NTNU
Norwegian University of
Science and Technology

MySmartHome

Forfattere
Jakob Fonstad
Kristian Sundhaugen
Martin Pukstad
Stian Fenstad

Bachelor i programvareutvikling
20 ECTS
Institute for Datateknikk og Informatikk
Norges teknisk-naturvitenskapelige universitet,

16.05.18

Veileder Frode Haug

Sammendrag av Bacheloroppgaven

Tittel:	**MySmartHome**
Dato:	16.05.18
Deltakere:	Jakob Fonstad Kristian Sundhaugen Martin Pukstad Stian Fenstad
Veiledere:	Frode Haug
Oppdragsgiver:	Fosen Utvikling
Kontaktperson:	Jonas Kirkemyr
Nøkkelord:	Programmering, Smarthjem, NodeJS, Arduino
Antall sider:	68
Antall vedlegg:	16
Tilgjengelighet:	Åpen

Sammendrag:

I verdenen vi lever i idag blir stadig vekk flere teknologier og problemområder digitalisert. Dette gjelder også i hjemmene våre. Fosen Utvikling ga oss derfor oppgaven om å utvikle en smarthjem løsning fra bunnen av. Ved å sette oss inn i en smarthjem løsning fra hardware nivået og opp til en programvare løsning, har vi i denne rapporten kunne gå igjennom et vidt spekter av temaer og problemområder som kommer opp i en slik løsning. Med dette prosjektet har vi levert grunnmuren i det som vil være et større prosjekt. Vi har fått på plass et bibliotek som skal kunne fungere på de fleste sensorer, og levert et system som kan kommunisere og sende data videre til en webløsning.

Summary of Graduate Project

Title:	**MySmartHome**
Date:	16.05.18
Authors:	Jakob Fonstad Kristian Sundhaugen Martin Pukstad Stian Fenstad
Supervisor:	Frode Haug
Employer:	Fosen Utvikling
Contact Person:	Jonas Kirkemyr
Keywords:	Programing, Smarthome, NodeJS, Arduino
Pages:	68
Attachments:	16
Availability:	Open

Abstract: The world we live in is constantly being modernized and digitalized, this process has also made it's way into our homes. Fosen Utvikling wanted us to develop a smarthome solution from scratch. By developing a smarthome solution from a hardware level up to a software solution, we have encountered a wide spectrum of topics and problems that occurs when developing a smarthome solution. With this project we have delivered the foundation of what may become a bigger project. We have made a library that can be easily setup and installed on most sensor setups, as well a system that allows for communication between the devices and sending data to a web solution.

Forord

Vil gi en takk til Fosen Utvikling, for å lage denne oppgaven for oss. Fosen Utvikling befinner seg i Indre Fosen kommune på Fosen halvøya like utenfor Trondheim, og er en bedrift som tar på seg konsulentoppdrag for alle som sender inn forespørsler. De ansatte ved Fosen Utvikling har tung kompetanse innen forskjellige fagfelt, som informasjonssikkerhet, utvikling og drift. De er hovedleverandør av utviklingstjenster for Dogi AS som er et søsterselskap av Rissa Kraftlag, men tar også på seg andre oppdrag.

Vil også takke vår kontaktperson hos Fosen Utvikling, Jonas Kirkemyr og vår veileder Frode Haug, for godt samarbeid og rådgivning underveis i prosjektet.

Innhold

Figurer

Tabeller

Listings

Ordliste

aktuator En aktuator er en teknisk innretning som ved hjelp av styresignaler utfører en mekanisk bevegelse.Aktuatorer er mye brukt i all slags mekatronsike systemer og automatisering.[1]. 15, 31

Arduino Arduino er en plattform for prototyping av elektronikk basert på program- og maskinvare med åpen kildekode[2]. vii, 1, 2, 4, 7, 14–17, 28, 41, 45, 50, 51, 60–62

Arduino Uno Arduino Uno er et microkontroller brett laget av Arduino CC[3]. 28, 60

baud Baud angir et antall informasjonsbærende, adskilte informasjonsenheter(symboler) i en informasjonsstrøm. Baud brukes for å angi informasojnshastighet ved signaloverføring.[4]. 60

DHT-22 DHT-22 er en digital-output fuktighet og temperatur sensor. 29

Domain Name System DNS er et hierarkisk desentralisert navngivings system for datamaskiner, tjenester og andre resursser tilkoblet internett. [5]. xiii

DoxyGen Doxygen er en dokumentasjonsgenerator, et verktøy for å skrive dokumentasjon og referanser til programvar[6]. 62

Embedded Javascript Templating EJS er et klient-side templating språk, som originalt var del av JavaScript MVC[7]. xiii, 41

Express.js Express er et JavaScript rammeverk basert på Node.Js. Express brukes til utvikling av serverside-programvare, slik som webapplikasjoner og HTTP-APIer[8]. 25, 28, 41, 48, 63

If This Then That IFTTT er en gratis web basert tjeneste som lager enkle betingede uttalser kalt applets[9]. xiii, 14, 32

Internet of Things Tingenes internett, også kjent under det engelske begrepet Internet of Things (IoT), er nettverket av identifiserbare gjenstander som er utstyrt med elektronikk, programvare, sensorer, aktuatorer og nettverk som gjør gjenstandene i stand til å koble seg til hverandre og utveksle data[10]. xiii, 25, 27

Jira Er et issue tracking product utviklet av Atlassian. Gjerne brukt i sammenheng med Scrum utvikling[11]. 5, 10, 12, 24, 64

Message Queuing Telemetry Transport MQTT er en ISO standard publish-subscribe basert meldings protokollen, som fungerer på toppen av TCP/IP protokkolen[12]. xiii, 25

Node.Js Node.js er et åpent kryssplattform runtime-system for server- og nettverksapplikasjoner[13]. ix, x, 7, 14, 25, 28, 41, 48, 49, 63

NodeMCU NodeMCU(ESP8266) er et microkontroller brett med innebygd Wifi Modul[14]. 28, 60

Open Source Åpen kildekode (eng: Open source) betyr at kildekoden til et dataprogram er gjort tilgjengelig (ofte på Internett) for alle[15]. 1, 3, 5, 15–18, 24, 30, 53, 62–64

Planning Poker Planning poker også kalt Scrum Poker er en konsesus basert teknikk for å estimering.[16]. 12, 61

Python Python er et programmeringsspråk laget av Guido Van Rossum.[17]. 7, 41, 51, 62

Raspberry Pi Raspberry Pi er en serie med små Single Board Computers utviklet av Raspberry Pi Foundation[18]. xi, 2–4, 7, 14–16, 31, 50, 53, 62, 63

Raspbian Raspbian er et Debian basert operativ system for Raspberry Pi[19]. 41, 53, 62

relé Et relé er en elektrisk komponent som fungerer som en strømbryter. Det kan brukes til å kontrollere en elektrisk strøm, gjerne mye større enn den som blir tilført relé-spolen, eller – sammen med flere andre reléer – til å lage en logisk styring[20]. 2, 3, 14, 16–18, 27, 28, 31, 51, 61, 65

schematic Et Schematic er en representasjon hvor man bruker symboler for å representere et system[21]. 1, 2, 7, 14, 16, 31, 51

SCRUM Scrum er et iterativt og inkrementelt rammeverk for å utivkle komplekse informasjonssystemer [22]. 5, 6, 9, 10, 12, 13, 24, 50, 61, 62

sensor En sensor er et instrument som registrerer en viss påvirkning, f.eks. varme eller kulde, og konverterer registreringen til et signal som kan leses av en observatør eller et annet instrument[23]. x, 1–3, 7, 14, 16–18, 24, 27, 28, 30–32, 34–36, 42–44, 51, 53–61, 63–65

Service Set Identifer et Service set er et set med alle enheter assosiert med et spesifikt WiFi nettverk[24]. xiii, 43

Single Board Computer En Single-Board computer er en fullstendig datamaskin på et enkelt krekskort, med en microprossessor, minne og I/O[25]. xi

smarthus Smarthus eller hjemautomasjon (eng: Home automation) er helheten av overvåkning, styring, regulering og optimeringsinnredninger i en bygning[26]. 1, 2, 16, 17

Socket.IO Socket.IO er et JavaScript bibliotek for sanntid toveis kommunikasjon mellom web klienter og servere[27]. 26, 28, 32, 41, 48, 49, 59, 63

SonarQube SonarQube (tidligere kjent som Sonar) er et statisk kodeanalyse verktøy utviklet av SonarSource.[28]. 7, 50, 51

Sublime Text Er en kryss platform kildekode editor.[29]. 41

Trello Trello er et web-basert prosjekt styrings applikasjon[30]. 5, 13, 64

VNC Viewer VNC Viewer er programvare laget av RealVNC som bruker Virtual Network Computing (VNC) protokollen for å fjern kontrollere en datamaskin. [31]. 53

WiFi Manager WiFi er et rammeverk som brukes sammen med Aurduino brettet ESP8266 for å koble et Arduino oppsett med et valgt WiFi nettverk.[32]. 56, 57

workshop Workshop er en kort og intensiv kurs eller utdanningsprogram rettet mot en smalt fagfelt [33]. vi, 51, 60

Akronymer

DNS Domain Name System. 43

EJS Embedded Javascript Templating. 41

IDE Integrated Development Enviroment. 7, 41, 50, 51, 60, 62

IFTTT If This Then That. vii, 14, 32, 33, 36, 39, 40, 65

IOT Internet of Things. 25

MQTT Message Queuing Telemetry Transport. 14, 17, 25, 27, 28, 43, 46, 53, 59, 63

SSID Service Set Identifer. 43, 63

1 Innledning

1.1 Problemområdet

Samfunnet digitaliseres mer og mer for hvert år som går. Vi deler mer informasjon over lengre avstander raskere enn noen gang tidligere i menneskelig historie, og vi automatiserer og kontroller stadig mer med hjelp av digitale verktøy. Bedrifter har lenge vært tidlig ute på slike automatiserte systemer, da det potensielt kan spare dem for store utgifter. Google har for eksempel tatt sensorer, automatiserte systemer og kunstig intelligens i bruk for å kutte strømutgifter betraktelig i sine serverparker[34]. Liknende automatiserte systemer har også, i de siste årene, kommet inn i hjemmene til privatpersoner. Dette kan for eksempel være i form av automatisert belysning etter hvilket rom man befinner seg i. Eller styrte systemer slik at man kan starte oppvarmingen av hytta via en app på telefonen, så det er varmt og godt når man ankommer. Nesten alt man kan tenke seg av elektroniske applikasjoner kan automatiseres eller fjernstyres i dag.

Hjemmeautomatisering er ikke et nytt tema i samfunnet. Automatisering av manuelle oppgaver er noe mennesket har jobbet med lenge. Den første versjonen av elektrisk hjemmeautomasjon kom gjennom applikasjoner som kunne spare oss for manuelt arbeid. For eksempel i form av vannkokere(1889), symaskiner(1889)[35], vaskemaskiner(1904), kjøleskap(1916), oppvaskmaskiner (1929)[36] og tørketromler(1938)[37]. Det var ikke før i 1975 at automatisering over nettverk kom inn i hjemmene til folk.

1.2 Avgrensing

Prosjektet har som mål å bli et Open Source prosjekt. Open Source vil Fosen Utvikling selv stå for å sette opp etter endt Bacheloroppgave. Eventuelle schematic for oppsett av de individuelle Arduino sensorene må være oversiktlige og tilgjengelige ved ferdgistilling av produktet.

Prosjektet behøver ikke å avgrenses kun til hjemmet, større fasiliteter som arbeidsplasser vil være en reell plattform der det er ønskelig å kunne anvende løsningen. Styringspunktet skal nå ut til alle sensorene. Dette løser vi ved å ta i bruk WiFi repeatere.

Oppgaven er satt opp slik at at vi selv skal eksperimentere med forskjellige sensorer, som kan tas i bruk i en smarthus løsning. Dette avgrenses etter hva som er mulig å koble til et Arduino brett, og hva som er funksjonelt i forhold til å bli tatt i bruk i et smarthus. Løsningen skal utvikles på en slik måte at det er forholdsvis enkelt å implementere en ny sensor inn i applikasjonen, og at data sendes på en slik måte at videreutvikling kan gå i andre retninger enn hva det er gjort i denne oppgaven.

1.2.1 Hva menes med Smarthus?

Smarthus er et bygg hvor ting som lys, oppvarming, ventilasjon eller sikkerhetsmekanismer automatisk blir kontrollert av et elektronisk system[26]. Slike systemer blir som

oftest overvåket og styrt over internett, da gjerne med bruk av WiFi på lokalt nivå. Moderne smarthus system består av sensorer og reléer som tar inn informasjon om omgivelsene sine og som så styrer elektriske applikasjoner. Disse er da koblet til et sentralt styringspunkt som brukes for å holde orden på all data og når reléer skal aktiveres eller deaktiveres. Et eksempel på dette er at en kan ha en sensor som måler temperatur, og når temperaturen faller under en viss grad, skrus en varmeovn på for å øke temperaturen[38].

1.2.2 Standardisering

Det er få globalt aksepterte standarder for smarthus løsninger[39], siden de fleste leverandører ønsker å utvikle egne løsninger for sine produkter og deler derfor ikke nødvendig informasjon om produktene sine så andre kan lage løsninger for dem[40]. Det finnes i dag noen kommersielle smarthus løsninger, men ofte gjelder disse kun elektriske applikasjoner fra en gitt produsent, og det er ingen global standard. Det begynner derimot sakte men sikkert å utvikles globale standarder[40].

1.3 Oppgavedefinisjon

Målet for oppgaven er å gi brukeren kontroll over sitt eget hjem ved bruk av sensorer og reeler som sender data til og kan styres fra en nettside. Oppgaven er delt opp i tre hoveddeler, de er; sette opp sensorer og reléer som kan sende og motta data eller styring, en nettside for visning av data fra sensorer og kontrollere reléer, og kommunikasjonen mellom disse. I utgangspunktet var oppgaven fra Fosen Utvikling å sette opp Arduino brettene, og ordne kommunikasjon mellom de og et sentralt styringspunkt, i vårt tilfelle en Raspberry Pi. Da det målet ble nådd relativt tidlig i utviklingsprosessen utvidet vi, i samarbeid med Fosen Utvikling, oppgaven til å også ha med en nettside, for styring av applikasjonen. Det innebærer at i tillegg til kommunikasjonen mellom sensorene og Raspberry Pien, må det også settes opp et server client forhold til en nettside.

1.3.1 Sensorer og reléer

Sensorene og reléene er selve grunnmuren i oppgaven. Hvis disse ikke fungerer er det ingen data å jobbe med, heller noen reléer å aktivere. Gruppen er blitt tilsendt Arduino utstyr (se til vedlegg O) som inneholder de nødvendige komponentene for gruppen å utbygge sensorer og fullføere oppgaven. Det er gruppens oppgave å sette disse sammen på en effektiv måte og programmere de så de kan sende og motta den nødvendige dataen. Gruppen skal også lage schematics av hvordan de forskjellige enhetene er satt sammen, for videre produksjon av disse. Gruppen skal ikke fokusere på avanserte reléer som kan ta full kontroll over et elektriskt apparat, men heller fokusere på at reléet skal skru apparatet på eller av.

1.3.2 Server og nettside

Dataen fra sensorene skal sendes til en Raspberry Pi som fungerer som et sentralt styringspunkt for sensorene og reléene. Raspberry Pien er ansvarlig for å sende denne dataen videre til en server, slik at dataen kan vises på en nettside. Nettsiden skal ha oversikt over hvilken Raspberry Pi, dens tilhørende sensorer og deres data som er knyttet til bru-

keren. Via nettsiden skal man også kunne sette opp regelverk for sitt system, som styrer når reléer skal skrus på eller av, basert på data fra sensorene.

1.3.3 Kommunikasjon

Det er også viktig at gruppen har en god løsning for kommunikasjon mellom alle disse komponentene, sensorer/reléer, Raspberry Pi og server/nettsiden. Gruppen står her fritt til å gjøre research og finne den løsningen som passer oppgaven best.

1.4 Målgruppe

Produktet skal være tilgjengelig for allmenheten ved bruk i både privat og offentlig sammenheng. Teknologien er hovedsakelig for oversikt og kontroll over klimaet i hjemmet eller bygninger. Den brede målgruppen er dermed alle de som ønsker disse kravene godkjent. For å tydeliggjøre det potensielle markedet og fremstille en mer realistisk målgruppe begrenses den ved å trekke inn ulike situasjoner som gruppen påvirkes av:

Tekonologisk interesserte

- Siden det skal være Open Source programvare vil gruppen av teknologisk interesserte ha muligheten å bidra til, og utbedre produktet. For denne målgruppen er dokumentasjon og instruksjoner viktig å ha tilstede slik at bidragsytere kan komme i gang. Teknologisk interesserte er også de som prioriterer bekvemmeligheten et styringssystem i lommen tilbyr.

Miljøvennlige

- Målgruppen som ønsker å spare miljø gjennom kontroll over sin egen energi konsumering og derav hvor mye utslipp de utsetter miljøet for.

Økonomi

- Målgruppen som ser muligheten til å redusere kostnadene sine ved ekstern styring av energiforbruk. Denne målgruppen ansees å være den største og mest potensielle målgruppen for produktet.

1.5 Faglig bakgrunn, rammer og øvrige roller

1.5.1 Gruppens faglige bakgrunn

Prosjektgruppen består av fire medlemmer, Martin Pukstad, Stian Fenstad, Jakob Fonstad og Kristian Sundhaugen. Hele gruppen består av programvareutviklere, der alle har tatt samme valgfag gjennom studietiden. Studentene har gjennom perioden hos NTNU fokusert på valgfag som er nyttige med tanke på utvikling, der hvor sikkerhet står sentralt. Disse valgfagene består blant annet av Database- og applikasjonsdrift og Programvaresikkerhet. Martin Pukstad har også en tidligere fagskoleutdanning fra TISIP fagskole innen IT-Drift.

Hva må læres?

Gjennom bacheloroppgaven vil gruppen få prøve seg innen flere nye teknologier, både innen hardware og software. Dette innebærer en stor mengde selvstudie innen Arduino og Raspberry Pi som tar den største biten innen ukjent hardware.

1.5.2 Rammer

Gruppen vil holde seg innenfor de rammene som er satt av universitetet når det gjelder gjennomføring av bachelor oppgaven. Dette medfører da at gruppen skal holde alle frister som er satt med tanke på statusrapporter, og selve innleveringen av prosjektrapporten den 16. mai. Gruppen jobber systematisk med fokus på planlegging, visualisering og gjennomføring etter kjente metoder innen IT-miljøet.

1.5.3 Øvrige roller

Oppdragsgiver

Oppdragsgiveren er Fosen Utvikling, som holder til i Indre Fosen kommune i Trøndelag. Dette er en liten bedrift med ildsjeler innenfor utvikling, sikkerhet og drift. Alle ansatte har andre arbeidsplasser og driver Fosen Utvikling som et hobbyprosjekt i tillegg. Fosen Utvikling tar på seg utvikling av programmvare for andre bedrifter i Trøndelag. De har også noen oppdrag hvor de står for vedlikehold og videreutvikling av produkter de selv har levert.

Prosessveileder

Veileder under bacheloroppgaven MySmartHome er Frode Haug. Han er universitetslektor ved NTNU Gjøvik og har undervist i flere av de obligatoriske fagene innen for programmvareutvikling. Veilederen har bidratt under hele prosjektet med tilbakemeldinger på arbeidsprosessen og rapporten.

Kontaktperson

Kontaktpersonen hos Fosen Utvikling er Jonas Tung Kirkemyr. Jonas har gjennomført Master i datateknikk hos NTNU i Trondheim og har jobbet for Fosen Utvikling i mange år. Han har gjennom prosjektet bidratt med tilbakemeldinger på produktet, i form av tips til planlegging og gjennomføring.

1.6 Arbeidsprosessen og prosjektorganisering

Gruppen består av heltidsstudenter og hadde dermed muligheten til å jobbe ofte og tett sammen. Gruppen valgte fra et tidlig tidspunkt at prosjektet skulle gjennomføres som om alle var på en arbeidsplass. Dette vil si i praksis at alle hadde en arbeidstid mellom 0900 til 1600 å forholde seg til, og at man skulle møtes på universitetet. Det ble innført et botsystem hvor man måtte betale om man var for sent ute til arbeidsdagens start. Denne boten gikk til en felles gode for studentene etter endt bachelor oppgave. Under hele prosjektet hadde gruppen regelmessig kontakt med både arbeidsgiver og veileder, da dette ville bidra til å skape et bedre produkt, og en bedre rapport.

Oppgaven var relativt åpen når det gjaldt valg av programmeringsspråk og gjennomføring. Eneste som var fastsatt var et ønske om at Arduino enhetene skulle ha oppkobling til WiFi og at all trafikk gikk igjennom en Raspberry Pi. Dette gjorde planleggingen inter-

essant ettersom gruppen selv måtte finne løsninger på hvordan hardware skulle bli satt opp før selve utviklingen kunne begynne. Dette gjaldt også hvordan gruppen valgte å planlegge utviklingsprosessen, da ingen spesielle krav ble satt utenom at oppdragsgiver ønsket fritt innsyn og at produktet skulle ende opp som Open Source.

Tidlig ble det bestemt at Jakob Fonstad skulle være prosjektleder da han har mest erfaring i lederstillinger, og dermed allerede var dyktig i delegering av arbeidsoppgaver. Gruppen valgte kort tid etter at prosjektet skulle gjennomføres ved hjelp av SCRUM. Dette ble valgt ettersom oppgaven hadde en uviss størrelse og at videreutvikling og utbygging krevde en smidig fremgang. Etter at gruppen hadde valgt å gjennomføre prosjektet ved hjelp av SCRUM metodikken ble også Jakob utnevnt til SCRUM-master, dette for å unngå rot innen lederskap og at gruppen fulgte en flat lederstruktur hvor alle tok ansvar for kommunikasjon med veileder og oppdragsgiver. Gruppen ønsket også en person med hovedansvar for referat av møtene internt i gruppen og møtene med veileder og oppdragsgiver. Dette ansvaret falt på Kristian Sundhaugen.

Etter samtaler med oppdragsgiver ble det gitt et tips om å gjennomføre én ukers spinter innen SCRUM, ettersom dette åpnet for muligheten til å legge til nye issues i backloggen jevnlig, og at man hadde større sannsynlighet for at man kunne gjennomføre sprinten før det dukket opp en annen oppgave som var viktigere. Alle sprintene ble planlagt å lagt på Jira, samt at det ble brukt Trello for planlegging av rapportskriving. I tillegg til dette ble også Google Drive brukt for å lagre å dele referater, kilder og annen viktig informasjon gjennom bacheloren.

1.7 Rapportens ordbruk og rapportorganisering

1.7.1 Ordbruk

Gruppen har basert seg på at leseren har kjennskap til ord og utrykk hyppig brukt innen IT-miljøet, men vil etter beste evne fornorske de engelske utrykkene. Dersom det skulle være uklare fagord vil disse befinne seg i ordlisten i begynnelsen av rapporten.

1.7.2 Avgrensing rapport

Denne oppgaven har basert seg på mye teknologi som ingen i gruppen har vært inne på i løpet av studietiden, det vil derfor være en del fokus i rapporten på valg som ble tatt, hvordan problem ble løst og hva slags forhåndsarbeid som ble gjort for at dette skulle kunne la seg gjennomføre. Det vil også være fokus på tematikk kjent fra fag som systemutvikling og objekt orientert systemutvikling. Da med tanke på temaer som programvaredesign, utviklingsmodell og patterns.

1.7.3 Rapportorganisering

Rapporten vil bestå av syv kapitler, med sine egne underkapitler. Når det kommer til rapportens struktur har gruppen valgt:

1.0 Innledning
Omhandler hele prosjektet i sin helhet. Her presenteres gruppen, arbeidsgiver, veileder, prosjektbeksrivelsen, avgrensingen og fremgangsmåten til oppgaven.

2.0 Forarbeid
Dette kapitelet går igjennom oppstartsperioden i prosjektet og forarbeidet gjort før utviklingen startet. Kapitellet beskriver også i detalj utviklingsmodellen SCRUM, som ble brukt under utviklingen.

3.0 Kravspesifikasjon
I dette kapitellet utdypes de kravene som er satt til programvaren som er utviklet. Det er også skrevet personas som gir eksempler på personer som kan tenkes å bruke programvaren. Use-cases blir også brukt for å beskrive bruksområdene til programvaren. Til slutt er det en detaljert kravspesifikasjon.

4.0 Arkitektur og design
Arkitektur og design kapitellet beskriver hver del av systemet og hvordan alt er satt sammen, uten å gå ned på kode nivå. Dette kapitellet gir også en god oversikt og beskrivelse av kommunikasjonen i systemet, og oppgavene til hver enkelt del i systemet.

5.0 Koding, kvalitetskontroll, implementering, og testing
I dette kapitellet går gruppen i dybden av programvaren og drar ut eksempler fra koden for å vise hvordan gruppen har tenkt og hvordan oppgaven ble løst. Kapitellet går også inn på hvordan kodekvaliteten ble ivaretatt.

6.0 Realisering og installasjon
Dette kapitellet viser hvordan man setter opp prosjektet selv for videreutvikling, samt hvordan installeringen av programvaren og tilkobling av systemet gjennomføres. Det er tatt eksempler fra hvordan løsningen ser ut per dags dato.

7.0 Avslutning og konklusjon
Her blir rapporten avsluttet ved å diskutere og kritisere oppgaven, samt å gå igjennom tanker og utfordringer gruppen har støttet på underveis i bachelor perioden.

Vedlegg
Alle vedlegg.

2 Forarbeid

2.1 Oppstartsfase

Innledningsvis i prosjektet hadde Fosen Utvikling et møte med gruppen hvor de la fram ideén for hva sluttproduktet kunne bli. Ettersom det eneste som var fastsatt av Fosen Utvikling når det gjaldt oppgaven var å benytte Arduino komponenter som sensorer og Raspberry Pi 2 eller 3 som gateway, så var det svært åpent for gruppen å velge utviklingsmetodikk, verktøy og fremgangsmåte selv. I møte fikk gruppen vite grunnleggende informasjon om hvordan en sensor skal settes opp og hvordan man kan utvikle mot hardware.

Det ble nødvendig å bruke første periode av prosjektet på å bli kjent med Arduino komponentene og Arduino sin IDE som skulle anvendes for utvikling. Dette ble gjort ved å sette opp hver enkelt sensor som temperatur/fuktighet, bevegelse, lys og lydsensor med tilhørende schematics og kode. Dette gjorde at gruppen fikk et godt innblikk i hvordan man skulle gå videre. Fra her var det viktig å undersøke om det fantes dokumentasjon tilgjengelig for hvordan man skulle koble opp sensorene til en gateway og videre til en server. Det fantes lite tilgjengelig annet en tips om hvilket språk som var lurt å anvende. Gruppen skjønte tidlig at det ville bli mange forskjellige språk og dermed vanskelig å ha ét verktøy for testing av kode.

Dette medførte at gruppen ønsket å sette opp en IDE som skulle håndtere alle forskjellige språk, og da falt valget på Eclipse. Hvis gruppen kunne forholde seg til én IDE ville testing foregå i SonarQube, men det viste seg fort at dette ble problematisk. Etter et møte valgte derfor gruppen å bruke Arduino IDE sammen med terminalen på enhetene som brukte Linux OS, dette da for å skrive Python, Javascript og Node.Js kode.

Det ble satt opp møte med veileder hvor det ble satt et fast tidspunkt hver uke hvor gruppen skulle møte opp å gå igjennom utviklingsprosessen, status og rapport. Dette var et godt anker for stabilt arbeid da gruppen alltid hadde noe å jobbe mot. I tillegg til møte med veileder, var det også avtalt et fast møte i uken med Fosen Utvikling som kun omhandlet utvikling.

2.2 Risikoanalyse

Under kommer det en risikoanalyse tabell(1), som tar for seg problematikk som kan oppstå i prosjektet. Det er også opprettet tiltak der risikoanalysen viser det er nødvendig.

2.2.1 Risikotabell

Nummer	Risiko	Sannsynlighet	Konsekvens	Tiltak
1	Prosjektet blir ikke ferdig i tide. Dette kan skyldes mange forskjellige årsaker som teknologiske problemer, tap av kildekode eller feilvurdering av tidsbruk.	Usannsynlig	Kritisk	Ja
2	Store endringer i kravspesifikasjoner fra oppdragsgiver.	Sannsynlig	Uproblematisk	Ja
3	Én eller flere utviklere blir syke over en lengre periode eller slutter under prosjektet.	Usannsynlig	Kritisk	Ja
4	Tap av kildekode eller rapport.	Usannsynlig	Kritisk	Ja
5	Andre firmaer utvikler lignende løsninger.	Svært Sannsynlig	Uproblematisk	Nei
6	Fosen Utvikling skrinlegger prosjektet.	Usannsynlig	Problematisk	Nei
7	Gruppen får ikke tilgang til all hardware og software nødvendig for oppgaven.	Usannsynlig	Kritisk	Ja
8	Problemer med kommunikasjon mellom sensorer og sentralpunkt.	Sannsynlig	Problematisk	Ja

Tabell 1: Risikotabell

2.2.2 Plan for håndtering av risikoanalyse

Nr	Tiltak
1	Hvis gruppen støter på problemer med å levere til deadline, så må produkteier informeres. Etter at dette er gjort må det diskuteres hva som kan bli fjernet fra kravspesifikasjonen. For å redusere sannsynligheten for at prosjektet blir forsinket, holder gruppen seg til sprintene, for at utviklingen skal ligge likt med planen som er satt i SCRUM. Planleggingen er viktig for å redusere risiko for forsinkelser.
2	Det er viktig med en god dialog med produkteier om utviklingens status, så han kan komme med innspill eller ønsker til nye user stories fortløpende. Jevne møter med produkteier, i gruppen internt og med veileder gir en minsker risikoen for at det dukker opp store endringer i kravspesifikasjonene som gruppen ikke kan håndtere. Det er også viktig å skrive gode referater fra alle møter og samtaler for å holde oversikt over endringer som kommer inn og hva som er fastsatt fra før, slik at det minker risikoen for misforståelser.
3	For at sykdom eller frafall ikke skal ha en stor innvirkning på prosjektet, holder gruppen daily SCRUM møter hver dag og retrospective møte i slutten av hver sprint. Dette gjør at alle i gruppen til enhver tid er oppdatert på hva de andre jobber med, slik at de lett kan hoppe inn der frafall eller sykdom intreffer.
4	For å unngå tap av rapport blir den lagret både i ShareLaTeXog Google Drive, i tillegg skal det tas en lokal backup med jevne mellomrom. Dette minimerer risikoen for at noe går tapt da alle har flere kopier. Når det gjelder kildekode så ligger dette lagret på Bitbucket samt lokalt hos alle utviklerne.
7	For å minimere risiko for at gruppen ikke får nødvendig hardware eller software, varsles oppdragsgiver der gruppen ser det oppstår mangler. Gruppen har også fullmakt for innkjøp av utstyr.
8	Hvis kommunikasjon mellom sensorer og sentralpunkt svikter må dette gi ut en varsel til nettsiden som holder oversikt slik at det er mulig å ta tak i problemet. Det skal også legges inn failsafe funksjonalitet i både sensorer og sentralpunkt slik at hvis internett eller strøm forsvinner må det kjøre automatiske restart eller oppstart for å fikse seg selv om mulig.

Tabell 2: Håndtering av risikoanalyse

2.3 Utviklingsmetodikk

En utviklingsmetode er en simplifisert representasjon av utviklingsprosessen. Hvor den valgte modellen viser utviklingen fra idé til et ferdig utviklet produkt[22]. Det finnes mange forskjellige metoder å velge i mellom, hvor hver har forskjellig tilnærming av hvordan prosessen for et ferdig produkt utvikles. Ut i fra hvilke krav som stilles av produktet, får man bestemt hvilke metode som egner seg best til dens prosess. I Prosjektplan(Se vedlegg G) kan man lese hvordan gruppen hadde planlagt å jobbe ved prosjektstart. Dette kapittelet vil gå grundig igjennom hvordan arbeidet gikk i forhold til planlegging og argumentasjon rundt valg tatt underveis.

2.3.1 Valg av metodikk

Karakteristika ved produktet som var styrende for valg av modell

- Vanskelig å estimere om først ønsket produkt var for lite for gruppen å ta fatt på, til gjengjeld hadde produktet mange muligheter til videreutvikling.
- Det var satt få rigide rammer og lite avgrensing rundt produkt, av produkteier.
- Gruppen måtte dynamisk tilpasse seg nye teknologier og utfordringer under utvikling, grunnet lite forkunnskaper rundt utvikling av lignende produkter.
- Produkteier ønsket å være en aktiv del av prosessen og komme med anbefalinger.

Argumentasjon
Fosen Utvikling hadde fra starten av gitt gruppen en tolkning av hvordan oppgaven kunne løses, hvor både rammer og avgrensingen av ønsket produkt var svært åpne. Dette ga fritt rom for gruppen å estimere hva som ville være tilstrekkelig å utvikle av produktets funksjonalitet, i den gitte tidsperioden, og hvordan gruppen ønsket selv å løse utviklingen. Siden oppgaven var fleksibel med tanke på inkludering av ny funksjonalitet, ville utviklingsmetodikker med mer rigide prosesser være mindre egnet. Ut i fra disse karakteristikkene bestemte gruppen seg for at en smidig utviklingsmetodikk, som SCRUM, ville være best egnet for produktets utviklingsprosess.

Ved å kunne dele produktet opp i sprinter, kan gruppen iterativt ta for seg de viktigste oppgavene i en sprint, som så gir muligheten til å legge inn mer funksjonalitet underveis i prosessen.

Siden produkteier ønsker å være en del av prosessen vil Sprint Review møter være godt egnet til dette. Møter med én til to ukers mellomrom lar produkteier tydelig se fremgang i utviklingen. Produkteier får da også komme med ønsker og anbefalinger til funksjonalitet mot produktet, som nødvendigvis ikke ble nevnt i starten av prosessen.

For å dokumentere prosessen og anvending av SCRUM vil verktøyet Jira bli brukt. Verktøyet ble valgt på grunnlag av tidligere erfaring.[22]

2.4 Gjennomføring

Gruppen hadde i forprosjektet valgt å kjøre én ukes sprinter, med mulighet til å gå over til to ukers sprinter. To ukers sprinter ble tatt i bruk når deler av prosjektets arbeidsmengde ble mer krevende enn estimert. Ved å ha to ukers sprinter ble det mindre Sprint Review

møter med produkteier, men fremdeles nok møter til å levere ønsket funksjoner.

2.4.1 Møter med produkteier

Sprint Review
Det ble avtalt med oppdragsgiver(i scrum terminologi: produkteier) at Sprint Review skulle holdes hver tirsdag på slutten av en sprint. Produkteier ville da informeres om fremgangen og hva som ble utført i sprinten, samt status for videre arbeid. Ved ønske ble fullført funksjonalitet demonstrert gjennom videoopptak.

Et eksempel er møte med produkteier 6. februar (se vedlegg M), hvor gruppen har sitt første Sprint Review møte. Her presenterer gruppen hva som ble utviklet til nå, eventuelle problemer oppstått og tiltak for disse. Møtet inneholder diskusjon rundt oppdeling av User Stories, siden daværende User Stories fort strakk over planlagt sprint. Dette ville da hjelpe gruppen å estimere omfanget og lettere se fremgangen mellom hver sprint.

2.4.2 Interne møter

Sprint Planning
Sprint Planning ble holdt under Sprint Review møte, dette var på grunnlag av produkteiers egen erfaring som utvikler. Produkteier kunne da komme med innspill til fordeling av User Stories eller Tasks mellom gruppens medlemmer, samt tids estimeringen av disse (se kapittel 2.4.5). I begynnelsen merket gruppen at flere User Stories ikke ble fullført innen gitt sprint, og ble derfor overført til neste. Dette var en indikator på for store User Stories, gruppen fokuserte dermed på å dele opp i mindre Tasks.

Sprint Retrospective
Gjennom utviklingsprosessen holdt gruppen Sprint Retrospective møter cirka én gang i måneden. Dette var så gruppen kunne drøfte om hva som hadde vært bra og hva som kunne blitt gjort bedre. Ved å holde Sprint Retrospective møter over en større samling sprinter, fikk gruppen et bedre innblikk i hva de kunne forbedre.

Det var et Sprint Retrospective møte som hjalp gruppen til å gå fra én ukes sprinter til to ukers, slik at sprintene strakk seg i forhold til hvor stor Tasken som skulle utføres, i stedet for at en Task ble flyttet over flere sprinter. Sprint Retrospective hjalp gruppen å forbedre utviklingen i rute med sprintenes krav, noe som kan ses i Sprintenes Burndown Chart for uke to og fire (Se til figur 1. og 2.).

Daily Sprint
Et daily sprint møte ble holdt ved starten av arbeidsdagen, hvor da et møte enten ville bli holdt samlet på grupperom eller over Skype. Dette var så gruppen fikk oversikt over hva som ble jobbet med, om det var noen problemer og om det var noe å vise frem. Eventuelt om det var noen spørsmål innad i gruppen eller avgjørelser som gruppen måtte ta.

2.4.3 Møte med veileder

Statusmøter med veileder Frode Haug har blitt holdt som oftest ukentlig, på torsdager kl. 14:45. Disse møtene har hatt fokus på prosjektplan og selve rapporten. Under utviklingsprosessen, hvor rapportskriving ble satt i mindre fokus, valgte gruppen også å ha en

pause fra statusmøter, grunnet lite rapport å vise til (Se vedlegg M).

2.4.4 SCRUM Board

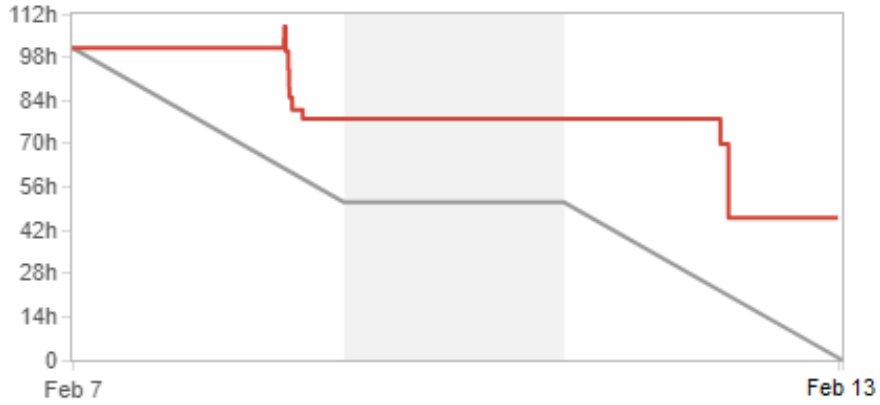
For issue tracking i SCRUM ble verktøyet Jira brukt, her la da gruppen inn samling av User Stories eller Tasks som kom opp etter samtaler med produkteier. Disse User Storiene eller Taskene ble lagt til i backloggen og neste sprints Tasks ble valgt ut ifra denne loggen.

2.4.5 Estimering

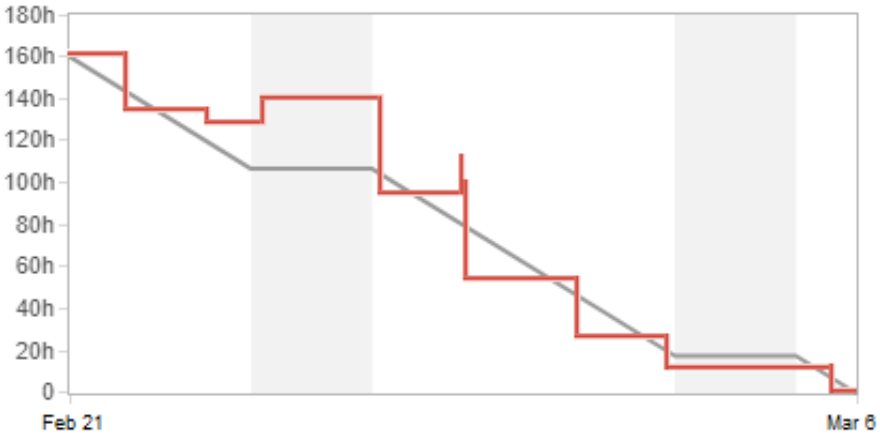
I SCRUM er User Stories ønskede funksjoner fra produkteier, disse deler da gruppen opp i Tasks, som gruppen skal estimere hvor mye arbeid som kreves. I starten hadde gruppen sett på arbeidsmetodikeken Planning Poker, som er en estimerings teknikk. Problemet med dette var at gruppen følte det var vanskelig å estimere hvor mye poeng en Task var verdt, og ønsket heller å komme igang med arbeidet, enn å sette seg inn i et helt nytt system. Derfor ble dette valgt bort for heller å ta i bruk times estimering, en annen metode gruppen hadde sett på.
Ved å bruke times estimering vil Tasks havne i kategorier ut i fra hvor lang tid det forventes å bruke på dem, hvorav:

- User stories som tar mindre enn én dag legges i kategoriene:
  - Én, to, fire, eller åtte timer
  - Eks. Estimerer man tre timer arbeid rundes dette opp til fire timer.
- User stories som tar mer enn én dag legges i kategoriene:
  - To, tre, fem eller sju dager
  - Eks. Estimerer man fire dagers arbeid rundes dette opp til fem.

I starten av prosessen var det vanskelig å estimere hvor lang tid man ville bruke på en Task, så gruppen støtte på hendelser hvor man estimerte eksempel åtte timers arbeid, men gjorde det ferdig på to. Dette medbringte ett fall i Burndown Chartet, men med tiden ble gruppen bedre til å estimere (Se til figur 1. og 2.). Ved å ha kontroll på hvor mye arbeid som måtte utføres hver sprint og å disiplinert fylle dette inn underveis, klarte gruppen å holde en jevn kurs mot ønsket mål.



Figur 1: Sprint 2



Figur 2: Sprint 4

2.5 Gjennomgang av sprinter

2.5.1 Oppsummert beslutninger tatt av gruppen

Gjennom gruppens arbeid med SCRUM metoden har visse valg måtte bli tatt i forhold til hva som ble planlagt under prosjektplanlegging. Først og fremst valgte gruppen å gå bort fra å jobbe med utviklingen og rapporten over 14 sprinter, grunnet at lengden på én ukes sprinter ikke strakk til. Etter sprint tre gikk gruppen over til to ukers sprinter for resterende arbeid, dette førte da til at det ble syv istedenfor 14 sprinter(Se vedlegg B. Samtidig ble også oppgaver relatert til rapporten flyttet fra SCRUM prosessen over til et eget Trello board. Dette var så gruppen kunne få et bedre overblikk over tidsbruken, og separere de to arbeids prosessene som skulle utføres. Dette førte til at enden på SCRUM prosessen ble to uker korter enn planlagt, og endte 17. april i stedet for 1. mai.

2.5.2 Oppsummering av sprinter

Prosjektplanlegging (9.januar - 31. januar)
Gruppen fullførte arbeid på prosjektplanen. Satte seg inn i de verktøy gruppen planlegger å bruke. Lagd repositories hvor koden til utført utvikling skal lagres. Hatt samtaler med produkteier om hva som ønskes av funksjonalitet.

Sprint 1 (31. januar - 6. februar)
Gruppen bygger ut sine første Arduino sensorer, samt kobler disse opp med kommunikasjon til en Raspberry Pi, og sender dens data over MQTT, i JSON format, og lagrer denne i database.

Sprint 2 (7. februar - 13. februar)
Gruppen har funnet en løsning til å sende sensor data ved bruk av WiFi, men har støtt på problemer med mangel på dokumentasjon for oppkobling, men fikk sensor for tempratur og luftfuktighet til å fungere. Raspberry Pi kan da motta data fra sensor koblet opp mot samme nettverk.

Sprint 3 (14. februar - 20. februar)
Det ble nødvendig å sette opp schematic for hvordan sensorer skulle lages, slik at arbeidet kunne gjenproduseres. Mulighet til å sende data over WiFi og lagre denne i database. Samtaler om videreutvikling rundt en webside, samt forlenge sprinter til to uker.

Sprint 4 (21. februar - 6. mars)
Ble brukt mye tid på backend løsning for hvordan sending av data fra Raspberry Pi til en webside, skulle gjøres. Gruppen så på mulighter, etter samtaler med produkteier, slik som rammeverket Node.Js for dette. Samtidig ble det holdt samtaler om frontend løsning ved bruk av React.JS, et interessant rammeverk for å lage singlepage.

Sprint 5 (7. mars - 20. mars)
Gruppen arbeider videre med mulighet til å sende data til webside. Samtidig brukes det tid på å skissere opp websidens utsendet, og mulig bruk av metodikken If This Then That(Se til 4.3.1), for oppsett av regelverk til websiden. Ble også valgt å ikke bruke tid på å sette opp systemt i React.JS, grunnet mangel på tid.

Sprint 6 (21. mars - 4. april)
En løsning for webside, med fokus på IFTTT er fungerende, men har støtt på problemer med integrering sammen med Node.Js. Webløsning med mulighet til å vise alle oppkoblede sensorer til en Raspberry Pi er fungerende. Samt mulighet til å vise temperatur data til en temperatur sensor dynamisk.

Sprint 7 (5. april - 17. april)
Opplasting av regelverk for sensorer til database, relé utbygging og tilkobling mot styringspunktet. Automatisering av ny Raspberry Pi koblet til et nettverk, lagring av data til webserver og avslutning av utviklingsperioden, for heller å fokusere på rapportskriving.

3 Kravspesifikasjon

Oppgaven går ut på å lage et automatisert system for avlesning av diverse sensorer med styring av aktuatorer gjennom et selvdefinert regelverk satt av en bruker.

Prosjektet består av en sammenkobling av Arduino sensorer, som gruppen selv har satt opp, et styringspunkt som er laget ved hjelp av en Raspberry Pi, og en webserver som håndterer brukersiden av produktet. Det er på nettsiden brukeren har tilgang til oversikter og informasjon om de forskjellige sensorene. Gruppen hadde i oppgave å lage grunnmuren til produktet da dette skulle ende med videreutvikling gjennom Open Source.

3.1 PACT Analyse

En viktig del av utviklingsprosessen er å kartlegge hvem som skal ta applikasjonen i bruk når prosjektet er ferdigstilt. Da dette prosjektet fokuserer på å få på plass grunnmuren i et større prosjekt, vil fokuset deles på brukere som tar i bruk den endelige løsningen, og de som ønsker å videreutvikle prosjektet. Her kan en PACT (People, Activites, Context and Technology) Analyse godt beskrive de bruksområdene og behovene prosjektet må dekke. Med en PACT analyse kan vi klart dekke hvordan målgruppen vår vil bruke Open Source prosjektet "MySmartHome".

3.1.1 People

Mennesker er forskjellige på mange måter, enten kognetivt eller fysisk. Det er derfor viktig å tenke på hvilke utfordringer personene som bruker applikasjonen kan ha. Dette kan være i form av fargekontraster, størrelsen på knapper i brukergrensesnittet, tilgjengelighet osv. Under utviklingen i dette bachelor prosjektet er det ikke satt et stort fokus på design for brukere, da prosjektet hovedsakelig er å lage grunnmuren i et Open Source prosjekt som skal videreutvikles. Det er likevel et tema som er verdt å ta opp i rapporten.

Vi ser for oss to forskjellige personer som skal ta i bruk løsningen, som skal bli beskrivet i denne PACT analysen. De som skal ta i bruk den endelige løsningen, og de som skal videreutvikle.

For de som skal ta i bruk den endelige løsningen er det viktig å ta høyde for at ikke alle har høy IT kompetanse. Den burde derfor designes med gjenkjennbare ikoner og knapper, som en bruker kan kjenne igjen fra tidligere IT løsninger. Samtidig som at det blir tatt høyde for eventuelle synsvansker, da den endelige applikasjonen for brukeren sin del vil være en nettside.

Det er satt fokus på at andre skal ta over utviklingen etter bachelor oppgaven er levert. Det er derfor viktig at hele prosessen og utviklingen blir grundig dokumentert. Samt at det er instruksjoner på plass for at det skal være så enkelt som mulig å installere prosjektet lokalt, og fortsette utvklingen. Det antas at de som skal videreutvikle prosjektet har en

relativt høy IT kompetanse, men også at de ikke nødvendigvis er kjent med teknologien som blir brukt.

3.1.2 Activities

Hovedhensikten med denne applikasjonen er å gi en person kontroll, styring og monitorering av eget hjem. Brukere skal, så lenge de er tilkoblet internett, kunne ha oversikt over data fra sensorer i hjemmet sitt og kontrollere diverse elektriske applikasjoner gjennom reléer, for eksempel varme eller lys.

De som skal videreutvikle skal lett kunne implementere nye sensorer inn i løsningen med bibliotekene som har blitt laget under utviklingen av dette bachelor prosjektet.

3.1.3 Context

Den endelige løsningen skal primært brukes i hjem, som en smarthus løsning. Løsningen er til en viss grad avhengig av en internett tilkobling. Det må derfor tas høyde for at internett tilkoblingen kan forsvinne. Det er også viktig at WiFi nettverket dekker nok av huset til at den når alle sensor og relée nodene. Løsningen er også satt opp slik at man kan se data fra sensorene, og konfigurere systemet, fra hvor som helst i verden så lenge man har tilgang på nett.

Technology

Applikasjonen for en bruker vil være en webløsning. Det vil si at det trengs internett tilkobling for å installere og sette opp systemet, samt monitorere og konfigurere senere. Systemet er satt opp slik at det vil fortsette å kunne fungere uten at den trenger å være koblet på internett, men da kun lokalt over WiFi. Eksempelvis kan en bruker sette opp et regelverk på nettisden, så sendes det regelverket og blir lagret lokalt på Raspberry Pi, som så kan styre over sensorene og releene uten tilkobling til internett. Nettsiden bruker ikke nok data til at det er noe spesielt å ta hensyn til med tanke på mobildata forbruk.

For de som skal videreutivkle er det viktig at det er så enkelt som mulig å ta opp og fortsette utviklingen, det settes derfor stor vekt på at installasjon og oppsett dokumenteres godt. Det er for eksempel laget Schematics for alle Arduino oppsett som er brukt i prosjektet, så det kun er å bestille deler å koble de sammen likt sånn som vist i schematicsene.

3.2 Personas

Personas er en metode brukt for å identifisere brukere av et produkt, i dette tilfellet MySmartHome løsningen. Videre kan det brukes for å lage use cases og videre definere hva hver persona trenger for å kunne ta full nytte av løsningen. Oppgaven i seg selv består av å gjøre grunnarbeidet for et Open Source prosjekt, så det vil ikke være en endelig ferdigstilt og klar til produksjon versjon av denne løsningen beskrevet i denne rapporten. Likevel er det ønskelig å lage personas som dekker behovene løsningen må dekke når den er klar til produksjon. Dette for å kunne gå i dybden på hva grunnarbeidet dekker nå, og hva som må være klart før overtakelse. Det er viktig å få tilfredstit et vidt spekter av personligheter som kunne tenke seg å ta i bruk løsningen, fra forskjellige ståsteder. Dette for å få et så stort bilde som mulig av det løsningen må kunne dekke av behov.

3.2.1 Julie Dahl

Julie er en vanlig person som eier et hus og ikke har all verdens med IT kunnskaper. Hun har nylig fått et tilbud om å få installert denne smarthus løsningen i hjemmet sitt. Julie ønsker å kunne kontrollere temperaturen og belysningen i sitt hjem. Julie ønsker at gjennomsnitts temperaturen skal være så nærme 22°grader i de rommene hun bruker mest, som for eksempel stuen og kjøkkenet. Mens soverommet burde være noe kjøligere. Dette løses ved at det installeres temperatur sensorer i disse rommene, og at de forskjellige varmeovnene kobles til reléer som kan skru disse på eller av hvis temperaturen er for langt under eller over det som er ønsket. Julie ønsker også å spare strøm ved å installere bevegelses sensorer som tar seg av å skru av og på belysning i rom hun ikke bruker så ofte, i dette tilfellet gangene i huset hennes, og garasjen. Julies ønsker skal lett kunne installeres av en installatør når løsningen er ferdig utviklet, og er det dette prosjektet har lagt ned grunnarbeidet for.

