

Projekt IoT 1

Tobias Ditlev Rasmussen,

Semester 1

Klasse A

Gruppe 2

Project CrashLiteX

Vejledere:

Malene Hasse

Natasha Fotel

Kevin Lindemark Holm

Kenneth Sandstrøm

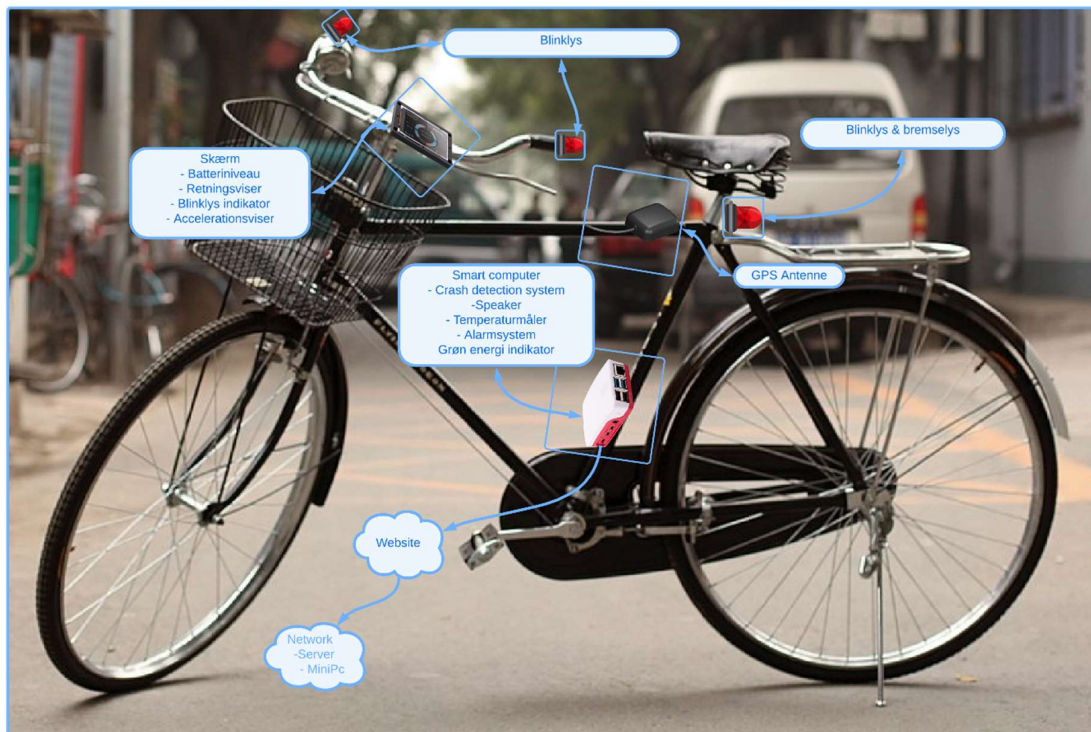
Tahseen Uddin

Eva Sjuve

75.929 Anslag

Project CrashLiteX





Figur 1: Ikke teknisk overblik over løsning

1. Resume

I dette projekt har vi arbejdet med en problemstilling omkring usikkerhed i trafikken.

Vi har sat sikkerhed i fokus eftersom cykelulykker er stigende, bl.a. på elcykler.

Vores tilgang til dette, var at lave et produkt eller prototype, hvor folk føler sig mere trygge/sikre i trafikken og give en mere strømlinet og direkte hjælp, hvis nu uheldet skulle være ude.

For at opnå dette, har vi sat en idé i værk som består af bremselys, crash-detection og blinklys. Derfor er brugeren i fokus, som kan benytte sig af retnings visning og andre informationer, uden at miste balancen, samt blinklysets muligheder i mørke sammenhænge, hvor en hånd måske ikke kan ses.

Dette er udarbejdet ved hjælp af en proces af design, både i kode, netværk og fysisk design.

Igenem rapporten viser vi fra fag til fag hvilke teorier og teknikker vi har brugt for at løse vores problemer og nå frem til et fysisk produkt.

Som der kan læses i konklusionen, så nåede vi i mål med vores vision, dog med små afskæringer, samt andre småproblemer, så en fuld funktionel helheds-prototype nåede ikke at få liv, men alle krav blev helt eller delvist opfyldt og testet med succes.

2. Indholdsfortegnelse

1. Resume	4
2. Indholdsfortegnelse.....	5
3. Indledning.....	6
3.1. Beskrivelse af virksomhed	6
4. Problemformulering	7
4.1. Afgrænsning	7
5. Indledende undersøgelse	8
5.1. Ideudvikling	9
5.2. Research	10
5.3. Brugerundersøgelse	10
5.4 Business Model Canvas	14
6. Indlejrede Systemer.....	15
6.1. Blokdigrammer	15
6.2. Diagrammer.....	17
6.3. Beregninger og målinger	21
6.4. Teori af relevante elektroniske blokke.....	23
7. Programmering.....	28
7.1. Flow charts	28
7.2. Kodebeskrivelse	39
7.3. Teori af relevante kodeblokke	49
8. Netværk.....	53
8.1. Router/switch interfaceforbindelser	55
8.2. Konfiguration af valgte servere	57
8.3. Teori af relevante netværk og server-blokke.....	66
9. Egenskaber, krav, testmetode og resultater	71
9.1. Krav, prioritet, accepttestprocedure og resultater	71
10. Implementering af løsningen i drift.....	92
11. Praktisk projektplanlægning og -ledelse	93
11.1. WBS	94
11.2. Gantt.....	95
11.3. Kanban	96
11.4. Projektanalyse	97
12. Konklusion	101
13. Projektforløbet	102
14. Perspektivering.....	103
15. Litteraturliste	104

3. Indledning

Vi har fået til opgave at lave en enhed, der kan omdanne en normal cykel til en smart-cykel. I forbindelse med denne opgave har vi, i ånd af kommende opgaver, valgt at lade som om vi har taget samarbejde med virksomheden Swytch.

Dade i deres løsning allerede har konverteret en cykel til elcykel, tager vi udgangspunkt i en udvidelse af deres katalog og laver et produkt der vil hjælpe cyklisteres sikkerhed i trafikken. Dette er valgt da ulykker på elcykel og cykler er stigende.¹

Vores tanke for at udrette dette, er at give cykler blinklys, samt en ulykkesdetektor, der når den udløses sender signal til pårørende og alarmcentralen med data, inkl. GPS. Udover det gives der også bremselys, samt display med diverse informationer fra cyklen og et alarmsystem, der kan slås til og fra, som underretter brugeren hvis cyklen bliver rykket på imens.

3.1. Beskrivelse af virksomhed

Swytch er en innovativ virksomhed baseret i London, de har til mission at gøre grøn transport tilgængelig for alle, i en tid med øget fokus på klimaforandringer og grøn transport.

De har udviklet et eftermonterbart kit der forvandler en traditionel cykel til en elcykel.

Kittet inkluderer en forhjulsmotor, et batteri der kan monteres enten på cyklens styr eller stel, en styringsenhed og en pedalsensor. Kittet er designet til at være nemt at installere, så enhver cyklejer kan opgradere sin cykel uden teknisk ekspertise.

Swytch blev lanceret i 2017 ved hjælp af en succesfuld crowdfunding-kampagne, og har allerede opnået global anerkendelse med over 80.000 loyale "Swtchere" (kunder) i over 100 lande der tilsammen har kørt over 15 millioner kilometer.

Swytch har indtil videre udviklet 4 generationer og tilbyder pt. to forskellige modeller, "GO" og "AIR" der har hver deres fordele, GO-modellen tilbyder mere kraft og længere rækkevidde, mens AIR-modellen er ultralet og designet til maksimal bærbarhed.

Swytch får gode anmeldelser af deres kunder (4.8 stjerner ud af 5 på REVIEWS.io med over 1500 anmeldelser) og har fået positiv omtale i flere prominente medier, herunder; The Telegraph, The Washington Post, CNN, Sky News, Forbes mm.

Swytch fortsætter med at innovere inden for grøn transport og viser, hvordan teknologi kan bidrage til en øget klimabevidsthed og derved skabe en grønnere fremtid.

¹ <https://www.dst.dk/da/Statistik/nyheder-analyser-publ/nyt/NytHtml?cid=49296>

4. Problemformulering

Hvad er problemet?

Der er sket en stigning af færdselsuheld med cyklister involveret, hvilket resulterer i en stigning i dødsfald og tilskadekomne².

Hvorfor er det et problem?

Det koster samfundet omkring 150 mio. Kr. om året for "langvarig pleje om omsorg for trafikofrene".³

For hvem er det et problem?

Det er et problem for både cyklister og samfundet. For cyklisterne har det både helbredsmæssige og økonomiske konsekvenser. For samfundet har det økonomiske konsekvenser.

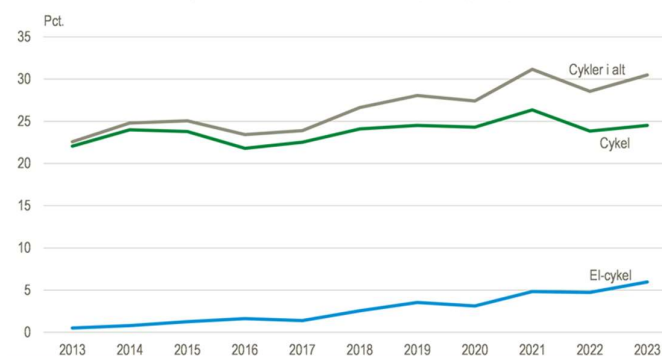
Hvordan vil man kunne løse problemet på teknisk vis?

Vi ønsker at løse problemet igennem en teknisk løsning som indebærer blinklys på cyklen for derved at skabe større synlighed af cyklisten, samt en crash detection (ulykke registrering) der tænder alle 4 blinklys (havari-blink) på cyklen som synliggør ulykke stedet og sender tidspunkt og GPS-lokation til ThingsBoard (og gemme data i en fil på, i dette tilfælde, mini-pc'en) samt en præ-defineret besked til ens nærmeste pårørende med GPS-koordinater.

Forventet udbytte?

Vi forventer at se et fald i rapporterede uheld med elcykler (og cyklister generelt), øge sikkerheden i trafikken, (og forhåbentlig) indføre nye industristandarder.

Andel tilskadekomne og dræbte i færdselsuheld på cykel og el-cykel



Figur 2: Graf over antallet af cykelulykker over årene

4.1. Afgrænsning

Vi har valgt at skære tilbage på vores ide, primært med at få det hele ind under et system. Det var bare ikke muligt i den tidsramme vi havde at arbejde i.

Windows Thingsboard blev installeret, men ikke testet da installation af klienter ikke var brugbart, men med en udvidet tidsramme, ville det ikke have været en problematik.

² <https://www.dst.dk/da/Statistik/nyheder-analyser-publ/nyt/NytHtml?cid=49296>

³ <https://ing.dk/artikel/antallet-af-trafikdraebte-og-tilskadekomne-i-koebenhavn-stiger-med-30-procent>

5. Indledende undersøgelse

Ideudvikling

Ideudvikling handler om, hvordan man finder frem til den idé, der skal være kernen i projektet. Der er fire faser til idéudvikling. Det er vigtigt at have en beskrivelse af, hvordan idéen opstod. Endnu en vigtig del er at dokumentere eventuelle metoder brugt i brainstormingen, fx. mindmap. Endnu et element er argumentation for valget af den endelige idé og dens relevans.

Research

Research på basis niveau handler om at undersøge relevante teknologier og værktøjer, som kan være med til at understøtte projektet. Udover det, så indebærer det også (muligvis) analyse af eksisterende løsninger. Man skal have i minde når det kommer til en teknisk løsning at eventuelle lovmæssige krav, skal tages i betragtning.

Brugerundersøgelse

Brugerundersøgelse (både kvantitativ og kvalitative) handler om at identificere den udvalgte målgruppe og deres mulige behov. Metoder til dataindsamling, fx. interviews, spørgeskemaer eller observationer er alle sammen relevant. Opsummering af resultaterne fra undersøgelserne og hvordan de kan påvirke projektet.

5.1. Ideudvikling

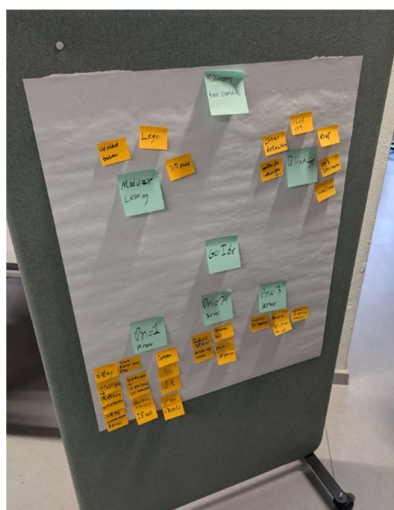
Ideudviklingen startede med en brainstorm indenfor rammerne af vores problemformulering. Den satte vi hurtigt over i et mindmap, for at kunne organisere vores ideer til de forskellige krav der skulle opfyldes, samt sætte en forbindelse mellem ideerne op. I den forbindelse begyndte vi også allerede der, at skære nogle af de ideer fra, som vi kunne se ikke vil virke, som ikke var inden for vores egne evner eller vurderet til ikke at kunne nås til vores deadline.

På vores mindmap satte vi vores krav op, for at hægte de ting på vi skulle have med, samt forskellige "overpunkter" til mindre ideer, for bedre at kunne se hvordan det hele ville hægte sig sammen med hinanden. Den blev også brugt til at sætte en proces op for hvordan vi evt. skulle producere vores prototype, så vi ville kunne få noget funktionelt at præsentere, samt få en ide om hvor og hvordan den fysisk skulle sidde på cyklen.

Både vores tekniske løsning blev tænkt over i sin egen bane og vores generelle ide til hvordan systemet skulle fungere er også inde på mindmappet.

Det kan være at vores mindmap ikke blev helt som forventet, den blev måske mere brugt som en slags "to-do" liste, til at give overblik over hvad vores ide skulle indeholde.

Derudover endte vi også med at have en "tavle" brugt til ideer der løbende dukkede op, samt huskeliste for ideer til senere brug. Begge figurer nedenfor [Figur 3 & Figur 4] er billeder fra vores førnævnte fra forskellige stadier i processen.



Figur 4: Tidlig udgave af vores mindmap



Figur 3: Huske/Ideboard

5.2. Research

Med udgangspunkt i vores Mindmap, som vi tidligere havde brugt til idéudvikling, undersøgte vi forskellige virksomheder for at få en bedre ide om, hvad der præcist var tilgængeligt i den "virkelige" verden. Vi tog udgangspunkt i en virksomhed ved navn Swytch. Deres vision om at gøre cykler til "smart cykler" igennem deres "upgrade kits" er en idé som vi så løsninger i! ⁴ Efter researchen tog vi en snak i gruppen om hvordan vi kunne forbedre eller udvikle Swytch nuværende sortiment.

Én ting som gik igen i vores samtaler var sikkerheden, derfor spurgte vi os selv, hvad vi præcist kunne gøre som team for at øge sikkerheden og hvordan vi kunne gøre det? Vi begyndte at undersøge statistikker på forskellige cykel relaterede ulykker og vi fandt også frem til flere artikler om dette emne, de er nu en del af vores problemformulering. Vi overvejede i hvilke situationer cyklister ofte kan ende i ulykker: Ifølge Københavns Kommune er en stor del af de tilskadekomne involverede højresvingsulykker.⁵ Med dette i mente, overvejede vi hvilke muligheder der var for at gøre cyklister mere synlig for andre i trafikken. Som man også kan se på vores Mindmap, var blinklys i vores tanker helt fra start og tanken blev understøttet af en artikel fra sikkertrafik, omhandlende højresvingsulykker.

Den anden del af vores research gik på, hvad hvis man er udenfor storbyen, hvor man måske kan opleve en hit & run situation? Vi tænkte efter at have læst flere artikler, som denne fra Politiken⁶, at hvis man som cyklist bliver ramt af en i trafikken, er der ikke mange til at hjælpe. Vi kom derfor op med vores crash detection system! Med dette system kan man altid være sikker på at ens pårørende, via vores automatiske system, vil få en besked hvis der er sket ulykke.

Alle disse aspekter gjorde at vi ønskede at tage hånd om problemet og forhåbentlig gøre en forskel.

5.3. Brugerundersøgelse

Vi endte med at bruge en kvantitativ brugerundersøgelse. I denne type vil man typisk se "lukkede spørgsmål" som respondenterne kan besvare ved at trykke på allerede defineret svar kategorier. Brugen af en kvantitativ brugerundersøgelse og de "lukkede spørgsmål" har diverse fordele og ulemper. Fordelene kan typisk være en hurtig måde at be- eller afkræfte hypoteser, mulighed for at generalisere resultaterne til en større gruppe af mennesker, plus, den data du opnår igennem denne type, er objektiv og nemt at præsentere (grafisk, diagrammer, etc.). Ulemperne kan typisk være at den kvantitativ brugerundersøgelse ikke har "dybde" nok og respondenterne ikke har mulighed for at uddybe eller nuancere deres svar. Denne type undersøgelse kræver generelt en stor del data, hvis man ønsker pålidelige resultater, plus, da man har "lukkede spørgsmål" som nævnt tidligere, vil man have en risiko for fejlfortolkning, hvis ens spørgsmål ikke er lavet ordentlige.

I dette specifikke scenarie, kigger vi som sagt på en problemstilling som involverer mange mennesker i Danmark: Trafikken er et farligt sted for mange, specielt cyklister.

Igennem vores brugerundersøgelse og vores "lukkede spørgsmål" har vi opnået muligheden for at få

⁴ <https://www.swytchbike.com/about-us/>

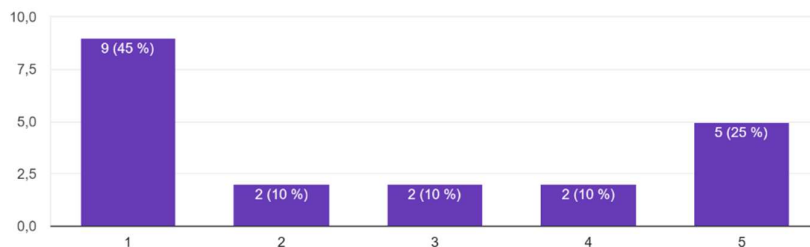
⁵ <https://ing.dk/artikel/antallet-af-trafikdraebte-og-tilskadekomne-i-koebenhavn-stiger-med-30-procent>

⁶ https://politiken.dk/danmark/art10198565/Ung-kvinde-er-d%C3%B8d-efter-p%C3%A5k%C3%B8rsel-af-formodet-spritbilist?srltid=AfmBOor06qQ8V4WksqqvD_1y8WY_zxnpEo64S_T-w9zBnmhrbs_2hBWD

et glimt (lille) indblik i de brugere, som vi har fået svar fra. Som sagt, så er den lille mængde data ikke helt retvisende, men stadigvæk nok til at give os en idé om hvad de adspurgte tænker om vores idé.

Hvor ofte cykler du til arbejde/skole? 1 = Aldrig 5 = Hverdag

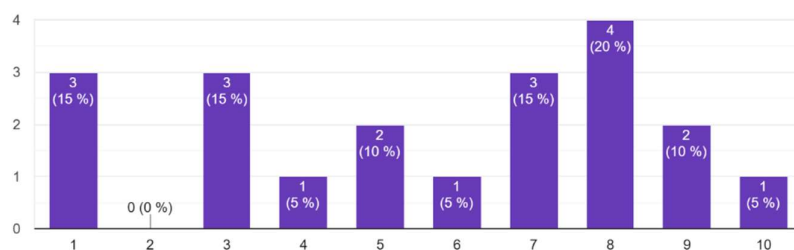
20 svar



Figur 5: Svar fra første spørgsmål fra spørgeskema

Hvor tryk føler du dig i trafikken, som cyklist? 1 = Ekstrem usikker 5 = Hverken usikker eller sikker 10 = Ekstrem sikker

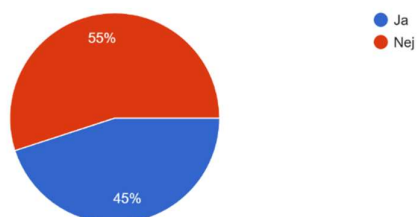
20 svar



Figur 6: Svar fra andet spørgsmål fra spørgeskema

Har du været ude for uheld, som cyklist?

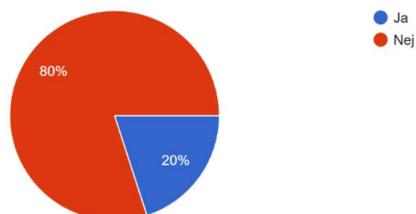
20 svar



Figur 7: Svar fra tredje spørgsmål fra spørgeskema

Har du allerede nogen form for sikkerheds løsninger monteret på din cykel?

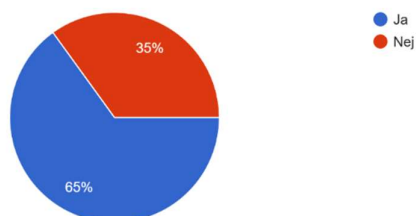
20 svar



Figur 8: Svar fra fjerde spørgsmål fra spørgeskema

Ville du føle dig mere sikker i trafikken, som cyklist, hvis du havde blinklys monteret på din cykel?

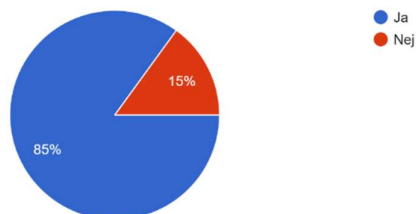
20 svar



Figur 9: Svar fra femte spørgsmål fra spørgeskema

Ville du ønske du havde et crash detection system til at sende en besked til udvalgte pårørende, hvis du var ude for et uheld på din cykel?

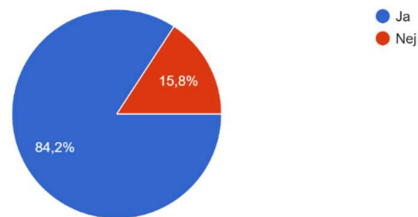
20 svar



Figur 10: Svar fra sjette spørgsmål fra spørgeskema

Ville du føle dig mere sikker i trafikken, som cyklist, hvis du havde et bremselys monteret på din cykel?

19 svar



Figur 11: Svar fra syvende spørgsmål fra spørgeskema

Vores kvantitative brugerundersøgelse giver os et tydeligt indblik i at vores problemstilling og løsningen vil være til gavn for en stor mængde mennesker, der færdes i trafikken. Et nærmere kig på specifikke dele af brugerundersøgelsen viser os at: 45% af vores respondenter havde, som cyklist, været ude for et uheld i trafikken. Dette er repræsentativt i forhold til vores problemstilling hvor, ifølge statistikbanken.dk udgør cyklister over 30% af uheld⁷ og ifølge Københavns Kommune udgør cyklister op i mod 60% af de alvorlige tilskadekomne i trafikken⁸. Vi mener derfor at vores sikkerheds løsning vil være med til at skabe en positiv ændring. Ifølge vores brugerundersøgelse kan vi se at 80% af vores respondenter, ikke på nuværende tidspunkt havde en sikkerhedsløsning monteret på deres cykel. Alt i alt mener vi, at svarene fra brugerundersøgelsen underbygger vores tidligere nævnte problemstilling og vores ønske om at skabe en positiv ændring i trafikken med CrashLiteX.

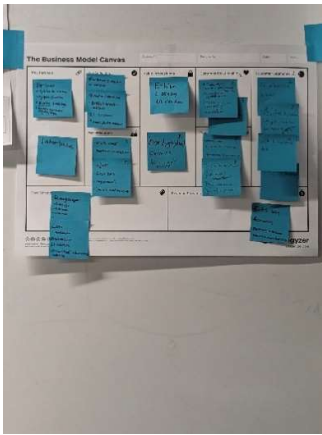
Data fra brugerundersøgelsen viser os hvilke features vi kan fremhæve ved CrashLiteX: Blinklys, crash detection og bremselys. Ifølge vores brugerundersøgelse havde 65% af vores respondere en positiv tilgang til blinklys og hvorvidt blinklysene ville få dem til at føle sig mere sikre i trafikken. Endnu et kig på dataene viser, at 85% af respondenterne havde en positiv tilgang til ideen om et crash detection system ville være med til at øge sikkerheden for dem og 84,2% svarede ja spørgsmålet om hvorvidt de ville føle sig mere sikre i trafikken, hvis de havde bremselys monteret på deres cykel.^[5]

Vi mener derfor at der er et ønske/ mangel / interesse for en løsning som vores og derfor et muligt marked, der valideret vores projekt, CrashLiteX og dermed være med til at skabe en positiv ændring i trafikken for cyklister (og trafiksikkerheden som helhed), og i sidste ende forhåbentlig nedsætte mængden af uheld generelt, samt være forløbere for nye industristandarder.

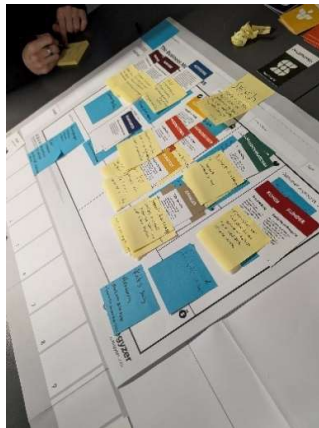
⁷ <https://www.dst.dk/da/Statistik/nyheder-analyser-publ/Nyt/NytHtml?cid=49296>

⁸ <https://ing.dk/artikel/antallet-af-trafikdraebte-og-tilskadekomne-i-koebenhavn-stiger-med-30-procent>

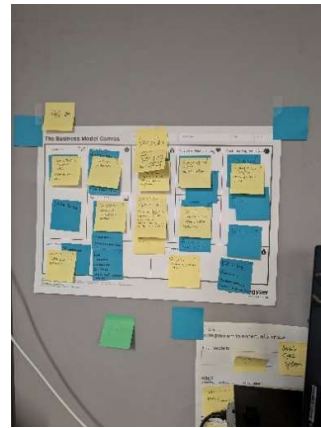
5.4 Business Model Canvas



Figur 12: Tidlig udgave af BMC



Figur 13: BMC i gang med spil



Figur 14: BMC efter spil

Business Model Canvas er en metode til beskrive/udvikle en forretningsmodel enten til en virksomhed specifikt, eller bare et projekt. Det er en forudsætning at man har enten et koncept til en virksomhed (eller) projekt som man ønsker at lave en forretningsmodel for. BMC indeholder 9 forskellige byggesten (ikke i specifik rækkefølge):

Value Propositions:

Dette indeholder de produkter/services som bliver tilbudt, for at opfylde kunders behov.

Customer Segments:

Dette er til at finde ud af hvem kunderne er.

Key Partners:

Dette er med til at identificere hvilke partner- og leverandører som er tilknyttet.

Key Activities:

Dette beskriver de vigtigste nøgleaktiviteter for at skabe værdi til kunder.

Key Resources:

Denne del kigger på de ressourcer som er nødvendige for at skabe værdi for kunderne.

Customer Relationship:

Denne del kigger på det type forhold man ønsker at have til sine kunder.

Channels:

Dette er hvilke distributions-/kommunikationskanaler som er nødvendige for at nå kunderne.

Revenue Streams:

Dette dækker over hvilke type indtægtskilder, som er mulige.

Cost Structure:

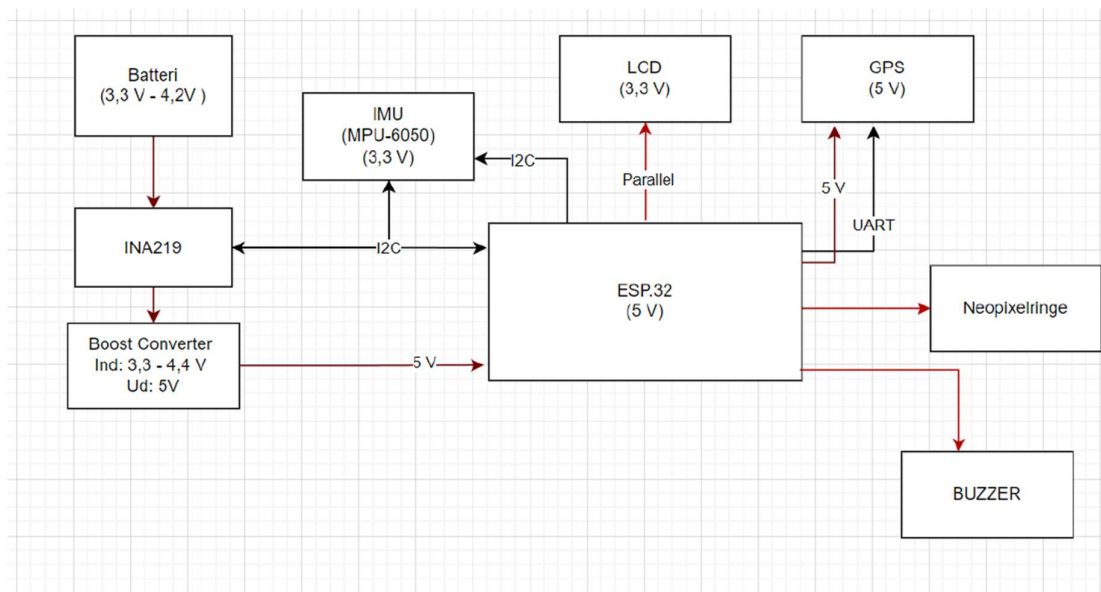
Denne del dækker over hvilke omkostninger der er.

I Gruppe 2, 1A har vi brugt BMC som et vigtigt redskab i vores indledende fase af projektet. Det gav os mulighed for at reflektere på de aspekter som indgår i en virksomhed og derefter bruge Flip Model Canvas til at kigge fra vores synspunkt og hvordan vi kunne levere et nyt produkt. Med dette i vores hænder var det muligt at skabe et design spørgsmål og efterfølgende de resterende dele af projektets indledende faser.

