Projectplan

Second\_Factor

Eindhoven

|  |
| --- |
| **Opdrachtgever : Fontys** |
| **Projectleden: Harm van Veen, Joco Bogdanovic, Rory Lynch, Ronald van den Burg, Yorick Laros, Lloyd van Zaalen** |
| **Datum :** |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versie | Datum | Auteur(s) | Wijzingen | Status |
| 0.1 | 10-09-2020 | Groep | Initiële versie | Klaar |
| 0.2 | 10-09-2020 | Yorick Laros | Bestand opgeschoond | Klaar |
| 0.3 | 12-09-2020 | Harm van Veen | Logo toegevoegd en nummering aanpassen | Klaar |
| 0.4 | 14-09-2020 | Groep | Milestone 1 ingevuld en concept gemaakt voor milestone 2. | Klaar |
| 0.5 | 18-09-2020 | Yorick Laros | Document overgezet naar een nieuw Word document. Oude originele verplaatste naar Archief in MS Teams Map | Klaar |
| 0.6 | 18-09 | Yorick Laros | Inhoudsopgave aangepast en Style document. Mijzelf bij teamleden toegevoegd en een stukje geschreven bij communicatie | Klaar |
| 0.7 | 19-09 | Harm van Veen | Eindproduct diagram en 1ste milestone weergave ingevoegd. Mijzelf bij teamleden toegevoegd. Daarnaast heb ik een stukje besluitvorming geschreven. | Klaar |
| 0.8 |  | Yorick Laros | Subhoofdstuk Github regels toegevoegd |  |
| 0.9 | 29-11-2020 | Ronald v.d. Burg | Een aantal kleine aanpassingen gemaakt. |  |

Inhoud

[Projectopdracht 4](#_Toc54190993)

[Doel van het project 4](#_Toc54190994)

[Begrenzing 5](#_Toc54190995)

[Strategie 5](#_Toc54190996)

[Afspraken versiebeheer 5](#_Toc54190997)

[MoSCoW 6](#_Toc54190998)

[Teamleden 6](#_Toc54190999)

[Communicatie 6](#_Toc54191000)

[Besluitvorming 6](#_Toc54191001)

[GitHub controle regels 7](#_Toc54191002)

[Ontwerp 8](#_Toc54191003)

[User case diagram 8](#_Toc54191004)

[Eindproduct 8](#_Toc54191005)

[Milestone 1: Structuur van de communicatie opzetten 9](#_Toc54191006)

[Functional requirements 9](#_Toc54191007)

[Non-functional requirements 9](#_Toc54191008)

[Milestone 2: De juiste (kleuren)communicatie wordt verzonden 11](#_Toc54191009)

[Functional requirements 11](#_Toc54191010)

[Non-functional requirements 11](#_Toc54191011)

[Milestone 3: beheerpanel opzetten 12](#_Toc54191012)

[Milestone 4: ? 12](#_Toc54191013)

# Projectopdracht

## Doel van het project

**Probleemstelling**

In deze hectische, digitale wereld kun je je soms zorgen maken om je veiligheid. Is je wachtwoord wel complex genoeg? En wat als iemand je wachtwoord heeft ontdekt? Om die reden is een tweede laag van authenticatie vandaag de dag zo belangrijk!

**Oplossing**

De meeste authenticatie methodes werken met willekeurige letters of cijfers die je toegang verschaffen tot jouw accounts. Maar wat als je last hebt van dyslexie? Wat als die willekeurige codes elke keer voor je ogen beginnen te zwemmen, waardoor je onnodig vaak fouten maakt? Dan zou een alternatief zonder letters en cijfers een perfecte uitkomst zijn!

Dat is precies wat we aan willen bieden bij Second\_Factor. Een veilige two-factor authenticatie die breed inzetbaar en makkelijk in gebruik is.

**Concept**

Wij bieden een simpele, fysieke ‘sleutel’ die op basis van kleurencodes wordt bestuurd. Zodra je de website benadert, krijg je een reeks kleuren die je op je ‘sleutel’ kan invullen. De door jou ingegeven kleurencode wordt vervolgens geverifieerd en als de kleurencode juist is ingegeven krijg je toegang tot de website.

