1. Navigatiestructuur

De toegang tot verschillende functionaliteiten vanuit een menu, of knoppen op schermen.

Afbeelding met diagram

Automatisch gegenereerde beschrijving

* 1. Mock ups

Vormgegeven schermen, waarbij de functionaliteit nog niet is gerealiseerd maar een beeld voor de klant wordt geschetst. De lay-out wordt geaccordeerd en daarvan wordt niet meer afgeweken, tenzij er een duidelijke reden vanuit de klant voor is.

1. Realisatie van eigen user story’s

Realiseer de eigen user story’s.

* **tijdens sprint**

1. 1. Programmeren
   2. Test scenario’s schrijven en uitvoeren
   3. Versiebeheer bijhouden, commits met git
   4. Documenteren (in de code of apart)
   5. Een database inrichten/aanpassen/aanvullen (indien nodig en afhankelijk van de situatie)
   6. Een codereview (onderling) op basis van de code conventies
   7. Voortgang in sprint doorspreken (stand up)
   8. Een planning bijhouden (werk de burn down chart bij)

* **na de sprint**
  1. Opleveren van sprint product
  2. Retrospective/ reflectieverslag
  3. Stel conclusies en verbetervoorstellen op voor je eigen user story’s.
  4. Review met klant
  5. Bundel de verbetervoorstellen van de review van de klant van jouw user story’s in een document.
  6. Leg de afspraken vast, neem ze mee met de volgende sprint
* **na afloop van de sprints**
  1. Voeg met het team de verschillende user story’s samen.
  2. Stel een scenario (inclusief testdata) op voor de gezamenlijke test.
  3. Voer een test uit voor alle user story’s uit de sprint die samen het product maken.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

1. Eindgesprek
2. 1. Presenteer het product
   2. Geef feedback op het proces
   3. Geef feedback op het product
   4. Toon aan wat je hebt gedaan met afspraken
   5. Reflecteer op je eigen functioneren
3. Resultaat

Als resultaat van deze opdracht lever je de volgende producten en/of diensten op.

1. 1. Ingevulde projectwijzer
   2. Applicatie en gegevensverzameling, via github/gitlab