6. Indlejrede Systemer

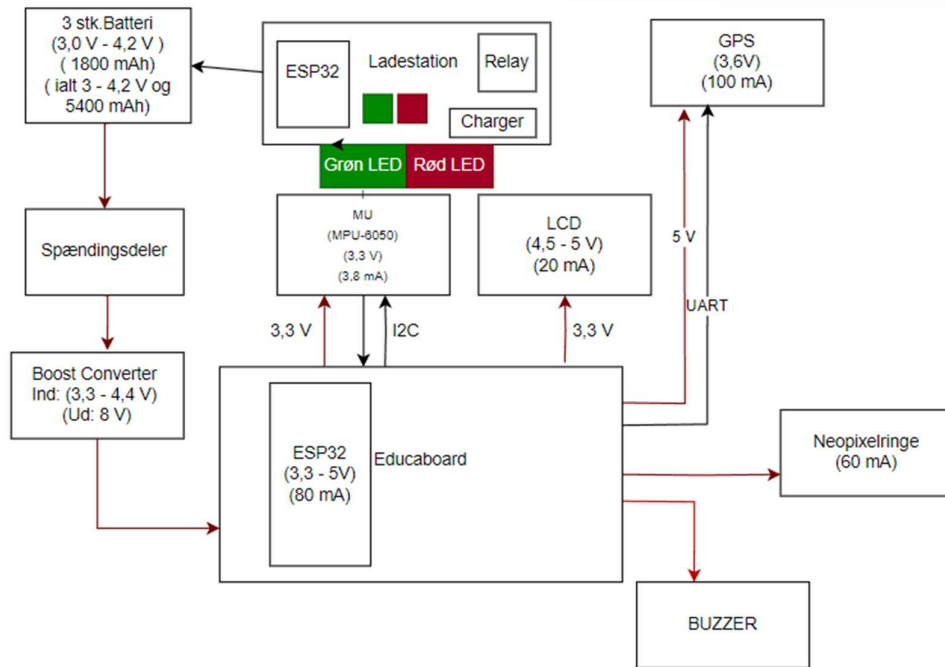
I projektet valgte vi i forhold til den indlejrede del at tage udgangspunkt i Educaboardet og bruge de muligheder og løsninger, protokoller og teknologier der var via det. Vi har arbejdet ud fra den viden vi har fået gennem undervisningen og prøvet at lave løsninger baseret på denne viden.

6.1. Blokdiagrammer



Figur 15: Første version af blokdiagram

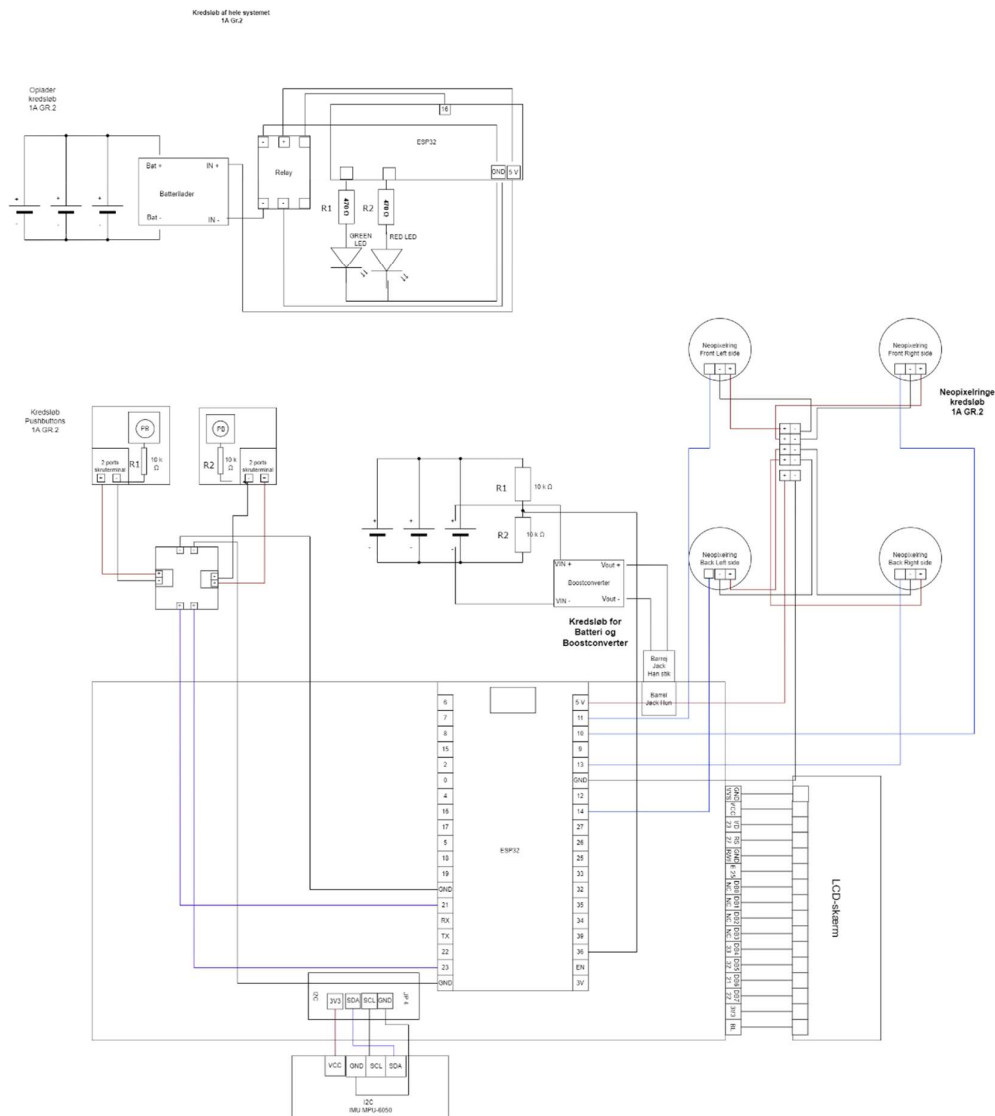
Blokdiagram
1A GR.2



Figur 16: Endelig version af diagrammet

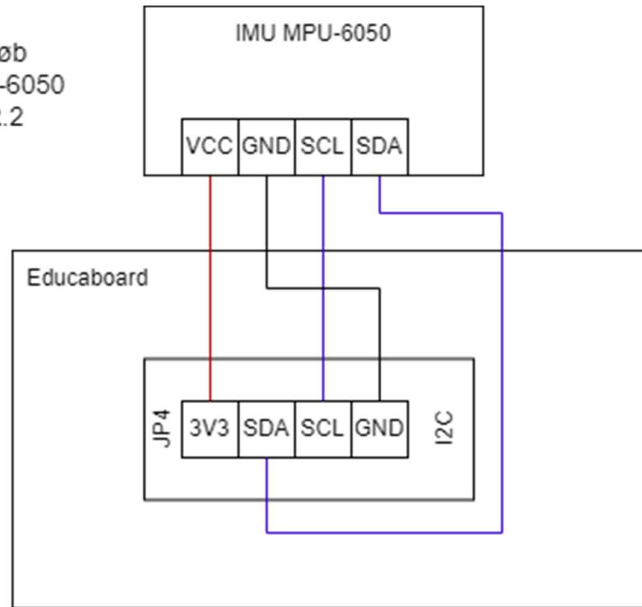
For nærmere information om blokdiagrammet se venligst teoriblokke under punkt 6.4 og datasheets i bilag.

6.2. Diagrammer

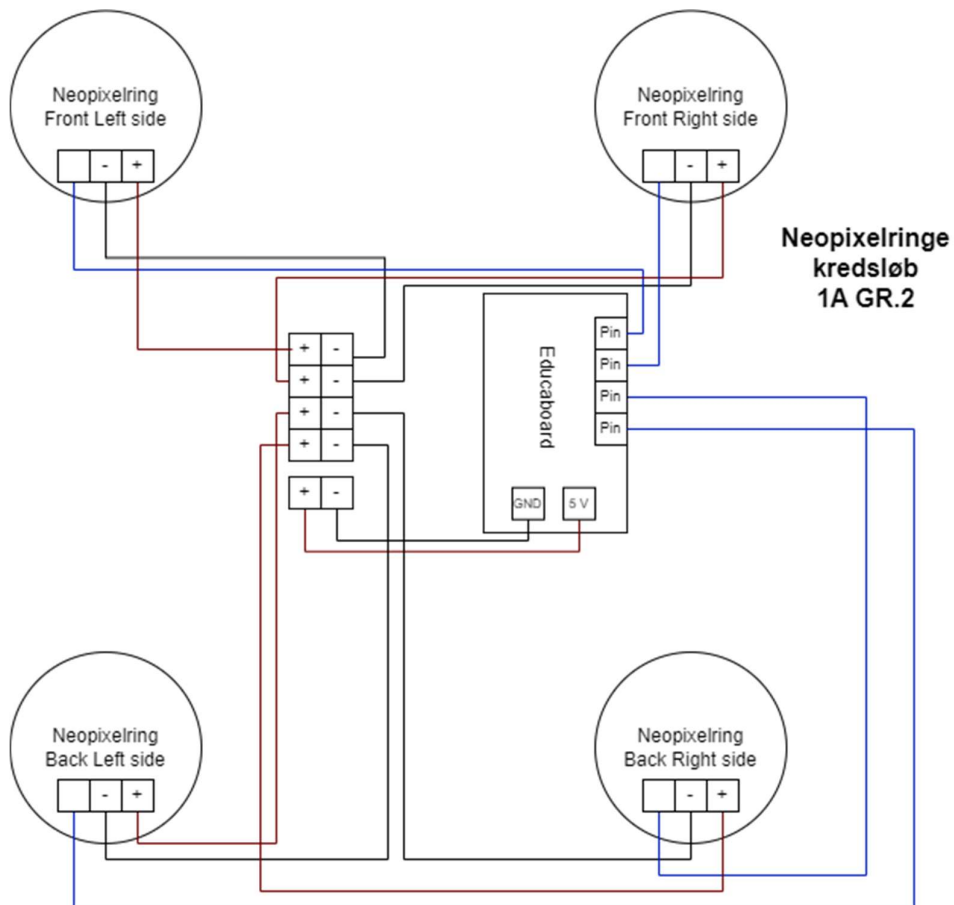


Figur 17: Diagram for hele systemet

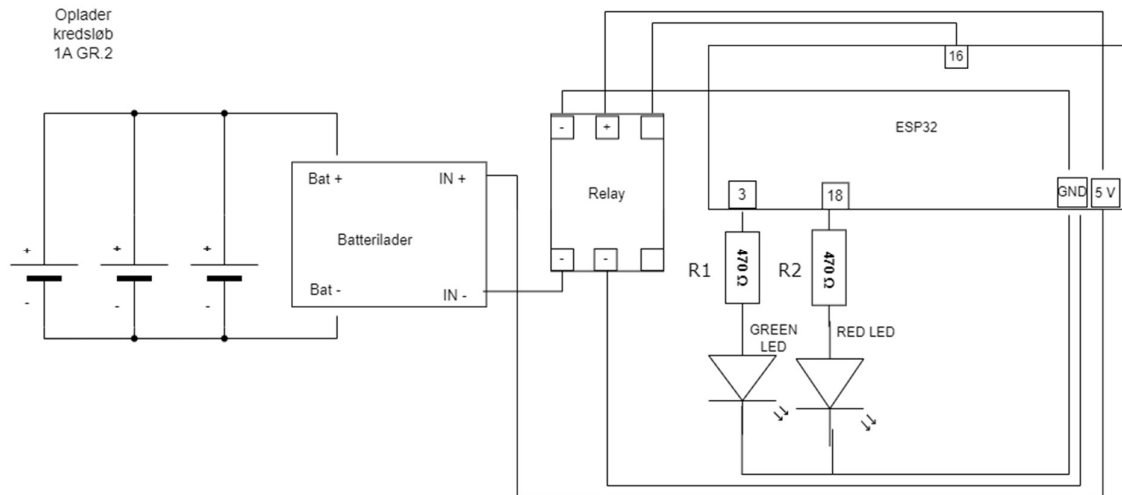
Kredsløb
IMU MPU-6050
1A GR.2



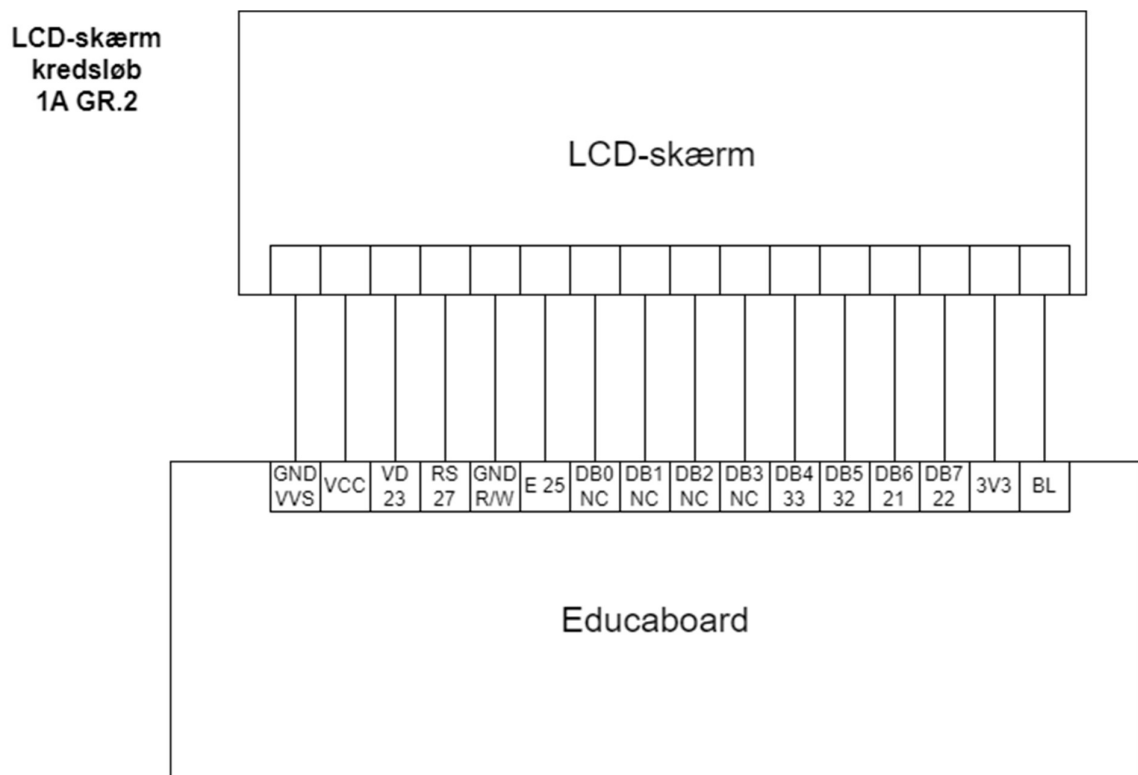
Figur 18: Diagram for IMU



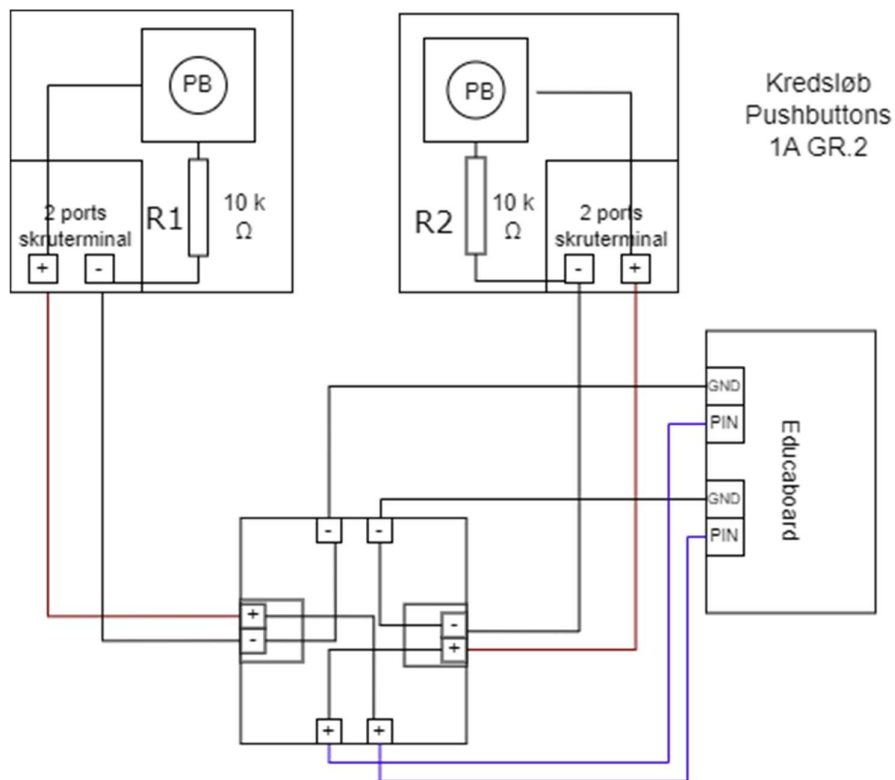
Figur 19: Diagram for Neopixel ringe



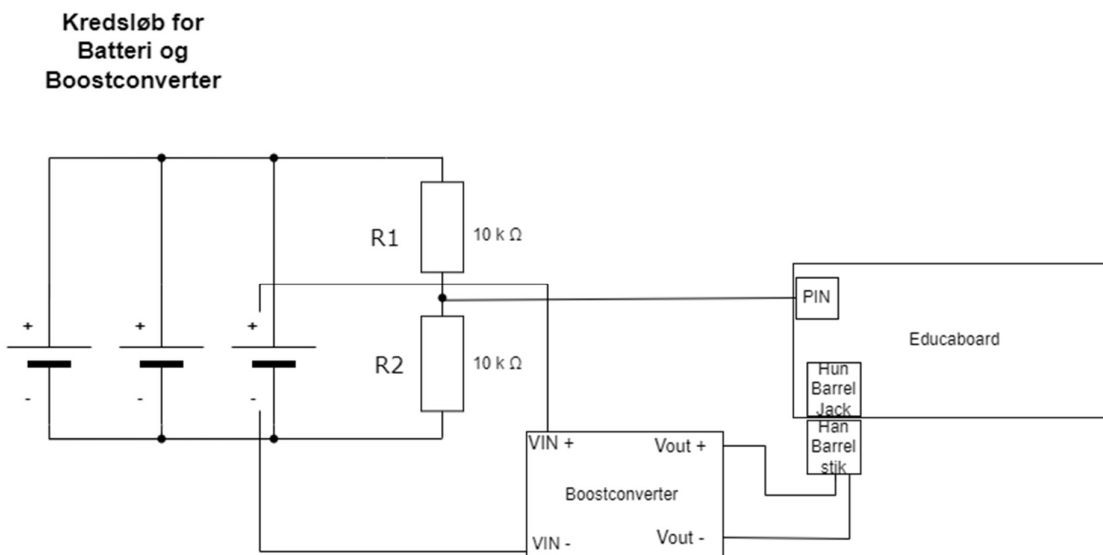
Figur 20: Diagram for opladerkredsløb



Figur 21: Diagram for LCD-skærm



Figur 22: Diagram for push-buttons



Figur 23: Diagram for batteri og boostconverter

6.3. Beregninger og målinger

Spændingsdeler Formler:

$$U_{R1} = U \cdot R_1 / (R_1 + R_2)$$

$$U_R = U \cdot R / (R + R) = U \cdot R / 2R$$

$$U_{R2} = U \cdot R_2 / (R_1 + R_2) = R_1 = R_2 = R$$

$$I_{\text{total}} = I_R = I_{R2} = U / R_{\text{total}}$$

$$R_{\text{total}} = U / I_{\text{total}}$$

$$R_2 = U_{R2} / I_{\text{total}}$$

$$P_{\text{total}} = U \cdot I_{\text{total}}$$

Udregninger for spændingsdeler:

$$R_1 = 10 \text{ k}\Omega$$

$$R_2 = 10 \text{ k}\Omega$$

$$U_{R1} = 4,2 \text{ V} \cdot 10 \text{ k}\Omega / (10 \text{ k}\Omega + 10 \text{ k}\Omega) = 4,2 \cdot 0,5 = 2,1 \text{ V}$$

$$U_{R2} = 4,2 \text{ V} \cdot 10 \text{ k}\Omega / (10 \text{ k}\Omega + 10 \text{ k}\Omega) = 4,2 \cdot 0,5 = 2,1 \text{ V}$$

$$I_{\text{total}} = 4,2 \text{ V} / (10 \text{ k}\Omega + 10 \text{ k}\Omega) = 0,00021 \text{ A} \approx 0,21 \text{ mA} \approx 210 \text{ }\mu\text{A}$$

$$R_{\text{total}} = 4,2 \text{ V} / 0,00021 \text{ A} = 20.000 \text{ }\Omega \approx 20 \text{ k}\Omega$$

$$R_2 = 2,1 \text{ V} / 0,00021 \text{ A} = 10.000 \text{ }\Omega \approx 10 \text{ k}\Omega$$

$$P_{\text{total}} = 2,1 \text{ V} \cdot 0,00021 \text{ A} = 0,000441 \text{ W} \approx 0,441 \text{ mW} \approx 441 \text{ }\mu\text{W}$$

Udregning for forventet restlevetid for batteriet.

Formel: Tid = Kapacitet / forbrug

Batterikapaciteten er 1.800 mAh pr. batteri og vi har 3 batterier, så i alt har vi 5.400 mAh. Hele vores system bruger ca. 500 mAh.

Forventet levetid for batteriet når det er fuldt opladt $5.400 / 500 \text{ mAh} = 10,8$ timer.

Når løsningen har kørt i en time, er restkapaciteten $= 5400 \text{ mAh} - (1 \cdot 500 \text{ mAh}) = 4900 \text{ mAh}$

Når løsninger har kørt i 3 timer, er restkapaciteten $= 5400 \text{ mAh} - (3 \cdot 500 \text{ mAh}) = 3900 \text{ mAh}$

Forventet restlevetid for batteriet i procent $= 3900 \text{ mAh} / 5400 \text{ mAh} \cdot 100 = 72,22 \%$

Udregning for brug af for modstande til LEDér:

$$U = 4,2 \text{ V}$$

$$I_F = 50 \text{ mA}$$

$$V_F \text{ Rød LED} = 1,8 \text{ V}$$

$$V_F \text{ Grøn LED} = 1,9 \text{ V}$$

$$\text{Formel for udregning af modstand: } R = (U - V_F) / I_F$$

$$\text{Formel for udregning af strømmen i lysdioderne: } I_F \text{ estimeret} = (U - V_F) / R \text{ valgt}$$

$$I_F \text{ estimeret Rød Lysdiode LED} = (4,2 \text{ V} - 1,8 \text{ V}) / 470 \Omega = 0,0051 \text{ A} \approx 5,1 \text{ mA}$$

$$I_F \text{ estimeret Grøn Lysdiode LED} = (4,2 \text{ V} - 1,9 \text{ V}) / 470 \Omega = 0,0049 \text{ A} \approx 4,9 \text{ mA}$$

Teoretisk udregning for R

$$R_1 \text{ Rød LED} = (4,2 \text{ V} - 1,8 \text{ V}) / 0,0051 \text{ A} = 470,59 \Omega$$

$$R_2 \text{ Grøn LED} = (4,2 \text{ V} - 1,9 \text{ V}) / 0,0049 \text{ A} = 469,39 \Omega$$

Udregning for brug af for modstande til Pushbutton:

$$U = 3,3 \text{ V}$$

$$R = 10 \text{ k}\Omega$$

$$I = 330 \mu\text{A}$$

$$\text{Formel for Ohms Lov: } U / R * I$$

$$I = U / R = 3,3 \text{ V} / 10 \text{ k}\Omega = 0,00033 \text{ A} \approx 330 \mu\text{A}$$

$$R = U / I = 3,3 \text{ V} / 330 \mu\text{A} = 10 \text{ k}\Omega$$

6.4. Teori af relevante elektroniske blokke

I vores elektroniske løsning har vi brugt flere forskellige dele. Vi har to batteriblokke, én der forsyner vores ESP32 med strøm og én der bruges til at give strøm til vores batterioplader.

Her har vi sat 3 batterier i parallel. Hvert batteri har en kapacitet på 1.800 mAh, samlet giver det en spænding på op til 4,2 V og 5.400 mAh.

For at kunne afmåle batteriprocenten, skal vi bruge en ADC-værdi og derfor bruger vi Pin 36 til at måle ADC-værdien. Pin 36 tåler max 3,3 V og da vi sender op til 4,2 V ud fra batteriet, har vi været nødt til at lave en spændingsdeler for at få strømmen ned. Vi har brugt 2 modstande a 10 KΩ, disse er sat i serie. Dette halverer derved spændingen i midten af de to modstande, hvilket er der vi tilknytter vores ADC-pin.

Da ESP32 skal bruge 5 V har det været nødvendigt at anvende en Boost-converter til at få strøm nok til ESP32. Der kommunikeres via I2C protokollen fra ESP32 ud til IMUén (se nærmere forklaring af I2C under punkt 7.3)

For at vise diverse målinger har vi brugt en LCD- skærm. Skærmen skal have 3,3 V. Flere af kravene i projektet skal også vises på Thingsboard. Vi kommunikerer til Thingsboard via Wi-Fi (Wireless Fidelity)⁹, en teknologi der anvender radiofrekvenser oftest 2,4 GHz og følger protokollen IEEE 802.11. MQTT (Message Queuing Telemetry Transport) protokollen bruges når der sendes til Thingsboard (se nærmere forklaring af MQTT under punkt 7.3).

For at vise længdegrad og breddegrad på hvor vores løsning befinder sig fysisk, skal vi bruge en GPS (Global Positioning System).

GPS'en kommunikerer via UART protokollen (se nærmere forklaring af UART under punkt 7.3).

Vi har brugt 4 stk. Neopixel ringe der hver især forbindes til Pins på ESP32.

Buzzeren sidder på Educaboardet og vi kommunikerer med buzzeren via en Pin.

Med GY-521 3-axis accelerometer/temperatursensor måler vi temperaturen og acceleration. Begge dele sendes til Thingsboard via WIFI og der bruges MQTT protokollen.

⁹ <https://www.networkworld.com/article/968819/what-is-wi-fi-and-why-is-it-so-important.html>

Bloknavn: Batteri



Produktnavn: LW 903052

Dette batteri er et Litium-ion Polymer batteri.

Batteri har en kapacitet på 1.800 mAh og har en spændingsprofil der går fra 3,0 V til 4,2 V.

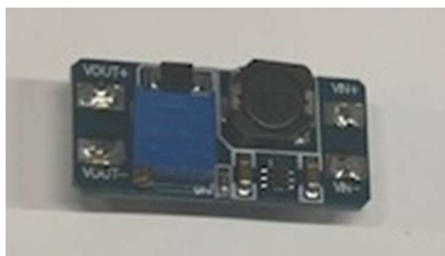
C værdien er 1 og derfor er maksimal afladnings- og opladningsstrøm 1,8 A.

Størrelsen er 52 mm x 30 mm x 9mm

Vægt: 28 g

I dette projekt bruges batteriet til at forsyne en ESP32, gennem en boost converter (beskrevet i blokken "Boost converter").

Bloknavn: Boost-converter



Produktnavn: MT3608

Boost-converteren får en lav spænding til at gå til en høj spænding. I vores løsning har vi brugt Boost-converteren til at øge spændingen til ESP32 til mellem 7-8 V.

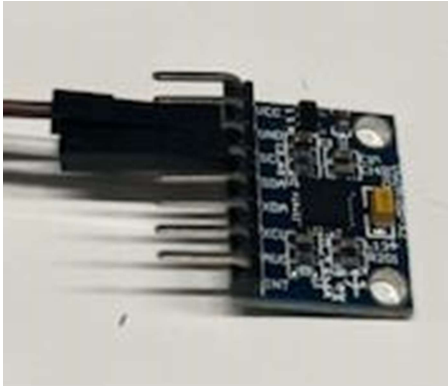
Indgangsspænding: 2 – 24 V

Udgangsspænding: 2 –28 V

Internal 4A Switch Current limit: 4 A

Integrated 80 q power Mosfett.

Bloknavn: GY-521/ IMU (MPU-6050)



Produktnavn: MPU-6050

3-axis gyroskop/ 3-axis accelerometer/ digital motion processor/temperatursensor.

Via en I2C seriel data bus kan man sende/modtage data til MPU-6050.

VDD supply Voltage range 2,3 – 3,4 V

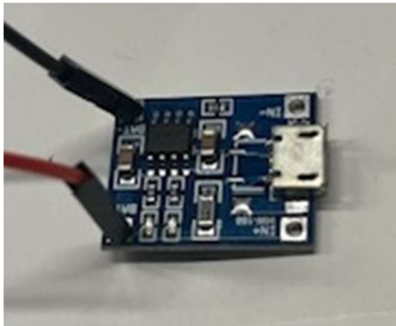
V Logic 1,8 V + - 5%

Gyro operation current fullpower 3,6 mA

Gyro + accelerometer 3,8 mA

Accelerometer 0,2 mA

Bloknavn: Batteristik



Produktnavn: TP40156/HV168

Absolute max Ratings:

Bat. Current 1200 mA

Prog.current 1200 uA

Bloknavn: LCD - skærm



Produktnavn: LCD HD44780 20x4

Numerisk displaymodul

Display protokol: STN positive

Forsyningsspænding 4,5 – 5 V

Low power supply 2,7 – 5,5 V

Bloknavn: Neopixel ring



Produktnavn:

3x12x20

720 mA

Bloknavn: ESP32



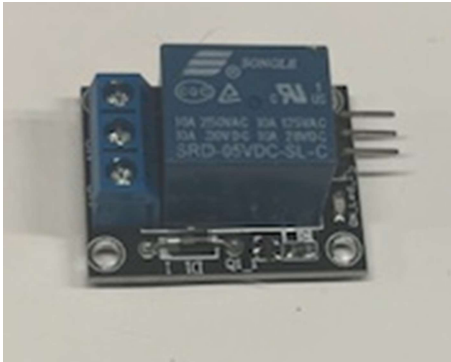
Produktnavn: ESP32- Wroom – 32 D

En mikrocontroller der bruger 5 V

80 mA i gns.