Snel, simpel, veilig.

## Begrenzing

|  |  |
| --- | --- |
| Tot het project behoort: | Tot het project behoort niet: |
| Werkend proof-of-concept | Onderhoud na oplevering |
| Correcte architectuur documentatie | Security updates |

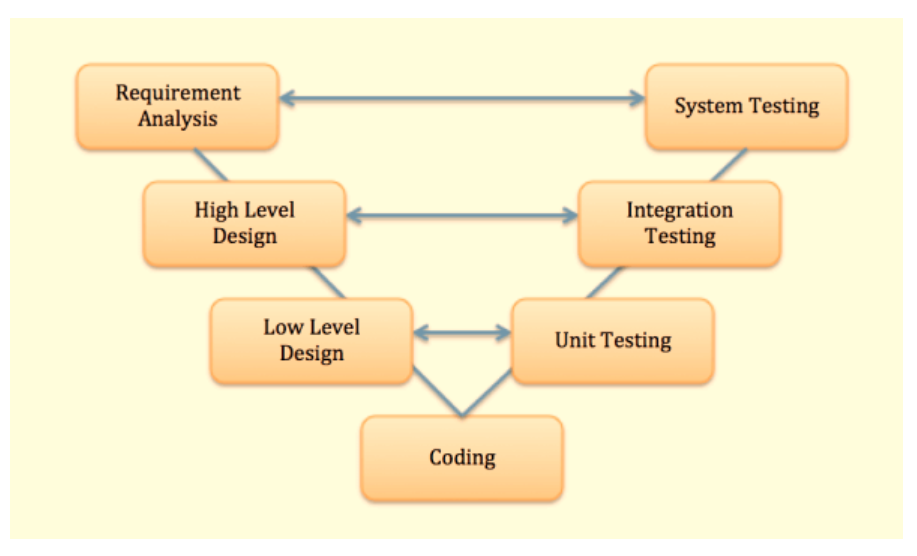
## Strategie

**//milestone beschrijving hoelang het duurt zolang die duurt en gaan pas over op de volgende als de huidige af is**

**Inleiding:**

Voor ons project gaan we per milestone werken. In elke milestone gaan we het V-model toepassen. We beginnen pas aan de volgende milestone als de voorgaande goed is afgerond. Het V-model is een veel toegepaste methode in de developer wereld. Het V-model is een verbeterde Waterval. Het voordeel van een V-model ten opzichte van het Waterval model is dat je per laag de benodigde stappen doorneemt. Zie onderstaande afbeelding.

## Afspraken versiebeheer



### MoSCoW

Must:

* Arduino heeft vijf knoppen (Rood, Geel, Blauw, Verzend, Reset)
* De Arduino kan de input van een corresponderende kleurenknop laten zien op een output RGB-led.
* Er is een inlogpagina op de frontend waar de gebruiker in kan loggen.
* Na het inloggen via de website krijgt de gebruiker een willekeurige kleurencode die ingevuld moet worden op de Arduino.
* De Arduino moet de door de gebruiker ingegeven kleurencode naar de backend kunnen sturen ter verificatie.

Should:

Could:

Inloggen kan gebeuren middels een NFC/RFID tag

Arduino laat middels een geluidsignaal weten dat iets is ingedrukt

Won’t:

## Teamleden

|  |  |
| --- | --- |
| Naam:  Email:  Mobiel: | Yorick Laros  [420471@student.fontys.nl](mailto:420471@student.fontys.nl)  0642407256 |
| Naam:  Email:  Mobiel: | Harm van Veen  [314527@student.fontys.nl](mailto:314527@student.fontys.nl)  0628783128 |
| Naam:  Email:  Mobiel: | Rory Lynch  [163178@student.fontys.nl](mailto:163178@student.fontys.nl)  0644614095 |
| Naam:  Email:  Mobiel: | Joco Bogdanovic  [214934@student.fontys.nl](mailto:214934@student.fontys.nl)  0623502312 |
| Naam:  Email:  Mobiel: | Lloyd van Zaalen  [l.vanzaalen@student.fontys.nl](mailto:l.vanzaalen@student.fontys.nl)  +316 53757106 |
| Naam:  Email:  Mobiel: | Ronald van den Burg  [403145@student.fontys.nl](mailto:403145@student.fontys.nl)  +31651304147 |

## Communicatie

Voor de korte communicatie gebruiken we Whatsapp. Met korte communicatie bedoelen we bijvoorbeeld dat je een korte vraag hebt of dat je wil doorgeven dat iets af is.

