Kamers Reserveren

Afstudeerverslag

Auteur:	**Mike van Es**
Studentnummer:	S1111981
Opleiding:	Informatica
Onderwijsinstelling	Hogeschool Leiden
Datum:	08-08-2022
Plaats:	Capelle aan de IJssel
Bedrijf:	Alten Nederland
Business manager:	A. Schutter
Stagebegeleider:	L. Stam

Colofon

Titel	Kamers Reserveren
Auteur	Mike van Es
Studentnummer	S1111981
Datum	28-08-2017
Plaats	Capelle aan de Ijssel
Bedrijf	Alten Nederland
Afdeling	Technical Software
Stagebegeleider	L. Stam
Business manager	A. Schuter
Onderwijsinstelling	Hogeschool Leiden
Domein	Techniek
Opleiding	Informatica
Schoolbegeleider	E. Verduin
Versie	3.0
Status	Definitief

Documenthistorie

Datum	Versie	Type	Beschrijving
04-07-2022	0.1	Ontwikkelversie	Opzet en voorwoord
10-08-2022	0.2	Ontwikkelversie	Opzet afronden
12-08-2022	0.3	Ontwikkelversie	Inleiding
18-08-2022	0.4	Ontwikkelversie	Organisatie
19-08-2022	0.5	Ontwikkelversie	Achtergrond en opdrachtomschrijving
22-08-2022	0.6	Ontwikkelversie	Aanpak en uitvoering
23-08-2022	0.7	Ontwikkelversie	Uitvoering
24-08-2022	0.8	Ontwikkelversie	Resultaten en competenties
30-08-2022	0.9	Ontwikkelversie	Feedback Kevin van Roemburg verwerkt
08-09-2022	0.9	Ontwikkelversie	Feedback Leon Stam verwerkt
09-09-2022	0.10	Ontwikkelversie	Onderzoek
15-09-2022	0.11	Ontwikkelversie	Batterijtest, conclusie en aanbevelingen
19-09-2022	0.12	Ontwikkelversie	Feedback Leon Stam verwerkt
20-09-2022	0.13	Ontwikkelversie	Batterijtest bijgewerkt en reflectie
29-09-2022	0.14	Ontwikkelversie	Nawoord Leon Stam toegevoegd
01-10-2022	0.15	Ontwikkelversie	Feedback Evert Verduin verwerkt
10-10-2022	3.0	Definitief	Feedback Leon Stam verwerkt

Voorwoord

Voor u ligt de scriptie “Kamers reserveren” een project dat is uitgevoerd door Mike van Es. Deze scriptie is geschreven in het kader van afstuderen aan de opleiding Informatica aan de Hogeschool van leiden van Faculteit Science and Technology. Het project is uitgevoerd van april 2022 tot en met september 2022 bij Alten Nederland.

Bij deze wil ik graag mijn begeleider Leon Stam hartelijk danken voor zijn begeleiding en ondersteuning tijdens de werkzaamheden. Ook wil ik de rest van mijn collega’s bedanken voor de gastvrijheid, steun en hulp tijdens mijn stageperiode.

Verder wil ik mijn begeleider van school, Evert Verduin bedanken voor zijn ondersteuning gedurende het project.

Tot slot wil ik mijn directe thuisomgeving bedanken. Voor de morele ondersteuning tijdens het afstudeertraject.

Ik wens u veel leesplezier toe,

Mike van Es
Leiden, 05-08-2022

Managementsamenvatting

In april 2022 begon het afstudeertraject bij Alten. Alten is een groot consultancy bedrijf met maar liefst 37.000 werknemers. Waarbij ze actief zijn in 25 landen. Iedere vestiging van Alten heeft zijn eigen expertise. De werkzaamheden tijdens het afstudeertraject zijn gepleegd op de vestiging in capelle a/d ijssel, deze vestiging focust zich voornamelijk op (technisch) softwareontwikkeling.

Binnen Alten hangt een informele open sfeer. De werkzaamheden worden op grote kantoorpleinen verricht. Hierdoor is het mogelijk om op ieder moment een praatje te maken met een medeconsultant en eventueel vragen te stellen als je ergens tegenaan loopt. Naast deze kantoorpleinen zijn er ook vergaderruimtes aanwezig. De vergaderruimtes zijn momenteel te reserveren via Outlook. Alleen je kan je voorstellen als je even snel een belletje wilt plegen of kort moet focussen dit niet altijd gedaan wordt. Het gevolg is dat vergaderruimtes bezet worden gehouden terwijl deze eigenlijk gereserveerd waren. Het gevolg hiervan is dan weer dat je ofwel uit de kamer gezet wordt, of de persoon die de kamer gereserveerd had opzoek moet naar een nieuwe ruimte.

De oplossing voor dit probleem is eigenlijk vrij simpel. Bij de vergaderruimtes komen kleine schermen te hangen, die informatie tonen over de vergaderruimte. Het betreft e-ink schemren(het scherm wat te vinden is op een e-reader) gezien het product op batterijen zal draaien en e-ink schermen heel energie zuinig zijn. Hiernaast moet het ook mogelijk zijn om via de ESP32 een nieuwe reservering te maken, dit kan vanzelfsprekend alleen als de kamer niet al gereserveerd is. Hierdoor kan men wanneer ze even snel een belletje moeten plegen de kamer gemakkelijk reserveren zonder in Outlook te duiken.

Het project wordt gerealiseerd met de watervalmethode. De waterval methode is een methodiek waarbij het project wordt opgedeeld in lineair opeenvolgende fasen. Voordat de nieuwe fase begonnen wordt, wordt eerst de voorgaande fase afgerond. Hierbij doorloopt een project de volgende fases: specificeren, ontwerpen, bouwen, testen en accepteren

Voor het project zijn er een tweetal onderzoeken gepleegd. Namelijk een literatuuronderzoek en een gebruikersonderzoek. Beiden onderzoeken zijn smart opgesteld waarbij eerst de deelvragen worden beantwoord om vervolgens de hoofdvraag te beantwoorden. Hiernaast is er in de onderzoeken ruimte gemaakt voor advies en discussie.

Voor het literatuuronderzoek is de volgende hoofdvraag opgesteld:

“Wat is de best geschikte combinatie van netwerk -en messaging protocollen om een batterijduur van minimaal twee maanden te behalen op de ESP32 gebruik makend van een 2500mAh batterij, waarbij er minimaal twee keer per uur informatie van een vergaderruimte wordt gesynchroniseerd?”

In dit onderzoek kwam in het advies naar voren dat er gebruik gemaakt zal worden van WiFi in combinatie van MQTT. Hiermee hebben we voldoende zendafstand en zal een batterijduur van rond de 2,5 maand gerealiseerd moeten kunnen worden.

Voor het gebruikersonderzoek luidt de hoofdvraag:

“Wat zijn de verwachtingen van de consultants, business- en technisch managers van het te realiseren reserveringssysteem.”

Uit het gebruikersonderzoek bleek vooral dat de informatie snel en gemakkelijk af te lezen moet zijn waarbij men de mogelijkheid moet hebben om een kamer te reserveren met het systeem.

Voor het project, is een ontwikkelstraat gebouwd in docker. De ontwikkel straat bouwt, test en deployed de code voor de server. Voor de ESP32 bouwt en test deze alleen de code gezien het deployen fysiek moet gebeuren op het device. Naast de ontwikkelstraat is het volledige product ontwikkeld inclusief een behuizing voor het product.

De resultaten van de batterijtest waren niet heel positief dit kwam uit op 10 dagen. Wat voornamelijk te danken was aan het hoge energieverbruik is slaapmodus. Wat op zijn beurt weer een defect van het gebruikte ontwikkelbord was, dit energieverbruik lag namelijk op 0,01A wat 0,00001A moet zijn. Doormiddel van het gebruik van een ander ontwikkelbord komt de batterijduur rond de twee jaar te liggen. Ruim binnen de gestelde eis van twee maanden.

Tijdens het traject is er beoogd een viertal A-Competenties aan te tonen namelijk: Leren leren, onderzoek, Professioneel werken en Innovatie. Naast deze vier A-competenties zijn er een drietal B-competenties aangetoond namelijk: Software analyseren, Software ontwerpen en Software realiseren.

Inhoud

Voorwoord 4
Managementsamenvatting 5
1 Inleiding 9
2 Organisatie 10
2.1 Cultuur 11
2.2 Stakeholders 11
3 Achtergrond 12
4 Opdrachtomschrijving 13
4.1 Probleem 13
4.2 Oplossing 13
4.3 Requirements 14
4.3.1 ESP32 14
4.3.2 Server 15
4.3.3 Web interface 15
4.4 Planning 16
5 Methodologie 17
5.1 Watervalmethode 17
5.1.1 Specificeren 18
5.1.2 Ontwerpen 18
5.1.3 Testen 18
5.1.4 Accepteren 18
6 Uitvoering 19
6.1 Specificeren 19
6.1.1 Afstudeerplan 19
6.1.2 Onderzoek 19
6.1.3 Requirements 19
6.1.4 Acceptatieplan 19
6.2 Ontwerpen 19
6.2.1 Functioneel ontwerp 19
6.2.2 Software architectuur document 20
6.2.3 Prototypes 20
6.3 Bouwen 21
6.3.1 Ontwikkelstraat 21
6.3.2 Server, ESP32 en Web interface 21
6.4 Testen 21

6.5 Accepteren 22
7 Onderzoek 23
7.1 Onderzoek netwerk- en messaging protocollen 23
7.1.1 Resultaten 23
7.1.2 Advies 25
7.2 Gebruikersonderzoek 26
7.2.1 Resultaten 26
7.2.2 Advies 27
8 Opbrengsten 28
8.1 Ontwikkelstraat 28
8.2 De server 30
8.3 De ESP32 31
8.4 Web interface 32
8.5 CAD-tekeningen 33
9 Testen 35
9.1 Unit tests 35
9.2 Batterijduur 36
10 Conclusie 37
11 Aanbevelingen 38
12 Reflectie 39
12.1 Aanpak 39
12.2 Planning 39
12.3 Resultaten 39
12.4 Ervaring 40
12.5 Begeleiding 40
12.6 Competenties 41
12.6.1 Leren leren 41
12.6.2 Onderzoek 41
12.6.3 Professioneel werken 42
12.6.4 Innovatie 42
12.6.5 Software analyseren 42
12.6.6 Software ontwerpen 43
12.6.7 Realiseren 43
12.7 Conclusie 43
13 Nawoord 44
14 Bijlagen 45

15 Bibliografie..... 46

1 Inleiding

Dit document bevat de scriptie voor de afstudeeropdracht ‘Kamers Reserveren’. De scriptie wordt geschreven ter verkrijging van de graad Bachelor of Science aan de Hogeschool van Leiden Faculteit Science and Technology.

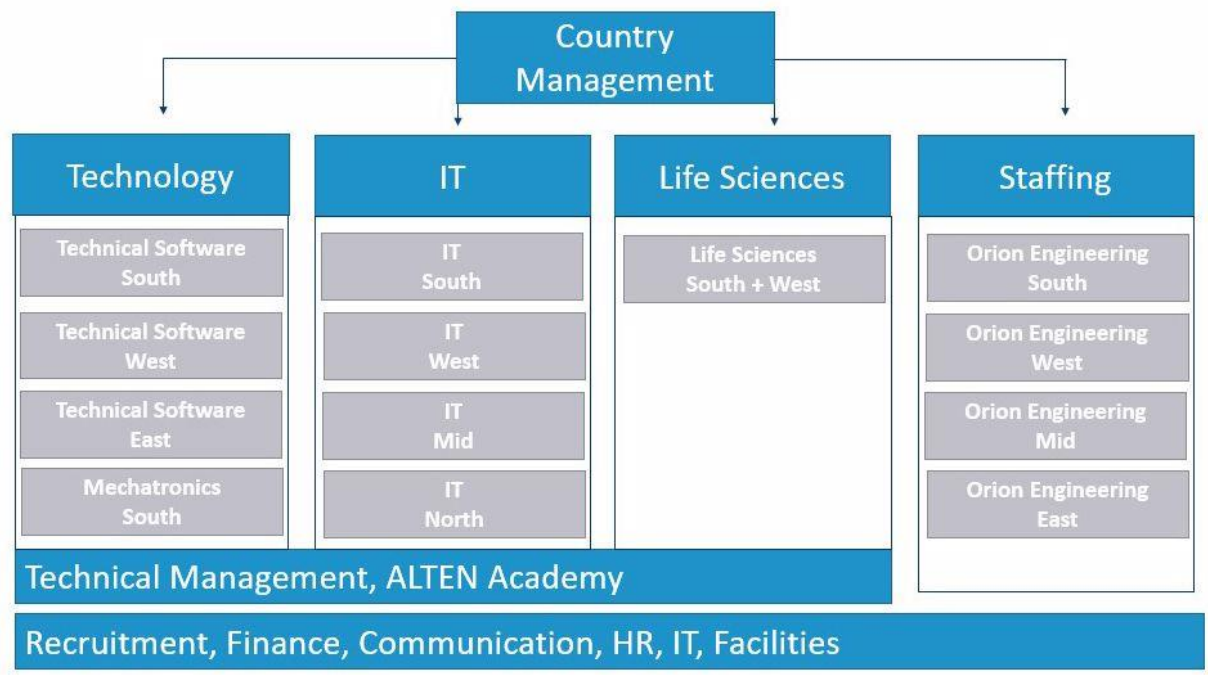
Allereerst zal in dit document de organisatie Alten toegelicht worden. In hoofdstuk 3 zal er achtergrondinformatie met betrekking tot de opdracht en de organisatie gegeven worden. Vervolgens wordt in hoofdstuk 4 de opdracht omschreven. Hierbij wordt er uitgebreid naar het probleem, de oplossing, de requirements en de planning. Eenmaal bekend met Alten en de opdracht zal in hoofdstuk 5 de aanpak beschreven worden, welke methodiek is er gebruikt tijdens het afstuderen. In hoofdstuk 6 zal de methodiek invulling krijgen, wat wordt er in welke fase gebouwd. Daarna wordt er in hoofdstuk 7 gekeken naar de gerealiseerde onderzoeken, hierbij wordt er vooral gericht op de resultaten van de onderzoeken. In hoofdstuk 8 en 9 zullen de opbrengsten en test resultaten beschreven worden, wat is er nu opgeleverd en is het project succesvol. Na het omschrijven van de opbrengen en de test resultaten zal er in hoofdstuk 10 en conclusie getrokken worden en in hoofdstuk 11 zijn de aanbevelingen te vinden voor eventuele vervolg werkzaamheden. In hoofdstuk 12 wordt er gereflecteerd op het gehele afstudeertraject. Tot slot zal er in hoofdstuk 13 een nawoord te vinden zijn van de afstudeerbegeleider Leon Stam. Hoofdstuk **Fout! Verwijzingsbron niet gevonden.** dient vooral ter verduidelijking van de gerealiseerde competenties.