Max output 1100mA

Bloknavn: Relay



Produktnavn: SRD-05VDC-SL-C

Relæet bliver brugt i vores batteriopladerløsning hvor det er forbundet med batterioplader

5 V

Max power AC: 10 A 125 V

Max power DC: 10 A 30 V

Bloknavn: GPS



Produktnavn: Neo-7M-0-000-u-blox

3,6 V

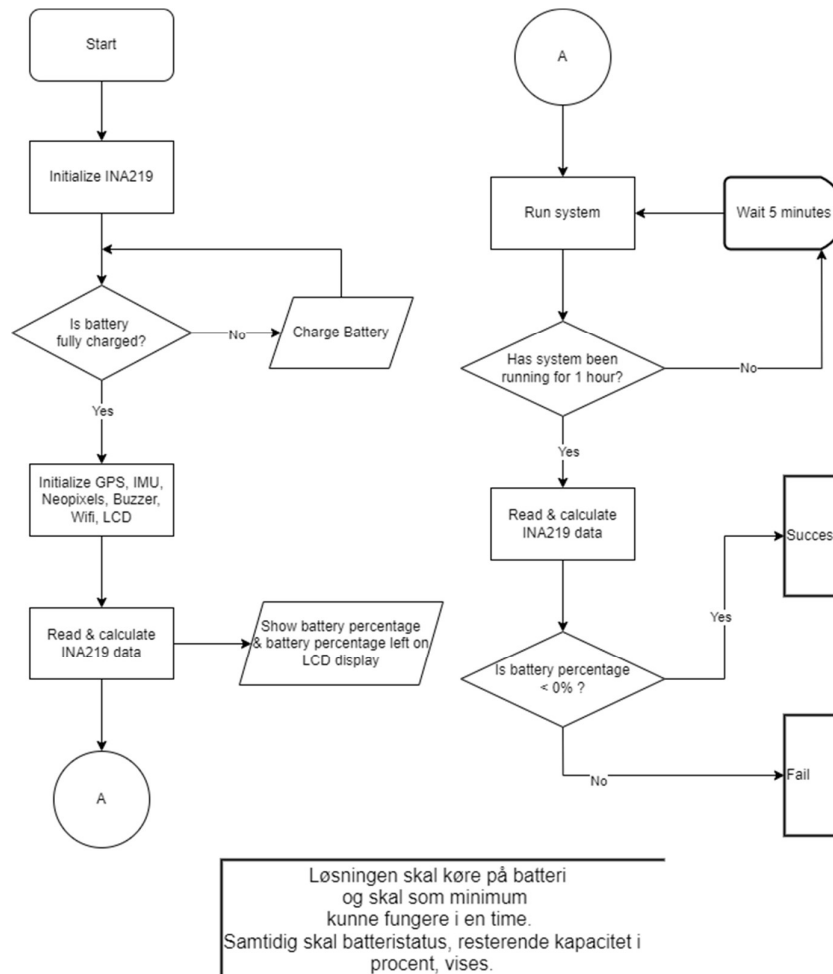
VCC Output current: 100 mA



7. Programming

7.1. Flowcharts

Krav 1 Flowchart:

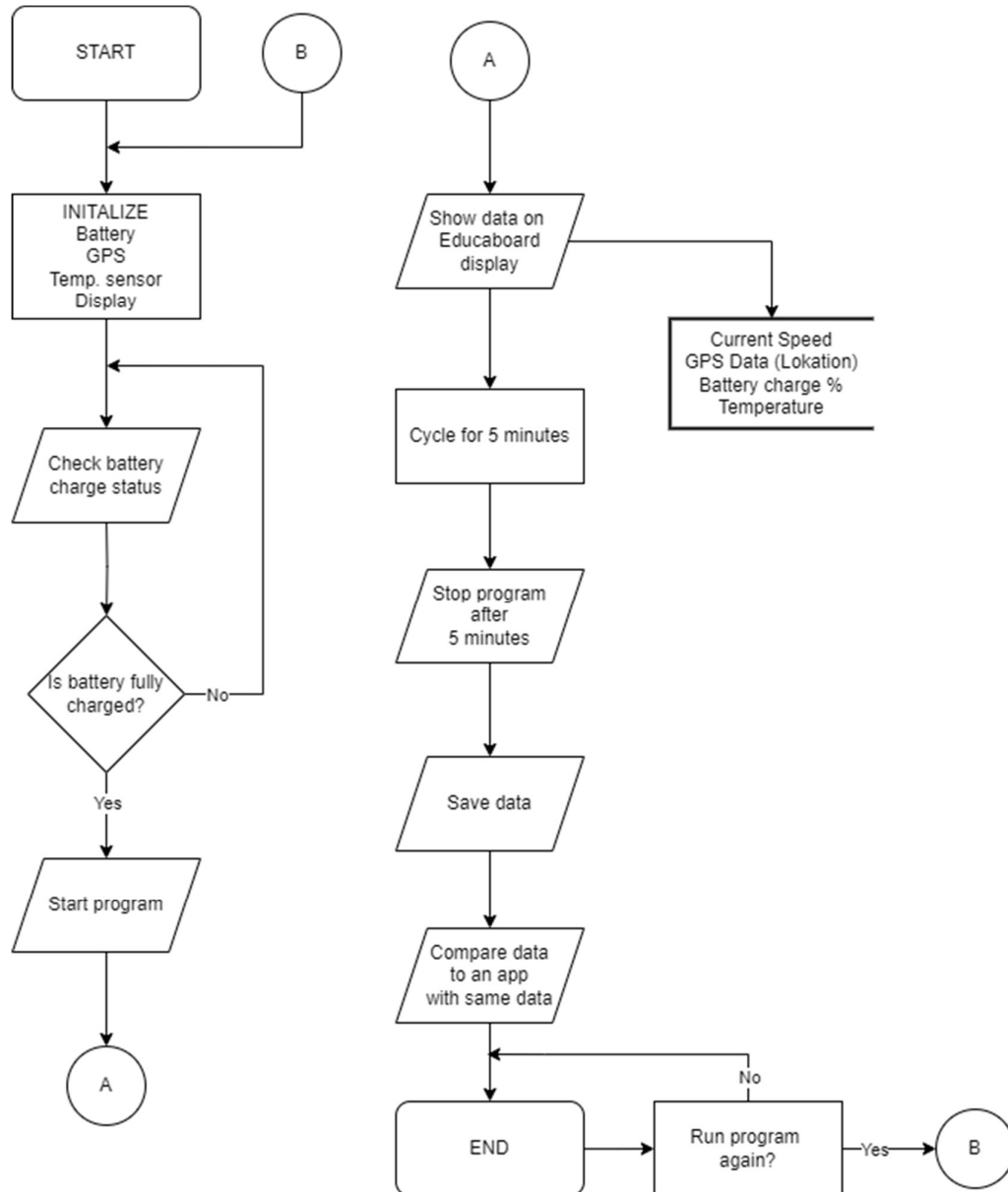


Figur 24: Flowchart for Krav 1

Dette flowchart viser processen, der tjekker om vores system overholder kravet om at kunne køre på batteri i minimum én time. Det gør det ved at starte med at tjekke om batteriet er fuldt ladet op, ved at læse måling fra komponentet INA2019. Hvis batteriet er ladet op går processen videre til næste trin, er det ikke ladet helt op lader vi det og tjekker igen indtil det er ladet tilstrækkeligt op.

Resten af flowchartet viser hele vores system der kører i én time - derefter er der to resultater, hvis der er over 0% strøm var testen en succes ellers fejlede det.

Krav 2 Flowchart:



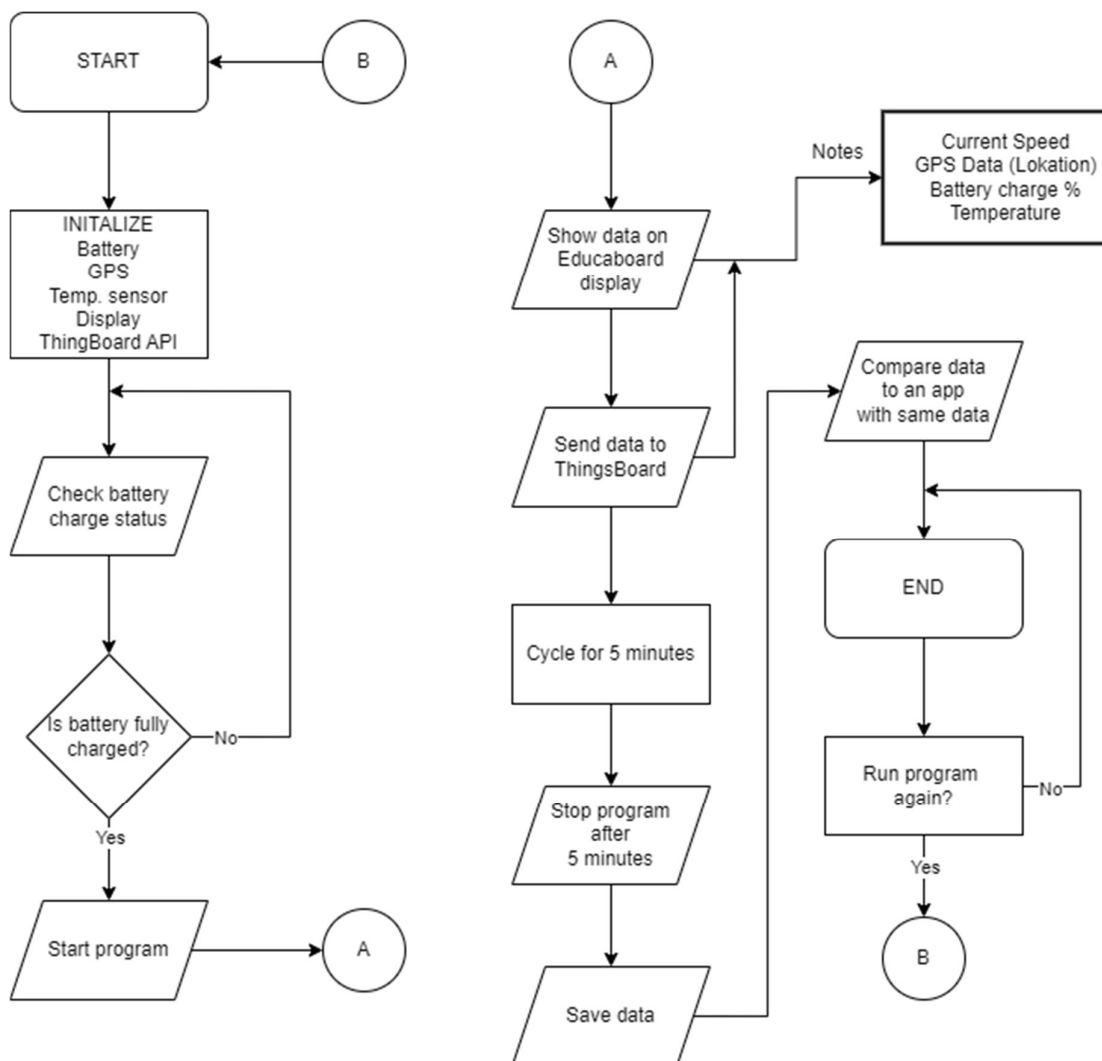
Figur 25: Flowchart for krav 2

Dette flowchart viser at vi tænder for vores system, specifikt GPS, skærm og temperatur måler

Herefter tjekker det om vores batteri er fuldt opladt, hvis batteriet er ladt op går processen videre til næste trin, er det ikke ladt helt op lader vi det og tjekker igen indtil det er ladt tilstrækkeligt op.

Når vi har sikret os at batteriet er ladt op går vi videre til at læse data fra GPS og temperatur måler, denne data viser vi på vores skærm. Vi skal cykle rundt med vores system i 5 minutter og derefter sammenligne den data vi har registret med vores system, med samme data målt på en ekstern mobiltelefon, hvis den målte data er inden for 5% nøjagtighed af mobiltelefonen er testen en succes, hvis ikke, er testen fejlet.

Krav 3 Flowchart:



Figur 26: Flowchart for krav 3

Dette flowchart viser at vi tænder for vores system.

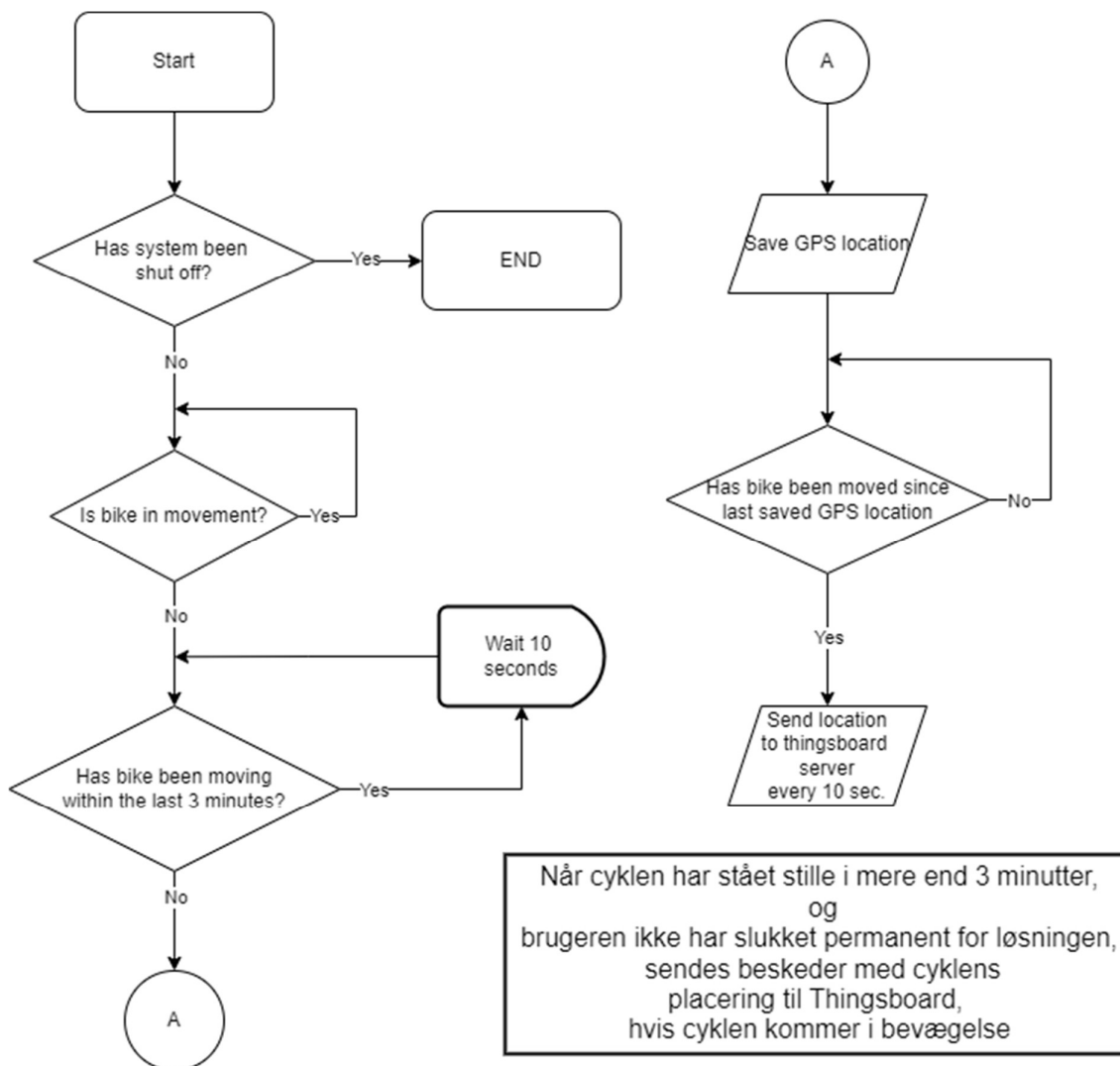
Herefter tjekker det om vores batteri er fuldt opladt. Hvis batteriet er ladt op, går processen videre til næste trin, er det ikke ladt helt op, lader vi det og tjekker igen indtil det er ladt tilstrækkeligt op.

Når batterieret er ladt op aflæser vi GPS, temperatur måler, batteriprocent og viser den målte data på vores skærm.

Udover at vise den målte data på vores skærm sender vi den også til Thingsboard hvor det præsenteres via forskellige widgets på et dashboard.

Hvis den målte data er inden for 5% nøjagtighed af den samme data målt på en ekstern mobiltelefon er testen en succes, hvis ikke er den fejlet.

Krav 4 Flowchart:



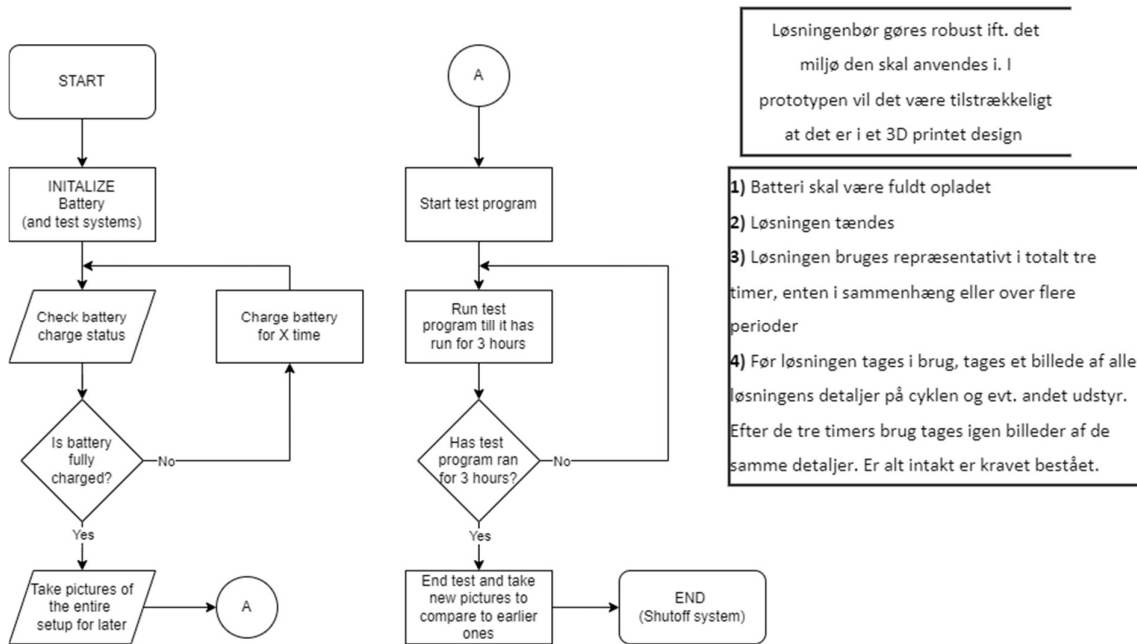
Figur 27: Flowchart for krav 4

Dette flowchart tjekker om systemet er blevet slukket for eller ej - hvis systemet er slukket sker der ikke mere.

Er systemet ikke slukket tjekker det via GPS om cyklen har rykket sig inden for de sidste 3 minutter, hvis cyklen ikke har rykket sig, venter det 10 sekunder og tjekker igen - dette gør det indtil cyklen enten har rykket sig eller systemet slukkes.

Hvis cyklen rykkes, sendes cyklens lokation til Thingsboard.

Krav 5 Flowchart:



Figur 28: Flowchart for krav 5

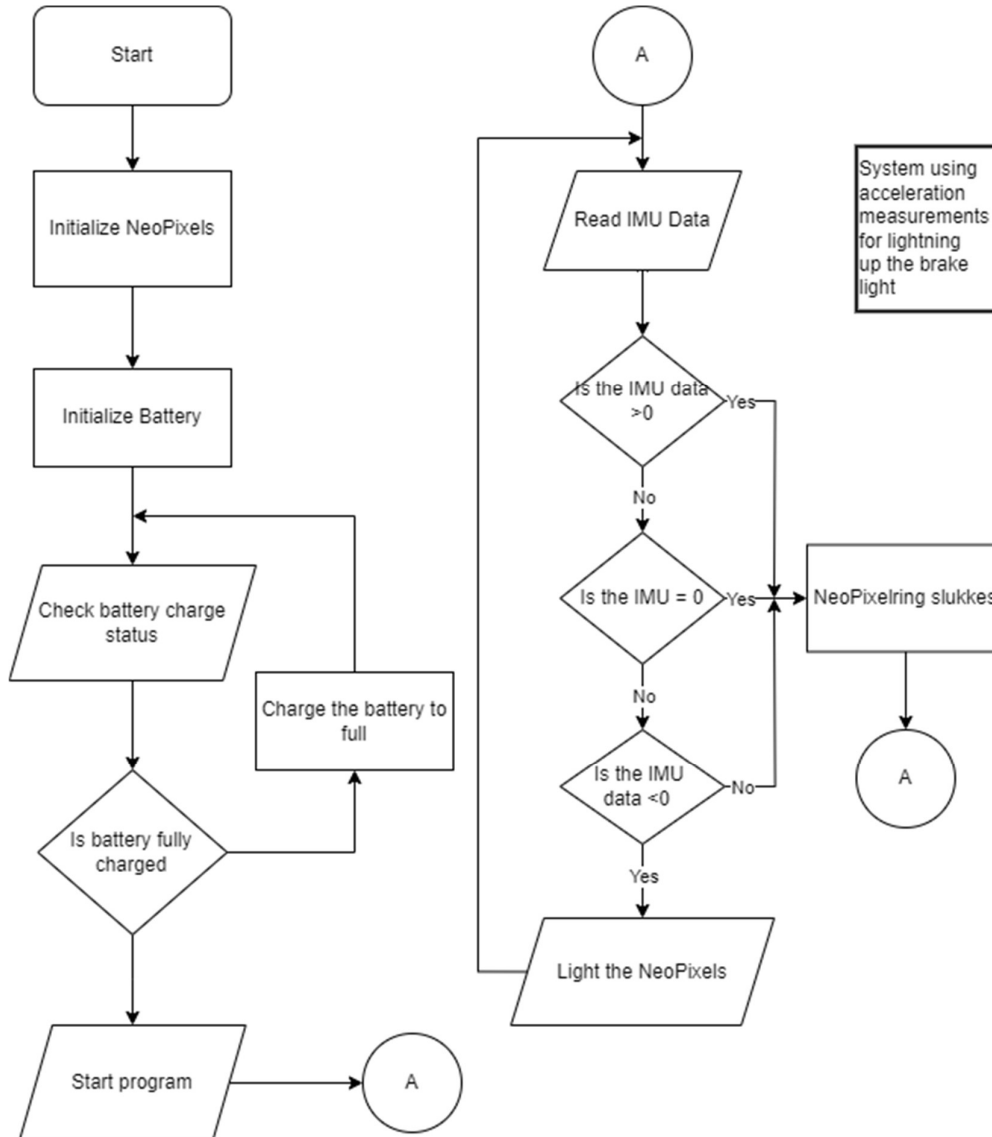
Dette flowchart tjekker det fysiske udstyr på cyklen og dets robusthed.

Det gør det ved at tjekke om vores batteri er fuldt opladt. Hvis batteriet er ladet op går processen videre til næste trin, er det ikke ladet helt op lader vi det og tjekker igen indtil det er ladet tilstrækkeligt op.

Når batteriet er ladet op tager vi et billede af det komplette system, herefter bruger vi systemet repræsentativt, tjekker om det samlet set er blevet brugt i tre timer, hvis ikke det er det, forsættes brugen af systemet indtil det samlet set er brugt i minimum tre timer.

Efter det er brugt i minimum tre timer tager vi et billede af det komplette system og sammenligner med det første billede. Hvis alt er intakt, er testen en succes, hvis ikke er den fejlet.

Krav 6 Flowchart:



Figur 29: Flowchart for krav 6

Dette flowchart viser at vi tjekker om vores batteri er fuldt opladt. Hvis batteriet er ladt op går processen videre til næste trin, er det ikke ladt helt op lader vi det og tjekker igen indtil det er ladt tilstrækkeligt op.

Er batteriet ladt op tændes systemet, dette system måler acceleration vha. IMU komponentet.

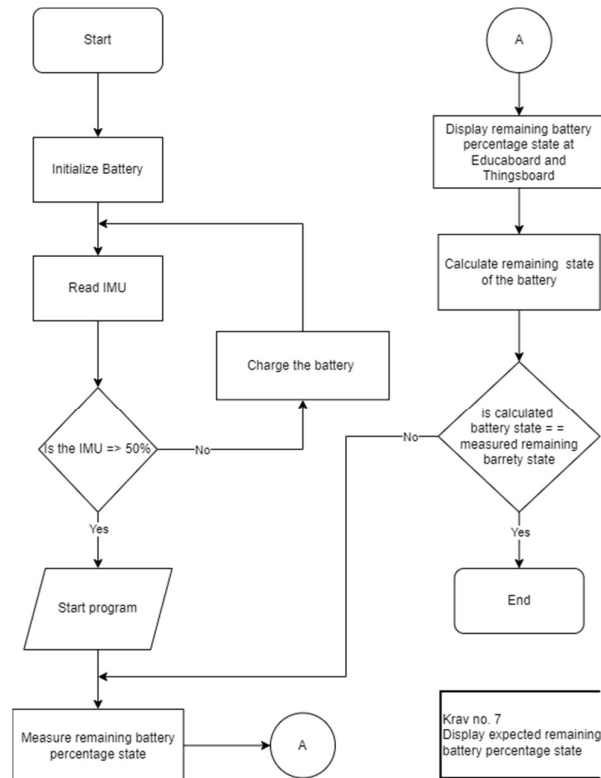
Hvis den målte IMU data er større end 0 slukkes vores bremselys hvis det var tændt

Hvis den målte data er lig med 0 slukkes vores bremselys hvis det var tændt

Hvis den målte data er mindre end 0 tændes vores bremselys

Dette system kører i ring og læser konstant IMU'ens data og opdaterer bremselyset tilsvarende.

Krav 7 Flowchart:



Figur 30: Flowchart for krav 7

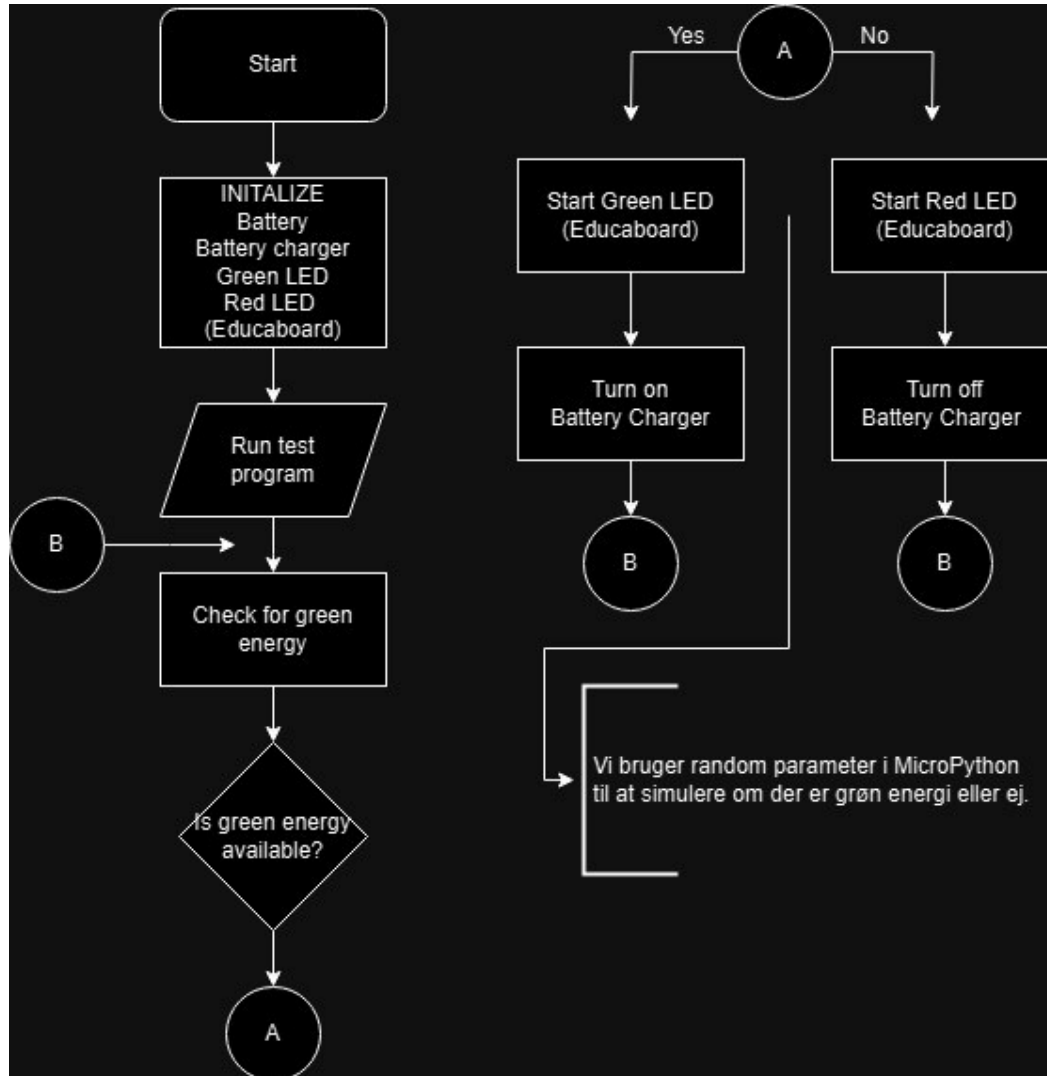
Dette flowchart viser at systemet tjekker om vores batteri er ladet minimum 50 procent op. Hvis batteriet er ladet 50 procent eller mere op, går processen videre til næste trin, er det ikke det, oplader vi det og tjekker igen indtil det er ladet tilstrækkeligt op.