3.2.2 Morten Johansen

Morten er en hobbyutvikler og Open Source entusiast. Han har lenge selv hatt et ønske om å utvikle med Arduino brett og sensorer men har ikke helt funnet en god løsning for at disse skal kommunisere med PCen hans hjemme. En dag kom han over dette Open Source prosjektet og så løsningen som ble tatt i bruk for å koble Arduino brettene opp mot et WiFi nettverk og sende data over MQTT. Dette passet perfekt i hans løsning, og da denne kode biten er standardisert for å kunne fungere på en hvilken som helst sensor var det lett å implementere det i hans løsning. Biblioteket som skal legges til på sensorene må programmeres slik at mest mulig kan bli gjenbrukt i en hvilken som helst sensor, uten forkunnskap om nettverks kommunikasjon og MQTT. Dette er løst ved å lage et eget bibliotek som lett kan implementeres. Hvor det eneste som må programmeres inn er biblioteket eller koden for å hente ut data fra selve sensor modulen på Arduino brettet.

3.2.3 Susanne Rosenlund

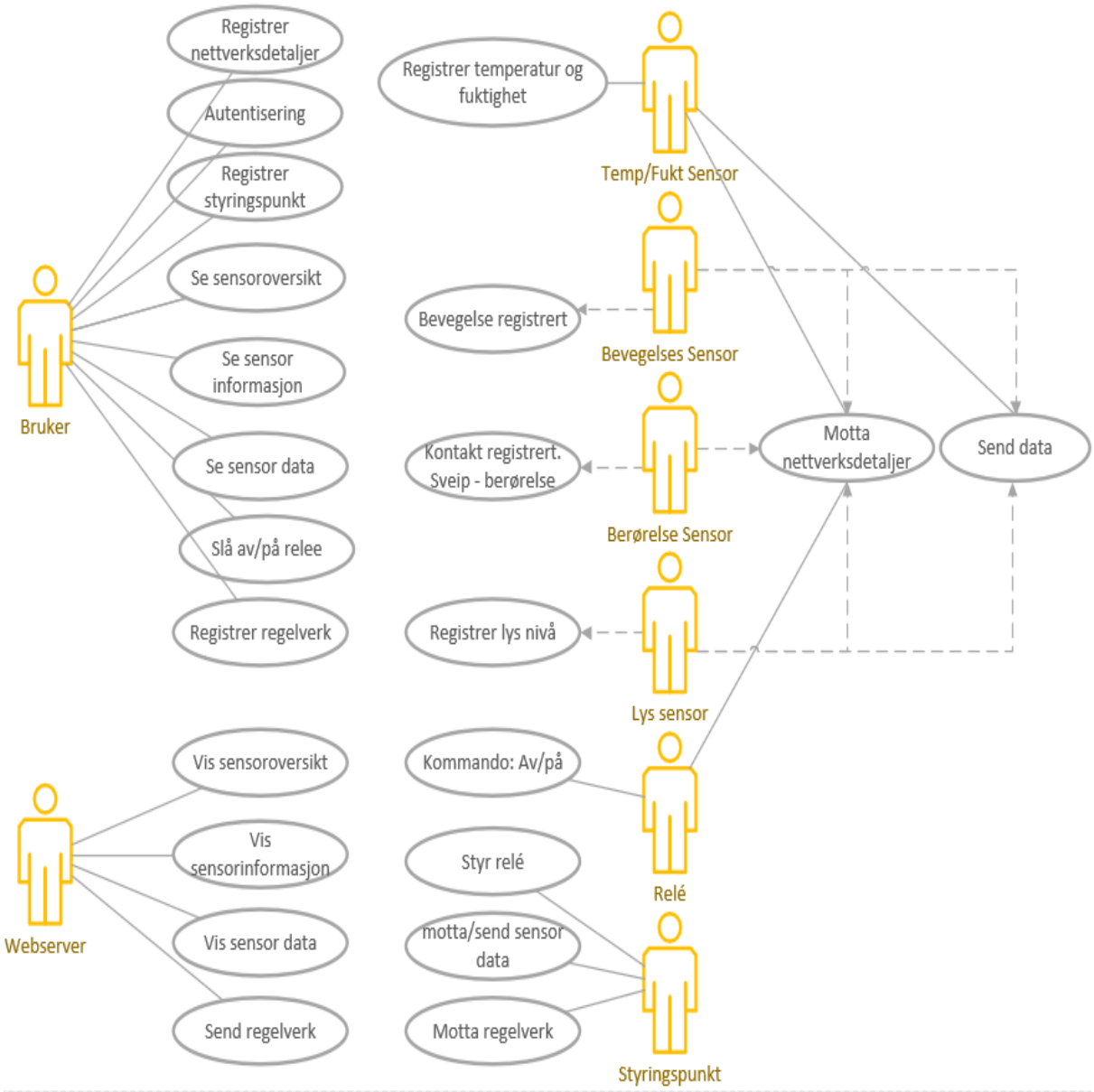
Susanne sitter i rullestol på grunn av en ulykke, og er derfor bevegelseshemmet. Dette gjør det vanskeligere å manøvrere seg rundt i hjemmet, og kan være spesielt frustrerende for små oppgaver som å skru av og på lys eller andre elektriske applikasjoner. Ved at denne løsningen blir installert i hjemmet hennes, kan hun styre individuelle reléer og slå ting av og på etter eget ønske fra et nettbrett, som er lett tilgjengelig. Dette vil gjøre hverdagen enklere for Susanne. Hun vil også kunne ha oversikt over sensor data i hjemmet sitt, dette kan for eksempel være bevegelses detektor, temperatur sensor eller andre sensorer som måtte ønskes.

3.3 Use Cases

Gruppen ønsket å bruke use case og use case-diagram da dette var kjent fra tidligere i utdanningen, og gruppen hadde god erfaring med nytten av en slik strukturert oversikt. Prosjektet er svært innviklet og bruker flere forskjellige maskinvarer, samt programmeringsspråk. Dette medfører at de forskjellige casene har variert kompleksitet, der hvor de tyngste blir forklart med utvidet use case.

Figur 3 viser alle use caser som har blitt implementert, samt noen som er planlagt til videreutvikling da dette prosjektet skal bli et Open Source produkt i henhold til oppgavebeskrivelse gitt av Fosen Utvikling. Use casene som er implementert er markert med hele linjer på use case-diagrammet, og de som skal implementeres gjennom videreutvikling er markert med den stiplede linjen.

3.3.1 Use case-diagram



Figur 3: Use case diagram (laget i Visio)

3.3.2 Use case-beskrivelser

Use case navn: Registrer temperatur og fuktighet
Aktører: Temp/Fukt Sensor
Mål: Hente ned temperatur og fuktighet
Beskrivelse: En bruker kobler opp sensoren i sitt hjem og sensoren begynner å registrere temperatur og fuktighet i rommet.

Use case navn: Motta nettverksdetaljer
Aktører: Temp/Fukt Sensor
Mål: Motta nettverksdetaljer fra WiFi manager
Beskrivelse: En bruker kobler til strøm til sensoren og får dermed muligheten til å endre nettverksinnstillinger gjennom Wifi Manager for oppsett mot styringspunkt med navn og tema.

Use case navn: Send data
Aktører: Temp/Fukt Sensor
Mål: Sende data til styringspunkt
Beskrivelse: Når en sensor er ferdig oppkoblet vil den automatisk sende data den har registrert til styringspunktet for lagring og videre arbeid.

Use case navn: Kommando Av/på
Aktører: Relé
Mål: Sende en av/på kommando til tilkoblet aktuator
Beskrivelse: Reléet sender en av/på kommando til tilkoblet aktuator, som varmepumpe, lys osv.

Use case navn: Motta nettverksdetaljer
Aktører: Relé
Mål: Motta nettverksinnstillinger gjennom WiFi Manager
Beskrivelse: Reléet settes opp gjennom WiFi Manager for å bli tilkoblet samme nettverk som sensorer og styringspunkt.

Use case navn: Motta/send sensor data
Aktører: Styringspunkt
Mål: Motta og sende data fra sensor
Beskrivelse: Styringspunkte lytter alltid etter ny data fra alle sensorer på et nettverk og sender til webserver.

Use case navn: Motta regelverk
Aktører: Styringspunkt
Mål: Motta regelverk fra webserver
Beskrivelse: En bruker sender selvdefinert regelverk til styringspunkt for styring av relé.

Use case navn: Styr relé
Aktører: Styringspunkt
Mål: Styre et relé ut ifra brukerens regelverk
Beskrivelse: Styringspunktet styrer av og på kommandoen til et relé ut i fra det selvdefinerte regelverket satt av brukeren. Ut i fra regelverket vet styringspunktet når reléet skal på og når det skal av.

Use case navn: Registrer nettverksdetaljer
Aktører: Bruker
Mål: Registrere nettverksdetaljer for sensor
Beskrivelse: Brukeren kobler seg opp mot sensor via WiFi Manager og registrerer nettverksinnstillinger for sitt hjemmenettverk og oppretter kontakt mellom sensor og styringspunkt.

Use case navn: Autentisering
Aktører: Bruker
Mål: Autentisering av bruker på nettside
Beskrivelse: Brukeren logger på nettsiden med brukernavn og passord for tilgang til sin profil.

Use case navn: Registrer styringspunkt
Aktører: Bruker
Mål: Registrer ens eget styringspunkt for oversikt over enheter
Beskrivelse: Brukeren skriver inn medsendt ID i nettsiden for oversikt over alle egne enheter, som også åpner tilgang til regelverk og styring.

Use case navn: Se sensoroversikt
Aktører: Bruker
Mål: Se oversikt over alle tilgjengelige sensorer
Beskrivelse: Brukeren ser oversikt over alle sine tilgjengelige sensorer på det interne nettverket.

Use case navn: Se sensor data
Aktører: Bruker
Mål: Se dynamisk oppdatert data fra sensor
Beskrivelse: Brukeren ser kontinuerlig oppdatert data fra sensoren inne på nettsiden, og kan følge med på endringer i sanntid.

Use case navn: Slå av/på relé
Aktører: Bruker
Mål: Slå av og på et relé gjennom nettsiden
Beskrivelse: Brukeren kan endre status på et relé gjennom nettsiden ved å sende nytt regelverk til styringspunktet.

Use case navn: Registrer regelverk
Aktører: Bruker
Mål: Lage egendefinert regelverk
Beskrivelse: Brukeren setter egendefinert regelverk som sender kommandoer til reléer ut i fra ønsker adferd.

Use case navn: Vis sensoroversikt
Aktører: Webserver
Mål: Vise sensoroversikt
Beskrivelse: Webserveren viser oversikt til brukeren over alle hans tilgjengelige sensorer i nettverket. Viser også all informasjon om sensoren, dvs navn, tema og lokasjon.

Use case navn: Vis sensor data
Aktører: Webserver
Mål: Vise sensor data
Beskrivelse: Webserveren viser kontinuerlig oppdatert data sendt fra sensor og styringspunkt ut på nettsiden slik at brukeren har oversikt.

Use case navn: Send regelverk
Aktører: Webserver
Mål: Sende egendefinert regelverk
Beskrivelse: Webserveren sender egendefinert regelverk fra brukeren til styringspunktet slik at adferd på relé og sensorer blir opprettholdt.

3.3.3 Use case (utvidet)

Use case navn: Motta/send sensor data
Primær aktør: Styringspunkt
Omfang: Sensor, styringspunkt og webserver
Mål: Motta og sende data fra alle sensorer på et nettverk.
Beskrivelse: Styringspunktet lytter etter alle temaer som blir sendt fra alle sensorer på et nettverk. Dette skal sendes videre fortløpende til webserver for visning til bruker.
Type: Essensiell
Pre-betingelser:

- MySmartHome Styringspunkt
- Tilgang til strømuttak.
- Tilgang til trådløst eller kabelnettverk.
- Oppkoblet sensor på samme nettverk.

Post-betingelser: Styringspunktet mottar data og sender videre kontinuerlig.
Grunnleggende programflyt:

1. Brukeren kobler styringsenheten til strøm.

Use case navn: Registrer nettverksdetaljer
Primær aktør: Bruker
Omfang: Temp/Fukt Sensor
Mål: Registrere nettverksdetaljer for Temp/fukt sensor
Beskrivelse: Brukeren kobler seg opp mot sensor via WiFi Manager og registrerer nettverksinnstillinger for sitt hjemmenettverk og oppretter kontakt mellom sensor og styringspunkt for oppdatering av data i sanntid over nettverket.
Type: Essensiell
Pre-betingelser:

- Trådløs internett tilgang, innenfor **50m** rekkevidde til sensor.
- MySmartHome DHT22 Temperatur/fuktighet sensor.
- Batteri.
- Enhet med tilgang til trådløse nettverk.

Post-betingelser: Sensor sender kontiuerlig temperatur- og fuktighets data.
Grunnleggende programflyt:

1. Brukeren setter inn batteri i sensoren.
2. Sensoren starter og oppretter et eget nettverk.
3. Brukeren går inn på telefon eller datamaskin med oversikt over alle tilgjengelige trådløse nettverk.
4. Brukeren trykker på MySmartHome nettverket som automatisk dukker opp.
5. Brukeren trykker på Konfigurer nettverksinnstillinger som automatisk kommer opp etter valgt nettverk.
6. Brukeren velger ønsket nettverk som sensoren skal kobles til.
    1. Brukeren skriver passord for valgt nettverk
7. Brukeren gir sensoren et navn.
8. Brukeren setter et tema for styringspunkt å lytte etter.
9. Brukeren skriver inn medsendt brukernavn og passord for styringspunktet som sensoren skal kobles til.

3.4 Product backlog

Product backlog ble fortløpende oppdatert i Jira med samarbeid fra kontaktperson hos Fosen Utvikling. Her ble alle Tasks og Issues registrert og lagt inn i sprinter i henhold til bruk av SCRUM metoden. Gjennom prosjektet har oppdragsgiver gått igjennom backlog sammen med gruppen og gitt tips for videre arbeid både på utvikling og planlegging. Gruppen lærte mye gjennom disse samtalene når det gjaldt størrelse på hver User Story, da gruppen gjorde hver Task litt for stor, når det kom til estimering av tid og arbeidsmengde.

3.5 Kvalitetsmessige og operasjonelle krav

3.5.1 Kvalitetsmessige krav

Brukervennlighet

Innenfor kvalitetsmessige krav er brukervennlighet viktig for utviklerene. Dette ble ikke sett på som et viktig punkt fra oppdragsgiver da dette skal videreutvikles i ettertid. Produktet er satt opp til å gjøre alt etter oppkoblingen automatisk. Først vil produktet bli satt opp av leverandør Fosen Utvikling med ID for autentisering mot nettside, samt brukernavn og passord som kan endres av brukeren i ettertid. Dette festes bak det medsendte styringspunktet, slik som det ville blitt gjort på et modem, slik at det er enkelt for brukeren å få tilgang til nødvendig informasjon. Etter dette er produktet fullt automatisert når det gjelder oppkobling.

Pålitelighet

Det er lagt inn fail-safes med tanke på tap av internett forbindelse og strømbrudd. Det vil da si at systemes selv starter alle nødvendige programmer på nytt til problemet er i orden. Dette sørger for at brukeren ikke trenger å bekymre seg.

3.5.2 Operasjonelle krav

Sammen med oppdragsgiver er det satt noen operasjonelle krav til et ferdigstilt, fullt oppkoblet produkt. Det er tatt hensyn til at produktet skal ende opp som Open Source og kravene som er stilt er med tanke på produktet etter videreutvikling.

- Systemet takler opp mot uendelig tilkoblinger av sensorer, så lenge det er IP-adresser tilgjengelig.
    - Systemet er bygd for gjennomsnittlig bruk av sensorer med opp til fem sensorer per rom i et hus.
- Det eksisterer fail-safe i forhold til tap av internett forbindelse og strømbrudd.
- Produktets nettside skal være tilgjengelig på alle enheter med internett tilkobling og nettleser.

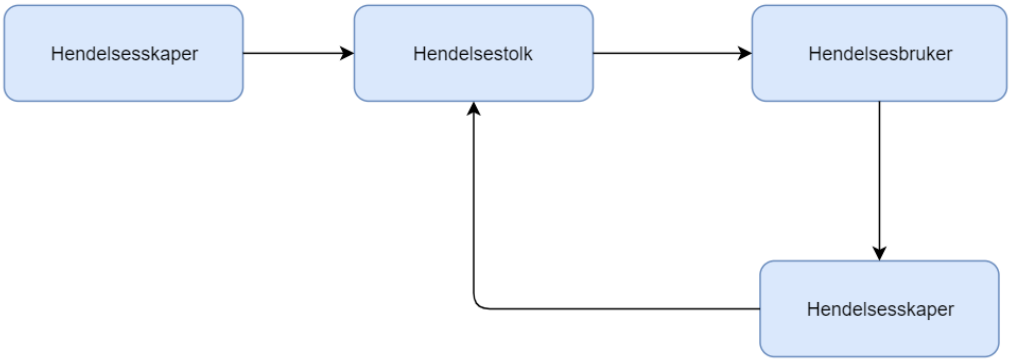
4 Arkitektur og design

Gjennom prosjektet har gruppen sett på forskjellige typer arkitekturer for å gjennomføre oppgaven. Valgene for arkitektur ble gjort med tanke på at prosjektet var et Internet of Things (IOT) system. I dette kapittelet vil de forskjellige arkitekturene som er blitt brukt bli forklart, samt en gjennomgang på idéen av et design for hvordan en nettside ville sett ut.

4.1 Arkitektur

4.1.1 Event-Driven Architecture

En hendelsesdrevet arkitektur [41] er sammensatt av en hendelsesskaper, hendelsestolk og en hendelsesbruker. Konseptet baserer seg på at det skal skapes en strøm av hendelser i sanntid, slik at lytterne på hendelser kan reagere og respondere så fort de oppstår. En hendelsesskaper vet ikke hvilke brukere som lytter, og hver bruker kommuniserer ikke direkte med hverandre, men ser alle hendelser som oppstår. I dette prosjektet fungerer dette konseptet to veier, ved at en bruker også kan skape nye hendelser, gjennom bruken av Express.js rammeverket og Node.Js.

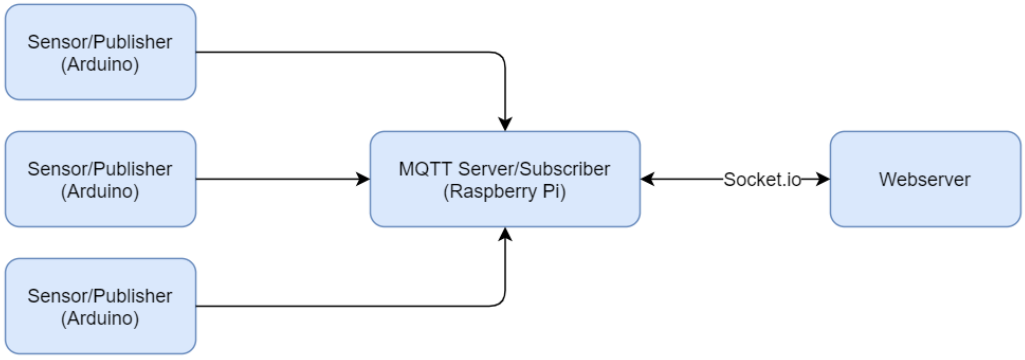


Figur 4: Event-Driven arkitektur

4.1.2 Publish/Subscribe Pattern

Et publiser/abonner mønster [42] ligger under et meldingsmønster hvor en publiserer sender en melding som kan bli tatt opp av en abonnent. Det er ingen garanti for at meldingen blir tatt opp ettersom meldingen ikke er programmert til å sende til en spesifikk abonnent, men heller åpner muligheten for at alle abonnenter som søker etter akkurat dette temaet, plukker opp meldingen og reagerer. I dette prosjektet blir dette gjort ved bruk av en Message Queuing Telemetry Transport (MQTT) server som er installert og konfigurert på styringspunktet. Styringspunktet lytter etter alle temaer innenfor et internt nettverk av sensorer og reagerer forskjellig ut i fra hvilket tema som sender data.

Videre fungerer dette mot og tilbake til en webserver gjennom bruken av Socket.IO, hvor det blir satt opp kanaler for lytting og publisering på begge sider som fungerer som triggere for hverandre.



Figur 5: Publish/Subscribe pattern

I dette prosjektet brukes JSON formatet for sending av data fra både sensor og styringspunkt gjennom publiser/abonner mønsteret.

Listing 4.1: JSON fra sensor

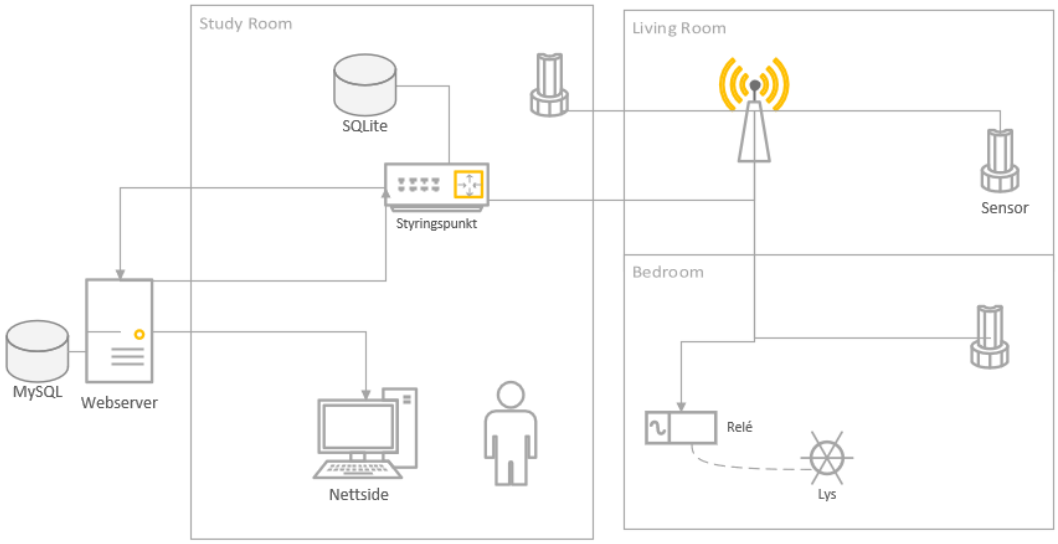
```
{
    "Sensor_ID" : "sensor_id",
    "data" : "payload"
}
```

Listing 4.2: JSON fra Styringspunkt

```
{
    "SensorPacket":
    {
        "Sensor_ID" : "sensor_id",
        "Raspberry_ID" : "rasp_id",
        "Topic" : "topic",
        "DateTime" : "datetime",
        "Data" : "payload"
    }
}
```

4.1.3 MySmartHome

Siden produktet skulle bygges fra bunnen av og opp var det nødvendig å foreta ulike valg om systemets oppsett og komposisjon under utviklingsperioden. Figur 6 viser systemets overordnede struktur, og hvordan de ulike komponentene kommuniserer med hverandre.



Figur 6: Komponent-diagram

Systemet inneholder mange sammenflettede enheter og ut ifra deres forbindelser er det nødvendig å dele opp systemet i moduler for å danne et helhetsbilde. Modulene består av de ulike komponentene i produktet, disse er sensor, relé, styringspunkt og webapplikasjonen. Hovedsakelig er produktet bygd på en hendelsesdrevet arkitektur [41]. Dette innebærer at systemet skaper og håndterer hendelser ettersom produserte hendelser oppstår. Prosjektet er basert på et oppsett av IoT, Internet of Things, og passer derfor godt inn under hendelsesdrevet arkitektur.

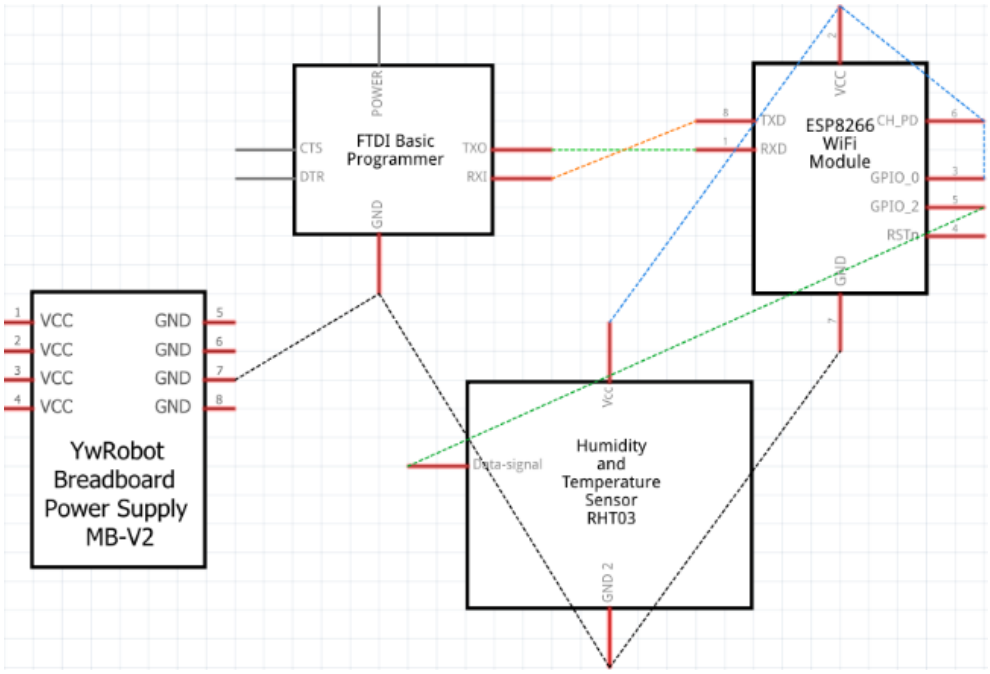
- **Sensor:**
  - Prosjektet legger vekt på muligheten til utbygging av flere sensorer om brukeren ønsker det. Det er mulig å installere flere av de samme sensorene i samme rom eller i forskjellige rom. Alt er tilkoblet og snakker sammen gjennom det interne trådløse nettverket. Prosjektet har hatt stort fokus på videreutvikling og har utviklet alle komponenter til å håndtere de nye kommende sensorene som er planlagt for videre arbeid. Disse sensorene er berørelse, bevegelse, mikrofon og lysstyrke-registrering. Innenfor en hendelsesdrevet arkitektur fungerer sensoren som en pågående hendelse som alltid sender oppdatert data. Dette håndteres gjennom publiser/abonner mønsteret som betyr at sensoren publiserer en hendelse, som igjen mottas gjennom styringspunktets MQTT server som abonnerer på alle temaer satt av Sensor.

- **Styringspunkt:**
  - Styringspunktet mottar temaer på sin MQTT server gjennom en programmert lytter. Denne lytter etter alle temaer på nettverket og lagrer i en database før videresending til webserver og nettsiden. Styringspunktet bruker Express.js, Node.Js og Socket.IO rammeverk som støtter hendelsesdrevet arkitektur og publiser/abonner mønsteret for asynkron avlesning av hendelser i sanntid.
- **Webserver:**
  - Webserveren benytter også Express.js, Node.Js og Socket.IO gjennom hendelser mellom styringspunktet og seg selv. Webserveren og styringspunktet kommuniserer sammen ved hjelp av triggere som mottar og sender sensor data fra styringspunktet og regelverk fra brukeren via nettsiden. Kommunikasjonen mellom Webserveren og styringspunktetene foregår over en Node.Js modul som heter Socket.IO. Socket.IO oppretter en socket på webserveren som styringspunktet kan koble seg opp mot. Socket.IO blir så differensiert på forskjellige kanaler, for generell kommunikasjon og forskjellige sensorer. Det er i tillegg satt opp slik at webserveren lagrer socket id på alle styringspunkt tilkoblinger, dette for å enkelt kunne sende ut for eksempel nye regelverk til et bestemt styringspunkt, i stedet for å sende til alle styringspunkt som lytter på samme kanalen.
- **Nettside:**
  - Nettsiden skal gi oversikt til brukeren over alle ens egne sensorer med muligheten til å sette regelverk for styring av relé som kan kobles opp til diverse aktuatorer, som varmepumpe, lys osv. Nettsiden var ikke en del av oppgaven fra begynnelsen men etter samtaler med oppdragsgiver fikk gruppen og Fosen Utvikling en forståelse for hvordan nettsiden ville se ut. Nettsiden viser fortløpende oppdatert data sendt fra sensorene gjennom styringspunktet og webserveren.

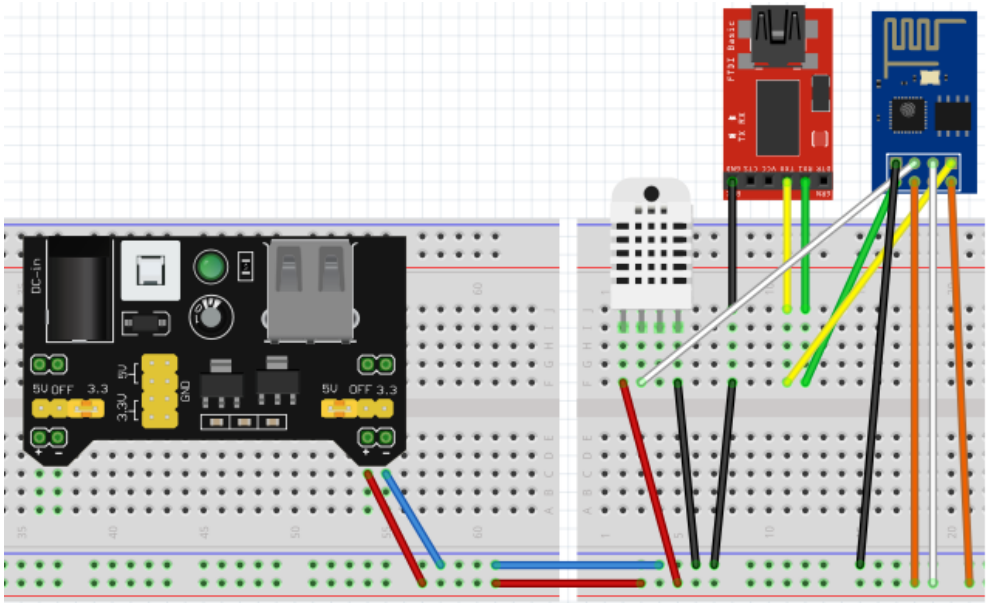
4.2 Design

4.2.1 Sensor

Sensoren er bygd opp med tanke på at den skulle koble seg til et nettverk, finne og gjenkjenne sitt styringspunkt for å deretter sende sin data. Ferdigstillingen av sensoren ble utbygd over flere iterasjoner av prototyping. Gruppen startet sensor utbyggingen mot mikrokontrolleren Arduino Uno, dette er et allsidig brett som er vennlig for nybegynnere. Under utviklingen og læringsprosessen var det nødvendig å integrere nye komponenter for å tilfredstille kravene en ferdigstilt sensor skulle ha. Ettersom sensoren skulle være eksternt kontrollert ble videre utvikling utført på en Arduino NodeMCU mikrokontroller, dette var et brett som var likt Arduino Uno brettet, men med integrert WiFi modul. Gjennom møter med produkteier viste det seg at både Arduino Uno og NodeMCU mikrokontrollere var kun egnet som en testplattform, og dersom sensoren skulle kunne produseres måtte sensoren kun ha de spesifikke komponentene som innfrir kravene stilt til enheten. Den ferdigstilte temperatur og fuktighet sensoren i Figur 7 og Figur 8 tilfredstiller en produksjonsklar enhet.



Figur 7: Schematic for temperatur/fuktighet sensor



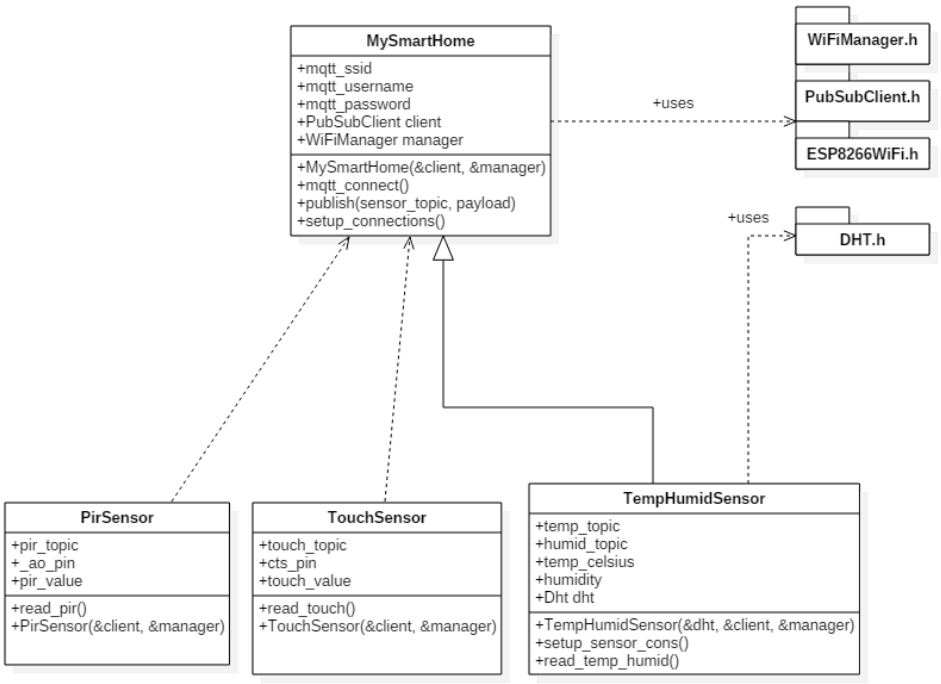
Figur 8: Sketch for temperatur/fuktighet sensor

Figur 7 og figur 8 viser til produktets ferdiglagde DHT-22 temperatur og fuktighetssensor. Mikrochips som sensoren er bygd opp med for å tilfredsstille dens funksjonelle krav:

- FTDI Basic Programmer
  - Tillater produkteieren Fosen utvikling å laste opp programvaren som skal kjøres på sensoren.

- ESP8266 WiFi Module
  - Tillater sensoren å koble seg til nettverk, samt opprette tilkoblingspunkter.
- Humidiy and Temperatur Sensor RHT03(DHT22)
  - Registrerer omgivelsenes temperatur og fuktighet gjennom en thermistor
- YwRobot
  - Tillater sensoren å kjøre ved hjelp av AA-batterier, USB 3.0 eller DC: 6.5-12V

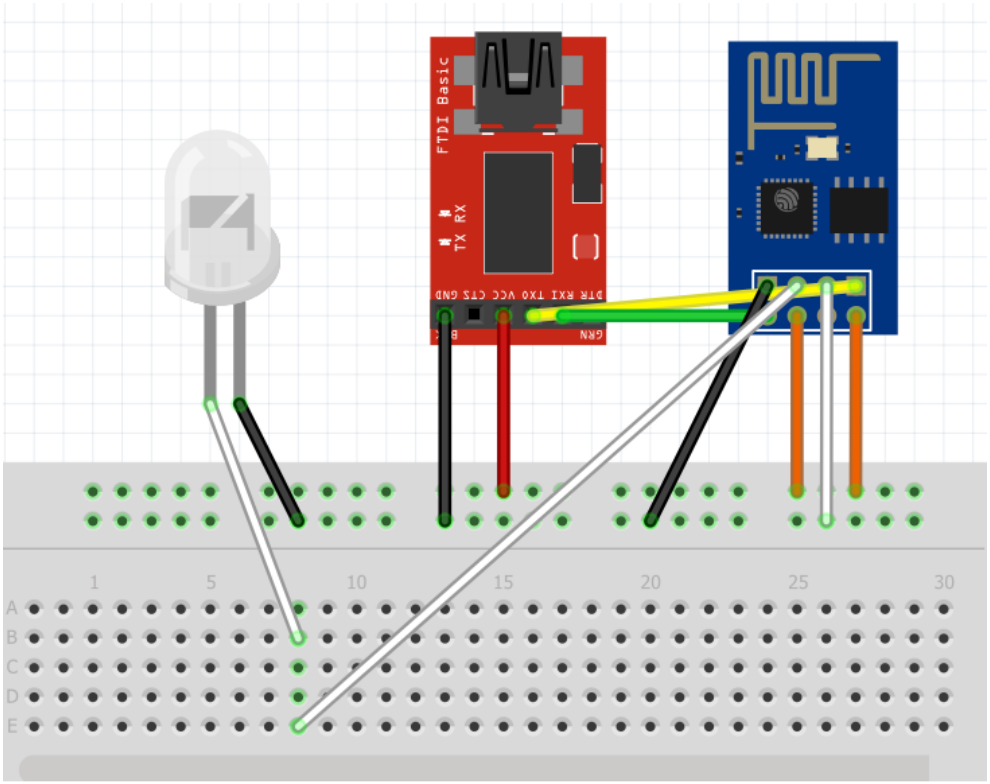
Videreutvikling er en viktig del av produktet siden det er et Open Source prosjekt. Det ble dermed dannet et bibliotek som inneholder en klassebasert arv struktur slik at bidragsytere enklere kan implementere nye sensorer. Biblioteket MySmartHome oppretter et tilkoblingspunkt dersom det er sensorens første oppstart. Dette utføres for å lagre hvilket nettverk sensoren skal operere på, og for å etablere en kommunikasjonsprotokoll mot styringspunktet som da ligger på det samme nettverket. Biblioteket tar imot innlest data og publiserer denne til styringspunktet. Bidragsytere behøver kun å sette opp en ny sensor klasse og forsikre seg at denne kan lese inn data.



Figur 9: MySmartHome sensor bibliotekoversikt

4.2.2 Relé

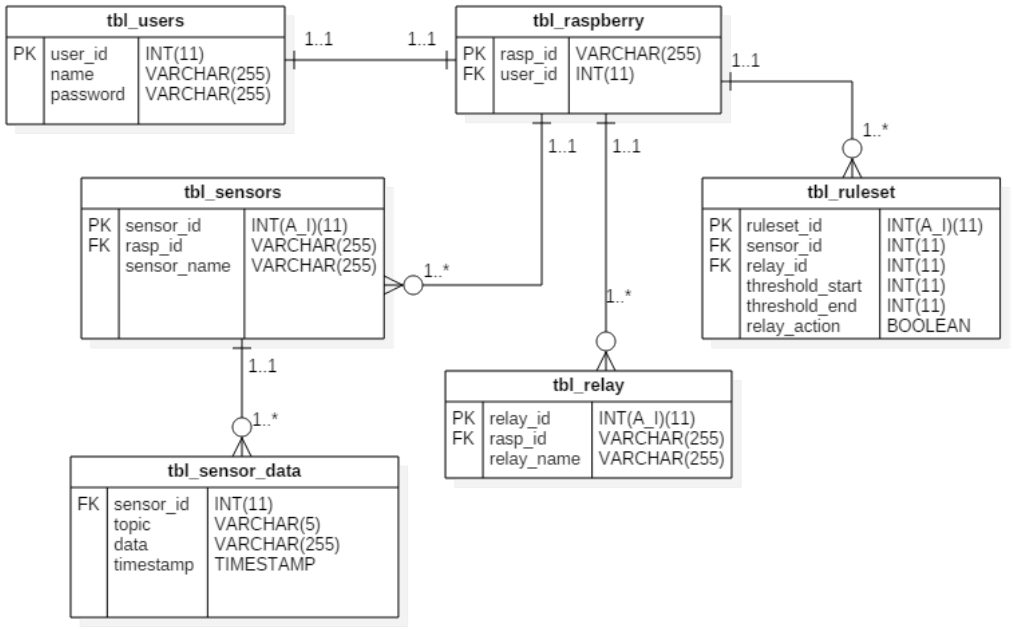
Relé schematicen i Figur 10 er kun en illustrasjon av hvordan et relé skulle fungere. Denne kan kun utføre binære handlinger, og dermed kan den reelt kun håndtere gjenstander med samme kompleksitet. Led lyset i figur 10 illustrerer en slik av/på funksjonaliteten. Tanken var at dette led lyset kunne byttes ut mot en aktuator som er tilkoblet en av og på gjenstand. Reléet har samme tilkoblingsoppsett som sensoren for å gjenkjenne styringspunktet, men i motsetning til sensoren vil reléet abonnere på et tema spesifisert fra styringspunktet.



Figur 10: Schematic for av/på relé

4.2.3 Database

Databasen skal strukturere informasjonen slik at en bestemt brukers sensors informasjon kan hentes. Gjennom studiene har det vært stort fokus på relasjons databaser og normalisering av data. MySQL ble da et naturlig valg for håndtering av database transaksjoner på webserversiden for å redusere redundant data. Brukertabellen skal forekomme på Fosen Utviklings egen plattform. Den er dermed lagt til kun for testing og visualisering av database oppsettet. Hver bruker skal kun kobles til en Raspberry Pi som er styringspunktet i smarthjemmet. Styringspunktet har relasjoner til alle sensorene, reléene og regelsett i sitt smarthjem. Hver sensor har relasjon til sin sensors informasjon.



Figur 11: Enhetsforhold diagram

Socket.IO protokollen fra styringspunktet sender kontinuerlig innsamlede data til webserverens adresse. Denne dataen tilsendes i JSON format som vist i Figur 4.2. Fra denne pakken sjekker webserveren om styringspunktets id er opprettet i databasen, og dermed tilkoblet en bruker. Dersom brukeren har koblet styringspunktet sitt til brukerprofilen sin vil tabellene automatisk fylles basert på datapakken tilsendt.

4.3 Idé for webapplikasjonens brukergrensesnitt

Gruppen ble enig om videreutvikling rundt et webgrensesnitt for produktet, gjennom samtaler med oppdragsgiver i Sprint Planning møte for sprint 4(Se vedlegg M. Referat 20. februar). Gruppen kom frem til at de ønsket å starte arbeid på en webløsning, for å representere data fra sensor, samt mulighet til å lage regelverk for handlinger rundt sensors data.

For løsning av logikken for regelverk fikk gruppen beskjed av oppdragsgiver å se på IFTTT - metoden. Metoden passet godt inn i forhold til hva gruppen ønsket å oppnå.[43] Videre gjennom brukertester kom gruppen frem til en mulig løsning av et brukergrensesnitt. Av dette ble fokus for gruppen å fremstille en løsning for å vise temperatur data, samt muligheter for å sette opp et regelverk for en sensor. Den endelige løsningen av brukergrensesnitt ble ikke implementert i nåværende produkt, grunnet mangel på tid, men den fremstilte logikken er hva gruppen så som en mulig løsning av brukergrensesnittet.

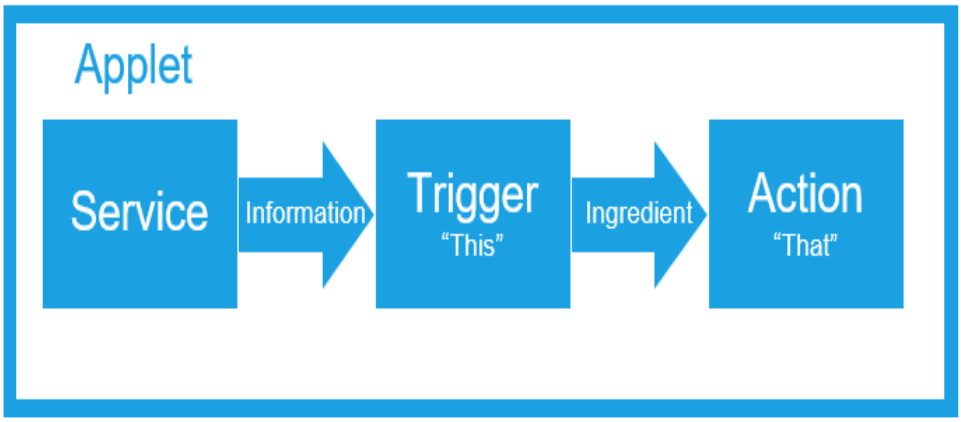
4.3.1 "If This Then That" metoden

Gruppen ble anbefalt metoden If This Then That (IFTTT), for logikken til regelverk. Dette er en web basert service skapt for å la flere teknologier koble seg sammen og samkjøre rundt en ønsket hendelse eller handling.[43]

Metoden kjører gjennom konseptene[43]:

- **Service:** hvor en samling informasjon kommer i fra. Dette er da informasjon som videre kan tas i bruk. Eksempel er informasjon rundt hvordan været på en spesifikk lokasjon er. Hver service har en viss mengde Triggers og Actions som kan brukes.
- **Trigger:** hva som får en Action til å skje. Dette kan da være et nøkkelord eller en viss verdi som nås. Trigger står for "This" i IFTTT metoden.
- **Action:** er da hva som kommer ut som resultatet av inputet til en Trigger. Action står for "That" i IFTTT metoden.
- **Applets:** kan sammenlignes med oppskrifter eller regelverk for hva en Trigger og Action fører til. Dette kan for eksempel være at når klokken er 8:00 (Trigger), så sendes en melding til en mobil med informasjon om været i dag(Action).
- **Ingredients:** er hva som er mulig av data å oppnå fra en Trigger. Dette er simpel data, slik som enkle verdier eller nøkkelord. Eksempel adresser, et ord, én verdi osv.

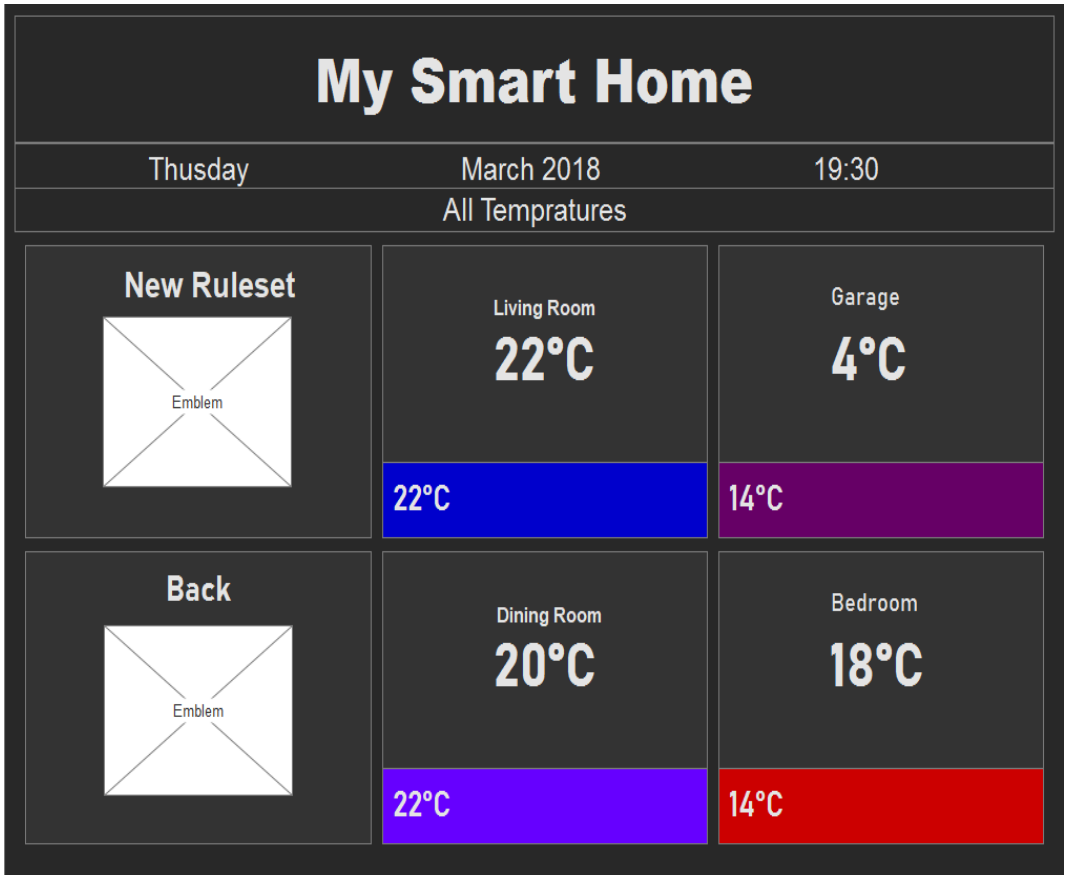
Figur 12 er en visualisering av logikken til IFTTT metoden. I kapitel 4.3.2 Idé for webløsning vil det bli vist hvordan gruppen tok i bruk denne metoden for brukergrensesnittet.



Figur 12: Representasjon av If This Then That metoden

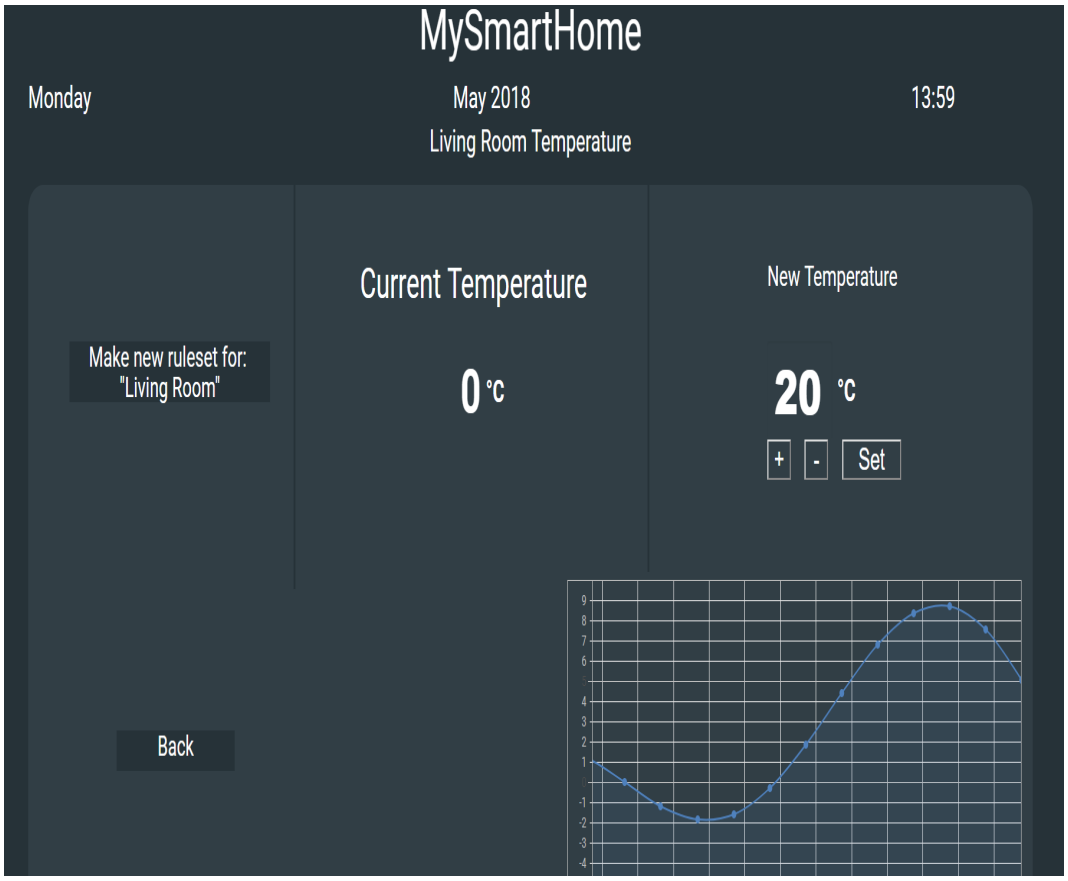
4.3.2 Idé for webløsning

Figur 13 er en skisse for hvordan gruppen ville representere data fra flere temperatur sensorer eller "All Tempratures" for én bruker. Dette ble satt opp som at hver sensor ville vises som en konteiner med informasjon. Hver konteiner inneholder sensors tittel for sin lokasjon, samt sin nåværende temperatur i stor tekst. Det vil også være en liten tekst for ønsket temperatur satt av bruker, nede til venstre i hver konteiner. Fargene ble valgt som skiller mellom de forskjellige lokasjonene. Videre ville det være mulighet til å gå tilbake til en tidligere side i webgrensesnittet med "Back" eller velge å sette opp et nytt regelverk gjennom å trykke på "New Ruleset". Ellers kan man også klikke seg inn på en spesifikk sensor, ved å klikke på dens konteiner. Dette bringer da bruker videre til figur 14.