2 Organisatie

Alten is een groot consultancy bedrijf met maar liefst 37.000 werknemers. Waarbij ze actief zijn in 25 landen. Van origine komt het bedrijf uit Frankrijk en is het opgericht in 1988 door Simon Azoulay, Laurent Schwarz en Thierry Woog. Tot op de dag van vandaag wordt dan ook de meeste winst behaald uit de vestiging in Frankrijk. In totaal heeft Alten een omzet van 2,3miljard.

Alten Nederland is opgericht in 2005 door Eric Haesen en Arjen van Herwaarden. Het bedrijf heeft ongeveer 750 medewerkers waarvan ongeveer 90% ingenieurs zijn. De eerste vestiging was geplaatst in het westen van het land in Capelle a/d Ijssel, al snel volgde een tweede locatie in Eindhoven. Inmiddels heeft Alten vijf vestigingen in Nederland.

Iedere vestiging heeft zijn eigen expertise. De werkzaamheden zullen op de vestiging in capelle a/d ijssel plaatsvinden, deze vestiging focust zich voornamelijk op (technisch) softwareontwikkeling. In Figuur 1 is te zien welke branche op welke locatie plaatsvindt.



Figuur 1: Organisatie overzicht Alten

Hierbij staat in Figuur 1 ook de Alten Academy vernoemd. Alten is namelijk een kennis organisatie, op deze manier verbreedt het niet alleen de technische kennis van zijn eigen werknemers maar die van heel Nederland. Alten geeft namelijk verschillende lezingen bij universiteiten en hogescholen. Hiernaast heeft Alten ook gecertificeerde werknemers in dienst waarbij het mogelijk is om trainingen en cursussen te volgen.

Iedere consultant binnen Alten wordt gekoppeld aan een technisch manager en een business manager. Hierbij wordt met de technisch manager inhoudelijk gesproken over de opdracht als je ergens tegen aanloopt en met de business manager worden meer persoonlijke zaken besproken. Dit kan van alles zijn o.a. hoe het gaat bij de klant en of je het werk nog leuk vindt. Eens per drie maanden vindt zo een gesprek verplicht plaats met de business manager, indien nodig kan je natuurlijk ook altijd bij de ze aankloppen voor een extra gesprek.

Ook stagiairs zijn gekoppeld aan een business manager en een technisch manager. Stagiairs spreken de business managers doorgaans iets vaker doordat de werkzaamheden doorgaans ook korter duren.

Naast de managers krijgen de stagiairs ook een begeleider toegewezen. Deze begeleider heeft minimaal twee uur in de week om je te begeleiden. Mijn ervaring is echter dat de begeleider altijd bereid is om je te helpen. Hierbij wordt verwacht dat je zelf aan de bel trekt wanneer je vastloopt of feedback wilt hebben.

2.1 Cultuur

Binnen Alten hangt een informele open sfeer. De werkzaamheden worden op grote kantoorpleinen verricht. Hierdoor is het mogelijk om op ieder moment een praatje te maken met een medeconsultant en eventueel vragen te stellen als je ergens tegenaan loopt. Aan het eind van de vrijdagmiddag is er altijd een korte borrel. In de pauzes is het mogelijk om met je collega's een rondje te lopen of gezellig in de kantine te lunchen. Daarnaast is er ook een voetbal tafel om even te ontspannen.

2.2 Stakeholders

Als we het hebben over stakeholder kunnen we deze onderscheiden in primaire en secundaire stakeholders. De primaire stakeholders zijn vaak direct betrokken bij het project, en dragen bij aan het resultaat. Secundaire stakeholders staan verder weg staan van het project, maar hebben desondanks baat hebben bij het project. (Stakeholdersanalyse, sd)

Het betreft een in-house product zonder commercieel belang. Het product wordt alleen voor Alten ontwikkeld en zal niet doorverkocht worden. Dit betekend dat Alten de voornaamste stakeholder is.

Binnen Alten zal niet iedereen even veel baat hebben bij het project. Voornamelijk de business- en technisch managers hebben baat bij het project. Dit komt doordat zij veelal bezig zijn met klanten en nieuwe collega's binnen te halen. Hierbij is het natuurlijk van belang dat er een vergaderruimte beschikbaar is.

De consultants die voor Alten werken zullen veel minder baat hebben bij het project. Dit komt doordat zij veelal op locatie bij de klant werken. Wanneer de consultants op kantoor bij Alten zijn en een vergaderkamer nodig hebben is het gewenst om snel in te kunnen zien of een ruimte vrij is.

3 Achtergrond

Zoals verteld in paragraaf 2 is Alten een consultancy bedrijf. Dit betekent dat de consultants veelal op locatie zitten bij de klant om een project uit te voeren. Naast het consultancy werk heeft Alten ook het Alten delivery center. Hier worden in-house projecten uitgevoerd. Het Alten delivery center bestaat uit een vast team, gecombineerd bevat dit team vrijwel alle expertises om een project van A tot Z te realiseren.

Hierbij is Alten actief in de volgende branches: Automative, high tech industie, defensie/beveiliging, energie, financieel, lucht- en ruimtevaart, multimedia, verkeer en vervoer en telecom. Zoals te zien is een breed heel scala aan branches.

De business en technisch managers zijn veelal opzoek naar nieuwe klanten, medewerkers en opdrachten. Dat betekent dat ze vaak op reis zijn of in de vergaderruimtes zitten. Hiernaast bieden de technisch managers technische ondersteuning aan de consultants dit kan van alles betekenen. Wanneer de consultant vast loopt op de opdracht kan hij hier bijvoorbeeld voor naar de technisch manager stappen

De business managers zijn er meer voor persoonlijke begeleiding. Hierbij wordt er gekeken hoe het gaat bij de klant, wat er goed gaat of wat er juist beter kan. Eens per drie maanden vindt er een gesprek plaats om dit te peilen.

4 Opdrachtomschrijving

In hoofdstuk 2 en 3 is Alten als organisatie omschreven. In dit hoofdstuk zal de opdracht omschreven worden. Hierbij wordt begonnen met het probleem te omschrijven waarna de oplossing voor het probleem wordt omschreven en de requirements voor de opdracht opgesomd worden.

4.1 Probleem

Zoals verteld in hoofdstuk 2 en 3 heerst er bij Alten een open werksfeer waarbij er meerdere vergaderruimtes beschikbaar zijn. Deze dienen voornamelijk om te vergaderen, echter, als je even in rust wilt werken of een belletje moet plegen kan dit ook in deze ruimtes.

De business -en technisch managers zijn de meest intensieve gebruikers van deze ruimtes. Ze zijn vaak in gesprek met ofwel klanten of de consultants. Hierbij komt ook nog eens dat de managers vaak moeten bellen en dit niet altijd kan op de kantoor pleinen. Hiervoor worden dan ook de vergaderruimtes gebruikt.

De vergaderruimtes zijn momenteel te reserveren via Outlook. Alleen je kan je voorstellen als je even snel een belletje wilt plegen of kort moet focussen dit niet altijd gedaan wordt. Het gevolg is dat vergaderruimtes worden bezet terwijl deze eigenlijk gereserveerd waren. Het gevolg hiervan is dan weer dat je ofwel uit de kamer geschopt wordt, of de persoon die de kamer gereserveerd had opzoek moet naar een nieuwe ruimte.

Hierbij komt ook nog eens dat het wat rommelig over komt als er een klant op bezoek is en de gereserveerde ruimte vervolgens bezet is.

4.2 Oplossing

De oplossing voor dit probleem is eigenlijk vrij simpel. Bij de vergaderruimtes komen kleine schermen te hangen, die informatie tonen over de vergaderruimte. Het betreft e-ink schemren (het scherm wat te vinden is op een e-reader) gezien het product op batterijen zal draaien en e-ink schermen heel energiezuinig zijn.

Deze e-ink schermen worden aangestuurd door een ESP32, dit is een microcontroller van Espressif systems. De esp32 is een goedkope microcontroller die energiezuinig is. Het is de directe opvolger van de ESP8266. Gezien het doel is dat de ESP32 zo min mogelijk aan het werk is zal er gebruik gemaakt worden van een server. Deze server haalt de outlook data op verwerkt het en zal de ESP32 alleen informeren wanneer er data op het scherm aangepast hoeft te worden. Wanneer deze data is aangekomen bij de ESP32 zal het al verwerkt zijn en hoeft het alleen nog getoond te worden. Op deze manier is de ESP32 zo min mogelijk aan het werk.

Hiernaast moet het ook mogelijk zijn om via de ESP32 een nieuwe reservering te maken, dit kan vanzelfsprekend alleen als de kamer niet al gereserveerd is. Hierdoor kan men wanneer ze even snel een belletje moeten plegen de kamer gemakkelijk reserveren zonder in Outlook te duiken.

4.3 Requirements

In dit hoofdstuk zijn de gestelde requirements ten behoeve van het product beschreven. De requirements zijn tot stand gekomen door diverse interviews met de consultants, business en technisch managers. Dit gebruikers onderzoek is te vinden in bijlagen C.

In het acceptatie plan (bijlagen A) staan de requirements verder uitgewerkt tezamen met de acceptatiecriteria voor het product. Het functioneel ontwerp (Bijlagen F) werkt deze requirements verder uit tot use case beschrijvingen, hiernaast worden er in het functioneel ontwerp de alternatieven en limitaties besproken. In het testplan (bijlagen B) staat de uitwerking m.b.t. het testen van het product en de hierbij gestelde criteria.

In de requirements wordt onderscheid gemaakt tussen de ESP32, de server en web interface. Hierbij zullen de requirements aan de volgende rollen worden gekoppeld:

- **Gebruiker:** De persoon die het scherm gebruikt.
- **Service:** De software zelf.
- **Installateur:** De persoon welk de ESP32 initieel instelt en onderhoudt.

Iedere requirement krijgt een weging met de MoSCoW-methode.

4.3.1 ESP32

In Tabel 1 staan de requirements welke gesteld zijn aan de ESP32.

Tabel 1: Requirements ESP32

ID	Omschrijving	MoSCoW
ESP-1	De **gebruiker** kan inzien of er een vergadering bezig is.	Must
ESP -2	De **gebruiker** kan opkomende vergaderingen inzien.	Should
ESP -3	De **gebruiker** kan een vergadering inplannen indien de ruimte vrij is.	Should
ESP -4	De **service** kan draadloos data versturen en ontvangen.	Must
ESP-5	De **service** implementeert een energiezuinig messaging-protocol	Must
ESP-6	De **service** kan een E-ink scherm aansturen.	Must
ESP-7	De **service** kan een website hosten.	Could
ESP-8	De **service** gaat automatisch in een staat van energiebesparing.	Must
ESP-9	De **installateur** kan de ESP32 initieel instellen	Must
ESP-10	De **installateur** kan achteraf de instellingen van de ESP32 aanpassen.	Should
ESP-11	De **gebruiker** kan inchecken wanneer hij een vergadering ingaat	Could
ESP-12	De **service** laat a.d.h.v. ledjes zien wat de status is van een kamer, groen vrij, rood bezet.	Could

4.3.2 Server

In Tabel 2 zijn de requirements te vinden voor de server. Omdat de gebruiker en de installateur geen interactie hebben met de server zijn er hiervoor geen requirements opgesteld.

Tabel 2: Requirements back-end

ID	Omschrijving	MoSCoW
BE-1	De **service** kan authentiseren bij microsoft.	Must
BE-2	De **service** kan data uitlezen van outlook.	Must
BE-3	De **service** kan data versturen naar de ESP32	Must
BE-4	De **service** heeft een database verbinding.	Must
BE-5	De **service** kan data in een database opslaan.	Must
BE-6	De **service** kan data uit een database uitlezen.	Must
BE-7	De **service** heeft REST-API functionaliteiten.	Must
BE-8	De **service** is beveiligd.	Won't
BE-9	De **service** kan data naar outlook toesturen	Should

4.3.3 Web interface

In Tabel 3 zijn de requirements te vinden voor de web interface. Gezien de web interface geheel optioneel was is geen enkele van deze requirements een must.

Tabel 3: Requirements website.

ID	Omschrijving	MoSCoW
WE-1	De **service** kan data van de back-end ophalen.	Should
WE-2	De **service** kan data van de back-end tonen.	Should
WE-3	De **service** kan dat naar de back-end versturen.	Should
WE-4	De **service** is beveiligd.	Could
WE-5	De **installateur** kan de instellingen van de ESP32 aanpassen.	Should

4.4 Planning

Voor het project is een uitgebreide planning gemaakt. In deze planning staan alle op te leveren producten met de huidige status. Hierbij staan in de planning de diverse vakanties vermeld van Leon Stam(stagebegeleider), de hogeschool en mij.

De planning is als volgt gecategoriseerd:

- **Vakantie:** Hierin zijn de vakantie van Leon Stam, de hogeschool en mij te vinden.
- **Ontwerp:** Dit betreft alle documenten welke nodig zijn voor het ontwerp en realisatie van het product.
- **Ontwikkeling:** Dit betreft alles met betrekking tot het bouwen en testen van het product. Deze fase zal het gros van de tijd in beslag nemen
- **Afronding:** In de afronding zal de scriptie opgeleverd worden, het testrapport, de handleiding en zal er een test gedraaid worden om de batterijduur te bepalen.