Når batteriet er ladet op, går programmet videre og kører en formel der regner resterende resttid løsningen kan bruges ud.

Denne beregning vises på skærmen og sendes til Thingsboard.

Herefter beregnes resttiden manuelt, hvis den resttid systemet/programmet viser stemmer overens med den beregnede resttid er testen en succes.

Krav 8 Flowchart:



Figur 31: Flowchart for krav 8

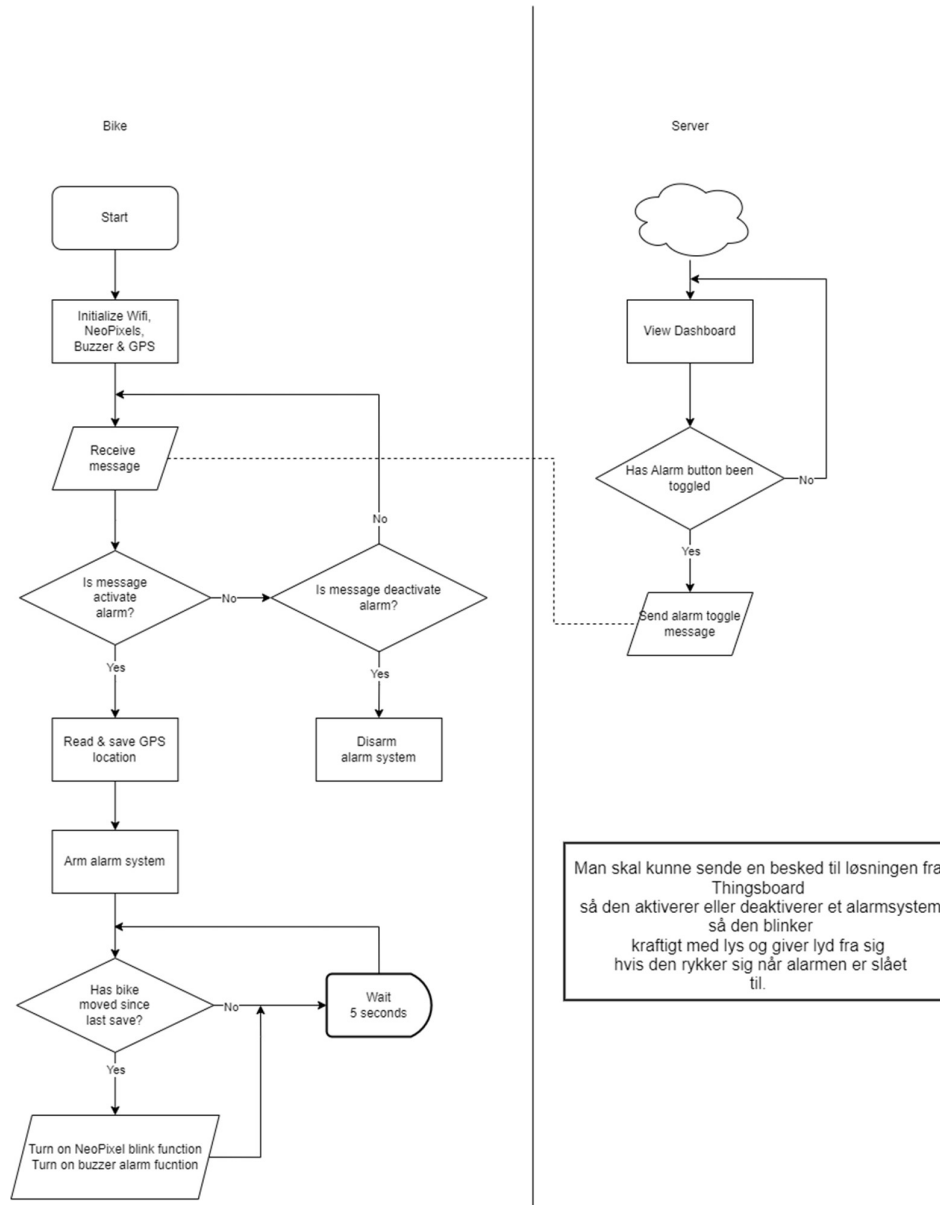
Dette flowchart viser at vi tænder vores opladersystem og tjekker om der er grøn strøm til rådighed eller ej.

Hvis der ikke er grøn strøm, vil den røde lysdiode på opladeren lyse og batterierne bliver ikke ladet op.

Så snart der kommer grøn strøm vil den grønne lysdiode på opladeren lyse og opladningen af batterierne startes.

Systemet tjekker konstant om der er grøn strøm eller ej og vil ligeledes automatisk starte og stoppe opladningen af batterierne tilsvarende.

Krav 9 Flowchart:



Figur 32: Flowchart for krav 9

Dette system tjekker beskeder sendt fra Thingsboard til systemet.

Hvis beskeden det modtager hverken er at aktivere eller deaktivere alarmsystemet sker der ikke noget.

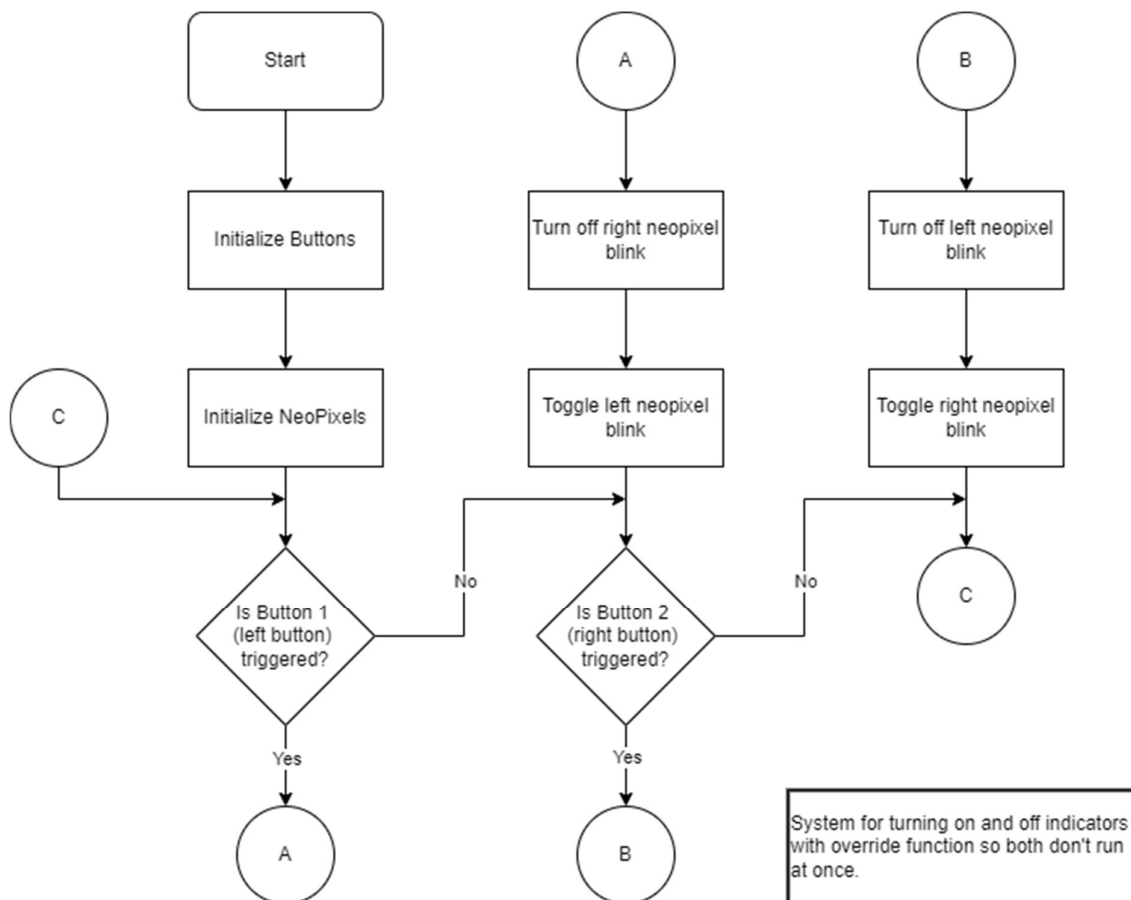
Hvis beskeden modtaget er aktiver alarmsystem aflæses GPS'ens data og gemmes.

Herefter tjekker systemet hvert 5. sekund om cyklen har rykket sig siden alarmsystemet blev aktiveret og GPS-lokationen blev gemt.

Hvis cyklen har rykket sig fra den gemte lokation, vil cyklens blinklys blinke synkront og en alarmtone vil lyde over buzzeren.

Denne funktion med alarmtone og blinklys kører indtil alarmen er blevet slukket via Thingsboard.

Krav 10 Flowchart:



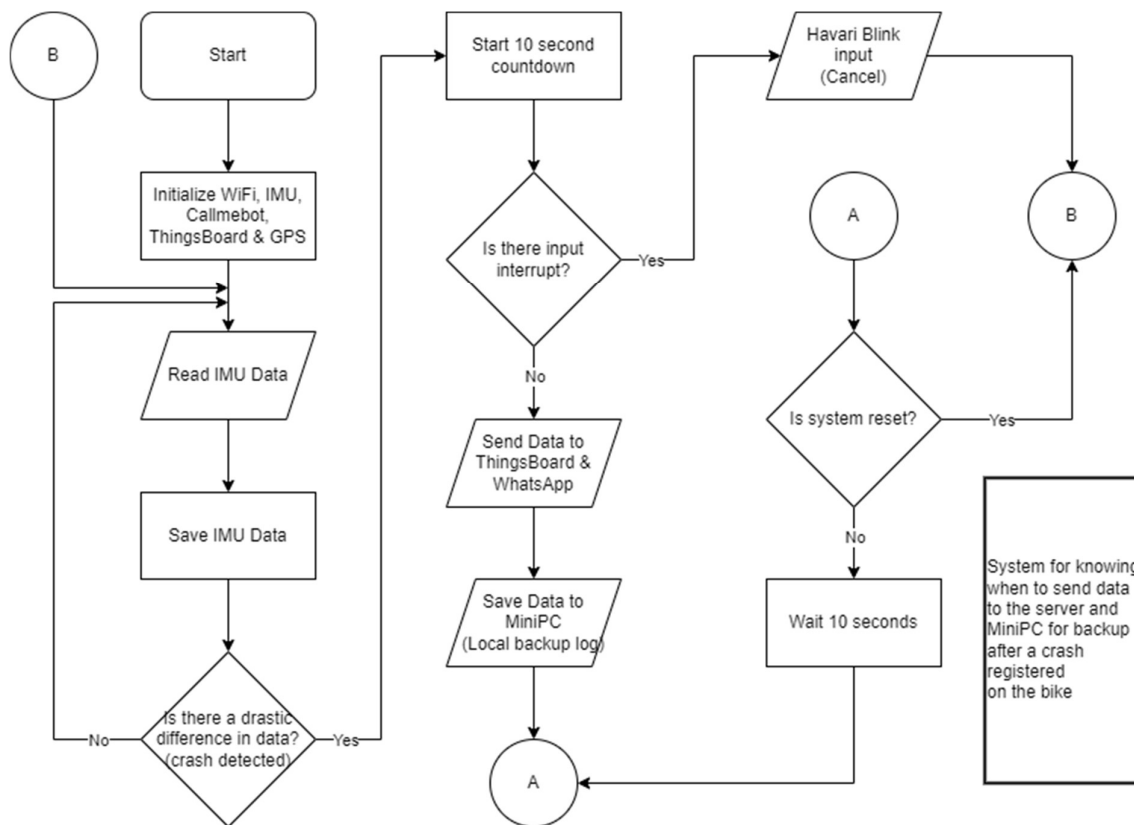
Figur 33: Flowchart for krav 10

Dette flowchart viser hvordan cyklens blinklys fungerer.

Her kan man vha. to knapper aktivere blinklys i hhv. højre og venstre side - når man tænder blinklys eks. til højre slukker man pr. automatik det modsatte blinklys.

Hvis man vil slukke helt for blinklyset, trykker man på den samme knap hvis side blinker.

Krav 11 flowchart:



Figur 34: Flowchart for krav 11

Dette flowchart tænder systemet og læser IMU data (accelerometer). Hvis der er sket et stort udsving i den læste data, dvs. et styrt, vil en 10 sekunders nedtælling begynde, her kan brugeren af systemet nå at annullere styrtregistreringen og systemet vil starte forfra med at læse IMU'en

Bliver registreringen ikke annulleret, vil et havariblink begynde (alle blinklys blinker synkront) og en styrtbesked vil blive sendt til prædefinerede kontakter via WhatsApp.

7.2. Kodebeskrivelse

Link til GitHub: <https://github.com/outofthecloud/loT1-1AGruppe2.git>

```

76 def left_button_pressed(pin):
77     global left_on, right_on, last_left_press_time
78     current_time = ticks_ms()
79     if current_time - last_left_press_time > debounce_time:
80         if left_on:
81             left_on = False
82             print("Left toggled OFF")
83         else:
84             left_on = True
85             right_on = False
86             print("Left toggled ON, Right toggled OFF")
87             last_left_press_time = current_time
88             clear_leds()
89
90 def right_button_pressed(pin):
91     global left_on, right_on, last_right_press_time
92     current_time = ticks_ms()
93     if current_time - last_right_press_time > debounce_time:
94         if right_on:
95             right_on = False
96             print("Right toggled OFF")
97         else:
98             right_on = True
99             left_on = False
100             print("Right toggled ON, Left toggled OFF")
101             last_right_press_time = current_time
102             clear_leds()

```

Figur 35: Kode fra "blinklys-krav10.py"

Kodeeksempel 1

Funktionaliteten der sørger for at programmet kan skifte imellem højre og venstre blinklys. Den består af to knapper (venstre & højre) som kan tænde og slukke som et blinklys. Denne del sørger for at Krav 10 ("Der skal være blinklys på både styr og bag på cyklen så der kan blinkes til både højre og venstre.") opfyldes.

I kodeeksempel 1, som er fra filen ved navn "blinklys-krav10.py", er der defineret en funktion til både venstre og højre knap (venstre linje 76-88, højre linje 90-102). De er styret af en defineret pin hver (se "blinklys-krav10.py"). Når der trykkes på venstre knap, vil venstre blinklys blive tændt (linje 79-81) og hvis man trykker igen, vil det blive slukket ("toggle funktion") og derfor opnå: "Left toggled OFF" (linje 82). Den næste del er til hvis højre blinklys (eksempel) er tændt, så er der mulighed for at trykke på venstre knap for at tænde venstre og slukke højre på samme tid (læs: vi skal skifte retning hurtigt, eller trykket forkert), dette vil derfor give os beskeden "Left toggled ON, Right toggled OFF" (linje 83-86). Koden til højre knap er den samme og har den samme kode, bare til højre i stedet (linje 90-102). I begge dele har vi defineret debounce for at stoppe tryk på knapperne blive aktiveret for hurtige og giver "false positives", se linje 77-79 & 87 og linje 91-93 & 101.

```

47 def caculatebearing(bearing):
48     if bearing >= 337.5 or bearing < 22.5:
49         return "N"
50     elif 22.5 <= bearing < 67.5:
51         return "NE"
52     elif 67.5 <= bearing < 112.5:
53         return "E"
54     elif 112.5 <= bearing < 157.5:
55         return "SE"
56     elif 157.5 <= bearing < 202.5:
57         return "S"
58     elif 202.5 <= bearing < 247.5:
59         return "SW"
60     elif 247.5 <= bearing < 292.5:
61         return "W"
62     elif 292.5 <= bearing < 337.5:
63         return "NW"

```

Figur 36: Kode fra "dataonlcdandthingsboard-krav3.py"

Kodeeksempel 2

Funktionaliteten der søger for at vi kan udregne hvilket retning vores GPS-modul peger. Den består af udregning af retnings grader, hvor det bliver omregnet til N/NE/E/SE/S/SW/W/NW. Denne del sørger for at Krav 3 ("Løsningen skal løbende sende data til ThingsBoard, der skal vise hastighed, retning, længde- og breddegrad, batteristatus og temperatur når cyklen er i brug. Hvis andet data måles skal det også præsenteres på Thingsboard.")

I kodeeksempel 2, som er fra filen ved navn "dataonlcdandthingsboard-krav3.py" er der defineret en funktion til at omregne de forskellige retningsgrader om til et brugbart (læselig) retning som vi alle kender. Denne del bruger data som kommer fra GPS-modulet, som bruger et GPS library ("gps_simple.py"). Udregningerne på diverse retnings grader kan ses på linje 58-63.

Et eksempel på dette fra en tilfældig linje, fx. linje 60: Hvis vores GPS-modul giver os en værdi 247.5 (retningsgrad) <= (mindre eller lig med) eller < (mindre end) værdi 292.5 (retningsgrad) så vil man kunne se W på LCD-skærmen. Denne funktion opdaterer sig selv hver gang der kommer ny data fra GPS-modulet.

```

1 from machine import Pin
2 from time import sleep
3 import random
4
5 red_pin = 18
6 green_pin = 3
7 relay_pin = 16
8 led_red = Pin(red_pin, Pin.OUT)
9 led_green = Pin(green_pin, Pin.OUT)
10 relay = Pin(relay_pin, Pin.OUT)
11
12 while True:
13     random_number = random.randint(1, 100)
14     if random_number >= 35:
15         led_green.on()
16         led_red.off()
17         relay.on()
18         print("Grøn energi! OPLADER!")
19     else:
20         led_green.off()
21         led_red.on()
22         relay.off()
23         print("Rød energi! OPLADER IKKE!")
24     print("Random number:", random_number)
25     sleep(30)

```

Figur 37: Kode fra "batterycharger-krav8.py"

Kodeeksempel 3

Funktionaliteten der sørger for at vi kan oplade vores batteri, igennem vores lade station, den kan tænde og slukke for vores relæ (og dermed vores oplader). Denne kode er med til at simulere grøn strøm til Krav 8 ("Løsningen skal være bæredygtigt og opladning af løsningens batteri skal lades med grøn energi. Der skal indikeres på løsningen når der er grøn energi til rådighed på elnettet og hvornår der ikke er grøn energi til rådighed. Løsningen skal tænde for laderen når der er grøn energi og slukke når det ikke er grøn energi til rådighed.")

I kodeeksempel 3, som er fra filen ved navn "batterycharger-krav8.py" bliver der på linje 1, 2, 3 importeret forskellige moduler. Specifikt random, som vi bruger til at simulere vores "grønne strøm". På linje 5-10 bliver de forskellige pins initialiseret og defineret til brug (red_pin, green_pin, relay_pin), hvor led_red, led_green og relay derefter bruger de satte pins (3, 16, 18). Efterfølgende i linje 13, bliver der lavet en funktion som kaldes random_number som bruger randint (dvs. random integer). I denne del bliver 1 til 100 brugt, derfor "ruller" den i mellem 1 og 100. På linje 14, tjekker den derefter om random_number (vores rul) er >= (højere eller lig med) 35. Hvis ja, så tændes den grønne led (linje 15), slukkes den røde led (linje 16) og tænder relæet (linje 17). Hvis der rulles mindre end 35, så vil den grønne led slukkes (linje 20), tænde den røde led (linje 21) og slukke relæet (linje 22). Linje 24, giver os en mulighed for at se hvilket tal der er blevet "rullet" af randomint (mellem 1 til 100). Linje 25 indikere at vi venter 30 sekunder mellem hvert "rul", dvs. den sover i 30 sekunder.

(Note: Dette kunne gøres "rigtigt" igennem API kald fra en virksomhed som giver mulighed for at bruge deres API, hvor der kan måles om vind/vejr udlevere grøn strøm, men det ville ikke være muligt at få det til at skifte hurtigt nok til acceptkrav.)

```

36         if not within_safe_zone:
37             print("Movement detected!")
38             inactivity_start_time = None
39             now = time()
40             if telemetry_last_sent_time is None or (now - telemetry_last_sent_time) >= tele_interval:
41                 telemetry_payload = {
42                     "moving": True,
43                     "latitude": current_location[0],
44                     "longitude": current_location[1],
45                 }
46                 client.send_telemetry(telemetry_payload)
47                 print(f"Telemetry sent to ThingsBoard: {telemetry_payload}")
48                 telemetry_last_sent_time = now

```

Figur 38: Kode fra "standstill-krav4.py"

Kodeeksempel 4

Funktionaliteten der sørger for at vi kan se om vores cykel (GPS-modul) stadigvæk er inden for vores "safe zone" (haversine radius, dette er til at balancere GPS modul drift). Hvis ikke, så bliver GPS modulets data sendt afsted med den nuværende position (latitude, longitude) til ThingsBoard. Denne kode er med til at løse Krav 4 ("Når cyklen har stået stille i mere end 3 minutter og brugeren ikke har slukket permanent for løsningen, sendes beskeder med cyklens placering til Thingsboard, hvis cyklen kommer i bevægelse.")

I kodeeksempel 4, som er fra filen "standstill-krav4.py", bliver der tjekket for en "safe zone" som er lavet igennem haversine udregninger, hvilket kommer fra et library (geofence.py), som startes på linje 36 (Grunden til dette er for at være sikker på cyklen faktisk rykker sig og ikke er en "false positive") hvor på linje 40 begynder den derefter at tjekke for sidste sendte telemetry til Thingsboard. Der er sat et interval igennem tele_interval (se yderligere info i "standstill-krav4.py". Linje 41-45 er vores "payload" til Thingsboard, som tjekker efter om der er blevet flyttet på cyklen (se "standstill-krav4.py"). Den sender longitude, latitude og at den er aktiv (moving). På linje 46 bliver selve payloadet sendt til Thingsboard (disse dele til Thingsboard er også igennem et library (uthingsboard.client.py).

```

28 def battery_voltage(adc_v):
29     u_batt = a * adc_v + b
30     return u_batt
31
32 def battery_percentage(u_batt):
33     without_offset = (u_batt - 3)
34     normalized = without_offset / (4.2 - 3.0)
35     percent = normalized * 100
36     return percent
37
38 def battery_average():
39     average_current = 500
40     return average_current
41
42 def battery_remaining(percent, average_current):
43     battery_capacity = 5400
44     remaining_capacity = (percent / 100) * battery_capacity
45     remaining_time_minutes = remaining_capacity / average_current
46     return remaining_time_minutes

```

Figur 39: Kode fra "leftoverbattery-krav7.py"

Kodeeksempel 5

Funktionaliteten der søger at udregne resttiden på batteriet, igennem udregninger i koden. Det er muligt at ændre batteristørrelsen (battery_capacity) og systemforbruget (average_current). Denne kode er med til at løse Krav 7 ("Visning af den forventede resttid løsningen kan bruges. Visningen skal være på lokal konsol, Educaboard og på Thingsboard.")

I kodeeksempel 5, som er fra filen "leftoverbattery-krav7.py", bliver der defineret flere funktioner (4), den ene battery_voltage (linje 28 til 30), nummer to battery_percentage (linje 32 til 36), nummer tre battery_average (linje 38 til 40), nummer fire battery_remaining (linje 42 til 46). De forskellige dele gør forskellige ting: Battery_voltage tager imod data igennem ADC (pin 36) og gemmes i u_batt (linje 30). Der bliver derefter udregnet ud fra den data (u_batt) som bliver gemt i "percent" på linje 36. Derefter bruger vi vores gennemsnitlig brug af mAh hvilket er sat til 500 (linje 39) og bliver gemt i average_current (linje 40). Derefter bliver den resterende tid på batteriet udregnet ud fra vores totale batteriet kapacitet (5.400 mAh, se linje 43, vi har 3x1.800 mAh batterier). Derefter bruger den de tidligere nævnte gemte dele til at udregne remaining_time_minutes (som ses på linje 46).

```

59         if not within_buffer:
60             print("BIKE IS MOVING!")
61         if alarm_activated:
62             print("ALARM ON!")
63             for _ in range(10):
64                 buzzer(buzzer_pwm, 3000, 0.1, 0.1)
65                 buzzer(buzzer_pwm, 2000, 0.1, 0.2)
66                 blink_neopixels()
67             client.send_telemetry({"alarm": "SOUND THE ALARM!", "location": current_location})
68         else:
69             client.send_telemetry({"location": current_location})

```

Figur 40: Kode fra "bikealarm-krav9.py"

Kodeeksempel 6

Funktionaliteten der søger for at håndtere alarmer, hvis cyklen begynder at rykke sig, hvis den er uden for vores haversine radius zone fra vores library (geofence.py).

Hvis alarmeren er aktiveret, så tænder vores alarm (buzzer) med en alarmtone, blinker vores neopixels og sender dataene til Thingsboard (med GPS-lokation, longitude & latitude). Denne kode er med til at løse Krav 9 ("Man skal kunne sende en besked til løsningen fra Thingsboard så den aktiverer eller deaktiverer et alarmsystem så den blinker kraftigt med lys og giver lyd fra sig hvis den rykker sig når alarmeren er slået til.")

I kodeeksempel 6, som er fra filen "bikealarm-krav9.py", bliver der tjekket for om cyklen er inden for vores selv satte haversine radius på linje 59 (se linje 57 i "bikealarm-krav9.py"). Den printer igennem linje 60 at cyklen rykker på sig, men ingen alarm bliver tændt. Hvis alarmeren er tændt (linje 61), bliver alarmeren startet og buzzeren begynder at lave lyde og neopixels begynder at blinke (se linje 63 til 66). Linje 67-69 gør det muligt at tracke cyklen igennem Thingsboard (GPS-lokation).

```
65 def updatelcd(battery_percent, temperature, speed, latitude, longitude, direction):
66     lcd.clear()
67     lcd.move_to(0, 0)
68     lcd.putstr(f"T: {temperature}C & Dir: {direction}")
69     lcd.move_to(0, 1)
70     lcd.putstr(f"Speed: {speed}m/s")
71     lcd.move_to(0, 2)
72     lcd.putstr(f"La: {latitude}")
73     lcd.move_to(13, 2)
74     lcd.putstr(f"& Batt:")
75     lcd.move_to(0, 3)
76     lcd.putstr(f"Lo: {longitude}")
77     lcd.move_to(15, 3)
78     lcd.putstr(f"{{battery_percent}}%")
```

Figur 41: Kode fra "dataonlcd-krav2.py"

Kodeeksempel 7

Funktionaliteten der søger for at opdatere LCD skærmen med de forskellige data punkter. Der er defineret en funktion til at opdatere battery procent, temperatur, hastighed, latitude, longitude, direction). Denne kode er med til at løse Krav 2 ("På den lokale konsol, Educaboard, skal vises hastighed, retning, længde- og breddegrad, batteristatus og temperatur.")

I kodeeksempel 7, som er fra filen "dataonlcd-krav2.py", bliver på linje 65 lavet en funktion som hedder updatelcd. Den er linket til de forskellige data inputs fra vores GPS, batteri pin (36) og MPU6050 til temperatur.

Linje 66 clearer den skærmen, så ny data kan blive displayet. Fra linje 67 til linje 78 bliver ny data opdateret på hver forskellige dele af displayet (0, 0) + (0, 1) + (0, 2) + (13, 2) + (0, 3) + (15, 3). De forskellige data punkter er opdelt på forskellige steder på skærmen.


```
132         formatted_message = MESSAGE.format(latitude=lat or "N/A",
133                                             longitude=lon or "N/A",
134                                             local_time=local_time)
135         Whatsapp_api.send_message(PHONE_NUMBER, formatted_message, API_KEY)
136         print("CRASH DATA SENT TO LOVED ONES, SENT!")
137
```

Figur 42: Kode fra "crashdetection-krav11.py"

Kodeeksempel 8

Funktionaliteten der søger for at opdatere skærmen, når den sender en crash besked. Den sender en besked igennem Callmebot's Whatsapp API til udvalgte personer. Koden er til at løse to forskellige krav:

- Krav 11

("Når accelerometer registrerer et styrt vil der blive vist en besked på lcd skærmen, at et styrt er registreret. Herefter vil en nedtælling på 10 sekunder begynde så man har mulighed for at annullere registreringen i tilfælde af at dette var en fejl og ikke et uheld. Annullering sker ved at trykke på begge knapper på styret på samme tid. Bliver der ikke trykket vil alle 4 blinklys blinke synkront 1 gang i sekundet indtil systemet nulstilles.")

- Krav 18

("Når cyklens accelerometer registrerer et styrt som overskrider grænsen, skal data (GPS Lokation/Tid) efter 10 sekunder sendes til ThingsBoard (API) og til relevante personer (igennem callmebot WhatsApp API). Data skal også gemmes lokalt på mini pc.")