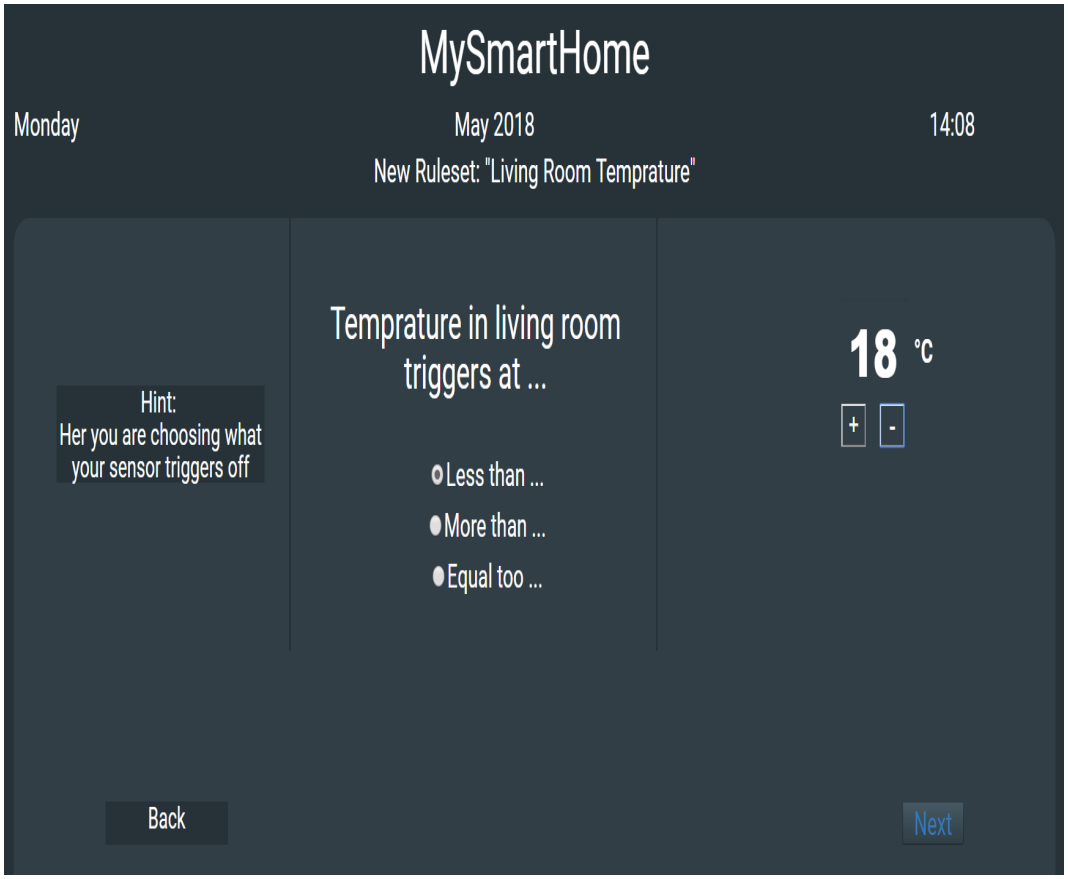
Figur 13: Sketch idé for å vise all tempraturdata

Figur 14 er hentet fra webgrensesnittet gruppen kodet som forslag til å representere én sensor sin data. Bilde er tatt av "Living Room" sin sensor, hvor bruker får vist dens temperatur under "Current Temprature" fra sensoren. Det er da også en graf som dynamisk representerer data historikk om hvordan temperaturen har forandret seg over tid. Denne bruker et bibliotek kalt vis.js, som brukes for å dynamisk representere data[44]. Videre har bruker mulighet til å velge ønske temperatur for lokasjonen. Dette gjøres under "New Temprature", hvor da å trykke "Set" setter ønsket temperatur for lokasjonen. Ellers vil bruker også ha muligheten til å lage et nytt regelverk for denne sensoren, ved å trykke på "Make new ruleset for: Living Room". Dette bringer bruker til figur 15.



Figur 14: Webgrensesnitt for Living Room Temprature sensor sin data

I figur 15 ser man starten på et nytt regelverk for en sensor. IFTTT metoden tatt i bruk for hvordan et regelverk lages. Brukeren får hint på venstre side over hvilke valg denne siden fører til. Én Trigger blir valgt for temperatur sensor gjennom å velge én av radio-knappene under “Temprature in Living Room Triggers At ...”, hvor da Triggeren sendes når temperaturen er enten “Less than ...”, “More Than ...” eller “Equal too ...” den valgte temperaturen fra bruken. Eksempel på dette vil være at bruker velger “Less than ...” og “18°C”, som da vil bety at en Trigger vil sendes når temperatur er “Less than 18°C”. Ved at man trykker “Next” fører det videre til figur 16.

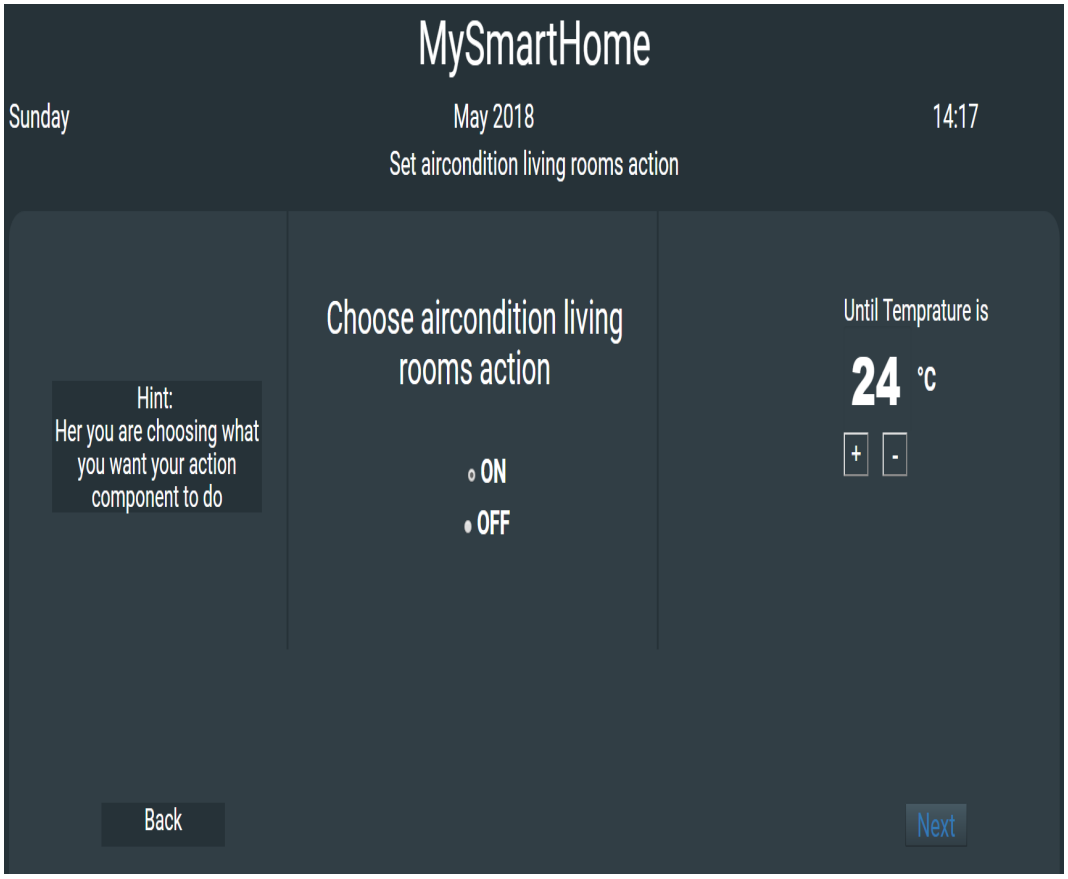


Figur 15: Webgrensesnitt for starten av et nytt regelverk for Living Room Temprature

Figur 16 tar for seg brukerens valg av Action komponent. Dette er da komponentett som skal utføre en handling eller "Action" av at Triggeren sendes. Eksempler på dette kan ses i figuren under "Choose what component takes action from trigger", hvor man får opp radioknapper som tar for seg komponenter knyttet mot temperatur, slik som aircondition, heat pump og fan. Eksempelvis når komponenten "Aircondition: Living Room" er valgt og trykker "Next", går det videre til figur 17.

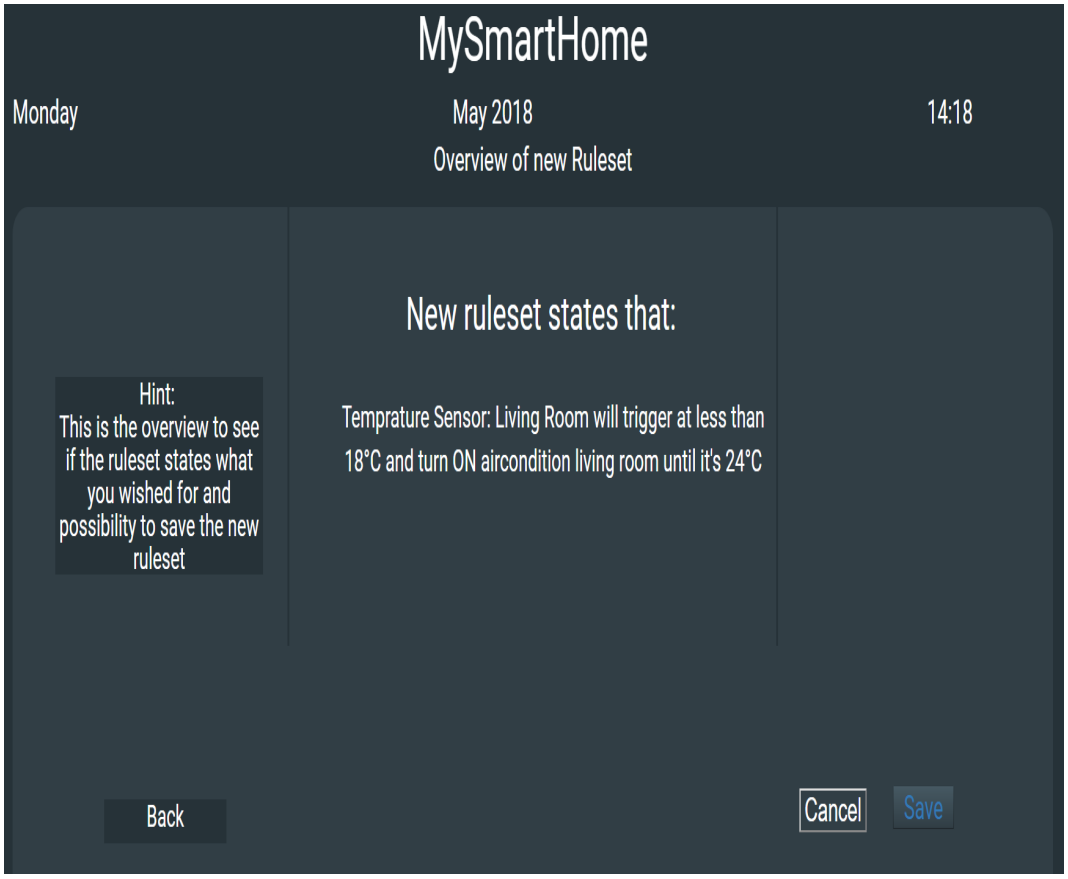
Figur 16: Webgrensesnitt for valg av en action komponent

I figur 17 settes hvilken Action som utføres av Action komponenten, når Trigger mottas. I figur 16 ble aircondition valgt til å være Action komponent. Dette medfører at valget "Choose aircondition living rooms action" dukker opp og motta valget "ON", til den har nådd temperatur valgt av bruker under "Until Temprature is ...". Bruker setter denne til å være 24°C, i dette eksempelet. Når bruker har valgt alt av informasjon sendes det videre til den siste siden, i figur 18.



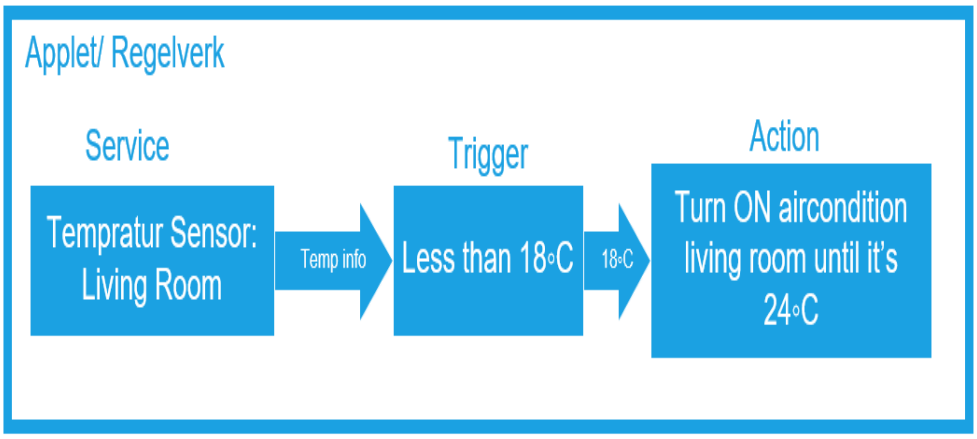
Figur 17: Webgrensesnitt for valg av en action kompoenent

I figur 18 får brukeren se en oversikt av regelverket under “New ruleset states that:". Dette blir dynamisk opprettet basert på tidligere valg. Ifølge IFTTT metoden er Service “Temperatur Sensor: Living Room” samlingen med informasjon om temperaturen. Denne informasjonen kan brukes til å lage Triggeren “Less than 18°C”. Som videre setter Actionen “Turn ON aircondition living room until it’s 24°C”. Samlingen av dette blir regelverk eller en Applet. Brukeren kan så velge å lagre regelverket eller kansellere.



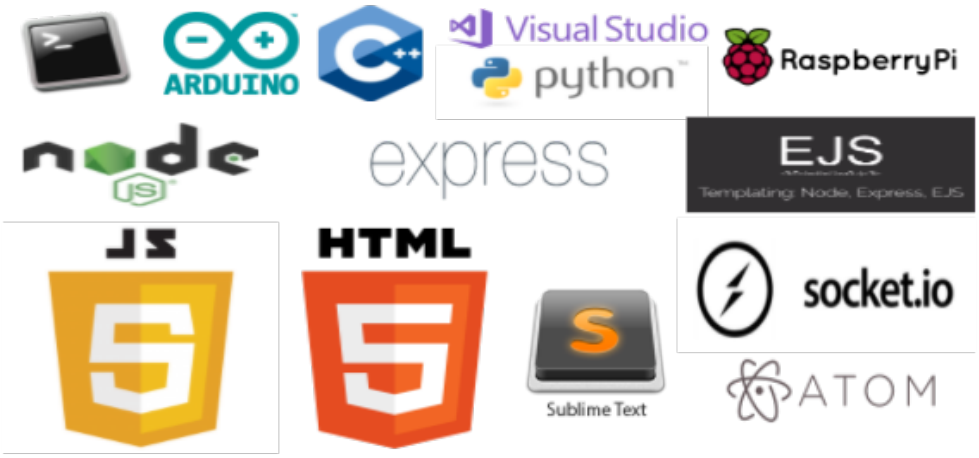
Figur 18: Webgrensesnitt for å se hvordan det fullførte regelverke ser ut

Under i figur 19 kan man se en reprenstasjon av hvordan regelverket fra figur 18 fyller ut IFTTT-metoden vist i figur 12.



Figur 19: Visualisering av IFTTT, med regelverk fra figur 12.

5 Koding, kvalitetskontroll, implementering, og testing



Figur 20: Programmeringsspråk og IDEer brukt

Fra tidlig i prosjektets utviklingsperiode skjønte gruppen at det ville bli en samling av forskjellige utviklingsverktøy og programmeringsspråk. For koding mot Arduino enheter er det brukt Arduino IDE og Visual Studio for utvikling av både sensorkode og biblioteker som er nødvendig for all oppkobling. Kodingen er gjort med programmeringsspråket C++.

Styringspunktet er satt opp med Linux, nærmere sagt Raspbian. Her har koding foregått i terminalen med programmeringsspråkene Python og JavaScript, med Node.Js og Express.js som rammeverk. Styringspunktet bruker også Socket.IO biblioteket.

Når det kommer til webserveren er Sublime Text tatt i bruk for koding av HTML, Embedded Javascript Templating (EJS), JavaScript og SQL, med bruk av Node.Js og Express.js som rammeverk for kontakt i sanntid mellom webserver og styringspunkt.

Databasene som er tatt i bruk er SQLite3 for lagring på styringspunktet. Dette er en lettversjon av MySQL, og fungerer som mellomlagring av data før det sendes videre for å unngå tap av data om styringspunkt og webserver mister kontakt. Denne databasen overskriver sitt gamleste innlegg om den skulle bli overfylt.

5.1 Koding og implementasjon

I dette kapittelet presenteres koden og implementasjonen for hele systemet. Koden kommuniserer gjennom hele prosessen, fra sensor og hele veien til nettsiden.

5.1.1 Implementering av Arduino sensorer

Arduino klasser

Nye sensorer som skal implementeres i produktet nedstammer fra baseklassen MySmartHome. Denne klassen setter opp tilkoblingspunktet for å lese inn variablene som er nødvendige for at sensoren skal kunne kommunisere med styringspunktet ved å publisere data. Sensor klassen håndterer innlesning av sin egen data og anvender baseklassen for resterende funksjonalitet.

Listing 5.1: MySmartHome Klasse

```
class MySmartHome {
        private:
                PubSubClient* _client;
                WiFiManager* _manager;
                char _sensor_id[str_length];
                char _mqtt_username[str_length];
                char _mqtt_password[str_length];
                char _mqtt_ssid[str_length];

        public:
                MySmartHome();
                MySmartHome(PubSubClient& client, WiFiManager& manager);
                void reconnect();
                void setup_connections();
                void mqtt_connection();
                void publish_sensor_json(char sensor_topic[], float payload);
                virtual ~MySmartHome();
};
```

Listing 5.2: Temperatur og fuktighets Klasse

```
class TempHumidSensor: public MySmartHome {
        private:
                int temp_celsius;
                int temp_fahrenheit;
                int humidity;
                char temp_topic[5];
                char hum_topic[5];
                DHT* dht;

        public:
                TempHumidSensor();
                TempHumidSensor(DHT& sensor, PubSubClient& client,
                                WiFiManager& manager);
                void setup_sensor_cons();
                void read_temp_humid();
                virtual ~TempHumidSensor();
};
```

Kodeeksempel setup_connections

Metoden setup_connections 5.3 anvender WiFiManager biblioteket for å opprette en DNS og webserver. Den legger til de ønskede parametrene: Sensor id(ønsket navn for bruker), MQTT Service Set Identifer (SSID)(tilkobling for hjemmenettverket), MQTT brukernavn og MQTT passord(Styringspunktets navn og passord). Til slutt setter sensoren opp server detaljene den skal publisere til.

Listing 5.3: Setup connections

```
WiFiManagerParameter custom_sensor_id("sensor id",
    "Sensor/Relay name(Ex: Bedroom/PorchLights)", _sensor_id, str_length);
WiFiManagerParameter custom_mqtt_ssid("ssid",
    "mqtt SSID/IP", _mqtt_ssid, str_length);
WiFiManagerParameter custom_mqtt_username("username",
    "mqtt username", _mqtt_username, str_length);
WiFiManagerParameter custom_mqtt_password("password",
    "mqtt password", _mqtt_password, str_length);

_manager->addParameter(&custom_sensor_id);
_manager->addParameter(&custom_mqtt_ssid);
_manager->addParameter(&custom_mqtt_username);
_manager->addParameter(&custom_mqtt_password);

_manager->autoConnect("SmartHome");

strcpy(_sensor_id, custom_sensor_id.getValue());
strcpy(_mqtt_ssid, custom_mqtt_ssid.getValue());
strcpy(_mqtt_username, custom_mqtt_username.getValue());
strcpy(_mqtt_password, custom_mqtt_password.getValue());

_client->setServer(_mqtt_ssid, mqtt_port);
```

Kodeeksempel mqtt_connection()

Sensoren sjekker om den er tilkoblet til en server. Så lenge den ikke har en tilkobling vil den prøve å koble seg til styringspunktet hvert femte sekund. Dersom en tilkobling er etablert kan sensoren begynne å overføre eller publisere pakker.

Listing 5.4: mqtt_connection

```
if(!_client->connected()){
    while (!_client->connected()) {
        Serial.print("Attempting MQTT connection..:");
        if (_client->connect("MySmartHome",_mqtt_username, _mqtt_password))
            Serial.println("Established connection");
        else {
            Serial.println("Retrying connection");
            delay(5000);
        }
    }
}
_client->loop();
```

Kodeeksempel read_temp_humid()

Avlesning av sensor data for den nedstammede klassen TempHumidSensor, sjekker om sensoren får etablert en kommunikasjonsvei til styringspunktet. Leser av temperaturen og fuktigheten i miljøet, jekker om sensoren fikk avlest riktig informasjon, og publiserer denne dataen til et av temaene styringspunktet abonnerer på.

Listing 5.5: Avlesning temperatur og fuktighet

```
mqtt_connection();

humidity = dht->readHumidity();
temp_celsius = dht->readTemperature();
temp_fahrenheit = dht->readTemperature(true);

if(isnan(humidity) || isnan(temp_celsius) || isnan(temp_fahrenheit)){
        Serial.println("Failed␣DHT");
        return;
} else {
        MySmartHome::publish_sensor_json(temp_topic, temp_celsius);
        MySmartHome::publish_sensor_json(hum_topic, humidity);
}
delay(5000);
```

Kodeeksempel publish_json()

Oppretter et JSON objekt og laster inn det som skal publiseres til styringspunktet.

Listing 5.6: Publisering av sensor informasjon

```
StaticJsonBuffer<300> JSONbuffer;
JsonObject& JSONencoder = JSONbuffer.createObject();

JSONencoder["Sensor_ID"] = _sensor_id;
JSONencoder[sensor_topic] = payload;

char JSONmessageBuffer[100];
JSONencoder.printTo(JSONmessageBuffer, sizeof(JSONmessageBuffer));
Serial.println("Sending␣message␣to␣MQTT␣topic..");
Serial.println(JSONmessageBuffer);

if (_client->publish(sensor_topic, JSONmessageBuffer) == true) {
        Serial.println("Success␣sending␣message");
}
else {
        Serial.println("Error␣sending␣message");
}
```

Arduino sketch

Arduino anvender ordet sketch til å beskrive et program. En sketch i Arduino må inneholde blokkene setup() og loop() før den kan lastes opp å kjøre. Setup blokken kjører bare en gang ved oppstart etter kodekompilering, og initialiserer alt som blir nødvendig for senere bruk i programmet. Etter setup kjører Arduino loop blokken så lenge den har strøm og vil dermed kontinuerlig lese av miljøet og publisere resulterende data til styringspunktet.

Listing 5.7: Arduino Sketch

```
TempHumidSensor sensor(dht, client, manager);

void setup() {
    Serial.begin(115200);
    while(!Serial) {;}
    sensor.setup_sensor_cons();
}

void loop() {
  sensor.read_temp_humid();
}
```

5.1.2 Implementering av Styringspunkt

Kodeeksempler fra initialize_DB_Tables.py

Figur 5.8 viser hvordan et pythonscript på styringspunktet under initiell oppstart oppretter en SQLite database med et gitt navn, og et gitt oppsett av databasestruktur. Den lukker også databasen etter bruk for sikkerhetsmessige grunner.

Listing 5.8: Kodeeksempel SQLite database for styringspunkt

```
#Connect or Create DB File
    conn = sqlite3.connect(DB_Name)
    curs = conn.cursor()

#Create Tables
    sqlite3.complete_statement(TableSchema)
    curs.executescript(TableSchema)

#Close DB
    curs.close()
    conn.close()
```

Figur 5.9 viser et utklipp av hvordan to sensorer vil bli satt opp i databasestrukturen. Det blir ikke opprettet en ny database med mindre det ikke finnes en fra før.

Listing 5.9: Kodeeksempel SQLite databasestruktur

```
 # SQLite DB Name
DB_Name =  "IoT.db"

# SQLite DB Table Schema
TableSchema="""
drop table if exists DHT22_Temperature_Data ;
create table DHT22_Temperature_Data (
  id integer primary key autoincrement,
  SensorID text,
  Date_n_Time text,
  Temperature text
);
drop table if exists DHT22_Humidity_Data ;
create table DHT22_Humidity_Data (
  id integer primary key autoincrement,
  SensorID text,
  Date_n_Time text,
  Humidity text
);
"""
```

Kodeeksempler fra mqtt_Listen_Sensor_Data.py

Figuren 5.10 demonstrerer hvordan tilkoblingen mot MQTT serveren på styringspunktet blir opprettet. Her er det nødvendig å sette et brukernavn og passord til MQTT serveren, noe som blir gjort av leverandøren av produktet før det blir sendt til bruker. Det blir satt en statisk IP-adresse foreløpig, men dette vil endre seg med bruk av domene. I utklippet kan man se variabel **mqtt_Topic =** "#" som betyr at lytteren skal lytte etter alle temaene sendt av sensorene.

Listing 5.10: Kodeeksempel for oppkobling av lytter på styringspunkt

```
#!/usr/bin/python
import json
import paho.mqtt.client as mqtt
from store_Sensor_Data_to_DB import sensor_Data_Handler

MQTT_username = "mittsmarthjem"
MQTT_password = "mittsmarthjem"
MQTT_Broker = "10.0.0.1"
MQTT_Port = 1883
Keep_Alive_Interval = 45
MQTT_Topic = "#"
```

Figur 5.11 viser hvordan lytteren kobler seg opp ved hjelp av de fastsatte variablene og mottar data i JSON format før det sendes til funksjonen **sensor_Data_Handler()** i scriptet **store_Sensor_Data_to_DB.py**. Lytteren vil fortsette å loope så lenge styringspunktet er aktivt.

Listing 5.11: Kodeeksempel for bruk av lytter på styringspunkt

```
def on_connect(mosq, obj, flags, rc):
        mqttc.subscribe(MQTT_Topic, 0)

def on_message(mosq, obj, msg):
        print("MQTT Data Received...")
        print("MQTT Topic: " + msg.topic)
        print( "Data: " + msg.payload)
        sensor_Data_Handler(msg.topic, msg.payload)

def on_subscribe(mosq, obj, mid, granted_qos):
    pass

mqttc = mqtt.Client()
mqttc.username_pw_set(MQTT_username, MQTT_password)

mqttc.on_message = on_message
mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe

mqttc.connect(MQTT_Broker, int(MQTT_Port), int(Keep_Alive_Interval))

mqttc.loop_forever()
```

Kodeeksempler fra store_Sensor_Data_to_DB.py

Som vist i figur 5.11 blir data i JSON format sendt til funksjonen **sensor_Data_Handler()** hvor det blir utført en sjekk for hvilket tema som blir sendt. Hvis temaet er **temp**" blir JSON dataen sendt til **DHT22_Temp_Data_Hander(jsonData)**.

Listing 5.12: Kodeeksempel på data handler på styringspunkt for SQLite

```
def sensor_Data_Handler(Topic, jsonData):
        if Topic == "temp":
                DHT22_Temp_Data_Handler(jsonData)
        elif Topic == "hum":
                DHT22_Humidity_Data_Handler(jsonData)
        elif Topic == "light":
                Light_Data_Handler(jsonData)
        elif Topic == "touch":
                Touch_Data_Handler(jsonData)
        elif Topic == "pir":
                Pir_Data_Handler(jsonData)
```

Figur 5.13 ser man hvordan dataen blir gjort om fra JSON og lagret i databasen **IoT.db**. Her legges det også til en dato og et tidspunkt for når dataen blir lagt inn i database for loggføring, oversikt og sikkerhetsmessige grunner.

Listing 5.13: Kodeeksempel på data handler på styringspunkt for SQLite

```
def DHT22_Temp_Data_Handler(jsonData):
        json_Dict = json.loads(jsonData)
        SensorID = json_Dict["Sensor_ID"]
        Data_and_Time =  time.strftime("%d/%m/%y␣%H:%M:%S")
        Temperature = json_Dict['Temperature']

        dbObj = DatabaseManager()
        dbObj.add_del_update_db_record
        ("insert␣into␣DHT22_Temperature_Data(" +
        "SensorID,␣Date_n_Time,␣Temperature)␣values␣(?,?,?)",
        [SensorID, Data_and_Time, Temperature])
        del dbObj
        print("Inserted␣Temperature␣Data␣into␣Database.")
        print("")
```

Kodeeksempler fra client.js

I dette prosjektet er klienten satt opp gjennom styringspunktet. Gruppen valgte å bruke Node.Js og Express.js rammeverkene med Socket.IO biblioteket for kommunikasjon mellom klient og server. Dette valget ble gjort ettersom prosjektet krever hurtiggående oppdateringer fra både sensoren og fra styringspunktet over til serveren. I figur 5.14 er det brukt en statisk IP-adresse for kobling mot serveren, men dette vil endre seg når Fosen Utvikling overtar prosjektet å benytter sine servere.

Listing 5.14: Initialisering av socket for klient

```
//client.js
var io = require('socket.io-client');
var socket = io.connect("http://192.168.1.142:8080",
{'reconnection': true, +
'reconnectionDelay': 1000, 'reconnectionAttempts': 99999});
// Change to remote host when not on local comp
const sqlite3 = require("sqlite3").verbose();
const fs = require("fs");
```

Når man bruker Socket.IO for tilkobling fungerer dette med bruk av triggere. Her sender klienten en forespørsel til kanalen "connect" med et callback for socketen som blir funnet. Om det blir funnet en kanal på server siden for "connect" vil styringspunktet få kontakt

Listing 5.15: Tilkobling av socket gjennom Node.Js rammeverket

```
// Add a connect listener
socket.once('connect', function (socket) {
  console.log('Connected!');
});
```

Socket.IO fungerer slik at det trengs triggere for å holde en kanal åpen for kommunikasjon. Dette er gjort ved at ved tilkobling sender serveren en forespørsel på en gitt kanal om å starte en loop så lenge den får svar. Her opprettes en kanal for temperatur data med en forsinkelse på tre sekunder for å ikke sende data fortere enn database tilkallingen klarer å håndtere.

Listing 5.16: Oppretting av en kanal gjennom socket og Node.Js rammeverket

```
socket.on('CH01', function(status){
    console.log('Status: ', status);
     setTimeout(function() {
      openTempData();
     }, 3000);
});
```

Når kommunikasjon er opprettet blir funksjonen **openTempData()** kjørt og JSON objektet blir laget. Her legges det til nøkkelord for enklere bruk på serverside etter at pakken har blitt sendt. Det blir også lagt inn en ID for dette styringspunktet slik at brukeren kan enkelt ved bruk av dette koble sin profil til denne IDen. Gjennom et script som gjennomfører all installasjon av styringspunktet automatisk, blir det opprettet et filsystem og egne filer som trengs for at styringspunktet skal konfigureres. Her blir det leverandør sitt ansvar å legge inn ID, og markere dette bak styringpunktet før levering.

Listing 5.17: Sending av temperatur data gjennom socket mot webserver

```
function openTempData() {
let db = new sqlite3.Database("/home/pi/PythonBrokerTest/IoT.db");

let sql = 'SELECT * FROM DHT22_Temperature_Data WHERE ID
= (Select MAX(ID) FROM DHT22_Temperature_Data)';

  db.all(sql, [], (err, rows) => {
   if (err) {
    throw err;
  }
  rows.forEach((row) => {
  fs.readFile("/home/pi/MySmartHome/RaspberryID", (err, data) => {
        if (err) throw err;
         var fileContent = data.toString("utf8");
         var raspid = fileContent.trim();
         var topic = "temp";
         var id = row.SensorID;
         var date = row.Date_n_Time;
         var data = row.Temperature;
         var msg = { "RaspberryID":raspid, "Topic":topic, "SensorID":id,
         "Date":date, "Data":data };

    socket.emit("CH01", "Rasperry Pi:", msg);
     });
  });
});
db.close();
}
```

5.1.3 Implementering av Webserver

Kodeeksempler fra socket_api.js

Figur 5.18 viser til serversiden sitt svar på ”connect” forespørsel fra klienten. Her sendes det svar om godkjent kobling, samt at det åpnes spesifikke kanaler for lytting gjennom triggere.

Listing 5.18: Opprett kontakt mellom klient og server på Webserver

```
io.on('connect', function (socket){
        var sessionid = socket.id;
        console.log(sessionid);
   console.log('connection');

   socket.emit("CH01", "Connecting to channel 1: Temperature");
   socket.emit("CH02", "Connecting to channel 2: Humidity");
```

Etter at koblingen har blitt godkjent og triggerene er blitt sendt kan serveren motta data fra klienten i form av JSON. Dette blir videresendt til webserveren sin SQL database.

Listing 5.19: Åpning av kanal for sending av spesifikk data

```
socket.on('CH01', function (from, msg) {
    console.log('MSG', from, msg);
    db.sensor_query(msg.SensorID.toString(), msg.Data.toString(),
    msg.RaspberryID.toString(), msg.Topic.toString());
    socket.emit('CH01', msg.Topic);
  });
```

5.2 Kvalitetskontroll og testing

Det ble tidlig satt et krav i gruppen at det skulle være et fokus på kodekvalitetssikring under utviklingen. Da utviklingen foregikk på flere forskjellige platformer og operativ systemer var det nødvendig å bruke litt tid på å finne riktig verktøy som kunne hjelpe med kodekvalitet. Det viste seg å være utfordrende å finne et verktøy som kunne se igjennom og sjekke kvaliteten på koden som ble laget for Arduino brettene. Dette er på grunn av lite støtte for kodekvalitets verktøy i Arduino IDE, som for eksempel et Linter verktøy eller andre statiske kodeanalyse verktøy. Det var derimot enklere å finne verktøy som kunne hjelpe oss med kodekvaliteten på Raspberry Pi, Webserver og nettsiden, hvor SonarQube ble tatt i bruk.

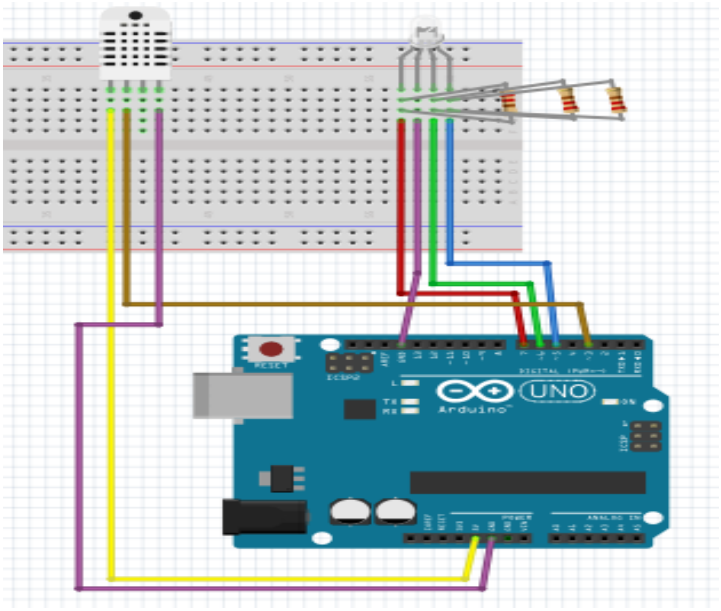
5.2.1 Code Review

En av måtene gruppen kunne sikre god kodekvalitet på var å markere vanskelige deler av utviklingen og gå igjennom dem i felleskap, såkalte code reviews. Det ble i SCRUM prosessen satt av tid til code review på tirsdager. I starten gikk gruppe gjennom koden selv, men etterhvert ble det mer naturlig at også oppdragsgiver ble med på denne prosessen.

Da oppdragsgiver selv jobber som utvikler og har en del erfaring, ble det veldig naturlig at han også bisto der gruppen følte de trengte hjelp med enkelte oppgaver i utviklingen. Det ble da også diskusjon om mulige løsninger, og problemer ble løst på en god måte.

5.2.2 Workshop

Da mye av teknologien som ble tatt i bruk var ny for de fleste i gruppen ble det i begynnelsen av utviklingen satt av tid til å lære seg å koble sammen Arduino brett og sensorer og utvikle i Arduino IDE. Alle i gruppen fikk tildelt eget utstyr(Se vedlegg P Utstyrsliste) som måtte settes opp, det skulle da kobles sammen riktig, legge inn riktig kode så sensorene sendte data til PCen gjennom COM porter. Så skulle det lages en schematic som de andre kunne følge for å sette opp samme sensor oppsettet. Dette gjorde at alle fikk erfaring med å sette opp en sensor og å programmere dem riktig. Gruppen fikk også litt erfaring med å lage gode schematics som andre kunne følge, uten mye forkunnskap.



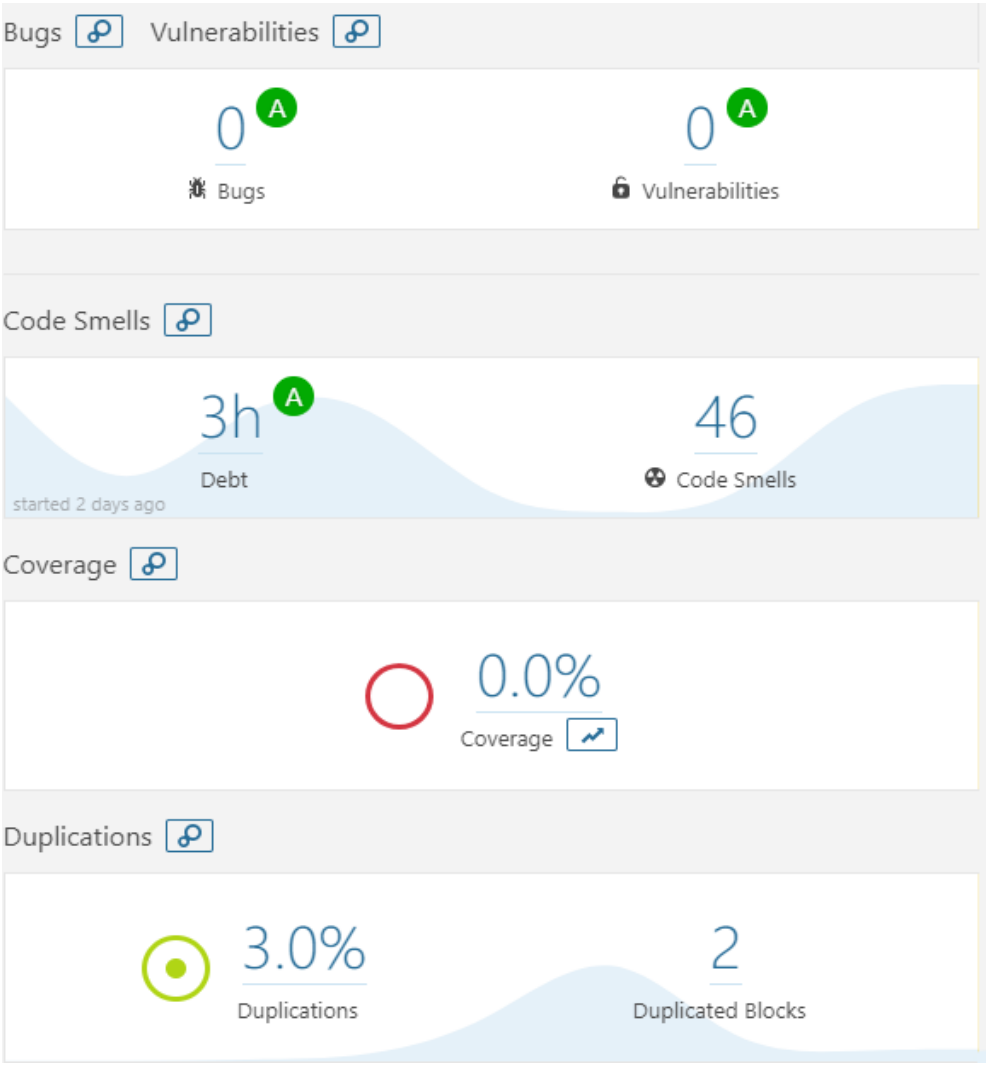
Figur 21: Eksempel på en Arduino Schematic

5.2.3 Verktøy

SonarQube

SonarQube dekker de språkene som blir brukt i prosjektet, derav JavaScript, C++ og Python. C++ blir kun brukt for Ardruino sensorene og reléene, og dekkes derfor ikke godt av SonarQube.

SonarQube går igjennom koden og ser etter bugs, code smells og duplisering[28]. Bugs er i SonarQube definert som feil som potensielt kan få applikasjonen til å kræsje, og som må fikses så fort som mulig. Code smells er definert som mulige vedlikeholds problemer som kan oppstå, dette kan være i form av at variabel navn ikke følger en standard, til at det er variabler eller funksjoner som ikke blir brukt. Disse er ikke kritiske, men kan gjøre det vanskeligere for andre å ta over prosjektet, eller drive med vedlikeholds arbeid. Duplikasjoner i koden sier seg nesten selv, det er når samme koden blir skrevet likt flere steder, i stedet for eksempel å ta i bruk en felles funksjon.



Figur 22: SonarQube scanner rapport.

6 Realisering og Installasjon

I dette kapittelet vil produktets installasjonprosess gjennomgåes med oppdeling på hva som blir leverandøren sitt ansvar og hva brukeren må gjøre for å få et operasjonelt produkt. Selv om det er et komplisert produkt er det fokusert på brukervennlighet både for oppsette gjort av leverandøren og brukeren. MySmartHome er et Open Source prosjekt og vil ikke være klart for salg før utviklingen har kommet lengre. Gruppens oppgave var å lage grunnmuren av et videreutviklings prosjekt med hovedfokus på kommunikasjonen mellom sensor og styringspunkt.

6.1 Leverandør

Når det forekommer en bestilling av produktet vil det første ansvaret falle på leverandøren, i dette tilfellet Fosen Utvikling. Leverandøren installerer Raspbian på en Raspberry Pi, deretter hentes ned AutoInstall scriptet fra MySmartHome sin Open Source bitbucket og kjøres. Dette scriptet vil automatisk opprette et filsystem som vil bli brukt av resten av systemet. Deretter installerer scriptet alle andre nødvendige script og oppdateringer som trengs for at produktet skal være operasjonelt. Pakken som blir lastet ned inneholder blant annet scriptet for lytting mot sensor for å hente data til styringspunktet, samt scriptet for installasjon og lagring mot en SQLite database.

```
 GNU nano 2.7.4                          Fil: MySmartHome.sh

#/!bin/bash
#Script to install all needed components for the MySmartHome application, this
#will include the needed downloads for the other scripts to run and the
#scripts themselves, and make them autorun.
```

Figur 23: AutoInstall script, MySmartHome.sh

Når AutoInstall blir kjørt som vist i figur 23, vil leverandøren få utskrifter av hva han må gjøre underveis for å få systemet opp. Scriptet vil automatisk installere og konfigurere MQTT server, samt VNC Viewer for at leverandør kan gjennomføre kundeservice om nødvendig. Herunder faller også utfylling av filen RaspberryID, som er en viktig del av oppkobling av styringspunkt og dens egne sensorer slik at Fosen Utvikling vil ha oversikt over hva som hører til hvor på sin administrator nettside. Personen som gjør klart styringspunktet til levering må inn i RaspberryID filen og skrive en unik ID for akkurat denne brukeren.

```
pi@raspberrypi:~/MySmartHome $ ls
RaspberryID
```

Figur 24: RaspberryId fil som opprettes av AutoInstall

Når AutoInstall scriptet er kjørt og en unik ID har blitt gitt til styringspunktet er det leverandøren sitt ansvar å merke styringspunktet med den informasjonen som brukeren trenger til videre installasjon. Dette gjøres slik som på et modem, hvor informasjonen blir medsendt bak på styringspunktet som vist i prototypemodell i figur 25. Gruppen har hatt stort fokus på automatisering av installasjon for å gjøre det lettest mulig for både leverandør og bruker.

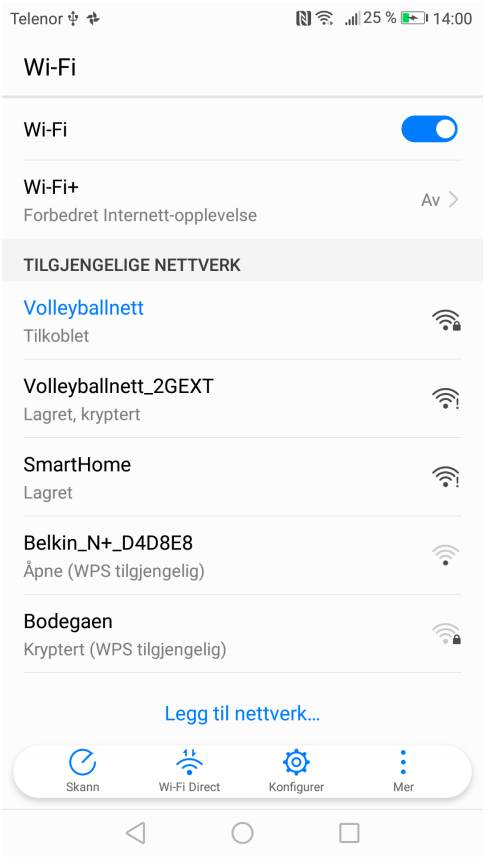
Figur 25: Prototype av styringspunkt

Etter at styringspunktet er ferdigkonfigurert må leverandøren laste opp koden til sensoren. Dette gjøres ved å koble sensoren til en datamaskin som deretter sender den ferdigskrevne koden til sensoren. Med dette er produktet klart til utsending og resterende installasjon vil bli gjort av brukeren.

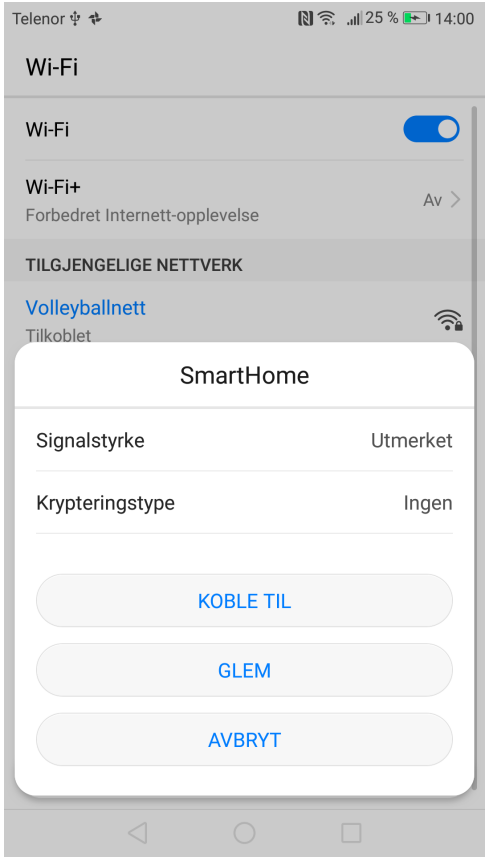
6.2 Bruker

For at styringspunktet skal være klart til bruk trenger brukeren kun å koble det til et strømuttak. Alt av konfigurering ble gjort hos leverandør, noe som gjør dette produktet enklere for en bruker å installere. For at sensoren skal kobles opp trenger brukeren enten å koble den til et strømuttak eller bruke batteri. Dette er et valg brukeren kan ta under bestilling av produktet. Etter at sensoren er koblet til strøm vil den opprette et egen nettverk som brukeren kobler seg til for å konfigurere som vist i figur 26a. Her ser man muligheten til å koble til MySmartHome nettverket som er opprettet av sensoren.

Figur 26b viser når brukeren trykker på MySmartHome nettverket og får en enkel forespørsel om han ønsker å koble seg til for å konfigurere sensorens WiFi tilkobling.

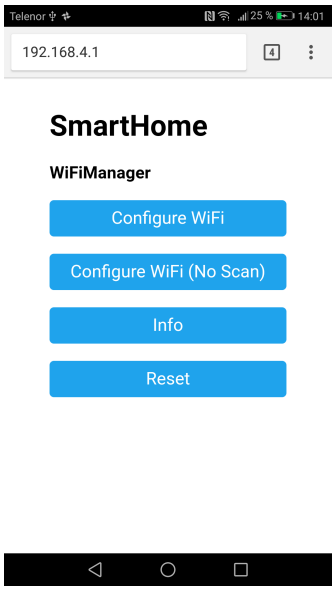


(a) 26 Bruk av WiFi Manager

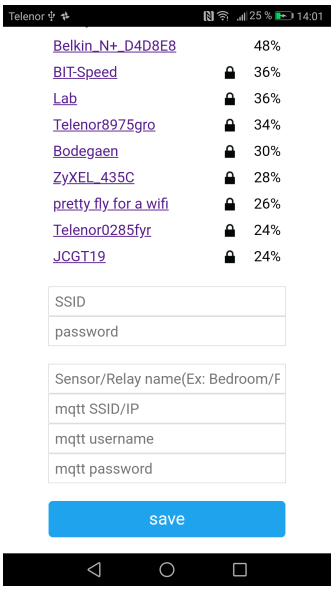


(b) 26 Bruk av WiFi Manager

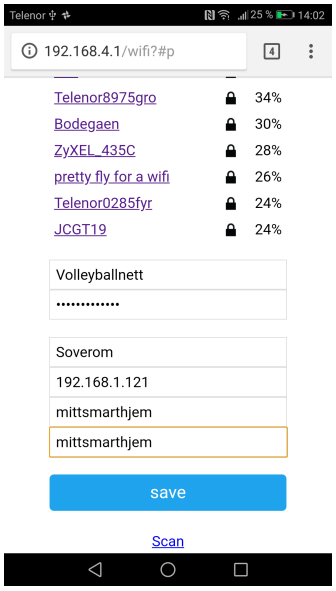
Etter at oppkobling til sensorens nettverk er gjennomført kan brukene velge å konfigurere sensoren. Brukeren får en oversikt over tilgjengelige nettverk som sensoren kan kobles til. Her er det viktig at brukeren velger det samme nettverket som styringspunktet er tilkoblet for at disse komponentene skal kunne kommunisere. Brukeren velger ønsket nettverk og skriver inn nettverkets passord. Deretter skrives informasjonen som er medsendt på styringspunktet, som vist i figur 25, inn i de forskjellige feltene i WiFi Manageren som vist i figur 27b og figur 28a.



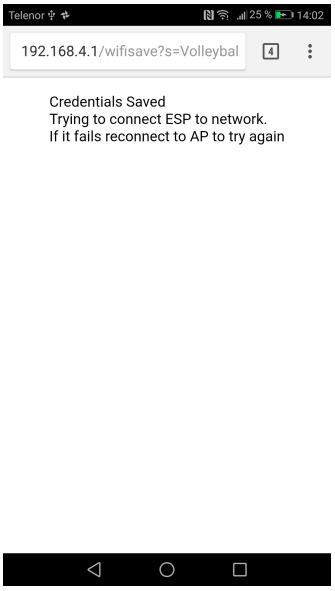
(a) 27 Bruk av WiFi Manager



(b) 27 Bruk av WiFi Manager

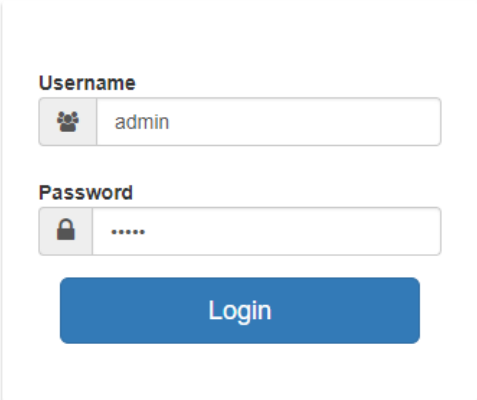


(a) 28 Bruk av WiFi Manager

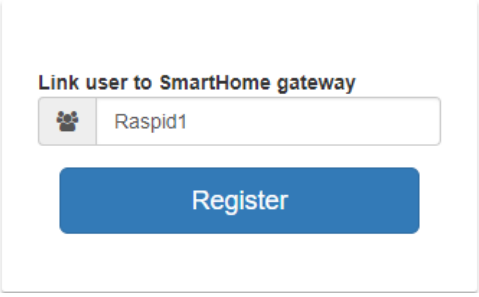


(b) 28 Bruk av WiFi Manager

Når brukeren har gjennomført konfigurering av WiFi Manager vil sensoren automatisk begynne å sende data. Dette må gjøres for hver nye sensor som skal tilkobles, og brukeren velger selv navnet som sensoren skal bruke, for eksempel Soverom, Kontor eller Kjøkken. Gruppen har tatt hensyn til utbygging med flere sensorer. Systemet håndterer like mange sensorer samtidig for en husstand som IP-adresser tilgjengelig. Videre vil brukeren ha tilgang til sin egen data ved å gå til MySmartHome sin nettside og registrere eller logge på med en bruker, og registrere den medsendte ID vist i figur 25, som brukeren finner som vist i figur 25, slik at brukerens egne sensorer blir koblet sammen med det riktige styringspunktet. Dette er demonstrert i figur 29a og figur 29b.

