Testplan

Semester 3

Doomen,Jens J.J.W.

2021

Inhoudsopgave

[Unit testen 2](#_Toc92793277)

[End-to-end testing 2](#_Toc92793278)

[SonarCloud 2](#_Toc92793279)

[Non functional test 2](#_Toc92793280)

[Performance test 2](#_Toc92793281)

[Test strategie 3](#_Toc92793282)

[Risk based testen 4](#_Toc92793283)

[Ideale pipeline 5](#_Toc92793284)

[Bijlage 6](#_Toc92793285)

# Unit testen

Unit testen gaan over de back-end ze gaan vooral over de authenticatie, dit betekent dat er getest wordt over de controller en model van de User en het creëren van een webtoken. Deze unit testen zijn een apart project en kunnen daarom in mijn CI gebouwd worden. Dit gaat alleen als de testen werken. Dat betekent dat als deze testen niet werken dat mijn CI ook niet werkt, om deze reden is het erg belangrijk dat deze soort testen werken en slagen.

# End-to-end testing

Mijn end-to-end testen zijn geschreven in cypress. Ik test hier ook wat dingen op de front-end om te kijken of deze ook werken. Ze gaan vooral over het inloggen, registreren en het versturen van een bericht. Wat ik nog zou kunnen doen is er voor zorgen dat deze test ook kijken naar het aanpassen van het account, denk hierbij aan het aanpassen van de naam of het email van de gebruiker. Omdat deze testen zijn geschreven in cypress is het erg lastig om deze aan mijn CI toe te voegen. Het runt namelijk op een browser en kan ik vanaf daar alle testen bekijken.

# SonarCloud

Wanneer is mijn project acceptabel om online te zetten? Kan ik zien of het beter is dan de vorige push? Kunnen mijn gebruikers meteen aan de slag? Dit zijn de vragen die met acceptatie testen beantwoord kunnen worden. Om deze vragen te kunnen beantwoorden en snel te kunnen kijken of de laatste push acceptabel is. Om dit te doen heb ik sonarqube, sonarqube kijkt naar je pushes en zet dan een code review op je push. Op basis van de aangegeven bugs kan ik bepalen of het product acceptabel is. Ook is het erg belangrijk om te kijken of er bugs op smells in je applicatie zitten. Daarom is het eigenlijk best belangrijk op dit zo vroeg mogelijk te implementeren want als bugs blijven zittten dan wordt het steeds lastiger om ze op te lossen.

In mijn ogen is het project geslaagd als er niet meer dan 2 major bugs zijn en 4 minor.

# Non functional test

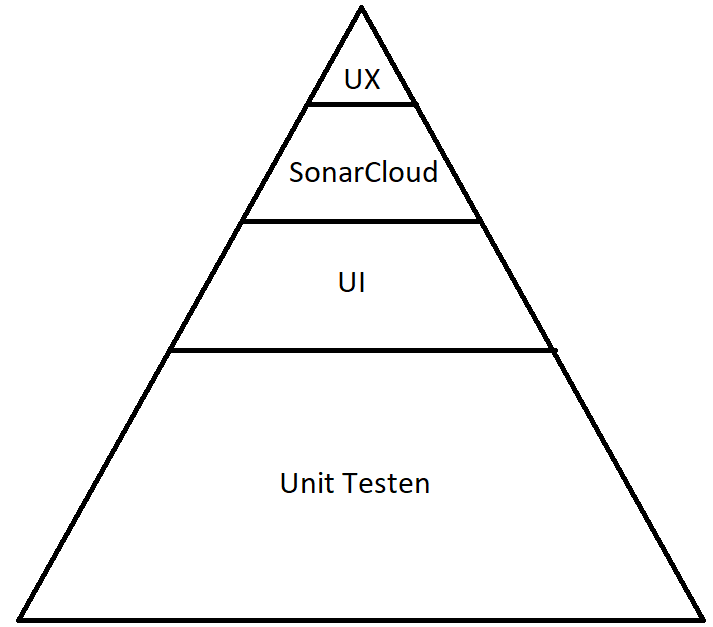
Voor de non-funtional system tests uit te voeren wil ik me gaan focussen op een stress test. Dit wil ik gaan uitvoeren op de connectie tussen de clients en de server om te kijken of de websocket server het aan kan om meerdere requests af te handelen in een korte tijd. Natuurlijk moet de server ook al deze berichten weer doorsturen. Aan het einde van de stress test hoop ik natuurlijk alle berichten die ik heb verstuurd weer terug te kunnen lezen op de andere client en natuurlijk vanaf de andere kant ook. De stress test kan ik helaas ook niet aan mijn CI toevoegen.