I kodeeksempel 8, som er fra filen "crashdetection-krav11.py", er det muligt at se programmets Whatsapp API besked som sender de udvalgte data (GPS, Tid), som bliver afsendt til vores udvalgte person(er) på linje 135. Selve "Payloadet" af data kan ses på linje 132, 134 som bruger ntptime/localtime og GPS-data longitude og latitude. Programmet kører i en "Crash!" mode indtil programmet bliver nulstillet igennem tryk på knapper.

```

23 while True:
24     current_time = ticks_ms()
25     if ticks_diff(current_time, last_loop_time) >= loop_interval:
26         last_loop_time = current_time
27         values = imu.get_values()
28         acc_x = values["acceleration x"]
29         if acc_x > 15000 and not red_active:
30             set_color(rnp, 10, 0, 0)
31             set_color(lnp, 10, 0, 0)
32             red_active = True
33             red_start_time = ticks_ms()
34         if red_active and ticks_diff(ticks_ms(), red_start_time) > red_duration:
35             set_color(rnp, 0, 0, 0)
36             set_color(lnp, 0, 0, 0)
37             red_active = False
38         print(acc_x)

```

Figur 43: Kode fra "bremselys-krav6.py"

Kodeeksempel 9

Funktionaliteten der søger for at styrer vores bremselys.

Den bruger MPU6050 til at styrer hvilken værdi som udløser bremselyset.

Koden er til at løse Krav 6 ("Bremselys skal lyse når der bremses over en valgfri negativ acceleration.")

I kodeeksempel 9, som er fra filen "bremselys-krav6.py", er det "Main Loopet" som ses, på linje 27- læses data fra IMU (MPU 6050) på x aksen, hvor den derefter tjekker om den var højere end 15000 og lyset ikke allerede lyser (linje 29), hvis ikke, så sættes neopixel ringene i gang med at blinke i rød på "brightness" 10 på linje 30-31. Derefter tjekker den igen via vores non-blocking delay og red_active definition, for at slukke for neopixel ringene igen, linje 34-37, til sidst printes værdien fra acc_x (linje 38).

```
16 pin_potmeter = 36
17 potmeter_adc = ADC(Pin(pin_potmeter))
18 potmeter_adc.atten(ADC.ATTN_11DB)
19 adc1 = 2716
20 adc2 = 3181
21 U1_volt = 3.0
22 U2_volt = 4.2
23 a = (U2_volt - U1_volt) / (adc2 - adc1)
24 b = U1_volt - a * adc1
25
```

Figur 44: Kode fra "batteryprocent-krav1.py"

Kodeeksempel 10

Funktionaliteten der søger for at styrer vores batteri procent udregning, som vises på LCD skærmen. Derefter kan vi se vores nuværende procent og dermed vores resterende tid. Alt dette vises på skærmen.

Koden er med til at løse Krav 1 ("Løsningen skal køre på batteri og skal som minimum kunne fungere i en time. Samtidig skal batteristatus, resterende kapacitet i procent, vises.")

I kodeeksempel 10, som er fra filen "batteryprocent-krav1.py", ses der forskellige tal som er taget ud fra vores målinger som er sket igennem brug af USB strømforsyning på 4,2v og 3,0v. På linje 16 er vores målings pin (Pin 36) defineret og vores opløsning på linje 18.

Linje 19 er vores 0% batteri/3,0 V ADC-værdi. Linje 21 er vores 0% batteri værdi (3,0 V).

Linje 20 er vores 100% batteri/4,2 V ADC-værdi. Linje 22 er vores 100% batteri værdi (4,2 V).

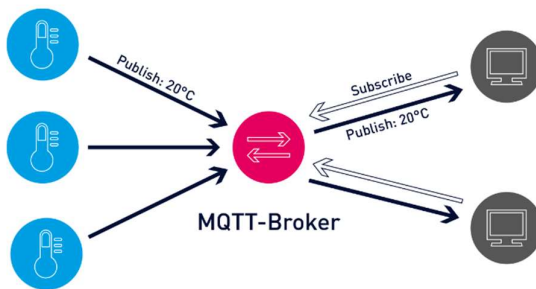
Disse udregnes direkte i programmet (linje 23-24) og derefter bruges i en defineret funktion som derefter vises på vores LCD-skærmen med "Battery Remaining".

7.3. Teori af relevante kodeblokke

MQTT – Pensum:

MQTT står for "Message Queueing Telemetry Transport", som er en letvægts messaging protokol, der bruges til at sende data imellem enheder. Den er oftest brugt til IoT (Internet of Things) enheder og systemer, som kræver en god forbindelse og lav båndbredde (og lavt energiforbrug). MQTT bruger en Publish/Subscribe model, hvor en centraliseret server styrer enhederne, som kan enten "publish" data eller "subscribe" på emner (topics).¹⁰

Hvis man skal kigge på mulige begrænsninger for MQTT: Systemet er afhængig af en central "broker", MQTT har ikke som "standard" indbygget kryptering, store MQTT-netværk kan have en del problemer



med skalering (hvis antallet af forbindelser stiger for hurtigt) eller i store MQTT-netværk kan man muligvis opleve problemer med QoS (Quality of Service).

Grunden til bruget af MQTT er for at kunne snakke sammen med Thingsboard for at sende data imellem vores ESP32 og Thingsboard til at fremvise data eksternt. Det er let at integrere, det er skalerbart og det er forholdsvis en pålidelig

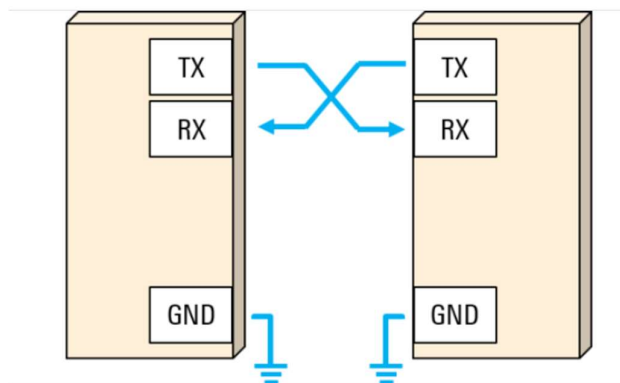
Figur 45: Grafik over basal funktion af MQTT

forbindelse. MQTT sender data igennem MQTT Protokollen via TCP/IP.

UART – Pensum:

UART står for "Universal Asynchronous Receiver / Transmitter", som er en protokol (eller også meget simpelt et sæt af regler) for at sende/modtage data imellem to enheder. UART, som protokol, er meget simpel. Den bruger kun to ledninger, imellem afsender og modtager, til at sende/modtage i begge retninger. Begge ender har en jordet forbindelse. Kommunikation i UART kan være både: Simplex (data er kun sendt en vej), Half-Duplex (data kan komme fra begge sider, men en ad gangen),

Full-Duplex (data kan komme fra begge side, på samme tid).



Figur 46: Basis diagram over hvordan UART er sat op fysisk

En af de store fordele ved UART er det er "asynchronous", hvilket betyder at sender/receiver ikke behøver at være på det samme clock signal. Det gør dog selve protokollen mere simpel, men det putter også andre nødvendigheder på plads, siden de ikke har det samme clock signal, er det nødvendigt at bruge den samme Baud Rate, hvor i vores tilfælde er sat til 9600.

¹⁰ <https://www.paessler.com/it-explained/mqtt>

UART sender dens data i "frames", hvor dette er et eksempel på et frame^{11, [8]}



Figur 47: Eksempel på frame

Her finder man: "Start/Stop bits", "Data bits" (disse bits indeholder dataene, den mest normale mængde af data bits er 7-8) og "Parity bits" (denne bit er til at tjekke, om der er fejl på forrige bits) og en masse logiske "1" og "0".

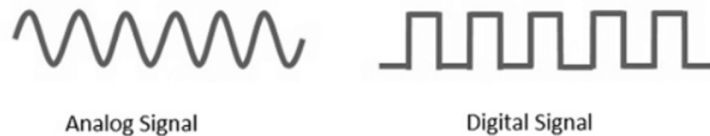
Som med mange binære systemer, så er "Høj" normalt brugt til at indikere et "1" og "Lav" er indikeret med et "0". I UARTs tilfælde når vores frame er sat til "1" (logisk høj) er den i en "idle state" (ingen data) og hvor "0" (logisk lav) er vores data.

Et kig på begrænsninger hos UART, så kunne en problematik være den lave hastighed, hvis vi sammenligner med andre protokoller (f.eks. I2C). En anden kunne være hvis Baud Rates ikke er de samme (mismatch) vil systemet lave fejl. Det er kun muligt at kommunikere imellem to enheder på samme tid. På den positive side, så er det nemt at implementere, kræver minimal hardware (dvs. det er billigt!) og er utrolig kompatibel med mange mikrocontrollers (og eksterne dele), vores ESP32 og GPS-modul, som eksempler. Dette er også vores formål med UART, til vores GPS-modul, som sidder på vores Educaboard (og kobles til vores antenne). UART er primært stadigvæk brugt som en seriel (f.eks. til RS-232) data protokol.

¹¹ https://www.rohde-schwarz.com/cz/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart_254524.html#:~:text=UART%20stands%20for%20universal%20asynchronous,and%20receive%20in%20both%20directions

ADC - Pensum:

ADC, der står for “Analog-til-digital-konverter”, bruges som et delkredsløb i et større elektronisk kredsløb, hvor et system (vores ESP32 på Educaboard) skal kunne aflæse (måle) en spænding. I vores system bruger vi ADC'en på Educaboardet, som kører via Pin 36. Grundlæggende skal en ADC omsætte et analogt signal til et digitalt signal, som en computer (ESP32 i denne sammenhæng) skal kunne forstå:¹²



Figur 48: Forskel mellem analogt og digitalt signal

Vores ADC bruger attenuation (opløsning) værdi 11 dB, hvor 12 dB er maksimum. Opløsning 12 dB giver en ADC Værdi på 4095 (3,3v er den maksimale spænding den kan tage). Hvis en spænding er større end 3,3v, så skal man have en spændingsdelers på (det skal den i vores tilfælde). Mulig problematik ved en ADC kunne være elektrisk støj. Vi har valgt at bruge ADC'en på Educaboard, da flere af vores krav kræver, at vi kan udregne & måle batteriprocenten. Vi mener det er den bedste måde at gøre det på, ud fra vores pensum. ESP32 har en indbygget ADC (Pin 34), som ikke fungerer perfekt, derfor er det Educaboardet som bliver brugt (Pin 36).

¹² <https://www.ic-components.dk/blog/analog-to-digital-converters-the-heart-of-modern-electronics.jsp>

I2C - Pensum:

I2C, kort for "Inter-Integrated Circuit" også kaldt "Two-Wire Interface", er en kommunikationsprotokol, der bruger 2 data-pins, SDA (Serial Data) og SCL (Serial Clock). Fordelen ved I2C er at én controller kan have styr over flere targets. Ved at den skal holde styr på flere targets, bruger den også en større datamængde når den sender og modtager. Det kalder man typisk en Message.¹³ En Message består af en Start Condition, derefter en Adress Frame, der siger hvor dataene skal sendes hen. De er typisk på 7-10 bit. Der kommer så en Read/Write-bit, som der giver besked om kontrollere giver eller beder om data fra det valgte target. Efter der sendes der er en ACK/NACK-bit, som står for Acknowledge/No-Acknowledge, for at se om target har modtaget dataene. Derefter kører den med skiftevis Data Frames, som er på 8 bit, og en ACK/NACK-bit, indtil den får en Stop Condition.

Vi har i vores løsning brugt I2C til vores IMU. Dette er valgt, delvist fordi det var hvad der allerede var sat til vores Educaboard, men også det vi har fået undervisning i ift. til den enhed, til hvad vi skal bruge.

WhatsApp API – Uden for pensum:

API står for "Application Programming Interface". Vores API kommer fra udbyderen "Callmebot.com", som tilbyder flere forskellige API'er til forskellige projekter.

Bruget af WhatsApp var en relevant (i dagens verden) måde at gøre det på og derfor bruger vi det til Krav 18. Man kunne bruge mange forskellige tjenester, men da dette er en prototype, tænkte vi at bruge en internetbaseret tjeneste som WhatsApp, frem for SMS, hvilket ville jo selvfølgelig være mere robust og kunde-tilgængelig løsning. Systemet (API) kaldet fungerer i vores crash detection system og sender en automatisk besked.

Wi-Fi – Uden for pensum:

Wi-Fi står for "Wireless Fidelity", er på et helt "basis niveau" en måde at trådløst koble to eller flere enheder til hinanden. Dette er muligt igennem forskellige standarder, så som IEEE 802.11n, ac, ax (Wi-Fi 4, 5, 6, 6E)¹⁴. Wi-Fi sender data igennem TCP/IP, hvor den bruger enten 2.4 GHz eller 5 GHz frekvensen som forbindelse. I vores system bruger vi ESP32, som har en indbygget Wi-Fi antenne (2.4 GHz). Det giver muligheden for at tilslutte til forskellige enheder (i vores projekt MQTT og derfor til Thingsboard).

Non-blocking delays – Pensum:

Non-blocking delays er en måde at kunne holde pauser i et stykke kode, uden at den stopper alt for at køre imens. Dvs. der ville kunne sættes to eller flere ting op, som begge skal kører med en form for forsinkelse i sig, men at de kan køre uafhængigt af hinanden, altså ikke blokere hinanden.

Dette kan opnås ved at i stedet for at bruge en sleep() eller sleep_ms(), så bruges der ticks() eller ticks_ms()¹⁵. Da ticks er en konstant optællende variabel fra enheden starter, kan man sætte koden op til at tjekke efter hvor mange ticks der gået, hvor man derefter kan sammenligne med en formel om nok ticks er gået siden sidst til.

¹³ <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>

¹⁴ <https://standards.ieee.org/beyond-standards/the-evolution-of-wi-fi-technology-and-standards/>

¹⁵ <https://docs.micropython.org/en/latest/library/time.html>

8. Netværk

I kapitel 9 får vi stillede en række krav (ID:12-18) over for vores gruppe, som vi vil løse ved en netværksopsætning, der består af 1 Router, 1 Switch, 1 Wireless acces point, 3 server (VM), 1 FW Virtuel, 6 Pc'er (der er fordelt til at hold et VM-miljø og være klient maskiner/ Virtuel og fysisk).

Ved brug af disse enheder, vil vi kunne opstille et netværk, som vil kunne tag brug af ID 12 (IP Plan) og leve op til ID 13 (fungerende fysisk netværk).

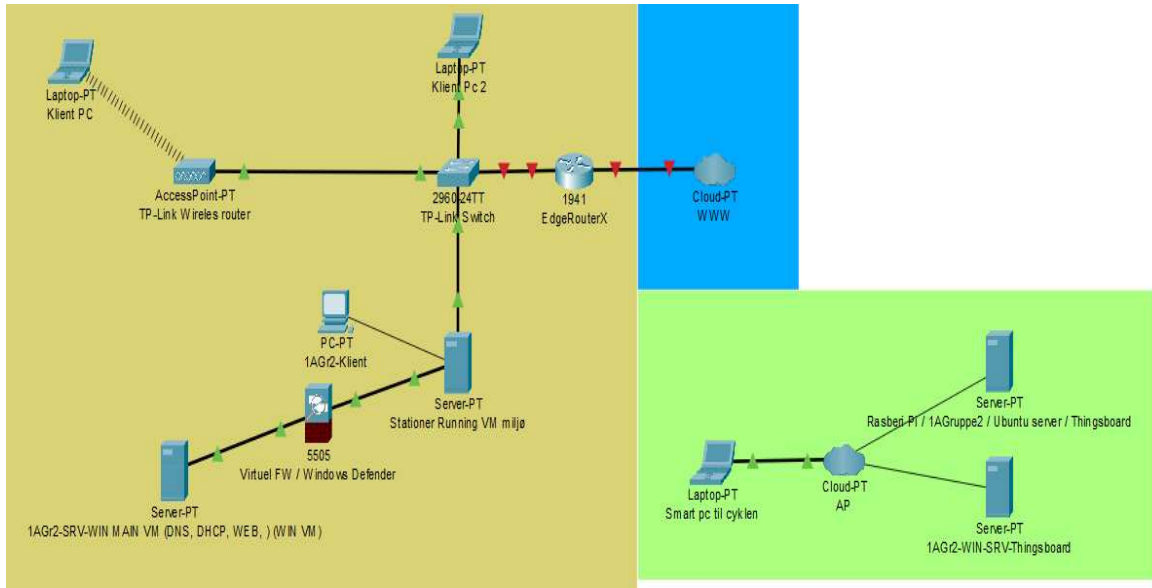
2 af de 3 server kører Thingsboard lokalt og vil have forbindelse med vores løsning igennem et ap og derved løse ID 15 og 16.

Den sidste server vil køre DHCP, sammen med DNS, ADDS og IIS, som vil køre vores hjemmeside og derved løse ID17.

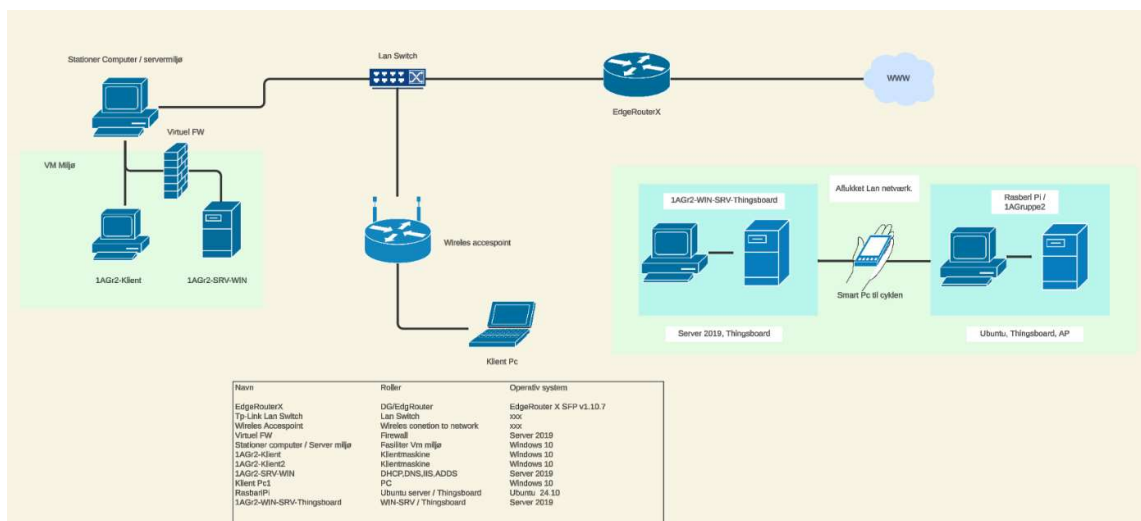
Routeren vil have en basal config, med inside og outside interfaces (Se ip-plan for mere info omkring interfaces). Vi har løst ID14 ved at køre en DHCP server på routeren og switchen vil ikke modtage nogle ændringer, da dens fabrik opsætning lever op til det arbejde den skal udføre.

Klientmaskinerne er placeret rundt i netværket strategisk, for at kunne fremvise at de forskellige løsninger fungerer efter hensigt.

Alle server i netværket kører med statiske IP-adresser og det samme gælder for to interfaces på routeren og tp-link (WAP). (Se ip-plan for mere information).



Figur 49: Netværkets topologi



Figur 50: Serverblok

8.1. Router/switch interfaceforbindelser

Device	Int	IP	Description
EdgeRouterX	eth0	xxx.xxx.xxx.xxx	Er ikke taget i brug
	eth1	xxx.xxx.xxx.xxx	Switch port mode
	eth2	192.168.145.2/25	Outside interface / Går til nettet (WWW)
	eth3	192.168.145.3/25	Inside interface / Går til TP-Link switch
	eth4	xxx.xxx.xxx.xxx	Er ikke taget i brug
	eth5	xxx.xxx.xxx.xxx	Er ikke taget i brug
TP-Link Switch	switch0	192.168.145.1/27	Ip for tilgang til router
	G0/1	xxx.xxx.xxx.xxx	Går til Router
	G0/2	xxx.xxx.xxx.xxx	Går til Wireles Accespoint
	G0/3	xxx.xxx.xxx.xxx	Går til VM Miljø
	G0/4	xxx.xxx.xxx.xxx	Er ikke taget i brug
	G0/5	xxx.xxx.xxx.xxx	Er ikke taget i brug

Tabel 1: Interface-tabel

Da alt konfiguration af routeren er gjort igennem GUI, vil der ikke være nogle commands at fremvise. For at kunne fremvise hvad vi har opsat på routeren, har vi lave et "show configuration" command for at se cli udgaven af hvad der er opsat på den.

interfaces {	description Local	hostfile -update disable
ethernet eth0 {	duplex auto	shared-network-name
address dhcp	speed auto	LAN1 {
description Internet	}	authoritative enable
duplex auto	ethernet eth4 {	disable
firewall {	description Local	subnet
in {	duplex auto	192.168.145.0/27 {
name WAN_IN	speed auto	default-router
}	}	192.168.145.2
local {	ethernet eth5 {	dns -server
name WAN_LOCAL	duplex auto	192.168.145.2
}	speed auto	lease 86400
}	}	start
speed auto	loopback lo {	192.168.145.14 {
}	}	stop
ethernet eth1 {	switch switch0 {	192.168.145.24
duplex auto	address	}
speed auto	192.168.145.1/27	system {
}	description Local	host-name ubnt
ethernet eth2 {	mtu 1500	login {
address	switch-port {	user ubnt {
192.168.145.2/25	interface eth1 {	authentication {
description Local	}	encrypted-
duplex auto	vlan -aware disable	<u>password</u>
speed auto	}	*****
}	service {	}
ethernet eth3 {	dhcp -server {	level admin
address	disabled false	}
192.168.145.3/25		

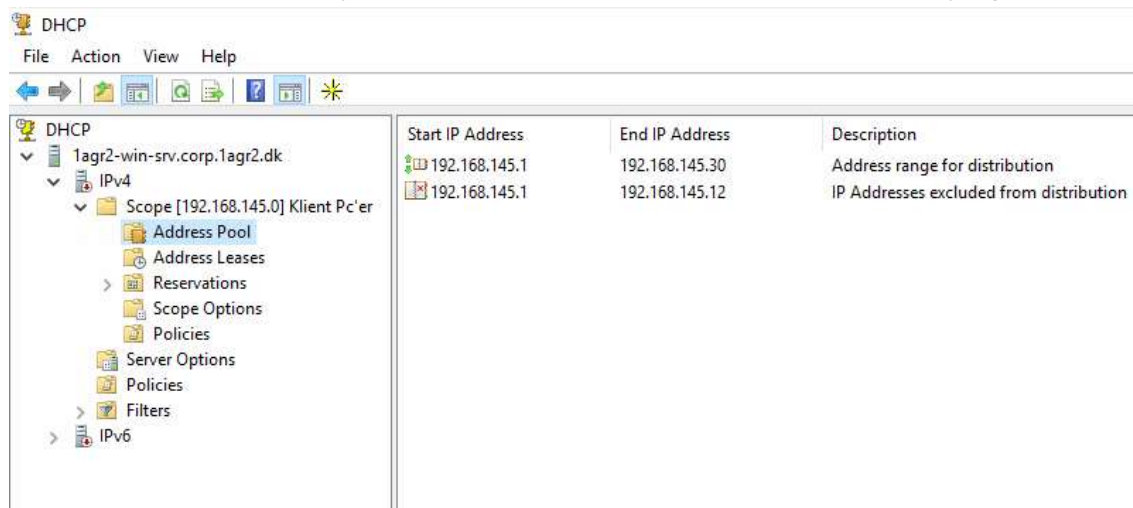
8.2. Konfiguration af valgte servere

På SRV1 har vi valgt at køre med Windows server 2019, som vi har lært at bruge igennem undervisningen. På denne server er installeret et userinterface, for nemmere administration af server. Serveren har 4 roller for netværket som er DHCP, DNS, ADDS og IIS

DHCP er sat op med et enkel scope for alle enhederne i netværket. Scopet er sat op sådan (Se billede for nede)

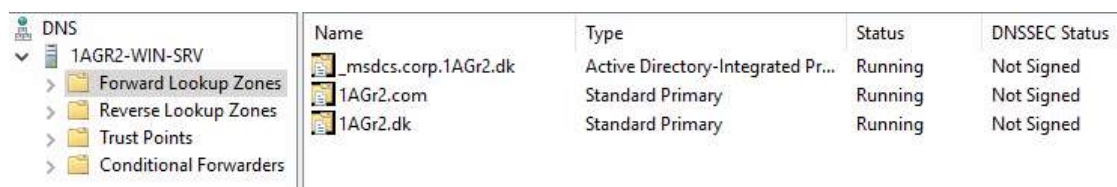
Scope: 192.168.145.1 - 30

Ekskluderede adresser fra scopet: 192.168.145.1 - 12 (Er reserveret til netværksudstyr og servere)



Figur 51: DHCP Manager

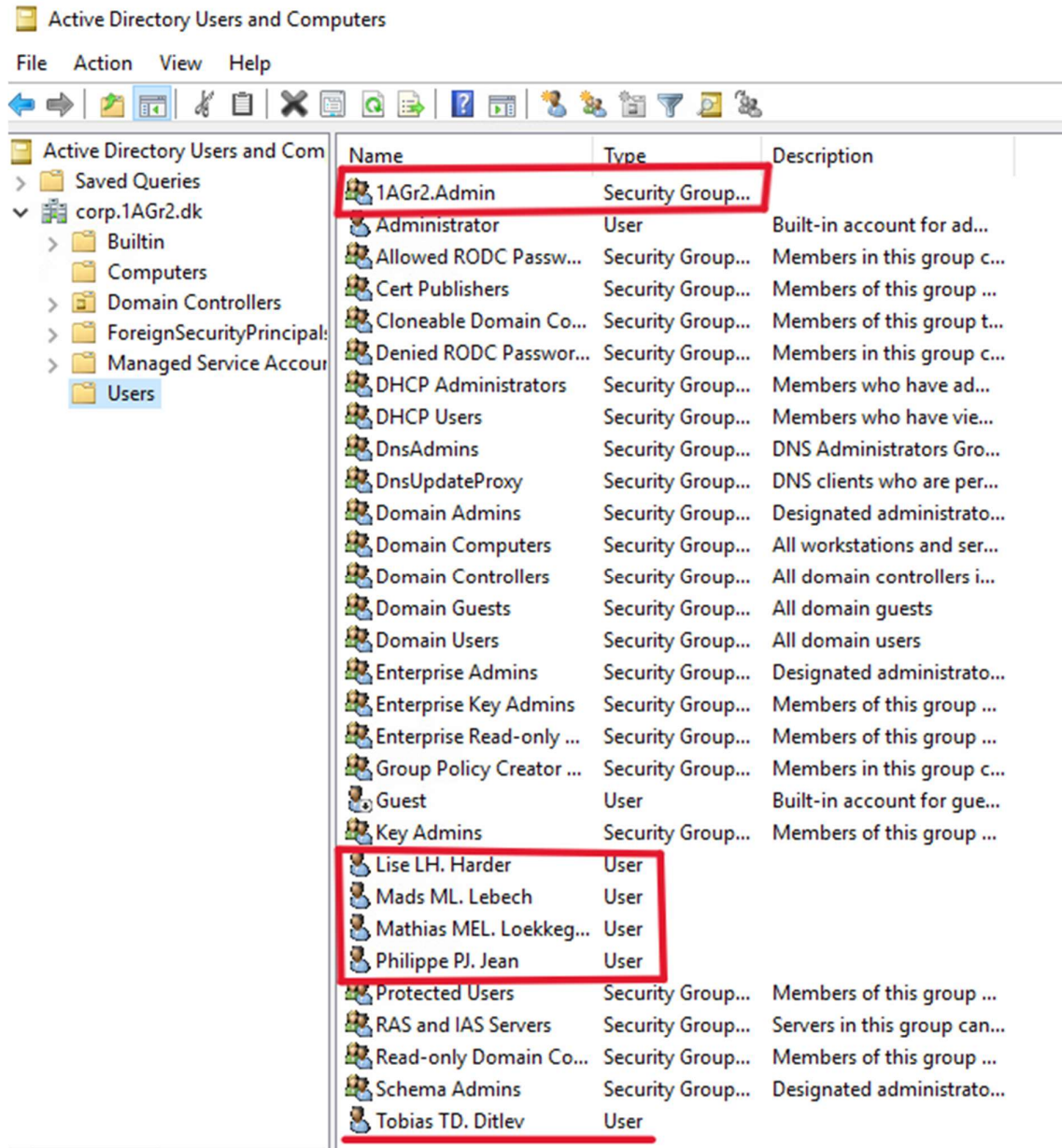
DNS er sat op med en forward lookup zone (Se billede for nede)



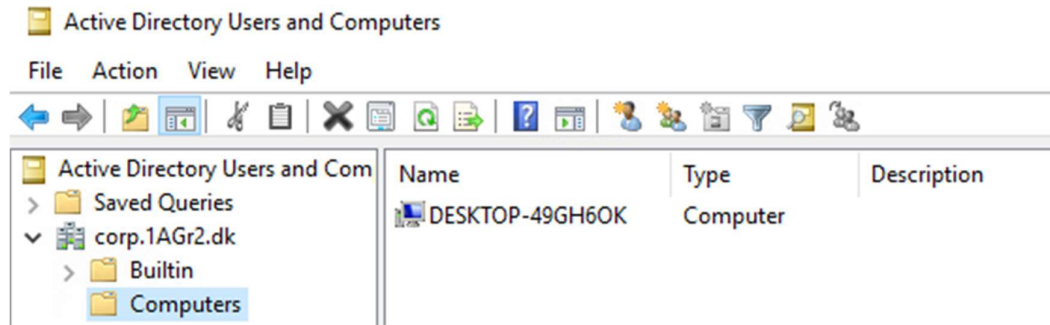
Figur 52: DNS Manager

ADDS er opsat med et Forest navn corp.1AGr2.dk og hvert gruppemedlem er oprettet i AD, sammen med en enkelt virtuel klientmaskine, som alle gruppemedlemmer kan logge in på med deres initialer og egen kode (Koden er den samme for alle bruger for at gøre det nemmere at teste (Kode: PasswOrd)).

Alle gruppemedlemmerne er sat i en admin gruppe for at kunne tildele dem gruppepolitikker og for nemmere administration.