Voor online vergaderingen maken we gebruik van Microsoft Teams. We gebruiken Teams ook voor opslag van onze project documentatie. Als versiebeheer voor onze code gebruiken we Git en GitHub.

## Besluitvorming

Onze besluitvorming zal democratisch zijn. Mocht het geval zich voordoen dat we ergens niet uitkomen als groep, dan zal er een gesprek met de docent ingepland worden.

## GitHub controle regels

Zoals we in het subhoofdstuk Communicatie beschreven, hebben we afgesproken om GitHub voor versiebeheer te gebruiken. Er zijn regels opgezet om de kwaliteit en de efficiëntie van GitHub en het product te waarborgen.

De volgende regels zijn van toepassing:

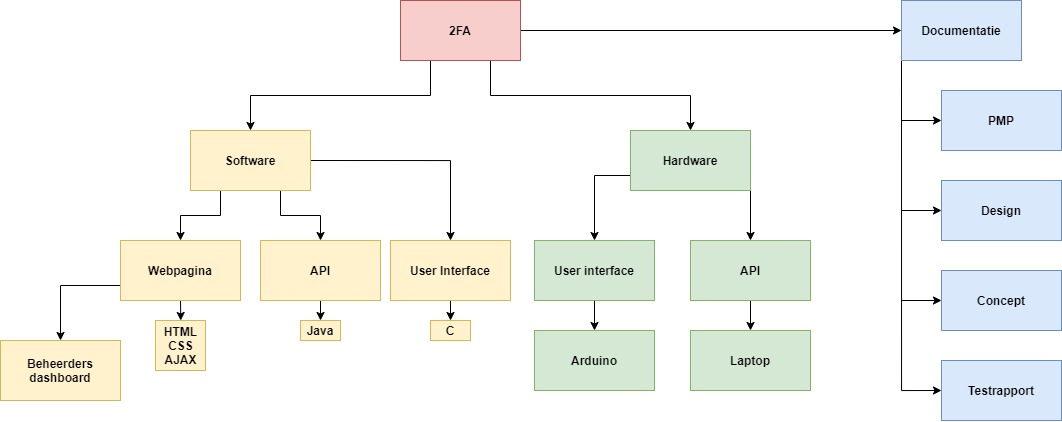
1. Er wordt niet zonder de minimale goedkeuring code naar de master gepushed. Als mogelijk dient ieder teamlid hier zijn goedkeuring over uit te spreken via een comment.
2. Nadat een teamlid zijn werk heeft gepushed, dient deze persoon een pull-request aan te maken.
3. Merge naar de master mag pas na goedkeuring van minimaal twee teamleden.

# Ontwerp

## User case diagram



## Eindproduct

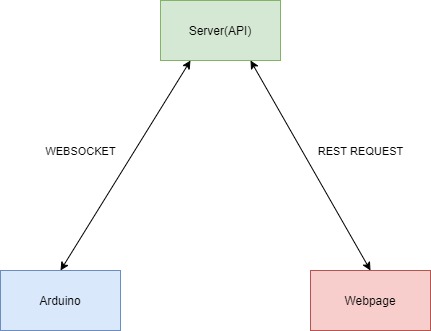


# Milestone 1: Structuur van de communicatie opzetten

**Inleiding**

Voor de eerste milestone willen we ervoor zorgen dat we de communicatie tussen de modules werkend krijgen. Hierbij willen we de modules met elkaar laten communiceren. In deze milestone houden we ons nog niet bezig met de inhoud van die communicatie.

De communicatie tussen de Server module en de Arduino module wordt stateful. De communicatie tussen de Website module en de Server module wordt stateless. Het eindproduct van milestone 1 is dat we in de frontend “Hello?” in kunnen voeren en dat als de Arduino dit signaal via de backend ontvangt, hier “Hello World!” van kan maken en via de backend terug kan sturen naar de frontend. Pas als dit is gelukt, gaan we naar milestone 2.