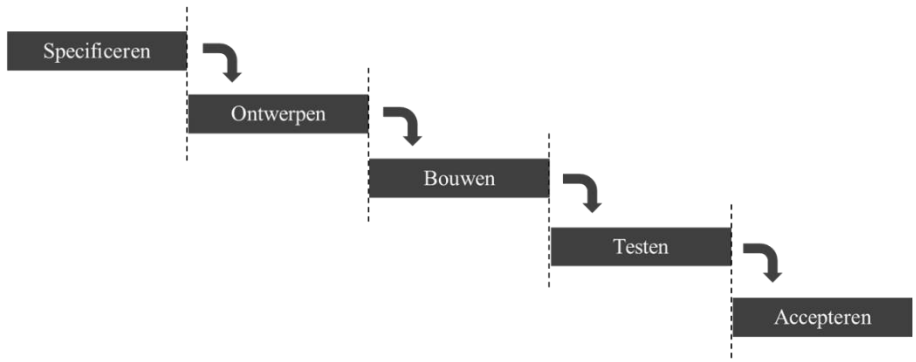
De planning is te vinden in bijlagen E.

5 Methodologie

In dit hoofdstuk ze de methodiek beschreven worden die tijdens het traject is toegepast. Hierbij is er in dit project gebruik gemaakt van de watervalmethode.

5.1 Watervalmethode

De waterval methode is een methodiek waarbij het project wordt opgedeeld in lineair opeenvolgende fasen. Voordat de nieuwe fase begonnen wordt, wordt eerst de voorgaande fase afgerond. Hierbij doorloopt een project de volgende fases: specificeren, ontwerpen, bouwen, testen en accepteren, te zien in Figuur 2. (Bijvank, sd)



Figuur 2: Waterval methodiek.

De watervalmethodiek is minder flexibel dan scrum, immers, als er een fase wordt afgerond kijk je doorgaans niet terug op de vorige fase. Bij scrum lopen deze fases wat door elkaar, waarbij je eigenlijk voor ieder stuk wat je oplevert alle vier de fases doorloopt en vervolgens naar het volgende stuk gaat. Echter, zal het in mijn geval veel overhead bieden, het werk wordt alleen verricht, hierdoor bieden o.a. de dagelijkse stand-ups en stand-down niet veel meerwaarde.

5.1.1 Specificeren

Dit is de eerste fase van de waterval methode. Hierin worden alle requirements vastgesteld in een requirement document. Alle requirements geprioriteerd volgens de MoSCoW-methodiek, hierbij geldt de volgende prioritering:

- Must have: Deze eisen moeten in het eindproduct terugkomen. Zonder deze eisen is het product niet bruikbaar. Al deze eisen zijn de te vinden in Bijlagen A.
- Should have: Deze eisen zijn gewenst, zonder dit is het product nog wel bruikbaar.
- Could have: Deze eisen komen aan bod als de tijd ervoor is.
- Won't have: Deze eisen zullen hoe dan ook niet in het product te vinden zijn.

De volledige lijst aan requiremets zullen in het functioneel ontwerp te vinden zijn (Bijlagen F) (SDLC - Waterfall Model, sd).

Voordat deze requirements opgesteld kunnen worden dient er eerst onderzoek verricht te zijn Tijdens het project zijn er een twee onderzoeken verricht, hierover valt meer te lezen in paragraaf.

Nadat de requirements zijn opgesteld zal het acceptatieplan opgesteld worden, deze dient als afbakening voor het project.

Zoals eerder omschreven vloeit iedere fase in elkaar door. Onder andere de gespecificeerde requirements worden meegenomen naar de ontwerpfase.

5.1.2 Ontwerpen

In deze fase worden de eerder vastgestelde requirements omgezet in ontwerpdocumenten. Allereerst zullen de requirements verder uitgewerkt worden tot use-case beschrijvingen. Hiernaast worden complexe systeem eisen a.d.h.v. uml-diagrammen uitgewerkt.

In deze fase worden ook diverse prototypes ontwikkeld voor de systeem kritische componenten. Deze prototypes zijn slechts om aan te tonen dat de componenten werken en het product daadwerkelijk gerealiseerd kan worden.

5.1.3 Testen

Als eenmaal alle must haves realiseerd zijn, zullen deze getest worden alvorens we doorgaan met de could- en should haves. De testen zullen automatisch uitgevoerd worden d.m.v. een ontwikkelstraat. Hierbij zal zowel de happy als de unhappy flow getest worden.

5.1.4 Accepteren

De laatste fase in de waterval methodiek is de acceptatie fase. Hierin zullen de gestelde acceptatiecriteria getoetst worden. De resultaten van deze toetsing zijn te vinden in het testrapport. Hierbij zal er ook een documentatie van het product opgeleverd worden, dit zal in de vorm van doxygen zijn.

6 Uitvoering

Eenmaal kennis van de aanpak van het project zal in dit hoofdstuk de uitvoering ervan besproken worden. Hierin zal concreet te vinden zijn wat er per fase van de waterval methodiek opgeleverd/gedaan is.

6.1 Specificeren

Zoals in paragraaf 5.1.1 verteld, zijn in deze fasen de onderzoeken en de requirements opgesteld. Hierbij is er begonnen met het verrichten van onderzoek.

6.1.1 Afstudeerplan

Het afstudeertraject begint bij het opstellen van een afstudeerplan. Het afstudeerplan is de eerste afbakening en specificering van het project. In dit plan staat onder andere ook de planning voor het traject.

Het afstudeerplan is te vinden in bijlagen G en de planning is de vinden in bijlagen E.

6.1.2 Onderzoek

Voor het project zijn er twee onderzoeken verricht. Het gaat hierbij om een literatuuronderzoek en een gebruikersonderzoek. Beiden onderzoeken zijn SMART opgesteld. Hierbij worden er hoofd- en deelvragen opgesteld en wordt er toelichting gegeven op de hoofd- en deelvragen. Nadat er onderzoek is verricht naar de deelvragen zullen hier conclusies op volgen waarna de hoofdvraag beantwoord wordt. Naast de conclusies staat er een advies, discussie en mogelijkheden tot vervolgonderzoek in de documenten.

De onderzoeken zijn te vinden in bijlagen C en D. De resultaten van de onderzoeken worden besproken in hoofdstuk 7.

6.1.3 Requirements

Beiden onderzoeken hebben belangrijke inzichten gegeven in de requirements van het product. Hiernaast is een gedeelte van de requirements gehaald uit de opdracht omschrijving te vinden in hoofdstuk 4 en het afstudeerplan. Mede hierom zijn de requirements pas volledig opgesteld nadat beiden onderzoeken verricht waren.

Alle requirements zijn te vinden in het functioneel ontwerp (Bijlagen F).

6.1.4 Acceptatieplan

In het acceptatieplan staan de eisen waaraan de applicatie moet voldoen. Het acceptatieplan kijkt naar de volgende zaken: beheerbaarheid, beveiliging, functionaliteit, gebruiksvriendelijkheid, standaarden en documentatie.

Het acceptatieplan is te vinden in bijlagen A.

6.2 Ontwerpen

De ontwerpfase is de tweede fase in de waterval methode. In deze fase zijn het functioneel ontwerp, het architectuur document en de prototypes gebouwd.

6.2.1 Functioneel ontwerp

In het functioneel ontwerp zijn alle requirements te vinden, zowel de must haves, should haves en could haves. Voor ieder van deze requirements wordt er indien mogelijk gekeken naar alternatieven voor de requirements. Hiernaast worden de limitaties van de requirements benoemd.

In dit document is ook het use-case model en de diverse use-case beschrijvingen te vinden.

Het functioneel ontwerp is te vinden in Bijlagen F.

6.2.2 Software architectuur document

De architectuur van het product staat beschreven in het software architectuur document. Deze is ingericht volgens de structuur zoals beschreven in RUP op Maat. Hierbij wordt er gebruikt gemaakt van het 4+1 view model. In dit document staan de verschillende state, sequentie, class en een entity relationship diagram(en).

Het software architectuur document is te vinden in Bijlagen H.

6.2.3 Prototypes

Voor het product zijn diverse prototypes gebouwd. Het betreft prototypes om aan te tonen dat de verschillende requirements mogelijk zijn. Hierbij zijn er prototypes gebouwd voor:

- Communiucatie ESP32 en MQTT-broker.
    - Dit betreft communicatie voor zowel het versturen als ontvangen van data.
- Aansturen E-ink scherm
    - Dit betreft een prototype voor het tonen van tekst op het e-ink scherm.
- Communicatie Server en MQTT-broker.
    - Dit betreft wederom de communicatie voor het versturen als ontvangen van data.
- Authenticatie bij de Microsoft Graph API.
    - Dit betreft een prototype voor het authentiseren en ophalen van data bij de Graph API.

Voor het gebruiken van de ESP32 prototypes is vanzelfsprekend hardware benodigd. Iedere willekeurige ESP32 Devkit zal moeten volstaan. Voor het E-ink scherm is er gekozen voor de waveshare 7.5inch E-Paper (B) E-Ink Raw Display, 800×480

De prototypes zijn te vinden in Bijlagen I.

6.3 Bouwen

Eenmaal klaar met het ontwerpen en aangetoond dat het product te maken valt a.d.h.v. de prototypes. Kan het product gebouwd gaan worden. Het bouwproces bestaat uit meerdere onderdelen. Deze zullen hieronder toegelicht worden.

6.3.1 Ontwikkelstraat

Allereerst is er begonnen met een ontwikkelstraat op te zetten. Doordat Alten hier geen standaard voor heeft moet dit zelf geregeld worden. Hiervoor is een combinatie van Gitea, Drone.IO en een docker repository gebruikt. Deze programma's draaien allen in een docker container. Hierdoor kunnen opvolgende stagiairs gemakkelijk verder bouwen op de ontwikkelstraat.

Gitea is een open-source Git service welke geschreven is in GO. Hierbij is het speciaal ontwikkeld om lightweight te zijn. (Gitea - Git with a cup of tea, sd)

Drone.IO is een continuous Integration pakket. Echter, kan het ook gebruikt worden voor continuous delivery doeleinden. Drone.IO werkt a.d.h.v. webhooks waarbij het veranderingen kan detecteren in een Git repository. Drone heeft zowel een opensource editie als een enterprise editie. (Drone - Automate Software Build and Testing, sd)

Naast Gitea en Drone.IO is er gebruik gemaakt van een Docker Repository, de bekendste hiervan is de docker hub. Echter, is er in ons geval gebruik gemaakt van een privé repository. Hier worden de succesvolle builds naartoe gepusht door Drone.IO. Op deze manier kunnen oudere builds teruggehaald worden indien nodig.

Alle docker compose files zijn bijgevoegd in Bijlagen J.

6.3.2 Server, ESP32 en Web interface

Het product zal bestaat uit twee onderdelen. Hiervan zijn de server en de ESP32 een must have. De web interface was een should have.

De server dient om informatie van de Microsoft Graph API te ontvangen. Deze verwerkt het vervolgens en stuurt het door naar de ESP32. De ESP32 ontvangt deze informatie, slaat ze op in zijn geheugen, zodat de informatie blijft hangen zodra hij uit deep-sleep wakker wordt. Vervolgens zal de ESP32 de huidig draaiende/opkomende vergadering tonen daarnaast toont het de drie opkomende vergaderingen.

De web interface dient om de instellingen van de ESP32 aan te passen. Hierin kunnen zaken als de WiFi en MQTT-gegevens aanpassen.

De broncode voor de gehele applicatie is te vinden in Bijlagen L

6.4 Testen

Zoals verteld wordt de applicatie automatisch getest door de ontwikkelstraat. Zowel de server als de ESP32 zullen getest worden. Zoals in het acceptatieplan verteld worden in eerste instantie alleen de must haves getest. Hieronder van de web interface niet, als er tijd over is zal deze alsnog getest worden.

In het testplan staat uitgebreid beschreven wat en hoe er getest zal worden. Dit plan is te vinden in bijlagen B. Naast het testplan zijn er voor alle geschreven testen testscenario's uitgewerkt, deze zijn verwerkt in het test rapport.

6.5 Accepteren

De laatste fase in de waterval methodiek is de acceptatie fase. Hierin wordt het gerealiseerde product getoetst aan de acceptatiecriteria. De bevindingen zullen uitgebreid gedocumenteerd worden in het testrapport.

Het test rapport is te vinden in bijlagen K.

7 Onderzoek

In het begin van het afstudeertraject zijn er een tweetal onderzoeken verricht. De resultaten van deze onderzoeken zullen hieronder belicht worden. Hierbij zijn er voor beiden onderzoeken eerst conclusies getrokken op de deelvragen waarna de hoofdvraag beantwoord wordt. In dit document zullen alleen de resultaten behandeld worden.

De volledige onderzoek documenten zijn te vinden in de bijlagen. Voor het onderzoek naar netwerk- en messaging protocollen is dit bijlagen D en het gebruikersonderzoek is te vinden in bijlagen C.

7.1 Onderzoek netwerk- en messaging protocollen

In dit onderzoek is er begonnen met de hoofdvraag:

“Wat is de best geschikte combinatie van netwerk -en messaging protocollen om een batterijduur van minimaal twee maanden te behalen op de ESP32 gebruik makend van een 2500mAh batterij, waarbij er minimaal twee keer per uur informatie van een vergaderruimte wordt gesynchroniseerd?”

Het betreft een kwantitatief literatuuronderzoek. Waarbij het doel was te achterhalen welk netwerk - en messaging protocol er gebruikt gaat worden voor de communicatie tussen de server en de ESP32. Om tot het antwoord op de hoofdvraag te komen is er gebruik gemaakt van aan vijftal deelvragen namelijk:

1. Wat zijn de verschillende draadloze netwerkprotocollen welke gebruikt worden in IOT-oplossingen met een batterij capaciteit van 2500mAh?
2. Welk netwerkprotocol kan een overdrachtssnelheid van minimaal 1Mb, energieverbruik van minder dan 150mAh en zendafstand van minimaal 50meter realiseren?
3. Wat zijn de mogelijkheden om de netwerkprotocollen op de ESP32 te implementeren rekening houdend met daughterboards welke het netwerkprotocol implementeren?
4. Wat zijn de verschillende messaging protocollen welke gebruikt worden voor het versturen van Outlook data?
5. Welk messaging protocol leent zich het beste om te gebruiken op een IOT-oplossing met een batterij capaciteit van 2500mAh?