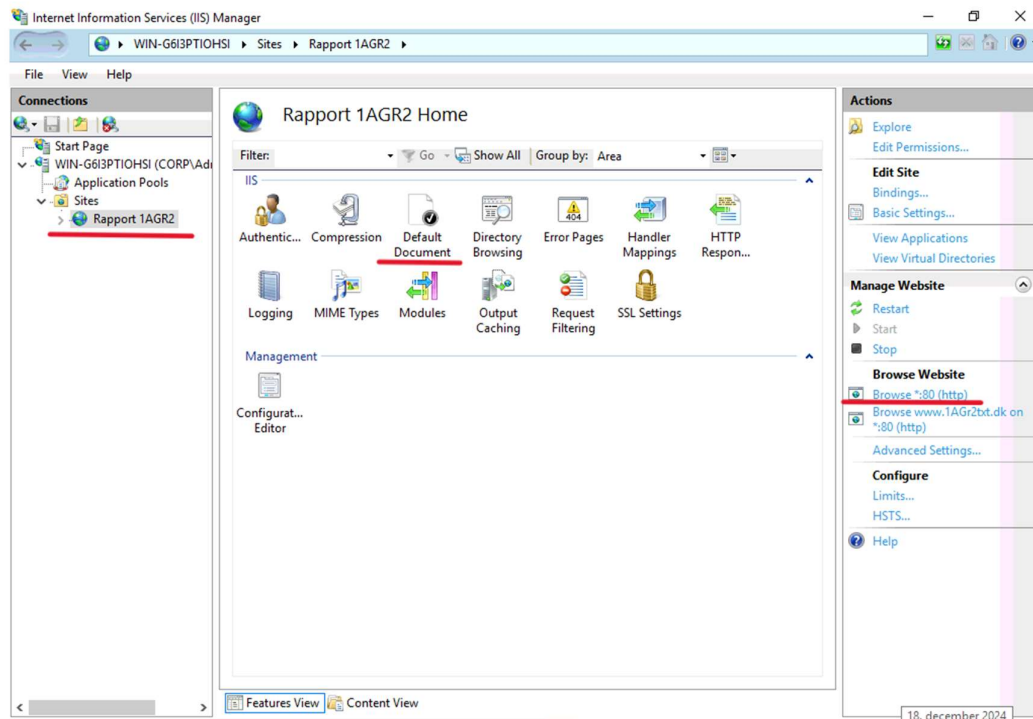
Figur 53: AD/Bruger



Figur 54: AD/Computere

IIS er opsat til at køre med en standard HTML hjemmeside som præsenterer vores løsning.

Under default doc... har vi sat vores HTML fil op til den lokale website på port *80 http



Figur 55: IIS Manager

HTML opsætningen

(Den fulde løsning er ikke sat ind endnu, men alle forudsætninger er der for at kravet er løst)

```

Rapport.html - Notepad
File Edit Format View Help
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
</style>
<title>Gruppe 2 klasse 1A Rapport</title>
</head>

<body>

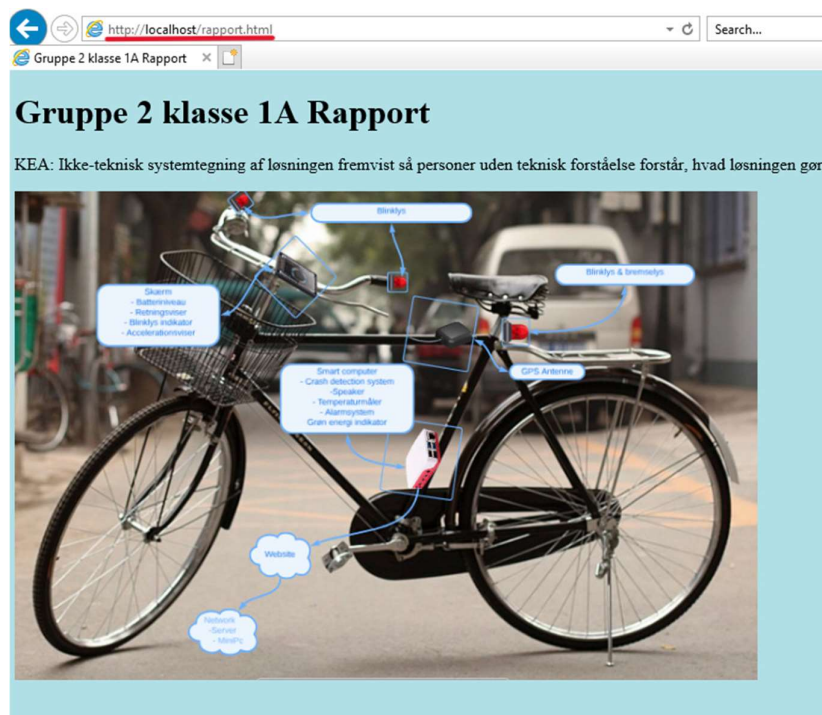
<h1>Gruppe 2 klasse 1A Rapport</h1>
<p>KEA: Ikke-teknisk systemtegning af løsningen fremvist så personer uden teknisk forståels


</html>

```

Figur 56: HTML kode

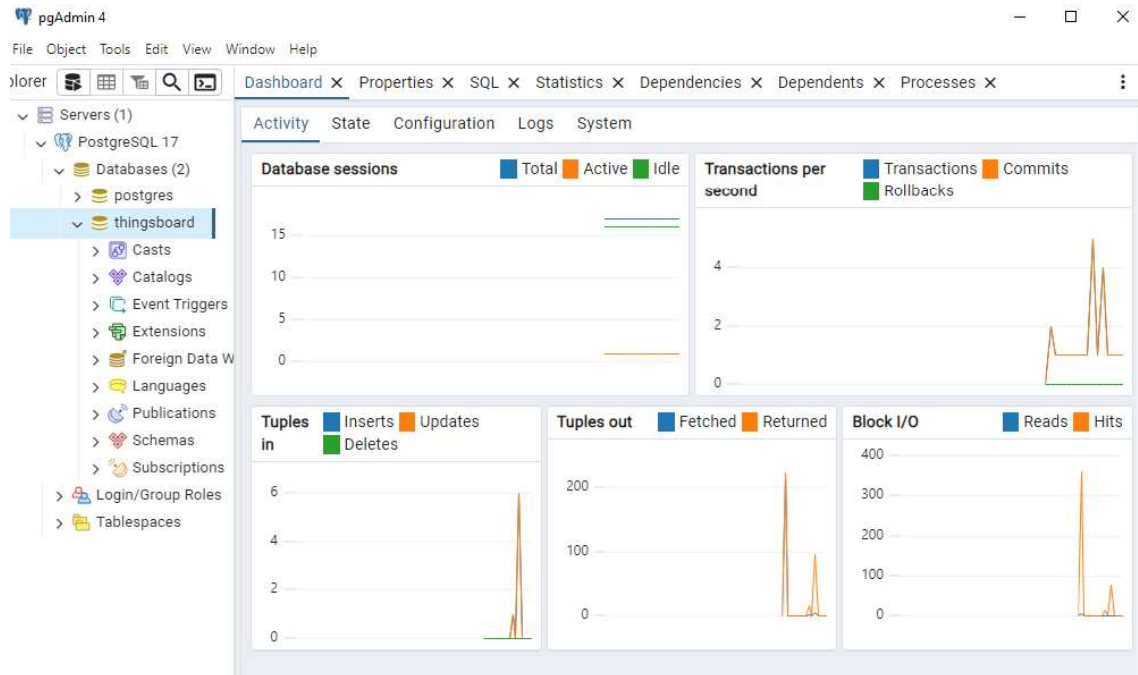
Selve html hjemmesidens udseende (Ikke det endelige udseende) (Highlightet lokaladressen)



Figur 57: Frontpage på website

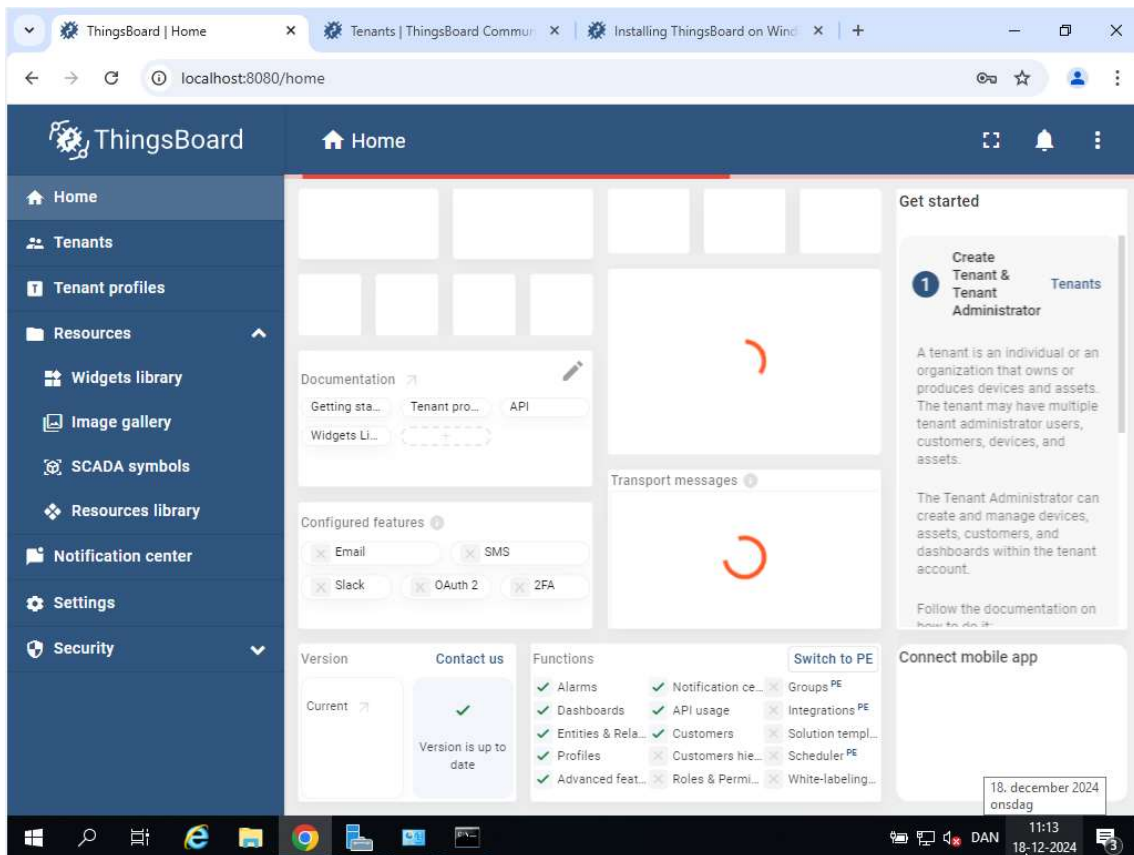
SRV-2 kører med Thingsboard sammen med en sql database lokalt på en server 2019 maskine, Under installationen har Thingsboard ikke oprettet de standard tenants bruger som skulle bruges til at udføre kravet til ende, vi formoder kraftigt at med mere tid at løsningen vil kunne komme op og køre til et tilfredsstillende niveau.

SQL Database (Databasen er oprette og klar til at opbevare data)



Figur 58: PGAdmin/SQL-database

Thingsboard (Installeret og klar til at blive konfigureret til at modtage data)



Figur 59: Thingsboard/windows server

Ubuntu Server på Raspberry pi 5 8 gb ram

Server- og netværksopsætning

Vi har valgt at bruge Ubuntu Server 24.10 på en Raspberry Pi 5 med 8 GB RAM. For at gøre det lettere at arbejde med systemet, har vi installeret en Desktop / Graphical User Interface (GUI) ovenpå Ubuntu Server. Det giver os mulighed for både at arbejde via terminalen og bruge en grafisk grænseflade, hvis vi har brug for det.

Java og ThingsBoard

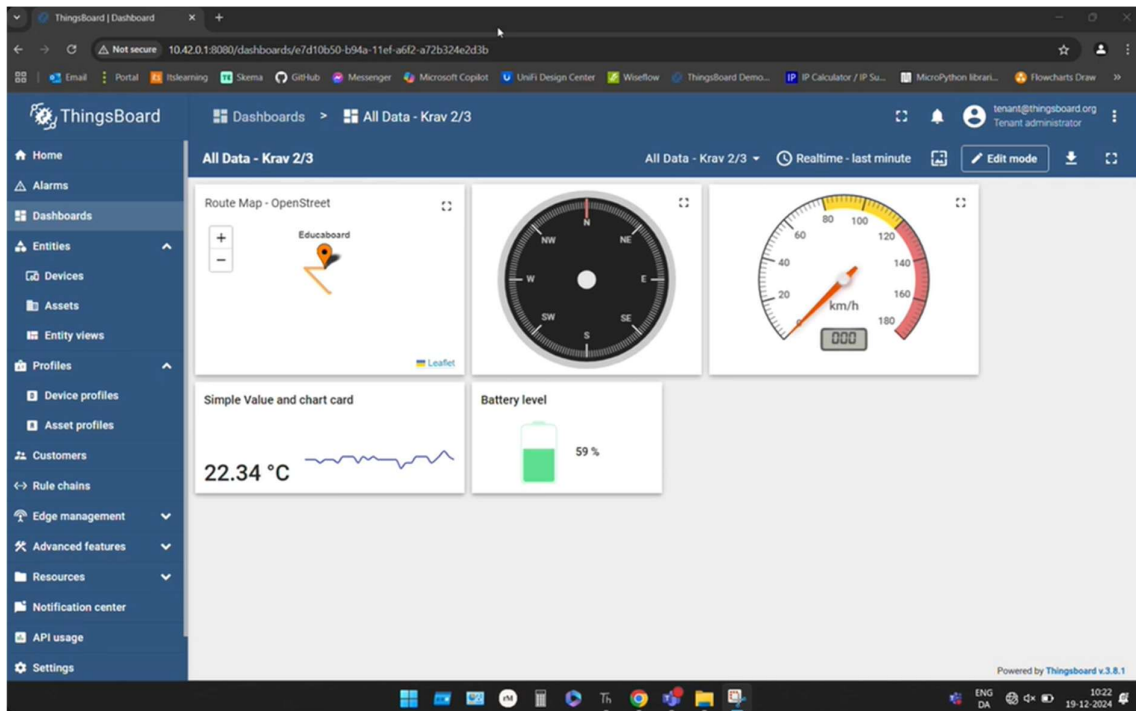
Da ThingsBoard kræver Java for at fungere, har vi installeret Java 17 (OpenJDK) på serveren. Vi fulgte ThingsBoards egen vejledning for at sikre, at alt blev sat korrekt op. Denne Java-version passer til de specifikationer, som ThingsBoard kræver og sikrer, at platformen kører stabilt.

```
export LOG_FILENAME=thingsboard.out
export LOADER_PATH=/usr/share/thingsboard/conf,/usr/share/thingsboard/extensions
export SQL_DATA_FOLDER=/usr/share/thingsboard/data/sql

# DB Configuration
export DATABASE_TYPE=sql
export SPRING_DATASOURCE_URL=jdbc:postgresql://localhost:5432/thingsboard
export SPRING_DATASOURCE_USERNAME=postgres
export SPRING_DATASOURCE_PASSWORD=Password
# Specify partitioning size for timestamp key-value storage. Allowed values: DAYS, MONTHS, YEARS, INDEFINITE.
export SQL_POSTGRES_TS_KV_PARTITIONING=MONTHS

# Update ThingsBoard memory usage and restrict it to 2G in /etc/thingsboard/conf/thingsboard.conf
export JAVA_OPTS="-Xms2G -Xmx2G"
agruppe2@1AGruppe2:~$ java -version
openjdk version "17.0.13" 2024-10-15
OpenJDK Runtime Environment (build 17.0.13+11-Ubuntu-2ubuntu124.10)
OpenJDK 64-Bit Server VM (build 17.0.13+11-Ubuntu-2ubuntu124.10, mixed mode, sharing)
agruppe2@1AGruppe2:~$
```

Figur 60: Java update & SQL-database



Figur 61: Thingsboard /Ubuntu

Databaseopsætning med PostgreSQL

Som database har vi valgt at bruge PostgreSQL, som også blev installeret i forhold til ThingsBoards officielle guide¹⁶. PostgreSQL fungerer som databasemotor til at gemme de data, der bliver sendt til ThingsBoard. Vi har oprettet en databasebruger med de nødvendige tilladelser og sørget for, at databasen er konfigureret til at køre på den lokale server.

```
valid_lft forever preferred_lft forever
agruppe2@1AGruppe2:~$ cat /etc/thingsboard/conf/thingsboard.conf
#
# Copyright © 2016-2024 The Thingsboard Authors
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

export JAVA_OPTS="$JAVA_OPTS -Dplatform=deb -Dinstall.data_dir=/usr/share/thingsboard/data"
export JAVA_OPTS="$JAVA_OPTS -Xlog:gc*,heap*,age*,safepoint=debug:file=/var/log/thingsboard/gc.log:time,uptime,level,tags:filecount=10,filesize=10M"
export JAVA_OPTS="$JAVA_OPTS -XX:+IgnoreUnrecognizedVMOptions -XX:+HeapDumpOnOutOfMemoryError"
export JAVA_OPTS="$JAVA_OPTS -XX:-UseBiasedLocking -XX:+UseTLAB -XX:+ResizeTLAB -XX:+PerfDisableSharedMem -XX:+UseCondCardMark"
export JAVA_OPTS="$JAVA_OPTS -XX:+UseG1GC -XX:MaxGCPauseMillis=500 -XX:+UseStringDeduplication -XX:+ParallelRefProcEnabled -XX:MaxTenuringThreshold=10"
export LOG_FILENAME=thingsboard.out
export LOADER_PATH=/usr/share/thingsboard/conf,/usr/share/thingsboard/extensions
export SQL_DATA_FOLDER=/usr/share/thingsboard/data/sql

# DB Configuration
export DATABASE_TS_TYPE=sql
export SPRING_DATASOURCE_URL=jdbc:postgresql://localhost:5432/thingsboard
export SPRING_DATASOURCE_USERNAME=postgres
export SPRING_DATASOURCE_PASSWORD=Password
# Specify partitioning size for timestamp key-value storage. Allowed values: DAYS, MONTHS, YEARS, INDEFINITE.
export SQL_POSTGRES_TS_KV_PARTITIONING=MONTHS

# Update ThingsBoard memory usage and restrict it to 2G in /etc/thingsboard/conf/thingsboard.conf
export JAVA_OPTS="$JAVA_OPTS -Xms2G -Xmx2G"
agruppe2@1AGruppe2:~$
```

Figur 62: Thingsboard config fil

¹⁶ <https://thingsboard.io/docs/user-guide/install/ubuntu/>

Netværksopsætning og adgangspunkt

Raspberry Pi'en er sat op som et lokalt access point. Det betyder, at vi kan forbinde enheder som vores ESP32 trådløst via Wi-Fi. På den måde kan ESP32'en sende data direkte til ThingsBoard-platformen, som kører på vores server. Dette gør det nemt at indsamle og visualisere data fra IoT-enheder.

```

agruppe2@1AGruppe2:~$ nmcli device show
GENERAL.DEVICE: wlan0
GENERAL.TYPE: wifi
GENERAL.HWADDR: 2C:CF:67:79:54:35
GENERAL.MTU: 1500
GENERAL.STATE: 100 (connected)
GENERAL.CONNECTION: Hotspot
GENERAL.CON-PATH: /org/freedesktop/NetworkManager/ActiveConnection/2
IP4.ADDRESS[1]: 10.42.0.1/24
IP4.GATEWAY: --
IP4.ROUTE[1]: dst = 10.42.0.0/24, nh = 0.0.0.0, mt = 600
IP6.ADDRESS[1]: fe80::1084:8962:9e1e:849/64
IP6.GATEWAY: --
IP6.ROUTE[1]: dst = fe80::/64, nh = ::, mt = 1024

```

Figur 63: Raspberry Pi/ AP show CLI

For at få det hele til at spille sammen, har vi givet Raspberry Pi'en en statisk IP-adresse, så vi altid kan tilgå serveren på samme adresse. Vi har også åbnet de nødvendige porte i firewall'en (f.eks. port 8080 til ThingsBoard), så vi kan tilgå ThingsBoard-dashboardet via en webbrowser fra andre enheder på netværket.

8.3. Teori af relevante netværk og server-blokke

Brugte teknologier: DHCP, DNS, ADDS, IIS, IP, Server VM (Ubuntu + Windows) TP-LINK TL-SG105 Switch (IEEE 802.3 / 5 x 1000Base-T RJ-45), Ubiquiti EdgeRouter X SFP – router (5-port switch (integreret) WAN / LAN: 5 x 10Base-T/100Base-TX/1000Base-T (PoE) - RJ-45, WAN / LAN: 1 x 10Base-T/100Base-TX/1000Base-T - SFP (mini-GBIC))

Server (WIN + Ubuntu), kan betegnes som en enhed, der giver funktionalitet og deler sine ressourcer med andre klienter, eksempelvis programmer eller computere.

En server kan tilbyde en række forskellige funktioner og disse bliver oftest omtalt som tjenester. En servers tjenester kan f.eks. dække over deling af ressourcer og data imellem klienter. Det er muligt for en enkel server at betjene flere klienter og samtidigt er det også muligt for en enkel klient at benytte flere servere og tjenester.

En server er kun så effektiv som dens opsætning og dens tildelte resurser, igennem eksamensperioden at vi ofte haft problemer med vores virtuel miljø i form at langsomme hastighederne, dette har begrænset hvor hurtigt og hvor effektivt vi har kunne udføre vores opsætning.

Serveren snakker med klienterne og giver dem adgang til dens ressourcer.

DHCP

DHCP, eller Dynamic Host Configuration Protocol, er en teknologi, der automatisk tildeler IP-adresser til enheder på et netværk. Det sikrer, at hver enhed får en unik adresse, hvilket gør kommunikationer mere effektive og fjerner behovet for manuelt at konfigurere hver enheds IP-indstillinger. (OSI-Model: L7)

DHCP har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Statiske adresse konflikter
2. Single point of failure (I vores tilfælde)
3. Rogue DHCP-servere
4. Man-in-the-middle-angreb
5. Dårlig opsætning af lease tider

Protokol og port: UDP, Port 67(Server) 68(klient)¹⁷

DHCP, servicen på WIN-SRV snakker med klienterne og uddeler ip-adresser til alle klienterne på netværket.

¹⁷ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

DNS

DNS, (domain name system) fungerer sådan at når en bruger indtaster en URL i sin webbrowser, går DNS i gang med at forbinde URL'en med IP-adressen på den aktuelle server. Dette kaldes DNS-navneopløsning og indebærer, at DNS-recursor forespørger forskellige navneservere for at finde ud af serverens faktiske IP-adresse. (OSI-Model: L7)

DNS har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Single point of failure (I vores tilfælde)
2. Man-in-the-middle-angreb
3. DDoS-angreb

Protokol og port: UDP/TCP, port 53¹⁸

DNS snakker med AD DS og Klienterne, hvor den indeholder en liste med hvem og hvor DC'en ligger, sammen med andet der skal oversættes fra navn til ip eller vice versa.

AD DS

AD DS, Active Directory Domain Services, giver metoderne til lagring af adressebogsdata og gør disse data tilgængelige for netværksbrugere og administrator. AD DS gemmer f.eks. oplysninger om brugerkonti, såsom navne, adgangskoder, telefonnumre og så videre og gør det muligt for andre autoriserede brugere på det samme netværk at få adgang til disse oplysninger. (OSI-Model: L7)

AD DS har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Kompleks design
2. Single point of failure (I vores tilfælde)

Protokol og port: LDAP, Kerberos, RPC, SMB, DNS, Port 386, 88, 135, 445¹⁹.

AD DS, snakker med klienterne der i dens domain og med DNS med dens "forward lookup zone"

¹⁸ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

¹⁹ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

IIS

IIS, Internet Information Services er en Microsoft web server service som presenter HTML/TXT kode/filer i læsbart format til uddeling af informationer på Lan eller Wan netværk. (OSI-Model: L7)

IIS har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Ressourceforbrug
2. Større mål for angreb
3. Kompleks konfiguration
4. Kompleks administration af større miljøer
5. Kompleks fejlfinding

Protokol og port: HTTP, port 80.²⁰

IIS'en snakker med klienterne der tilgår dens HTML hjemmeside

IPv4

IPv4, en IP-adresse er et unikt nummer, som bliver brugt til at identificere din computer, mobil, tablet osv. når du bruger internettet. Din IP-adresse fungerer lidt som din enheds telefonnummer og giver andre enheder på internettet mulighed for at kommunikere med dine enheder. En ip består af en 32 bit værdi, fordelt på 4 oktetter (eksempel: 192.168.145.4: 11000000.10101000.10010001.00000100) og køre på Layer 3 på OSI-Modellen.

IPv4 har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Begrænsede Ip adresser
2. Ingen interne sikkerhedsprotokoller

²⁰ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

Router

Router, en router i netværket har til arbejde at skabe forbindelse imellem LAN- og WAN-udstyr og holde styr på de forskellige resurser på netværket. Routeren bruger netværksprotokoller som Ethernet (IEEE802.3) som har udviklet sig over årene og kan køre med hastigheder op til 1000Base-T/ Giga bit Ethernet, en router arbejder primær på Layer 3 i OSI-modellen, men har ofte switch-porte på sig (som arbejder med Layer 2 i OSI-modellen), afhængige af størrelsen og modellen af routeren.

Protokoller: ICMP, ARP, TCP, UDP og IP²¹

Router har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. begrænset kapacitet
2. Latency
3. Tabelstørrelse
4. Komplekse routnings protokol opsætning
5. Komplekse opsætninger af udstyr
6. Single point of failure (I vores tilfælde)

Protokoller: IP, ARP, ICMP, DHCP Relay²²

Routeren snakker med alt på netværket da den indeholder Default Gateway og forbindelsen ud til nettet.

²¹ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

²² <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

Switch

Switch, en switch laver en MAC-adressetabel over alle forbundet maskiner med port nummer, når der så bliver sendt data til en computer på netværket ved den hvilken port den er på og kan derved starte unicast-transmission, collision domain isolation og fuld-duplex kommunikation (I nogle tilfælde så vil der også blive oprettet Flow control for at håndtere data flow). En switch arbejder primær på layer 2 i OSI modellen, men nogle switche har moduler i sig som gøre den kan arbejde småt på layer 3

En switch har nogle vigtige begrænsninger som en tekniker bliver nødt til at holde i tankerne,

1. Broadcast-Storm
2. MAC-table overflow angreb
3. looping

Protokoller: ARP, STP, VLAN(802.1Q)²³

Switchen snakker sammen med alle devices den er forbundet til den på netværket og transporter data rundt.

Diverse teknologier der er beskrevet og snakket om ovenover er valgt på baggrund af de krav og udfordringer der er sat for vores gruppe til at løse (For beskrivelse af krav se kapitel 9).

²³ <https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

9. Egenskaber, krav, testmetode og resultater

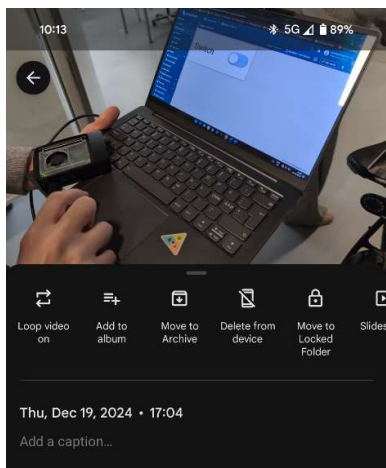
9.1. Krav, prioritet, accepttestprocedure og resultater

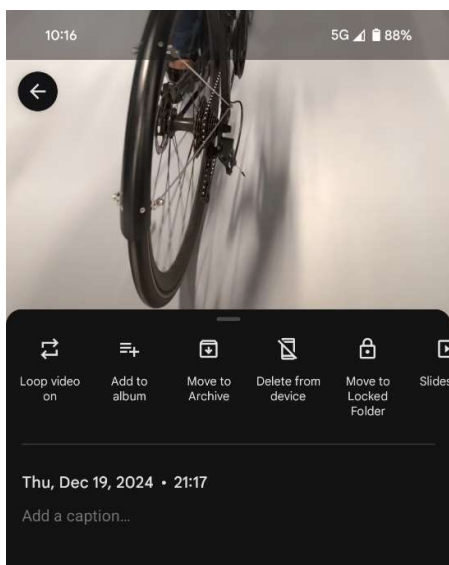
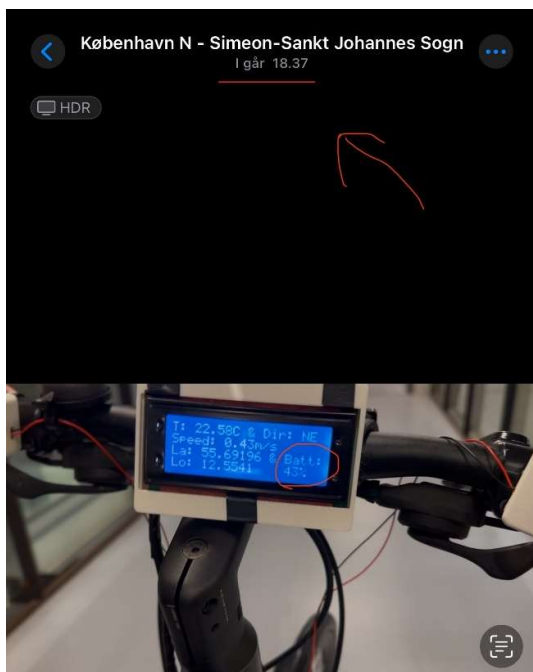
Test af krav 1: Kort overskrift

ID	Krav	Prioritet
ID: 1	Løsningen skal køre på batteri og skal som minimum kunne fungere i en time. Samtidig skal batteristatus, resterende kapacitet i procent, vises.	1
Kategori Batteri	Accepttest 1) Batteri skal være fuldt opladet 2) Systemet tændes i almindelig brugstilstand og brugeren udfører en "repræsentativ brug" af sin cykel. Der er op til udviklergruppen at finde frem til, hvad "repræsentativ brug" er 3) Undervejs observeres batteristatus, på konsollen, der skal ligge inden for 10 % nøjagtighed 4) Efter en times brug noteres batteristatus Hvis batteristatus i 4) er mere end 0 % er kravet opfyldt.	Godkendt

Link til video af testen

<https://www.youtube.com/watch?v=NF6e0YArCHc>





I billederne ovenfor vises klokkeslæt for video af første test kl. 17:04, på første billede (her var batteriet fuldt opladt.