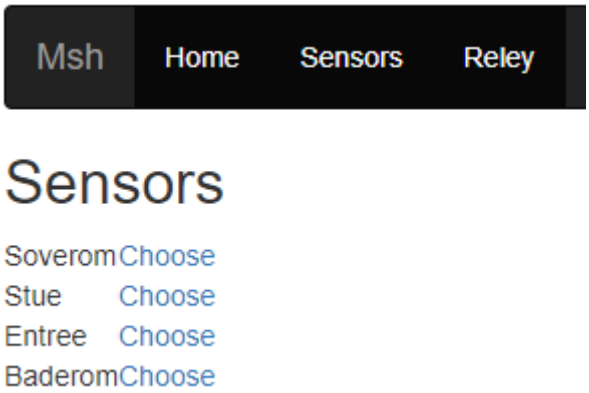


(a) 29 Login



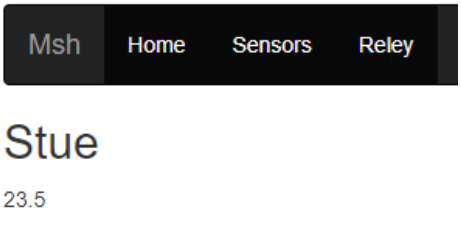
(b) 29 Registrere medsendt ID

Når brukeren har logget på og registrert ID, slik at hans bruker er koblet sammen med styringspunktet og sensorene, kan brukeren se data som blir produsert av sensoren. Brukeren får oversikt over alle tilkoblede sensorer, samt muligheten til å gå inn på en enkelt sensor for å se utskrift av registrert data.

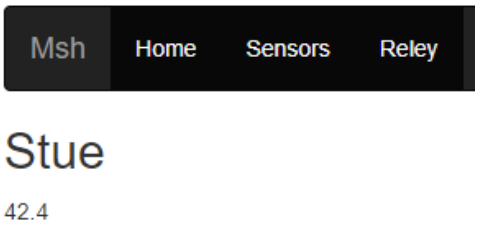


Figur 30: Oversikt over tilgjengelige sensorer

I figurene under er en demonstrasjon av utskrevet data fra sensorene. I figur 31a er nåværende temperatur og i figur 31b er nåværende fuktighet fra en sensor som er plassert i stuen.



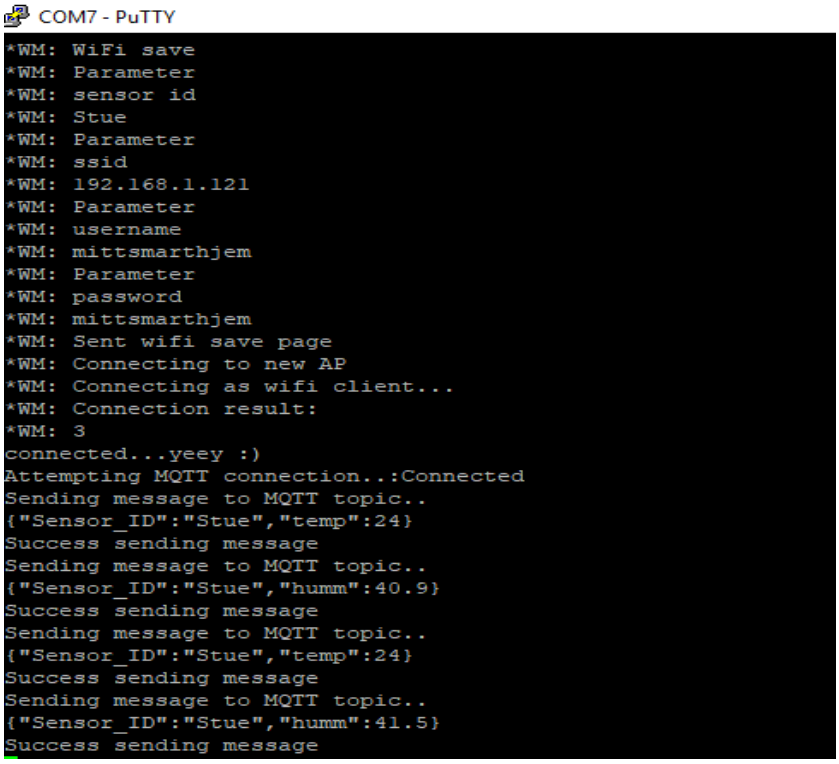
(a) 31 Utskrift av temperatur



(b) 31 Utskrift av fuktighet

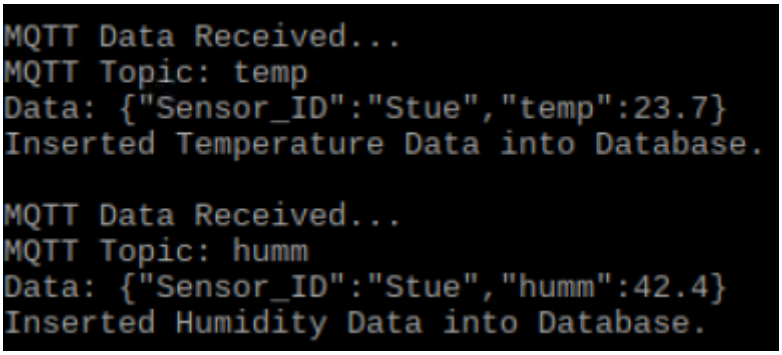
6.3 Demonstrasjon av backend

Med ønsket fra Fosen Utvikling om at det skal være kommunikasjon mellom sensor, styringspunkt og en webserver har gruppen lyktes. Produktet har fokus på brukervennlighet når det gjelder oppkobling og automatisering. Som vist tidligere i dette kapittelet er det ikke mange steg en bruker må gjennomføre før produktet sender data. Denne datan sendes fra sensoren først, som vist i figuren 32. Her blir fuktighets data sendt videre over WiFi til styringspunktet.



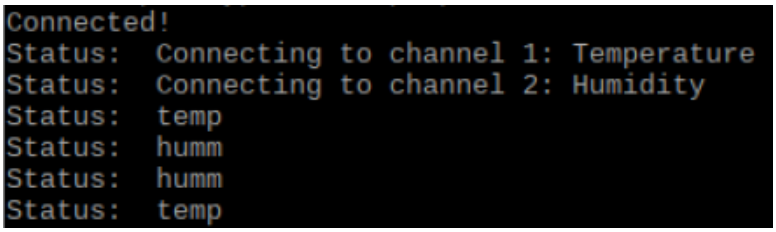
Figur 32: Utskrift av fuktighet

Videre er det demonstrert at styringspunktet henter ned data sendt fra sensor ved hjelp av en lytter som alltid lytter til tema som blir satt på sensor. Dette følger publish/subscribe mønsteret som nevnt i kapittel 4 og fungerer sammen med MQTT serveren. Videre blir temperatur- og fuktighets data lagret i en SQLite database for loggføring og en fail-safe om det skulle oppstå nettverks problemer eller strømbrudd.

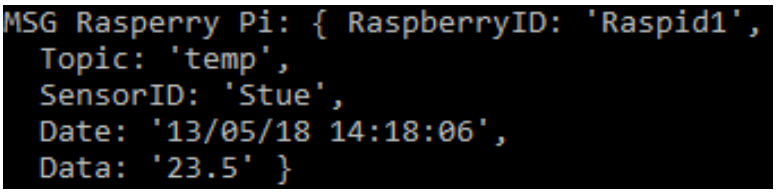


Figur 33: Styringspunktet mottar data og lagrer i SQLite

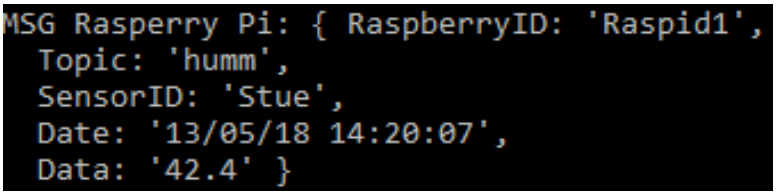
Videre kobler styringspunkt til serveren, gjennom Socket.IO, som mottar alt som et JSON objekt som blir lagret i en SQL database og vist til bruker i sanntid på MySmartHome nettsiden som demonstrert i figurene 31a og 31b.



(a) 34 Styringspunkt kobler seg til server



(b) 34 Server mottar temperatur



(c) 34 Server mottar fuktighet

7 Avslutning og konklusjon

7.1 Diskusjon

Gruppen begynte prosjektfasen sammen med Fosen Utvikling i et lynkurs for Arduino. Dette lynkurset gikk ut på å lære basis kunnskapene som trengtes for å sette opp en enkel sensor uten noen ekstrakomponenter, ved bruk av Arduino Uno prototypeenhet. Denne returnerer kun dataen gjennom Arduino IDE terminale ved bruk av USB til USB 2.0 B. Dette medførte stort ansvar for gruppen med å lære seg alt nødvendig av informasjon om Arduino, og en frihet til å velge hvordan utviklingen skulle gjennomføres. Fosen Utvikling satte kun ett ønske på levert produkt, og dette var bruken av WiFi for kommunikasjon mellom sensor og stryingspunkt. Om dette ikke var mulig å utføre, skulle gruppen selv finne en annen løsning i forhold til kommunikasjon mellom enhetene. Ettersom gruppen ikke hadde tidligere kunnskaper innen bruk av Arduino og hardware programmering, innebar første del av prosjektet en workshop(Se 5.2.2) med prototyping, læring og testing. Dette resulterte i en iterativ prosess for å kombinere nye komponenter mot å oppnå ønsket funksjonalitet.

Det å starte med maskinvareutvikling og prototyping fra bunnen av viste seg å være en stor oppgave å ta på seg. Det skal spesifiseres at gruppen har ingen tidligere erfaring eller kompetanse innen elektro ingeniøring og ord som kretskort, resistanse og spenning var kun kjent i fysikk fra videregående skole. Arduino utviklingen var en iterativ prosess hvor gruppen tilførte komponenter til enheten ettersom en prototype med bestem funksjonalitet fungerte. Fra Arduino Uno brett med sensor komponent og NodeMCU med WiFi muligheter, til anvendelse av kun de nødvendige komponentene som utførte sensorens funksjonalitet. Hver iterasjon av enhetene tilførte egne problemer som bidro til frustrasjon og utsettelser. Feilsøking av Arduino tilsvarer masse forum leting og man kan enkelt rote seg ned i kaninhullet. Problemer som kunne ligne på feilkoblinger mellom komponentene avslørte seg å være knekte pinner på kablene eller ødelagte komponenter. Et av de største hodebryene var når alle komponentene til sluttproduktet av sensoren var sammensveiset. Koden kunne lastes opp, ingen feilmeldinger var tilstede, men enheten tillot ikke å kjøre koden. Det viste seg å være behov for mer strømforsyning til brettet og komponentene.

Gruppen hadde stor probematikk ved opplasting av sketcher til sensor enheten. Problemer med data opplastingshastigheten eller baud rates. ESP8266 WiFi modulens dokumentasjon hevder at modulen har en standard baud rate på 115200 symboler overført per sekund[45], men FTDI komponenten som konverter koden til ESP komponenten derimot fungerer ikke optimalt på en slik opplastingsrate(Se 4.2 Design). Opplastingsrater ble da gjennomført gjennom en prøv og feil metodikk, og debuggingen bestod av USB monitorering ved hjelp av konsoll logg som ville enten vise suksess, ingenting eller uforståelig symboler dersom hastigheten ikke var optimal. En gang i skuddåret under en fullmåne vil planetene stilles på rekke og sensoren kunne kompilere og laste opp koden.

Planen var å inkludere ulike sensorer i produktet, blant annet bevegelse, lys og berørelse. Relé utbyggelsen beviste seg å være vanskelig. Dette er på grunn av at hvis relé enheten skulle bli integrert i en gjenstand er det nødvendig å tilkoble den direkte til dens strømkrets. Slik elektronisk kapabilitet er utenfor gruppens kunnskaps spekter, og ikke minsst uforsvarlig. Ettersom det oppstod så mye problematikk rundt Arduino utvikling ble det tatt opp til diskusjon med oppdragsgiver, og han var enig at produktets funksjonalitet kunne fremstilles ved bruk av kun én sensor, og reléet kunne forenkles med en demonstrasjon av et led lys med en av og på funksjon.

Et av diskusjons temaene som oppstod underveis var hvordan en sensor kunne bindes til et styringspunkt med tanke på brukervennlighet. Hvordan skulle sensoren få tak i tilkoblings detaljene for nettverket og publiserings protokollen? Mulighetene var mellom Bluetooth, WiFi eller USB kabel til egen PC. Sluttproduktet endte opp med at brukeren skal skrive inn disse detaljene gjennom et tilkoblingspunkt sensoren oppretter gjennom WiFi. Dette var på grunn av at både USB kabel og Bluetooth ville behøvd en ekstra komponent for å håndtere inputene, og siden et av kravene fra oppdragsgiver var at sensorene skulle kommunisere med styringspunktet gjennom WiFi uansett, falt også valget om initiering av nettverksdetaljer på WiFi.

Som tatt opp i seksjon 2.3, valgte gruppen å ta for seg utviklingsmetodikken SCRUM. Valget av denne metodikken ble gjort på bakgrunn av oppgavens karakteristika (Se 2.3.1 Valg av metodikk), slik som: vanskeligheter rundt estimering av produktets størrelse, muligheter for videreutvikling, få rigide rammer, avgrensninger for hvordan løsing for produkt skulle oppnås og dynamisk tilpasse seg hendelser rundt utvikling. Produkteier ønsket også å være en aktiv del av utviklingen.

Andre metodikker som ble sett på av gruppen var Fossefall. Denne ble valgt bort på bakgrunn av dens krav til at oppgaven må være ferdigstilt med en kravspesifikasjon fra produkteier, samt dets rigide krav til sekvensielt arbeid gjennom utviklingsprosessen. Dette ville ikke passe oppgavens karakteristika. Gruppen tok opp i seksjon 2.3.1 Valg av metodikk, under “Argumentasjon”, valgene rundt hvert av punktene satt av oppgavens karakteristika. Den åpne oppgave tolkningen, gitt av Fosen Utvikling, ga gruppen frie muligheter til å sette egne krav til hva som var tilstrekkelig å utvikle av funksjonalitet i den gitte tidsrammen. Oppdragsgiver var også fleksibel for beslutninger tatt av gruppen, og ønsket selv å være aktiv del av prosessen. Med disse punktene ville ikke en rigid utviklingsmetodikk som Fossefall, gi mening for oppgaven. Dermed falt valget på SCRUM.

Videre valg som gruppen tok rundt SCRUM, var bruken av timesestimat. Diskusjon rundt dette ble tatt i seksjon 2.4.5. Har ble det diskutert rundt valget om å bruke Planning Poker, men grunnet vanskeligheter med å estimere verdiene for en Task, ble timesestimat valgt. I begynnelsen oppdaget gruppen at de estimerte alt for mye rundt enkelte Tasks, for eksempel et estimat som var satt til én uke, kunne ta fem timer, eller motsatt. Etter samtaler innad i gruppen ble det sett at User Stories og Tasks var alt for store og måtte deles opp i mindre deler. Dette ga bedre flyt av arbeidsfordeling gjennom de senere sprintene, samt at Burndown Chartene jevnere holdt ved den anbefalte fallhastigheten,

se til figur 1 og 2 for forskjell.

Gruppen ble også nødt til å ta et valg rundt lengden på Sprintene, som tatt opp i møter med produkteier (Se seksjon 2.4.2 Interne møter) under "Sprint Retrosepective". Gruppen så at én ukes Sprinter ikke strakk til, siden det fremdeles lå igjen ufullførte Tasks og gjorde at disse ble flyttet over til neste Sprint (Se F). Etter et Sprint Retrospective møte innad i gruppen og Sprint Planning møte holdt med produkteier i slutten av Sprint tre, ble det enighet at Sprinter ville forlenges til to uker (Se M referat 20.02.2018). Dette førte til at gruppen klarte bedre å fullføre valgte Tasks satt i videre sprinter (SeF Sprint 4-7).

Når gruppen forandret på lengden til sine sprinter gikk de også bort fra planen lagt i Gant-skjemaet (Se B Gant Skjema), hvorav det var forventet at det skulle kjøres 14 én ukes sprinter. Dette ble da halvert ned til syv, men hvor hver enkelt var nå to uker istedenfor én. Gruppen kuttet også ut sprint 15 og 16 i Gant-skjemaet, fordi det ble valgt å legge fokus på å bruke SCRUM kun til utvikling av produktet, ikke rapportskriving. Gruppen stoppet også utviklingen tidligere enn planlagt. I følge Gant-skjemaet (Se B Gant Skjema), så stopper utvikling 1. mai, mens gruppen valgte å stoppe utviklingen i enighet med produkteier den 17. april. Dette på grunn av at gruppen ønsket å fokusere mer på rapporten.

Gjennom utviklingen har gruppen hatt lite fokus med tanke på sikkerhet. Tanken var fra Fosen Utvikling at gruppen ikke skulle bruke tid på implementering av kode med tung sikkerhet, men heller tenke ut hva som var lurt å gjøre i forskjellige situasjoner. Gruppen noterte seg forskjellige sikkerhetshull som de selv er kjent med, og legger dette inn i potensielt videre arbeid(Se Videre Arbeid 7.5) for Open Source prosjektet.

7.1.1 Valg av språk og utviklingsmiljø

Under lynkurset med oppdragsgiver fikk gruppen vite at programmering mot Arduino hardware er fastsatt til å bruke en variant av C++, og har et eget integrert utviklingsmiljø som kan bli tatt i bruk. Fra tidligere fag var gruppen kjent med bruk av JavaDoc for dokumentering av kode, og ønsket derfor å finne en mulighet til å bruke dette mot C++ og Arduino. Ettersom JavaDoc ikke fungerer mot C++, ble DoxyGen et alternativ som fungerer med JavaDoc syntax. Dette ble forsøkt gjennomført ved bruk av Eclipse som utviklingsmiljø, hvor gruppen ønsket å samle kode fra hele prosjektet for enkel testing og dokumentering. Dette medførte store vanskeligheter, da Eclipse ikke var godt nok utstyrt for programmering mot Arduino enheter. Derfor endte valget på å utvikle Arduino i sitt eget utviklingsmiljø, Arduino IDE.

Når gruppen var i gang med utviklingen av styringspunktet dukket spørsmålet opp igjen om hvilke programmeringsspråk og utviklingsmiljø som skulle bli brukt. Ettersom styringspunktet bruker Raspbian, som er en lettvariant av Linux, var det ikke nødvendig å bruke frittstående utviklingsmiljø, men heller bruke terminalen for utvikling. Valget på språk ble basert på råd fra personer som drev med utvikling av Arduino og Raspberry Pi. Forslagene strekte seg over mange språk, men Python stakk ut som det språket som oftest ble anvendt. Ingen i gruppen hadde forkunnskaper innen utvikling i Python, men alle var ivrige etter å lære. Python ble brukt for å utvikle lytteren mot sensoren, dette

i gjennom publish/subscribe med MQTT server installert på styringspunktet. Valget for bruk av MQTT var en felles avgjørelse mellom gruppen og oppdragsgiver, da dette fjernet begrensningene på antall sensorer i et nettverk. For at MQTT kommunikasjonen skulle opprettes var det nødvendig for sensoren å ha tilgang til styringspunktets SSID, og MQTT serveren sitt brukernavn og passord. For at dette skulle være mulig, måtte gruppen finne en løsning hvor opplysningene kunne sendes til sensor etter at koden hadde blitt lastet opp. Dette ble da løst gjennom bruken av et tilkoblingspunkt, i form av en WiFiManager. Tanken bak dette er at leverandør laster opp kode med gitt brukernavn og passord til MQTT som blir medsendt bak på styringspunktet, slik som et modem. Gjennom koblingspunktet må det også skrives IP-adressen til MQTT serveren, denne blir den samme som IP-adressen til styringspunktet og skal etterhvert bli satt statisk av leverandør gjennom bruken av domene.

Først og fremst var idéen å utvikle en applikasjon på Raspberry Pi som skulle anvendes av brukeren. Denne skulle kobles til styringspunktet direkte via en mobil eller datamaskin for å vise sensor data og gi brukeren muligheten til å sette egendefinerte regelverk. Dette viste seg å være en dårlig ide da gruppen hadde problemer med å låse resten av Linux systemet vekk fra applikasjonen og brukeren ville få alt for mye informasjon på en gang. I et møte med oppdragsgiver endret planen seg til å lage et forslag til nettside, siden Fosen Utvikling skulle selv lage en nettside når prosjektet går over til Open Source (Se vedlegg M. Referat 20. februar). Når det kom til videre valg av språk for kommunikasjon mellom styringspunkt å webserver, falt avgjørelsen på JavaScript med rammeverkene Node.Js og Express.js med bruk av Socket.IO biblioteket. Dette ble valgt ettersom prosjektet krevde en stor hastighet på data sendt i sanntid, hvor dataen er små pakker med tekst og tall.

7.2 Resultater

7.2.1 Mål

Før prosjektet startet formulerte vi effektmål, resultatmål, og læringsmål (Se G Prosjektplan). Disse er stort sett formulert av gruppen selv da målene fra Fosen Utvikling ikke var spesielt spesifikke. De fleste målene er blitt møtt, med unntak av en ferdigstilt funksjonalitet for regelverk og operasjonell nettside. Disse ble ikke ansett som viktig fra oppdragsgiver da det egentlige målet satt fra dem var å opprette kommunikasjon mellom sensor og styringspunkt.

Produktet ser kanskje ikke ut som særlig mye arbeid basert på mengden funksjonalitet som finnes, men det ligger mye arbeid i å få et slikt produkt til å fungere med så mange komponenter. Den underliggende funksjonaliteten er omfattende og særlig funksjonalitet knyttet til kommunikasjon er godt planlagt og implementert i samsvar med effektmålene angående videreutvikling i et Open Source miljø.

Når det kommer til læringsmålene satt av gruppen i prosjektplanen(Se G Prosjektplan), er alle mål oppnådd med tanke på utvikling mot hardware og prosessen for gjennomføring av et prosjekt fra start til slutt. Gruppen brukte litt tid på å skjønne verdiene av møter seg i mellom, og grunnet en del sykdom og andre hendelser var det ikke alltid like enkelt å foreta et møte. Men møter med veileder og oppdragiver har vært uvurderlige

gjennom hele prosessen.

7.3 Gruppeevaluering

Under hele planlegging- og utviklingsprosessen har gruppen jobbet tett sammen. Det ble oppdelt forskjellige hovedansvar innad i gruppen, men selv om de hadde hovedansvaret for hver sin del, hadde hele gruppen godt innblikk i alt som ble utviklet. Dette gjorde at det alltid var enkelt å spørre om hjelp om noen skulle bli sittende fast.

Gruppen satte opp i gruppereglene at for sent oppmøte skulle straffes med bøtlegging, dette var en stor motivasjon til å møte opp i tide og legge ned en innsats for felleskapet. Gruppen har ikke hatt noen uenigheter og jobber godt sammen.

Noe hele gruppen kunne ha vært bedre på var å utnytte bruken av Trello og Jira for oversikt over hvem som jobbet med hva, de gangene det ikke ble brukt et felles arbeidssted. Men ettersom hver enkelt person hadde sitt eget ansvarsområde var det sjeldent at det ble noen konflikter i arbeidet.

7.4 Kritikk av oppgaven

Da Fosen Utvikling la frem oppgaven for gruppen var det mye forvirring i oppstartsfasen om hva som egentlig skulle bli utviklet. Gruppen hadde en plan og oppdragsgiver hadde få fastsatte punkter som måtte oppnåes(Se J Original Oppgavebeskrivelse. I bunn og grunn var tanken at gruppen ville bruke hele prosjektperioden på å utvikle et system for kommunikasjon mellom en sensor og et styringspunkt, men dette viste seg fort å være for lite. Dette ga gruppen en stor frihet i forhold til valg videre, og det ble holdt jevnlig kontakt med oppdragsgiver med nye idéer. Fosen Utvikling ble fort imponert over arbeidsinnsatsen og fremgangen i prosjektet. Et annen kritikk til oppgaven er at gruppen brukte mye tid på å lære emner som for eksempel å sette opp Arduino brett, som er utenfor fagfeltet programvareutvikling. Etter å ha sett på tidligere bachelor oppgaver ser gruppen at det kunne ha vært enklere å kun fokusert på å produsere en mobil app eller en webside løsning. Det ville vært enklere å komme til et ferdig produkt, og mer å vise til i rapporten.

Under initielt møte med oppdragsgiver fikk gruppen utlevert alt av utstyr(Se P Utstyrsliste) som var nødvendig for å gjennomføre prosjektet, men ble også gitt fullmakt til innkjøp av nytt utstyr om dette ble nødvendig.

Fosen Utvikling har stilt opp gjennom hele prosjektet og har alltid vært til stedet om det skulle være noen mangler eller spørsmål fra gruppen. Oppgaven som de sendte inn var en stor og åpen oppgave noe som satte mye ansvar på gruppen selv, og krevde en bratt læringskurve.

7.5 Videre arbeid

Med tanke på videre arbeid av prosjektet så skal dette bli Open Source når Fosen Utvikling tar over kodebasen. Gruppen har gjennom hele prosessen tenkt langt fram i tid med tanke på koden som er skrevet. Det er gjort klart til utbygging av nye sensorer og

det skal være enkelt for en ny bidragsyter å bli med i prosjektet. Det er mye å ta tak i ettersom gruppen har laget grunnmuren for et videreutviklingsprosjekt, som var et ønske fra Fosen Utvikling.

Siden sikkerhet ikke har vært fokusert på underveis i utviklingen har produktet tilhørende åpenbare sikkerhetshull. Publish og Subscribe anvendelsen er særdeles åpen for angrep eller avlytting, siden alle har tilgang til kildekoden hvor noen av de fastsatte avlyttings temaene ligger. Brukere er da utsatt for avlesing av deres sensorers data, MySQL database injekson på serversiden og i ekstreme tilfeller at andre kan ta kontroll over brukernes reléer. Skape løsninger for disse sikkerhetshullene vil da være forslag for videre arbeid.

Gruppen tok opp i seksjon 4.3 Idé for webapplikasjonens brukergrensesnitt utvikling for et webgrensesnitt. Grensesnittet skal representere sensor data, samt mulighet til å lage og sende regelverk tilbake til styringspunkt. Gruppen har produsert skisser for en løsning av grensesnittet. Skissene representerer temperatur sensors data og prosessen for å lage et nytt regelverk ved bruk av IFTTT metoden. Gruppen har også skrevet kode for en mulig løsning av webgrensesnitt ut i fra disse skissene. Videre arbeid vil være å implementere backend med webgrensesnittet. Data må vises på websiden og ha mulighet til å sende informasjon om valg fra bruker tilbake til styringspunkt. Det vil også være nødvendig å utvikle resterende grensesnitt for manglende sensorer.

7.6 Konklusjon

Det er forståelig å se at selskaper spesialiserer seg, og holder utviklingen innenfor kjente fagfelt siden det er tidskrevende å sette seg inn i nye teknologier, språk og rammeverktøy. Positive utfall av et prosjekt med slikt omfang gir større utbytte i form av kunnskaper og ferdigheter. Selv om det mangler ønsket funksjonalitet i form av ferdigstilt nettside og regelverk for relé, er håpet at produktet står til forventningene og at det kan bli anvendt og videreutviklet i ettertid. Avslutningsvis fortjener Fosen Utvikling ros for oppgaven med tanke på omfang og læringspotensial.

Bibliografi

[1] Wikipedia. Aktuator. URL: https://no.wikipedia.org/wiki/Aktuator.

[2] cc, A. Arduino. URL: https://www.arduino.cc/.

[3] cc, A. Arduino uno rev3. URL: https://store.arduino.cc/arduino-uno-rev3.

[4] Wikipedia. Baud. URL: https://no.wikipedia.org/wiki/Baud.

[5] Wikipedia.

[6] Wikipedia. Doxygen. URL: https://no.wikipedia.org/wiki/Doxygen.

[7] Eernisse, E. Ejs. URL: http://ejs.co/.

[8] Wikipedia. Express.js. URL: https://no.wikipedia.org/wiki/Express.js.

[9] Wikipedia. Ifttt. URL: https://en.wikipedia.org/wiki/IFTTT.

[10] Forbes, J. M. A simple explanation of 'the internet of things. URL: https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#6d898c2f1d09.

[11] Wikipedia. Jira (software). URL: https://en.wikipedia.org/wiki/Jira_(software).

[12] Wikipedia. Mqtt. URL: https://en.wikipedia.org/wiki/MQTT.

[13] Wikipedia. Node.js. URL: https://no.wikipedia.org/wiki/Node.js.

[14] Systems, E. Esp8266. URL: https://www.espressif.com/en/products/hardware/esp8266ex/overview.

[15] Wikipedia. Åpen kildekode. URL: https://no.wikipedia.org/wiki/Åpen_kildekode.

[16] Wikipedia. Planning poker. URL: https://en.wikipedia.org/wiki/Planning_poker.

[17] Rossum, V. Foreword for programming python. URL: https://www.python.org/doc/essays/foreword/.

[18] Cellan-Jones, R. A 15 pound computer to inspire young programmers. URL: http://www.bbc.co.uk/blogs/thereporters/rorycellanjones/2011/05/a_15_computer_to_inspire_young.html.

[19] Wikipedia. Raspbian. URL: https://en.wikipedia.org/wiki/Raspbian.

[20] Wikipedia. Relé. URL: https://no.wikipedia.org/wiki/Relé.

[21] Wikipedia. Schematic. URL: https://en.wikipedia.org/wiki/Schematic.

[22] IS, S. 2011. *Software Engineering 9th ed.* Boston: Pearson.

[23] Wikipedia. Sensor (instrument). URL: https://no.wikipedia.org/wiki/Sensor_(instrument).

[24] Wikipedia. Service set identifier(ssid). URL: https://en.wikipedia.org/wiki/Wi-Fi#Service_set_identifier_(SSID).

[25] Wikipedia. Single-board computer. URL: https://en.wikipedia.org/wiki/Single-board_computer.

[26] Hill, J. The smart home: a glossary guide for the perplexed. URL: https://www.t3.com/features/the-smart-home-guide.

[27] Wikipedia. Socket.io. URL: https://en.wikipedia.org/wiki/Socket.IO.

[28] SA, S. Sonarqube concepts. URL: https://docs.sonarqube.org/display/SONAR/Concepts.

[29] Wikipedia. Sublime text. URL: https://en.wikipedia.org/wiki/Sublime_Text.

[30] Wikipedia. Trello. URL: https://en.wikipedia.org/wiki/Trello.

[31] Wikipedia. Realvnc. URL: https://en.wikipedia.org/wiki/RealVNC.

[32] tablatronix. Wifimanager. URL: https://github.com/tzapu/WiFiManager.

[33] Merriam-Webster. Workshop. URL: https://www.merriam-webster.com/dictionary/workshop.

[34] Verge, J. V. T. Google uses deepmind ai to cut data center energy bills. URL: https://www.theverge.com/2016/7/21/12246258/google-deepmind-ai-data-center-cooling.

[35] Wikipedia. Sewing machine. URL: https://en.wikipedia.org/wiki/Sewing_machine.

[36] Wikipedia. Dishwasher. URL: https://en.wikipedia.org/wiki/Dishwasher.

[37] Wikipedia. Clothes dryer. URL: https://en.wikipedia.org/wiki/Clothes_dryer.

[38] Wikipedia. Home automation. URL: https://en.wikipedia.org/wiki/Home_automation.

[39] Dormehl, L. 5 open source home automation projects we love. URL: https://www.fastcompany.com/3038442/5-open-source-home-automation-projects-we-love.

[40] Fahmy, H. M. A. Wireles sensor networks: Concepts, applications, experimentation and analysis. URL: https://books.google.no/books?id=oYmwCwAAQBAJ&pg=PA108&lpg=PA108&source=bl&sa=X&redir_esc=y#v=onepage&q&f=false.

[41] Azure, M. Event-driven architecture. URL: https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/event-driven.

[42] Wikipedia. Publish/subscribe pattern. URL: https://en.wikipedia.org/wiki/Publish-subscribe_pattern.

[43] Inc, I. If this then that method. URL: https://platform.ifttt.com/docs.

[44] B.V., A. Vis.js bibliotek. URL: http://visjs.org/.

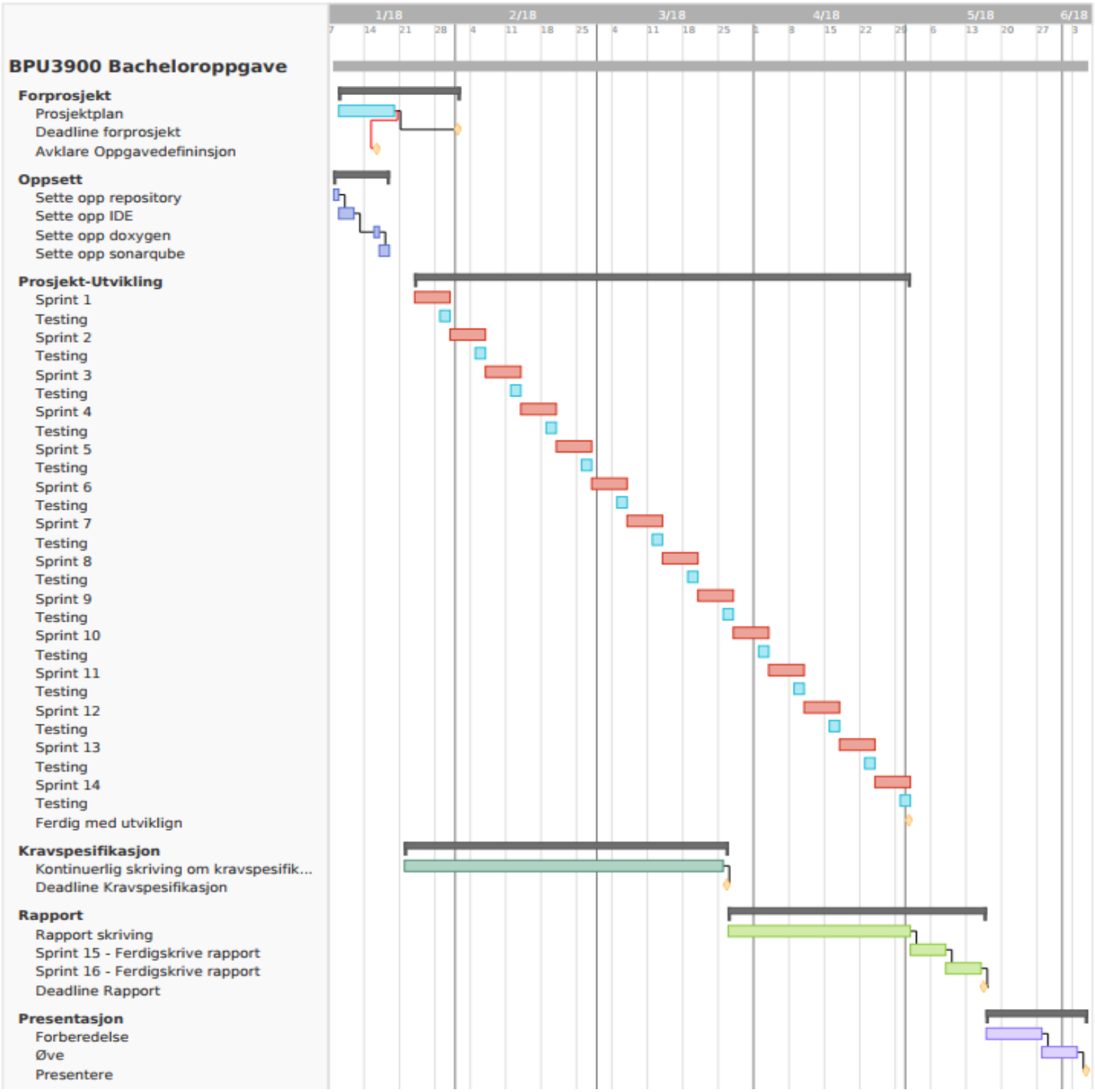
[45] Espressif. Esp8266 overview, technical references. URL: https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf.

Vedlegg

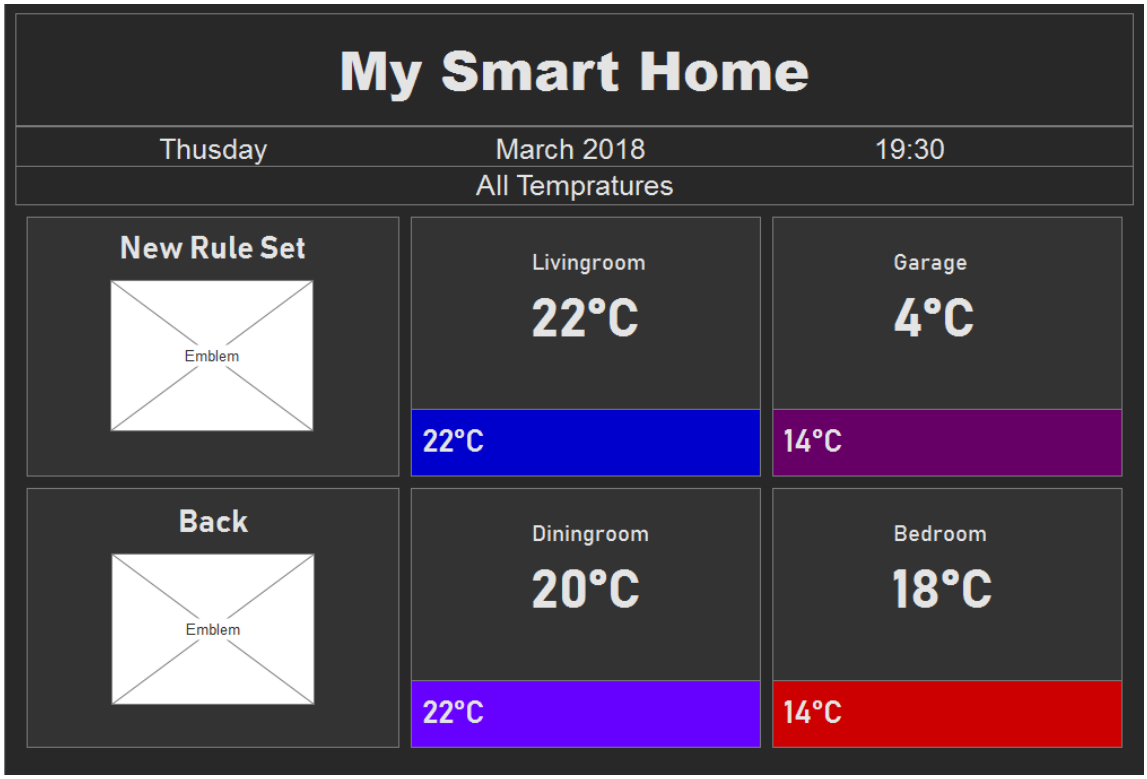
A Terminologi

Se Ordliste

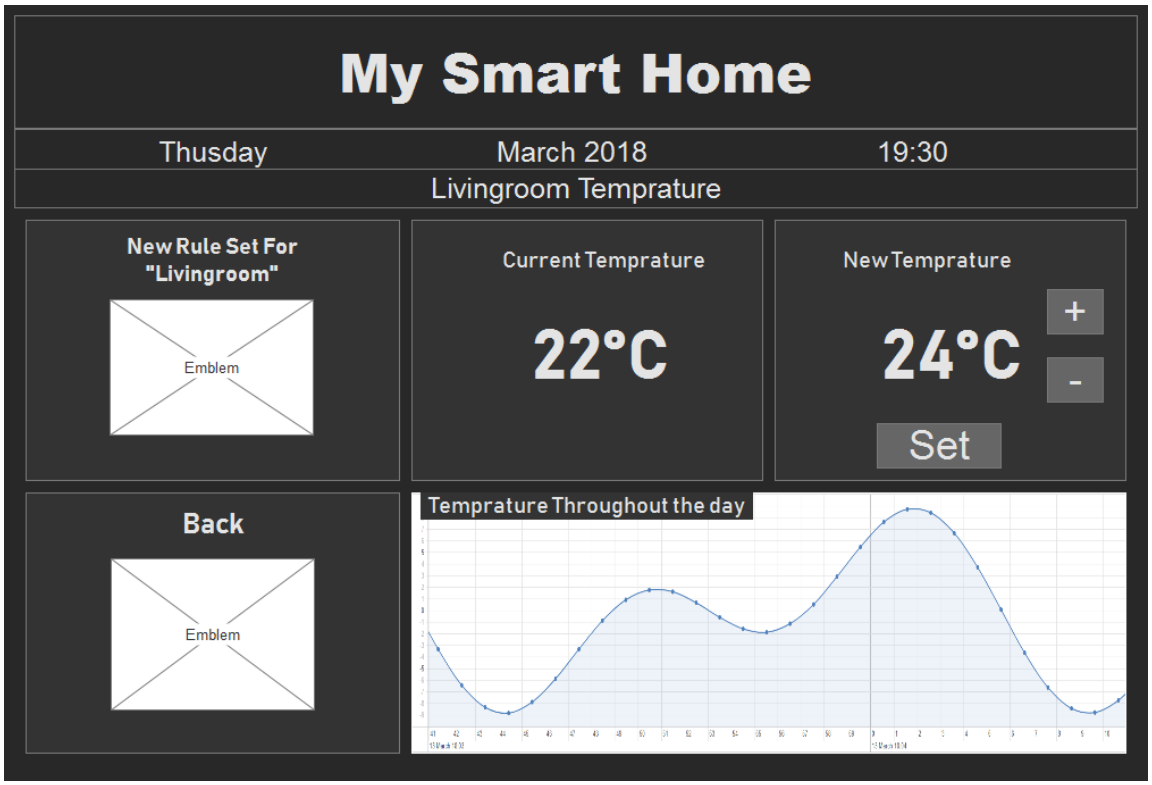
B Gant Skjema



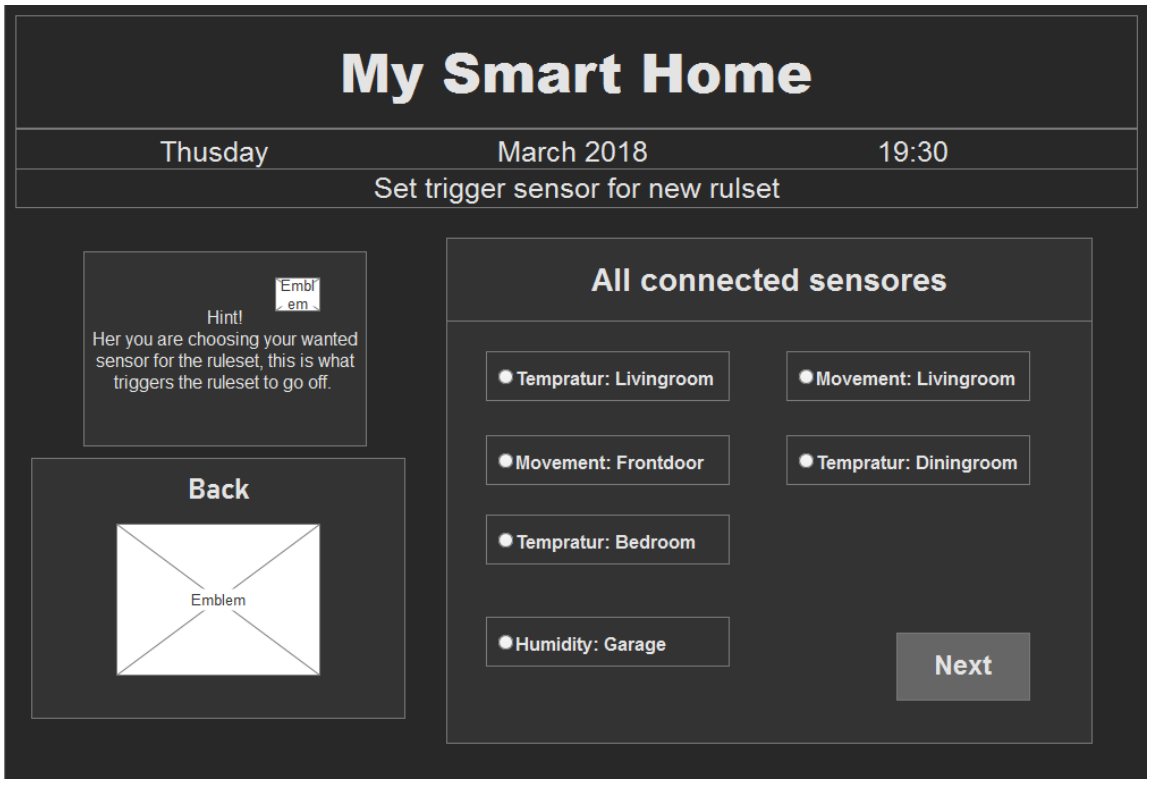
C Skisser for webside



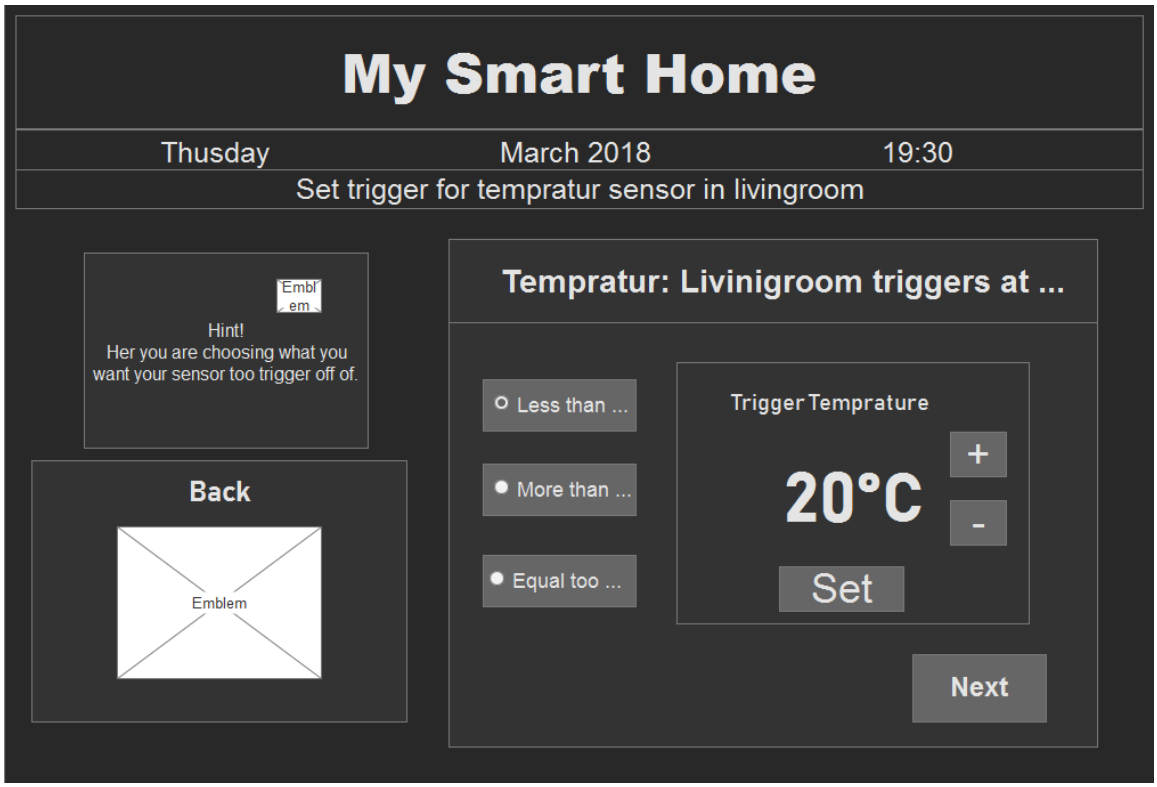
Figur a Skisse for å vise all temperatur data



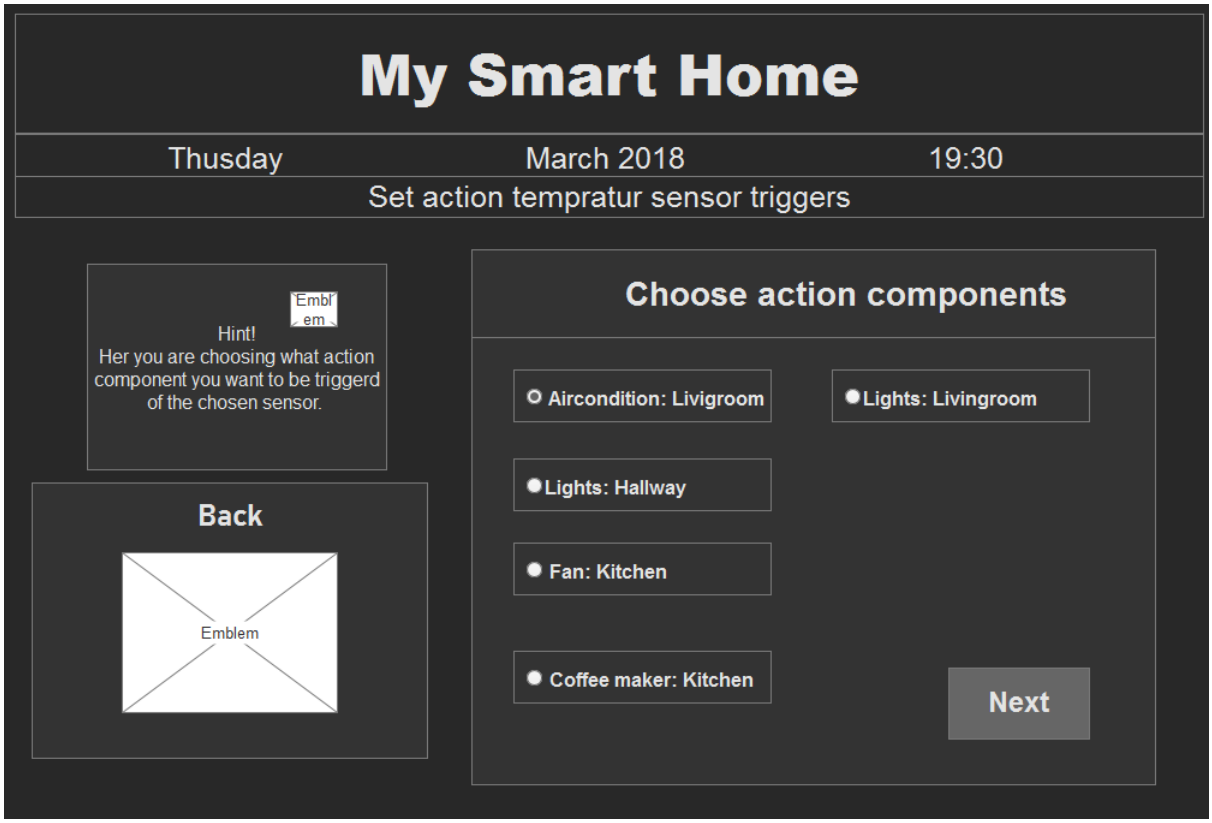
Figur b Én sensor lokasjon sin data



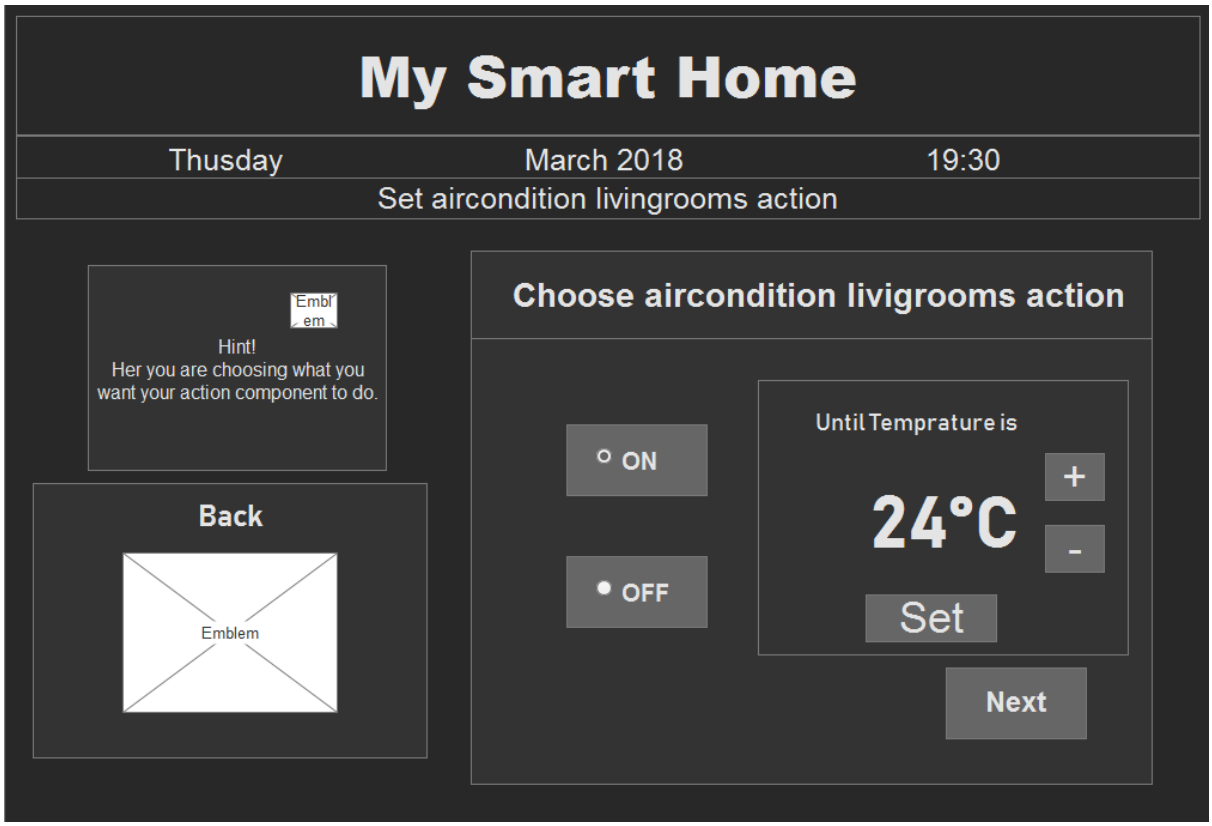
Figur c Start av nytt regelverk, velger ønsket sensor