Het volledige onderzoek is te vinden in bijlagen D, de resultaten van het onderzoek zijn hieronder belicht.

7.1.1 Resultaten

De resultaten van het onderzoek zijn in twee delen op te splitsen, namelijk het te gebruiken netwerkprotocol en het te gebruiken messaging protocol. Hierbij is er gekeken naar: Bluetooth low energy (BLE), ultra-wideband (UWB), ZigBee, LoRaWAN en Wi-Fi.

BLE en UWB vielen af doordat deze gemaakt zijn voor korte aftands communicatie, beiden rond de 10 meter (Karunakar Pothuganti, 2014). Hierbij kan de afstand van BLE verlengd worden door gebruik te maken van een mesh netwerk echter, moet er dan ongeveer elke 10 meter een node geplaatst worden. Dit zal de kosten opdrijven, veel opzet werk en overhead met zich meebrengen.

LoRaWAN, kan communiceren over grootte afstanden, rond de 1000 á 1500 meter. Het probleem van LoRaWAN is echter dat het netwerk niet gemaakt is voor veel dataverkeer en hierdoor is de kans op package loss vrij hoog, waardoor er mogelijk data verloren gaat (Day, 2020). Dat is niet gewenst gezien de ESP32 dan niet meer accuraat zijn data kan vertonen.

Zoals in Tabel 4 te zien is verbruikt ZigBee ongeveer 50mA en heeft het protocol een zendafstand van rond de 100 meter, ruim binnen de gestelde eisen van 150mAh en 50 meter. WiFi heeft net als ZigBee een zendafstand van rond de 100 meter maar verbruikt ongeveer 250mA.

Tabel 4: Communicatie en energieverbruik verschillende netwerkprotocollen

Protocol	Communicatie afstand	Energieverbuik
BLE	10m (Karunakar Pothuganti, 2014)	~20mA (Eduardo Garcia-Espinosa, 2018)
UWB	10m (Karunakar Pothuganti, 2014)	~130mA (Dieter Coppens, 2022)
ZigBee	100m (Karunakar Pothuganti, 2014)	~50mA (Olaide O. Kazeem, 2017)
LoraWAN	1000-1500m (What are LoRa and LoRaWAN?, 2022)	~40mA (The curious case of lorawan devices and coin cell batteries, 2022)
Wifi	100m (Karunakar Pothuganti, 2014)	~250mA (Olaide O. Kazeem, 2017)

De ESP32 verbruikt in slaapmodus ongeveer 0,00001A. Eenmaal bekend wat het verbruik is wanneer de ESP32 actief is en wanneer de ESP32 slaapt kan worden geschat wat de levensduur zal zijn bij een capaciteit 2500mAh. Hiervoor kan de volgende formule gebruikt worden (Awati, sd):

$$Q = I * T$$

De pollingrate is ingesteld op ieder half uur ofwel tweemaal per uur. Voor de schatting is zal ervan uitgegaan worden dat het binnenhalen van pakketten rond de 10 seconden duurt. In realiteit zal dit veel korter zijn gezien het pakketten van rond de 10KB zijn. Krijgen we de volgende formule voor ZigBee:

$$I_{gem} = I_{piek} * \left(\frac{t_{on}}{t_{on} + t_{sleep}}\right) + I_{sleep} * \left(\frac{t_{sleep}}{t_{on} + t_{sleep}}\right)$$

$$I_{gem} = 0{,}05A * \left(\frac{20}{3600}\right) + 0{,}00001A * \left(\frac{3560}{3600}\right) = 0{,}00029A$$

Wat uitkomt op een geschatte batterijduur van:

$$t = \frac{C}{I_{gem}} = \frac{2.5Ah}{0.00029A} = 8621\ uur\ of\ 12maanden$$

Voor WiFi komt de geschatte batterijduur uit op:

$$I_{gem} = 0{,}25A * \left(\frac{20}{3600}\right) + 0{,}00001A * \left(\frac{3560}{3600}\right) = 0{,}0014A$$

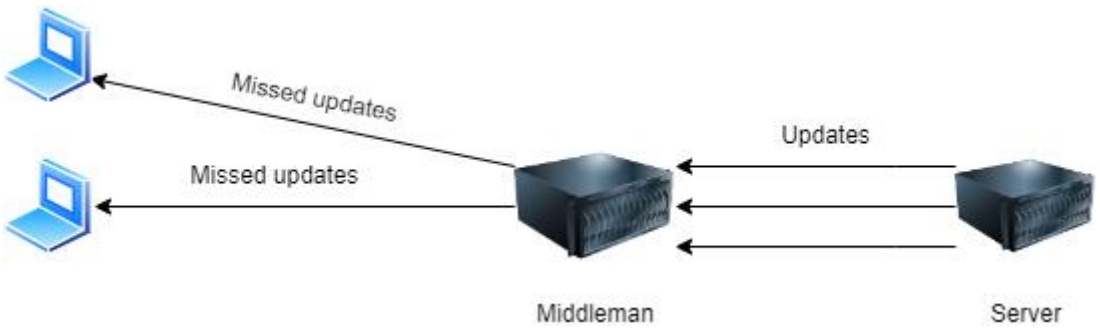
$$t = \frac{C}{I_{gem}} = \frac{2.5Ah}{0{,}0007A} = 1786\ uur\ of\ 2{,}5\ maanden$$

Met zowel WiFi als ZigBee blijft de batterijduur binnen de gestelde batterijduur van 2 maanden. Met ZigBee wordt mogelijk zelfs een batterijduur van ongeveer een jaarbehaald.

Wat betreft de messaging protocollen is er gekeken naar: CoAP, HTTP, MQTT en AMQP.

CoAP en HTTP vielen af doordat deze geen berichten kunnen queuen of bewaren. Dit betekent dat de ESP32 langer uit deepsleep moet blijven doordat we moeten wachten tot de server de gemiste events heeft opgehaald en heeft teruggestuurd. Gezien we proberen het energie verbruik te verminderen is dit niet gewenst. MQTT als AMQP ondersteunen dit wel.

In Figuur 3 zien we de gewenste flow van de pakketten. Hierbij komen de pakketten eerst aan bij de middenman, en worden ze daar opgeslagen in de queue. Zodra een ESP32 verbindt met het internet en de middenman zal het bericht doorgestuurd worden naar de ESP32. Hierdoor hoeven we geen aanvragen te versturen naar de server, waarna de server gaat kijken of er nieuwe data is maar kan de server direct data versturen zodra het beschikbaar is.



Figuur 3: Gewenste flow van de pakketten.

Hierbij is AMQP een groter pakket met aanzienlijk meer opties, doordat het gebouwd is voor zakelijke doeleinden waarbij er veel data verwerkt moet worden. Dit brengt met zich mee dat het meer resources kost en trager is dan MQTT. MQTT daarentegen is gebouwd voor IOT-oplossingen welk gelimiteerd zijn in resources wat bij ons het geval is. Hierdoor lijkt MQTT de beste oplossing voor de toepassing.

7.1.2 Advies

Het advies voor het onderzoek is om gebruik te maken van WiFi in combinatie met MQTT. Dit komt voornamelijk doordat de ESP32 versie die ZigBee ondersteund nog niet uitgebracht is en er geen vooralsnog geen mogelijkheid is om ZigBee werkend te krijgen op de ESP32.

Hoewel het gebruik van WiFi niet ten goede komt van de batterijduur lijkt het alsnog gemakkelijk om de gestelde twee maanden te behalen.

Dit wordt verder ondersteund als we kijken naar al gemaakte projecten op het internet. Er zijn twee projecten gevonden waarbij de ESP32 iedere vijftien minuten verbond met het internet en gebruik maakte van een 2500mAh batterij. In het eerste project is een batterijduur behaald van rond de zeven maanden (Eichhorn, 2021). In het tweede project is een batterijduur behaald van ongeveer een jaar. (How to Run Your ESP8266 for Years on a Battery, 2022)

Een andere mogelijkheid is om de ESP32 op de netstroom aan te sluiten, waardoor er geen zorgen meer zijn m.b.t. batterijverbruik.

7.2 Gebruikersonderzoek

In het gebruikersonderzoek is er begonnen met de hoofdvraag:

“Wat zijn de verwachtingen van de consultants, business- en technisch managers van het te realiseren reserveringssysteem.”

Dit onderzoek richtten zich op het vergaren van informatie met betrekking tot het product, wat zijn de verwachtingen van de gebruikers. Om tot een antwoord hierop te komen zijn er een tiental interview vragen gesteld. Deze vragen zijn gesteld aan twee consultants, twee business managers en één technisch manager en luidde:

1. Maakt u gebruik van vergaderruimtes?
2. Hoe vaak per dag/week/maand maakt u gebruik van vergaderruimtes?
3. Hoelang gebruikt u gemiddeld een vergaderruimte?
4. Is een scherm welke informatie toont over de vergaderruimtes handig?
5. Welke informatie mag dit scherm tonen over een vergaderring denk hierbij aan agendanaam, organisator etc.?
6. Is het tonen van opkomende vergaderingen nuttig, indien nuttig hoeveel moeten dit er zijn?
7. Is een knop om snel een vergadering in te kunnen schieten gewenst?
8. Als een snel boeken knop gewenst is, hoelang moet deze vergadering duren?
9. Is een knop om een vergadering te annuleren gewenst?
10. Eventuele verdere eisen?
11. Moeten de instellingen van het apparaat instelbaar/aanpasbar worden? Denk hierbij aan wifi-instellingen, polling-rate en gekoppelde kamer.
12. Als de instellingen instelbaar/aanpasbaar moeten zijn, wat is hierbij de voorkeurs methode voor het instellen e.g. Website, Config file of via de DB?

Hierbij zijn de vragen als volgt gecategoriseerd:

- Algemene vragen: vragen 1 t/m 3
- Product specifieke vragen: vragen 4 t/m 10
- Technische vragen: vragen 11 en 12

De algemene en product specifieke vragen aan alle geïnterviewde gesteld. Waarbij de technische vragen alleen aan de technisch manager gesteld zijn.

Het volledige onderzoek is te vinden in de bijlagen C, de resultaten van het onderzoek zijn hieronder belicht.

7.2.1 Resultaten

Uit het gebruikers onderzoek kwam naar voren kwam dat vrijwel iedereen meerwaarde zag in het product. Hierbij is het wel belangrijk dat het scherm gemakkelijk af te lezen is, er moet in een oogopslag te zien zijn of de kamer vrij of bezet is. Hiernaast moet er een mogelijkheid komen om een kamer te reserveren via het systeem, waarbij een tijdsduur van 30 minuten gehanteerd kan worden. Als er een boeking aankomt voor de 30minuten kan de resterende tijd geboekt worden. Hierbij is het van uiterst belang dat de huidige boekingen niet overschreven worden. Een knop om een vergadering te annuleren is niet gewenst i.v.m. mogelijk misbruik hiervan.

Het scherm hoeft niet veel informatie te tonen van een vergadering, tijdsduur en organisator is voldoende. Het tonen van de titel is niet gewenst omdat dit mogelijk privacygevoelige informatie kan bevatten.

De ESP32 zal initieel instelbaar moeten zijn, een configuratie bestand hiervoor is voldoende. Echter, zal een web interface ideaal zijn. Beveiliging technisch hoeft er geen rekening gehouden te worden met het hashen van WiFi-wachtwoorden gezien het systeem lokaal draait op het netwerk van Alten draait en niet beschikbaar gesteld wordt op het internet.

7.2.2 Advies

Het reserveringscherm moet vooral gemakkelijk af te lezen zijn, waarbij alleen de reserveringsduur en organisator van belang zijn. Hierbij moeten de opkomende vergaderingen ook getoond worden op het scherm, in eerste instantie zullen dit twee à drie vergaderingen zijn, wat overeenkomt met één à twee uur vooruitkijken.

Alle geïnterviewde gaven aan dat een knop om snel een vergadering te boeken nuttig was. Hierom zal dit zeker van waarde zijn in het eindproduct. Waarbij bij de initiële implementatie een reserveringsduur van een half uur zal volstaan. Deze wens zal als must have beschouwd worden.

Een annuleren knop is voor nu geen prioriteit te zijn. Het nut hiervan moet uit praktijk nog blijven en niet iedereen gaf aan er nut in te zien. Van annuleren knop kan gemakkelijk misbruik gemaakt worden door vergaderingen uit het systeem te halen die er niet uit gehaald mogen worden. Voor nu is een annuleren knop een won't have.

Het is van belang dat het apparaat ingesteld kan worden, initieel kan dit via een config file gedaan worden. In de toekomst is het gewenst om hiervoor een web interface te gebruiken. Het gebruik van een config file zal als must have beschouwd worden waarbij en een webinterface als should have.

Verder kan de bezetting van een vergaderruimte beter inzichtelijk gemaakt worden door het toepassen van een rood en groen led lampje. Hierbij is een rood lampje bezet en een groen lampje vrij. Dit is een vrij gemakkelijk systeem om te implementeren en naar wens van Dennis Karmelk. Dit wordt wederom als should have beschouwd gezien het niet essentieel is voor de werking van het product.

Het Check-in systeem van Jordi Roer is lastiger om te implementeren en zal extra interactie van de gebruiker vereisen. Voordat dit systeem geïmplementeerd zal er eerst onderzoek gedaan moeten worden naar de belangstelling hiervan. Zoals Jordi Roer zelf al aangaf zal dit gezien worden als could have.

Damian Verbeek gaf aan een apart reserveringssysteem voor vergaderruimtes die niet aan Outlook gekoppeld zijn te kunnen reserveren. Dit zal aanzienlijk veel extra werk kosten, waarbij het gemakkelijker is de ruimte toe te voegen aan het al bestaande Outlook systeem. Deze wens valt buiten de scope van het project.

8 Opbrengsten

In dit hoofdstuk zullen de opbrengsten van het gemaakte werk beschreven worden. Dit betreft alle fysieke en softwarematige producten.