Andet billede viser en anden test optaget kl. 18:37 (1 time og 33 minutter senere) med 43% batterikapacitet tilbage.

Vi testede samlet set over 4 timer med vores batterier som strømforsyning. (det kan ses på billede 3)

ID 2	Krav På den lokale konsol, Educaboard, skal vises hastighed, retning, længde- og breddegrad, batteristatus og temperatur	Prioritet 1
Kategori UX	Accepttest 1) Batteri skal være fuldt opladet 2) Løsningen tændes 3) Konsollen observeres samtidig med, at brugeren cykler rundt i fem minutter. Imens observeres hastighed, retning, længde- og breddegrad og sammenlignes med identiske data fra en mobiltelefon. Hvis data for smartcyklen er inden for 5 % af mobiltelefonen, er kravet opfyldt.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=mDh8ccRmBxs https://www.youtube.com/watch?v=OF0lvV8KHpE		
Diskussion og evt. ønskede ændring af krav		

ID 3	Krav Løsningen skal løbende sende data til ThingsBoard, der skal vise hastighed, retning, længde- og breddegrad, batteristatus og temperatur når cyklen er i brug. Hvis andet data måles skal det også præsenteres på Thingsboard.	Prioritet 1
Kategori Internet	Accepttest 1) Batteri skal være fuldt opladet 2) Løsningen tændes 3) Konsollen observeres samtidig med, at brugeren cykler rundt i fem minutter. Imens observeres hastighed, retning, længde- og breddegrad og sammenlignes med identiske data fra en mobiltelefon. Hvis anden data måles, skal det også vises på Thingsboard. Hvis data for smartcyklen er inden for 5 % af mobiltelefonen, er kravet opfyldt.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=GoP_MnkLIfE https://www.youtube.com/watch?v=OF0lvV8KHpE		
Diskussion og evt. ønskede ændring af krav		

ID 4	Krav Når cyklen har stået stille i mere end 3 minutter og brugeren ikke har slukket permanent for løsningen, sendes besked med cyklens placering til Thingsboard, hvis cyklen kommer i bevægelse	Prioritet 1
Kategori Internet	Accepttest 1) Batteri skal være fuldt opladet 2) Løsningen tændes 3) En kort tur på 1 minut køres 4) Cyklen stilles, men løsningen slukkes ikke og tiden nulstilles 5) Når der ikke er registreret bevægelse på cyklen i 3 minutter fra 3) skal der inden for 30 sekunder komme en besked på Thingsboard hvis cyklen kommer i bevægelse og positionen sendes hvert tiende sekund til Thingsboard. Sker dette er kravet opfyldt.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=emqBWYmtysw		
Diskussion og evt. ønskede ændring af krav		

ID 5	Krav Løsningen bør gøres robust ift. det miljø den skal anvendes i. I prototypen vil det være tilstrækkeligt at det er i et 3D printet design	Prioritet 2
Kategori Robusthed	Accepttest 1) Batteri skal være fuldt opladet 2) Løsningen tændes 3) Løsningen bruges repræsentativt i totalt tre timer, enten i sammenhæng eller over flere perioder 4) Før løsningen tages i brug, tages et billede af alle løsningens detaljer på cyklen og evt. andet udstyr. Efter de tre timers brug tages igen billeder af de samme detaljer. Er alt intakt er kravet bestået.	Godkendt

Billeder fra efter testen

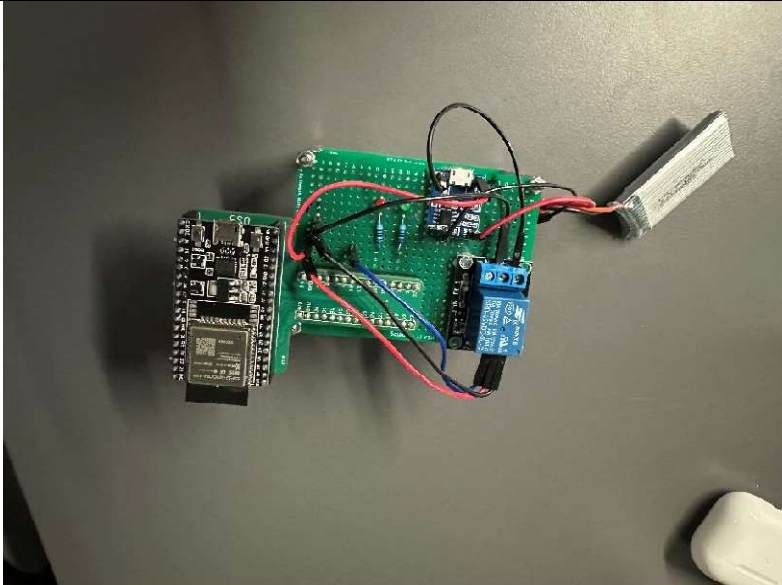


Diskussion og evt. ønskede ændring af krav

Vi fik udviklet evner til at lave 3D modeller i Autodesk Fusion, hvilket var vores måde at lave skræddersyet indpakning til vores enheder. Vores print blev dog ikke perfekte, da der var forskellige ting vi ikke havde tænkt over skulle være i designet og da køen til 3D-printerne var alt for lange. Så vi blev nødt til at give den huller og andre modifikationer efter, noget som vi helst ville have haft ordnet in-print. Det er inklusiv at ville kunne gøre den vejr-tæt, noget den ikke er nu. Ud over det er printene robuste og kan holde til hvad de skal.

ID 6	Krav Bremselys skal lyse når der bremses over en valgfri negativ acceleration	Prioritet 2
Kategori Signallering	Accepttest 1) Batteri skal være fuldt opladet 2) Løsningen tændes 3) Når der bremses over en valgfri negativ acceleration, skal et bremselys tændes. Sker det, er kravet bestået.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=8Uv-azKmiR8 		
Diskussion og evt. ønskede ændring af krav Grundet hardware fejl med Neopixel ringe virkede bremselyset ikke på cyklen (ligesom med blinklys) derfor er det monteret som på videoen. Dette er snakket med NAFO omkring (Neopixel ring problemer).		

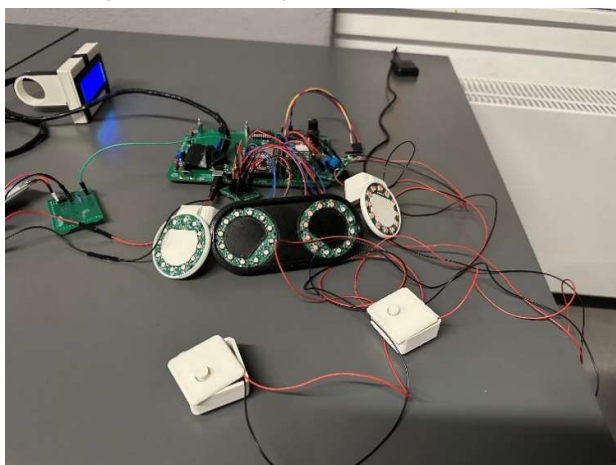
ID 7	Krav Visning af den forventede resttid løsningen kan bruges. Visningen skal være på lokal konsol, Educaboard og på Thingsboard	Prioritet 3
Kategori UX	Accepttest 1) Batteri skal være 50 % opladet 2) Løsningen tændes 3) Visningen af restlevetiden, på konsollen og Thingsboard, skal stemme med den beregnede restlevetid. Denne beregnes ud fra det gennemsnitlige strømforbrug, målt over tre timer og sættes i relation til den målte batteriprocent.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=NF6e0YArCHc		
Diskussion og evt. ønskede ændring af krav		

ID 8	Krav Løsningen skal være bæredygtigt og opladning af løsningens batteri skal lades med grøn energi. Der skal indikeres på løsningen når der er grøn energi til rådighed på elnettet og hvornår der ikke er grøn energi til rådighed. Løsningen skal tænde for laderen når der er grøn energi og slukke når det ikke er grøn energi til rådighed.	Prioritet 2
Kategori Grøn opladning	Accepttest Step 1) Kør test programmet Step 2) Simulér at løsningen går over og under tærsklen for om der er grøn energi 10 gange. Hvis løsningen indikerer at der er grøn energi og tænder ladekredsen og løsningen indikerer når der IKKE er grøn energi og slukker ladekredsen 9 ud af 10 gange vurderes kravet som opfyldt.	Godkendt
 https://www.youtube.com/watch?v=7ISyulMt8qc		
Diskussion og evt. ønskede ændring af krav I videoen har vi simuleret energidataservices Grøn strøm API vha. at generere tilfældige tal i Python og fastsat en grænse for hvornår der var grøn strøm om hvornår der ikke var. Dette har vi gjort for at vise at opladeren fungerer efter hensigten da det kan tage flere dage mellem der er skift i den "rigtige verden"		

ID 9	Krav Man skal kunne sende en besked til løsningen fra Thingsboard så den aktiverer eller deaktiverer et alarmsystem så den blinker kraftigt med lys og giver lyd fra sig hvis den rykker sig når alarmen er slået til.	Prioritet 3
Kategori Alarm	Accepttest Step 1) Start testprogrammet. Step 2) Send besked til løsningen om at aktivere alarmen. Vent 30 sekunder uden at flytte løsningen og i denne periode skal alarmen IKKE udløses. Prøv nu at flytte løsningen og bekræft at alarmen går i gang ved at blinke lys og give lyd fra sig. Step 3) Send besked til løsningen om at deaktivere alarmen og prøv at flytte løsningen og kontroller at alarmen ikke udløses når cyklen flyttes. Hvis accepttesten kan udføres 8 ud af 10 gange uden fejl vurderes kravet som godkendt.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=vkHxOlwfi-M		
Diskussion og evt. ønskede ændring af krav		

ID Blinklys	Krav Der skal være blinklys på både styr og bag på cyklen så der kan blinkes til både højre og venstre.	Prioritet 1
Kategori Blinklys	Accepttest 1) Knapper kobles med ledninger til blinklys både på styr og bagved. 2) Knapper kobles med ledninger til ESP32. 3) ESP32 kobles med ledninger til en strømforsyning. 4) Lys i højre side skal blinke til højre både foran og bagved, når der trykkes på knappen på højre side af styret. Når der trykkes på knappen igen, stopper blinklyset. Lys i venstre side skal blinke til venstre både foran og bagved, når der trykkes på knappen på venstre side af styret. Når der trykkes på knappen igen, stopper blinklyset.	Godkendt

Billede og link til video af testen



<https://www.youtube.com/watch?v=XFjrmCrQQNs>

Diskussion og evt. ønskede ændring af krav

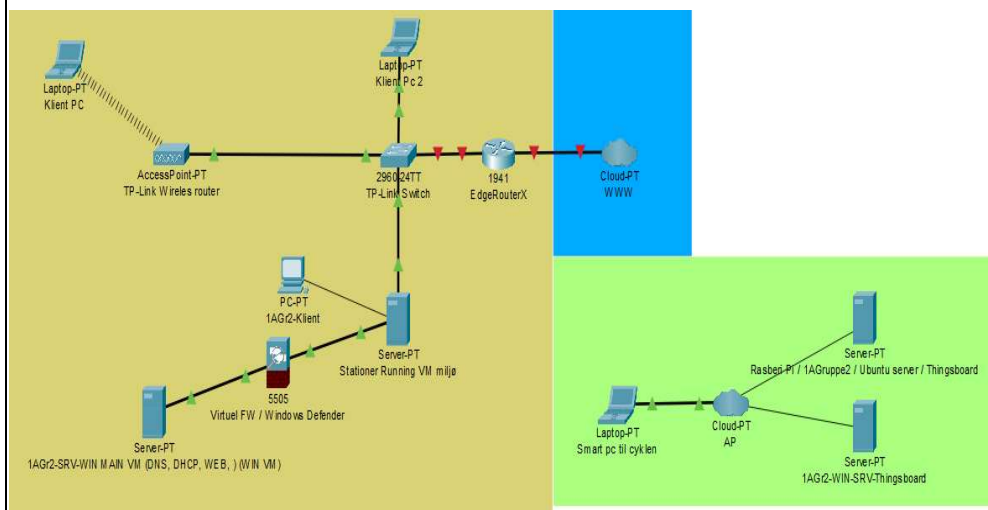
Det var desværre ikke muligt at vise blinklysene monteret på cyklen grundet hardware problemer med Neopixel ringene.

Efter aftale med NAFO har vi taget en video af blinklysene med kortere ledning på så vi kunne vise funktionaliteten.

Problematikken ligger med kablerne, hvor vi har afprøvet kabler med modstande, men uden held. Vi har prøvet med netværks kabler (kobber ledere) og samme problem. Dette tages med til perspektivering.

ID 11	Krav Når accelerometer registrerer et styrt vil der blive vist en besked på lcd skærmen, at et styrt er registreret. Herefter vil en nedtælling på 10 sekunder begynde så man har mulighed for at annullere registreringen i tilfælde af at dette var en fejl og ikke et uheld. Annullering sker ved at trykke på begge knapper på styret på samme tid. Bliver der ikke trykket vil alle 4 blinklys blinke synkront 1 gang i sekundet indtil systemet nulstilles.	Prioritet 1
Kategori Havariblink	Accepttest 1) Start test program 2) Styrt registreres ved at accelerometer måling overskrider given grænse. 3) Besked omkring styrt, samt 10 sekunders nedtælling vises på skærm. 4) Registrering skal kunne annulleres ved at trykke på begge knapper samtidigt. 5) Bliver annulleringen ikke registeret, vil alle 4 blinklys begynde at blinke med 1 sekunds mellemrum indtil systemet nulstilles.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=yImdv9BNxDU		
Diskussion og evt. ønskede ændring af krav		

ID 12	Krav Gruppen Justerer deres tidligere designede netværk ud fra følgende specifikationer: Ip adresser jf. RFC 1918 til Router/default gateway, en hardware firewall, et trådløst Access punkt (WAP) 10 servere, 10 printere og 30 klienter og udvidelsesmulighed på 10-30% -med hvad der kommer af ændringer i IoT-projektet	Prioritet 1
Kategori Fysisk netværksopsætning	Accepttest Der forelægger fulddokumentation i form af: - IP-plan - Adressetabel - Diagram over fysisk topologi -Topologi udført i Packet tracer Hvis dette er til stede, er det accepteret.	Godkendt



Device	Int	IP	Description
EdgeRouterX	eth0	xxx.xxx.xxx.xxx	Er ikke taget i brug
	eth1	xxx.xxx.xxx.xxx	Switch port mode
	eth2	192.168.145.2/25	Outside interface / Går til nettet (WWW)
	eth3	192.168.145.3/25	Inside interface / Går til TP-Link switch
	eth4	xxx.xxx.xxx.xxx	Er ikke taget i brug
	eth5	xxx.xxx.xxx.xxx	Er ikke taget i brug
TP-Link Switch	switch0	192.168.145.1/27	Ip for tilgang til router
	G0/1	xxx.xxx.xxx.xxx	Går til Router
	G0/2	xxx.xxx.xxx.xxx	Går til Wireless Accespoint
	G0/3	xxx.xxx.xxx.xxx	Går til VM Miljø
	G0/4	xxx.xxx.xxx.xxx	Er ikke taget i brug
	G0/5	xxx.xxx.xxx.xxx	Er ikke taget i brug

1 A GR.2

IP Plan		Kommentarer
Net ID	192.168.145.0/27	
Subnetmask	255.255.255.224/27	
Firewall (1)	192.168.145.4	
Router	192.168.145.2-3	Inside and outside interface
Servere (2)	192.168.145. 4	DHCP, DNS, WEB, AD DS
Wireless Access Point (1)	192.168.145.6	
Frie Adresser	192.168.145.9 - 12	Til eventuel udvidelse
Klienter	192.168.145.13 - 30	
Broadcast	192.168.145.31	

Adresse Tabel		Kommentarer
192.168.145.0/27	Net ID	
192.168.145.1		
192.168.145.2	Router 1 Outside interface	
192.168.145.3	Router 1 Inside interface (DG)	Default Gateway
192.168.145.4	SRV1	DNS, DHCP, WEB, FW, Windows VM
192.168.145.5	Frie adresser	Til eventuel udvidelse i fremtiden
192.168.145.6	Wireless Access Point	TCP Wifi Router
192.168.145.7	Ubuntu Server VM	Lokalt ThingsBoard
192.168.145.8	WIN Server 2019	Lokalt ThingsBoard
192.168.145.9	Frie adresser	Til eventuel udvidelse i fremtiden
192.168.145.10	Frie adresser	Til eventuel udvidelse i fremtiden
192.168.145.11	Frie adresser	Til eventuel udvidelse i fremtiden
192.168.145.12	Frie adresser	Til eventuel udvidelse i fremtiden
192.168.145.13	Klienter	Til DHCP fordeling
192.168.145.14	Klienter Lise	Fysisk hardware pc'er til krav 13
192.168.145.15	Klienter Tobias	Fysisk hardware pc'er til krav 13
192.168.145.16	Klienter Mads	Fysisk hardware pc'er til krav 13
192.168.145.17	Klienter Philippe	Fysisk hardware pc'er til krav 13
192.168.145.18	Klienter Mathias	Fysisk hardware pc'er til krav 13
192.168.145.19	Klienter	Til DHCP fordeling
192.168.145.20	Klienter	Til DHCP fordeling
192.168.145.21	Klienter	Til DHCP fordeling
192.168.145.22	Klienter	Til DHCP fordeling
192.168.145.23	Klienter	Til DHCP fordeling
192.168.145.24	Klienter	Til DHCP fordeling
192.168.145.25	Klienter	Til DHCP fordeling
192.168.145.26	Klienter	Til DHCP fordeling
192.168.145.27	Klienter	Til DHCP fordeling
192.168.145.28	Klienter	Til DHCP fordeling
192.168.145.29	Klienter	Til DHCP fordeling
192.168.145.30	Klienter	Til DHCP fordeling
192.168.145.31	Broadcast	

Adresse Tabel Hotspot/Pi.		Kommentarer
10.42.0.0/24		
255.255.255.0/24	Subnetmask	
10.42.0.1	Rasber Pi	Default Gateway
10.42.0.2	Klienter	Til DHCP fordeling
10.42.0.3	Klienter	Til DHCP fordeling
10.42.0.4	Klienter	Til DHCP fordeling
10.42.0.5	Klienter	Til DHCP fordeling
10.42.0.6	Klienter	Til DHCP fordeling
10.42.0.7	Klienter	Til DHCP fordeling
10.42.0.8	Klienter	Til DHCP fordeling
10.42.0.9	Klienter	Til DHCP fordeling
10.42.0.10	Klienter	Til DHCP fordeling
10.42.0.11	Klienter	Til DHCP fordeling
10.42.0.12	Klienter	Til DHCP fordeling
10.42.0.13	Klienter	Til DHCP fordeling
10.42.0.14 - 255	Klienter	Til DHCP fordeling
10.42.0.256	Broadcast	

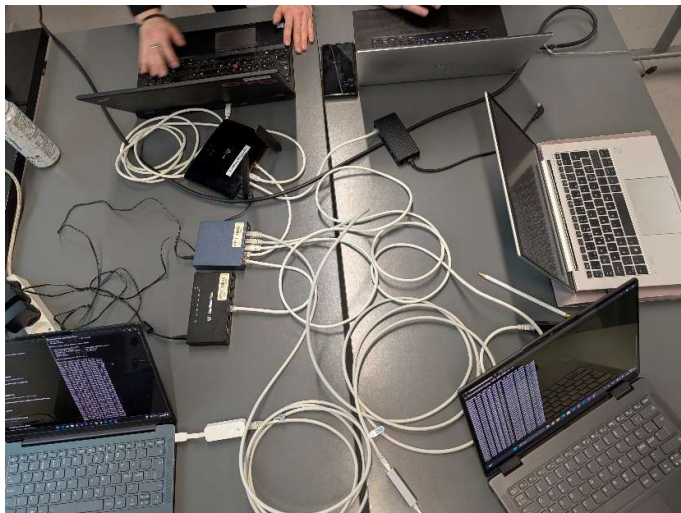
TP-Wireles accespoint	LAN1	xxx.xxx.xxx.xxx
	LAN2	xxx.xxx.xxx.xxx
	LAN3	xxx.xxx.xxx.xxx
	LAN4/WAN	192.168.145.6/27

Diskussion og evt. ønskede ændring af krav

ID 13	Krav Der opsættes et fungerende fysisk netværk indeholdende Router switch, Wap/WiFi router og gruppeditagerenes laptops og egenproducerede patchkabler	Prioritet 1
Kategori Fysisk netværksopsætning	Accepttest - Der er anvendt IP-adresser jf. egen IP-plan - Alle aktive enheder kan pinges, uden sikkerhedsforanstaltning aktiveret. - Kan ikke pinges med sikkerhed aktiveret.	Godkendt

Billede og link til video af testen

The screenshot displays a Wireshark capture of network traffic on the interface \Device\NPF_{0E3F8DFF-CD62-4D6F-8C56-E26F189E595A}. The packet list shows several ICMP Echo (ping) requests and replies. The packet details pane for packet 3140 shows the Ethernet II, Internet Protocol Version 4, and Internet Control Message Protocol (ICMP) layers. The ICMP layer shows an Echo (ping) request with ID 0x0001, sequence number 891, and TTL 128. The packet bytes pane shows the raw data of the ICMP request. A command prompt window is overlaid on the right, showing the results of a ping command: 'ping 192.168.145.20 -t'. The command prompt shows multiple successful replies from 192.168.145.20: bytes=32 time=2ms TTL=128.

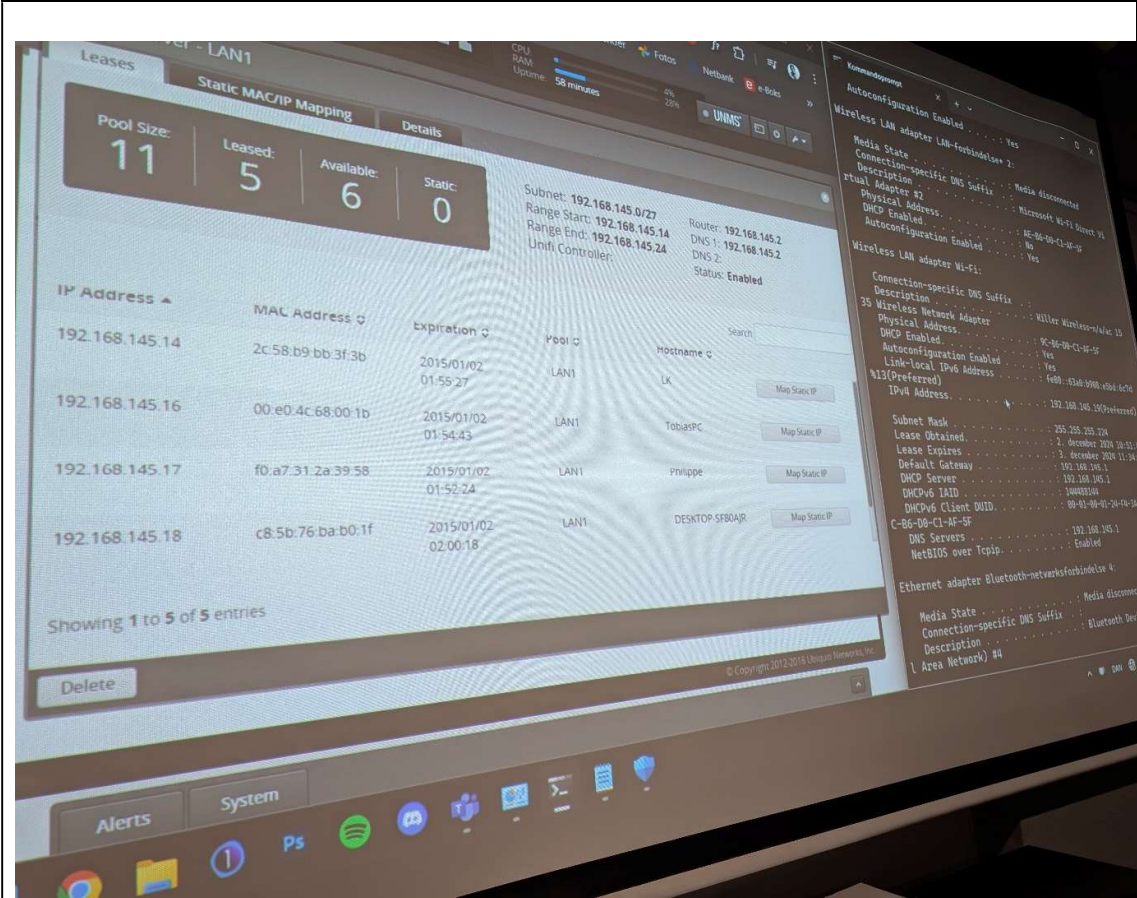


<https://www.youtube.com/watch?v=E-AQqXV-Uo0>

<https://www.youtube.com/watch?v=KNf4Xzep8c>

Diskussion og evt. ønskede ændring af krav

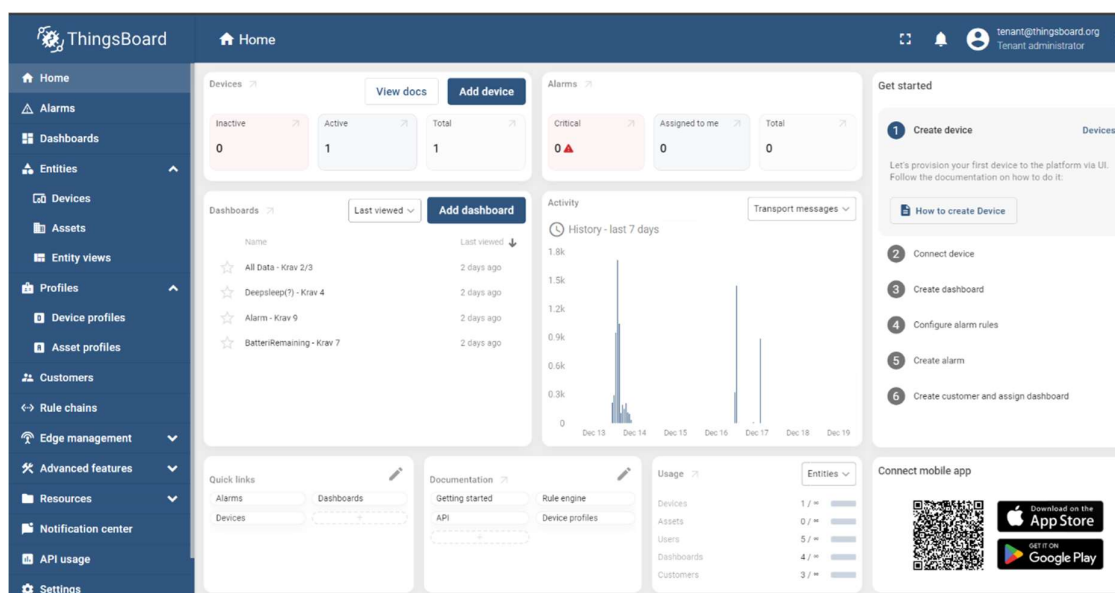
ID 14	Krav En selvvalgt andel af netværkets enheder tildeles dynamiske ip-adresser jævnfør ip-planen	Prioritet 1
Kategori Netværksopsætning	Accepttest Dette verificeres ved hjælp af f. eks CMD og status vindue i Windows Hvis nogle af gruppens Pc'er/ enheder tildeles dynamiske IP-adresser, når de tilsluttes gruppens LAN, vurderes kravet at være opfyldt.	Godkendt



Diskussion og evt. ønskede ændring af krav

ID 15	Krav På mini pc installeres Ubuntu, med Thingsboard, for dataopsamling via mqtt	Prioritet 1
Kategori VM opsætning	Accepttest 1. Historiske data kan ses i Thingsboard på mini pc 2. "Online" data kan ses i Thingsboard på mini pc	Godkendt