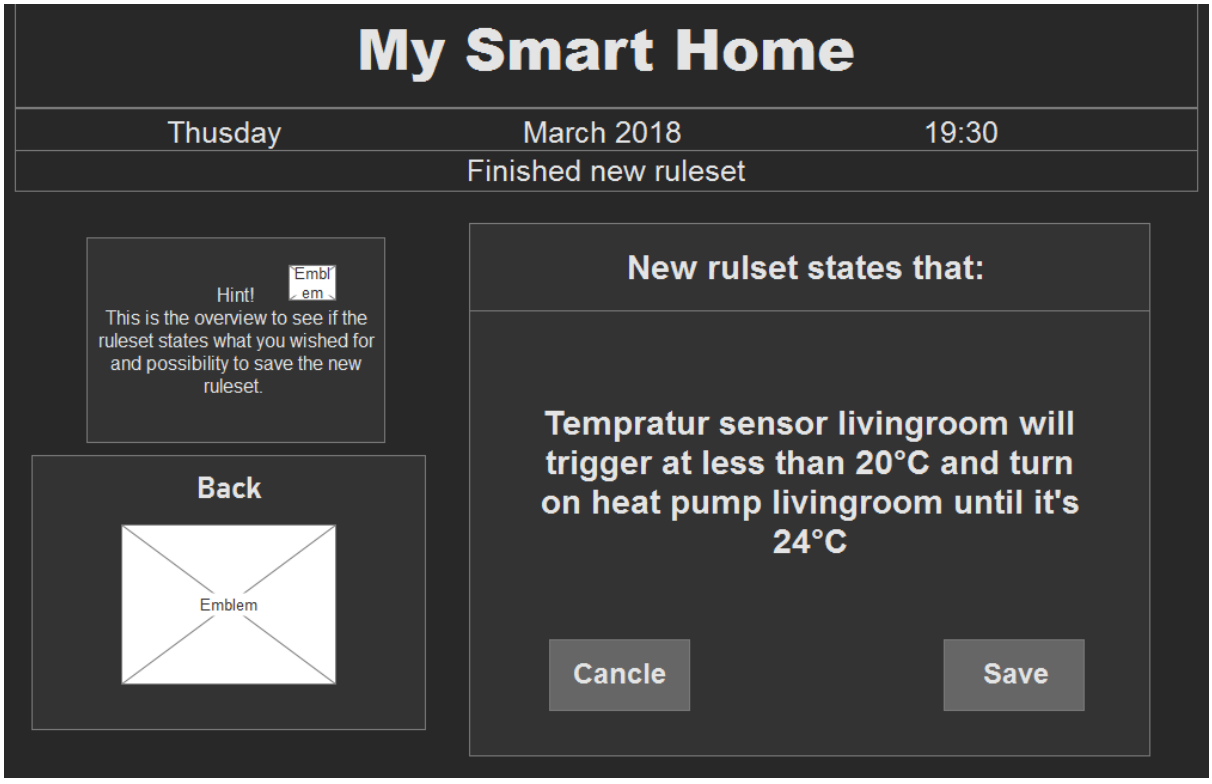
Figur d Velger hva temperatur sensor skal trigge på



Figur e Velger action komponent som skal gjøre handling



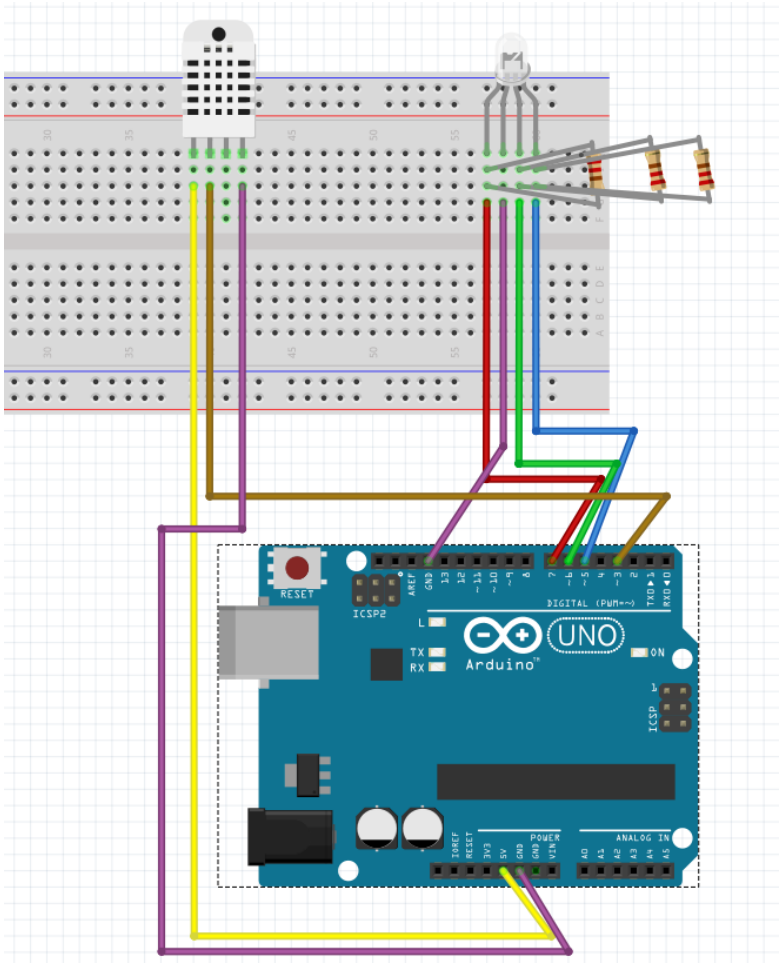
Figur f Velger hva action komponent sin handling er



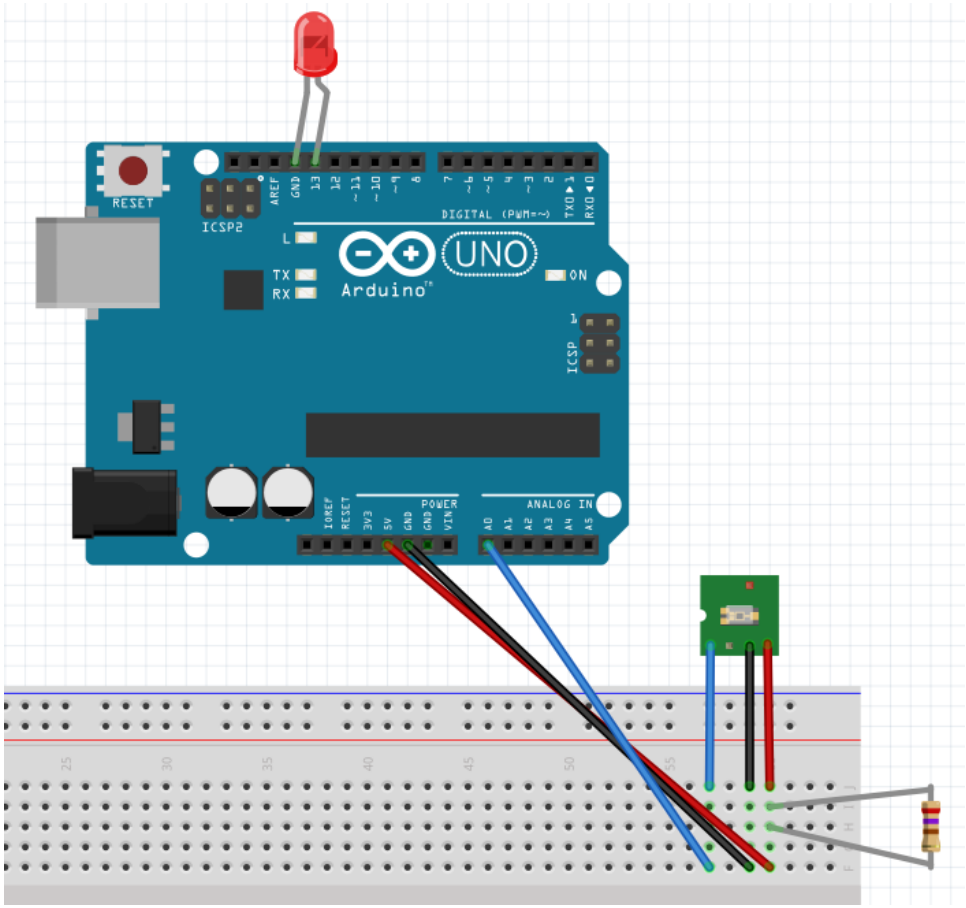
Figur g Ser fullført regelsett med trigger og action komponenter

D Schematics

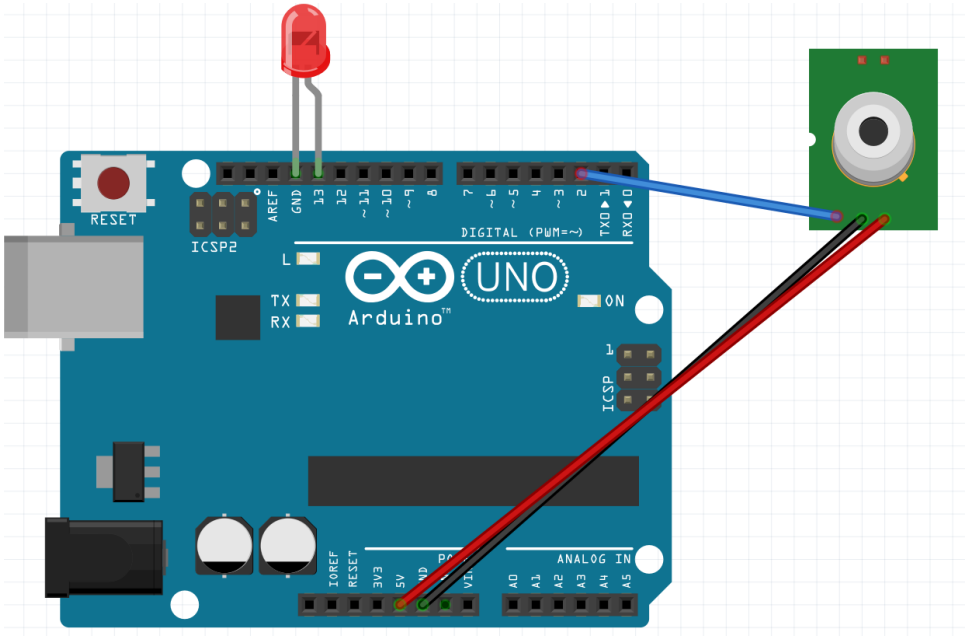
Arduino Schematics



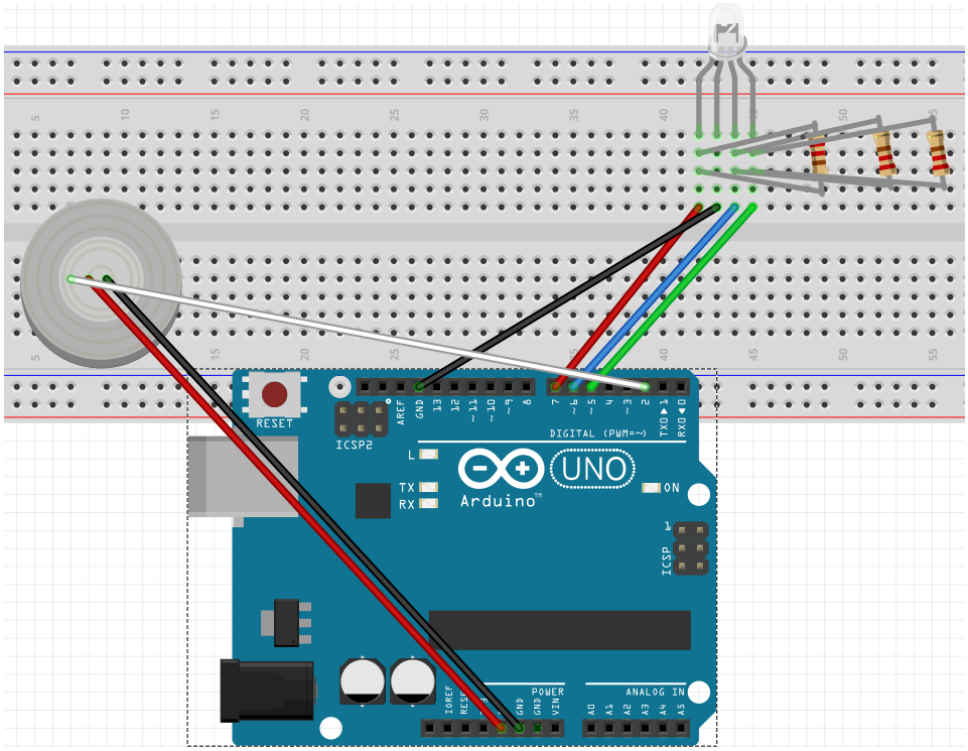
DHT22 thermistor, ESP8266 WiFI Module, YP-05 FTDI, YwRobot. Temp Humidity Sensor



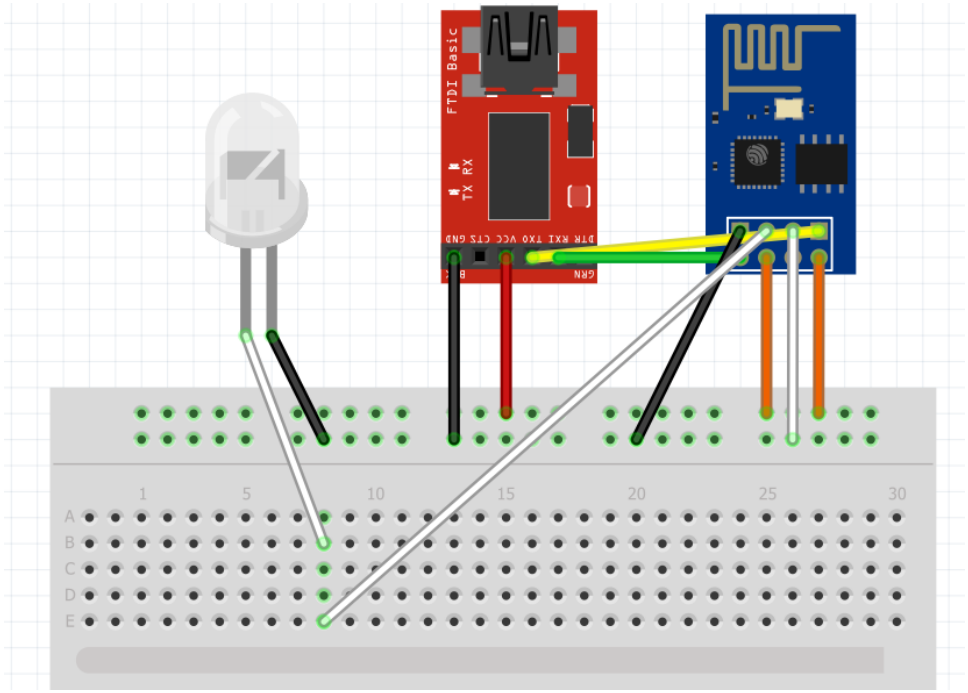
Arduino Touch with Led



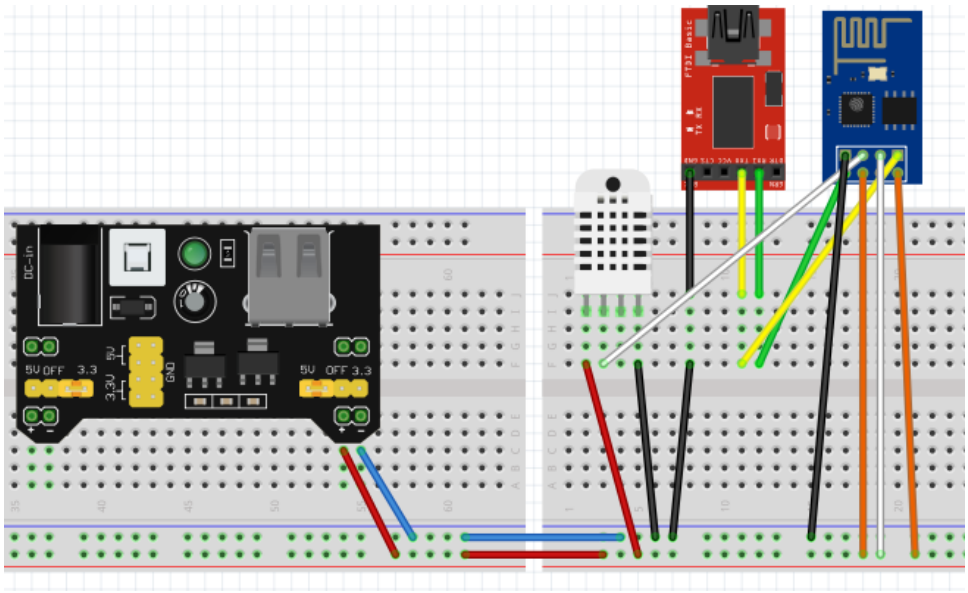
Arduino Uno Pir sensor



Arduino UNO touch with RGB led response



Relay YP-05 FTDI, ESP8266 WiFI Module Relay On/Off led



DHT22 thermistor, ESP8266 WiFI Module, YP-05 FTDI, YwRobot. Temp Humidity Sensor

E Ekstra Use Cases

Use case navn: Bevegelse registrert
Aktører: Bevegelses Sensor
Mål: Registrere bevegelse foran sensor
Beskrivelse: Sensoren registrerer bevegelse foran linsen.

Use case navn: Motta nettverksdetaljer
Aktører: Bevegelses Sensor
Mål: Motta nettverksdetaljer fra bruker
Beskrivelse: Sensoren mottar nettverksinnstillinger fra brukeren som navn, tema osv.

Use case navn: Send data
Aktører: Bevegelses Sensor
Mål: Send data til styringspunkt
Beskrivelse: Sensoren sender registrert bevegelses data til styringspunktet.

Use case navn: Kontakt registrert, Sveip-berørelse
Aktører: Berørelses Sensor
Mål: Registrere kontakt, sveip eller berørelse
Beskrivelse: Sensorer registrerer kontakt, sveip eller bevegelse mot sensor overflater.

Use case navn: Motta nettverksdetaljer
Aktører: Berørelses Sensor
Mål: Motta nettverksdetaljer fra bruker
Beskrivelse: Sensoren mottar nettverksinnstillinger fra brukeren som navn, tema osv.

Use case navn: Send data
Aktører: Berørelses Sensor
Mål: Sende registrert berørelse av sensor til styringspunkt
Beskrivelse: Sensoren sender registrert berørelse av sensoren til styringspunktet for videre arbeid.

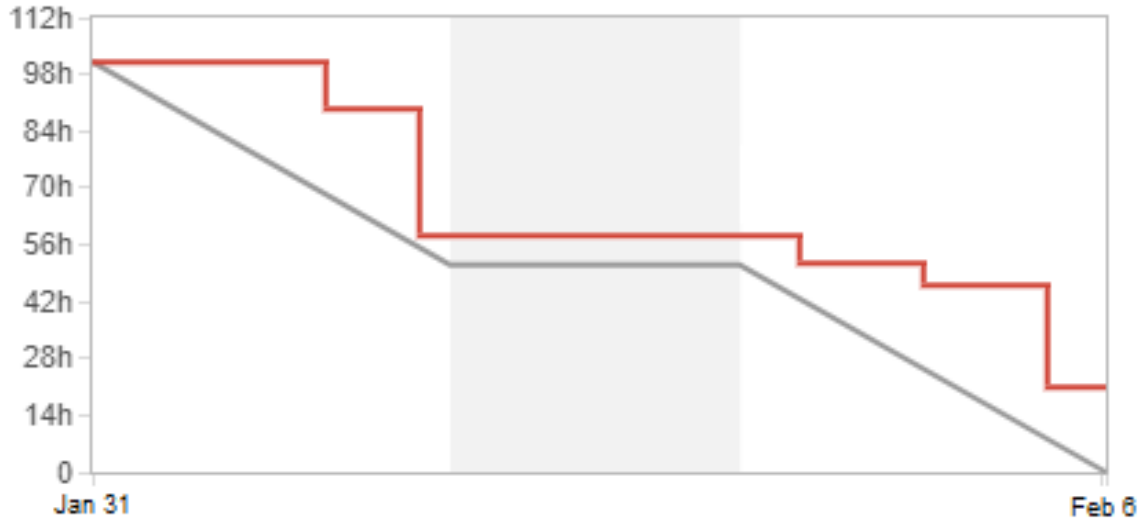
Use case navn: Registrer lys nivå
Aktører: Lys Sensor
Mål: Registrerer lysstyrke i et rom
Beskrivelse: Sensoren registrerer lysstyrken i et rom.

Use case navn: Motta nettverksdetaljer
Aktører: Lys Sensor
Mål: Motta nettverksdetaljer fra bruker
Beskrivelse: Sensoren mottar nettverksinnstillinger fra brukeren som navn, tema osv.

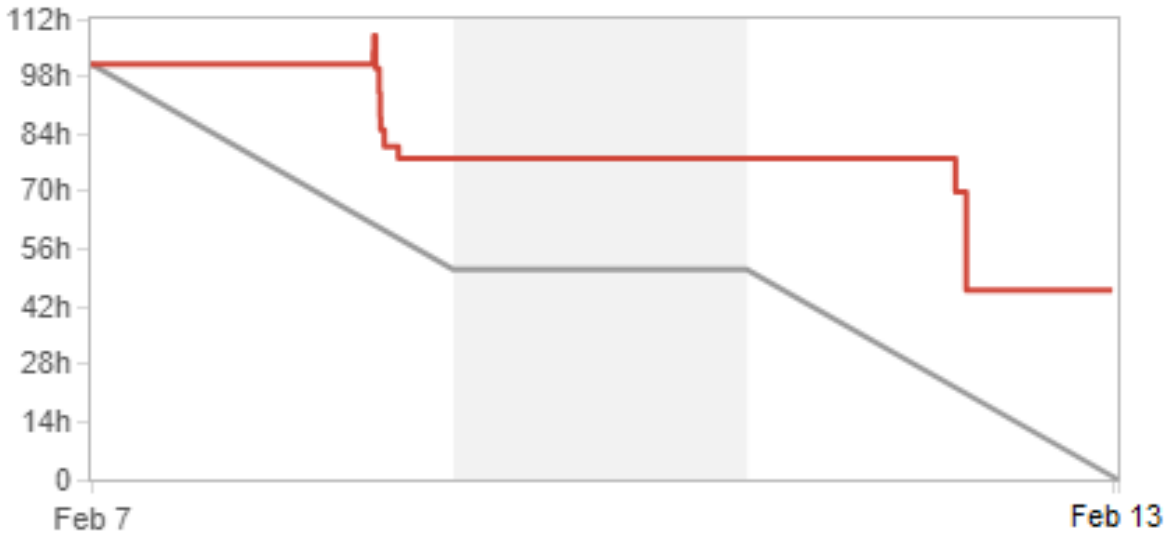
Use case navn: Send data
Aktører: Lys Sensor
Mål: Sende data til styringspunkt
Beskrivelse: Lys sensoren sender lysstyrken i det rommet den er montert i, til et styringspunkt for videre arbeid.

F Grafer for Sprinter

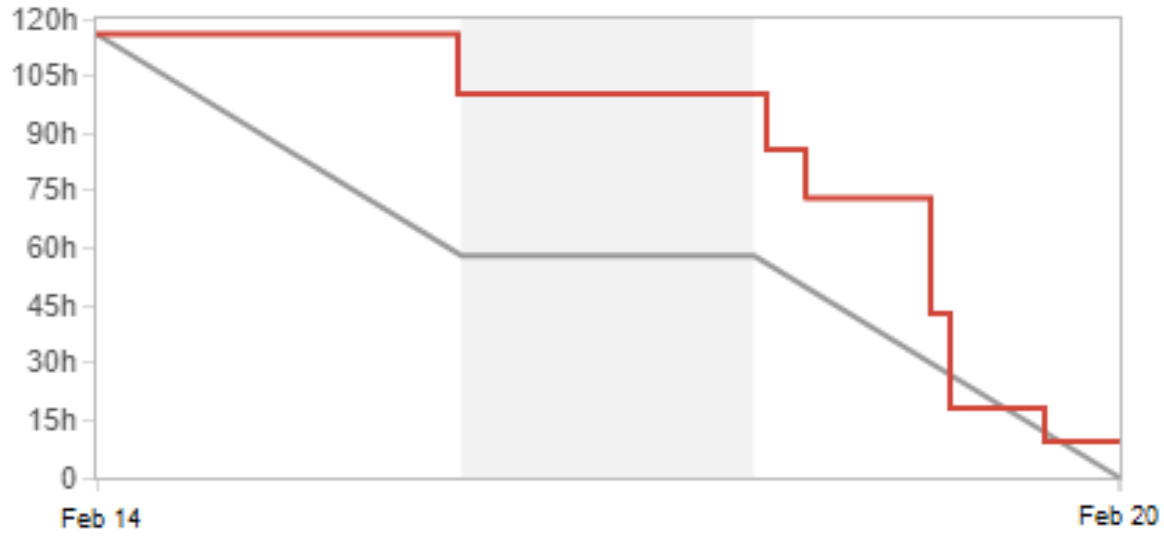
Sprint 1



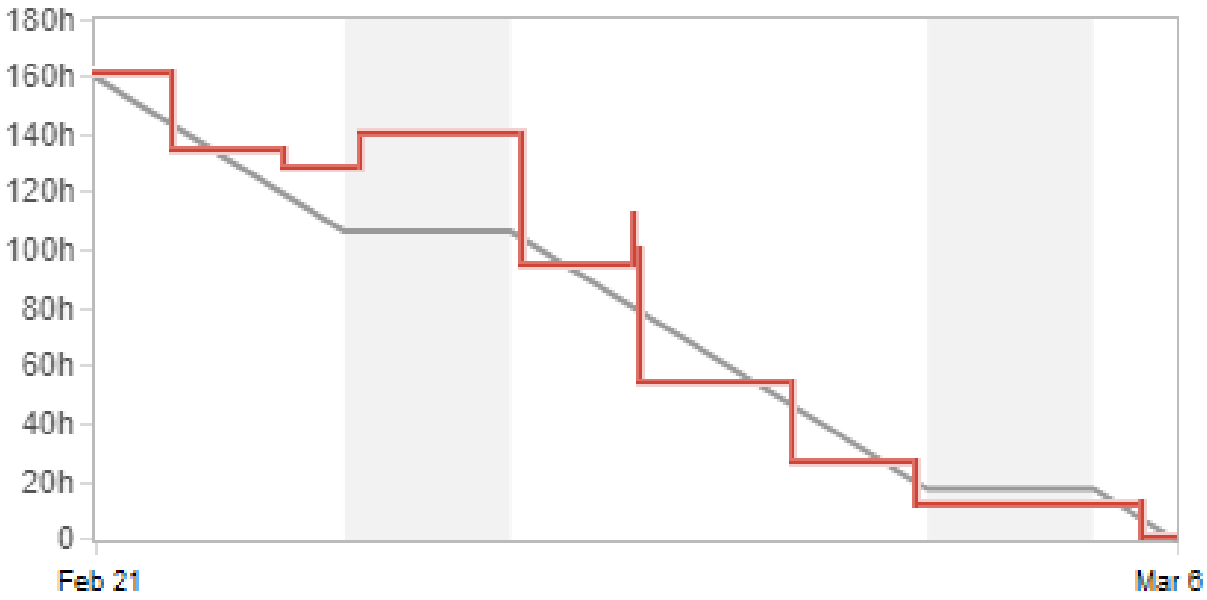
Sprint 2



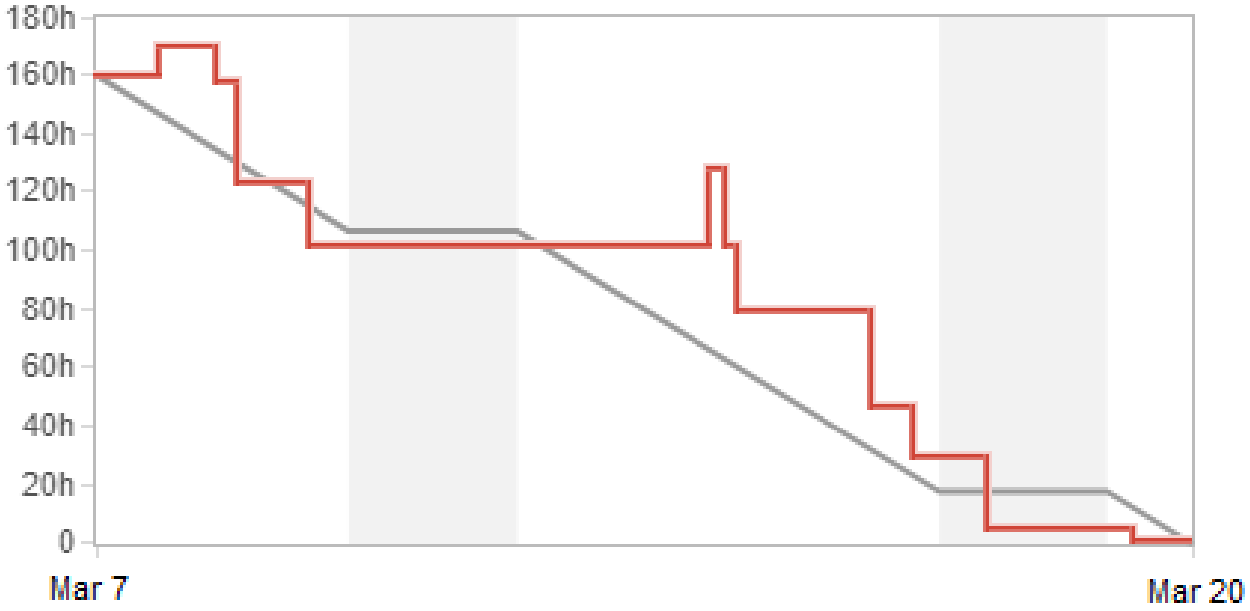
Sprint 3



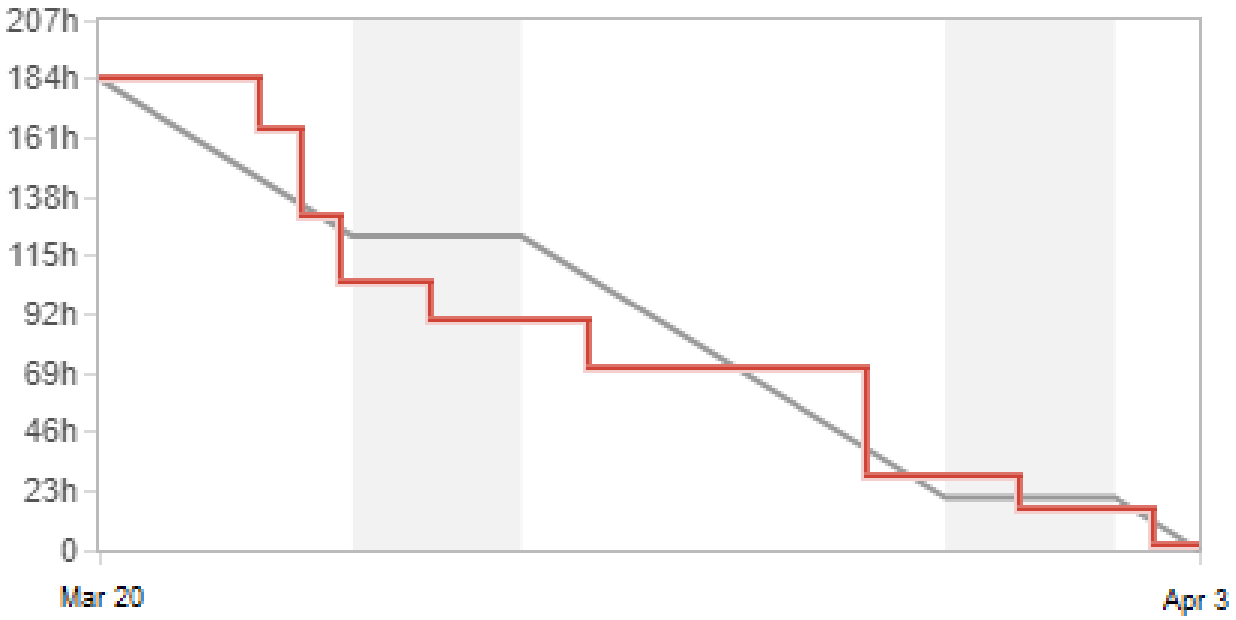
Sprint 4



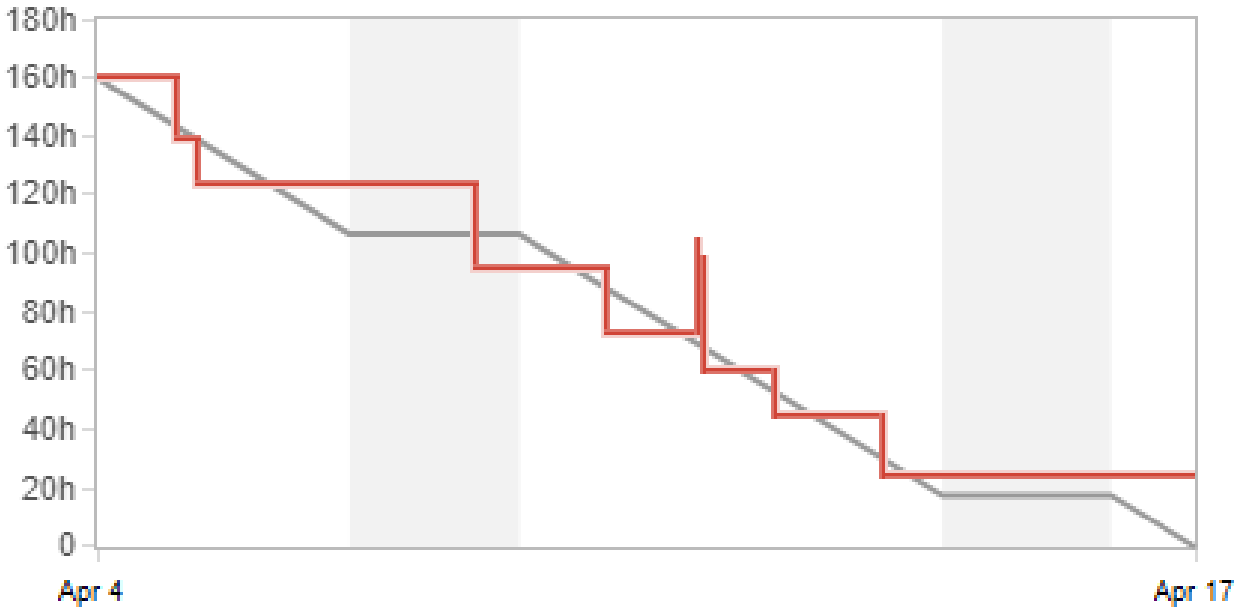
Sprint 5



Sprint 6



Sprint 7



G Prosjektplan

Prosjektplan - MySmartHome

Martin Pukstad, Stian Fenstad, Jakob Fonstad og Kristian Sundhaugen

May 13, 2018

Contents

1 Mål og Rammer

1.1 Bakgrunn

Fosen Utvikling befinner seg i Indre Fosen kommune på Fosen halvøya like utenfor Trondheim, og er en bedrift som tar på seg konsulentoppdrag for alle som sender inn forespørsler. De ansatte ved Fosen Utvikling har tung kompetanse innen forskjellige fagfelt, som informasjonssikkerhet, utvikling og drift. De er hovedleverandør av utviklingstjenster for Dogi AS som er et søsterselskap av Rissa Kraftlag, men tar også på seg andre oppdrag.
Fosen Utvikling er en bedrift som har vokst ut i fra interesse av utvikling, sikkerhet og drifting av IT-systemer. Firmaet ble opprettet av familien Kirkemyr Nilsen og var i en lang periode drevet kun av disse på hobby basis. Fosen Utvikling har siden den gang vokst, og blitt en av de store leverandørene av konsulenter på Fosen halvøya.
Fosen Utvikling ønsker å utvikle sensorer og styringsenheter som skal installeres i hus for overvåkning av strøm, varme, bevegelse og lys, samt ha muligheten til å videreutvikle til et potensielt alarmsystem. Målet for oppgaven er å gi brukeren kontroll over sitt eget hjem over ett lett konfigurerbart sentralpunkt med styring av regler for sensorer, så lenge brukeren har tilgang til internett.

1.2 Prosjektmål

1.2.1 Resultatmål

Prosjektet har som mål å produsere et produkt med sensorer som lett kan settes opp og konfigureres etter egne ønsker. Det skal også være en mulighet for videreutvikling mot et alarmsystem som baserer seg på bevegelsessensorer og fukt sensorer som da kan varsle bruker om noe er galt. Dataen fra sensorene vil bli sendt til en nettside, som skal vise verdiene til enhver tid. Det skal også være mulig å koble seg mot en sentral styringsenhet som kan kontrollere forskjellige punkter i huset, som for eksempel lys eller varme. Det skal også settes opp logikk som gjør det mulig for en bruker å selv definere når og hva som skal skje når individuelle eller flere sensorer registrerer en hendelse. For eksempel at brukeren bestemmer at varmepumpen skrus på når temperaturen i rommet når et visst punkt.

1.2.2 Effektmål

Prosjektet skal:

- Gi oppdragsgiver et nytt produkt for salg.
- Ende opp som Open Source og skal dermed bli en del av produktene som oppdragsgiver har tilgjengelig for allmenheten.

Systemet skal:

- Bidra til at oppdragsgiver kan ha oversikt over forskjellige parametere i sitt hjem, som f.eks temperatur. Samt kunne kontrollere og sette opp egne logiske system med de forskjellige sensorene.
- Ha muligheten til å utbygges med nye sensorer og styringsenheter mot ett sentralpunkt.

1.2.3 Læringsmål

Studentene vil lære:

- Å utvikle mot hardware, i dette prosjektet vil det si Arduino med sensorer og styringsenheter.
- Å gjennomføre et utviklingsprosjekt fra start til slutt med all tilhørende planlegging og koding.
- Verdiene av hyppige møter med gruppen, oppdragsgiver og veileder for utvikling av et produkt.
- Anvending av Scrum-metodikk i et reelt prosjekt.

1.3 Rammer

- Arduino Sensorer skal fungere som dummies, men kommunisere med hverandre gjennom MQTT mot en Raspberry Pi som fungerer som kontrollpunkt.
- In Memory Database skal stå for lagring lokalt på alle Raspberry Pi enheter i JSON format før det sendes til nettside.
- Lagres også på server hos Fosen Utvikling for backup, i SQL.
- Utvikling mot Arduino Hardware i C++

2 Omfang

2.1 Fagområde

Samfunnet digitaliseres mer og mer for hvert år som går. Vi deler mer informasjon over lengre avstander raskere enn noen gang tidligere i menneskelig historie, og vi automatiserer og kontroller stadig mer med hjelp av digitale verktøy. Bedrifter har lenge vært tidlig ute på slike automatiserte systemer, da det potensielt kan spare dem for store utgifter. Google har for eksempel tatt sensorer, automatiserte systemer og kunstig intelligens i bruk for å kutte strømutgifter betraktelig i sine serverparker[2]. Liknende automatiserte systemer har også, i de siste årene, kommet inn i hjemmene til privatpersoner. Dette kan for eksempel være i form av automatisert belysning etter hvilket rom man befinner seg i. Eller styrte systemer slik at man kan starte oppvarmingen av hytta via en app på telefonen, så det er varmt og godt når man ankommer. Nesten alt man kan

tenke seg av elektroniske applikasjoner kan automatiseres eller fjernstyres i dag. Det er også derfor slike systemer fort blir personlige, og det er derfor vanskelig å lage en løsning som passer alle. I dag finnes det ferdig utviklede standard systemer for hjemmeautomatisering, disse er gjerne kommersielle og selges av bedrifter, i små eller større pakkeløsninger. Slike løsninger for privatpersoner kalles "Smarthjem"(SmartHome).
Et smarthjem er per definisjon et hus(hjem) hvor lys, temperatur, ventilasjon eller andre elektriske applikasjoner er styrt eller automatisert via et digitalt system. Når slike systemer er fjernkontrollert via internett regnes de også som å være en del av "Internet of things"[5].
Hjemmeautomatisering er ikke et nytt tema i samfunnet. Automatisering av manuelle oppgaver er noe mennesket har jobbet med lenge. Den første versjonen av elektrisk hjemmeautomasjon kom gjennom applikasjoner som kunne spare oss for manuelt arbeid. For eksempel i form av vankokere(1889), symaskiner(1889)[6], vaskemaskiner(1904), kjøleskap(1916), oppvaskmaskiner(1929)[4] og tørketromler(1938)[3]. Det var ikke før i 1975 at automatisering over nettverk kom inn i hjemmene til folk. Dette gjennom kommunikasjonsprotokollen X10. Denne baserer seg på å bruke strømkabler for å sende signaler og kontrollere elektriske applikasjoner. Disse signalene inneholder radio frekvenser som representerer digital informasjon[7]. X10 brukes fortsatt i dag, da det er et enkelt og billig alternativ. Det har derimot i nyere tid blitt mer og mer normalt å koble seg opp via internett for slike løsninger. Da det gir mulighet til koble seg opp mot et kontroll system fra hvor som helst i verden via for eksempler telefoner, tablets eller andre tredjeparts system. Noe de fleste alt er kjent med og bruker daglig.

2.2 Avgrensning

For prosjektet vil det ikke være nødvendig å utvikle en nettside for overvåkning og styring, da dette allerede er på plass fra Fosen Utvikling, men all informasjon til nettsiden vil være tilgjengelig. Så det kan fokuserers på å sette opp den logisk konfigurerbare funksjonaliteten for de ulike enhetene. Prosjektet har som mål å bli et Open Source prosjekt. Open Source vil Fosen Utvikling selv stå for å sette opp etter endt Bacheloroppgave. Eventuelle blåkopier for oppsett av de individuelle Arduino sensorene må være oversiktlige og tilgjengelige ved ferdgistilling av produktet.
Prosjektet behøver ikke å avgrenses kun til hjemmet, større fasiliteter som arbeidsplasser vil være en reell plattform det er ønskelig å kunne anvende løsningen i. Kontrollpunktet skal nå ut til alle sensorene. Dette løser vi ved å ta i bruk WiFi repeatere.
Oppgaven er satt opp slik at at vi selv skal eksperimentere med forskjellige sensorer, som kan tas i bruk i en smarthjem løsning. Dette avgrenses etter hva som er mulig å koble til et Arduino brett, og hva som er funksjonelt i forhold til å bli tatt i bruk i et Smarthjem.

2.3 Oppgavebeskrivelse

Prosjektet skal bygge ut et smarthjem ved hjelp av noder i et nettverk. Nodene består av Arduino enheter som skal gjenkjenne og tilkoble seg til et sentraltpunkt gjennom WiFi eller Bluetooth. Informasjonen samlet av et sentralpunkt vil videreføres til produkteiers nettside for fremvisning av data gjennom deres RESTful web API. Et sentralpunkt vil bestå av en Raspberry Pi med et brukergrensesnitt der brukerne kan differensiere de ulike sensorene ved å tildele de unike navn, samt å sette opp egne logiske regler. Brukergrensesnittet skal være så enkelt å anvende at brukeren lett kan endre regelverk for sensorene i sitt eget hjem, gjennom en web-server som kjører lokalt på Raspberry Pi og vil være tilgjengelig for brukeren så lenge han er tilkoblet det samme nettverket. Det vil også gjøres tilgjengelig gjennom pålogging på Fosen Utvikling sin nettside.
Typiske sensorer som prosjektet inneholder:

- Temperatur
- Fuktighet
- Bevegelse
- Lys

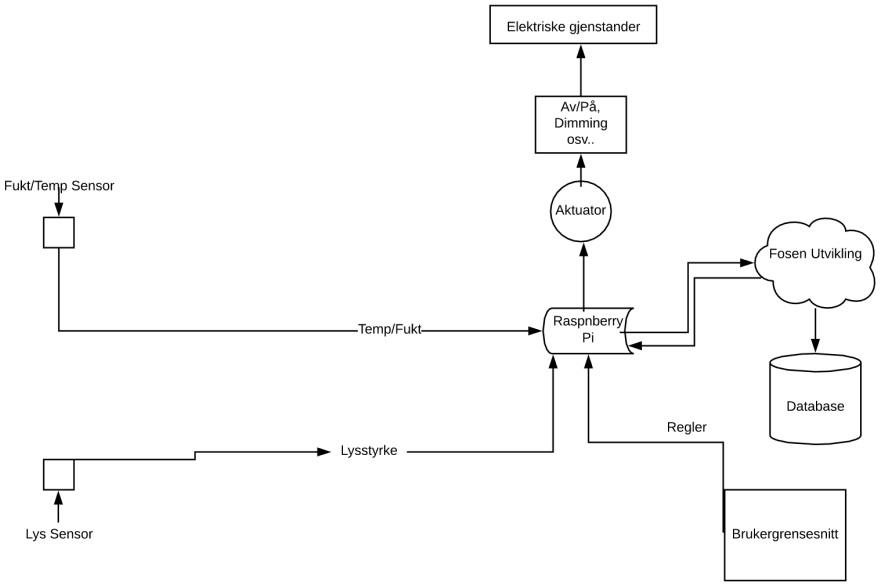


Figure 1: Eksempel på sensoroppkobling

Sensorene skal ha mulighet til å manipulere omgivelsene ved hjelp av aktuatorer. All utvikling vil foregå som prototyping ved programmering mot et "Development Kit". Dette medfører at all hardware er fysisk større en hva sluttproduktet

vil være, og kan ikke testes mot husets elektriske gjenstander (som for eksempel varmepumpen) men vil heller bli representert gjennom bruk av LED lys som reagerer på gitte regelverk satt av en bruker. Ett eksempel på dette vil være at brukeren har satt en regel på at det alltid skal være 24 grader i et gitt rom. Sentralpunktet får tilsendt jevnlige data fra temperatursensoren som indikerer at temperaturen har falt under regelen og vil dermed videresende informasjonen til styringsenheten for varmepumpen (i dette tilfellet et LED lys) om å starte. En eller flere sensorer skal kunne påvirke oppførselen til en eller flere aktuatorer.

3 Prosjektorganisering

3.1 Organisasjonskart

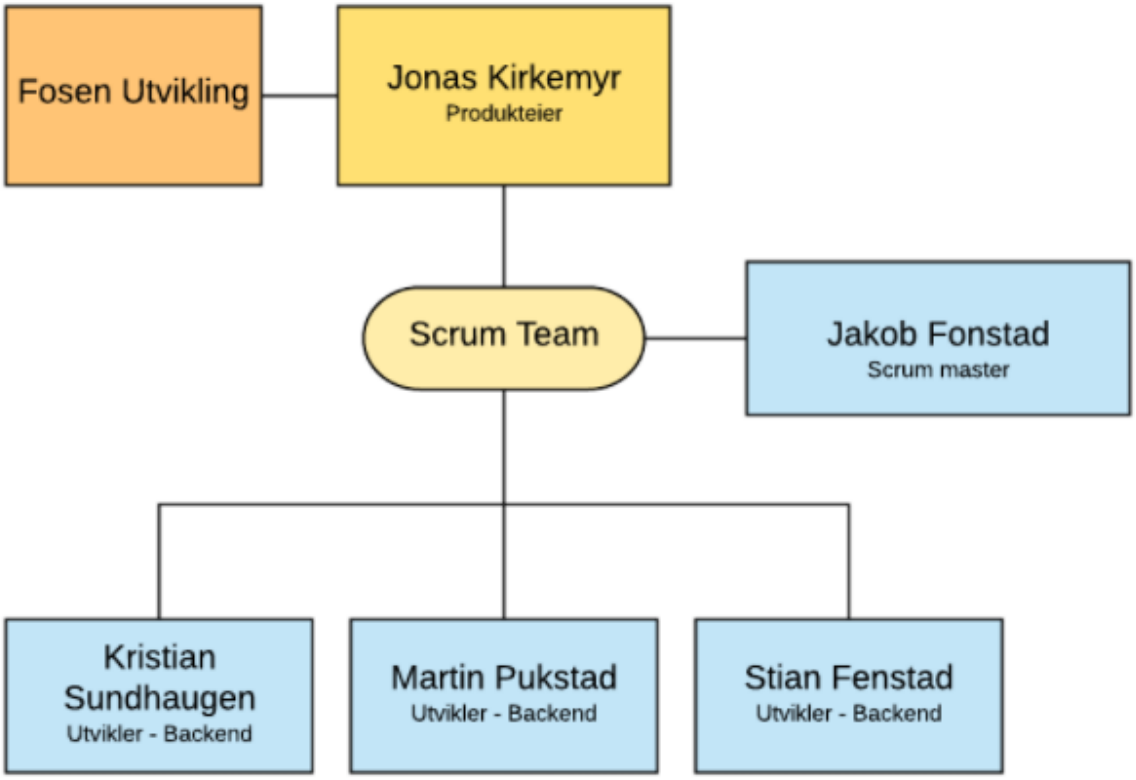


Figure 2: Organisasjonskart

3.2 Ansvarsforhold

Teamet er organisert etter vanlige Scrum rutiner.