# Performance test

Tot slot wil ik nog een performance test gaan uitvoeren. Om dit aan te tonen wil ik een lighthouse test doen op mijn applicatie. Dit is een snelle en simpele vorm om de performance aan te geven. Dit heb ik zo gedaan omdat er niet heel veel tijd meer over is in dit semester. Deze vorm van testen wordt gedaan in de browser zelf en kan ik daarom ook niet aan mijn ci pipeline toevoegen

# Test strategie

We naderen het einde van het semester er moet nog wat gebeuren aan het project, ook moet er nog wat getest worden. Maar welke test ga ik wel doen en welke wat minder. Mijn hoogste prioriteit ligt bij het maken van de unit testen, hierna komen de cypress testen en als laatste de code analysis. Ook ga ik kijken naar welke onderdelen ik ga testen. Voor nu zijn er testen op het inloggen en registreren, dit is gedaan omdat dit moet. Als dit niet werkt heb je niks aan de applicatie. Hetzelfde gaat dus ook gelden voor het sturen van berichten, als deze functie faalt dan is er niks wat je met de app kan doen. Ook moet er natuurlijk naar de UX worden gekeken. Dit wil ik gaan doen door iemand door mijn applicatie te laten lopen en feedback te geven. Dat betekent wel dat dit niet automatisch kan.



# Risk based testen

Om te beginnen ga ik mijn use cases analyseren en bekijken hoe hoog de risk is per user story.

1. Als Gebruiker, wil ik accountgegevens invullen, zodat ik toegang krijg om een account aan te kunnen maken. **M**
2. Als Gebruiker, wil ik een account kunnen aanmaken, zodat ik de functies van de app kan gebruiken. **M**
3. Als Gebruiker, wil ik mijn account gegevens kunnen bevestigen , zodat ik kan controleren of ze goed zijn ingevuld. **M**
4. Als Gebruiker, wil ik mijn account kunnen aanpassen , zodat ik foute gegevens kan verbeteren. **H**
5. Als Gebruiker, wil ik mijn account kunnen verwijderen, zodat ik geen account meer heb als ik geen behoefte meer heb aan de services die de app bied. **H**
6. Als Gebruiker, wil ik gegevens kunnen invullen , zodat ik kan inloggen. **M**
7. Als Gebruiker, wil ik berichten kunnen ontvangen , zodat ik kan zien wat een ander persoon heeft getypt. **H**
8. Als Gebruiker, wil ik berichten verzenden , zodat ik een reactie kan typen naar iemand. **H**
9. Als Gebruiker, wil ik berichten kunnen verwijderen , zodat een bericht wat ik ongedaan wil maken kan weghalen. **H**
10. Als Gebruiker, wil ik een bericht kunnen bewerken, zodat ik spellingsfouten er uit kan halen. **H**
11. Als Gebruiker, wil ik een bericht kunnen vastzetten , zodat ik belangrijke informatie kan bijhouden. **M**
12. Als Gebruiker, wil ik een bestand kunnen versturen , zodat ik bestanden kan delen met de mensen die het nodig hebben. **H**
13. Als Gebruiker, wil ik een groep kunnen aanmaken , zodat ik een bericht naar meerder mensen kan sturen. **M**
14. Als Gebruiker, wil ik leden kunnen toevoegen aan een groep , zodat er meer mensen bij kunnen. **M**
15. Als Gebruiker, wil ik leden kunnen verwijderen, zodat er minder mensen het bericht tegelijk kunnen zien. **M**
16. Als Gebruiker, wil ik een groep chat kunnen muten , zodat ik niet altijd een melding krijg bij een nieuw bericht. **M**
17. Als Gebruiker, wil ik een vriend kunnen zoeken, zodat ik met meer mensen kan chatten. **M**
18. Als Gebruiker, wil ik een vriend kunnen toevoegen , zodat hij in een lijstje komt te staan en chatten makkelijker wordt. **M**
19. Als Gebruiker, wil ik een vriend kunnen verwijderen , zodat hij weer uit dit lijstje gaat. **M**
20. Als Gebruiker, wil ik een vriend kunnen muten, zodat ik niet altijd een melding krijg van een nieuw bericht. **M**
21. Als Admin, wil ik iemand een waarschuwing kunnen geven , zodat ik iemand die zich misdraagt kan benaderen. **L**
22. Als Admin, wil ik een account inactief kunnen zetten, zodat lang niet gebruikte accounts weg kunnen. **L**
23. Als Admin, wil ik een account kunnen bewerken , zodat gegevens die ongepast zijn weg kunnen worden gehaald. **L**