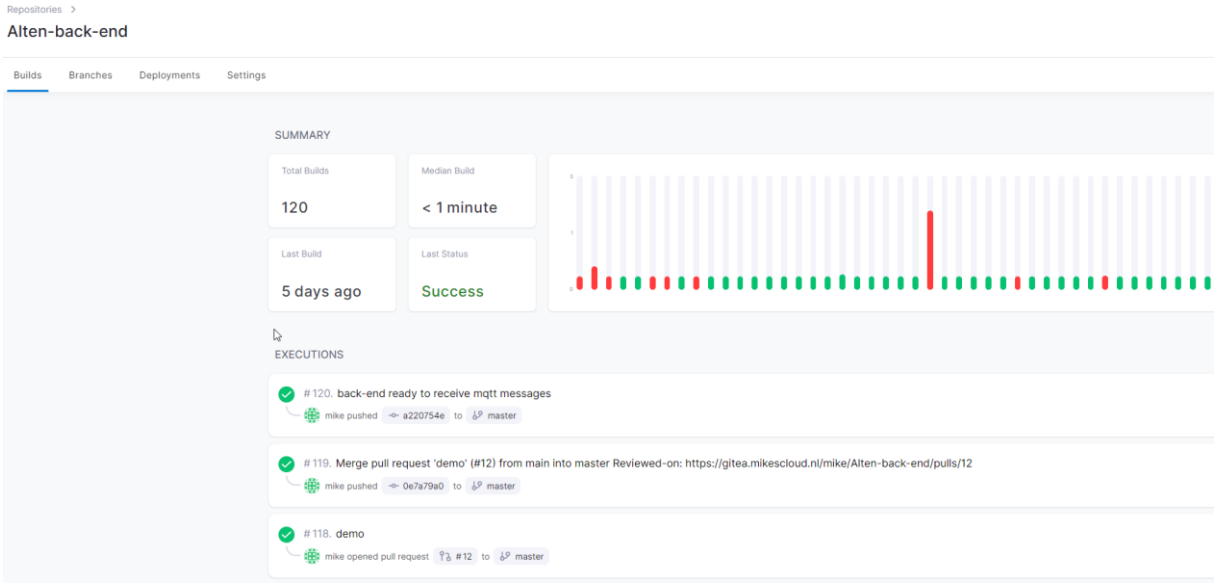
8.1 Ontwikkelstraat

Zoals in paragraaf 6.3.1 verteld is, is er een ontwikkelstraat opgezet. Hiervoor is een combinatie van: Gitea, Drone.IO en een docker repository opgezet. Allen van deze applicaties draaien in Docker. Gezien Alten geen ontwikkelstraat toekent aan stagiairs moest deze zelf opgezet worden.

Om het werk voor de volgende stagiairs makkelijker te maken zijn de docker compose files bijgevoegd in de bijlagen. Gezien deze omgeving op mijn persoonlijke VPS draait moeten de bestanden minimaal aangepast worden om werkend te krijgen op een andere server.

De pijplijn is op de volgende manier opgezet. Wanneer een gebruiker nieuwe code pusht naar de repository zal de Drone worker deze oppakken en de code builden en testen. Gezien de master branch beveiligd is kan hier alleen code naar toe gepusht worden met een pull-request. Wanneer deze wordt aangemaakt zal de drone de code builden, testen en als deze fases succesvol doorlopen worden ook deployen.

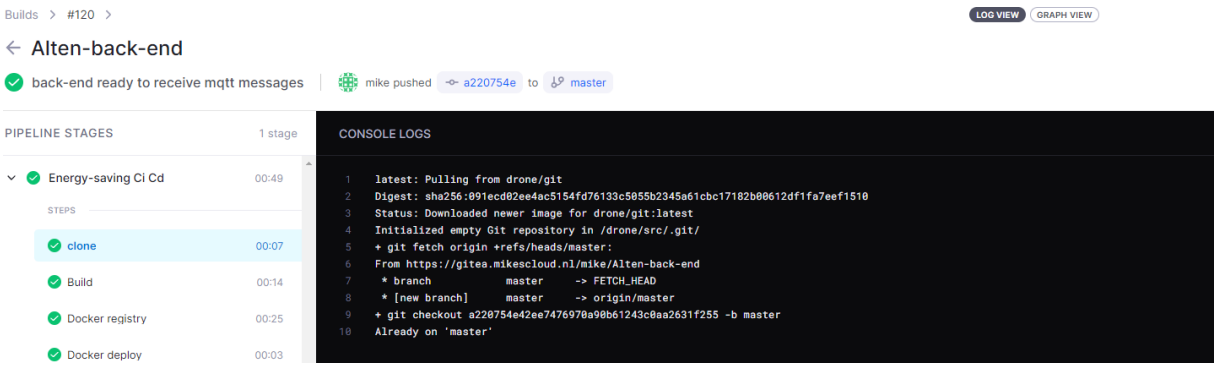
In Figuur 4 is te zien hoe de Drone omgeving eruitziet.



Figuur 4: Drone omgeving.

Hierbij zien we in Figuur 8 wanneer de laatste build is gedraaid, in dit geval vijf dagen geleden. In een grafiekje de gefaalde en succesvolle builds, en onderin alle gedraaide builds.

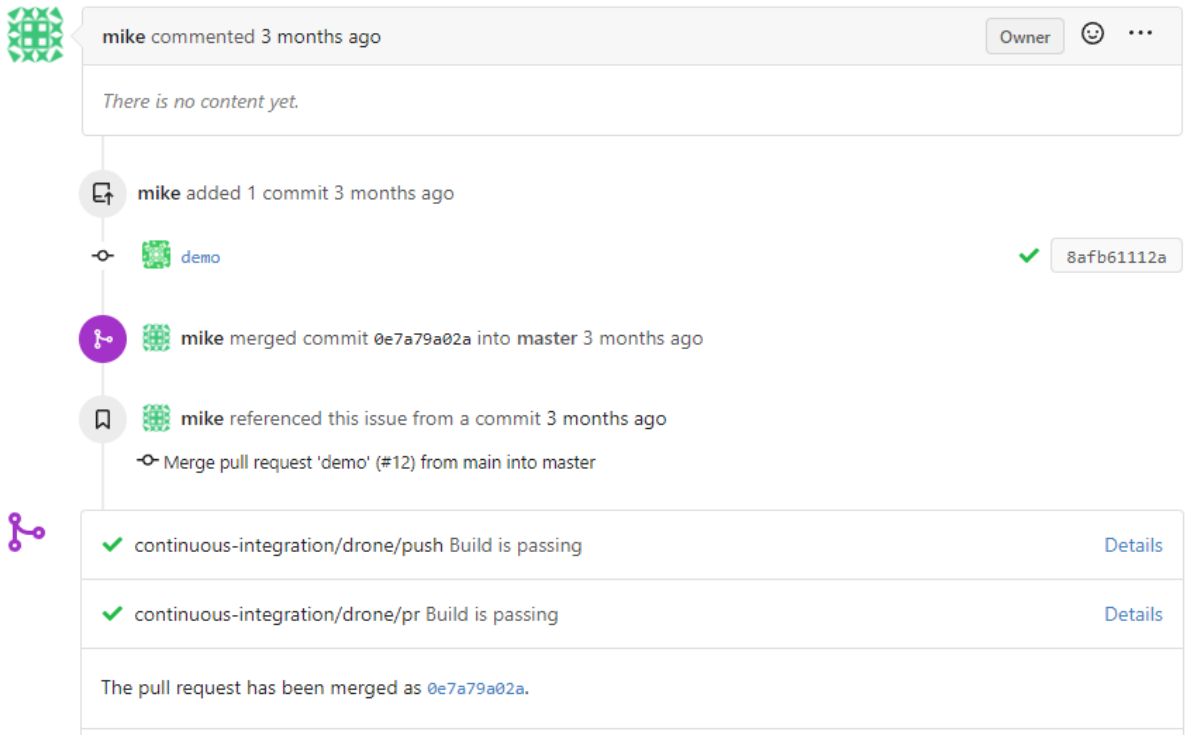
Als we zo'n build aandrukken krijgen we een duidelijk beeld van de stappen die doorlopen worden. Dit is te zien in Figuur 5.



Figuur 5: Inzicht Drone build

Hierbij wordt iedere versie van die naar de master wordt gepusht ook opgeslagen in de Docker registry. Hierdoor kunnen voorgaande versies terughalen als er fouten zitten in de nieuwe versie.

Naast de drone omgeving staan in Gitea ook de resultaten van de Drone build te zien in Figuur 6.



Figuur 6: Gitea resultaten Drone build.

Via de Drone omgeving is het mogelijk om zogeheten secrets mee te builden. Hierdoor kunnen we o.a. API-keys die normaliter in de settings staan verbergen en mee builden tijdens het deployen. Hierdoor blijft de repository vrij van deze gegevens en zal een hacker deze gegevens niet inzien als ze bij de repository komen.

8.2 De server

De server is geschreven in .NET. Hierbij is er gebruik gemaakt van subscriptions, om de data van Microsoft te ontvangen. Hierdoor hoeft er niet iedere x aantal seconden gekeken te worden of er nieuwe reservering gegevens zijn, maar zal Microsoft ons informeren zodra er nieuwe data beschikbaar is.

De subscriptions zijn maar een bepaalde periode geldig voordat deze vernieuwd moeten worden. Om dit te regelen is er gebruik gemaakt van een background worker welke ieder uur kijkt of er subscriptions aflopen. Als er subscriptions zijn die aflopen worden deze automatisch vernieuwd.

Wanneer de server informatie ontvangt van de Microsoft Graph API zal deze de gegevens automatisch verwerken. Hierbij zijn er een aantal voorwaarden voordat de server de gegevens doorstuurt naar de MQTT-broker

- De informatie moet voor de huidige dag zijn
    - Als de server informatie ontvangt voor een andere dag zal hij de MQTT-broker niet informeren, dit gebeurt dan automatisch de volgende dag.
- De informatie mag niet in het verleden zijn
    - Als er een boeking in het verleden wordt aangepast/aangemaakt zal de server dit negeren en hier niets mee doen. Dit voorkomt dat de ESP32 onnodig data gaat verwerken. Subscriptions
- De informatie moet gekoppeld zijn aan een bestaand systeem
    - De server kijkt of de kamer waarvan de informatie binnenkomt gekoppeld is aan een ESP32. Als dit niet het geval is wordt er niets gedaan met de gegevens.

De server kan ook informatie ontvangen van de MQTT-broker. Hierbij wordt er naar twee topics geluisterd namelijk; devices en booking. Via het devices topic kunnen nieuwe ESP32’s zich bekend maken bij de server. Wanneer dit gebeurt zal de server de ESP32 opslaan in de database, en de kalender gegevens ophalen en doorsturen voor de gekoppelde kamer.

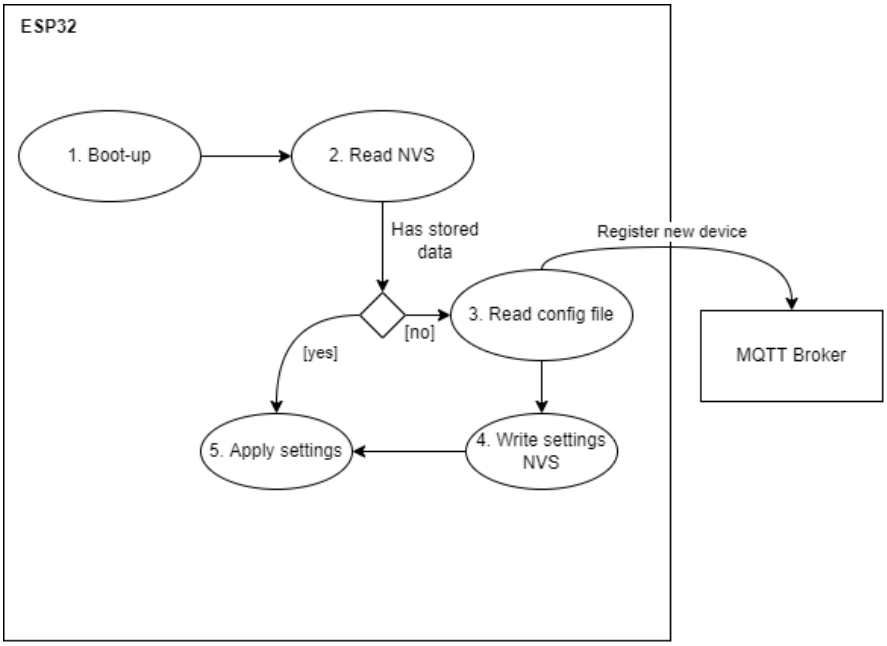
Het booking topic wordt gebruikt om nieuwe reserveringen in te schieten. Op de ESP32 is het namelijk mogelijk om via een knop een reservering aan te maken. Als de ruimte niet al gereserveerd is zal het systeem de ruimte reserveren.

Beiden van deze topics maken gebruik van wildcards. Hierbij stuurt de ESP32 de data naar het volgende adres: devices/{mac_address}. Hierbij is het mac_address uniek per ESP32. De server weet vervolgens aan waar de data vandaan komt.

8.3 De ESP32

De ESP32 dient ter aansturing van het E-ink scherm. Deze is geschreven in C++ gebruikmakend van de ESP-IDF library.

Wanneer de ESP32 voor het eerst opgestart wordt zal hij zijn configuratie file uitlezen en deze instellingen toepassen. Vervolgens zullen deze instellingen naar de server gestuurd worden via MQTT over het devices/{mac_address} topic. Dit geeft ons het state diagram uit Figuur 7. Hierover valt meer te lezen in het software architectuur document.



Figuur 7: State diagram eerste keer opstarten ESP32

Wanneer de ESP32 zich bekend heeft gemaakt bij de server, zal hij de kalender van de gekoppelde kamer toegestuurd krijgen. Dit betreft de gegevens voor de hele dag. Deze slaat hij vervolgens op in zijn non-volatile storage (NVS), dit is namelijk het enige type geheugen wat opgeslagen blijft wanneer de ESP32 uit deepsleep komt. Wanneer deze gegevens binnenkomen zullen de eerste drie items uit de array getoond worden. Als de array leeg is zal er “Room is free” getoond worden. Indien er geen vergadering bezig is zal er “Free till: xx:xx” getoond worden.