3.2.1 Produkteier

Jonas Kirkemyr Fosen Utvikling:

- Representerer Fosen utviklings interesser i prosjektet og er hovedkontaktperson
- Skal være med på sprint planning meeting, i starten av hver sprint.

3.2.2 Prosjektveileder

Frode Haug NTNU:

- Veileder og rådfører gruppen.
- Har ukentlige møter med prosjektgruppen.

3.2.3 Prosjektleder / Scrum master

Jakob Fonstad:

- Har ansvar for at utviklingen går i riktig retning og oppgaver blir fullført innen angitt tid.
- Har ansvar for å sette møte agenda
- Har siste ordet der det oppstår uenigheter i gruppen.
- Vil i tillegg jobbe som backend utvikler med resten av gruppen.

3.2.4 Utviklere

Kristian Sundhaugen Martin Pukstad og Stian Fenstad:

- Lære seg opp i relevante fagfelt innenfor sine tildelte oppgaver
- Utføre sine oppgaver innen angitt tid.

3.3 Rutiner og regler i gruppa

Grupperegler er vedlagt rapporten i selve Bacheloroppgaven. Utdrag med viktige punkter fra gruppens regler:

- Det er satt faste arbeidstider fra 0900-1600 på skolen mandag til fredag. Det er åpent for at gruppemedlemmer skal kunne gå i forelesninger o.l i arbeidstiden.
- Det forventes at det arbeides ca 30 timer per uke per person. Arbeidstid samt hva en har jobbet med skal loggføres.
- Alle i gruppen plikter å møte til fastsatt arbeidstid, samt prosjektmøter, statusmøter og møter med oppdragsgiver og veileder.
- Det skal skrives referat etter alle fastsatte møter. Kristian Sundhaugen er referent.

4 Planlegging, oppfølging og rapportering

4.1 Hovedinndeling av prosjektet

Prosjektet består av to hoveddeler: Arduino brett med tilhørende sensorer og styringskomponenter og Raspberry Pi som sentralt kontrollpunkt. Sensorer kan for eksempel være temperatur-, lys-, bevegelses-komponenter som sender tilsvarende data til Raspberry Pien. Styringskomponenter vil være releene som må til for å kontrollere forskjellige elektriske applikasjoner. Raspberry Pi datamaskinen brukes for lagring av informasjon og videresende informasjonen til Fosen Utvikling sin nettside. Den skal også brukes til å sette opp det logiske regelverket for sensorer og styringskomponentene
Gruppen består av fire utviklere, hvor to vil ta for seg utvikling rundt Raspberry Pi, og to vil ta for seg oppsettet av Arduino brett, sensorer og styringsenheter. Disse to delene er likevel så avhengig av hverandre at utviklerene vil ha et tett sammarbeid underveis i prosjektet. Videre vil gruppen tilpasse seg eventuelle krav som dukker opp i prosjektet, enten fra oppdragsgiver eller som dukker opp organisk i utviklingen. Dette har vi tatt hensyn til i valg av utviklingsmodell.

4.2 Valg av utviklingsmodell

Utvikling av prosjektets sensorkomponenter må ha muligheten til å utvides, dermed vil utviklingen fortsette så lenge dette er et behov av produkteier. Det er også mulig at deler av hardwaren ikke fungerer som ønsket, det må derfor være mulig å tilpasse prosjektet til ny teknologi og krav.
Dette samsvarer da godt med smidige utviklingsmodeller, hvor utvikling gjennom iterasjoner til et ferdig produkt og aktiv inkludering av produkteier er i fokus. Og derfor har gruppen valgt å bruke den smidige utviklingsmodellen Scrum.[1]
Scrum som rammeverk sentrerer seg rundt inkrementelt og iterativt arbeid over perioder med utvikling, i et tett samarbeid med produkteier og selvstyring av team. Av den grunn er den godt egnet til prosjektet, ut i fra prosjektets rammer. Andre metoder, slikt som fossefallsmetoden ble derimot valgt bort, grunnet at kravspesifikasjonen og estimering av prosjektet vil måtte skje på et tidspunkt hvor gruppen har minst kunnskap om prosjektets rammer.[1]
Oppdragsgiver, også kjent som produkteier i Scrum, har tung kompetanse innenfor fagfeltet og ønsker derfor å ta stor del i prosjektet. Dette passer godt inn i Scrum, hvor produkteier er en aktiv del av planleggingsprosessen i hver sprint.[1]

4.2.1 Anvendelse av systemutviklingsmodell

Gruppen har planlagt én ukes Scrum sprinter, dette er på bakgrunn av at utviklingen av produktet vil foregå over en kort tidsperiode. Planen er å utvikle mindre deler av produktet ukentlig og fleksibelt kunne tilpasse funksjonalitet i henhold til hva som dukker opp. Dersom sprint backlogen skulle fylle seg med

sprinter hvor estimeringen overskrider én ukessprinter, vil gruppen sammen med produkteier ha samtaler om å gå over til to ukers sprinter og sette møter i henhold til dette.[1]
Ukessprinter vil starte med fastsatt møte med produkteier tirsdag kveld kl. 20:30. Dette er da et sprint planning meeting, hvor gruppen og produkteier bestemmer hva som skal være i fokus kommende sprint. Slike møter vil være satt opp og holdt kontroll over av prosjektets Scrum master, Jakob Fonstad. Møtene vil gi en indikasjon på hvordan utviklingen til neste iterasjon ligger an, samt planleggingsmøte for utvikling av ønsket funksjonalitet. Disse vil da bli user stories som legges i product backlog, slik at alle produkteiers ønskede funksjoner er samlet. Fra product backlog vil de mest relevante user stories bli valgt. User storiene legges inn i Release Backlog, der hver user story blir tildelt et estimat av tid, en rangering av produktets tyngde og hvilke utviklere som vil håndtere storien. Dette er så vi kan fordele det innad i sprinter, som legges i en sprint backlog.[1]
Estimeringsprosessen vil basere seg på tid gjennom fokus på timer, hvorav:

- User stories som tar mindre enn én dag legges i kategoriene:
  - Én, to, fire, eller åtte timer
  - Eks. Estimerer man tre timer arbeid rundes dette opp til fire timer.
- User stories som tar mer enn én dag legges i kategoriene:
  - To, tre, fem eller sju dager
  - Eks. Estimerer man fire dagers arbeid rundes dette opp til fem.

Antall timers arbeid utført på user stories noteres, for å danne et burndown chart. Burndown chart tillater gruppen en beregningen av burndown velocity, som gir en referanse om prosjektet treffer innenfor slutten av utviklingsperioden, satt den 1. mai 2018. Ved utgangen av hver sprint skal gruppen holde et sprint retrospective meeting, for diskusjon av fordeler og ulemper i gjennomført sprint. En sprint er anslått ferdig ved slutten av arbeidsdagen hver tirsdag. [1]
Gruppen vil ha daily Scrum møte, 15 minutter avsatt hver morgen fra mandag til fredag. Slik at alle i gruppen er klar over hvilket arbeid hver enkelt har utført, utfordringer som har blitt støtt på og hva som skal jobbes med videre. Dersom det oppdages utfordringer som skaper hindringer i utviklingen, vil gruppen informeres og håndtere dette i plenum før det blir problematisk.[1]

4.3 Plan for statusmøter og Beslutningspunkter

Gruppen vil ha ukentlige samtaler med arbeidsgiver og veileder, i tillegg skal gruppen holde daily Scrum møter og interne planleggingsmøter rundt ukens sprint hver onsdag morgen. Videre gir veileder ønske om statusrapporter underveis i prosjektet, disse leveres ved avtalt tidspunkt. Scrum master vil ha ansvar for møter med veileder og arbeidsgiver.
Interne møter vil foregå på grupperom, møte med veileder vil foregå på hans

kontor og møter med produkteier vil foregå over videokonferanse. Kristian Sundhaugen har ansvar som referent.
Beslutninger som kommer opp rundt valg i utviklingen av produktet vil bli tatt opp under møter med produkteier, slik at både gruppen og produkteier er enig om retningen og så det ikke oppstår misforståelser.
Der gruppen selv ikke kan løse problematikk rundt rapporten, settes det opp et møte med veileder. Veileder skal ha all nødvendig dokumentasjon på papir minst 24 timer før avtalt møtetidspunkt. Ved eventuelle uenigheter i gruppen skal prosjektleder, Jakob Fonstad, ta avgjørende beslutning om saken.

4.4 Metode

Gruppen anvender Google Docs for loggføringen av tid, referater og utkast versjoner av rapporten. Rapporten skrives inn i Sharenår et utkast fra Google Docs anses ferdigstilt. Disse verktøyene ble valgt på grunnlag av muligheten til enkelt dele dokumenter som redigeres, i tillegg til at det er mulig å laste ned backup lokalt.
Rundt prosjektstyring anvender gruppen Trello ved utførelsen av arbeidsoppgaver rundt rapport og det å sette seg inn i informasjon. Mens for anvending av Scrum vil verktøyet Jira bli brukt. Verktøyet ble valgt på grunnlag av tidligere erfaring med bruk av verktøyet, samt muligheten til å trekke ut statestikk om arbeidsflyten i gruppen.
For å lage figurer og illustrasjoner vil gruppen bruke Draw.IO eller Lucidchart, grunnet enkelt å bruke og ingen krav for betalt lisens.

5 Organisering av kvalitetssikring

5.1 Dokumentasjon, standardbruk og kildekode

Gruppen skal ta i bruk fagkunnskapen oppsamlet gjennom studietiden for organisering og utvikling i prosjektet.

5.1.1 Eclipse

Programvaren for Arduino sensorer utvikles i Eclipse. Det er ønskelig fra arbeidsgiver at det dannes et bibliotek for sensorene slik at det lettere kan videreutvikles etter endt bacheloroppgave. Derfor falt valget på Eclipse som utviklingsplattform for Arduino sensorene, da Eclipse allerede er godt utstyrt for å utvikle biblioteker, noe Arduino IDE ikke tilbyr.

5.1.2 SonarQube

For analyse av kodekvaliteten i prosjektet anvendes SonarQube. SonarQube støtter, blant annet, språk som C++, Java og Python, som alle anvendes i prosjektet. Målet med SonarQube er å unngå dårlig struktur, kompleks kode,

duplisering og mangel på dokumentering. SonarQube vil kunne brukes kontinuerlig under utvikling.

5.1.3 Doxygen

Doxygen er et verktøy for å strukturere kode dokumentasjon. Ved å standardisere kommentering av funksjoner, klasser og liknende vil opprettes det et HTML eller LaTeXdokument som vil fungere som dokumentasjon for koden. I tillegg til C++ støtter Doxygen også blant annet PHP, Java og C. Doxygen er veldig likt Javadocs, noe alle i gruppen er kjent med fra tidligere.

5.1.4 Testmiljø

Siden det er hardware som skal testes, er det nødvendig å sette opp et eget testmiljø i prosjektet. Dette for å teste at alle enheter kommuniserer og fungerer som tiltenkt, og for funksjonstesting underveis i utviklingen. Det planlegges også å sette opp en lokal webside for å vise sensordata, samt styring av reléer.

5.1.5 Dokumentering

- All kode lagres prosjektets repository. Gruppen har valgt å bruke Bitbucket som versjonskontrollsystem, da alle i gruppen har kjenskap til dette.
- Rapporten skrives i LaTeX, gjennom verktøyet ShareLaTeX. Dette dette gjør det mulig for gruppen å samskrive.
- Google Disk er valgt som lagringsområdet for alle dokumenter, f.eks møtereferater, kilder, grupperegler o.l viktige dokumenter.

5.2 Utviklingsrutiner

I Scrum skal prosjektet deles inn i mindre arbeidsoppgaver, som heter user stories, inn i backloggen. All utvikling skal skje på sin egen branch. Som vil si at hver gang en utvikler starter på en ny user story skal det også lages en ny branch, som merges med development når oppgaven er løst. Issues skal også rettes på en egen branch og merges med development når issuet er løst. I slutten av hver sprint skal development branchen merges med master. Alle commits skal klart markeres med user story og issues id. Denne arbeidsmetoden vil minske sannsynligheten for merge conflicts, og minimalisere risikoen for at det oppstår konflikter der utviklere jobber på samme sted i prosjektet.

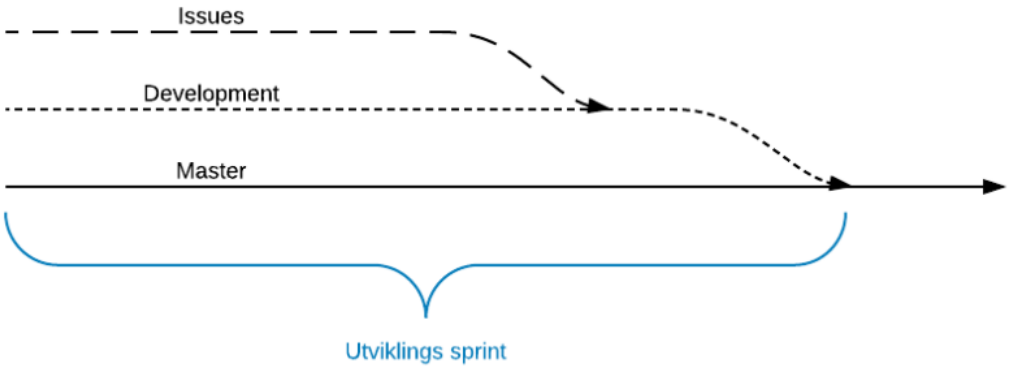


Figure 3: Utviklingsrutiner

Gruppen har faste rutiner på alt som committes.

- Hver committ skal starte med enten user story eller issue ID tag. For at det lett skal være mulig å følge historikken til user stories eller issues.
- Committer skal være korte og informative. Så det er lett for alle i gruppa å se hvilke endringer som har blitt utført.
- Det skal jevnlig committes opp mot sin respektive branch, for å unngå tap av kode.
- Alle funksjoner og klasser skal kommenteres etter standarden for Doxygen dokumentasjon.
  - I tillegg skal all kode kommenteres før den committes.
- Klasse, funksjon og variabelnavn skal følge de retningslinjene som er satt i prosjektet og for gjeldende kodespråk.

5.3 Risikoanalyse

Under kommer det en risikoanalyse tabell, som tar for seg problematikk som kan oppstå i prosjektet. Det er også opprettet tiltak der risikoanalysen viser det er nødvendig.

5.3.1 Risikotabell

Nummer	Risiko	Sannsynlighet	Konsekvens	Tiltak
1	Prosjektet blir ikke ferdig i tide. Dette kan skyldes mange forskjellige årsaker som teknologiske problemer, tap av kildekode eller feilvurdering av tidsbruk.	Usannsynlig	Kritisk	Ja
2	Store endringer i kravspesifikasjoner fra oppdragsgiver.	Sannsynlig	Uproblematisk	Ja
3	Én eller flere utviklere blir syke over en lengre periode eller slutter under prosjektet.	Usannsynlig	Kritisk	Ja
4	Tap av kildekode eller rapport.	Usannsynlig	Kritisk	Ja
5	Andre firmaer utvikler lignende løsninger.	Svært Sannsynlig	Uproblematisk	Nei
6	Fosen Utvikling skrinlegger prosjektet.	Usannsynlig	Problematisk	Nei
7	Gruppen får ikke tilgang til all hardware og software nødvendig for oppgaven.	Usannsynlig	Kritisk	Ja
8	Problemer med kommunikasjon mellom sensorer og sentralpunkt.	Sannsynlig	Problematisk	Ja

5.3.2 Plan for håndtering av risikoanalyse

Nr	Tiltak
1	Hvis gruppen støter på problemer med å levere til deadline, så må produkteier informeres. Etter at dette er gjort må det diskuteres hva som kan bli fjernet fra kravspesifikasjonen. For å redusere sannsynligheten for at prosjektet blir forsinket, holder gruppen seg til sprintene, for at utviklingen skal ligge likt med planen som er satt i Scrum. Planleggingen er viktig for å redusere risiko for forsinkelser.
2	Det er viktig med en god dialog med produkteier om utviklingens status, så han kan komme med innspill eller ønsker til nye user stories fortløpende. Jevne møter med produkteier, i gruppen internt og med veileder gir en minsker risikoen for at det dukker opp store endringer i kravspesifikasjonene som gruppen ikke kan håndtere. Det er også viktig å skrive gode referater fra alle møter og samtaler for å holde oversikt over endringer som kommer inn og hva som er fastsatt fra før, slik at det minker risikoen for misforståelser.
3	For at sykdom eller frafall ikke skal ha en stor innvirkning på prosjektet, holder gruppen daily Scrum møter hver dag og retrospective møte i slutten av hver sprint. Dette gjør at alle i gruppen til enhver tid er oppdatert på hva de andre jobber med, slik at de lett kan hoppe inn der frafall eller sykdom intreffer.
4	For å unngå tap av rapport blir den lagret både i ShareLaTeXog Google Drive, i tillegg skal det tas en lokal backup med jevne mellomrom. Dette minimerer risikoen for at noe går tapt da alle har flere kopier. Når det gjelder kildekode så ligger dette lagret på Bitbucket samt lokalt hos alle utviklerne.
7	For å minimere risiko for at gruppen ikke får nødvendig hardware eller software, varsles oppdragsgiver der gruppen ser det oppstår mangler. Gruppen har også fullmakt for innkjøp av utstyr.
8	Hvis kommunikasjon mellom sensorer og sentralpunkt svikter må dette gi ut en varsel til nettsiden som holder oversikt slik at det er mulig å ta tak i problemet. Det skal også legges inn failsafe funksjonalitet i både sensorer og sentralpunkt slik at hvis internett eller strøm forsvinner må det kjøre automatiske restart eller oppstart for å fikse seg selv om mulig.

5.4 Verktøyliste

5.4.1 Verktøy:

Arduino IDE

- Arduino sin egenutviklet kompilator. Denne har en del begrensninger, men er et verktøy for å sjekke at kode fungerer slik det er ønsket.

Bitbucket

- Samlet og delt kode innad i gruppen og til arbeidsgiver, har også tilkobling til Jira og Trello.

Eclipse

- Kompilator brukt for koding av Arduino Hardware. Eclipse blir tatt i bruk på grunn av muligheten til å bygge biblioteker selv for Arduino.
  - **Doxygen**
    Ganske likt JavaDocs, som er et dokumentasjonsverktøy for kode. All kommentar som blir skrevet over vår kode ender opp som et samlet dokument i Doxygen, med god oversikt.
  - **SonarQube**
    Verktøy for å holde oversikt over kodekvalitet. Varsler om bugs, rotete kode og forteller endringer som bør gjøres.

Google Drive

- Alle dokumenter lagret i skyløsningen Google Drive som rapport, bilder og planer.

IntelliJ

- Kompilator for koding av nettside som skal brukes som en del av et test-miljø.

Jira

- Verktøy brukt for å holde styr på smidig utvikling, i dette tilfellet Scrum.

Raspian OS

- Operativsystemet som blir kjørt på Raspberry Pi. Vil inneholde noen endringer for å fungere slik vi ønsker, alt vil bli rapportert.

ShareLaTeX

- Skriveprogram som følger LaTeXStandard.

TeamGantt

- Verktøy som forenkler tegning av Gannt skjema.

Trello

- Oversikt over user stories i utviklingsprosessen, og oppgaver som er ferdige.

5.4.2 Hardware:

220Ω Resistor

- Strøm resistor for en farges LED lys.

270Ω Resistor

- Strøm resistor for tre farges LED lys.

Arduino BroadBoard

- Oppkoblingsbrett nødvendig for prototyping av Arduino sensorer.

Arduino Development Board

- Miniatyrvariant av Arduino Uno. Denne skal kobles opp sammen med Arduino WiFi, YwRobot, YP-05 Code Transmitter for å lage et tilsvarende brett som Arduino Uno med WiFi.

Arduino Humidity/Temperature Sensor (DHT22)

- Sensor for måling av temperatur og fuktighet i et rom.

Arduino Light Sensor (LM393)

- Lys sensor for deteksjon av lysstyrke mellom 0(veldig lyst) og 1450(Veldig mørkt).

Arduino PIR Sensor

- Bevegelsessensor for deteksjon av bevegelse foran linse.

Arduino Touch Sensor (Touch Switch 1000)

- Knapp for styring av eventuelle sensorer eller styringsenheter i et hus.

Arduino Uno

- Hovedkortet til Arduino. Dette er en miniatyrmaskin og inneholder små mengder RAM og lagring. Brukes for prototyping av oppkobling og koding av sensorene.

Arduino WiFi

- WiFi enhet koblet til Arduino, for kommunikasjon mellom Arduino brettene og Raspberry PI.

Arduino YwRobot Powersupply

- Strømforsyning av Arduino WiFi.

Arduino YP-05 Code Transmitter

- Mellomledd mellom Arduino WiFi og Arduino Development Board for overføring av kode og data.

Raspberry Pi

- Egen liten maskin som skal ta inn all data fra sensorer og sende til en nettside for så å vise frem. Skal også senere bli brukt for sending av data tilbake fra nettside for styring. Skal være sentralpunktet i et hus.

6 Plan for gjennomføring

6.1 Gantt skjema (se til vedlegg 1.)

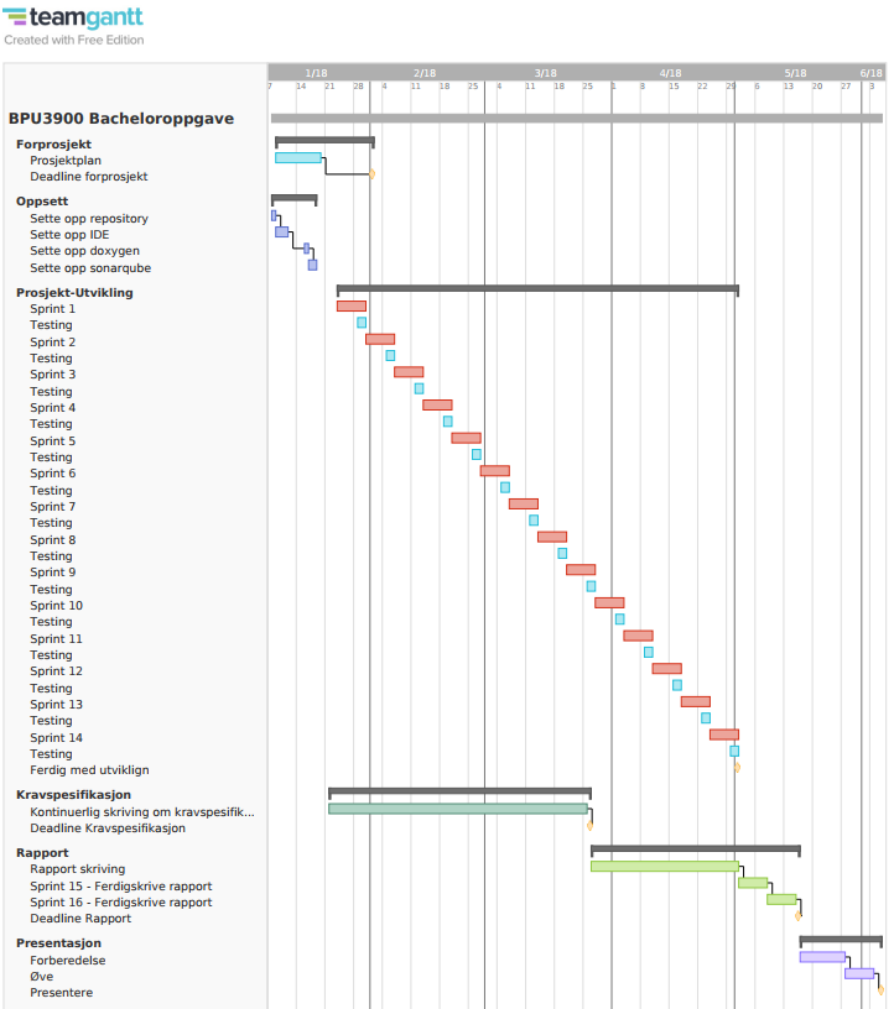


Figure 4: Gantt Skjema

6.1.1 Detaljer om Gantt-fremdriftsplan

Gruppens plan for januar er å få satt opp utviklingsmiljøer, repository og verktøyene for å holde kodekvaliteten høy. Videre vil arbeidet med forprosjektet foregå i tiden fra 9. januar til fristen for innlevering 1. februar, å levere forprosejktet vil være en milepæl for gruppen. Dette vil knyttes videre til oppstarten av neste periode, prosjektets utvikling.
Prosjektets utvikling estimeres å gå over 13 sprinter, dette er da mellom fastsatte tider forklart i punkt 4.2.1. Testingen av iterasjoner vil bli gjort ved slutten av hver sprint. Utviklingen ses å være ferdig 1. mai, slik at gruppen kan legge fokuset over til finpussing av rapport.
Ved siden av utvikling vil gruppen arbeide med rapporten. Det skal legges fokus på utvikling i starten, for så å gå over til mer fokus på rapporten underveis i prosjektet. Etter forprosjekt skal kravspesifikasjon være ferdigstilt til fristen 26. mars. Med denne arbeidsfordelingen planlegges det at gruppen skal være ferdig med rapporten innen fristen 16. mai.
Når rapporten er levert vil gruppen gå over til arbeid med presentasjon. Her vil gruppen jobbe med å presentere bachelor oppgaven og rapporten.

6.2 Liste over aktiviteter

6.2.1 Gruppens liste over aktiviteter

- Skrive prosjektplan
- Skrive kravspesifikasjon
- Skrive rapport
- Utvikle hovedspesifikasjon av prosjektet
- Holde fremføring av oppgaven

6.3 Milepæler og beslutningspunkter

6.3.1 Milepælene

Under er de milepælene gruppen har fastsatt og ønsker å ha som en plan for å fullføre oppgaver underveis i prosjektet. Dette gir oss fastsatte datoer og oppgaver å jobbe mot.

- **Milepæl 1, 30. januar:** Endelig versjon av prosjektplan fullført
  - Prosjektplan er gått igjennom flere utkast og samtaler med veileder, til en endelig versjon av prosjektplanen er ferdigstilt. Leveres 1. februar.
- **Milepæl 2, 26.mars:** Endelig versjon av kravspesifikasjon fullført

  - Kravspesifikasjon er gått igjennom flere utkast og samtaler med veileder og produkteier, til en endelig versjon av kravspesifikasjonen er ferdigstilt. 26.mars.

- **Milepæl 3, 16. mai:** Endelig versjon av rapport fullført
  - Rapport er gått igjennom flere utkast og samtaler med veileder, til en endelig versjon av rapporten er ferdigstilt. Leveres 16.mai.

- **Milepæl 4, 1. mai:** Utvikling av prosjektet er ferdig
  - Alle deler av prosjektets utvikling skal være ferdigstilt til denne datoen

6.3.2 Større milepæler for utviking

Videre vil prosjektet ha milepæler uten en fastsatt dato, dette er grunnet at estimatet for tid er uvisst, men det vil fremdeles være en stor milepæl å nå under utvikling.

- **Milepæl 5:** Fått på plass sensor komponentene som skal samle informasjon
  - Fått på plass sensor komponentene som skal samle informasjon, og deres tilhørende kode

- **Milepæl 6:** Kommunikasjon mellom sensorer og Raspbery Pi
  - Sensorer og Raspberry Pi kan kommunisere over nettverk, som vil si å sende data og motta styringer.

6.3.3 Besluttninger for prosjektet

Besluttninger gruppen har tatt rundt prosjektet for:

- **Utviklingsmiljø**
  - Eclipse Oxygen; for utvikling med C++
  - IntelliJ; for utvikling rundt webgrensesnitt

- **Utviklingsmodell**
  - Scrum; Se til punkt 4.1

- **Kode inspeksjon/dokumentasjon**
  - SonarQube
  - Doxygen

- **Tekstbehandling**
  - Google Docs
  - ShareLaTeX
- **Utviklingsplanlegger**
  - Jira; for å utvikle i Scrum
  - Trello
- **Hosting av kode**
  - Bitbucket
- **Git GUI**
  - SourceTree

For å lese mer om disse verktøyene, se 5.4.1 Verktøy.

References

[1] Sommerville IS. *Software Engineering 9th ed.* Boston: Pearson, 2011.

[2] James Vincent - The Verge. *Google uses DeepMind AI to cut data center energy bills.* URL: https://www.theverge.com/2016/7/21/12246258/google-deepmind-ai-data-center-cooling. (accessed: 24.01.2018).

[3] Wikipedia. *Clothes dryer.* URL: https://en.wikipedia.org/wiki/Clothes_dryer. (accessed: 24.01.2018).

[4] Wikipedia. *Dishwasher.* URL: https://en.wikipedia.org/wiki/Dishwasher. (accessed: 24.01.2018).

[5] Wikipedia. *Home Automation.* URL: https://en.wikipedia.org/wiki/Home_automation. (accessed: 24.01.2018).

[6] Wikipedia. *Sewing Machine.* URL: https://en.wikipedia.org/wiki/Sewing_machine. (accessed: 24.01.2018).

[7] Wikipedia. *X10 (industry standard).* URL: https://en.wikipedia.org/wiki/X10_(industry_standard). (accessed: 24.01.2018).

7 Ordliste:

MQTT - meldingsprotokoll mellom sensorer og enheter

NTNU

Norges teknisk-naturvitenskapelige universitet

Vår dato Vår referanse 110

H Prosjektavtale

Prosjektavtale

mellom NTNU Fakultet for informasjonsteknologi og elektroteknikk (IE) på Gjøvik (utdanningsinstitusjon), og

FOSEN UTVIKLING

(oppdragsgiver), og

MARTIN PUKSTAD, JAKOB FONSTAD,
KRISTIAN SUNDHAUGEN, STIAN FENSTAD

(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 01.01.2018 til 01.07.2018.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der NTNU IE på Gjøvik yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra NTNU å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra NTNU på Gjøvik. Studentene dekker utgifter for ferdigstillelse av prosjektmateriell.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. NTNU IE på Gjøvik står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av intern og ekstern sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Alle bacheloroppgaver som ikke er klausulert og hvor forfatteren(e) har gitt sitt samtykke til publisering, kan gjøres tilgjengelig via NTNUs institusjonelle arkiv hvis de har skriftlig karakter A, B eller C.

Tilgjengeliggjøring i det åpne arkivet forutsetter avtale om delvis overdragelse av opphavsrett, se «avtale om publisering» (jfr Lov om opphavsrett). Oppdragsgiver og veileder godtar slik offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og instituttleder/fagenhetsleder om de i løpet av prosjektet endrer syn på slik offentliggjøring.

Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode mv. som inngår som del av eller vedlegg til besvarelsen, kan vederlagsfritt benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av NTNU til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved NTNU og/eller studenter har interesser.

5. Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

6. Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer oppgavebesvarelsen med vedlegg (pdf) i NTNUs elektroniske eksamenssystem. I tillegg leveres ett eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med ett eksemplar til hver av partene. På vegne av NTNU, IE er det instituttleder/faggruppeleder som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og NTNU som regulerer nærmere forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene. Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale med oppdragsgiver, skjer dette uten NTNU som partner.

10. Når NTNU også opptrer som oppdragsgiver, trer NTNU inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene imellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

NTNUs veileder (navn): FRODE HAUG

Oppdragsgivers kontaktperson (navn): JONAS KIRKEMYR

Student(er) (signatur): Martin Pulstad dato 24.01.18

Julie Fonstad dato 24.01.18

dato 24.01.18

Kristian Sandhaugen dato 24.01.18

Oppdragsgiver (signatur): Jonas Kirkemyr dato _____

Signert avtale leveres digitalt i Blackboard, rom for bacheloroppgaven.
Godkjennes digitalt av instituttleder/faggruppeleder.

Om papirversjon med signatur er ønskelig, må papirversjon leveres til instituttet i tillegg.
Plass for evt sign:

Instituttleder/faggruppeleder (signatur): _____ dato _____

I Grupperegler

GruppeRegler - MySmartHome

Dette er en kontrakt for gruppe regler i Bachelor gruppen Mitt Smart Hjem. Den skal leses og undertegnes av gruppens medlemmer.

Regler:

1. DON'T BE A DOUCHEBAG
2. Fast arbeidstid på skolen er 0900 - 1600 (1300 i oppstartsperioden). Det forventes 6-7 timer arbeid i ukedager, til sammen ca 30 timer per uke per pers.
    a. Arbeidstid skal daglig føres i angitt excell ark(Finnes i google driven).
        i. Føring skal inneholde antall timer + stikkord for hva en har jobbet med.
    b. Hvis en ikke møter til angitt tid (10 min eller mer forsinkelse) straffes det med bot på 10,-. Evt bot penger går til noe felles.
3. Møter
    a. Det holdes møte etter avtale med veileder, per dags dato hver Torsdag kl 13.00.
        i. Referat skal skrives og legges inn i drive(Egen mappe for veileder møter "Referat->Veileder(Frode)->Referat Veileder").
    b. Hver mandag skal det holdes gruppemøte, hvor det diskuteres hva som har blitt gjort, og hva som skal gjøres.
        i. Referat skal skrives og legges inn i egen mappe i drive.
    c. Møte med oppdragsgiver er hver tirsdag kl 2030.
4. Det forventes at tildelte oppgaver er fullført til avtalt tid. Der det oppstår problemer skal en så tidlig som mulig informere om dette, så gruppen i en helhet kan hjelpe til, og ha et oversiktlig bilde av prosjektets status.
5. Ved faglige uenigheter skal de uenige partene få komme med sine argumenter, for å så la gruppen stemme på hvilken løsning man velger
    a. Kommer ikke gruppen til enighet må prosjektleder eller evt veileder ta avgjørelsen.
    b. Når en avgjørelse er tatt så må alle i gruppen, om uenig eller ikke, stå for avgjørelsen som ble tatt.
6. Hvis noen er misfornøyd med arbeidsinnsatsen til en annen i gruppen må dette tas opp så fort som mulig.

Ved alvorlige / gjentatte brudd på regler:

1. Problemet skal bli tatt opp som sak i gruppen. Gruppen må så komme til enighet om hva de kan gjøre for å løse problemet.
2. Hvis problemet ikke blir løst med diskusjon innad i gruppen. Skal prosjektleder gi en skriftlig advarsel(tidsfrister, oppgaver...etc) om regelbruddet, komme med forslag om hva som må til for å bedre problemet (“Bøte på skadene”), og komme med konkrete konsekvenser av manglende kravoppfyllelse. (evt i samarbeid med veileder).
3. Hvis ingen av øvrige punkter fikser problemet, skal det være samtale mellom hele gruppen og veileder. Så evt skriftlig ekskludering fra gruppen, i samråd med veileder.

J Original Oppgavebeskrivelse

Fosen Utvikling AS - http://fosen-utvikling.no/

MySmartHome

Fosen-Utvikling AS leverer web-løsninger til flere bedrifter som digital signace (Infoplakat), helpdesk og vikar system (DinVikar). Nå ønsker vi å utvikle et smarthus-konsept, som har mulighet til å kommunisere med vår eksisterende platform *MySmartHome*, et system for behandling av sensor data, og interasksjon med "embedded hardware".

Oppgragsgiver:	Fosen Utvikling AS
Kontaktperson:	Jonas Kirkemyr
Adresse:	Rådhusveien 18, 7100 Rissa
Telefon:	+47 992 57 386
Email:	jonas@fosen-utvikling.no

Vi ønsker å utvikle et nettverk av noder (sensorer og aktuatorer) som kan sende og motta instruksjoner fra en "gateway" enhet.

Oppgaven

Utvikle software for en gateway for å kommunisere med noder i et nettverk. Gatewayen skal kunne motta og sende data til hver node i nettverket. Gatewayen burde også kunne kommunisere med tredjeparts-api, som kan brukes til å propagere data fra dens nettverk, eller handle på mottatt data fra tredjepart. Mottatt data fra tredjepart kan igjen brukes for å handle på enheter i nettverket, IFTTT.

Utvikle software for et sett av sensorer/aktuatorer som kan respondere på innkommende kommandoer fra gateway, samt sende live-data til gateway om dens miljø.

Typiske sensorer kan være:

- Temperatur
- Fuktighet
- PIR
- Lys

Typiske aktuatorer kan være:

- Rele
- Lys
- Høytaler
- Motor

Under utvikling bør det tenkes at data fra en eller flere sensorer kan påvirke oppførsel til andre aktuatorer. Dette kan bla brukes til å spare på strømforbruket i et hjem, basert på mottatt sensor-data. Et slikt nettverk av noder har mange potensielle bruksområder, som også studentgruppen kan og bør utforske.

Fosen Utvikling AS - http://fosen-utvikling.no/

Hardware

Som gateway er det foreslått å bruke RaspberryPi. For nodene foreslås det å bruke Arduino. Oppgradsgiver vil stille med hardware for gateway og enkelte sensorer.

Programmering

Programmeringsspråk:

- Arduino – C/C++
- RaspberryPi – Python (C/C++)

Programmeringsoppgaven vil bestå i bla:

- Database mysql/postgres/mongodb
- Utvikle applikasjon på gateway, for å håndtere kommunikasjon med noder
- Hente og sende sensor data på noder
- Motta data fra gateway og håndtere tilgjengelige aktuatorer på noder
- Integrasjon med tredjeparts api
- Utvikle API for å kontrollere noder gjennom gateway

Studentgruppen vil gjennom prosjektet få erfaring på:

- Utvikling av embedded hardware
- Database
- C / C++ / Python

Fosen Utvikling AS vil bistå studentgruppen i gjennomføringen av oppgaven. Det vil bli gjennomført jevnlige møter online. Fosen Utvikling AS har fokus på agile utvikling, og bruker scrum som programmeringsmetodikk. Om ønskelig kan dette også brukes ved gjennomføring av bacheloroppgaven.

Fosen Utvikling AS bryr seg om open-source, og ønsker derfor å gjøre all utviklet kode tilgjengelig på github med MIT-lisensen.

Kontaktperson i Fosen Utvikling AS har ikke mulighet til å presantere oppgaven tirsdag 7.november.

Oppgave forbeholdt:

Jakob Fonstad, Stian Fenstad, Kristian Sundhaugen, Martin Pukstad

K Statusrapporter

Statusrapport

Martin Pukstad, Stian Fenstad, Kristian Sundhaugen og Jakob Fonstad

February 19, 2018

Contents

1 Status For

1.1 Planlegging

Vi har hatt en litt løs plan for hva vi skal gjøre og fulgt den godt. Vi har også fått litt råd fra oppdragsgiver om hvordan vi kan dele ting bedre opp i sprintene våre. Vil si vi i det store og hele har fint klart å sette opp sprintene våre og jobbe med dem. Vi kunne nok likevel blitt litt flinkere på å planlegge i detalj nivå, noe vi alt har diskutert i gruppen og er enige om å jobbe mer med.

1.2 Organisering av gruppens arbeids og ansvarsområder

Vi føler vi jobber godt sammen i gruppen, og har god kommunikasjon på problemer o.l som oppstår, både innad i gruppen og i utviklingen/prosjektet.

1.3 klargjøring av problemstilling

Gjennom samtaler med oppdragsgiver, og diskusjoner i gruppen, har vi fått bedre klarhet i akkuratt hva oppgaven går ut på, og hvordan vi skal løse den.

1.4 Løsningsmetode

1.5 Rapportskriving

Vi har dokumentert hva vi har arbeidet med, og notert evt diskusjoner og valg vi har tatt. Det har vært stort fokus på å få på plass backend biten av prosjektet vårt så vi kan komme i gang med den litt større delen av oppgaven. Derfor har ikke rapporten vært i noe stort fokus enda

2 Oppsummering for punktene over

Helhetlig har vi jobbet godt, og kommet et godt stykke på vei med utviklingen. Vi kunne nok ha jobbet litt mer med rapporten, men føler vi fortsatt er i rute der og.

3 Muligheter Trusler/Problemer

Vi har hatt problemer med å finne et fast sted (Grupperom). Det er en del hardware som må settes opp før vi kan begynne å jobbe, og det hadde vært fint å ha et fast sted hvor alt allerede var satt opp på forhånd, slik at vi raskere kunne kommet i gang med arbeidet.

4 Hva er avsluttet

Vi har fått på plass automatisk påkobling til wifi gjennom en wifimanager på alle arduino enheter. Alle sensorer sender data som JSON til MQTT Broker som deretter lagrer alt til en SQLite database. Kommunikasjon mellom sensorene og styringsenheten(Raspberry PI) er på plass og håndterer data sendt fra sensoren til styringsenheten, dette vil bli videreutviklet til å kunne sende data fra styringspunktet tilbake til sensoren. Prosjektplan er ferdig og levert. Grupperegler er satt og underskrevet av alle i gruppa. Når det kommer til koderyddighet har all kode som blir gjenbrukt blitt gjort om til biblioteker for enkelt bruk videre.

5 Hva er under arbeid

Videre blir det stor fokus på nøye planlegging av en webapplikasjon som skal fungere som brukerstyring av sensorene. Dette vil være hvor brukeren skal sette regelværk for sine enheter, for eksempel at en varmepumpe skal starte når temperaturen går under 24 grader.

6 Tidsfrister

Det har vært noen tekniske problemer som ikke har latt seg løse enda, som har gjort at noen user stories i sprintene våre har gått over lenger tid en først estimert.

7 Hva med motivasjon:

7.1 Overholdt

Så langt har vi klart å holde alle våre selvsatte frister når det kommer til både utvikling og rapportskriving. Det vil bli et større fokus på rapportskrivingen fremover da vi har virkelig kommet godt i gang med utviklingen.

7.2 Overskredet

Jakob har slitt en del med søvn siste ukene, og derfor ikke møtt til avtalt tid på morgenen (0900). Har likevel arbeidet på egenhånd.

7.3 Kritiske(”brenner det et blått lys”)

Så langt ligger vi godt an både i følge oss selv og oppdragsgiver. Fosen Utvikling har vært imponert over fremgangen og arbeidsmetodikken vår.

8 Hvordan oppleves veilederkontakt

Vi opplever veilederkontakten som svært tilfredstillende. Det at vi får tilbakemeldinger på kvaliteten på skrivingen vår gjør oss motivert til å skrive bedre. Vi går nøye gjennom hver linje flere ganger for å luke ut setninger som ikke er akademiske nok og får dermed et mye bedre resultat.

Statusrapport

Martin Pukstad, Stian Fenstad, Kristian Sundhaugen og Jakob Fonstad

April 23, 2018

Contents

1 Status For

1.1 Planlegging

Planen som ble lagt videre i resterende utviklingsperiode, ble at gruppen skulle videreutvikle produktet, med å legge ved en web løsning, med muligheter til å displaye data og muligens sende tilbake informasjon. Dette ble gjort i godt samarbeid med produkteier, hvor vi har snakket om mulige løsninger for webgrensesnitter, som vi valgte å videreutvikle, samt tanken om å gå fra én ukes sprinter til to uker, for at tidsperioden skulle strekke til.

1.2 Organisering av gruppens arbeids og ansvarsområder

Vi føler vi jobber godt sammen i gruppen, og har god kommunikasjon, om det er i person, over chat eller i Skype. Vi samkjører med bruken av Jira og litt Trello, så vi vet at arbeid ikke krysser.

1.3 Rapportskriving

Under denne delen av prosessen har vi også dokumentert underveis kilder og linker vi har hentet fra, slik at den informasjonen er samlet. Samt avsluttet vår utviklingsprosess den 17. april, og startet fokuset på å fullføre rapporten.

2 Oppsummering for punktene over

Helhetlig har vi jobbet godt, og kommet svært godt gjennom utviklingsprosess, mye lenger enn hva Fosen Utvikling forventet av oss. Vi har kommet i gang med rapporten og er i god rute for å levere innen fristen.

3 Mulige Trusler/Problemer

Kristian hadde i lengre tid mangel på pc, grunnet at hans laptop sitt batteri ikke fungerte lenger. I denne perioden ble han brukende en låne pc, som dessverre ikke klarte å kjøre noe IDE, og ble derfor sittende med fokus på rapport skriving.

Vi hadde noen problemer med integrering av webgrensesnitt mot Node.JS, dette gjorde at vi lagde en dummy versjon, for å kunne presentere at vår backend fungerer og at vi har produsert noe med design for frontend i tankene.

4 Hva er avsluttet

Videre fra hva som er avsluttet fra forrige status rapport, er vi ferdigstilt med utviklingsprosessen. Dette betyr at vi fikk på plass mulighet til å displaye data ut på et webgrensesnitt mottatt fra en Raspberry. Samt en form for å sette opp regelverk for sensorer, gjennom IFTTT metoden, men problemer med å få integrert dette sammen.

5 Hva er under arbeid

For øyeblikket er fokuset lagt over på rapportskriving, siden innlevering av det er gruppens neste frist. Vi ser også litt på mulighetene til å fortsette utviklingen smått, om vi har tid.

6 Tidsfrister

Vi har møtt på visse tekniske problemer underveis, som har gjort at ikke alle ønskede funksjoner fra vår side er på plass, men dette var selvsagt noe vi valgte å videreutvikle, siden vi nådde målet svært kjapt fra starten av.

7 Hva med motivasjon:

7.1 Overholdt

De fleste frister har blitt overholdt hele veien. Med noen unntak grunnet tekniske problemer og sykdom.

7.2 Overskredet

Alle i gruppen har hatt sykdomsproblemer i denne perioden, men har til en viss grad kunne stille allikevel.

7.3 Kritiske(”brenner det et blått lys”)

Av hva vi har hørt av oppdragsgiver, så har vi kommet mye lengre enn hva han forventet, noe vi tar godt til oss, med tanke på at han har høyt kvalifisert bakgrunn innen utvikling. Innlevering av rapporten nærmer seg, og er nok det vi har som hovedfokus, siden det alltid vil være noe å fikse på eller gjøre bedre der.

8 Hvordan oppleves veilederkontakt

I perioden som har vært har fokuset ligget stort på utvikling, så vi har ikke oppsøkt samtaler med veileder så stort, siden vi ikke har hatt mye rapport å vise til, men ellers har vi opplevd kontakten med veileder som svært tilfredsstillende. Det at vi får tilbakemeldinger på kvaliteten på skrivingen vår gjør oss motivert til å skrive bedre. Av det vi skriver går nøye gjennom hver linje flere ganger for å luke ut setninger som ikke er akademiske nok og får dermed et mye bedre resultat.

L Møte logg

Oversikt over møter med veileder

11.01.2018 - Generell informasjon rund bacheloroppgaven

Gruppen møtte med veileder for første gang og fikk introduksjon til rapport aspektet av bachleroppgaven, samt punkter som kan være viktig å huske på underveis i prosessen.

18.01.2018 - Gjennomgang av første utkast av prosjektplan

Gruppen leverte sitt første utkast av prosjektplanen og fikk tilbakemeldinger på denne fra veileder.

26.01.2018 - Gjennomgang av andre utkast av prosjektplan

Gruppen leverte sitt andre utkast av prosjektplanen og fikk tilbakemeldinger på denne fra veileder.

22.03.2018 - Gjennomgang av første utkast til kapittel 1 i rapport

Gruppen går gjennom første utkast av kapittel 1 fra rapporten med fokus på innledning og fange leseren, med veileder.

05.04.2018 - Gjennomgang av andre utkast til kapittel 1 i rapport

Gruppen går gjennom andre utkast av kapittel 1 fra rapporten, etter å ha rettet opp i ting etter tilbakemelding fra veileder.

30.04.2018 - Gjennomgang av første utkast til kapittel 2 i rapport

Gruppen går gjennom første utkast av kapittel 2 fra rapporten, rundt forarbeid med risikoanalyse og utviklingsmetodikk, med veileder.

11.05.2018 - Gjennomgang av utkast til kapittel 3 til 7 i rapport

Gruppen leverer resterende kapitler til siste veiledningsmøte med veileder. Går igjennom småpirk på hver av kapitlene, samt at kapittel 7 må ha mer innhold.

Oversikt over møter med oppdragsgiver

16.01.2018 - Sprint Planning

Første møte med oppdragsgiver. Gruppen introduserer seg selv og diskuterer hva oppdragsgiver forventer av oppgaven. Finner enighet om program vi skal bruke for videre kommunikasjon, fast setter ukentlige møter, snakker om hvilke verktøy som anbefales for prosessen, hvilke krav han setter gruppen og om han vil anbefale SCRUM, med tanker på oppgavens karakteristika.

06.02.2018 - Sprint 1

Samles til første møte etter sprint 1. Forteller at gruppen har levert prosjektplanen og diskuterer hva som er blitt gjort i Sprint 1. Snakker så videre om hva som er planen for

sprint 2.

13.02.2018 - Sprint 2

Møte for Sprint 2. Snakker om hva gruppen har gjort, spesielt rundt MQTT, oppsett av WiFi bibliotek, Raspberry og problemer gruppen har støtt på. Videre samtaler om hva gruppen skal gjøre i Sprint 3.

20.02.2018 - Sprint 3

Samlet for samtaler til Sprint 3. Gruppen forteller hva de har fått utgjort med sensorer og raspberry. Gruppen har kommet langt i utviklingen og spør om hva produkteier synes om at gruppen lager en webside. Snakker også om å øke sprinter til to uker. Videre samtaler om hva som skal skje i Sprint 4.

06.03.2018 - Sprint 4

Sprint 4 ble videre fokus på sensor, samt starte arbeidet med webside. Her har gruppen spørsmål om bruk av forskjellige rammeverk som kan anbefales til kommunikasjon mellom Raspberry og webserver, samt andre interessante rammeverk for utvikling av webside. Gruppen har også startet å skrive på rapporten. Samtaler holdes om hva som vil skje i Sprint 5.

20.03.2018 - Sprint 5

Snakket om Sprint 5, hvor webside med fokus på rammeverket Node.js var i fokus. Mulig funnet en løsing. Også blitt tipset om en metode kaldt If This Then That, som kan være interessant for logikk rundt rammeverk. Planlegger så Sprint 6.

04.04.2018 - Sprint 6

Snakket om hva som har skjedd i Sprint 6. Fått på plass sending av data mellom sensor, Rapsberry og webside. Støtt på problemer med regelverk. Og planlegger så Sprint 7.

17.04.2018 - Sprint 7

Samtaler rundt Sprint 7. Enighet om at utvikling nå er over, så fokus kan legges på rapport skriving. Støtte på problemer med integrering av regelverk, men nådde fleste mål ellers med sending av data fra sensorer, Raspberry til websider.

M Referat fra møter med veileder og oppdragsgiver

Referater fra møter med veileder

Referat 11.01.2018

kl. 10:00. Alle møtte.

Agenda
Generell informasjon om bacheloroppgave

Generell informasjon om bacheloroppgave

- Prosjekt avtale(Skal skrives under av student, arbeidsgiver og skole)
- Et dokument som alle på gruppen skal lese
- Sjekke PP på blackboard, bli kjent med kravene
- Skolen står utenfor om prosjektet velger å gå opensource. Skal kun ha koden til vurdering
- Oppgaven er ikke helt eksakt enda, mangler problemstilling og hva vi konkret skal utvikle
- Pass på at vi har nok omfang
- Ca. 600 timer per person
- Prosjektet gir muligheter til videreutvikling, men få på plass hva som skal lages og leveres i første omgang
- Lag grupperegler
- Gå til ooprog siden for å se gruppereglermal
- Ha fast at vi tar opp uenighet underveis
- Ikke la det ligge å bygge seg opp
- Leveringsfrist 1. februar om vi ikke hører annet. Skal sjekkes med Tom Røyse
- Prosjektavtale skal leveres på blackboard
- Innlevering av prosjektplan er fremdeles litt usikkert
- Omfang = Kap 1. i oppgaven
- Denne må skrives så den fanger leseren
- Lag som en trakt, start i det store tekniske og gå ned på spesifikt oppgaven
- Enkelt å lese(Mamma og pappa skal til og med klare å forstå)
- Gi en oppsummering av hva vi har utviklet
- Eks. Gå inn med “Dagens samfunn . . . ”
- Komme med eksempler
- Hva er typisk
- Hva skal vi se på
- Kravspek = Kap. 2
- Samle metoden vi bruker
- Utviklingsmodell
- Fossefall eller Scrum er interessant

- Kan eventuelt kjøre fossefall først, for så å gå over til scrum for videreutvikling
- Står på hvor mye vi vet vi skal levere om prosjektet, må da ha spesifikt hva oppgaven vår er å levere.
- Timelogg
- Se til gamle bacheloroppgaver hvordan de har lagd eller hva timelogger de har brukt
- Regneark fungerer veldig bra
- Skal inneholde: Dato, Navn, Timer, Hva man har gjort den dagen (2 linjer oppsumering)
- Møtereffereat
- Stikkord
- Tidspunkt, dato
- Hvem møtte
- Statusrapport 20. februar, kommer to til etter
- Frode har satt av fast 1 time hver uke
- Torsdag kl. 1300
- Prosjektet må deles i effektmål og resultatmål
- Effektmål: hva oppnår arbeidsgiver
- Resultatmål: hva leverer vi
- Teknisk rammer
- Hvilke verktøy bruker vi
- Føringer
- Hva bes av arbeidsgiver
- Se til bacheloroppgaver fra tidligere
- V 16 SmartHus Brekke, Flaten, Hukaas
- Lag et Gantt skjema
- Milepæler og beslutningspunkter
- Milepæl
- eks. levert prosjektplan
- 2 til 5 punkter
- Beslutningspunkter
- hva vi har bestemt
- eks. hvilke utviklingsmiljø vi skal bruke
- 2 til 5 punkter
- Lag prosjektplan nå så fort som mulig
- Mål og rammer kommer senere
- Ikke spør Frode om tekniske biter til prosjektet
- Pass på å dokumentere hele tiden under prosjektet. Spesielt hvor kode hentes ifra. Hva som er vårt.