**H** = High risk

**M** = Medium risk

**L** = Low Risk

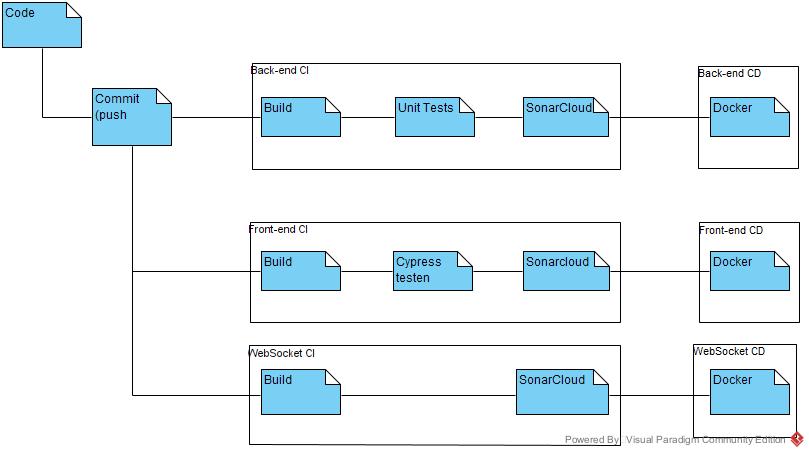
De hoge risico’s moeten als eerste getest wordt en moeten de meeste testen hebben. Op deze manier kan ik ervoor zorgen dat de hoge risico’s minder kans hebben op falen en bij het toevoegen van een nieuwe feature de applicatie niet breekt.



# Ideale pipeline

Al de testen die ik in mijn ci pipeline kan hebben wil ik er natuurlijk ook gewoon inhebben. Dit betekent dat bij het pushen naar git en de pipeline in zijn werk gaat het project gebouwd wordt met de unit testen erbij. En voor zowel mijn front-end als back-end repo wil ik in de pipeline dat er een scan van sonarcloud wordt gedaan. Ook is het handig om voor mijn websocket server een repo te hebben

Dit is alles wat ik wil hebben in mijn pipeline maar is dat ook haalbaar?

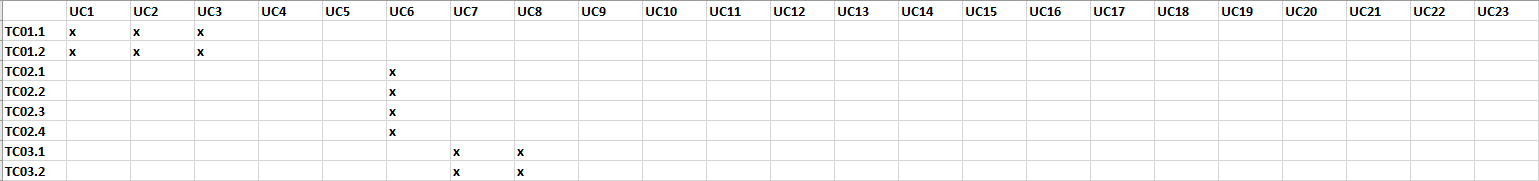


Heb ik het ook gehaald om deze pipeline te realiseren? Helaas heb ik niet alles in mijn pipeline kunnen zetten. Mijn back-end CI en CD zijn gerealiseerd en worden dus uitgevoerd. Mijn front-end CI heeft geen cypress testen erin zitten. Deze heb ik wel gemaakt maar ze draaien niet met mijn CI. Bij websockets heb ik alleen het docker gerealiseerd, dit betekent dat het bouwen en het scannen met sonarcloud niet is gemaakt in mijn CI.