## Functional requirements

We hebben hierbij de volgende functional requirements: 

1. Via de website kan een HTTP request gestuurd worden naar het endpoint van de API die op de Server module draait.
2. De website kan de response van de API weergeven.
3. De API kan een websocket openen
4. De API stuurt een handshake request naar de Arduino.
5. De Arduino kan verbinden met de websocket.
6. De Arduino bevestigt de handshake met de API.
7. Zowel Arduino als website geven de status van overall connectie weer.

## 

## Non-functional requirements

En de volgende non-functional requirements

1. De Arduino laat een led branden om het moment dat er verbinding is.
2. De Server laat op command-line zien dat de websocket is geopend en verbonden is.
3. De Website laat een bericht zien als er een response is gestuurd vanuit de API.

# Milestone 2: De juiste (kleuren)communicatie wordt verzonden

**Inleiding**

Voor de tweede milestone willen we communicatie passend maken. Omdat we in milestone 1 al voor een werkende communicatie-lijn door heel onze applicatie hebben gezorgd, kunnen we hier invulling gaan geven aan die communicatie.

In deze milestone willen we dat onze frontend (de website) een gebruikersnaam en wachtwoord kan accepteren als input, deze gegevens kan versturen naar de backend (Server), om deze gegevens vervolgens te verifiëren met de gegevens in de backend. Als deze gegevens overeenkomen met de gegevens in de backend, wordt er in de backend een kleurencode willekeurig gegenereerd en getoond op de website.

Deze kleurencode kan door de gebruiker worden ingevoerd op de Arduino. De Arduino zal deze code weer doorsturen naar de Server ter verificatie. Als de door de gebruiker ingevoerde kleurencode overeenkomt met de door de backend gegenereerde kleurencode, zal de gebruiker toegang krijgen tot de website.

## Functional requirements

We hebben hierbij de volgende functional requirements:

# Frontend (Website - inlogpagina)

# Frontend (Website – gebruikerpagina)

### Backend (Server)

### Arduino

* Arduino maakt en behoudt een verbinding met de backend
* Zolang er connectie is met de backend zal er een led lampje branden.
* De gebruiker kan doormiddel van 3 knoppen een kleurencode opstellen in de Arduino
* De gebruiker kan de ingevoerde kleurencode versturen met de ‘Verzend’ knop.
* De gebruiker kan de ingevoerde kleurencode wissen met de ‘Reset’ knop.

## Non-functional requirements

En de volgende non-functional requirements:

1. Het systeem moet de gebruiker de door hem/haar ingegeven kleurencombinatie tonen middels LED-lampjes voordat de gebruiker deze bevestigt.
2. Bij het foutief ingeven van de gebruikersnaam/wachtwoord dient het systeem een foutmelding te tonen op het inlogscherm.
3. Bij het niet ingeven van de gebruikersnaam/wachtwoord dient het systeem een foutmelding te tonen op het inlogscherm.
4. Als er binnen de 2 stappen van de inlog-procedure gegevens foutief worden ingevoerd, dient het systeem de gebruiker opnieuw een kans te geven met een nieuwe inlog-procedure.
5. De kleurencombinatie die voor elke inlog-procedure wordt gegenereerd moet uniek en willekeurig zijn.
6. De gebruiker moet maximaal 8 handelingen verrichten om de inlog-procedure succesvol af te kunnen handelen.
7. Voordat de gebruiker inlogt, dient de hardware aangeschakeld te zijn.
8. De totale laadtijd van de inlogpagina dient maximaal 2 seconden te zijn.
9. Bij een X aantal verkeerde inlogpogingen vindt er een time-out plaats van X aantal seconden (zowel voor het wachtwoord als voor de kleurencode)?
10. Een kleurencode is alleen 30 seconden valide, hierna wordt de kleurcombo ververst/opnieuw gegenereerd?
11. Authenticatie van wachtwoord vindt via de server plaats en mag niet langer dan X seconden duren.

# Milestone 3: beheerpanel opzetten

# Milestone 4: ?