Wanneer het scherm is bijgewerkt zal de ESP32 in deepsleep modus gaan. Dit is de modus waarin de meesten energie bespaard wordt. Voor hoelang de ESP32 in deep sleep gaat ligt aan de volgende zaken:

1. Na 18:00 slaapt de ESP32 voor 12uur lang, tot 06:00.
2. Als er tijd vrij is slapen we voor de volledige polling rate instelbaar in het configuratie bestand.
3. Als er minder tijd vrij is dan de ingestelde polling rate wordt de resterende tijd gebruikt om te slapen.

Bovenstaande lijst is op volgorde van prioriteit geplaatst.

Wanneer de ESP32 wakker wordt uit zijn deepsleep controleert hij allereerst bij de MQTT-broker voor nieuwe informatie m.b.t. de outlook gegevens. Als er nieuwe informatie is vervangt hij de al bestaande informatie. Het scherm wordt alleen geüpdatet indien de huidig draaiende vergadering voltooid is of als er nieuwe informatie beschikbaar is gekomen via de MQTT-broker.

Om te bepalen of de huidige vergadering voltooid is wordt er gebruikt gemaakt van de systeemtijd. Deze synchroniseert wanneer de ESP32 voor het eerst opstart met de Nederlandse NTP server. Hierna gebeurt dit ongeveer iedere twee uur omdat de Real-time clock(RTC) van de ESP32 niet gekalibreerd is een kleine afwijking heeft naar verloop van tijd. Op deze manier blijft de afwijking minimaal en is de vertoning van de data accuraat.

Als er fouten optreden e.g. het niet kunnen verbinden met het WiFi netwerk zal dit getoond worden op het e-ink scherm. Hierbij zal de ESP32 automatisch proberen te reconnecten, als dit niet lukt valt hij terug in deepsleep voor de ingestelde polling-rate.

De ESP32 heeft ook de mogelijkheid om een kamer te reserveren via een knop. Deze knop heeft meerdere functionaliteiten, wanneer deze eenmalig ingedrukt wordt zal de geforceerd ESP32 uit deepsleep komen en bij de MQTT-broker kijken of er nieuwe informatie beschikbaar is.

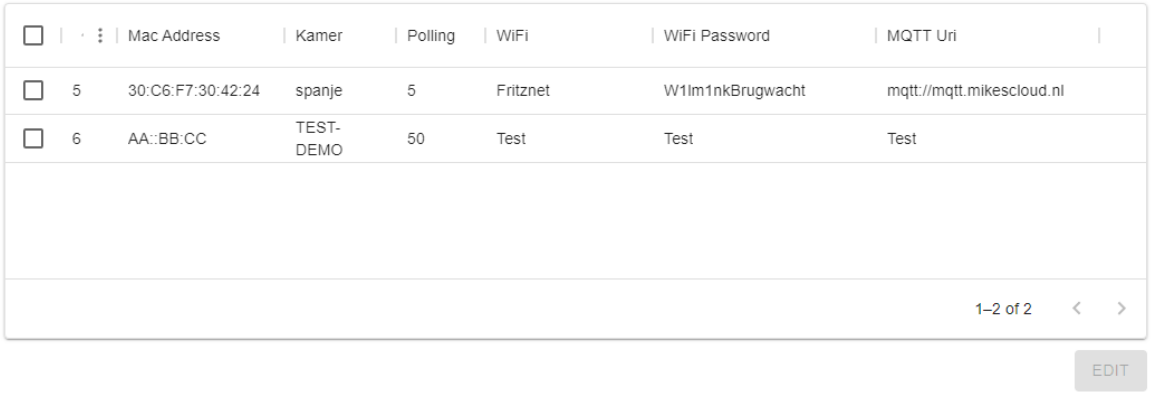
Voor iedere opvolgende druk op de knop zal de kamer gereserveerd worden. Hierbij geldt dat iedere druk op de knop 30 minuten aan de reserveringtijd toevoegt, waarbij er maximaal vier keer gedrukt kan worden of twee uur gereserveerd kan worden. Als er niet genoeg tijd beschikbaar is zal de beschikbare tijd gebruikt worden.

Hierbij heeft de ESP32 ook twee ledjes, een rode en een groene. Deze geven aan of de kamer bezet of vrij is. Wanneer het rode ledje brand is de kamer bezet, wanneer de groene brand is de kamer vrij. Hierdoor is het op afstand makkelijk te zien of een kamer vrij of bezet is.

Zowel het langer boeken dan 30 minuten als de ledjes waren geen must haves en zijn geïmplementeerd doordat er tijd over was.

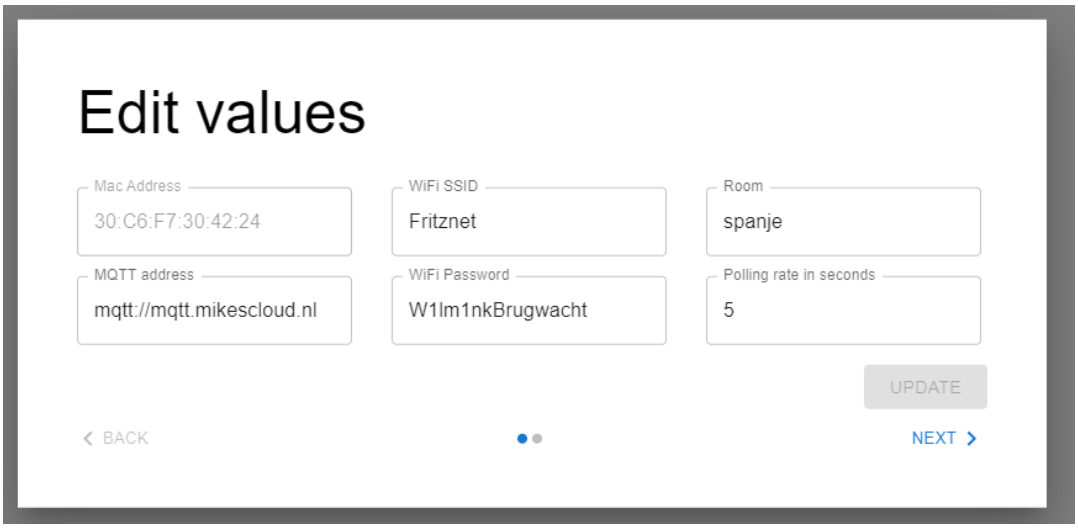
8.4 Web interface

Het is mogelijk om de ESP32 in te stellen via een web interface. Hiervoor is het wel verplicht dat het systeem kenbaar is bij de server, initieel instellen moet dus vooralsnog via het configuratie bestand. Echter, iedere opvolgende aanpassing kan via de web interface. De interface is een simpel tabelletje waarin alle geregistreerde apparaten te vinden zijn. Te zien in Figuur 8.



Figuur 8: Web interface room availability energy saving

In Figuur 8 zijn er ook checkboxes te zien. Wanneer we deze aanklikken zal de “Edit” knop drukbaar worden. Er is een mogelijk om één enkele rij te updaten of meerdere regels, wanneer er meerder geselecteerd zijn, te zien is in



Figuur 9: Aanpassen regels

Wanneer er iets is aangepast in de regels zal de “update” knop beschikbaar worden. De gegevens worden vervolgens naar de REST api van de server verstuurd, waarna de database bijgewerkt wordt. En de nieuwe instellingen doorgestuurd worden naar de ESP32 via de MQTT-broker.

Als de kamer wordt aangepast zal ook de nieuwe kalender opgehaald worden en doorgestuurd worden naar de ESP32.

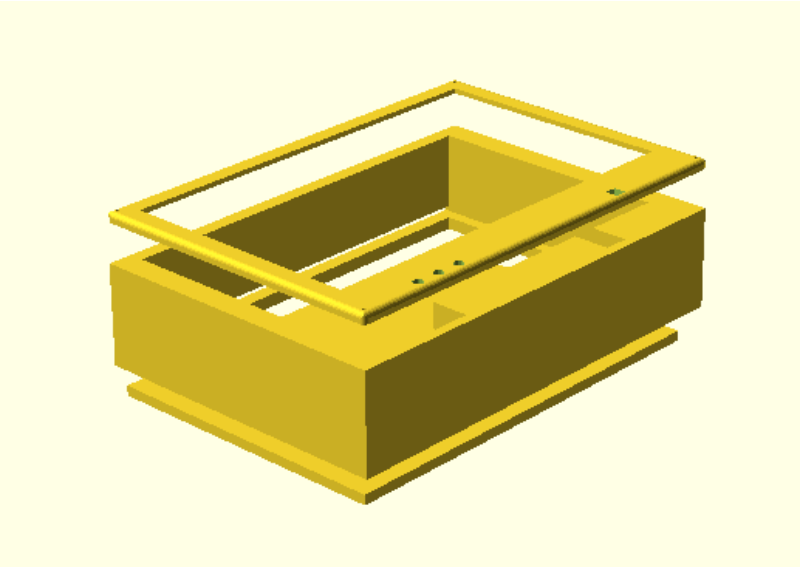
Zoals in het functioneel ontwerp te lezen was is de web interface geen must have, maar een should have. Deze is geïmplementeerd doordat er tijd beschikbaar was.

8.5 CAD-tekeningen

Hoewel CAD-tekeningen volledig buiten de scope van het product vallen zijn deze alsnog gemaakt buiten de stage uren om. Dit is gedaan om een vollediger en professioneler eindproduct te realiseren.

In de behuizing valt het scherm en de hardware zit in een doos achter het scherm. Deze hardware kunnen we benaderen door een schuifluikje aan de achterkant van de doos. De CAD-tekeningen zijn gemaakt in OpenSCAD. Dit is een gratis programma waarin met programmeermatig een 3D-ontwerp kan maken.

In Figuur 10 zien we het 3D-ontwerp van de behuizing. Hierin valt het scherm tussen de grootte doos en het frontje om als een soort sandwich vast te komen. Alle hardware kan in de doos geplaatst worden, welke dicht kan door middel van een schuif mechanisme.



Figuur 10: CAD-tekening behuizing

In Figuur 11 zien we het resultaat van de CAD-tekeningen. Hierbij is het scherm nog niet dicht geschroefd maar zit het vast met punaises zodat het geheel makkelijk uit elkaar komt.

Figuur 11: Resultaat CAD-tekeningen

9 Testen

Ten behoeve van het product zijn er unit tests geschreven voor het valideren van de code en is er een batterij test uitgevoerd om de levensduur van het product te valideren. In dit hoofdstuk zullen de resultaten van deze tests te vinden zijn.

In het testrapport wordt onder andere nog getoetst of de ISO-25010 normen voldaan, of de gestelde requirements behaald zijn en of de code aansluit op het ontwerp. Het testrapport is bijgevoegd in bijlagen K.

9.1 Unit tests

Voor zowel de server als de ESP32 zijn unit tests geschreven voor de functionaliteiten van de geschreven code te valideren.

Om de ESP32 automatisch te kunnen builden en testen door middel van de ontwikkelstraat is er gebruik gemaakt van QEMU. QEMU is een opensource machine emulator (QEMU - A generic and open source machine emulator and virtualizer, n.d.). De emulator kan gedraaid worden in docker en ondersteund de ESP32 in combinatie met de ESP-IDF firmware. Hierdoor kan de ontwikkelstraat een container openen die QEMU draait, de code builden en flashen en worden de unit tests automatisch gedraaid door de ontwikkelstraat.

Het gebruik van QEMU brengt wel een aantal limitaties met zich mee. Alles wat internet verreist kan niet gebruikt worden op de emulator omdat dit een exceptie met zich meebrengt waardoor de emulator crasht. Alles wat via SPI moet kan niet worden uitgevoerd omdat de emulator hierdoor oneindig lang op een antwoord blijft wachten en deze nooit komt. Normaliter zal de ESP32 doorgaan met de rest van de code.

Het gebruikte test framework op de server is MSTest, dit is het officiele test framework van Microsoft. Voor de ESP32 is gebruik gemaakt van Unity, dit is het test framework waarmee de ESP32 standaard komt met de ESP-IDF library.

In Tabel 5 staan de resultaten van de geschreven unit tests.

Tabel 5: Resultaten unit tests

Platform	Aantal Tests	Line coverage	Branch coverage	Method coverage
Server	98	34.84%	65.05%	67.61%
ESP32	28	-	-	-

Naast het de unit tests is de code voor de server getest op een Windows, IOS en Linux machine. De code van de ESP32 is vanzelfsprekend alleen getest op de ESP32.

9.2 Batterijduur

Voor de batterijduur test is er gesteld dat de levensduur bij voorkeur minimaal twee maanden is. Dit moet behaald worden bij een polling rate van tweemaal per uur.

Bij het testen is ervan uitgegaan dat alleen de must haves gerealiseerd zijn. De ESP32 heeft hierbij dus geen aansturing van ledjes die de batterijduur doen verlagen. Ten behoeve van het testen is de polling rate op iedere vijf seconden gezet. Hiermee is een batterijduur behaald van twee en een halve dag.

Dit komt grofweg overeen met de resultaten meetresultaten van de multimeter. Tijdens de metingen was het piek verbruik 0,14A en in deep sleep rond de 0,01A. Een cyclus duurt in totaal rond de 7 seconden waarvan 2 seconden actief en 5 seconden in deep sleep. De formule om van A naar Ah te gaan is als volgt (Awati, sd):

$$Q = I * T$$

Dit komt uit op een verbruik van:

$$I_{gem} = I_{piek} * \left(\frac{t_{on}}{t_{on} + t_{sleep}}\right) + I_{sleep} * \frac{t_{sleep}}{t_{on} + t_{sleep}}$$

$$I_{gem} = 0{,}14A * \left(\frac{2}{2+5}\right) + 0{,}01A * \left(\frac{5}{2+5}\right) = 0{,}0462A$$

Wat uitkomt op een batterijduur van:

$$t = \frac{C}{I_{gem}} = \frac{2.5Ah}{0{,}0462A} = 54uur$$

Dit komt ongeveer overeen met de eerder gemeten twee en een halve dag gezien de ESP32 maar zeer kort het piek verbruik van 0,14A verbruikt en in onze berekening dit constant is. Dit kan mogelijk het verschil verklaren.