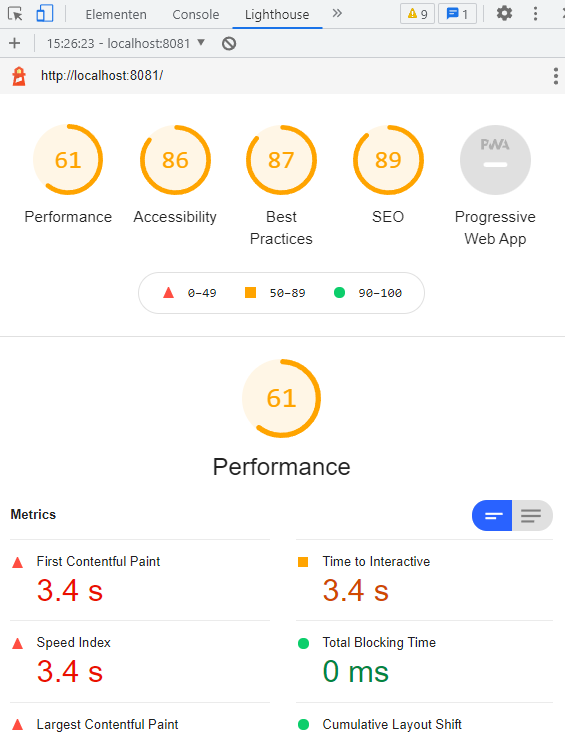
# Bijlage

In de bijlage zitten de resultaten van de testen die zijn uitgevoerd.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **User Story** | **Verwachte uitvoer** | **Geobserveerde resultaat** | **Resultaat van testcase** |
| TC01 | 6 | De back-end geeft een token terug en logt in | De back-end geeft een token terug en logt in | Geslaagd |
| TC02 | 6 | De back-end logt in | De back-end logt in | Geslaagd |
| TC03 | 2 | De back-end registreert een gebruiker | De back-end registreert een gebruiker | Geslaagd |
| TC04 | 6 | De back-end haalt een User op | De back-end haalt een User op | Geslaagd |
| TC05 | 6 | De back-end maakt een token aan | De back-end maakt een token aan | Geslaagd |
| TC06 | 6 | De back-end leest een token uit | De back-end leest een token uit | Geslaagd |
| TC07 | 6 | De back-end creëert een nieuwe token voor een verlopen token | De back-end creëert een nieuwe token voor een verlopen token | Geslaagd |
| TC08 | 6 | De back-end maakt een token aan | De back-end maakt een token aan | Geslaagd |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case** | **User Story** | **Invoer** | **Verwachte uitvoer** | **Geobserveerde resultaat** | **Resultaat van testcase** |
| TC01.1 | 1,2,3 | Cypress navigeert naar de registreer pagina. | Het systeem weergeeft de registreren pagina met de invoervelden:  Naam  Email  Wachtwoord. | De juiste pagina komt op het scherm | Geslaagd |
| TC01.2 | 1,2,3 | Cypress vult de juiste gegevens in. | Het systeem registreert de nieuwe gebruiker. | De gebruiker komt bij de database te staan | Geslaagd |
| TC02.1 | 6 | Cypress navigeert naar de inlog pagina. | Het systeem weergeeft de inlog pagina met de invoervelden:  Email  Wachtwoord. | De juiste pagina komt op het scherm | Geslaagd |
| TC02.2 | 6 | Cypress vult de juiste gegevens in. | Het systeem logt de gebruiker in en gaat naar de volgende pagina. | De gebruiker wordt ingelogd | Geslaagd |
| TC02.3 | 6 | Cypress vult onjuiste gegevens in. | Het systeem logt de gebruiker niet in en blijft op deze pagina. | De gebruiker wordt toch ingelogd | Niet geslaagd |
| TC02.4 | 6 | Cypress vult geen gegevens in. | Het systeem geeft aan dat er nog wat ingevuld moet worden en gaat niet door naar de volgende pagina. | Het systeem geeft aan dat er nog wat ingevuld moet worden en gaat niet door naar de volgende pagina. | Geslaagd |
| TC03.1 | 7,8 | Cypress typt een bericht en drukt op send. | Het systeem laat de naam van de gebruiker en het bericht zien. | Het bericht wordt niet weergegeven | Niet geslaagd |
| TC03.2 | 7,8 | Een 2e client logt in | Het systeem laat het bericht zien. |  | Niet getest |



Afbeelding met tekst

Automatisch gegenereerde beschrijving

Er zijn nog wat errors te zien bij mijn testen. Er is helaas weinig tijd over om de testen te fixen. De app doet het en dat betekent dat het bij de testen fout gaat, ik zie ook waar het fout gaat en waarom. Helaas weet ik niet hoe ik het fix, ik hoop het nog voor elkaar te krijgen voordat het semester is afgelopen.