Referat 18.01.2018

Alle møtte.

Agenda
Kommentarer til utkast av prosjektplan
Annet

Kommentarer til utkast av prosjektplan

- Vær obs på bruken av muntlig skrift
- Klarer hvem som helst å forstå hva vi forteller om i introduksjonen vår?
- Utdyp mer ikke skriv for tåkete
- Hva kan noen gjøre med produktet vårt?
- Rapport skal skrives i tredjeperson
- Fikse så innrykk gjøres riktig i Latex
- Det var teknisk kjedlig å lese om fagområdet
    - Fortell om ting i folks hverdag
    - Digitalisering av hverdagen
    - Typiske eks. på internett i hverdagen
- Avgrensning må være lenger (Hva skal vi drive med?)
- Få mere trakt, mindre teknisk i begynnelsen. Fang lesern.
- Begrunn valgene vi tar
    - Eks. fortell mer utfyllende hvorfor vi valgte Scrum, istendenfor andre utviklingsmodeller.
    - Her blir videreutvikling av mobilapplikasjon for stort, tenk smått, på komponenter
- Alle tall mellom 1 og 10 skal skrives som tekst
- Være mer konsekvent i skrivemåten rundt navn på verktøy og hardwear
- Konfigurasjonsstyring (BitBucket)
- Risikoanalyse (Med farger, grønn, gul, rød)
- Verktøy og Hardware
- Fiks Gantt diagram
- Milepæl (Noe vi er ferdig med, ikke noe vi starter med)
- Kravspek (På plass gjennom SCRUM)

Annet

- Må passe på at det ikke blir for lite å gjøre på gruppen rundt utvikling
- Ny møtetid 14:45 Torsdag
- Må ha gruppereglene skriftlig
- Prosjektavtale skannes inn på blackboard
- prosjektplan siste versjon digitalt
- Scrum Master rollen har mer fokus på utvikling enn prosjektleder rollen

Referat 26.1.2018

Martin, Stian og Kristian møtt

Agenda
Prosjektplan andre utkast

Prosjektplan andre utkast

- Litt muntlig fremdeles
- Pass på bruken av superativer
- Pass på bruken av stor forbokstav
- For mye X10 i starten av setningene
- Si mer nøye at Fosen setter opp opensource, ble litt tåkete
- Fortell hva en schematic er
- Fjerne blanke hopp for avsnitt, kjør heller /newline for nytt avsnitt

Referat 22.03.2018

Alle Møtt

Agenda
Gjennomgang av Kap. 1

Gjennomgang av Kap. 1

- Bakgrun skal heller stå som Forrord
    - Se til Lynkurs
- Ikke tydlig hva problemområdet er
    - Mangler introen (Halv side til èn)
- 1.3 til 1.5
    - Burde kanskje heller puttes inn under kravspek
- Skriv hvor Fosen ligger
    - Presenter først så takk dem.

Referat 05.04.2018

Alle Møtt

Agenda
Gjennomgang av kap. 1 utkast
Forventet til neste ukes møte

Gjennomgang av kap. 1 utkast

- Ha men at veileder er del av både arbeidsprossessen og rapporten i kapittel 1
- Ikke for personlig (Vi, jeg osv.)
- Innledning
    - Definere smarthus èn gang i 1.1
    - Standarisering av smarthus 1.2
    - Flytt om på innledningen slik at det har flyt
    - Er det nødvendig å fortelle om x10 så tidlig? Blir for avansert?
    - Hvilket område innen smarthus er det vi skal lage.
- Få frem formål med oppgaven

Forventet til neste ukes møte

- Hvor ser Veileder for seg at vi skal være neste ukene
    - Kravspek
    - Design
    - Koding

Referat 30.04.2018

Martin, Kristian og Stian møtt

Agenda
Gjennomgå utkast for Kap. 2
Annet

Gjennomgå utkast for Kap. 2

- Ikke så normalt å ha med Forarbeid i et bachelorprosjekt, men kapittelet ser godt nok ut.
- Noe muntlig og skrivefeil i visse deler av teksten, veileder har streket under disse stedene.
- Pass på sideombrekking til innleveringen
    - Forkort tekst om mulig
- Ord på norsk som er likeverdig på engelsk, kan skrives
    - Eks. Tastatur istedenfor keyboard
- Fredag 11. kl. 11 er siste møte
    - Innlevering av utkast vil da senest kunne leveres torsdag 10. kl 11.

Annet

- Veileder har ikke lest kapittel 1 igjen, grunnet det ikke var lagt ved forrige kopi og at det ikke var noen forandringer å se til.
    - Ser da kap. 1 som klart

Referat 11.05.2018

Alle møtt

Agenda
Gjennomgang av kap 3 - 7
Annet

Gjennomgang av kap 3 - 7

- Alltid ny side for nytt kap
- Hvor mange use cases?
    - Greit nok med overordnede
- Set usecase i samme rekkefølge som de detaljerte kommer
- 4.1.3 hva var andre "valgene"
- Nettside må frem på figurer
- Legg inn tekst over figurer
- Ikke punktum bak figur . . . .
- Fortell mer om hva som var utfordrende med å finne kode sjekker
- kap. 7
    - Henvis til tidligere diskusjoner i kap . . .
- 7.2
    - Henvis til prosjektplan
- 7.5
    - Nettside
- 7.6
    - "Mangler ønsket funksjonalitet" nettside

Annet

- Hvordan tar med timeloggen?
    - Hvis én måned går på hver side så går det bra
- Send mail til Tom om spørsmål rundt personas og use case

Referater fra møter med oppdragsgiver

Referat 16.01.2018 Sprint Planning

kl. 20:30 Alle møtte

Agenda
Hvilket program skal brukes for møter?
Hvilke verktøy for utviklingsprosess som anbefales
Vise frem timeliste som er blitt lagd
SCRUM
Hvilke krav av språk/ rammeverk har vi?
Hva ønsker produkteier at vi starter med utviklingsmessig?
Når skal videre møter holdes?
Annet

Hvilket program skal brukes for møter?

- Har brukt appear.in for samtale nå, men vurderer å bytte til noe annet så alle kan delta
- Skype?

Hvilke verktøy for utviklingsprosess som anbefales

- Snakker om bruken av trello, og valget om vi ønsker å kjøpe Jira for å kjøre scrum
- Produkteier sier at han har holdt seg til Trello og Asana. Ellers har Microsoft TFS som er en git sharing av kode. Mulighet til å dele opp i mye user stories, statistikk og lignende.

Vise frem timeliste som er blitt lagd

- Vist Produkteier timelog, loggbok og statistikken laget rundt; Produkteier synes dette så veldig bra ut.

SCRUM/Når skal videre møter holdes?

- Planlegger å sette opp User Stories med estimater gjennom planning poker og times estimat
- Tenker at Produkteier skal være dele starten på nye sprinter, når dette blir finner vi videre ut av.
- I utgangspunktet holder vi møter med Produkteier til kl. 20:30 tirsdager så langt det lar seg gjøre.

Hvilke krav av språk/ rammeverk har vi?

- Vi står fritt frem til å løse hvordan vi best finner ut av det selv når det kommer til språk og rammeverk, men ser for seg at vi skal skrive mye i C++.
- Vi har fritt fram på js rammeverk.
    - Node.js
    - Reackt.js
    - Angular.js

Hva ønsker produkteier at vi starter med utviklingsmessig?

- Først er det å hente data fra sensor og sende activators
- Hvordan motta og sende informasjon på rasp pi; Bruke rest api, sende hva vi kan forholde oss til
- Sette opp en server vi kan sende dette imot
- Api server skal opp og hvilke endepunkter vi kan forholde oss til
- Mest interessert i at vi skal fokusere på sensorene og automatikken rundt det.
- Få til et samspill mellom de forskjellige enhetene som skal styres av Raspberry pi'en
    - Enheter skal være dumme, bare mota kommando
- Raspberry Pi skal ha logikk for å hente ut informasjon om en enhet trigges
- Mest mulig konfigurerbart/styrbart av en bruker
- Starte med statiske regler for hvordan komponenter reagerer
- Flere enheter sammmen kan påvike mange enheter
- Hva som kan være fornuftige hendelser i eks. hus eller forretningsbygg
- Man kan skrive om caser hva man kan fortsette med i prosjektet
    - Ai med maskinlæring
    - Mobil applikasjon
- Planen er å sette opp dette på et kryptert wifi nettverk vi kan teste dette på.
- Så lenge Raspberry Pien kan ta imot kommandoer fra en utenforstående part, så er mye gjort.
- Store deler å konfigurere Raspberryen og Web kommunikasjonen
- Mqtt
    - Kan vi se til maskin til maskin kommunikasjon
- Kan skrive at eventuelt det å bruke WIFI på Raspberry Pi'en var helt unyttig, og at vi fant en bedre måte
- Kodestandard forventes kommentert og lettleselig
- Jobb med å få opp enhetene
    - Satt opp Raspberry Pi'en
    - Satt opp informasjonsending til Raspberry Pi
    - Få satt kommunikasjon mellom enhet og Raspberry Pi'en
    - Definere hvordan enheter skal snakke sammen
- Sammenkoblede sensorer; skriv gjerne om det, men enklest å forholde seg til en sensor på en enhet.

Annet

- Produkteier får invite til vår Google Drive gjennom hans email.
- Produkteier Jonas og Terje forventer å stille på presentasjonen
- Bare kontakte om det skulle være noe.

Referat 06.02.2018 Sprint 1

kl. 20:30 Alle møtte

Agenda
Fortelle at gruppen har levert prosjektplan
Snakke om hva som er gjort i Sprint 1
Hva er planen for Sprint 2
Annet

Fortelle at gruppen har levert prosjektplan

- Snakket om at gruppen har skrevet om prosjektplan fra møte med veileder og at denne er nå levert

Snakke om hva som er gjort i Sprint 1

- Satt opp våre første sensorer. Fått opp lys, touch, tempratur humidity, bevegelse.
- Bestemte oss for å sende data som JSON til database.
- Satt opp Raspberry og satt opp webserver som kan lagre data i database
- Kan sende sensor data lokalt over kabel mellom Sensor og Raspberry, da vet vi at JSON format fungerer

Hva er planen for Sprint 2

- Se på en mulig WiFi løsning, slik at sensor kan sende JSON data til Raspberry
- Sette opp schematics for WiFi løsning

Annet

- Fortalt at gruppen har kjøpt en egen ruter som vi skal koble raspberryen opp imot for testing på eget nettverk
- Skal sende invitering for produkteier til Jira prosjektet gruppen bruker, slik at han kan følge user storiesene nøyere
- Må settes opp sensor tar imot komandoer som skrur lys av og på
- Styringspunkt vil være på et grensesnitt fra raspberry, samt et tredjeparts løsning gjennom nettside

Referat 13.02.2018 Sprint 2

kl. 20:30 Alle møtte

Agenda
Hva har vi gjort i Sprint 2
MQTT
WiFi bibliotek
Raspberry
Problemer
Hva er planen for Sprint 3
Annet

Hva har vi gjort i Sprint 2
MQTT

- Mqtt automatisk skal være i nærheten av fullført. Hvor ESP har visse problemer med å av og til ikke funke, ukjent grunn
- MQTT broker klarer å ta imot alt av topics som sendes, trenger ikke å subscribe, tar imot alt av datapakker som sendes.
- MQTT kanaler går på id'ene.

WiFi bibliotek

- Fant et eget WiFi bibliotek lagd for automatisk oppkobling av Arduino Sensorer.
- Har fått Temperature og Humidity sensor til å fungere med dette.
- Wifi Manager skal klare å gi sensorer en kobling mot et ønsket nettverk dynamisk. Slik at man ikke trenger å hardkode inn ønsked SSID og passord

Raspberry

- Rasspery klarer å ta imot meldinger fra wifi og sende de som json data og sende de til database
    - Mangler opp å sette opp for resten av sensorer sin data
    - Pytonscript må kunne parse disse dataene

Problemer

- Rasspberryen sletter vlan interfacet når den kobler seg til et nytt nettverk.
    - Kan være pyton manager roter med rasspberry pi 3 sin wifi
- Leverer ikke nok strøm gjennom yp-05 brettet til dh22

Hva er planen for Sprint 3

- Finn ut hva som feiler dh22
- Del opp de oppgavene vi har jobbet med i mindre grupper, mindre estimeringer
- Flytt mys 19 til minor
- Sette opp databasemodell
- Lage schematics for hvordan sensorer fungerer

Annet

- Sette opp flere subtasks underveis
- Ha forskjellige mapper for koden, så eks. mappe for sensor, library, pyton osv.
- Development board er en større utgave av den mindre biten igjen.
- Kompilerte koden for dev board kan direkte overføres til det “lille” brettet.
- Produkteier mener at vi er på rett vei innen 1. sprint, kommunikasjon og tildeling av kanaler er viktigste
- Av sikkerhetsmessige grunner kan det være smart å ha et eget nettverk for sensorer.
- Spørre Elektro Ingeniører om de kan lage schematic for brettene våre rundt ferdig utvikla sensorer

Referat 20.02.2018 Sprint 3

kl. 20:30 Alle møtte

Agenda
Snakke om Sprint 3
Sensor
Raspberry
Spørsmål om å lage webside
Øke lengde på sprint til to uker
Hva skal skje i Sprint 4
Annet

Snakke om Sprint 3
Sensor

- Ser på Jira, snakker rundt problemet med at lys sensor ikke vil la oss overføre kode
    - Ikke avhengig av touch sensor og pir sensor for bachelor
- Lagd schematics for hvordan sensoren settes opp for å få tempratur informasjon, samt sende denne til Raspberry

Spørsmål om å lage webside

- Produkteier sier "Peis på" rundt webside

Hva er nødvendig til en webside?

- Gruppen forteller at de ønsker å sette opp en webside, hvor hver raspberry fungerer som server
- Regelverk vil lagres lokalt på raspberry
- Backend viktigst, design brukervennlig, men ikke like viktig i første omgang
- Produkteier tenker: Rasp skal kunne motta og sende data fra en tredjepart
- Virker som det finnes mye informasjon rundt oppsett, så skal ikke være vanskelig å jobbe rundt

Øke lengde på sprint til to uker

- Sprinter på en uke er for korte, ser vi støter på problemer med å fullføre innenfor tiden.
- Produkteier er enig at Sprinter kan økes til to uker og møter holdes deretter
- Snakket om at de user storiesene vi har i to do er blokkert av samme problemene som med lys sensor
- Flytter over alle user stories som ikke er gjort, over til neste sprint
- Identifisering er ikke så viktig, viktigere å heller få funksjonalitet på plass, men senere er nødt å legges til

Hva skal skje i Sprint 4

- Starte på å se på løsninger rundt webside
- Finne en måte å sende inforamsjon fra backend Raspberry til webside

- Se på React og Node
- Se om gruppen kan sette opp flere sensorer, eller om fokus kun skal legges på Tempratur
- Finn en metode som er god for oppsett av regelverk

Annet

- Produkteier er ekstremt tilfreds med arbeid så langt

Referat 06.03.2018 Sprint 4

kl. 20:30 Alle møtte

Agenda
Snakke om Sprint 4
Sensor
Webside
Node.js
React.js
Rapportskriving
Hva skal skje i sprint 5
Annet

Snakke om Sprint 4
Sensor

- Gruppen snakket med produkteier om at de har møtt på Hardware problemer rundt oppsett av andre sensorer enn Tempratur og Humidity. Produkteier mener at gruppen skal stoppe fokus på dette å heller legge fokus helt over på å lage fungerende kommunikasjon fra en Raspberry til en Webside.

Webside

- Gruppen har brukt mye tid på å sette seg inn i nye rammeverk. Siden de verken har jobbet med React eller Node før, har det tatt tid å gjøre seg kjent med.
- La brukerfunksjonalitet ligge å lag noe som fungerer heller

Node.js

- Så langt ser gruppen god mulighet til å bruke Node til å sende data fra Raspberry til en webside, dette vil da forsøkes å gjøres på JSON format.
- Hvordan det sendes er fortsatt noe gruppen forsøker å løse

React.js

- Gruppen ser på mulighet til å bruke React for å gjøre websiden til singelpage, siden det forventes mye sidebytter når gruppen skal starte på regelverk logikk, men legger ikke mye fokus i dette, siden det ikke vil være en viktig del av funksjonalitet.

Hva skal skje i sprint 5

- Fortsette å se på muligheter til å sende informasjon til webside
- Lese videre opp på om React er interessant å bruke
- Sjekk ut “If This Then That” metoden for regelverk

Rapportskriving

- Rapportskriving er påbegynt, som gjør at fokus legges litt mindre på utvikling

Annet

- Å øke lengde på Sprint har fungert svært godt for gruppen.

Referat 20.03.2018 Sprint 5

kl. 20:30 Alle møtte

Agenda
Snakke om Sprint 5
Webside
Node.js
React.js
IFTTT
Planen for Sprint 6

Snakke om Sprint 5
Webside

- Gruppen har startet å finne en løsning til å sende informasjon til Websiden.
- Det vil ta i bruk Express.js, Node.js og Socket.io

Node.js

- Gruppen tror de kan sende informasjon til websiden gjennom å koble seg til en socket på en webserver. Dette gjøres da i gjennom bruken av Socket.io, som da har setter opp en kanal for styringspunkt å sende informasjon til webserver gjennom.
- Må lese videre på Socket.io

React.js

- Gruppen har valgt å gå bort fra bruken av React, grunnet tiden det ville ta å gjøre seg kjent med.

IFTTT

- Gruppen ser god mulighet til å kunne bruke IFTTT til å lage regelverk for hvordan sensorer skal reagere.
- Produkteier sier at gruppen skal fokusere kun på brukergrensesnitt for Tempratur sensor, og muligheten til å sette opp et regelverk for denne.

Planen for Sprint 6

- Jobbe videre med Node.js og gjøre oss bedre kjent med Socket.io
- Starte å lage skisser for hvordan en Brukergrensesnitt løsning vil se ut med IFTTT

Referat 04.04.2018 Sprint 6

kl. 20:30 Alle møtte

Agenda
Snakke om Sprint 6
Sending av data
Regelverk
Plan for Sprint 7
Annet

Snakke om Sprint 5
Sending av data

- Gruppen har nå fått på plass så data kan sendes fra sensor til en webside gjennom bruken av Node.js, express.js og socket.io
- Dataen for en spesifikk tempratur sensor kan ses på en "Dummy" nettside gruppen har lagd for midlertidig bruk
- Ser nå på muligheter til å lagre en brukers data i en Database
- Produkteier anbefaler å se på en SQLit3 database, grunnet at det er små mengder data som skal lagres.

Regelverk

- Gruppen har satt opp skisser for hvordan en webside kan se ut, Produkteier synes dette ser fornuftig ut.
- Det er startet arbeid med å lage disse skissene i realtitet, men har støtt på problemer mellom logikken til IFTTT og hvordan det må skrives i sammsvar med express.js.

Plan for Sprint 7

- Fortsette arbeid med lagring av data, som spesifikke sensorer, raspberrys, data som kommer inn og lignende.
- Fortsett arbeid med brukergrensesnitt for regelverk. Se om det kan integreres med express.js, om ikke lag en simpel versjon for midlertidig bruk.
- Jobb så langt det går, men ikke la det gå utover andre fag. Gruppen har kommet meget lengre enn forventet

Annet

- Gruppen jobber med prosjekt i mobilutvikling, dette krasjer en del med siste Sprint og gjør at gruppen er usikker hvor mye mer som rekker å bli gjort.

Referat 17.04.2018 Sprint 7

kl. 20:30 Alle møtte

Agenda
Snakke om Sprint 7
Database
Regelverk
Ferdig med utvikling
Annet

Snakke om Sprint 7
Database

- Gruppen har fått på plass SQLit3 database til lagring av forskjellig informasjon som sendes om sensorer, raspberry og informasjon fra sensorer.
- Dummy siden lister nå ut sensorer hentet fra database for en bruker
- Database Logikken er også skrevet
- Satt inn logikk for Regelverk for temperatur som lagres i database, men usikkert om denne vil tas i bruk

Regelverk

- Fullført siden for å sette opp regelverk, men klarte ikke å integrere dette med express.js sitt oppsett i tide, så det mer detaljerte brukergrensesnittet for å vise data og lage regelverk fungerer foreløpig ikke med det logikken til backend.
- Lagd en dummy side av regelverk for å sende en ferdig lagd regelverk til DB og se om vi får tid til å fullføre å sende informasjon tilbake igjen til reeler

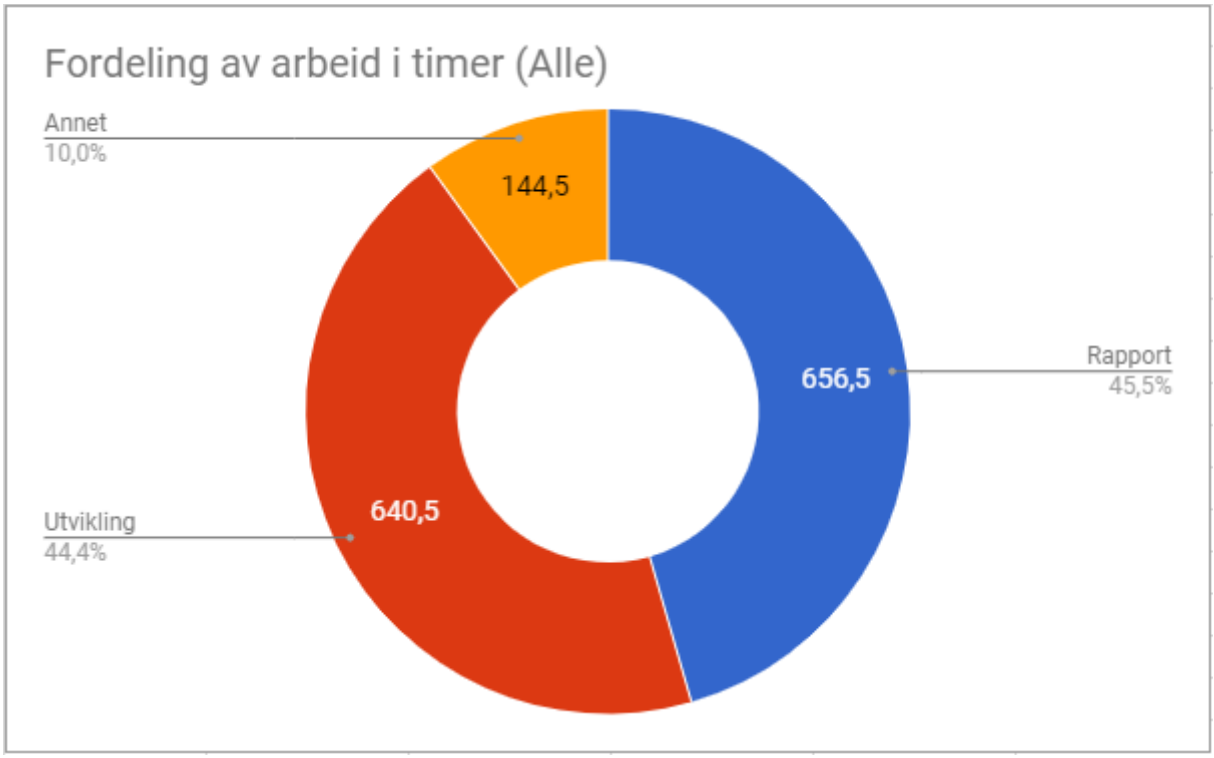
Ferdig med utvikling

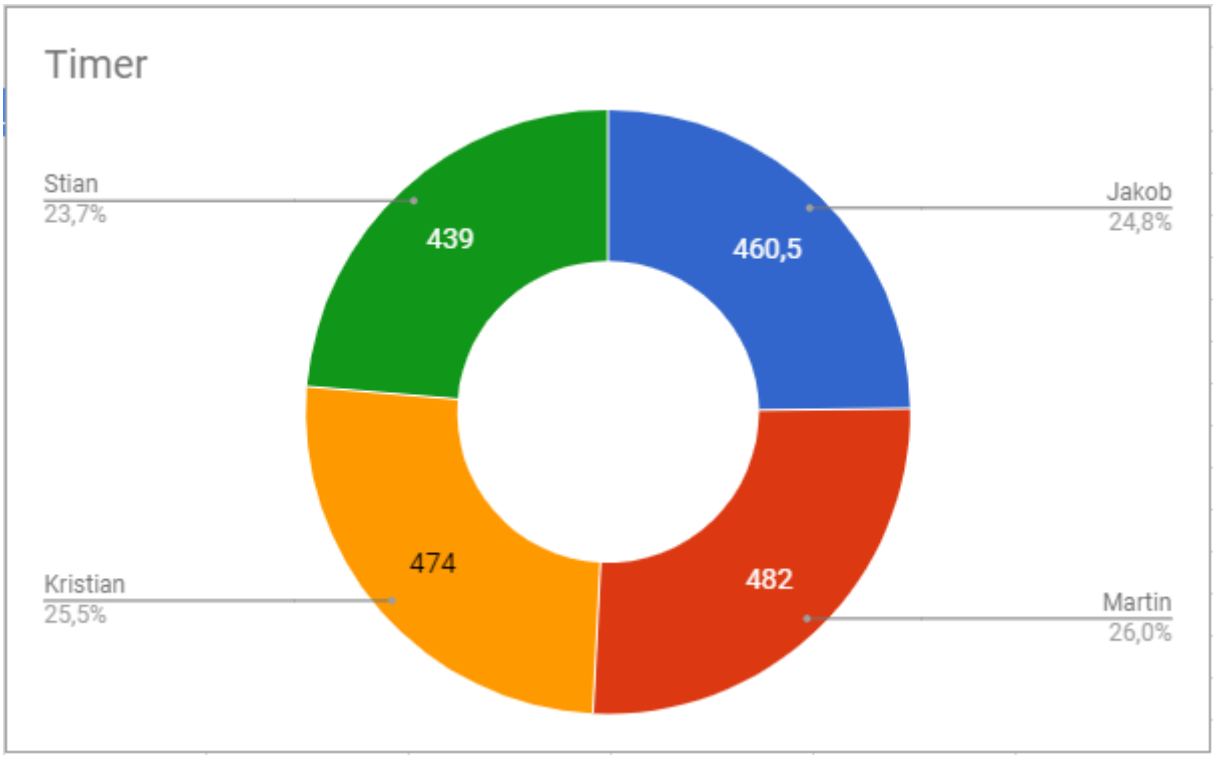
- Gruppen ser seg nå ferdig med utvikling, siden tiden før innlevering av resterende rapport ikke er stor, samt et stort prosjekt i Mobil Utvikling som må gjøres.
- Takker for samarbeidet og ser frem til å kunne delta på produktet videre under Open Source.

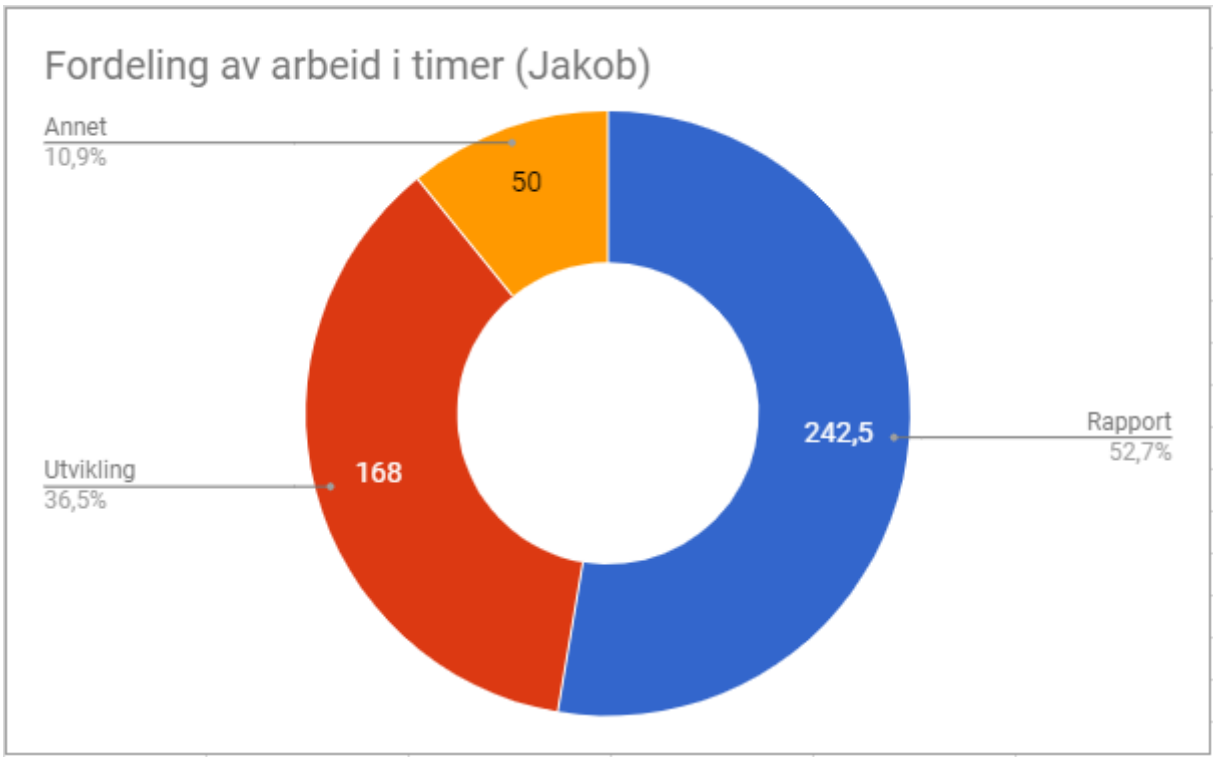
Annet

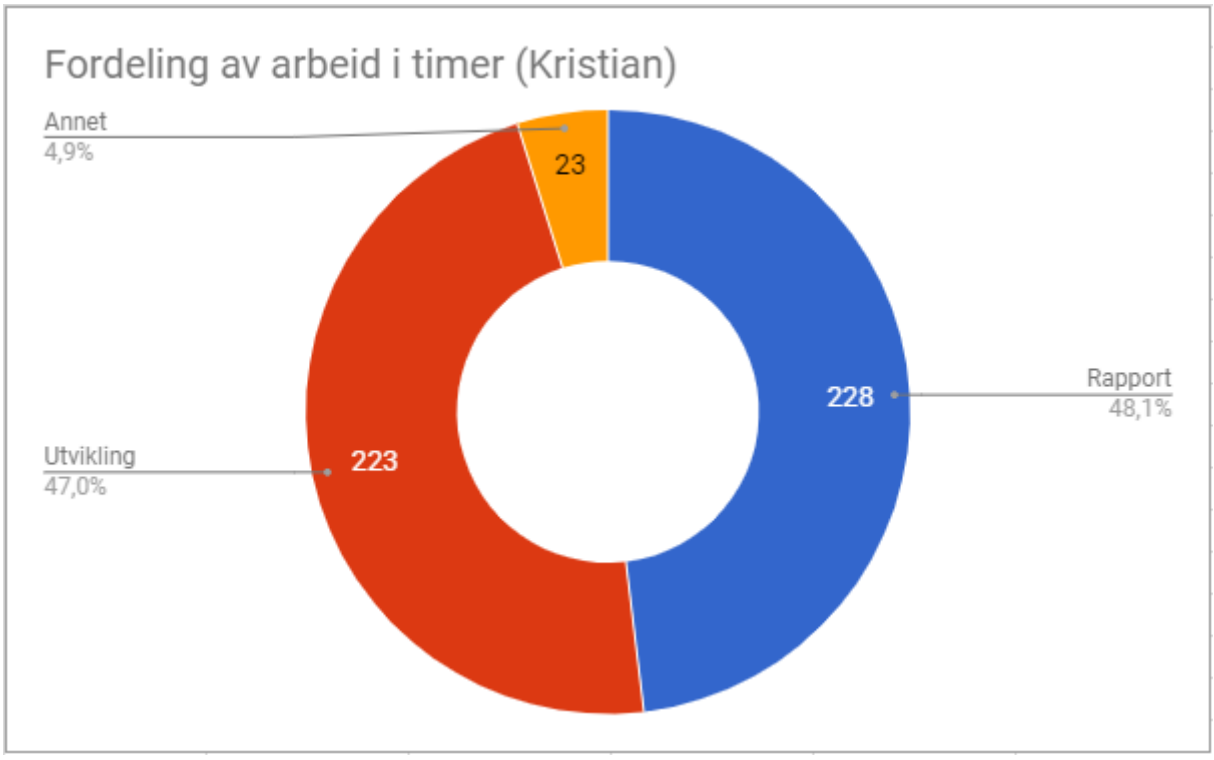
- Ble litt mindre gjort enn ønsket, grunnet stort prosjekt i Mobil Utvikling med kort frist.

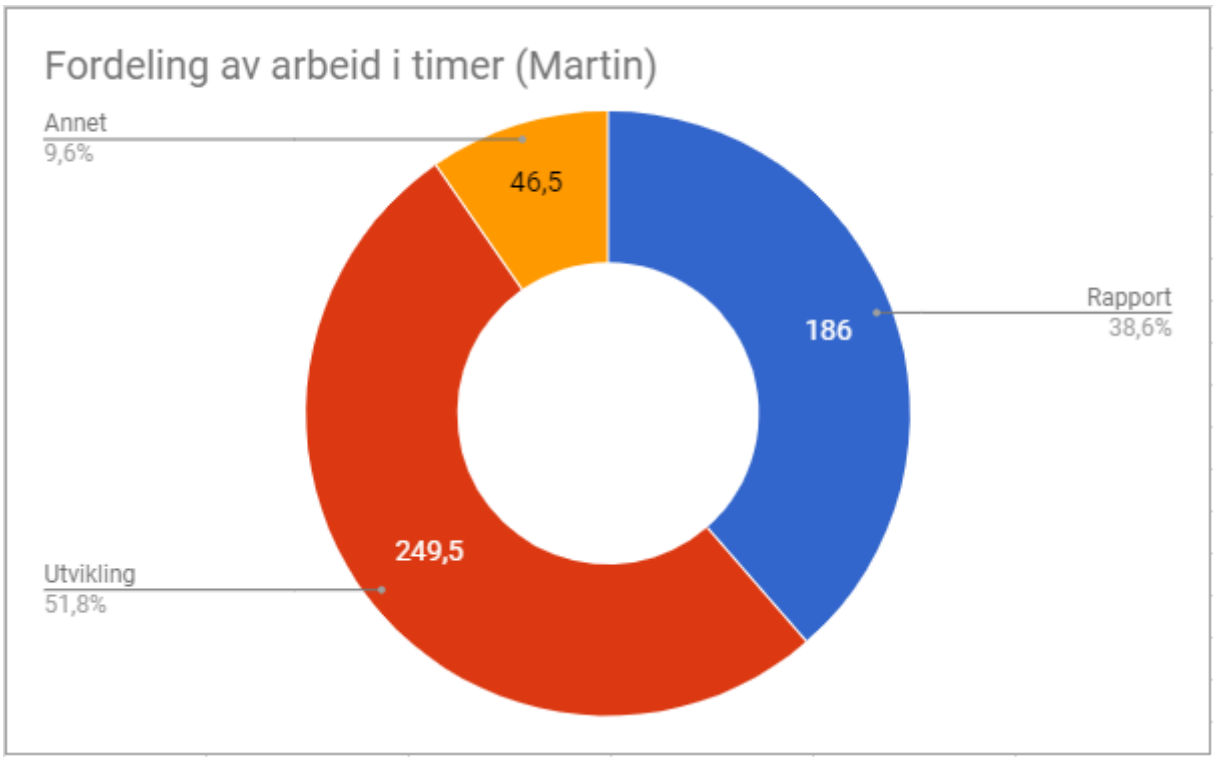
N Timelister

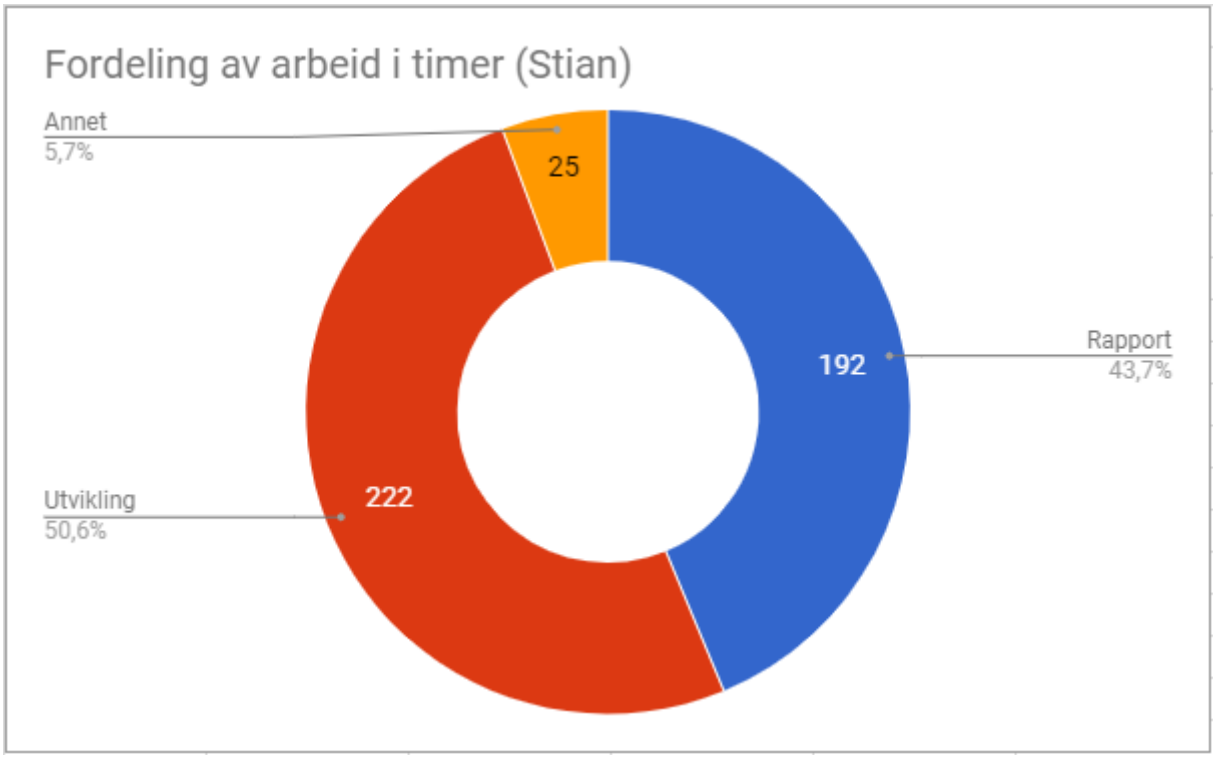


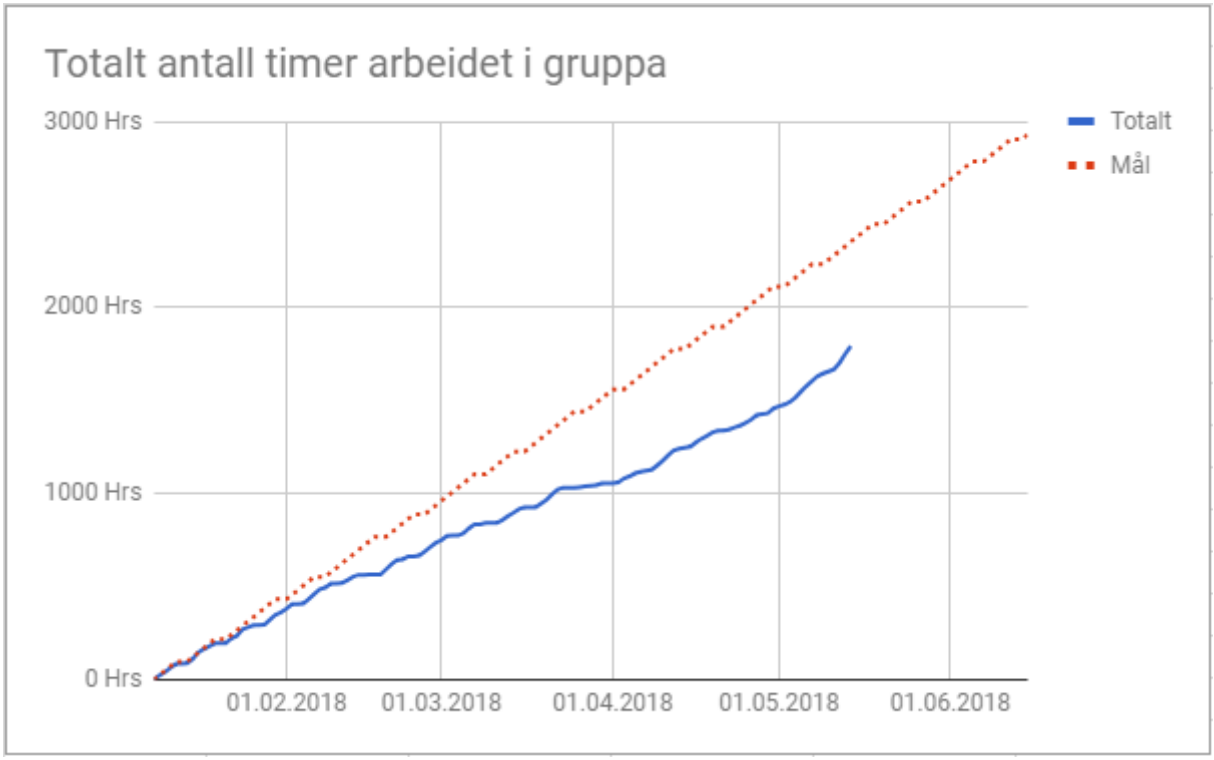


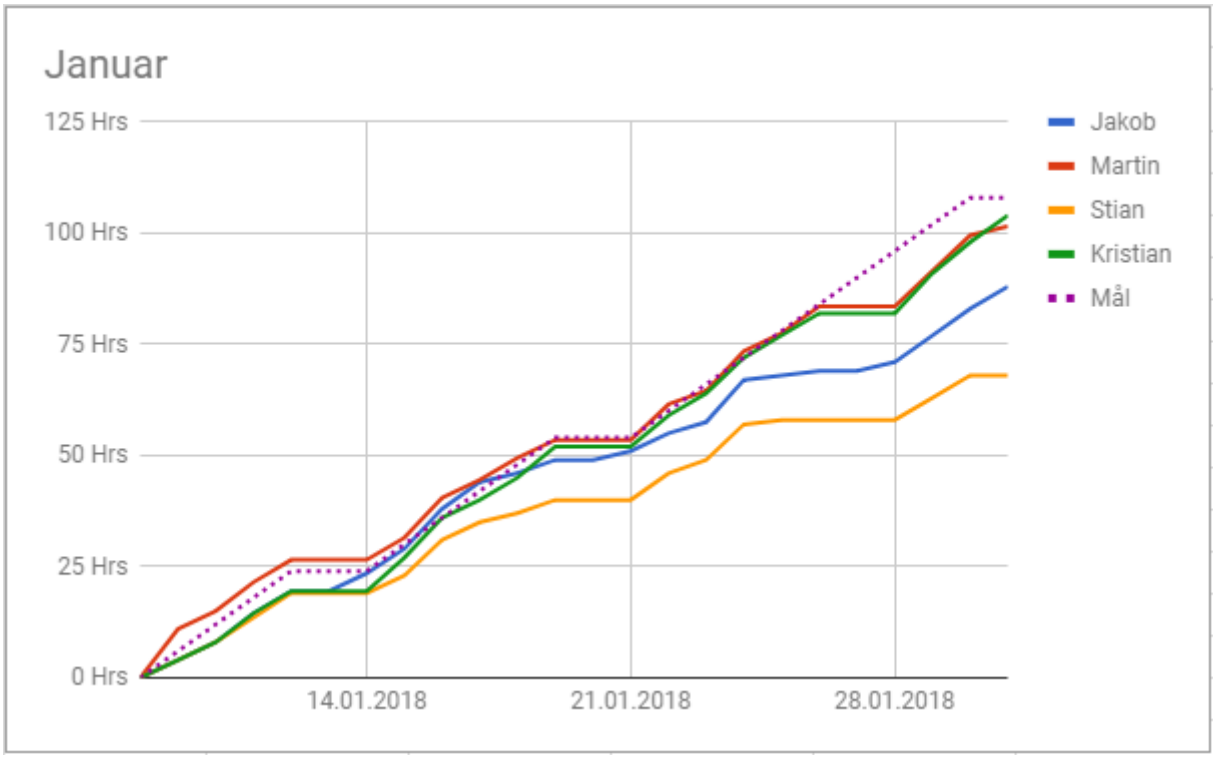

Fordeling av arbeid i timer (Jakob)
Annet
10,9%
50
242,5
Rapport
52,7%
Utvikling
36,5%
168

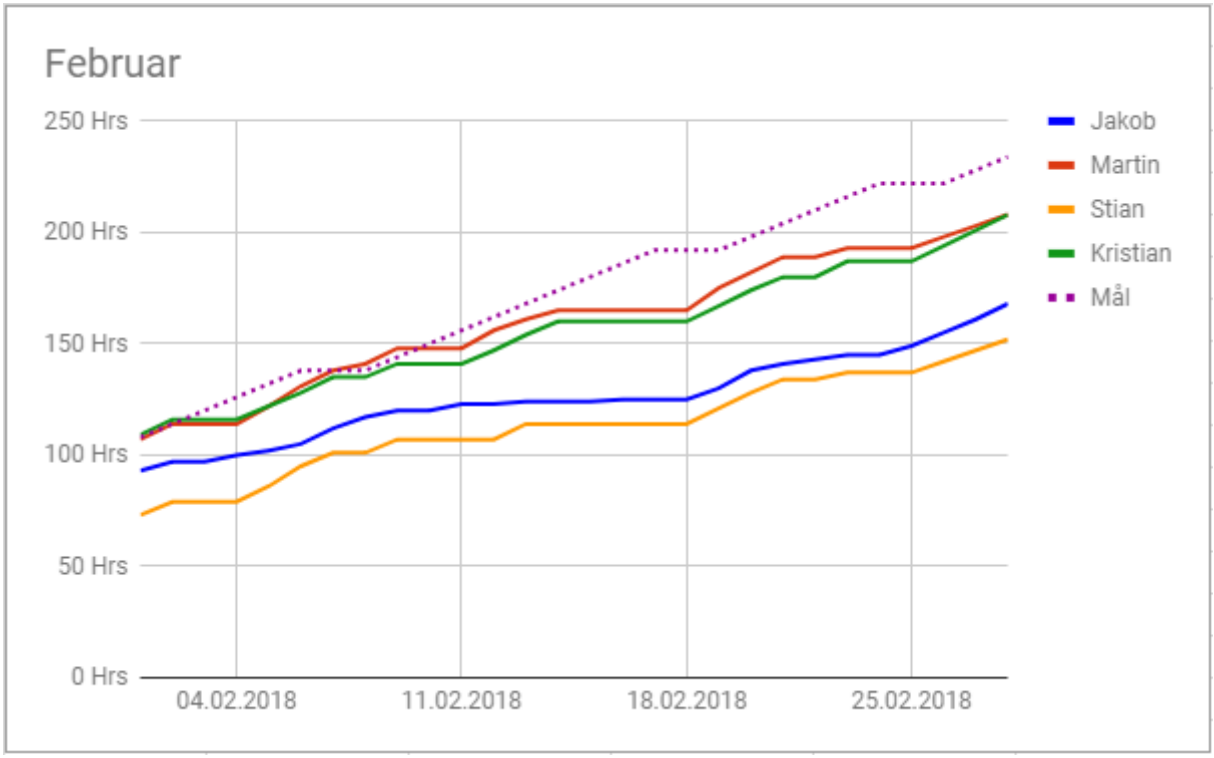

Fordeling av arbeid i timer (Kristian)
Annet
4,9%
23
228
Rapport
48,1%
Utvikling
47,0%
223

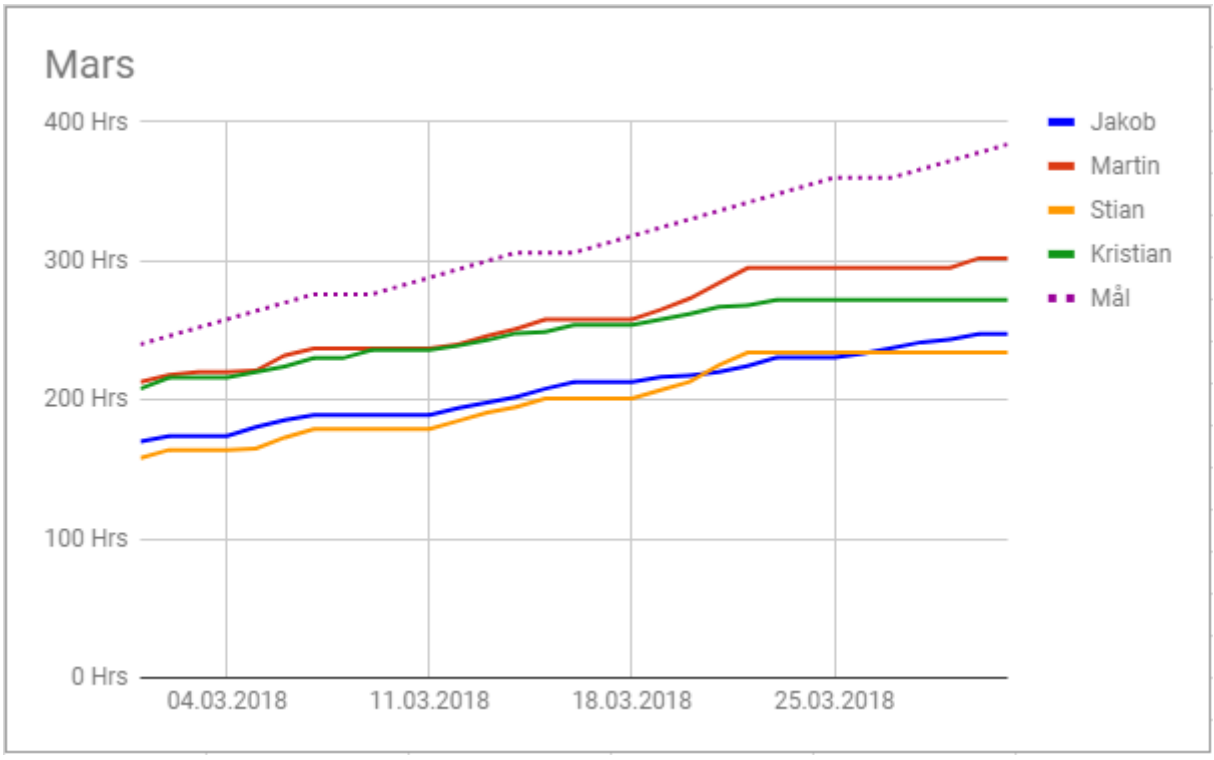

Fordeling av arbeid i timer (Martin)
Annet
9,6%
46,5
Rapport
38,6%
186
249,5
Utvikling
51,8%

