**S2 Software  
Project verslag**

**Studentnaam: Mattijs Pronk**  
**Studentnummer: 456410**  
**Klas:**   
**Vakdocent:**  
  
**Versie: 1**  
**Datum:**

**Versiebeheer**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versienummer** | **Datum** | **Auteur** | **Veranderingen** |
| *0.1* | *08-02-2022* | *Mattijs Pronk* | *Opzet verslag.* |
| 0.2 | 11-02-2022 | Mattijs Pronk | Analyse. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Inhoudsopgave

Inhoudsopgave 3

1 Inleiding 4

1.1 Aanleiding 4

1.2 Onderwerp 4

1.3 Leeswijzer 4

2 Introductie 5

3 Aantonen leerdoelen 6

3.1 Sprint 1: Proof of concepts 6

3.2 Sprint 2: Tussenproduct 6

3.3 Sprint3: Eindproduct 6

4 Reflectie / evaluatie 7

4.1 Waar ben ik trots op? 7

4.2 Wat doe ik een volgende keer anders? 7

4.3 Welke formatieve indicatie zou ik mezelf geven voor de verdieping Software? 7

* Inleiding
* Aanleiding

*Dit verslag wordt geschreven om de leraar software te overtuigen dat de aangeleerde stof goed terecht is gekomen en in praktijk is toegepast.*

* Onderwerp

*Dit verslag gaat over het eigen project van semester 2 met de nadruk op software.*

* Introductie

Dit project gaat over een applicatie die bedoeld is voor een fitness bedrijf, zowel voor het bedrijf zelf als voor de klant.

Een duidelijk overzicht in de sportschool is een must, door gebruik te maken van een planning wordt er meer voortgang geboekt. Zo is het mentaal voor een sporter veel makkelijker om toch nog even te gaan sporten. Ook is het fysiek voor de sporter belangrijk om te weten welke spiergroepen die dag getrained zijn of juist getrained moeten worden.

Voor het bedrijf is een planning makkelijk om te weten hoeveel mensen er hoelaat aanwezig zijn, vooral met de huidige corona maatregelen is dit erg handig. Ook komt er door deze applicatie automatisch een limiet op het aantal sporters per uur in de sportschool.

De planning houd in dat er gereserveerd kan worden voor een bepaald tijdstip en welke apparaten de klant wilt gebruiken tijdens zijn gereserveerde tijd.

* Aantonen leerdoelen
* Sprint 1: Analyse

**Requirements:**

Functionele requirement: de klant moet kunnen inloggen op zijn account.  
Beperkingen: de klant moet inloggegevens hebben.  
Kwaliteitseisen: bij foute inlog poging geef error “inloggen mislukt”.

Functionele requirement: de klant moet een tijdstip kunnen reserveren.  
Beperkingen: de klant kan maar een keer per uur reserveren.  
Kwaliteitseisen: bij twee keer op hetzelfde uur reserveren geef error “al gerserveerd”.

Functionele requirement: de klant moet een aantal machine’s kunnen reserveren.  
Beperkingen: de klant kan machine niet twee keer achter elkaar reserveren.  
Kwaliteitseisen: reserveringstijd per machine bijvoorbeeld 2-15 minuten.

Functionele requirement: de klant moet zijn reservering kunnen cancelen.  
Beperkingen:   
Kwaliteitseisen:

Functionele requirement:   
Beperkingen:  
Kwaliteitseisen:

**Use case:**

Samenvatting: Na het reserveren van tijdstip en machine’s moet de klant dit kunnen gebruiken. 

Actors: Klant 

Aannamen: Klant moet ingelogd zijn om te kunnen reserveren. 

Senario: 1. Inloggen in het systeem.

2. Tijd reserveren.

3. Machine’s reserveren. 

Uitzonderingen: klant heeft geen profiel, maak profiel aan. 

Resultaat: Klant reserveerd zijn tijd en machine’s hierna kan de klant rustig gaan sporten.

**Wireframe:**



**Website/Forms layout:**

Logo:



Login scherm:



Reservering scherm:

* Sprint 2: Tussenproduct

*<Hier beschrijf je de door jou geleverde bijdrage aan het tussenproduct met daarin de volgende onderdelen.*

* *Een beschrijving van de functionaliteit van het tussenproduct, eventueel met relevante code snippets.*
* *Een beschrijving van je ontwerp (eventueel screenshot UML classdiagram)*
* *De gebruikte bronnen*>
* Sprint3: Eindproduct

*<Hier beschrijf je de door jou geleverde bijdrage aan het eindproduct met daarin de volgende onderdelen:*

* *Een beschrijving van de functionaliteit van het eindproduct, eventueel met relevante code snippets.*
* *Het uiteindelijke class diagram.*
* *Eventueel testresultaat*
* *De gebruikte bronnen*>

* Reflectie / evaluatie
* Waar ben ik trots op?

*<vul in>*

* Wat doe ik een volgende keer anders?

*<vul in>*

* Welke formatieve indicatie zou ik mezelf geven voor de verdieping Software?

|  |  |  |
| --- | --- | --- |
| Leeruitkomst verdieping Software (bouwt voort op de oriëntatie) | | |
| Onderdeel | Criterium | Rating |
| Aandacht voor algoritmiek | Je kunt (eenvoudige) object-georiënteerde applicaties schrijven die stapsgewijs oplossingen voor problemen vinden door het uitvoeren van logische testen en eenvoudige stapsgewijze berekeningen | *<vul in U S G O met motivatie>* |
| Basisvaardigheden | Je begrijpt en past de volgende programmeerconcepten toe:objects/ classes, constructors, private fields/ encapsulation, get/ set-methodsen/ofproperties, method/ constructor overloading, class diagrams/ relations/ multiplicity. - De focus hierbij is op leesbare (b.v. naamgeving, indentation) en onderhoudbare software programma’s. Je ontwerpt een class diagram (met uitleg) waarin de belangrijkste functionaliteit terug te vinden is. | *<vul in U S G O met motivatie>* |
| Aantonen | Je vraagt feedback van een docent met software engineeringskennis en laat zien dat je deze feedback verwerkt hebt. | *<vul in U S G O met motivatie>* |