Eenmaal bekend wat het verbruik is kan er gemakkelijk uitgerekend worden wat dit zou betekenen als we de polling rate van tweemaal per uur nemen. Hierbij worden er twee cyclussen gedraaid van ongeveer 2 seconden wat betekend dat er in totaal 4 seconden data verwerkt wordt, de overige 3596 seconden slaapt de ESP32. Dit komt uit op een verbruik van:

$$I_{gem}\ 0{,}14A * \left(\frac{4}{3600}\right) + 0{,}01A * \left(\frac{3596}{3600}\right) = 0{,}0101Ah$$

Wat een batterijduur zal opleveren van:

$$t = \frac{2.5Ah}{0{,}0101A} = 247\ uur\ of\ 10\ dagen$$

Als de ESP32 constant in deepsleep zal verkeren en nooit iets anders doet verbruiken het apparaat 0,01Ah dit zal een batterijduur van 250 uur opleveren. Wanneer de WiFi chip constant aanstaat en er constant 0,14Ah verbruikt wordt zal de batterijduur 18 uur zijn.

10 Conclusie

De resultaten zijn grotendeels positief; Alle verplichte requirements zijn geïmplanteerd en getest. Hiernaast zijn er nog een aantal should- en could haves gerealiseerd en is er zelfs een behuizing gemaakt voor het geheel. Het builden, testen en deployen gaat automatisch door middel van de ontwikkelstraat. De ontwikkelstraat is volledig opgezet in Docker waardoor eventuele vervolg stagiairs gemakkelijk de Docker scripts kunnen draaien en een functionele ontwikkelstraat hebben.

De resultaten van de batterijduurtest waren minder positief. Wanneer de ESP32 tweemaal per uur polled en de rest van de tijd in deepsleep verkeerd wordt er een batterijduur van tien dagen behaald. Hierbij was het streven naar een batterijduur van twee maanden bij een polling rate van tweemaal per uur. Als de ESP32 constant in deepsleep wordt er alsnog een batterijduur van 10 dagen behaald. Dit komt doordat het vebruik in deepsleep te hoog zit namelijk op 0.01Ah tegenover de verwachte 0,00001A (Insight Into ESP32 Sleep Modes & Their Power Consumption, sd). Een mogelijke verklaring hiervoor is dat er een devkit-c gebruikt wordt van de ESP32. Deze hebben een aanzienlijk hoger deepsleep verbruik dan andere ontwikkelborden zie Figuur 12.

	Reference [mA]	Light-Sleep [mA]	Deep-Sleep [mA]	Hibernation [mA]
ESP32 – DevKitC	51	10	9	9
Ai-Thinker NodeMCU-32S	55	15.05	6.18	6.18
Adafruit HUZZAH32	47	8.43	6.81	6.80
Sparkfun ESP32 Thing	41	5.67	4.43	4.43
FireBeetle ESP32	39	1.94	0.011	0.008
WiPy 3.0	192	-	0.015	-

Figuur 12: Energieverbruik verschillende ESP32's (Guide to Reduce the ESP32 Power Consumption by 95%, sd)

Het gebruik van een ander ontwikkelbord zal de batterijduur mogelijk aanzienlijk verbeteren. Wanneer de deepsleep waardes van de FireBeetle ESP32 worden gebruikt zal de batterijduur 311 maanden of 25 jaar zijn als er alleen gebruik gemaakt wordt van deepsleep modus. Dit is een aanzienlijk verschil tegenover de huidige 10 dagen. Wanneer de gestelde pollingrate van tweemaal per uur toegepast wordt zal dit een verbruik geven van:

$$I_{gem}\ 0{,}14A * \left(\frac{4}{3600}\right) + \ 0{,}000011A * \left(\frac{3596}{3600}\right) = \ 0.00017Ah$$

Wat een batterijduur oplevert van 15011 uur ofwel bijna twee jaar. Ruim binnen de gestelde eis van twee maanden.

11 Aanbevelingen

Voor de doorontwikkeling van het product is het belangrijk dat er gekeken wordt naar de volgende onderwerpen:

- De testen van de batterijduur moeten opnieuw uitgevoerd worden met een ander ontwikkelbord, bij voorkeur met de FireBeetle ESP32 gezien deze het laagste verbruik heeft in deepsleep.
- De behuizing van het product moet opnieuw ontwikkeld worden zodat deze op een muur gehangen kan worden.
- Het effect van het gebruik van de led lampjes ter indicatie van de status van een vergaderruimte moet gemeten worden.
- Het gebruik van een ander E-ink scherm welk partial updating toestaat. Het huidig gebruikte scherm staat dit niet toe wat het updaten van het scherm aanzienlijk vertraagd, de huidige updatetijd zit rond de 7 minuten.
- Momenteel werkt het systeem alleen voor de ontwikkelomgeving. Er moet onderzocht worden wat de implicaties zijn als het systeem op de Outlook omgeving van Alten gaat draaien en wat de mogelijkheden hiervan zijn.
- De ESP32 krijgt soms dubbele updates van de broker, dit is een limitatie van het QoS niveau welk ingesteld staat op één. Door de dubbele updates ververst het scherm soms met dezelfde informatie. Wanneer dit niet gebeurt zal het telkens 7 seconden schelen dat het apparaat actief bezig is met het verwerken van data.

12 Reflectie

In dit hoofdstuk zal er gereflecteerd worden op het gehele afstudeertraject. Te beginnen bij de aanpak.

12.1 Aanpak

In het begin van het afstudeertraject, was er eigenlijk geen duidelijke aanpak. Vrijwel direct na het onderzoek omtrent messaging- en netwerk protocollen begon ik uit enthousiasme met bouwen van de concepten. Naarmate de opdracht steeds duidelijker werd kwam er meer structuur, daarbij kwam er een duidelijke planning met een duidelijke opzet wat wanneer verwacht werd. Dit schepte een goede structuur en kwam de aanpak ten goedde.

12.2 Planning

Zoals in paragraaf 12.1 begon de planning wat onrustig. Ik had niet helemaal duidelijk wat er nu van me verwacht werd en wat er gebouwd moest gaan worden. De eerste versie van de planning was dan ook heel basaal, en bood eigenlijk niet veel meerwaarde. Naar maten het project vorderde en duidelijk werd wat er verwacht werd, kwam er een nieuwe planning. In deze planning stond duidelijk wat de staat was van het op te leveren product en wanneer het opgeleverd wordt.

De planning werd eens per week doorgenomen met Leon Stam waardoor we beiden een goed beeld kregen wat de staat nu was van het project. Dit schepte veel rust omdat er een duidelijk overzicht was wat wanneer gedaan wordt.

12.3 Resultaten

Er is veel ontwikkeld tijdens het project. Voor school was het verplicht dat de applicatie automatisch kon builden, testen en deployen. Hiervoor is een ontwikkelstraat opgezet. Gezien Alten hier geen omgeving voor bied aan stagiairs moest dit zelf geregeld worden. Hiervoor is mijn privé VPS gebruikt. Echter, wou ik het wel zo opzetten dat opvolgende stagiairs iets hadden aan mijn werk en niet nogmaals de hele ontwikkelomgeving moesten opzetten. Hierom is Docker gebruikt, een grootte leerstap omdat ik nagenoeg nog nooit met Docker heb gewerkt. Hierdoor heb ik ook een aantal dagen tot in de avond en in het weekend gewerkt om de ontwikkelstraat op te zetten, dit was een slechte gewoonte aan het worden gedurende het traject, waarin er vaak in privé tijd nog doorgewerkt werd. Dit is nooit verplicht van uit Alten. Het resultaat van de ontwikkelstraat ben ik dan ook trots op, het werkt goed en alles gebeurt automatisch. Zelfs de unit tests op de ESP32 kunnen automatisch gedraaid worden door een Emulator, iets wat zowel Leon Stam als ik heel mooi vonden.

Het uitkiezen van de hardware viel eigenlijk buiten mijn scope, in een voorgaand project was dit namelijk al vastgelegd, echter, ben is hier tegenin gegaan omdat de gebruikte microcontroller in dat project verouderd. De ESP32 is eigenlijk een directe upgrade is qua snelheid en energiezuinigheid van de ESP8266 die in dat project gebruikt werd. Toen ik dit voorlegde aan Leon Stam en Dennis Karmelk (Technisch manager) waren ze het met mij eens. Aan het begin van het project werd gedacht dat het al bestaande project konden hergebruiken. Eenmaal ingelezen op dat project bleek dat het om een ander product ging en hierbij het Arduino framework werd gebruikt wat nogal inefficiënt is voor onze oplossing, gezien dit allemaal overhead zou bieden en we o.a. de Loop methode van Arduino nooit zouden gebruiken. Dit medegedeeld aan Leon Stam en Dennis Karmelk waren beiden het ermee eens en is er gebruik gemaakt van het door mij gekozen framework en hardware.

Een belangrijke leer les hierin voor mij was dat niet elke microcontroller dezelfde specificaties hebben ondanks dat ze dezelfde chip gebruiken hierdoor valt de batterijduur van het huidige product tegen. Dit is wel gemakkelijk op te lossen door een andere versie van de microcontroller te halen, de code zal op zowel de huidige als de nieuwe microcontroller moeten werken gezien ze dezelfde chip gebruiken. Een belangrijke les met gelukkig niet al te grootte gevolgen.

Het product zelf voldoet aan alle harde eisen. De batterijduur is een tegenvaller maar gemakkelijk op te lossen. Het werken op een Microcontroller was wel even wennen. Op school hebben we nagenoeg nog nooit in C++ geprogrammeerd waarin dit project bijna volledig in C++ geschreven is. Vooral het werken met pointers en het zelf managen van geheugen is iets waar in programmeertalen op hoger niveau geen rekening mee gehouden hoeft te worden. In dit project was dat wel anders, alles moest efficiënt omdat de ESP32 snel in slaapmodus moet gaan, daarbij komt dat het beschikbare geheugen op het apparaat zelf maar beperkt is. Rondzwevende pointers kunnen dus niet gepermitteerd worden. Naarmate het project vorderde werd krijg ik het steeds beter onder de knie, en helpt het gebruik van smart pointers enorm.

Het werken met hardware brengt nog een uitdaging met zich mee, er moet namelijk gezorgd worden dat de verschillende stromen allemaal goed op elkaar aansluiten en er geen kortsluiting gemaakt worden. Een voorbeeld hiervan zijn de batterijen die gebruikt worden. Voor de batterij wordt gebruik gemaakt van drie AA-batterijen zodat deze gemakkelijk te vervangen zijn in het echte product. Eén AA-batterij heeft een spanning van 1,5V drie batterijen in serie geeft dan een spanning van 4,5V. De ESP32 ondersteund alleen 3V, dit moest ik dus onderzoeken tijdens het project en oplossen. Dit is na wat onderzoek gemakkelijk op te lossen door een level converter. Echter, als ik dat onderzoek nooit gepleegd had was de ESP32 dus doorgebrand.

De behuizing is wederom iets wat ik in mijn vrijetijd uit eigen motivatie heb gemaakt gezien dit buiten de scope viel. Dit vond ik vooral leuk om te doen en hierdoor oogt het eindproduct ook een stuk professioneler. Echter, zal Alten zelf niet veel aan de behuizing hebben omdat die niet opgehangen kan worden wat uiteindelijk wel de bedoeling is. De behuizing is wel gemakkelijk aan te passen om hem op te kunnen hangen.

12.4 Ervaring

Zoals in paragraaf 12.4 al duidelijk is geworden was het project eigenlijk een grootte leerweg. Het werken met hardware was nieuw, het opzetten van een ontwikkelstraat was nieuw en het maken van een behuizing hebben is ook nooit op de opleiding geleerd. Ondanks dat heb ik mezelf goed kunnen redden. Leon Stam heeft me vrijwel alleen geholpen met de formules mooi opschrijven en het reviewen van de opgestelde documenten.

12.5 Begeleiding

De begeleiding vanuit Alten was heel goed. Leon Stam reageerde vrijwel direct als ik iets vroeg en was altijd bereid om te helpen. Ook alle andere collega’s waren altijd bereid om te helpen bij bijvoorbeeld het gebruikersonderzoek. Een goede werksfeer is belangrijk omdat dit de productiviteit ten goedden komt.

Hoewel er niet veel begeleiding uit school was zoals afgesproken, kon dat wel beter. Evert Verduin heeft het heel druk en kan hierdoor niet altijd snel reageren. Soms ontlopen er dingen, een voorbeeld hiervan zijn mijn advies formuleren die Evert nooit gezien had ondanks dat ze wel gestuurd waren. Hierdoor ontstond toen zelfs een kleine discussie die verder geen naam mag hebben en we als volwassen hebben opgelost. Het was overigens ook deels aan mijzelf te danken doordat het opsturen van mijn afstudeerplan niet helemaal soepel verliep en de link naar mijn OneDrive ineens verdwenen was.

De feedback die Evert Verduin gegeven heeft op mijn afstudeerplan was overigens wel heel waardevol voor het opstellen van juiste hoofd- en deelvragen.

12.6 Competenties

Voor het afstudeertraject heb ik beoogd een zevental competenties te bewijzen. Hiervan zijn de vier A-Competenties verplicht voor school en de drie B-competenties gekozen n.a.v. de afstudeeropdracht. In dit hoofdstuk zal er gereflecteerd worden op de competenties te beginnen bij de A-competenties.

12.6.1 Leren leren

De competentie leren leren heeft betrekking op het continu proces. Het continu zoeken naar nieuwe betere methodes, het verbeteren van het gemaakte werk en het zelfstandig problemen op kunnen lossen.

Ik ben van mening dat deze competentie van nature naar voren komt, Iedereen zal continu bezig moeten zijn met het verbeteren van zijn of haar functioneren. De opdracht vanuit Alten was hier perfect voor, op de opleiding hebben we nagenoeg geen embedded ervaring opgedaan en dit betreft voor een groot deel een embedded opdracht. Het werken met C/C++ was in zijn geheel nieuw en hoewel het zelf regelen van het geheugen en het juist opruimen hiervan een uitdaging is. Geeft het wel extra veel voldoening als de geschreven code fysiek de resultaten laat zien, dit maakt het tastbaar.