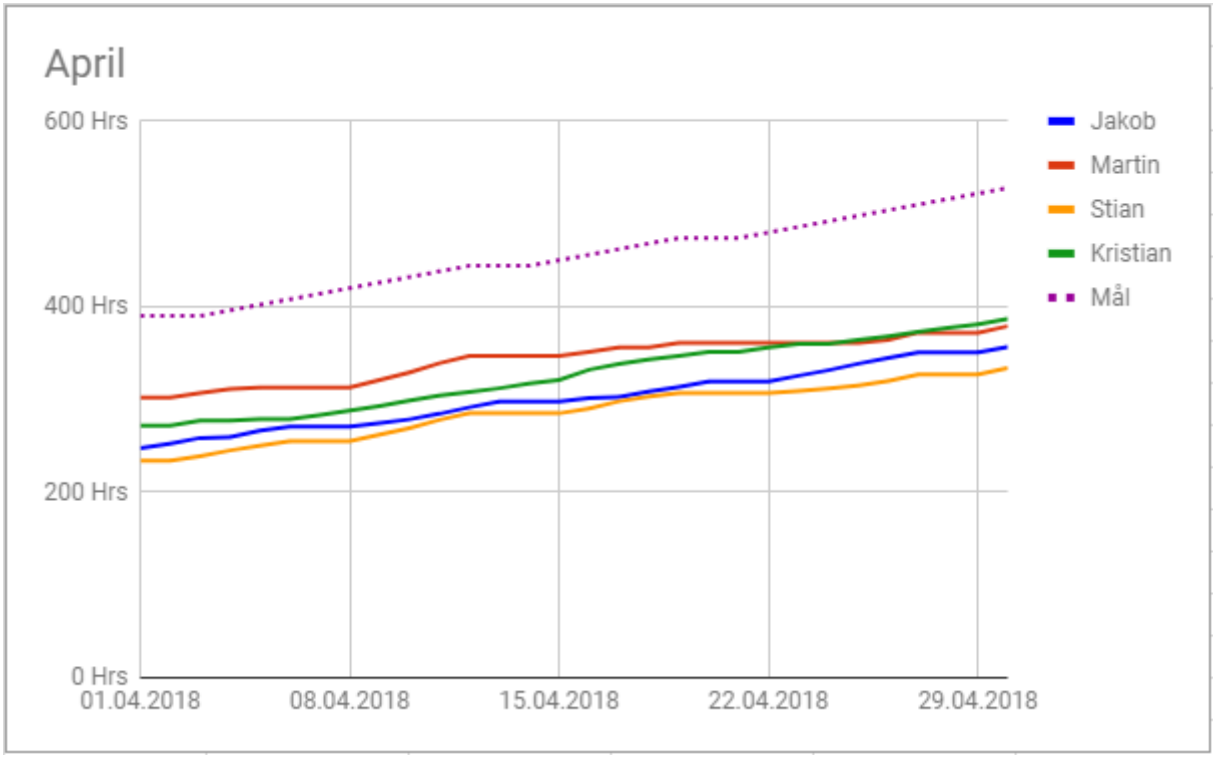

Fordeling av arbeid i timer (Stian)
Annet
5,7%
25
Rapport
43,7%
192
222
Utvikling
50,6%

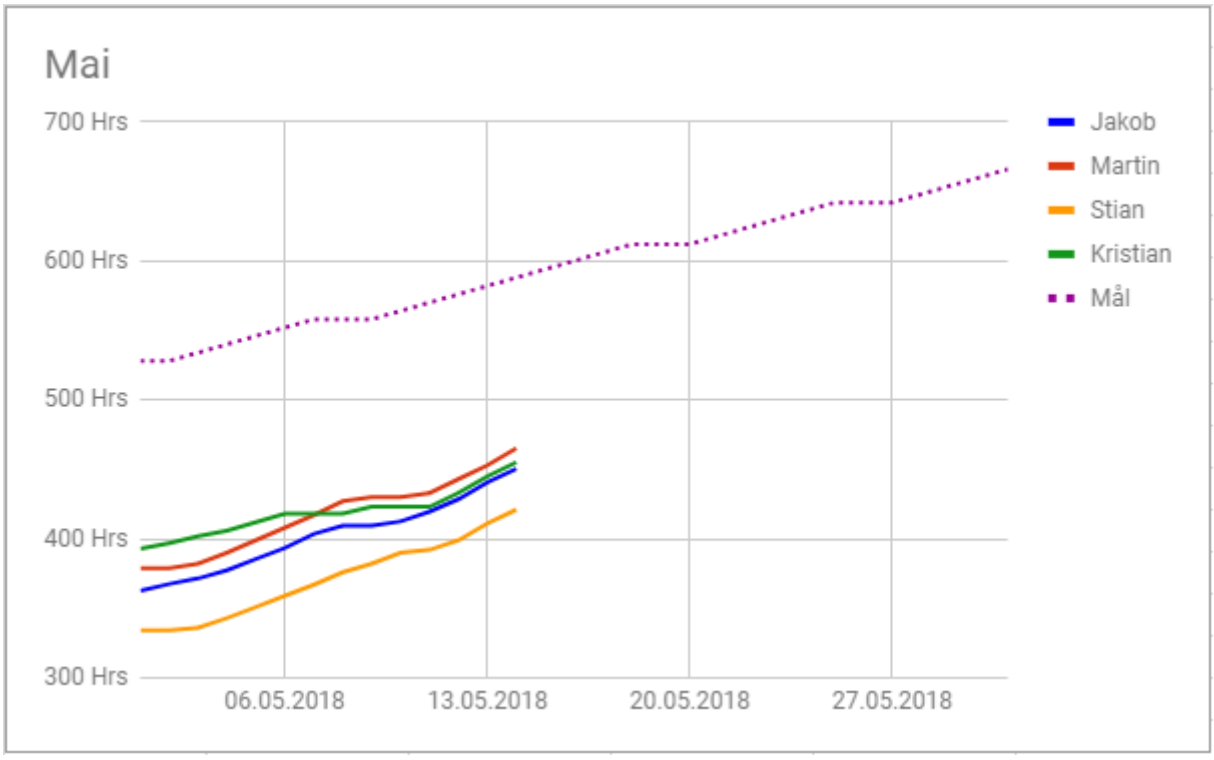

Totalt antall timer arbeidet i gruppa
3000 Hrs
2000 Hrs
1000 Hrs
0 Hrs
01.02.2018
01.03.2018
01.04.2018
01.05.2018
01.06.2018
Totalt
Mål


Januar
125 Hrs
100 Hrs
75 Hrs
50 Hrs
25 Hrs
0 Hrs
14.01.2018
21.01.2018
28.01.2018
Jakob
Martin
Stian
Kristian
Mål


Februar
250 Hrs
200 Hrs
150 Hrs
100 Hrs
50 Hrs
0 Hrs
04.02.2018
11.02.2018
18.02.2018
25.02.2018
Jakob
Martin
Stian
Kristian
Mål


Mars
400 Hrs
300 Hrs
200 Hrs
100 Hrs
0 Hrs
04.03.2018
11.03.2018
18.03.2018
25.03.2018
Jakob
Martin
Stian
Kristian
Mål


April
600 Hrs
400 Hrs
200 Hrs
0 Hrs
01.04.2018
08.04.2018
15.04.2018
22.04.2018
29.04.2018
Jakob
Martin
Stian
Kristian
Mål


Mai
700 Hrs
600 Hrs
500 Hrs
400 Hrs
300 Hrs
06.05.2018
13.05.2018
20.05.2018
27.05.2018
Jakob
Martin
Stian
Kristian
Mål

Dato	Antall Timer	Arbeidets art - Jakob	Rapport	Utvikling	Annet [1]
Totalt	460,5		242,5	168	50
	JANUAR				
09.01.2018	4	Lære arduino		4	
10.01.2018	4	Lære arduino		4	
11.01.2018	6,5	Grupperegler, timelister, trello, lese opp på gamle bachelor oppg	5		1,5
12.01.2018	5	Satt opp Eclipse.		5	
13.01.2018	0				
14.01.2018	4	Leste på tidliger bachelor + Fikset Timelister			4
15.01.2018	5,5	Prosjektplan og research	5,5		
16.01.2018	9	Prosjektplan, timelister + møte med Jonas	4		5
17.01.2018	6	Prosjektplan kort møte med røyse + Fikse Timelister	1,5		4,5
18.01.2018	2	Jobbet litt mer med timelistene			2
19.01.2018	3	Prosjektplan	3		
20.01.2018	0				
21.01.2018	2	Prosjektplan - Omfang	2		
22.01.2018	4	Skrive Fagområde	4		
23.01.2018	2,5	Prosjektplan - Omfang	2,5		
24.01.2018	9,5	Prosjektplan siste finnish	9,5		
25.01.2018	1	Møte med veileder			1
26.01.2018	1				1
27.01.2018	0				
28.01.2018	2	Forslag til sprinter			2
29.01.2018	6	Arbeid med user stories		3	3
30.01.2018	6	Møte med produkteier, user stories		2	4
31.01.2018	5	research		5	
	88	Antall Timer	37	23	28
	FEBRUAR				
01.02.2018	5	research + kort møte med veileder		5	
02.02.2018	4	WIFI Rasp finne løsning		4	
03.02.2018	0				
04.02.2018	3	WIFI Rasp		3	
05.02.2018	2	WIFI Rasp Problemer med Rasp nettverksdriver		2	
06.02.2018	3	WIFI Rasp + Sette opp usercases etc sprint 2		3	
07.02.2018	7	WIFI Rasp + Sprint møte 1 holdt med produkteier. (Se til referat)		6	1
08.02.2018	5	WIFI Rasp Fungerer men sletter nettverksdrivere etter første oppkobling		5	
09.02.2018	3	WIFI Rasp		3	
10.02.2018	0				
11.02.2018	3	WIFI Rasp - ferdig går bort ifra denne løsningen		3	
12.02.2018	0				
13.02.2018	1	Sprint møte 2 holdt med produkteier. (Se til referat)			1
14.02.2018	0				
15.02.2018	0				
16.02.2018	1	Innom veileder(Hente papirer)			1
17.02.2018	0				
18.02.2018	0				
19.02.2018	5	Sett på wifi. Blitt enig om å ikke jobbe mer på wifi intill videre		5	
20.02.2018	8	Rapport + møte om node. Sprint møte 3 holdt med produkteier. (Se til referat)		2	6
21.02.2018	3	Rapport		3	
22.02.2018	2	Lære node		2	
23.02.2018	2	Lære node		2	
24.02.2018	0				
25.02.2018	4	NodeJS - Socketio		4	
26.02.2018	6	NodeJS - Socketio		6	
27.02.2018	6	NodeJS - Socketio		6	
28.02.2018	7	NodeJS - Socketio		7	
	80	Antall timer	0	71	9
	MARS				
01.03.2018	2	rapport + research	1		1
02.03.2018	4	rapport + research	2		2
03.03.2018	0				
04.03.2018	0				
05.03.2018	6,5	Rapport og gruppemøte	5,5		1
06.03.2018	5	Rapport. Sprint møte 4 holdt med produkteier. (Se til referat)	4		1
07.03.2018	3,5	Rapport	3,5		

Dato	Antall Timer	Arbeidets art - Jakob	Rapport	Utvikling	Annet [1]
Totalt	460,5		242,5	168	50
08.03.2018	0	Fri			
09.03.2018	0	Fri			
10.03.2018	0	Helg			
11.03.2018	0	Helg			
12.03.2018	5	Rapport	5		
13.03.2018	4	Rapport	4		
14.03.2018	4	Utvikling		4	
15.03.2018	6	Utvikling + Møte		5	1
16.03.2018	5	Utvikling		5	
17.03.2018	0	Helg			
18.03.2018	0	Helg			
19.03.2018	3,5	Rapport	3,5		
20.03.2018	1	Sprint møte 5 holdt med produkteier. (Se til referat)			1
21.03.2018	2,5	Utvikling		2,5	
22.03.2018	4,5	Utvikling + møte med Veileder		4,5	
23.03.2018	6	Rapport	5		1
24.03.2018	0	helg			
25.03.2018	0	helg			
26.03.2018	3	Utvikling Regelverk		3	
27.03.2018	4	Utvikling Regelverk		4	
28.03.2018	4	Utvikling Regelverk		4	
29.03.2018	2	Utvikling Regelverk		2	
30.03.2018	4	Utvikling Regelverk		4	
31.03.2018	0	helg			
	79,5	Antall timer	33,5	38	8
	APRIL				
01.04.2018	0	helg			
02.04.2018	5	Utvikling Regelverk		5	
03.04.2018	6	Utvikling Regelverk		6	
04.04.2018	1	Sprint møte 6 holdt med produkteier. (Se til referat)			1
05.04.2018	7	Utvikling Regelverk		7	
06.04.2018	4	Utvikling Regelverk		4	
07.04.2018	0	helg			
08.04.2018	0	helg			
09.04.2018	4	Utvikling Regelverk		4	
10.04.2018	4	Utvikling Regelverk		4	
11.04.2018	6	Utvikling Regelverk		6	
12.04.2018	7	Rapport Innledning	7		
13.04.2018	6	Rapport Innledning	6		
14.04.2018	0	helg			
15.04.2018	0	helg			
16.04.2018	4	Rapport + møte	3		1
17.04.2018	1	Sprint møte 7 holdt med produkteier. (Se til referat)			1
18.04.2018	6	Rapport Innledning	6		
19.04.2018	5	Rapport Innledning	5		
20.04.2018	6	Rapport Innledning	6		
21.04.2018	0	helg			
22.04.2018	0	helg			
23.04.2018	6	Rapport Innledning	6		
24.04.2018	6	Rapport Innledning	6		
25.04.2018	7	Rapport Personas + møte	6		1
26.04.2018	6	Rapport Personas + møte	5		1
27.04.2018	6	Rapport Personas	6		
28.04.2018	0	helg			
29.04.2018	0	helg			
30.04.2018	6	Rapport Personas	6		
	109	Antall timer	68	36	5
	MAI				
01.05.2018	6	Rapport Personas	6		
02.05.2018	5	Rapport Personas	5		
03.05.2018	4	Rapport Personas	4		
04.05.2018	6	Rapport Personas	6		
05.05.2018	8	Rapport Personas	8		
06.05.2018	8	Rapport PACT	8		

Dato	Antall Timer	Arbeidets art - Jakob	Rapport	Utvikling	Annet [1]
Totalt	460,5		242,5	168	50
07.05.2018	10	Rapport Kodekvalitet + research	10		
08.05.2018	6	Kodekvalitet + research + jobbe med vedlegg	6		
09.05.2018	0	Pause			
10.05.2018	3	Ny Template	3		
11.05.2018	7	Rapport Ordliste / Referanser /	7		
12.05.2018	9	Rapport Ordliste / Referanser /	9		
13.05.2018	12	Rapport finpuss	12		
14.05.2018	10	Rapport finpuss	10		
15.05.2018	10	Rapport finpuss	10		
16.05.2018	0				
17.05.2018	0				
18.05.2018	0				
19.05.2018	0				
20.05.2018	0				
21.05.2018	0				
22.05.2018	0				
23.05.2018	0				
24.05.2018	0				
25.05.2018	0				
26.05.2018	0				
27.05.2018	0				
28.05.2018	0				
29.05.2018	0				
30.05.2018	0				
31.05.2018	0				
	104	Antall timer	104	0	0

Dato	Antall Timer	Arbeidets art - Kristian	Rapport	Utvikling	Annet
Totalt	465		228	223	23
	JANUAR				
09.01.2018	4	Satt opp Arduino IDE, blitt kjent med denne, samt forskjellige sensorer.		5	
10.01.2018	4	Jobbet videre med å bli kjent med sensorene. Begynt å sette opp dokumenter rundt videre arbeid		5	
11.01.2018	6,5	Møtt med veileder, skrevet refferat og lagt i mappen. Videre lest dokumenter fra blackboard og sett på kapittel 1 til "MittSmartHjem"	7,5		
12.01.2018	5	Satt opp Eclips IDE med Arduino pluggin. Sett på projektplaner fra tidliger og startet fordelingen av hva hver av oss skal skrive	4	6	
13.01.2018		Helg			
14.01.2018		Helg			
15.01.2018	7,5	Skrevet referat fra gruppemøte. Ligger på google driven. Skrevet utkast til pungt 4 til prosjektplan	8,5		
16.01.2018	9	Skrevet om milepæler, besluttningspunkt og gantt. Møte med arbeidsgiver	8		1
17.01.2018	4	Skrevet detaljer om Gantt-skjema, levert førsteutkast av prosjektplan til veileder	4		
18.01.2018	5	Vært på møte, skrevet referat fra møte. Skrevet om litt i prosjektplan, er nødt til å gjøre større forandringer	4		1
19.01.2018	7	Skrevet om deler av rapport	7		
20.01.2018	0	Helg			
21.01.2018	0	Helg			
22.01.2018	7	Skrevet om deler av rapport for punkt 4	7		
23.01.2018	5	Skrevet referat fra møte med produkteier, og skrevet om deler av punkt 4	4		1
24.01.2018	8	Skrevet om deler av prosjektplan, samt gått igjennom med gruppen så alle er enig i hva vi har skrevet	8		
25.01.2018	5	Jobbet med prosjektplan	5		
26.01.2018	5	Skrevet referat fra møte med veileder. Gjort om deler av prosjektplan etter 2 utkast	4		1
27.01.2018	0	Helg			
28.01.2018	0	Helg			
29.01.2018	9	Skrevet om siste delene av prosjektplan. Satt meg inn i hvordan få dynamisk tilkobling mellom sensorer og wifi	5	4	
30.01.2018	7	Hatt møte med produkteier, skrevet referat. Satt inn ulike userstories. Startet å se på wifimanager		6	1
31.01.2018	6	Lest mer opp på wifimanager.		6	
	104	Antall Timer	76	32	5
	FEBRUAR				
01.02.2018	5	Hatt møte med veileder, skrevet referat. Testet ut WiFiManager		4	1
02.02.2018	7	Jobbet med WiFi manager		7	
03.02.2018	0	Helg			
04.02.2018	0	Helg			
05.02.2018	6	Jobbet med WiFi manager		6	
06.02.2018	6	Jobbet med WiFi manager		6	
07.02.2018	7	Jobbet med WiFi manager. Sprint møte 1 holdt med produkteier. (Se til referat)		6	1
08.02.2018	0	Dag med fokus på mobilutvikling			
09.02.2018	6	Jobbet med WiFi manager		6	
10.02.2018	0	Helg			
11.02.2018	0	Helg			
12.02.2018	6	Jobbet med WiFi manager		6	
13.02.2018	7	Jobbet med WiFi manager. Sprint møte 2 holdt med produkteier (Se til referat)		6	1
14.02.2018	6	Jobbet på skolen, problemer med å få gjort arbeid		6	
15.02.2018	0	Dag med fokus på mobilutvikling			
16.02.2018	0	Avspasering			
17.02.2018	0	Helg			
18.02.2018	0	Helg			
19.02.2018	7	Fullført oppkobling for sensorer mot Raspbarry over internett og sender data. Startet vurdering av å webgrensesnitt for å vise frem data, samt muligheter å sende tilbake til reler.		6	1
20.02.2018	7	Sprint møte 3 holdt med produkteier (Se til referat)		6	1
21.02.2018	6	Startet å se på mulighetene til å bruke React.JS til websiden		6	
22.02.2018	0	Dag med fokus på mobilutvikling			
23.02.2018	7	Jobbet med å lage sketches for webside		7	
24.02.2018	0	Helg			
25.02.2018	0	Helg			
26.02.2018	7	Gjort noen oppgaver med React.JS og forsøkt implementering av dette til en dummy versjon av websiden		7	
27.02.2018	7	Laget ferdig sketches for hvordan websiden kan se ut.		7	

Dato	Antall Timer	Arbeidets art - Kristian	Rapport	Utvikling	Annet
Totalt	465		228	223	23
28.02.2018	7			7	
	104	Antall timer	0	99	5
	MARS				
01.03.2018	0	Dag med fokus på mobilutviklign			
02.03.2018	8	Sett på mulighet til å bruke "If This Then That" metoden for oppsett av regelverk, puttet dette inn i sketchene for websiden		8	
03.03.2018	0	Helg			
04.03.2018	0	Helg			
05.03.2018	4	Møte med gruppen anngående vidreutvikling av webside. Kommet frem til at React.JS vil ta for mye tid å bli kjent med		3	1
06.03.2018	4	Jobbet videre med å lage en display mulighet, med muligheter til å sette tempratur		4	
07.03.2018	6	Sprint møte 4 holdt med produkteier (Se til referat)		5	1
08.03.2018	0	Dag med fokus på mobilutviklign			
09.03.2018	6	Jobbet med design for webgrensesnitt		6	
10.03.2018	0	Helg			
11.03.2018	0	Helg			
12.03.2018	3	Jobbet med webgrensesnitt		3	
13.03.2018	4	Startet å lage regelverk for webgrensesnitt		4	
14.03.2018	5	Startet å lage regelverk for webgrensesnitt		5	
15.03.2018	1	Dag med fokus på mobilutviklign. Møte med veileder (Se referat)			1
16.03.2018	5			5	
17.03.2018	0	Helg			
18.03.2018	0	Helg			
19.03.2018	4	Lest opp mer for IFTTT metoden, satt opp diagram for hvordan det skal løses		4	
20.03.2018	4	Bygd ut videre på sketcher. Sprint møte 5 holdt med produkteier (Se til referat)		3	1
21.03.2018	5			5	
22.03.2018	1	Dag med fokus på mobilutviklign. Møte med veileder (Se referat). Pc'en min sluttet å fungere. Batteriet vil ikke lade lenger. Sendt til verksted og fått tak i låne pc, men klarer ikke å kjøre IDE på den			1
23.03.2018	4	Lest opp på mulige rapporter vi kan sammenlignes oss med	4		
24.03.2018	0	Helg			
25.03.2018	0	Helg			
26.03.2018	0	Påskeferie			
27.03.2018	0	Påskeferie			
28.03.2018	0	Påskeferie			
29.03.2018	0	Påskeferie			
30.03.2018	0	Påskeferie			
31.03.2018	0	Påskeferie			
	64	Antall timer	4	55	5
	APRIL				
01.04.2018	0	Påskeferie			
02.04.2018	0	Påskeferie			
03.04.2018	5	Lest opp på mulige rapporter vi kan sammenlignes med. Sprint møte 6 holdt med produkteier (Se referat)	4		1
04.04.2018	0	Jobbet med mobilutvikling obliger			
05.04.2018	2	Dag med fokus på mobilutvikling. Møte med veileder (Se referat). Møte med gruppen			2
06.04.2018	0	Jobbet med mobilutvikling obliger			
07.04.2018	4	Rapport skriving, pc'en er på verksted	4		
08.04.2018	5	Jobbet hos Jakob med utvikling av webgrensesnitt for regelverk		5	
09.04.2018	5	Arbeid med utviling av webgrensesnitt for regelverk		5	
10.04.2018	6	Arbeid med utviling av webgrensesnitt for regelverk		6	
11.04.2018	5	Arbeid med utviling av webgrensesnitt for regelverk		5	
12.04.2018	4	Arbeid med utviling av webgrensesnitt for regelverk		4	
13.04.2018	4	Sriving av rapport, lest opp på andres bruk av utviklingsprosess	4		
14.04.2018	5	Sriving av rapport, utviklingsprosess kap. 2. Skrevet om utviklingsmetodikk, valg av metodikk, argumentasjon	5		
15.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2. Lest over gårsdagen og skrevet om deler	4		
16.04.2018	11	Integering av webgrensesnitt med backend, støtet på problemer, innser at vi kanske må lage en simplere versjon. Samtaler med Martin rundt backend		7	4
17.04.2018	6	Ferdig med utviklingsperioden. Lagde en simplere versjon for det å lagre regelverk til db. Sprint møte 7 holdt med produkteier (Se referat)		5	1

Dato	Antall Timer	Arbeidets art - Kristian	Rapport	Utvikling	Annet
Totalt	465		228	223	23
18.04.2018	5	Sriving av rapport, utviklingsprosess kap. 2. Gjennomføring, Møter med oppdragsgiver, Interne møter	5		
19.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2. Lest over gårsdagen og skrevet om deler på det	4		
20.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2	4		
21.04.2018	0				
22.04.2018	5	Sriving av rapport, utviklingsprosess kap. 2. Skrevet om scrum board, estimering og møter med veileder	5		
23.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2 Lest over gårsdagens arbeid	4		
24.04.2018	0				
25.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2 Startet å skrive om hvordan sprinten gjennførtes	4		
26.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2 Samlet informasjon for hver sprint og skrevet mer detaljert om det	4		
27.04.2018	5	Sriving av rapport, utviklingsprosess kap. 2. Skrevet ferdig all info for sprinter	5		
28.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2. Lest over og forandret på ting i kap 2	4		
29.04.2018	4	Sriving av rapport, utviklingsprosess kap. 2. Startet arbeid med modeller for kap. 2	4		
30.04.2018	6	Jobbet videre med figurer. Hatt møte med veileder, skrevet om kap 2 i henhold til hva han sa. (Se til referat). Startet arbeid på kap. 4	6		
	115	Antall timer	70	37	8
	MAI				
01.05.2018	6	Skrevet ferdig kap 2 i henhold til tilbakemeldinger	6		
02.05.2018	4	Jobber videre med figurer for Kap 2	4		
03.05.2018	5	Startet å skrive om design til brukergrensesnitt	5		
04.05.2018	4	Skriver om design til brukergrensesnitt	4		
05.05.2018	6	Skriver om design til brukergrensesnitt	6		
06.05.2018	6	Skriver om design til brukergrensesnitt	6		
07.05.2018	0	Syk med feber			
08.05.2018	0	Syk med feber			
09.05.2018	5	Skrevet om vedlegg	5		
10.05.2018	0	Syk med feber			
11.05.2018	0	Syk med feber			
12.05.2018	10	Siste dagene med bacheloroppgaven	10		
13.05.2018	12	Siste dagene med bacheloroppgaven	12		
14.05.2018	10	Siste dagene med bacheloroppgaven	10		
15.05.2018	10	Lest over og fullført oppgaven	10		
16.05.2018	0				
17.05.2018	0				
18.05.2018	0				
19.05.2018	0				
20.05.2018	0				
21.05.2018	0				
22.05.2018	0				
23.05.2018	0				
24.05.2018	0				
25.05.2018	0				
26.05.2018	0				
27.05.2018	0				
28.05.2018	0				
29.05.2018	0				
30.05.2018	0				
31.05.2018	0				
	78	Antall timer	78	0	0

Dato	Antall Timer	Arbeidets art - Martin	Rapport	Utvikling	Annet
Totalt	475		186	249,5	46,5
	JANUAR				
01.01.2018	5	Lynkurs med FosenUtvikling		6	
08.01.2018	2	Bli kjent med produktene for utlevering til gruppa		3	
09.01.2018	4	Kobling, schematics og koding av Temp/fukt sensor og Touch Sensor og Planlegging		4	1
10.01.2018	4	Kobling, schematics og koding av lys sensor og Planlegging		4	1
11.01.2018	6,5	Kobling, schematics og koding av PIR sensor og Planlegging og møte med veileder		5	2,5
12.01.2018	5	Prosjektplanskriving og oppkobling av utviklingsplattform Eclipse	5		1
13.01.2018		HELG			
14.01.2018		HELG			
15.01.2018	5	Skriving av prosjektplan	6		
16.01.2018	9	Skriving av prosjektplan (omskriving til Latex) og møte med Fosen Utvikling	8		1
17.01.2018	4	Rapportskriving	4		
18.01.2018	5	Rapport og møte med veileder	4		1
19.01.2018	4	Skriving av rapport, retting av Prosjektplan	4		
20.01.2018	0	HELG			
21.01.2018	0	HELG			
22.01.2018	8	Skriving av Prosjektplan og forb Raspberry Pi	7		1
23.01.2018	3	Rapport og møte med Fosen Utvikling	2		1
24.01.2018	9	Prosjektplanskriving og konfigurering av Raspberry Pi	5	4	
25.01.2018	4	Møte med veileder og forb Raspberry Pi		3	1
26.01.2018	6	Møte med veileder og installering av MQTT på Raspberry Pi		5	1
27.01.2018	0	HELG			
28.01.2018	0	HELG			
29.01.2018	8	Temp/humidity over Wifi til Raspberry Pi med MQTT		8	
30.01.2018	8	Touch og lyssensor mot MQTT over wifi og møte med produkteier		7	1
31.01.2018	2	Research MQTT			2
	101,5	Antall Timer	45	49	14,5
	FEBRUAR				
01.02.2018	5,5	Research MQTT Raspberry Pi, møte med veileder og installasjon av WIFI Adapter		4,5	1
02.02.2018	7	MQTT lagring av publish data på SQLite database på raspberry Pi		7	
03.02.2018	0	HELG			
04.02.2018	0	HELG			
05.02.2018	8	Hente data på MQTT broker og skrive til database		8	
06.02.2018	9	Sende data som JSON fra sensor til MQTT broker for så å lagre i database, og møte med veileder. Sprint møte 1 holdt med produkteier. (Se til referat)		9	
07.02.2018	7	Oppsett av Sprint 2, oppsett av ESP wifi/sensor/powersup		7	
08.02.2018	3	Research		3	
09.02.2018	7	Database og Listener på raspberry Pi, samt Databasemodell		7	
10.02.2018	0	HELG			
11.02.2018	0	HELG			
12.02.2018	8	Listener og JSON		8	
13.02.2018	5	Listener og JSON. Sprint møte 2 holdt med produkteier. (Se til referat)		5	
14.02.2018	4	Oppkobling av Raspberry på et annet nettverk (skolen)		4	
15.02.2018	0	Jobbet med annet fag			
16.02.2018	0	Avspasering			
17.02.2018	0	HELG			
18.02.2018	0	HELG			
19.02.2018	10	Raspberry oppkobling og Listener på annet nettverk (Success) og statusrapport skriving	5	5	
20.02.2018	7	Planlegging av WebApplikasjon (node.js) og Sprint møte 3 holdt med produkteier. (Se til referat)			7
21.02.2018	7	Oppsett av NodeJS server/client websockets		7	
22.02.2018	0				
23.02.2018	4	Forts NodeJS Server/Client		4	
24.02.2018	0	HELG			
25.02.2018	0	HELG			
26.02.2018	5	Forts NodeJS Server/Client		5	
27.02.2018	5	Forts NodeJS Server/Client		5	
28.02.2018	5	Mer research NodeJs		5	
	106,5	Antall timer	5	93,5	8
	MARS				
01.03.2018	5	NodeJs Server/Client, sqlite pakke		5	

Dato	Antall Timer	Arbeidets art - Martin	Rapport	Utvikling	Annet
Totalt	475		186	249,5	46,5
02.03.2018	5	NodeJs kobling mellom sqlite og server		5	
03.03.2018	2	SQLite bug fikset (HELG)		2	
04.03.2018	0	HELG			
05.03.2018	1	Møte med gruppen (Planlegging for utvikling av nettside)			1
06.03.2018	11	Utvidelse av NodeJs server og klient for sending av data som JSON med korrekt visning. Sprint møte 4 holdt med produkteier. (Se til referat)		10	1
07.03.2018	5	NodeJS		5	
08.03.2018	0	Annet fag			
09.03.2018	0	Annet fag			
10.03.2018	0	HELG			
11.03.2018	0	HELG			
12.03.2018	3	Database serverside for nettside med kobling mot SQLite3 på raspberry gjennom nodeJS		3	
13.03.2018	6	Videre bygging av NodeJS og møte med produkteier		5	1
14.03.2018	5	Forsøk på automatisering av client / server rasp - windows.		5	
15.03.2018	7	Feilsøking av Temp/Hum sensor og møte med veileder.		6	1
16.03.2018	0	Annet fag			
17.03.2018	0	HELG			
18.03.2018	0	HELG			
19.03.2018	7	NodeJs Client / Server, kommunikasjon fungerer nå slik den skal, mangler automatisering.		7	
20.03.2018	8	Automatisering av NodeJs på Raspberry Pi, samt automatisering av Listener MQTT. Har også satt sammen Rasp med server og første del av nettside er på plass. Gjennomført møte internt med gruppen. Sprint møte 5 holdt med produkteier. (Se til referat)		6	2
21.03.2018	11	Rapportskriving: Innledning	5	6	
22.03.2018	11	Raspberry sender nå en egen ID for å koble til database. Møte med veileder og jobba på rapport under skrivedag på skolen.	5	5	1
23.03.2018	0	Annet fag			
24.03.2018	0	HELG			
25.03.2018	0	HELG			
26.03.2018	0	Påskeferie			
27.03.2018	0	Påskeferie			
28.03.2018	0	Påskeferie			
29.03.2018	0	Påskeferie			
30.03.2018	7	Møte med fosen utvikling			7
31.03.2018	0	Påskeferie			
	94	Antall timer	10	70	14
	APRIL				
01.04.2018	0	Påskeferie			
02.04.2018	0	Påskeferie			
03.04.2018	5	Skriving av rapport, finpussing av kap 1	5		
04.04.2018	4	Skriving av rapport og levering av 1.utkast kap 1. Sprint møte 6 holdt med produkteier. (Se til referat)	3		1
05.04.2018	2	Møte med veileder og gruppen			2
06.04.2018	0				
07.04.2018	0	HELG			
08.04.2018	0	HELG			
09.04.2018	8	Feilsøking av Arduino sensorer, fungerer igjen nå		8	
10.04.2018	8	Fjernet bugs i koden for kommunikasjon mellom rasp og server		8	
11.04.2018	10	La inn en kanal for humidity for å fjerne duplikater		10	
12.04.2018	8	Fortsettelse på fjerning av duplikater på server shutdown, og kommunikasjon mot relay		8	
13.04.2018	0	annet fag			
14.04.2018	0	HELG			
15.04.2018	0	HELG			
16.04.2018	4	Planlegging av rapport og demonstrasjon av RaspPi til Kristian			4
17.04.2018	5	Siste innspurt på utvikling og Sprint møte 7 holdt med produkteier. (Se til referat)		3	2
18.04.2018	0	annet fag			
19.04.2018	5	Rapport skriving kap 2 og internt møte med gruppen	5		
20.04.2018	0	Annet fag			
21.04.2018	0	HELG			
22.04.2018	0	HELG			
23.04.2018	0	syk			
24.04.2018	0	annet fag			
25.04.2018	0	annet fag			

Dato	Antall Timer	Arbeidets art - Martin	Rapport	Utvikling	Annet
Totalt	475		186	249,5	46,5
26.04.2018	3	Skriving av rapport kap 2	3		
27.04.2018	8	Skriving rapport kap 2	8		
28.04.2018	0	HELG			
29.04.2018	0	HELG			
30.04.2018	7	Møte med veileder om kap 1 og 2, godkjent, skriving av kap 3	7		
	77	Antall timer	31	37	9
	MAI				
01.05.2018	0	annet fag			
02.05.2018	0	annet fag			
03.05.2018	3	Rapport skriving kap 3	3		
04.05.2018	8	Rapport skriving kap 3	8		
05.05.2018	9	Rapport Skriving kap 3	9		
06.05.2018	9	Rapport skriving kap 4	9		
07.05.2018	9	Rapport skriving kap 5	9		
08.05.2018	10	Rapport skriving kap 7	10		
09.05.2018	3	Finpussing av rapporten før levering til Frode	3		
10.05.2018	0	Raudag			
11.05.2018	3	Møte med veileder, jobbing med endringer i rapport	2		1
12.05.2018	10	Retting av Bachelor etter veiledermøte	10		
13.05.2018	10	Siste finish på bachelor rapport	10		
14.05.2018	12	Siste finish på bachelor rapport	12		
15.05.2018	10	Siste finish på bachelor rapport	10		
16.05.2018	0				
17.05.2018	0				
18.05.2018	0				
19.05.2018	0				
20.05.2018	0				
21.05.2018	0				
22.05.2018	0				
23.05.2018	0				
24.05.2018	0				
25.05.2018	0				
26.05.2018	0				
27.05.2018	0				
28.05.2018	0				
29.05.2018	0				
30.05.2018	0				
31.05.2018	0				
	96	Antall timer	95	0	1

4	Antall Timer	Arbeidets art - Stian	Rapport	Utvikling	Annet
Totalt	431		192	222	25
	JANUAR				
09.01.2018	4	Eclipse IDE oppsett / arduino		4	
10.01.2018	4	Eclipse IDE oppsett / arduino		4	
11.01.2018	5,5	Arduino sammenkobling av ulike sensorer		5,5	
12.01.2018	5,5	Eclipse oppsett på gruppemedlemmers dataer. Enkel mal for oppsett av Arduino eclipse		5,5	
13.01.2018					
14.01.2018					
15.01.2018	4	Skriving av Prosjektplan	4		
16.01.2018	8	Prosjektplan og oppsett av Doxygen i eclipse	4	4	
17.01.2018	4	Prosjektplan og ferdigstilling av doxygen	4	2	
18.01.2018	2	Møte med Frode og doxygen		2	1
19.01.2018	3	Prosjektplan	4		
20.01.2018	0				
21.01.2018	0				
22.01.2018	6	Prosjektplan	6		
23.01.2018	3	Prosjektplan	3		
24.01.2018	8	Gjennomgang og finpuss av prosjektplan.	8		
25.01.2018	1	møte frodis			1
26.01.2018		Research av mqtt i henhold til arduino		3	1
27.01.2018	0				
28.01.2018	0				
29.01.2018	5	Temp/humidity over Wifi til Raspberry Pi med MQTT		5	
30.01.2018	5	PIR/touch overfører data til rasp		5	
31.01.2018	0				
	68	Antall Timer	33	40	3
	FEBRUAR				
01.02.2018	5	UML Diagram for arduino sensorer/mqtt/wifi setup		5	
02.02.2018	6	Konstruksjon av bibliotek for sensorene		6	
03.02.2018	0				
04.02.2018	0				
05.02.2018	7	Tildele mqtt parametre for sensorer		7	
06.02.2018	9	Tildele mqtt parametre for sensorer. Sprint møte 1 holdt med produkteier. (Se til referat)		8	1
07.02.2018	6	MySmartHome biblio		6	
08.02.2018	0				
09.02.2018	6	MySmartHome biblio		6	
10.02.2018	0				
11.02.2018	0				
12.02.2018	0				
13.02.2018	7	Ferdigstilling av Wifi kobling for sensorer. Sprint møte 2 holdt med produkteier. (Se til referat)		6	1
14.02.2018	0				
15.02.2018	0				
16.02.2018	0				
17.02.2018	0				
18.02.2018	0				
19.02.2018	7	Fiksa temphumid setupen igjen - så egentli itj gjort ein sjit.		6	1
20.02.2018	7	Research nodejs. Sprint møte 3 holdt med produkteier. (Se til referat)			7
21.02.2018	6	Ferdigstilling av bibliotek		6	
22.02.2018	0				
23.02.2018	3	research nodejs			3
24.02.2018	0				
25.02.2018	0				
26.02.2018	5	research nodejs		5	
27.02.2018	5	research nodejs		5	
28.02.2018	5	research nodejs		5	
	84	Antall timer	0	71	13
	MARS				
01.03.2018	6	NodeJs og express installering		6	
02.03.2018	6	Webside Frontend		6	
03.03.2018	0				
04.03.2018	0				
05.03.2018	1				1
06.03.2018	8	Sprint møte(se referat). MySQL phpmyadmin databaseoppsett		7	1

4	Antall Timer	Arbeidets art - Stian	Rapport	Utvikling	Annet
Totalt	431		192	222	25
07.03.2018	6	Database oppsett: bruker - gateway - sensor - sensordata tables		6	
08.03.2018	0				
09.03.2018	0				
10.03.2018	0				
11.03.2018	0				
12.03.2018	6	Backend webside - MySQL queries - bruker - rasp tilkobling, socketio tilkobling		6	
13.03.2018	6	Backend webside - MySQL queries		4	2
14.03.2018	4	Backend webside - MySQL queries		4	
15.03.2018	6	Backend webside - MySQL queries		5	1
16.03.2018	0				
17.03.2018	0				
18.03.2018	0				
19.03.2018	6	Nodejs database		6	
20.03.2018	6	Sprint møte 5 holdt med produkteier. (Se til referat)		5	1
21.03.2018	12	Rapportskriving + nodejs database	4	7	1
22.03.2018	9	Rapportskriving + nodejs database	4	5	
23.03.2018	0				
24.03.2018	0				
25.03.2018	0				
26.03.2018	0				
27.03.2018	0				
28.03.2018	0				
29.03.2018	0				
30.03.2018	0				
31.03.2018	0				
	82	Antall timer	8	67	7
	APRIL				
01.04.2018	0				
02.04.2018	0				
03.04.2018	5	Notater for database, queries	1	4	
04.04.2018	6	Sprint møte 6 holdt med produkteier. (Se til referat)	5		1
05.04.2018	5	Rele utbygging	1	4	
06.04.2018	5	Rele utbygging	1	4	
07.04.2018	0				
08.04.2018	0				
09.04.2018	7	Rele utbygging		7	
10.04.2018	7	Rele problemer - arduino feilmeldinger(boot mode (1,7)(1,8)		7	
11.04.2018	9	Rele kontakt med gateway av og på		9	
12.04.2018	7	Publish subscribe rele		7	
13.04.2018	0				
14.04.2018	0				
15.04.2018	0				
16.04.2018	5	Use case diagram - notater	5		
17.04.2018	8	Sprint møte 7 holdt med produkteier. (Se til referat)	5	2	1
18.04.2018	5	Use cases	5		
19.04.2018	4	Rapportskriving	4		
20.04.2018	0				
21.04.2018	0				
22.04.2018	0				
23.04.2018	2	Rapportskriving	2		
24.04.2018	3	Rapportskriving	3		
25.04.2018	3	Rapportskriving	3		
26.04.2018	5	Rapportskriving	5		
27.04.2018	7	Rapportskriving	7		
28.04.2018	0				
29.04.2018	0				
30.04.2018	7	Rapportskriving	7		
	100	Antall timer	54	44	2
	MAI				
01.05.2018	0				
02.05.2018	0				
03.05.2018	2	Rapportskriving: kap 3	2		
04.05.2018	7	Rapportskriving: endringer/gjennomgang	7		

4	Antall Timer	Arbeidets art - Stian	Rapport	Utvikling	Annet
Totalt	431		192	222	25
05.05.2018	8	Rapportskriving: kap 4 arkitektur og design - enhetstdiagram forkl	8		
06.05.2018	8	Rapportskriving: kap 4 design sensor	8		
07.05.2018	8	Rapportskriving: kap 5	8		
08.05.2018	9	Rapportskriving: kap 5	9		
09.05.2018	6	Rapportskriving: kap 7	6		
10.05.2018	8	Rapportskriving: endringer/gjennomgang	8		
11.05.2018	2	Rapportskriving: endringer/gjennomgang	2		
12.05.2018	7	Rapportskriving: endringer/gjennomgang	7		
13.05.2018	12	Rapportskriving: endringer/gjennomgang	12		
14.05.2018	10	Rapportskriving: endringer/gjennomgang	10		
15.05.2018	10	Rapportskriving: endringer/gjennomgang	10		
16.05.2018	0				
17.05.2018	0				
18.05.2018	0				
19.05.2018	0				
20.05.2018	0				
21.05.2018	0				
22.05.2018	0				
23.05.2018	0				
24.05.2018	0				
25.05.2018	0				
26.05.2018	0				
27.05.2018	0				
28.05.2018	0				
29.05.2018	0				
30.05.2018	0				
31.05.2018	0				
	97	Antall timer	97	0	0

LoggBok

- Jakob Fonstad, Stian Fenstad, Martin Pukstad og Kristian Sundhaugen

O Loggbok

1.Januar:

Gjennomførte lynkurs sammen med Fosen Utvikling i oppsett av Arduino hardware, samt gjennomgang av mål og forventninger av prosjektet i sin helhet.

Uke 2. 8-14 Januar:

Mandag-Onsdag

Vi har jobbet med å bli kjent med Arduino og hardwaren vi er blitt tilsendt. Vi har også jobbet med Arduino IDE og kommet frem til at vi må gå over til å jobbe i eclipse, da Arduino IDE ikke har muligheten til å lage biblioteker. Ansvaret for å lære seg og sette opp utviklings verktøyet ble tildelt Stian
Martin og Jakob har laget schematics og demo koder for de forskjellige sensorene, Kristian har brukt disse for å teste om det går å sette opp det på sin egen PC.

Torsdag:

Vi har hatt første møte med veileder Frode.
Vi har blitt enige om grupperegler, de er signert og klart for å vise veileder.
Vi har valgt prosjektleder, som ble Jakob.

Vi har satt opp Trello Board, et for utviklingen og et for andre arbeidsoppgaver som dukker opp i prosjektet.

BitBucket er også satt opp, og alle har fått koblet seg på med SourceTree (eller liknende)

Fredag:

Eclipse er nå oppe å kjører, vi har fått c++ og arduino packages installert. Vi har også fått inn et terminal view så vi slipper å åpne putty for hver gang vi vil lytte til arduino brettet.

Fått startet så vidt på prosjektplan. Lagt inn en ramme for hva vi skal skrive.

Uke 3. 15-21 Januar:

Mandag:

Starter uka med gruppemøte. Ikke mye som blir tatt opp, da arbeidsplanen for uka alt er lagt: Prosjektplan. Første vi jobber med er å se på potensielle sprintene vi kan sette i backlogen. Alle får utdelt individuelle oppgaver i prosjektplanen og begynner å skrive på det. Kristian gjør en del research relatert til scrum, og tar notater vi kan se på i felleskap. Jakob Stian og Martin jobber med prosjektplan, punk 1,2,3,4,5.

Tirsdag:

Fortsetter arbeid med prosjektplan. Vi har også en liten runde med planning poker, får å se på potensielle user cases i utviklingsprosessen vår.
Martin har fått et utkast av rapporten i latex format.
Kristian har jobbet med milepæler og beslutninger, og har startet arbeidet med Gantt skjema.
Stian har jobbet med prosjektplan og doxygen (utviklingsmiljø).
Jakob har jobbet med prosjektplan og timelistene (grafer etc etc).

Første møte med arbeidsgiver 2030.

Onsdag:

Alle jobber med prosjektplan, så vi kan levere et utkast til veileder(Frode).

Torsdag:

Alle har forelesninger. Mulig det må settes opp et kveldsmøte, eller noe fast arbeidstid på kvelden for å ta igjen manglende timer på bachelor.

Fredag:

Brukt dagen på å rette opp i de punktene frode kom med under møte og presentasjon av første utkast til prosjektplan. Avtalt at alle skal gjøre seg ferdig med sitt. samt lage et eget utkast til punkt 2 omfang til mandag, så jobber vi med omfang sammen.

Uke 4. 22-28 Januar:

Mandag - Onsdag

Hele uken ble i hovedsak brukt på prosjektplan.

Onsdag gikk vi muntlig gjennom hele prosjektplanen og prøvde å rette på det vi syns var tåkete eller “muntlig”. Begynne også med forberedelser på Raspberry Pi.

Fredag:

Møte med veileder angående prosjektplan, og arbeid med de problemene/tilbakemeldingene vi fikk.

Uke 5. 29 Jan - 4 Februar :

Jobbet med sprint hovedsakelig. Martin har jobbet med MQTT på Raspberry PI, Stian jobber med å få på plass et fungerende library. Kristian har fått på plass wifimanager på arduino brettene. Jakob har jobbet med Automatisk wifi tilkobling på raspberry pi

Uke 6. 5 - 11 Feb:

Jobbet med sprint hovedsakelig. Martin har jobbet med MQTT på Raspberry PI, Stian jobber med å få på plass et fungerende library. Kristian har fått på plass wifimanager på arduino brettene. Jakob har jobbet med Automatisk wifi tilkobling på raspberry pi

Uke 7. 12 - 18 Feb:

Jobbet med sprint hovedsakelig. Martin har jobbet med MQTT på Raspberry PI, Stian jobber med å få på plass et fungerende library. Kristian har fått på plass wifimanager på arduino brettene. Jakob har jobbet med Automatisk wifi tilkobling på raspberry pi

Uke 8. 19 - 25 Feb:

Diskutert om å ikke bruke automatisk wifi tilkobling, da wifimanager skal brukes til start oppsett av sensorene. Fått alt inn i bitbucket. Føler oss nå relativt ferdig med backend biten av oppgaven, og beveger oss mer inn på den logiske applikasjonen, og visning av data.

Uke 9. 26 Feb - 4 Mars:

Denne uken har gått med på å Reaserche NodeJS Server client oppsett. Etter en del research har vi kommet frem til at Socket.io vil være en god løsning for vårt system. Da det tillatter toveis kommunikasjon, og kan lett settes opp med de fleste applikasjoner som har tilgang på en nettleser.

Uke 10. 5 - 11 Mars:

Fortsetter arbeidet med å sende data fra Clienten (Raspberry Pi) til en server. Bruker for øyeblikket bare en lokal server kjørende på en av våre PCer. Er i tillegg en del arbeid med Mobilutvikling denne uken, pga lab oppgaver som må levers.

Uke 11. 12 - 18 Mars:

Denne uka gikk også med på å sette opp klient/server kommunikasjon. Kommet et godt stykke på vei, og jobber nå med å lagre dataen som blir sendt over fra klienten(Rasp).

Uke 12. 19 - 25 Mars:

Har fått til kommunikasjon mellom node klient og server, og at MQTT listeneren sender data automatisk. Også fått på plass første del av en eksempel nettside som kan vise denne Dataen. Har også startet arbeidet med å få på plass rapporten, innledning nesten ferdig

Uke 13. 26 Mars - 1 April:

Påskeferie denne uka, Er også en del fokus på ny innlevering i Mobile dev faget.

Uke 14. 2 - 8 April:

Finnpusset Innledningen vår for å levere et første utkast til veileder frode. I Samråd med produkteier har vi bestemt at vi snart skal stoppe utviklingen for å fokusere på rapporten, og skal snart fokusere mer på rapporten.

Uke 15. 9 - 15 April:

-Vi har finpusset innledningen (Kap 1.). Har også jobbet med en del bug fixing, det har dukket opp en del duplikasjoner i databasen, men dette ble fikset.

Uke 16. 16 - 22 April:

Dette ble siste uke med utviklingen. Produktet er nå ferdig, og det er fullt fokus på rapporten. Kapittel 2 er ferdig skrevet og levert til Veileder for tilbakemelding. Det er også litt fokus på Mobile Dev fag her for å levere et prosjekt.

Uke 17. 23 - 29 April:

Kapittel 1 og 2 er godkjent og skriver nå ferdig kapittel 3.

Uke 18. 30 April - 6 Mai:

Skriver denne uken alt som mangler i kapitell 3-4-5-6-7 og finpusser for å levere til veileder så vi kan få siste tilbakemelding fra veileder før vi leverer det endelige produktet.

Uke 19. 7 - 13 Mai:

Denne uken ble brukt til å finpusse etter tilbakemelding fra Veileder. Vi har også overført Latex Prosjektet vårt over på NTNU templaten. Jobbed med å få inn ordlister, referanser og vedlegg på en god måte.

Uke 20 14 - 16 Mai:

Denne uken blir kun brukt på gjennomlesning og rettskriving. Leverer etter planen 15 Mai før midnatt.

P Utstyrsliste

Utstyrsliste

1 stk Raspberry Pi model 2B

1 stk USB kabel

Arduino:

1 stk NodeMCU utviklingskort

4 stk Arduino UNO utviklingskort

6 stk ESP8266 WiFi Module

6 stk YP-05 FTDI semiconductor

4 stk DHT22 Sensor

4 stk Force sensitive sensor

4 stk Michrophone sensor

4 stk LDR photocell

4 stk Pir Sensor

4 stk Solderless breadboard

1 stk RFID Reader

1 stk DC: 6.5 - 12V

1 stk Channel Relay module

2stk YwRobot powersupply

1 stk strømmåler

20 stk laser diode module

20 stk buttons

4 stk AA battery holder

∞ stk led lys

∞ stk 10k ohm resistor

∞ stk 220 ohm resistor

∞ stk him to her/him to him/her to her kabel