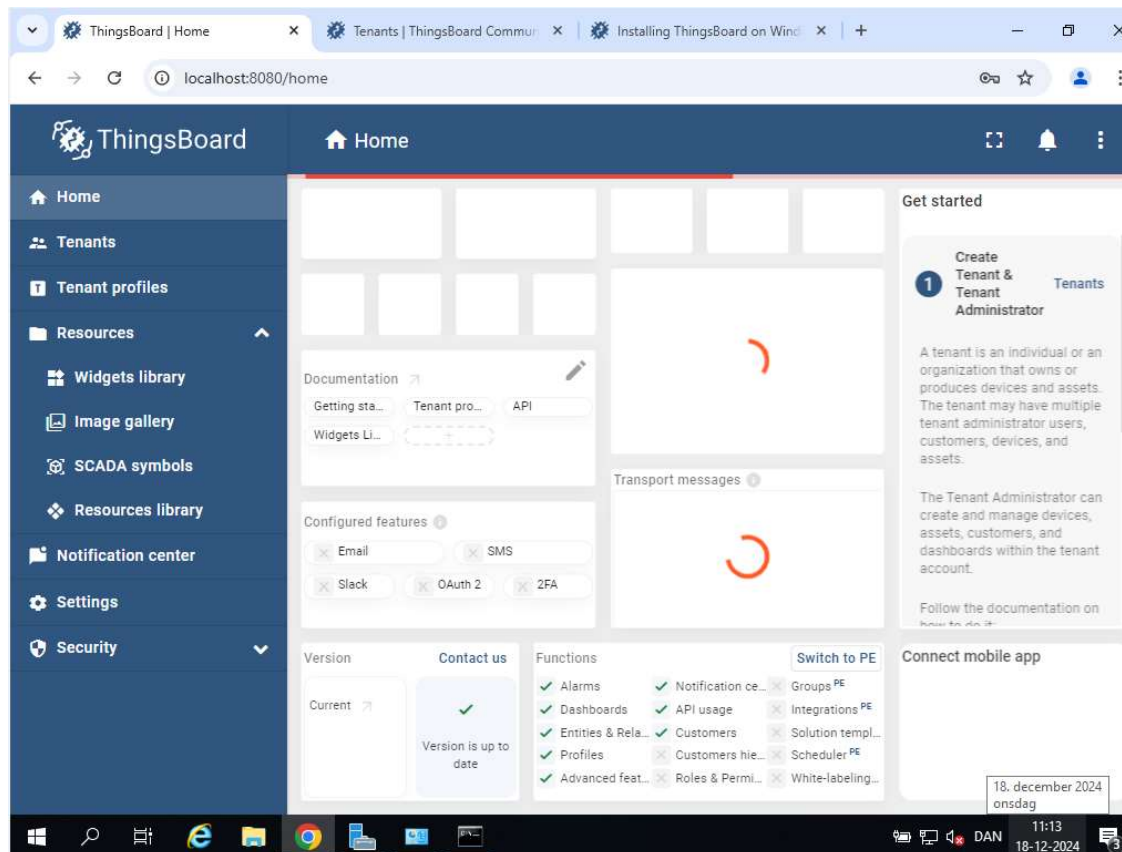
Ubuntu Thingsboard:



<https://www.youtube.com/watch?v=337adUrvdZU>

Diskussion og evt. ønskede ændring af krav

ID 16	Krav På mini pc installeres Thingsboard i Windows miljø for datadataopsamling via mqtt	Prioritet 3
Kategori VM opsætning	Accepttest 1. vi kan se historiske data i Thingsboard – på mini pc 2. Vi kan se "online" data i Thingsboard -på mini pc	Delvis Godkendt

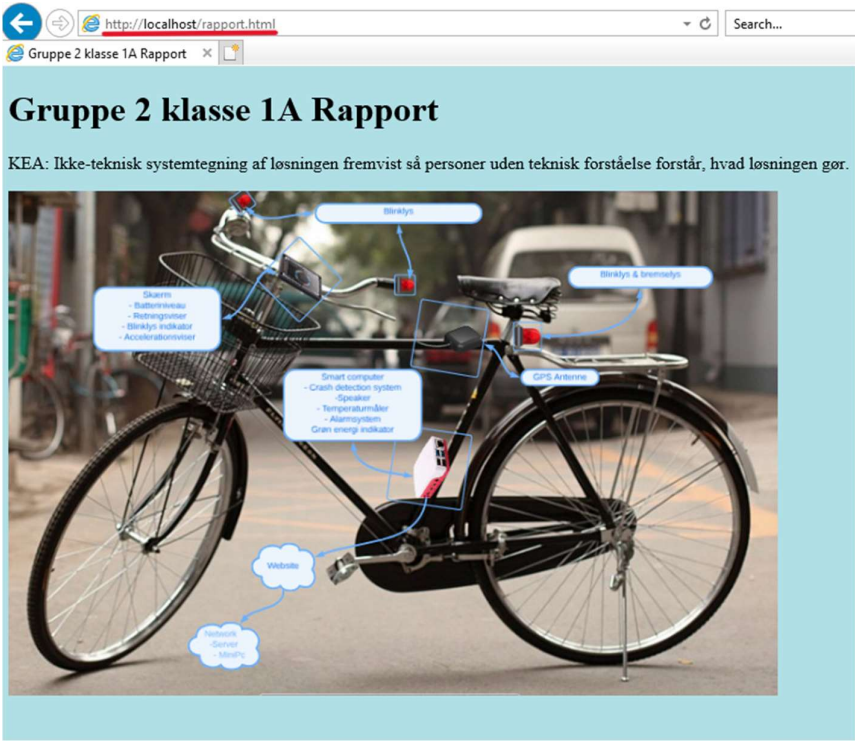


Under installationen har Thingsboard ikke fået oprettet de 2 standardbrugere som var nødvendige for at oprettet et dashboard og acces token. Da vi var i en periode under tidspres, havde vi ikke tid til at udbedre fejlen og få det op og køre 100%. SQL og Thingsboard blev installeret og oprettet på en Windows maskine.

Diskussion og evt. ønskede ændring af krav

ID 17	Krav På minipc'en installeres en web-server. Webserveren skal vise en hjemmeside der præsenterer løsningen.	Prioritet 2
Kategori Virtual opsætning	Accepttest Visuel præsentation af fungerende website, med system tegning og blokdiagram af løsningen	Godkendt

Billede



Diskussion og evt. ønskede ændring af krav

ID 18	Krav Når cyklens accelerometer registrerer et styrt som overskrider grænsen, skal data (GPS Lokation/Tid) efter 10 sekunder sendes til ThingsBoard (API) og til relevante personer (igennem callmebot WhatsApp API). Data skal også gemmes lokalt på mini pc.	Prioritet 2
Kategori Styrt registrering	Accepttest 1) Test program startes og styrt registrering er nu aktivt. 2) Styrt registreres gennem accelerometer grænsen. 3) Data sendes til ThingsBoard, data sendes også til en specifik person på WhatsApp (gennem callmebot WhatsApp API). Data vil blive gemt lokalt på mini pc i en "crash" log fil.	Godkendt
Link til video af testen https://www.youtube.com/watch?v=AqWqM2mA-s4		
Diskussion og evt. ønskede ændring af krav		

10. Implementering af løsningen i drift

Da vores løsning ikke endte med at fungere helt med alle dele sat sammen til rapportens deadline, men delene virker hver for sig, vil der i stedet for uddybes de vanskeligheder eller mangler der skal fixes for at enheden kan køre som forventet.

Som en af de første, skal alle de forskellige programmer sættes ind som libraries på ESP'en, hvilket derefter vil importeres i en main.py fil til at kører alle de forskellige funktioner i et hovedloop, der refererer til de forskellige funktioner som enhederne bruger til at køre.

Et andet problem vi stødte ind i, hvor vi prøvede at fikse undervejs, var at kablerne fra vores main board (Educaboardet) til vores Neopixels på Neopixel ringen var for lange til at give et stabilt signal på data-pin'en. Dette resulterede i forkerte farver, forkerte pixels og at den ikke reagerede overhovedet.

Vi prøvede os frem med forskellige løsninger, som endte med ikke at virke. Det var, blandt andet, både at sætte en modstand mellem data-pin og ground i neopixel enden, samt prøve at bruge et ethernet cat6 kabel, noget som andre havde succes med, både i klassen og på nettet. Førstnævnte virkede til dels, men begyndte at lave ikke forventede fejl, selv efter at have virket til at starte med og med tidligere afprøvninger af hvilke komponenter skulle bruges. Netværkskablet virkede ikke og det kan være enten dårlig forbindelse i selve kablet, eller dårlig påsætningen af connectors, men tiden var der ikke til at sidde og nørkle længere tid med det, men er blevet sat på to-do listen til den mundtlige eksamen.

Et problem vi også kom ud for, som ikke var helt forudset, var at pga. manglen på GPIO pins på vores ESP32 på Educaboardet, da mange allerede var reserveret til forskellige komponenter, gjorde at vi havde mærkelige former for interferens mellem enheder. Det gjorde bl.a. at vores Crash Detection, vores knapper på styret og vores skærm ikke kunne arbejde sammen. Det resulterede i at ingen af dem virkede, som de ellers skulle.

11. Praktisk projektplanlægning og -ledelse

I projektet har vi anvendt flere forskellige projektledelsesværktøjer til at strukturere arbejdsopgaver og planlægge vores tid. Vi startede med at lave et WBS (Work Break-Down Structure) med alle de opgaver vi kunne komme i tanke om. På den måde fik vi brudt opgaven ned i mindre opgaver hvilket også gjorde det lettere at få et overblik over hvor mange opgaver, der i alt skulle løses og hvilke opgaver der var sammenhæng imellem.

WBS har givet os et visuelt overblik, over hvilken aktivitet hver enkelt opgave hører til under og hvor der er flest opgaver tilknyttet.

Vi startede med ca. fem overordnede aktivitetsgrupper, men måtte hurtigt udvide med flere kategorier af aktiviteter, for at få opgaverne brudt op i mindre opgaver for at tydeliggøre hvor mange opgaver vi havde at løse.

Herefter tog vi hver opgave fra vores WBS og flyttede den over til vores Kanban tavle. For hver opgave blev der oprettet et Kanbankort på en Post-It med information om opgaven (se nærmere beskrivelse under punkt 11.3). Alle opgaver startede i To Do og bevægede sig løbende gennem kolonnerne, for til sidst at ende i Done.

På vores Kanban tavle har vi også haft en kolonne med overskriften vigtig, den brugte vi især i starten for at holde styr på deadlines for afleveringer. Vi oprettede farvekoder for kategorier, uddelte farvekoder til alle gruppemedlemmer og optegnede hvad alle forkortelser på vores Kanbankort står for. Derved var der fuldt overblik over både arbejdsopgaver, hvem der var i gang med hvilke opgaver og skulle man miste overblikket var alle informationer på samme sted.

Vi har dagligt afholdt Kanban møde og fået styr på om nye opgaver var dukket op, status på igangværende opgaver og om alt i øvrigt gik efter planen.

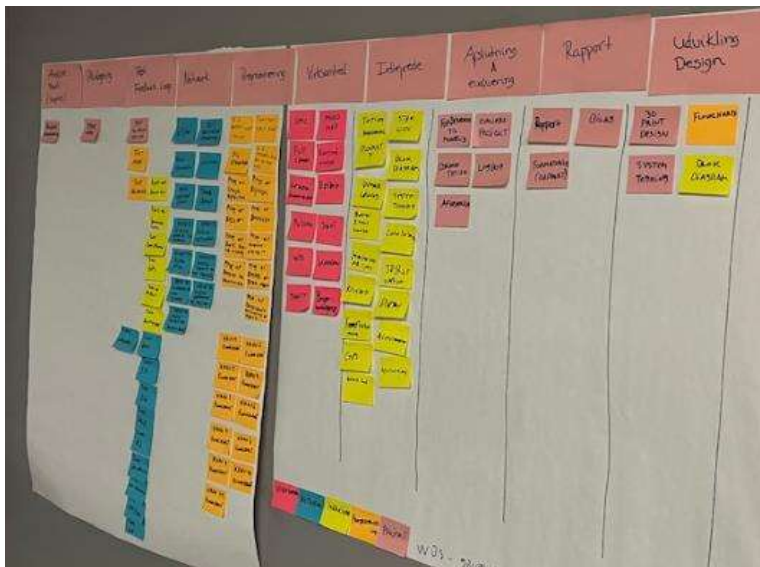
En af de første dage fik vi oprettet vores Gantt kort, men da det på dette tidspunkt ikke var helt, havde overblikket over omfanget af opgaver og kun et meget løseligt tidsestimat for hver opgave, blev det til et meget enkelt Gantt kort, som vi senere i projektet har gjort mere detaljeret.

Vi vil have mere fokus på dette i kommende projekter. Det endelige Gantt kort fungerer godt da det er opdelt så det giver et tidsoverblik helt ned på dagsplan.

Vores Logbog har også fungeret godt, da den har givet overblik over hvad der er blevet nået på dagen, men især også fordi det har givet anledning til snakke omkring, hvad der skulle være fokus på den efterfølgende dag.

Vi har ikke haft de store udfordringer omkring den praktiske projektplanlægning og -ledelse, men de første dage prøvede vi os lidt frem indtil vi fandt en løsning, der gav os struktur og overblik. Vi har alle været glade for vores projektledelsesværktøjer og vi synes de har virket i praksis.

11.1. WBS

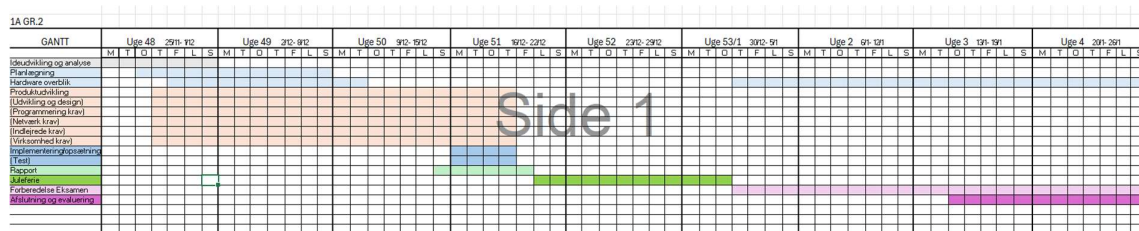


Figur 64: Endelig version af WBS tidligere versioner af WBS er i bilag

En WBS eller Work Breakdown Structure er et nøgle element i et godt projekt. Det giver et projekt en måde at holde opgaver overskueligt.

Formålet er at bryde projektet ned i mindre opgaver, så det er nemmere at planlægge og kontrollere. Det skaber en visuel struktur som gør det nemt at have en klar struktur omkring projektets omfang som helhed.

11.2. Gantt



Figur 65: Færdig Gantt, første udgave er i bilag

Et Gantt Diagram er et tidsplanlægningsværktøj til projekter. Det giver muligheden for at have noget visuelt, som kan vise let og overskueligt varigheden på forskellige dele af projektet (i tid). Generelt er et par af nøgleelementerne at vise hver opgave for sig, hvor den har en start/slutdato, hvilket er tydelige.

Opgaverne forbindes til tider (og oftest ender de hvor en anden starter), men nogle gange er det nødvendigt at overlape dele af et projekt. Gantt kan også indeholde milepæle og forskellige deadlines (møder, etc.).

I sidste ende er det endnu et vigtigt værktøj for et projekt som er med til at give overblikket og muligvis identificere flaskehalse på et projekt (eller mulige forsinkelser).

11.3. Kanban

(Husk at indsætte endelig version)



Figur 66: Færdig Kanban tavle, første udgave er i bilag

Et Kanban Board er endnu et vigtigt værktøj i styring af arbejdsopgaver på et projekt. Det er et agilt (fleksibel, tilpasningsdygtigt) værktøj. Nøgledelen af et Kanban Board er at vise de forskellige stadier på et projekt, sammen med diverse opgaver som er brudt ned fra vores WBS.

Som det ses på vores billede specifikt, så har man: To Do, Doing, Waiting, Test, Done.

Endnu et positivt element er at hver opgave har dens eget kort, som kobles til en person hvor start dato, forventet slutdato, endelig slutdato, f.eks.

Kanban Boardet fokuserer på et flow og begrænser mængden af opgaver som kan være i Doing, da ingen (i vores projekt) må være mere end 2 på en opgave. I hvert fald, så er brugen af Kanban Boardet let og fleksibel (agilt!) til at styre opgaver. Optisk set øger det overskueligheden og kommunikationen i et team og er en effektiv løsning til løbende forbedring på et projekt. I denne omgang brugte vi Kanban fysisk, men Trello er en ekstrem god alternativ løsning i fremtiden.

11.4. Projektanalyse

SWOT-Analyse, 1A Gruppe2 - IoT1

Interne	
Strengths (styrker) Gode ideér/ løsninger Teknisk kendskab Bredt netværk Konkurrencedygtige løsninger	Weaknesses (svagheder) Sygdom Manglende viden Tid/ planlægning Problemer i udviklingsfasen Begrænset arbejdskraft i teamet
Eksterne	
Opportunity (muligheder) Skalérbar løsning/ salg Udvide kundebase	Threats (trusler) Konkurrenter Leveringstid/ komponent mangel Lovgivning (trafikstyrelsen, GDPR)

Figur 67: SWOT-analyse

En SWOT-Analyse er et redskab som sætter fokus på de interne/eksterne faktorer både de positive og de negative. Alle 4 aspekter har indflydelse på et projekt (eller en virksomhed).

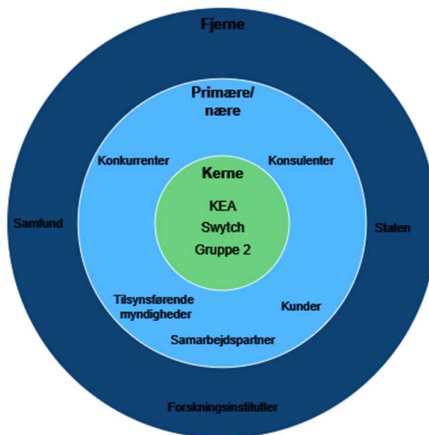
Interne, Styrker: Dette er de ting som virksomheden er gode til eller de ting som gennemslagskraft, dette kunne være et godt kendskab til et specifikt marked, viden på et kerneområde, et godt brand eller bare loyale kunder, ny teknologi og god markedsføring.

Interne, Svagheder: Dette kan være forskellige kompetencer som vores virksomhed ikke har, eller muligvis ting, som vores virksomhed ikke gør godt nok i forhold til andre (konkurrenter). Andre aspekter kan også være faktorer, f.eks. Hurtig personaleudskift, mangel på viden, dårlig brand, dårlig teknologi.

Eksterne, Muligheder: Dette er mulighederne på et specifikt marked, hvor de kan enten være mange eller få. Det kan f.eks. "Vi" som virksomhed har fået nye kunder eller der er nye markedstrends som "vi" har mulighed for at udnytte, konkurrenter gør det dårligt økonomisk, f.eks..

Eksterne, Trusler: Denne faktor er de ting som truer os, som virksomhed og de forretningsmuligheder der er. Som eksempel kunne det være at vores konkurrenter gør det bedre end os. Vores kunder lukker deres firma (eller) de finder en bedre løsning hos vores konkurrenter, lav vækst i marked som helhed, ny teknologi som er med til at gøre vores forældet.

Interessent-Analyse (Cirkel), 1A Gruppe2 - IoT1



Figur 68: Interessent-Analyse

En interessent i denne sammenhæng er en person (eller) virksomhed som har betydning for en virksomheds drift (eller) påvirker et projekt. Ved at lave en interessentanalyse gør vi det muligt at få det fulde overblik over projekts interessenter og hvordan de skal håndteres og prioriteres. Derudover kan man identificere mulige risici eller barrierer fra fjerne interessenter og udvikle en plan for at informere og engagere relevante interessenter gennem hele projekts forløb.

Vi har identificeret interessenter ved en brainstorm.

For at skabe et overblik over projektets interessenter, er der her udarbejdet en interessentcirkel, som kategoriserer interessenter i forhold til deres nærhed til projektet.

Cirkelens formål er at identificere hvor tæt de forskellige interessenter er på projektet og hvor stor indflydelse de har på deres forløb.

Ovenstående interessentcirkel opdeler interessenter i følgende tre niveauer:

Kerneinteressenter:

- Disse interessenter er direkte involveret i projektet. De omfatter **KEA, Swytch og Gruppe 2**. De har høj indflydelse og interesse i projektets succes.

Primære/nære interessenter:

- Disse interessenter er tæt på projektet, men er ikke direkte involveret i udførelsen. Eksempler er **konkurrenter, konsulenter, kunder, samarbejdspartnere og tilsynsførende myndigheder**.

Fjerne interessenter:

- Disse interessenter har mindre direkte indflydelse, men kan påvirke projektet på et bredere niveau. Dette inkluderer **samfundet, staten og forskningsinstitutter**.

Risiko-Analyse, 1A Gruppe2 - IoT1

Hvad kan der ske?	Sandsynlighed 1-5 (S)	Konsekvens 1-5 (K)	Risikopoint (S) x (K)	Forebyggende eller afhjælpende foranstaltning	Ansvarlig
Sygdom (kortvarig)	4	1	4		
Sygdom (langvarig)	1	5	5		
Konflikter	2	3	6	God dialog, faste rammer, aftaler	Alle
Manglende engagement	1	2	2	Forventningsafstemning, varierende opgaver	Alle
Budget	2	3	6	Finde konkurrerende leverandører, budget opfølgning	Alle
Dårlig planlægning	1	5	5	Brug planlægningsværktøjer	Alle
Hardware fejl eksternt	1	4	4		
Lovgivning	1	2	2	Research glædende lovgivning	Alle
Design fejl	4	2	8	Test	Alle
Tekniske fejl internt	2	4	8	Test, være omhyggelige	Alle

Tabel 2: Risikoanalyse tabel

En risikoanalyse identificerer potentielle risici, der kan påvirke projektets fremdrift, samt vurderer sandsynligheden og konsekvensen af disse risici.

Formålet med at lave en risikoanalyse er at kunne implementere forebyggende tiltag og planlægge foranstaltninger til at håndtere risici hvis de opstår. Dette hjælper med at minimere uforudsete udfordringer og sikre at projektet gennemføres gnidningsløst.

I risikoanalysen vurderes hver risiko ud fra følgende;

- **Sandsynlighed**
- **Konsekvens**
- **Risikopoint**, som er resultatet af at gange sandsynlighed med konsekvens.

Dertil er der skrevet forbyggende eller afhjælpende initiativer til hvor det er muligt.

Interessent-Analyse (Firkant), 1A Gruppe2 - IoT1



Figur 69: Interessent analyse

For at skabe et klart overblik over projektets interessenter er de kategoriseret i fire forskellige grupper.

Denne kategorisering hjælper med at forstå hvordan de respektive interessenter skal håndteres og kommunikeres til.

De fire kategorier er følgende:

- **Gidsel (informeres)**

Disse interessenter påvirkes af projektet, men har begrænset indflydelse på projektets forløb. De skal løbende informeres for at sikre forståelse og accept.

Eksempler på gidsler kan være følgende; **Kunder, samarbejdspartnere, tilsynsførende myndigheder**

- **Ressourceperson (involveres)**

Disse interessenter er centrale for projektets succes og skal involveres aktivt i beslutningsprocesser og opgaveløsning

Eksempler på ressourcepersoner kan være følgende; **KEA, SWYTCH, Gruppe 2**

- **Ekstern interessent (orienteres)**

Disse interessenter er perifere, men skal holdes orienteret om projektets udvikling for at sikre opbakning.

Eksempler på eksterne interessenter kan være følgende; **Staten, samfund, forskningsinstitutter**

- **Grå eminence (høres)**

Disse interessenter har indflydelse på projektet, men arbejder ofte i baggrunden. Deres input skal høres og tages i betragtning, da de kan påvirke projekts udfald indirekte.

Eksempler på grå eminencer kan være følgende; **Konsulenter, konkurrenter**

Ved at opdele interessenterne på denne måde hjælper det med at tilrettelægge kommunikation og involvering på en struktureret måde.

12. Konklusion

Igennem de sidste 4 uger har vi taget udgangspunkt i vores problemformulering og "samarbejde" med Swytch, om at skabe en løsning, der øger sikkerheden for cyklister i trafikken. I scopet af projektet har vi fået udarbejdet og produceret en blinklysløsning og et crash detection system.

Igennem projektet har vi udarbejdet en løsning igennem brug af flere forskellige teknologier, Thingsboard, Ubuntu, Windows Server og flere indlejrede dele som et accelerometer/temperatur måler, et relæ, en booster converter, batterier, lcd skærm.

Vores arbejde med alle disse aspekter har gjort, vi har store forventninger og forhåbninger til vores løsning og at den vil være med til at forbedre sikkerheden for alle i trafikken fremadrettet.

Vores blinklys system skaber et blikfang for bilister og fodgængere, når en cyklist vil tage et højresving, som er en af de førende årsager til uheld hvor cyklister er involveret.

Endnu et kig tilbage mod problemformuleringen og vores crash detection system, som har til opgave at skabe tryghed for cyklisten i form af opmærksomhed og at informere nærmeste pårørende, så cyklisten ikke føler sig alene eller overset, hvis de skulle komme til skade, skulle gerne være med til at fjerne problematikker med hit'n'run ulykker for cyklisten, hvis de ikke selv er i stand til at indikere at de er i fare og/eller kommet til skade.

Vi mener hos CrashLiteX at dette projekt er med til at øge sikkerheden i den daglige trafik og vil skabe en ny samtale om sikkerhed generelt og i sidste ende være med til generelt at udvikle nye sikkerhedsstandarder og reformer for trafikken.

13. Projektforløbet

Tidligere på året da vores gruppe blev oprettet, lavede vi en gruppekontrakt, hvor vi fik en god snak om hvordan vi ville arbejde som gruppe. Vi fik hurtigt sat nogle klare rammer for samarbejdet og det har vi holdt os til under hele projektet. Alle har været engagerede og lagt energi i projektet og selvom vi hver især har haft egne opgaver, har vi været gode til også at være opmærksomme på andre gruppemedlemmers arbejde og svare på spørgsmål fra resten af gruppen. Vi har haft det sjovt og hyggeligt selvom vi har haft både lange og travle dage.

Vores kommunikation er foregået primært over teams, mail og sms. Selvom vi har syntes at projektet var omfattende og at vi nogle gange følte os udfordret på vores kompetencer, har vi holdt humøret oppe og været gode til at rose hinanden for de processer der var i gang og de opgaver der blev løst.

Vi har talt om hvad der kan laves bedre eller gøres anderledes til næste gang og her er vi talt om at det vil være fint med mere klare rammer for hvad der forventes af gruppen i forhold til hvornår vi skal arbejde og hvor længe vi skal arbejde. Vi synes selv at vi har ydet en god indsats alle sammen for at få lavet et godt projekt, men det ville være fint hvis det var tydeligt beskrevet hvornår man har helt fri fra projektet, derved kan alle frit disponere over tiden væk fra projektet uden dårlig samvittighed og selv vælge om fritiden skal bruges på fritidsarbejde eller til afslapning.

14. Perspektivering

Vores løsning, selv mødt med en håndfuld problematikker for at danne en færdig prototype, mener vi stadig selv, har en god salgsmulighed. Dog skal der jo rettes på flere ting. Vores løsning skulle måske også gøres mindre og designes mere fra bunden op i stedet for igennem Educaboardet.

Den nemmeste måde at få ideen videre ud i verdenen på, er ved at enten sælge ideen videre til mere kompetente udviklere, eller måske tage patent på systemet og sælge ud af licenser til fabrikker.

Eftersom det er første projekt på uddannelsen, med begrænset tid, ville det være oplagt at få undersøgt mere omkring det emne vi har endt med. Vi føler selv vi fik et okay overblik ud fra de rammer vi blev sat i, men bedre brugerundersøgelse og test ville kunne gå lang vej til et bedre produkt.

Ideen omkring at gøre trafikken mere sikker for cyklister, både generelt men specielt folk med motoriske vanskeligheder, er stadig også en vi værdsætter højt og tror der er et stort nok marked til at udvide sig ud i. Hövding og vores udvalgte virksomhed, Swytch, er to navne man kan pege på for at vise at teknologiske udviklerne indenfor cykelkulturen er alive and well. Endda måske noget der kunne veje over i nye sikkerhedsmuligheder.

15. Litteraturliste

[4] "About us Meet the team and find out how it all began..."

<https://www.swytchbike.com/about-us/>

(2024-11-25)

[12] "Analog til digitale konvertere: hjertet af moderne elektronik"

<https://www.ic-components.dk/blog/analog-to-digital-converters-the-heart-of-modern-electronics.jsp>

(2024-12-09)

[3] + [5] + [8] Andersen, Ulrik "Antallet af trafikdræbte og tilskadekomne i København stiger med 30 procent"

<https://ing.dk/artikel/antallet-af-trafikdraebte-og-tilskadekomne-i-koebenhavn-stiger-med-30-procent>

(2024-11-25)

[13] Campbell, Scott "Basics of the I2C Communication Protocol"

<https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>

(2024-10-12)

[1] - [2] + [7] "Fald i dræbte og tilskadekomne i trafikken"

<https://www.dst.dk/da/Statistik/nyheder-analyser-publ/nyt/NytHtml?cid=49296>

(2024-11-25)

[16] "Installing Thingsboard CE on Ubuntu Server"

<https://thingsboard.io/docs/user-guide/install/ubuntu/>

(2024-12-13)

[10] "IT Explained: MQTT"

<https://www.paessler.com/it-explained/mqtt>

(2024-12-10)

[15] "Micropython – Time related functions"

<https://docs.micropython.org/en/latest/library/time.html>

(2024-12-12)

[14] "The Evolution of Wi-Fi Technology and Standards"

<https://standards.ieee.org/beyond-standards/the-evolution-of-wi-fi-technology-and-standards/>

(2024-12-04)

[11] "Understanding UART"

https://www.rohde-schwarz.com/cz/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart_254524.html#:~:text=UART%20stands%20for%20universal%20asynchronous,and%20receive%20in%20both%20directions

(2024-12-10)

[6] "Ung kvinde er død efter påkørsel af formodet spritbilist"

https://politiken.dk/danmark/art10198565/Ung-kvinde-er-d%C3%B8d-efter-p%C3%A5k%C3%B8rsel-af-formodet-spritbilist?srsId=AfmBOor06qQ8V4WksqqvD_1y8WY_zxnpEo64S_T-w9zBnmhrbs_2hBWD

(2024-11-25)

[9] Fruhlinger, Josh "What is Wi-Fi and why is it so important?"

<https://www.networkworld.com/article/968819/what-is-wi-fi-and-why-is-it-so-important.html>

(2024-12-10)

[17] - [23] "Windows Server – Service overview and network port requirements"

<https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/service-overview-and-network-port-requirements>

(Sidst tjekket 2024-12-19)