Hoewel ik aan het begin van de opdracht wat onzeker was; ik had geen goed beeld van wat ik moest onderzoeken en embedded programmeren was toch spannend. Hierbij was het ook belangrijk dat ik het e.e.a. afwist hoe stroom werkt, ik wou niet degene zijn die de spullen van Alten sloopt doordat ik de kabels verkeerd aansluit. Desondanks is de zelfzekerheid met de tijd goed, ik kreeg een goed beeld wat er onderzocht moest worden en wat er verwacht werd. Het programmeren in C++ kreeg ik redelijk vlot onder de knie. Het werken met stromen en het juist aansluiten van alles kwam ging gemakkelijker en op dit vlak heb ik zeker veel kennis opgedaan van Leon Stam.

Zoals Leon Stam ook aangeeft in het nawoord heb ik mezelf goed kunnen redden en is er een mooi product opgeleverd waar ik trots op ben.

12.6.2 Onderzoek

Voor het onderzoek naar netwerk- en messaging protocollen was ik aan het begin wat onzeker. Het betreft een onderwerp waar ik niet veel vanaf wist en daarnaast was het me nog niet goed duidelijk wat ik nu ging onderzoeken. Gelukkig kwam dit goed na een aantal gesprekken met Dennis Karmelk (de technisch manager) en Leon Stam. Leon heeft me ook goed geholpen tijdens het onderzoek door altijd kritisch te zijn op het geschreven werk. Hierbij is de input van Leon enorm nuttig geweest met het juist opstellen van de hoofd- en deelvragen, en de formules goed op te schrijven/te interpreteren.

Beiden onderzoeken zijn naar mijn mening enorm nuttig geweest en hebben de basis gevormd voor het opgeleverde product. Het onderzoek naar netwerk- en messaging protocollen heeft hierin de basis gelegd voor de communicatie tussen de ESP32 en de server waarbij het gebruikers onderzoek de basis heeft gelegd voor de opgestelde requirements en het prioriteren van het werk.

12.6.3 Professioneel werken

De competentie professioneel heeft betrekking op mijn houding als stagiair, hoe communiceer ik met Leon en de overige medewerkers, ben ik punctueel en werk ik planmatig. Wat betreft het communicatie gedeelte komt dit redelijk vanzelf, ik ben niet mensenschuw en ook niet bang om kritiek te krijgen, hier leren we tot slot alleen maar van. Het actief opzoeken van feedback is dan ook terug te zien in ieder document waarin in de documenthistorie te zien is dat Leon zijn feedback verwerkt is.

Het planmatig werken heb ik echter wel moeite mee. Van nature ben ik redelijk chaotisch en daarbovenop verlies ik gemakkelijk me aandacht en redelijk impulsief. Ik kan op het ene moment aan het documenteren zijn en ineens bedenken dat er toch een stukje geprogrammeerd gaat worden. Dit gaat natuurlijk niet geheel samen met planmatig en methodisch werken.

Aan deze slechte gewoontes heb ik gewerkt gedurende het afstudeertraject. Door een hele duidelijk en concrete planning op te stellen, iedere week met Leon te bespreken wat ik volgende week ga doen en wat er volgens de planning gedaan wordt. Dit heeft geholpen met het methodisch en planmatig werken doordat dit een stok achter de rug gaf, Leon verwacht tot slot iets van me aan het eind van de week en als ik maar wat loop te klooien gaat dat nooit lukken.

12.6.4 Innovatie

Met deze competentie heb ik moeite gehad. Hoe ga ik nu aantonen dat er out of the box is gedacht en dat ik niet bang ben om tegen de gevestigde orde in te gaan. Wat als de gevestigde orde prima is en er niet tegenin hoeft worden gegaan.

Toch ben ik van mening dat deze competentie gelukt is. Mede doordat door mijn input de gekozen hardware is veranderd, we zijn van een ESP8266 naar een ESP32 en het eerder gebruikte Arduino framework is omgeruild voor en ESP-IDF framework, het framework waarvoor zowel de ESP32 en ESP8266 ontworpen zijn. Dit is beiden naar aanleiding van mijn input gedaan en op die manier de “gevestigde orde“ doorbroken is.

Het betreft het out of the box denken denk ik dat het gebruik van QEMU een mooi voorbeeld hiervan is. Normaliter kan een embedded apparaat niet automatisch getest worden door de ontwikkelstraat doordat deze fysiek de code geflasht moet krijgen. Door het gebruik van QEMU kan dit wel automatisch gebeuren door de ontwikkelstraat doordat de code geflasht kan worden op de emulator en de functionaliteiten van de ESP32 gesimuleerd worden. Persoonlijk vind ik dit een erg slimme oplossing en ben ik trots dat het gelukt is om dit werkend te krijgen in Docker en de ontwikkelstraat omdat dit toch een fysieke handeling scheelt en hierdoor ook tijd scheelt.

12.6.5 Software analyseren

Aan het begin van het traject had ik niet echt een goed beeld van de opdracht. Er werd gedacht dat een al geschreven project gebruikt kon worden als basis. Nadat er ingelezen was over dit project bleek het een dermate verschillend product dat dit toch niet besloten was het te maken product vanaf scratch geschreven zou worden.

Naar aanleiding van deze inzichten konden er al een aantal requirements opgesteld worden. Maar dit was nog lang niet genoeg om de hele opdracht te omvangen. Hiervoor heb ik uiteindelijk besloten dat een gebruikersonderzoek nodig was om een juist beeld te krijgen van de verwachtingen en alles goed kort te sluiten. Het gebruikersonderzoek was extreem waardevol en vormen de basis van de gestelde criteria.

12.6.6 Software ontwerpen

Deze competentie begon wat moeizaam, was moet ik nu schrijven in het software architectuur document, hoe begin je eraan en wat is de basis. Daarbij was ik uit enthousiasme al de prototypes gaan bouwen wat anzich niet heel erg is, dit bewees namelijk al in een vroeg stadia dat de gekozen methodieken kunnen werken en de communicatie tussen de ESP32 en de server goed verloopt.

Echter, zat ik nog steeds met het probleem wat er geschreven moest worden in het software architectuur document. De requirements waren wel al goed in beeld en ik had een idee hoe de software opgezet ging worden maar het opschrijven van mijn gedachten vond ik lastig. Uiteindelijk kwam op bij de 4+1 model doordat deze al vaker gebruikt was bij het maken van schoolprojecten en vormde dit een goede basis voor het logisch opschrijven van mijn gedachten en het op de juiste volgorde uitwerken van de schema's.

12.6.7 Realiseren

Zoals verteld was het realiseren wel een spannend moment. Alles was goed doorgedacht maar nu moest het nog uitgevoerd worden. Geen gemakkelijk taak zeker niet als we in beschouwing nemen dat ik nagenoeg geen embedded ervaring had, had wel eens wat geknutseld in Arduino maar hierbij wordt toch je hand nog een beetje vastgehouden. Bij dit project was dat niet het geval, en het werken met pointers en referenties was aan het begin een uitdaging, ik wist niet goed wanneer je welke gebruikte en de syntax was compleet nieuw. Naarmate het project vorderde kwam dit vanzelf goed.

De Redmine omgeving die Alten beschikbaar stelt heeft slechts een zeer summiere Git integratie en geen mogelijkheid tot het automatisch builden, testen en deployen. Hierdoor heb ik besloten mijn eigen ontwikkelstraat op te zetten maar wel op een manier dat volgende stagiairs de omgeving gemakkelijk konden nabootsen. Om dit te doen kom je al vrij snel uit op een Docker omgeving gezien dit scripts zijn die in hun eigen virtuele OS draaien. Hierdoor kunnen de volgende stagiairs deze scripts uitvoeren en zal de omgeving vrijwel identiek zijn.

Tijdens het testen van de batterij kwam wel naar boven dat ik helaas niet het juiste ontwikkelbord gekozen had en hierdoor de batterijduur tegenvalt. Gelukkig is dit gemakkelijk op te lossen door een ander ontwikkelbord aan te schaffen. Hierdoor zal de batterijduur naar waarschijnlijkheid wel binnen de gestelde eisen vallen.

12.7 Conclusie

Al met al kijk ik heel positief terug op het afstudeertraject bij Alten, de begeleiding was top en het was een erg mooi en leerzaam traject. Tijdens het traject heeft Alten ook een baan aangeboden deze heb ik natuurlijk geaccepteerd.

De evaluatieformulieren die Leon Stam heeft opgesteld, het nawoord en het feit dat ik een baan aangeboden heb gekregen als technisch consultant spreken naar mijn mening boekdelen wat Alten van mij als Afstudeerder vond.

Per 1 januari begin zal ik in dienst treden bij Alten als Technisch Consultant.

13 Nawoord

Hieronder volgt een nawoord van Leon Stam, de afstudeerbegeleider vanuit Alten gedurende het afstudeertraject.

Het begeleiden van het stageproject van Mike van Es heb ik als erg positief ervaren. De samenwerking met Mike liep goed. Daarnaast heb ik het zelf ook als een leerzaam proces ervaren.

Het projectplan van Mike is gedurende de hele stageperiode goed verlopen. Er is hard gewerkt om zelfs extra functionaliteit toe te voegen. Ik ben ook van mening dat Mike hier veel heeft geleerd door werken met embedded apparaten. Waarbij goed te zien was dat Mike in staat is zelf te onderzoeken en software te ontwikkelen. Terwijl het schijven van embedded software geen onderdeel is geweest van zijn opleiding.

Het is jammer dat het niet gelukt is om de gewenste batterij duur te behalen. Toch is er een stabiel en net stuk software opgeleverd. Ook is een advies uitgebracht waar we de resultaten wel mee kunnen halen. Dus voor Alten is het project geslaagd.

Leon Stam

Oud-Beijerland, september 2022

14 Bijlagen

Bijlagen	Naam	Type
A	Acceptatie plan	PDF
B	Test plan	PDF
C	Gebruikers onderzoek	PDF
D	Onderzoek netwerk- en messaging protocollen	PDF
E	Planning	XLSX
F	Functioneel ontwerp	PDF
G	Afstudeerplan	PDF
H	Software architectuur	PDF
I	Prototypes	ZIP
J	Docker-files	ZIP
K	Test rapport	PDF
L	Broncode	ZIP
M	Tussentijdse beoordeling	PDF
H	Eindevaluatie Bedrijfsbegeleider	PDF

15 Bibliografie

Awati, R. (sd). *What is an ampere hour (Ah or amp hour)?* Opgehaald van techtarget: https://www.techtarget.com/whatis/definition/ampere-hour-Ah-or-amp-hour

Bijvank, S. O. (sd). *Watervalmodel / V-model*. Opgehaald van house-of-control.

Day, L. (2020, 07 06). *LoRaWAN vs Zigbee – Which Wireless IoT Network is the best for me?* Opgeroepen op 05 30, 2022, van LinkedIn: https://www.linkedin.com/pulse/lorawan-vs-zigbee-which-wireless-iot-protocol-best-me-luke-day/

Dieter Coppens, E. D. (2022, 02 04). *An Overview of Ultra-WideBand (UWB) Standards(IEEE 802.15.4, FiRa, Apple): Interoperability Aspects and Future Research Directions*. Opgeroepen op 05 24, 2022, van arxiv: https://arxiv.org/abs/2202.02190

Drone - Automate Software Build and Testing. (sd). Opgehaald van drone.io: https://www.drone.io/

Eduardo Garcia-Espinosa, O. L.-G. (2018, 12). *Power Consumption Analysis of Bluetooth Low Energy Commercial Products and Their Implications for IoT Applications*. Opgeroepen op 05 2022, 30, van Researchgate: https://www.researchgate.net/publication/329401909_Power_Consumption_Analysis_of_Bluetooth_Low_Energy_Commercial_Products_and_Their_Implications_for_IoT_Applications

Eichhorn, D. (2021, 07 17). *ESP32 – Ultra-Long Battery Life With ESP-NOW*. Opgehaald van Thingpulse: https://thingpulse.com/esp32-ultra-long-battery-life-with-espnow/

Gitea - Git with a cup of tea. (sd). Opgehaald van gitea: https://gitea.io/en-us/

Guide to Reduce the ESP32 Power Consumption by 95%. (sd). Opgehaald van diyi0t: https://diyi0t.com/reduce-the-esp32-power-consumption/

How to Run Your ESP8266 for Years on a Battery. (2022, 09 09). Opgehaald van makecademy: https://makecademy.com/esp8266-battery

Insight Into ESP32 Sleep Modes & Their Power Consumption. (sd). Opgehaald van lastminuteengineers: https://lastminuteengineers.com/esp32-sleep-modes-power-consumption/

Karunakar Pothuganti, A. C. (2014, 09). *A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi*. Opgeroepen op 05 30, 2022, van researchgate: https://www.researchgate.net/publication/312471356_A_comparative_study_of_wireless_protocols_Bluetooth_UWB_ZigBee_and_Wi-Fi

Olaide O. Kazeem, O. A. (2017, 05). *Comparative Study of Communication Interfaces for Sensors and Actuators in the Cloud of Internet of Things*. Opgehaald van researchgate: https://www.researchgate.net/publication/318054538_Comparative_Study_of_Communication_Interfaces_for_Sensors_and_Actuators_in_the_Cloud_of_Internet_of_Things#pf3

QEMU - A generic and open source machine emulator and virtualizer. (sd). Opgehaald van QEMU: https://www.qemu.org/

SDLC - Waterfall Model. (sd). Opgehaald van tutorialspoint: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm

Stakeholdersanalyse. (sd). Opgehaald van projectmanagementsite: https://projectmanagementsite.nl/stakeholdersanalyse/

The curious case of lorawan devices and coin cell batteries. (2022, 05 30). Opgehaald van thethingsnetwork: https://www.thethingsnetwork.org/article/the-curious-case-of-lorawan-devices-and-coin-cell-batteries

What are LoRa and LoRaWAN? (2022, 05 24). Opgehaald van theThingsNetwork: https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan/