

Hovedprosjektrapport

ELE301 Elektronikk, bachelor

Pasientarmbånd med tilhørende mobilapplikasjon

av

Aksel Melby Haugen, Robin Omslandseter og Thomas André Rislåa

BAC19/EL-01

Utgavedato: 16. mai 2019

Status: Endelig

Veiledere: Ken Henry Andersen og Lei Jiao

Fakultet for teknologi og realfag

Universitetet i Agder

Grimstad, mai 2019

Bachelor-rapport innen elektronikk, ELE301

Fakultet for teknologi og realfag, Grimstad

Tittel: Pasientarmbånd med tilhørende mobilapplikasjon		Antall sider: 139
		Distribusjon Åpen
Forfattere: Aksel Melby Haugen, Robin Omslandseter, Thomas André Rislåa		Semester: Vår 2019
Nøkkelord: E-helse, Nordic Semiconductor nRF52832-mikrokontroller (SoC), Rohm Semiconductor BH1790GLC pulssensor, TDK InvenSense MPU-6050 akselerometer, app-utvikling (Google Android OS, Android Studio), Bluetooth Low Energy (Bluetooth LE eller BLE), kretskort-design (Altium Designer), IoT (Internet of Things)		
Sammendrag: Ny teknologi er stadig i utvikling for å gjøre fremtidens helsevesen mer effektivt, tryggere og bedre for pasientene[1], [2]. Denne rapporten omhandler et bachelor-prosjekt innen elektronikk ved Universitetet i Agder, som har foregått i samarbeid med konsultentselskapet Bouvet. Hovedoppgaven gikk ut på å utvikle en prototype til et armbånd som aktivt overvåker helsedata til pasienter og overfører denne dataen videre til en mobilapplikasjon slik at dataen kan leses direkte. Kretskortet til armbåndprototypen ble designet i Altium Designer, hvor hovedkomponenten er en nRF52832 SoC som ble programmert i SEGGER Embedded Studio. En tilhørende mobilapplikasjon ble utviklet i Android Studio. Resultatet er en prototype med en fysisk størrelse på $38,40 \times 29,15$ mm som kan overføre data til mobilapplikasjonen over Bluetooth Low Energy og vise puls-data direkte i appen med bedre enn $\pm 10\%$ nøyaktighet. I appen kan helsepersonell lese data fra flere enheter samtidig. Det konkluderes med at Bouvets konsept (<i>Mitt Sykehusopphold</i>) kan effektivisere arbeidsprosesser blant ansatte og at det har potensiale til å gi pasienter bedre innsyn i personlig helsetilstand. Samtidig konkluderes det med at konseptet fremdeles har en lang vei å gå før det kan implementeres på norske sykehus.		
Telefon: +47 37 23 30 00	Adresse: Jon Lilletuns vei 9, 4869 Grimstad	Telefax: +47 38 14 10 01

Forord

Denne rapporten utdyper et prosjekt som har foregått i samarbeid med konsulent-selskapet Bouvet. Vi ønsker å gi en spesiell takk til representanter fra Bouvet som opprinnelig definerte prosjektoppgaven og veiledet oss underveis i prosjektet: Tonje Holand Salgado og Ivar Sønstabø. Vi ønsker også å gi en takk til Eivind Auestad som ga oss mye teknisk hjelp med oppgaven. Vi har lært utrolig mye. Tusen takk for et hyggelig samarbeid.

Underveis i prosjektet har vi også fått god hjelp av universitetslektorer ved Universitetet i Agder. Vi gir en takk til våre veiledere: Lei Jiao og Ken Henry Andersen for god veiledning og teknisk hjelp. Vi takker også universitetslektorer: Geir Jevne og Trond-Ivar Lynghaug. Hjelpen vi fikk fra dere har vært ekstremt nyttig i prosjektsammenheng, og vi tar med oss mye erfaring og kunnskap videre.

Vi vil også gi en takk til representanter fra MeshTech: Torjus Færnes og Eirik Aanonsen for at vi fikk låne ressurser fra deres selskap til oppgaven. Vi takker dere også for nyttige innspill i starten av prosjektperioden. Dere satte oss på riktig spor.

En takk også til e-helse-professor ved UiA: Rune Werner Fensli for engasjementet i vår prosjektoppgave og for nyttige innspill i forhold til utvikling og implementering av e-helse-produkter for sykehus.

Vi ønsker å takke Bouvets andre bachelor-gruppe (også kalt *Rød snor*-gruppen). Takk til Fawad Fazli og Daniel Skomedal Breland for utvekslingen av teknisk informasjon i starten av prosjektperioden og videre i prosjektet.

Takk også til Durapart som dedikerte tid og ressurser til å bistå med lodding av våre prototyper. Hjelpen vi fikk fra dere var essensiell for å komme i mål med den fysiske enheten. Takk til Tore Myrene Rislåa som stilte opp for å gjennomføre lodding ved flere anledninger.

Grimstad 16. mai 2019

Aksel Melby Haugen

Robin Omslandseter

Thomas André Rislåa

Innhold

Forord	iii
Tabeller	viii
Figurer	ix
Formler	xi
1 Innledning	1
1.1 Bakgrunn	1
1.1.1 Mitt Sykehusopphold	1
1.1.2 Pasientarmbånd	2
1.2 Problemdefinisjon	4
1.3 Forutsetninger og begrensinger	6
1.4 Litteraturstudie	7
1.5 Problemløsning	8
1.5.1 Validering av konseptet	8
1.5.2 Fastvare-programmering	8
1.5.3 Kretskort-utvikling	8
1.5.4 App-utvikling	9
1.6 Rapportstrukturen	10
2 Teoretisk bakgrunn	11
2.1 E-helse	11
2.1.1 Noen teknologistandarder	11
2.2 Nåværende markedsteknologi	13
2.2.1 E-helse-armbånd	13
2.2.2 Pulsmålinger	13
2.2.3 Fallsensorer	13
2.3 Mikrokontroller: Nordic Semiconductor nRF52832	14
2.3.1 Nordic Semiconductor	14
2.3.2 nRF52832	14
2.3.3 Fastvareutvikling og feilsøkingstøytøyt	15
2.3.4 Strømbesparing	16
2.3.5 Tredjepartsmoduler	16

2.4	Puls-sensor: Rohm Semiconductor BH1790GLC	17
2.4.1	Rohm Semiconductor	17
2.4.2	BH1790GLC	17
2.4.3	Tekniske spesifikasjoner	18
2.4.4	Virkemåte	18
2.4.5	Initialisering.....	19
2.4.6	Typisk krets	20
2.5	Akselerometer: TDK InvenSense MPU-6050.....	21
2.5.1	Initialisering.....	22
2.5.2	Typisk krets	22
2.6	Bluetooth	24
2.6.1	Bluetooth (IEEE 802.15.1)	24
2.6.2	Bluetooth Low Energy	24
2.6.3	Piconet.....	24
2.6.4	GATT	25
2.6.5	Nordic Bluetooth LE GATT.....	26
2.7	MQTT (Message Queuing Telemetry Transport)	27
2.7.1	CloudMQTT	27
2.8	App-utvikling	28
2.8.1	Operativsystemer og plattformer	28
2.8.2	IDE (Integrert Utviklingsmiljø).....	28
2.8.3	Android Studio	28
2.8.4	Aktiviteter og tjenester	29
2.8.5	Prosjekthierarki i Android Studio.....	29
2.8.6	Nordic Semiconductor.....	30
2.8.7	Versjonshistorikk (Google Android OS).....	30
2.9	Kretskort-utvikling	32
2.9.1	Kretskort:.....	32
2.9.2	Low-dropout regulator (LDO).....	32

2.9.3	Jordplan	32
2.9.4	Avkoblingskondensator	33
2.9.5	Digitalt kretskortdesign	33
2.9.6	Fresing	33
3	Løsning	35
3.1	Krav	35
3.2	Designspesifikasjoner	37
3.2.1	Armbånd	37
3.2.2	App	39
3.3	Implementering	43
3.3.1	Kretskortdesign	43
3.3.2	Prototype 1	46
3.3.3	Prototype 2	49
3.3.4	Armbåndets funksjonalitet	52
3.3.5	Kalkulering av pulsverdier	53
3.3.6	Falldeteksjon	56
3.3.7	«Mitt Sykehusopphold» (mobilapplikasjon)	57
4	Validering og testing	62
4.1	Målenøyaktighet	62
4.2	Strømforbruk	64
4.3	Appens funksjonalitet	68
4.3.1	Validering av konseptet	69
5	Diskusjon	70
5.1	Hovedproblemstilling	70
5.2	Sekundær problemstilling	71
5.3	Fysisk størrelse	72
5.4	Strømforbruk og batteritid	72
5.5	Arbeidsinndeling og tidsbruk.	74
5.6	Videre arbeid	75
6	Konklusjon	79

6.1	Hovedproblemstilling.....	79
6.2	Sekundær problemstilling.....	79
6.2.1	Hovedkonklusjon.....	79
6.2.2	Puls alene er ikke nok.....	79
6.2.3	Trygghet	80
6.3	Prosjektets måloppnåelse.....	80
6.3.1	Fastvare	80
6.3.2	Prototype-kretskort (fysisk enhet)	80
6.3.3	Mobilapplikasjon.....	81
6.4	Videre arbeid	81
	Referanser.....	82
	Vedlegg	89
	Vedleggsliste.....	89
	Vedlegg A: Ordliste og forkortelser	90
	Vedlegg B: Regning av materialer/BOM (Bill of Materials)	92
	Prototype 1	92
	Prototype 2	93
	Vedlegg C: Pressemelding	94
	Vedlegg D: Prosjektplan	95
	Vedlegg E: Gruppekonerter.....	96
	Gruppekonertrakt.....	96
	Felles gruppekonertrakt	97
	Vedlegg F: Møtereferater	98
	Vedlegg G: Timelister.....	119
	Aksel Melby Haugen.....	119
	Robin Omslandseter	124
	Thomas André Rislå	136

Tabeller

Tabell 1: Relevante standarder og tjenester.....	12
Tabell 2: Nøkkelegenskaper for Nordic Semiconductor nRF52832 QFN48 SoC.[44].....	14
Tabell 3: Sammenlikning av to tredjeparts moduler (nRF52832).....	16
Tabell 4: Tekniske spesifikasjoner knyttet til Rohm Semiconductor BH1790GLC.[54].....	18
Tabell 5: Oversikt over registre i BH1790GLC.	19
Tabell 6: Sentrale tekniske spesifikasjoner for MPU-6050.[61]	21
Tabell 7: Oppløsning for sensorer i MPU-6050.[9]	21
Tabell 8: Typisk strømforbruk i MPU-6050. [61].....	22
Tabell 9: Noen registre i MPU-6050.[62]	22
Tabell 10: Komponenter som ble brukt i Prototype 1.	46
Tabell 11: Komponenter brukt i Prototype 2.....	49
Tabell 12: Liste over knapper på prototypen og deres funksjoner.	52
Tabell 13: Sammenlikning av pulsmålinger i sittende stilling.	62
Tabell 14: Sammenlikning av pulsmålinger i stående stilling.....	63
Tabell 15: Sammenlikning av pulsmålinger i gående tilstand.....	63
Tabell 16: Strømforbruk og batteritid i ulike scenarioer/moduser.	67
Tabell 17: Sentral appfunksjonalitet testet på ulike mobiltelefoner.	68
Tabell 18: Liste over vedlegg.	89
Tabell 19: Ordliste og forkortelser.	91
Tabell 20: Regning av materialer for Prototype 1.	92
Tabell 21: Regning av materialer for Prototype 2.	93
Tabell 22: Prosjektdeltakere og roller.	95
Tabell 23: Inndeling av ansvarsområder.	95
Tabell 24: Timeliste for Aksel Melby Haugen.....	123
Tabell 25: Timeliste for Robin Omslandseter.	135
Tabell 26: Timeliste for Thomas André Rislåa.	139

Figurer

Figur 1: Illustrasjon av pasientarmbåndet (rent konseptuelt).....	3
Figur 2: SIS` logo.....	12
Figur 3: FHIRs logo.	12
Figur 4: HL7s logo.	12
Figur 5: DIPS` logo.	12
Figur 6: EVERYs logo.....	12
Figur 7: Tellu sin logo.	12
Figur 8: Nordic Semiconductor nRF52832 QFN48 SoC.[44].....	14
Figur 9: Typisk krets for nRF52832.[45]	15
Figur 10: nRF52 DK.[47]	15
Figur 11: aconno GmbH ACN52832.[49].....	16
Figur 12: Insight SIP ISP1507.[50]	16
Figur 13: aconno GmbHs logo.	16
Figur 14: Insight SiPs logo.....	16
Figur 15: Rohm Semiconductor BH1970GLC.[54]	17
Figur 16: Pulssensorens virkemåte (BH1970GLC).[59]	19
Figur 17: Typisk krets for BH1970GLC.[54].....	20
Figur 18: TDK InvenSense MPU-6050.[60]	21
Figur 19: Typisk krets for MPU-6050.[61]	23
Figur 20: Bluetooths logo.....	24
Figur 21: Illustrasjon av to typer piconet.[72].....	24
Figur 22: Et eksempel på en GATT-profil.	25
Figur 23: MQTT.ORG sin logo.....	27
Figur 24: Android Studios logo.....	28
Figur 25: Prosjekt-hierarki i Android Studio.....	29
Figur 26: Google Android OS versjonshistorikk.[96]	31
Figur 27: LPKF ProtoMat S103.[111].....	34
Figur 28: Forenklet illustrasjon av den planlagte prototypen.	37
Figur 29: Skisse av innlogging-side i «Mitt Sykehusopphold»-appen.....	40
Figur 30: Skisse av pasientenes «hovetside» i appen, også navngitt «Helsemonitor».	41
Figur 31: Skisse av hovetsiden til helsepersonell i «Mitt Sykehusopphold»-appen.....	42
Figur 32: Størrelsesforhold for ACN52832 hentet fra datablad.[51]	44
Figur 33: Egenutviklet fotavtrykk for ACN5283.	44
Figur 34: Skjematiske-komponent for ACN52832.....	44

Figur 35: Prototype 1.....	46
Figur 36: Kretsskjematikk for Prototype 1.....	47
Figur 37: Undersiden av kretskortdesignet (Prototype 1).	48
Figur 38: Oversiden av kretskortdesignet (Prototype 1).	48
Figur 39: Prototype 2.....	49
Figur 40: Kretsskjematikk for Prototype 2.....	50
Figur 41: Undersiden av kretskortdesignet (Prototype 2).	51
Figur 42: Oversiden av kretskortdesignet (Prototype 2).	51
Figur 43: Knapper på prototypen nummerert.....	52
Figur 44: Graf som viser rådata (oransje) sammenliknet med utglattet data (blått).	53
Figur 45: Graf med rådata, gjennomsnitt og firkantpuls.	54
Figur 46: Graf som illustrerer falldetektering.	56
Figur 47: Logo for appen «Mitt Sykehusopphold».	57
Figur 48: Implementert innloggingside i MSO-appen.	57
Figur 49: ACL-mønster for å gi alle MQTT-brukere lese- og skrivetilgang.....	58
Figur 50: Implementert hovedside for pasienter i MSO-appen.....	59
Figur 51: Implementasjon av helsepersonell-siden i MSO-appen.....	61
Figur 52: Oppkobling av oscilloskop for å måle strømforbruk.	65
Figur 53: Oscilloskop-måling når BLE er tilkoblet og armbåndet klarer å lese puls.	66
Figur 54: Oscilloskop-måling når BLE er tilkoblet, men puls ikke kan leses.	66
Figur 55: Oscilloskop-måling når enheten er i «advertising»-modus.	67
Figur 56: Appskisse av pasientenes hovedside i «Mitt Sykehusopphold».	76
Figur 57: Ideelt konsept for kommunikasjonssystem.....	78
Figur 58: Prototypen til e-helse-armbåndet.....	94
Figur 59: Gantt-chart/fremdriftsplan.	95

Formler

Formel 1: Effekttap i en LDO-regulator.[99]	32
Formel 2: Generell kalkulering av puls.	53
Formel 3: Kalkulering av puls i nåværende implementasjon av prototypen.	54
Formel 4: Ohms lov.....	64
Formel 5: Kalkulasjon av batteritid.[115]	64

1 Innledning

Denne rapporten omhandler et bachelor-prosjekt innen elektronikk ingeniørfag ved *Universitetet i Agder*, våren 2019. Prosjektets tittel er *Pasientarmbånd med tilhørende mobilapplikasjon* og har foregått i samarbeid med konsultentselskapet *Bouvet Norge AS* (Bouvet) med en stasjonert avdeling i Arendal. Bouvet er oppdragsgiver for oppgaven. Prosjektet har gått ut på å utvikle en prototype for et armbånd som kan brukes av innlagte pasienter på sykehus eller sykehjem for å aktivt overvåke helseverdier og overføre denne dataen til en mobilapplikasjon, hvor både pasient og helsepersonell kan lese av dataen direkte. Dette vil potensielt kunne gjøre det enklere å overvåke pasienters helsetilstand og gi pasienter bedre innsikt i sin egen helsetilstand.

1.1 Bakgrunn

Det jobbes kontinuerlig med utvikling av ny teknologi innen helsesektoren som sikter mot å gjøre det fremtidige helsevesenet mer kostnadseffektivt, tryggere og bedre for pasienter[1], [2]. Norske helseforetak (HF) sier selv at de ønsker en økt satsning på pasientopplevelsen[3], [4]. I fremtiden vil vi sannsynligvis kunne se en økning av IoT-, AI og andre e-helse-løsninger på norske sykehus[2], [5].

Bakgrunnen til prosjektet presentert i denne rapporten har utspring i en problemstilling definert av oppdragsgiver; Bouvet. Bouvet har kartlagt pasientopplevelsen på sykehus og har sett på mulige løsninger som kan gjøre oppplevelsen bedre. Som en hoved-tittel for det fullstendige konseptet til Bouvet vil vi videre i teksten referere til *Mitt Sykehusopphold*.

Prosjektgruppen gjennomførte sammen med Bouvet et styringsmøte med flere avdelingsledere ved Sørlandet Sykehus HF (SSHF) 6. mars 2019. På møtet presenterte Tonje Holand Salgado Bouvets konsept. Teksten nedenfor er i stor grad basert på presentasjonen Tonje foreholdt på dette møtet.

1.1.1 Mitt Sykehusopphold

Konseptet *Mitt Sykehusopphold* består av flere deler, hvor den mest sentrale er en mobilapplikasjon (app) som gir pasienter bedre innsyn i sitt eget sykehusopphold. I denne appen skal pasienter blant annet kunne registrere egen brukerinformasjon og ha tilgang til informasjon om prosedyrer, avdelingsrutiner, legetimer og prøveresultater. Pasienter skal kunne se målinger av direkte helsedata, slik som puls, blodtrykk, temperatur og bevegelse. Pasientene skal få en indikasjon på om prøveresultater og helseverdier ser normale ut i forhold til personlige normalverdier. I tillegg skal pasienter kunne sende varsler med flere alvorlighetsgrader til helsepersonellet. Dette varselssystemet fungerer da som et

supplement til *rød snor*-konseptet vi ser på norske sykehus i dag. Nedenfor er en liste over appfunksjonalitet rettet mot pasienter:

- Pasienter skal kunne legge inn egne personlige opplysninger.
- Pasienter skal ha en oversikt over prosedyrer, avdelingsrutiner, legetimer og prøveresultater.
- Pasienter skal kunne se direktemålinger av sine helseverdier med indikasjon på om dataen ser normal ut.
- Pasienter skal kunne sende varsler og forespørsler til helsepersonellet.
- Pasienter skal kunne legge inn ønsker og planlegge middagsmåltider.

Fra helsepersonellet sin side, vil man i appen kunne legge inn pasientnotater og prosedyrer for hver enkelt pasient. Man vil få notifikasjoner dersom pasienter sender ulike typer forespørsler eller om en alarm utløses. I tillegg får helsepersonell mulighet til å overvåke pasienters helsedata fra ekstern lokasjon. Helsepersonell vil dermed kunne prioritere tidsbruk bedre og effektivisere enkelte arbeidsprosesser. Man kan også slippe ekstra gåturer dersom pasienter på forhånd definerer hva slags assistanse som behøves. For eksempel kan man ta med seg medisiner på vei bort, fremfor å gå tilbake og hente.

I styringsmøtet med SSHF (nevnt tidligere), viste avdelingslederne fra SSHF interesse for Bouvets konsept. De bekreftet at konseptet potensielt kan effektivisere enkelte arbeidsprosesser blant ansatte.

1.1.2 Pasientarmbånd

Dette prosjektet tar kun for seg en liten del av Bouvets konsept, hvor det er tenkt at pasienter skal utstyres med et pasientarmbånd. Dette armbåndet vil kontinuerlig overvåke pasienters helseverdier, slik som puls, blodtrykk, temperatur og bevegelse. Potensielt vil man også kunne bruke pasientarmbåndet som en personlig ID-tag til låsing/opp-låsing av skaper eller kjøp i kantinen. Denne rapporten omhandler utvikling av et slikt pasientarmbånd, samt en mobilapp med funksjonalitet som supplerer dette armbåndet. Se *Figur 1* for en illustrasjon av hvordan et slikt armbånd kan utformes som ferdig produkt.



Figur 1: Illustrasjon av pasientarmbåndet (rent konseptuelt).

1.2 Problemdefinisjon

Dette prosjektet har i hensikt å utvikle et produkt som potensielt kan forbedre pasientopplevelsen og effektivisere arbeidsprosesser på norske sykehus. Hovedproblemstillingen i dette prosjektet er å finne ut hvordan man kan utvikle et pasientarmbånd med tilhørende mobilapplikasjon og finne ut hvordan et slikt system kan implementeres på norske sykehus. En prototype til armbåndet og en demo-app skal dermed utvikles. En annen problemstilling er å finne ut om et slikt system alene kan forbedre pasientopplevelsen eller effektivisere arbeidsprosesser blant ansatte. Det bør også drøftes om det fullstendige konseptet av *Mitt Sykehusopphold* kan forbedre pasientopplevelsen eller arbeidsprosesser blant ansatte.

Hovedproblemstilling:

Å utvikle et e-helse-armbånd som aktivt overvåker helsedata og viser denne dataen direkte i en dedikert mobilapplikasjon.

Resultatet som er forventet, er at prosjektgruppen er i stand til å produsere et prototype-kretskort (PCB) med en puls-sensor, og som kan gi en rimelig tilnærming til faktisk pulsverdi. Det er forventet at den endelige prototypen vil være fysisk større enn det som vil være ideelt i en praktisk setting på grunn av begrenset tilgang til utstyr, men at den allikevel vil kunne være relativt liten. Det er også forventet at prototypen vil kunne kommunisere med en egenutviklet mobilapplikasjon via *Bluetooth Low Energy* (Bluetooth LE eller BLE) og vise målinger direkte i denne appen på minst én mobiltelefon av gangen.

Sekundær problemstilling:

Vil kontinuerlig overvåking av pasienters helseverdier via et pasientarmbånd bedre pasientopplevelsen eller effektivisere arbeidsprosesser blant ansatte?

Her er en kort oppsummering av hovedmålene til prosjektet:

- Utvikle et prototype-kretskort for et pasientarmbånd.
- Lese pulldata på håndledd og kalkulere tilnærmet puls-verdi med $\pm 10\%$ målenøyaktighet.
- Designe en mobilapplikasjon hvor data fra armbåndet kan leses direkte.
- Finne ut hvordan et slikt system kan implementeres på sykehus.
- Drøfte om et slikt system vil kunne bedre pasientopplevelsen eller effektivisere arbeidsprosesser blant ansatte.

1.3 Forutsetninger og begrensinger

Prosjektet bør foregå i tråd med oppdragsgiveren Bouvet og ta hensyn til Bouvets opprinnelige visjon (Mitt Sykehusopphold). I tillegg må det tas hensyn til innspill fra veiledere Ken Henry Andersen og Lei Jiao. Ettersom dette er et bachelor-prosjekt innen elektronikk, var det viktig at oppgaven ble utformet slik at prosjektgruppen kunne utvikle noe som viser god elektronikk-faglig kompetanse. Oppgaven ble derfor definert i dialog mellom Bouvet og prosjektgruppen med innspill fra veiledere.

Det må tas hensyn til tilgjengelige ressurser til prosjektoppgaven. Blant annet må det tas hensyn til at prosjektgruppen (kun) består av tre personer. Ingen av gruppemedlemmene har tidligere erfaring med app-utvikling og ingen av gruppemedlemmene har tidligere erfaring med utvikling av *printkort*. Dermed vil disse delene av prosjektet kreve en del arbeid, rett og slett for å opparbeide seg kompetanse innen disse feltene.

Til disposisjon hadde prosjektgruppen et budsjett på rundt tusen kroner disponert av universitetet, noe som setter en viss begrensning til innkjøp av komponenter. Ved utvikling av kretskort, må det brukes utstyr tilgjengelig ved universitetet blant annet for å frese kretskortet. Universitetet disponerer en *LPKF ProtoMat S103* til fresing av tosidige kretskort. At kretskortet maksimalt kan være tosidig fører til noen begrensinger med tanke på fysisk størrelse. Man må gjøre kortet større for å få plass til nødvendige koblinger. Komponentene må også være utformet slik at de er *enkle nok* å lodde på kretskortet med vanlig loddeutstyr.

I forprosjekt-perioden definerte prosjektgruppen noen begrensninger i forhold til hva som burde oppnås og hva som var utenfor rekkevidde. En fullstendig implementasjon av appen *Mitt Sykehusopphold* er tenkt utenfor rekkevidde. Pasientarmbåndet kommer heller ikke til å være klar for kommersiell produksjon etter prosjektperioden. Det vil ikke bli implementert noe sikkerhets-system eller innlogging-system med *MinID* eller *BankID*. Sikkerhetsaspektet og testing med tanke på en faktisk implementasjon av løsningen på sykehus, er ikke en del av dette prosjektarbeidet.

1.4 Litteraturstudie

I starten av prosjektperioden var det viktig å tilegne seg kunnskap rundt de forskjellige prosjektdelene, for å kunne løse hovedproblemstillingen. Prosjektgruppen var derfor i dialog med flere personer med relevant faglig kompetanse. Prosjektgruppen gjennomførte blant annet en *workshop* 24. januar 2019 på bedriften *MeshTech*. På workshopen ga Eirik Aanonsen tips om mikrokontroller og batteri/ladesystem for armbånd-prototypen. Gruppen hadde også dialog med Eivind Auestad fra Bouvet rundt app-utvikling. 8. mars 2019 hadde gruppen et møte med e-helse-professor ved UiA, Rune Werner Fensli, som ga teknisk informasjon om hvilke standarder som brukes innen helsevesenet og hvilke hensyn som må tas om man ønsker å utvikle et *pasientarmbånd*. To av de mest relevante standardene for dette prosjektet vil eventuelt være *Social Care Alarm Internet Protocol (SCAIP)*[6] og *Fast Healthcare Interoperability Resources (FHIR)*[7].

Prosjektgruppen brukte en periode på å lære seg opp i forskjellig programvare, blant annet det integrerte utviklingsmiljøet (IDE)-en *SEGGER Embedded Studio* for programmering av vanlige *ARM*-mikrokontrollere[8] og *Altium Designer* for kretskort-design[9], [10]. Det ble også brukt en del tid til å lære IDE-en *Microsoft Visual Studio*[11] med UI-verktøyet *Xamarin.Forms* for flerplattform-apputvikling[12]. Visual Studio ble senere byttet ut *Android Studio*[13] og kodespråket *Java*[14] da dette ga enkelte fordeler i forhold til utvikling rettet mot Android-plattformen. Informasjon rundt disse ulike programvarene ble funnet via opplæringsvideoer på nettet, hovedsakelig på video-delings-plattformen; *YouTube*[15]. I tillegg ble den offisiell dokumentasjon for *Xamarin.Forms*[16] og den offisielle dokumentasjon for Android apputvikling[17] brukt. For å lære seg kodespråkene, som var ukjente fra før av, ble det brukt en app/nettside som heter *SoloLearn*, hvor man kan lære kodespråk gratis[18].

I starten av prosjektperioden ble det brukt en del tid på å finne ut hvilke komponenter som skulle benyttes for prototypen. Blant annet ble det hentet informasjon fra *Nordic Semiconductor* sin hjemmeside angående deres mikrokontroller SoC-varianter[19] og tilgjengelige tredjeparts-moduler[20]. For å finne ut hvilke puls-sensorer og andre komponenter som er tilgjengelige, ble nettbutikker som *digikey.no*[21], *mouser.com*[22] og *farnell.com*[23] undersøkt. Prosjektgruppen leste seg opp på informasjon tilgjengelig i eventuelle datablader for de ulike komponentene som ble oppdaget.

1.5 Problemløsning

Problemløsningen består av flere arbeidsprosesser, og forskjellig teknisk utstyr må benyttes for å gjennomføre de ulike prosessene. Først og fremst må det fastslås hvilke fysiske komponenter og teknisk utstyr som skal benyttes. Deretter må prototypen samt appen utvikles ved hjelp av verktøyene. Kunnskap rundt de ulike verktøyene må tilegnes ved hjelp av faglig kunnskap fra veiledere eller universitetslektorer, samt ressurser tilgjengelig på internett. Slik informasjon kan blant annet være opplæringsvideoer, artikler, datablader, offisiell dokumentasjon eller nettforum. Det ble valgt å dele opp oppgaven i fire deler: Validering av konseptet, fastvare-programmering, kretskort-utvikling og app-utvikling. Disse delene er videre beskrevet.

1.5.1 Validering av konseptet

For å kunne validere potensialet til pasientarmbåndet med tilhørende mobilapp, samt Bouvets konsept; *Mitt Sykehusopphold*, må gruppen konversere med personer som har den rette kunnskapen om hvordan et slikt system rent praktisk ville kunne fungert på et sykehus kontra dagens løsning. Prosjektgruppen deltok derfor på styrmøte med flere avdelingsledere ved SSHF i Arendal 6. mars 2019. Gruppen hadde også et møte med e-helse-professor ved UiA: Rune Werner Fensli 8. mars 2019. I tillegg har gruppen hatt kontakt med en sykepleier ved SSHF i Kristiansand. Prosjektgruppen hadde også ønsket å kunne teste applikasjonen i et simulert testmiljø, men dette ble ikke gjennomført.

1.5.2 Fastvare-programmering

De fysiske komponentene inkluderer blant annet mikrokontroller-chip og potensielle sensorer. I dette prosjektet ble det valgt å bruke en *nRF52832*-mikrokontroller-chip fra *Nordic Semiconductor*. Til å programmere mikrokontrolleren benyttes et integrert utviklingsmiljø (IDE), slik som f.eks. *SEGGER Embedded Studio*. Først brukes et utviklingskort (DK) sammen med valideringskort for å teste de ulike sensorene. *nRF52832s* utviklingskort heter *nRF52 Development Board*. Programmet utvikles slik at mikrokontrolleren på DK-en kommuniserer med de ulike sensorene og håndterer dataen som blir mottatt. Her må også andre fysiske funksjoner programmeres, slik som knapper eller lys-dioder (LED-er). Programmet testes og optimaliseres før det blir programmert (*flashet*) på en eventuell kretskort-prototype. Programmet som ligger *fast* inne i mikrokontrolleren blir gjerne kalt for *fastvare* eller på engelsk: *firmware*.

1.5.3 Kretskort-utvikling

Å utvikle en kretskort-prototype er en annen del av arbeidsprosessen. Nødvendige komponenter må bestilles inn. Et kretskort (PCB) designes ved hjelp av en dedikert programvare for dette, slik som f.eks. *Altium Designer*. Først designes det elektriske kretssystemet, og deretter designer man det fysiske kretskortet med utgangspunkt i den elektriske kretsen. Som utgangspunkt kan man følge *typiske kretser*

som ofte står opptegnet i *datablader*. Her må det også tas hensyn til fysiske størrelser og hvilke komponenter og moduler som tas i bruk. Når et PCB-design er ferdig utviklet, er det lurt å få en bekreftelse på at PCB-en ser *riktig ut* av erfarne universitetslektorer, før kortet kan freses i en dedikert maskin for dette. Skolen disponerer her en *LPKF ProtoMat S103* for lodding av kretskort med ett lag. Etter man har et fysisk kretskort, må fysiske komponenter loddess på. Mikrokontrolleren må deretter programmeres (eller *flashes*), før man kan teste og validere om PCB-en fungerer.

1.5.4 App-utvikling

For å utvikle en mobilapp må man først bestemme hvilken plattform man ønsker å utvikle appen for. I dette prosjektet ble det bestemt å utvikle appen primært for *Google Android OS*. Deretter finnes det flere IDE-er hvor appen kan utvikles, slik som f.eks. *Android Studio*[13]. Appen kan installeres og testes med feilsøkingssverktøy på mobiltelefon eller emuleres direkte på PC/Mac.

Her er en oppsummering av stegene for å utvikle armbåndprototypen og app-demoen:

- Velge mikrokontroller-modul og sensorer.
- Bestille komponenter.
- Programmere mikrokontroller. (SEGGER Embedded Studio)
- Designe kretskort. (Altium Designer)
- Designe og programmere mobilapplikasjon. (Android Studio)
- Frese kretskort og montere komponenter.
- Teste og validere.

1.6 Rapportstrukturen

Denne rapporten er strukturert på en slik måte at den skal være enkel og oversiktlig å lese. Det er totalt seks hovedkapitler. I kapittel 2, vil det bli gitt en teoretisk bakgrunn for elementene som løsningen består av. I kapittel 3 forklares designspesifikasjonene og den endelige implementasjonen. Testing og validering av produktet presenteres i kapittel 4. I kapittel 5 diskuteres arbeidsprosessen, resultatene av prosjektarbeidet og hva som kunne ha blitt gjort annerledes. I kapittel 6 konkluderes arbeidet og vi presenterer mulige fremtidige forbedringer og arbeid som kan gjøres. Alle hovedkapitler starter med en kort ingress som forklarer hva hvert kapittel omhandler.

2 Teoretisk bakgrunn

Dette kapitlet omhandler teknisk teori relevant for oppgaveløsningen, blant annet e-helse-standarder, mikrokontrollere, sensorer, Bluetooth Low Energy, MQTT, apputvikling og kretskort-utvikling. Denne teorien blir senere brukt videre i kapittel tre og fire.





2.1 E-helse

2.1.1 Noen teknologistandarder

Dersom et e-helse-produkt skal kunne installeres i helsevesenet, må produktet ta hensyn til de ulike standardene som brukes innen helsevesenet i dag. Prosjektgruppen gjennomførte et møte med e-helse-professor og forsker for *Senter for eHelse* ved Uia, Rune Werner Fensli. På møtet nevnte Rune flere standarder og systemer som er tatt i bruk i helsevesenet.

Teksten nedenfor tar utgangspunkt i møte med e-helse-professor Rune Werner Fensli 8. mars 2019. Alle standardene og tjenestene som nevnes nedenfor er listet i *Tabell 1*.

En av standardene som brukes innen helsesektoren i dag er Social Care Alarm Internet Protocol (SCAIP) laget av Swedish Standard Institute (SIS)[6]. SCAIP-protokollen brukes primært til alarmsystemer, og forklarer hvordan en melding mellom alarm-sender og alarm-mottaker kan sendes over et IP-kommunikasjonsnettverk, slik som internett[6]. En annen standard som brukes er Fast Healthcare Interoperability Resources (FHIR) av Health Level Seven International (HL7). FHIR er en internasjonal standard som benyttes for å utveksle ulike type helsedata mellom ulike organisasjoner[7]. Direktoratet for e-helse anbefaler at aktører innen helsesektoren benytter seg av FHIR-grensesnittet for datadeling[24]. Norske sykehus bruker Distribuert Informasjons og Pasientdatasystem i Sykehus (DIPS) som leverandør av Elektronisk pasientjournal (EPJ)[25]. MetaVision eller Elektronisk kurveprosjekt er en del av EPJ-prosjektet hvor et dataprogram skal kunne føre pasienters medisinkurve elektronisk[26]. Helse Nord signerte i desember 2014 en avtale med systemleverandøren EVRY om anskaffelse av MetaVision-løsningen[27] og flere norske sykehus bruker denne løsningen i dag[26]. Sky-plattformen TelluCloud ble også nevnt på møtet med Rune. TelluCloud-plattformen gjør det enklere å koble sammen og styre flere IoT-enheter med standard API-er og tjenester[28]. Alle standardene og tjenestene nevnt ovenfor er listet i *Tabell 1* på neste side.

Standard/tjeneste	Utvikler/leverandør	Funksjon	Lisens
Social Care Alarm Internet Protocol (SCAIP)	Swedish Standatd Insitute (SIS)  <i>Figur 2: SIS` logo.</i>	Protokoll for håndteringer av alarm- signaler[6].	Dokumentasjonen kan kjøpes for 1025 SEK.[6]
Fast Healthcare Interoperability Resources (FHIR)  <i>Figur 3: FHIRs logo.</i>	Health Level Seven International (HL7)  <i>Figur 4: HL7s logo.</i>	Formatering av ulike type helsedata og API for datautveksling mellom foretak[29].	CC0[29]
Distribuert Informasjons og Pasientdatasystem (DIPS)	DIPS AS  <i>Figur 5: DIPS` logo.</i>	Leverer Elektronisk pasientjournal- systemer (EPJ).[25]	Utilgjengelig.
MetaVision, Elektronisk kurveprosjekt	EVRY[27]  <i>Figur 6: EVRYs logo.</i>	Dataprogram for elektronisk føring av pasienters medisinkurve.[26]	Utilgjengelig.
TelluCloud	Tellu IoT AS  <i>Figur 7: Tellu sin logo.</i>	Plattform for standard API-er og tjenester for IoT-enheter.[28]	Utilgjengelig.

Tabell 1: Relevante standarder og tjenester.

2.2 Nåværende markedsteknologi

2.2.1 E-helse-armbånd

På markedet finnes det et stort utvalg med trådløse armbånd og digitalklodder som er i stand til å måle ulike helsedata[30]. Eksempler noen velkjente produkter er blant annet *Apple Watch* og *Fitbit* sine aktivitetsarmbånd/klokker[31], [32]. Denne teknologien er populært brukt innenfor fitness, personlig helse og idrett for å måle aktivitet over tid. De første kroppsbårne puls-sensorene ble tatt i bruk helt tilbake i 1981[30]. Moderne produkter er i stand til å telle skritt, regne ut kaloriforburning og måle puls. Nyere versjoner av *Apple Watch* er også i stand til å utføre elektrokardiografi (EKG)-målinger[33]. Det er utviklet enkelte enheter som sitter på håndleddet som er utviklet for å måle blodtrykk samt oksygenmetningen i blodet. Slike armbånd får gjennomført målinger, men er ikke sett på som spesielt nøyaktige sammenlignet med annet utstyr som blir benyttet[34], [35].

2.2.2 Pulsmålinger

Målingene som kommer fra et aktivitetsarmbånd vil sjelden gi en god indikasjon på om en person har et fysisk problem eller ikke. En pulsmåling fra et aktivitetsarmbånd vil kunne gi et relativt nøyaktig indikasjon på pulsverdi[36], men pulsen til en person varierer basert på mange forskjellige faktorer[37]. Blant annet påvirkes pulsen av fysisk bevegelse, sykdom, hva en person har spist, fysisk form, lufttemperatur og stressnivå[37], [38]. I tillegg er normalverdier for puls svært individuelt[38]. Det er dermed svært vanskelig å bruke puls til å gi indikasjon på om det er noe galt med en persons helsetilstand.

2.2.3 Fallsensorer

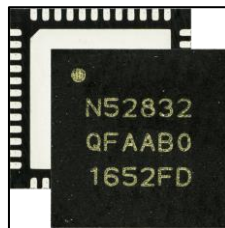
Å registrere et fall gjennom bruk av sensorer har vist å være problematisk[39]. Blant annet er det vanskelig å definere hvilke bevegelser som inngår i et fall[39]. Hastigheten på fall varierer også betraktelig. Løsninger som er på markedet for øyeblikket inkluderer sensorer som detekterer fall ved enkelte bevegelser eller ved bruk av knapper[40], [41]. Forskingsinstitusjonen SINTEF[42] i samarbeid med Tellu[28] (nevnt tidligere) er i utviklingsfasen av en fallsensor som baserer seg på lufttrykk[39]. Denne fallsensoren fungerer ved å ha en sensor i rommet og en på pasienten[39]. Dersom trykkendringene i sensoren festet til pasienten øker, detekteres et fall[39].

2.3 Mikrokontroller: Nordic Semiconductor nRF52832

2.3.1 Nordic Semiconductor

Nordic Semiconductor er et norsk selskap med lokalisert i Trondheim som utvikler integrerte kretser[43]. Hovedfokuset til Nordic Semiconductor ligger i trådløs teknologi som blant annet har blitt tatt i bruk av internasjonale selskaper som *Logitech* og *Microsoft*[43]. Blant produktene som Nordic Semiconductor leverer, er *nRF52 System on Chip (SoC)*-serien[44].

2.3.2 nRF52832

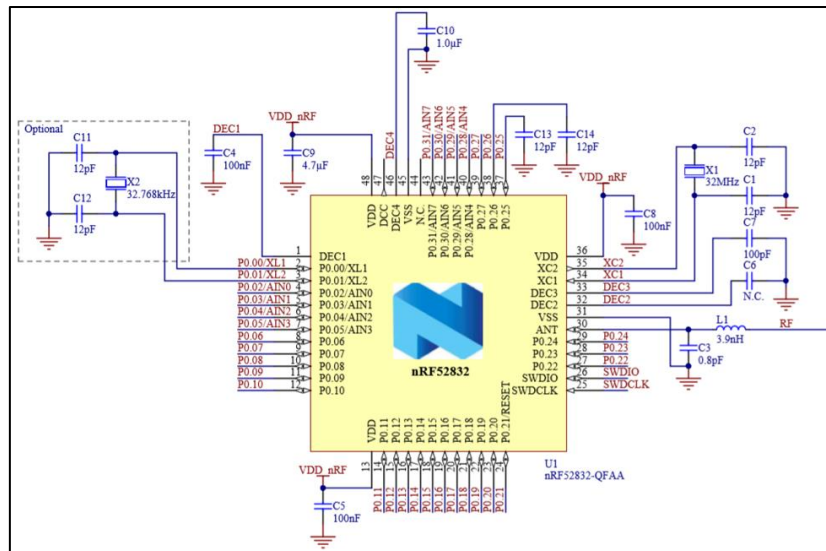


Figur 8: Nordic Semiconductor nRF52832 QFN48 SoC.[44]

Nordic Semiconductor har et bredt utvalg med SoC-enheter som er designet for lavt strømforbruk og dataoverføring over korte avstander[19]. Slike produkter tilhører *nRF51*- og *nRF52* SoC-seriene[44]. *nRF52832* er betegnet som en *mainstream* (hovedstrømmen) SoC i *nRF52*-serien. Dette vil si at det er en chip med medium klasses ytelse og passer til de fleste scenarioer[44]. *Figur 8* viser et bilde av chip-en. Spesifikasjoner for *nRF52832* står listet i tabellen nedenfor (se *Tabell 2*). Nordic Semiconductors *nRF52832* SoC har typisk krets som vist i *Figur 9*.

Egenskap	Gjeldende for nRF52832
Prosesor	ARM Cortex-M4F
Kjernerokke	64 MHz
Flashminne	512/256 KB
RAM	64/32 KB
Radiofrekvens	2.4 GHz
Bluetooth	Bluetooth 5, Bluetooth mesh ANT
Funksjoner	UART, SPI, TWI, PDM, I2S, PWM, 12-bit ADC, NFC-A

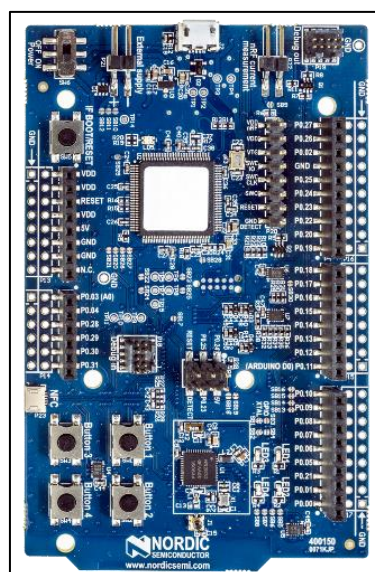
Tabell 2: Nøkkelegenskaper for Nordic Semiconductor nRF52832 QFN48 SoC.[44]



Figur 9: Typisk krets for nRF52832.[45]

2.3.3 Fastvareutvikling og feilsøkingstverktøy

Nordic Semiconductors SoC-produkter har støttes av de integrerte utviklingsmiljøene *IAR Embedded Workbench*, *Keil MDK* og *SEGGER Embedded Studio*. Disse utviklingsmiljøene blir brukt for å programmere chipene som Nordic Semiconductor leverer[46]. For å gjennomføre feilsøking og utvikle programmer for mikrokontrollere basert på Nordic Semiconductor-chiper, brukes utviklingskort (DK). For nRF52832 (og *nRF52810*) brukes DK-en *nRF52 DK*[47] (se *Figur 10*). Dette utviklingskortet baserer seg på en nRF52832 SoC og har GPIO-innganger tilsvarende som for denne SoC'en[47]. Utviklingskortet benyttes også når man ønsker å overføre (*flashe*) programmet til en ekstern nRF52-SoC (kun chipen)[5].




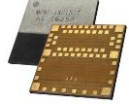


Figur 10: nRF52 DK.[47]

2.3.4 Strømbesparing

nRF52832 baserer seg på en *ARM Cortex-M4F* prosessor[44]. Denne arkitekturen bruker noe som kalles *Nested Vectored Interrupt Controller (NVIC)* som vekker prosessoren opp fra søvnmodus ved interruptus[48]. Dette er en nyttig funksjon i applikasjoner hvor strømbruk er en begrensning. Prosessoren kan settes i søvnmodus, eller det som kalles *Wait for Event (WFE)* eller *Wait for Interrupt (WFI)*, frem til et avbrudd eller en hendelse vekker opp prosessoren igjen[48].

2.3.5 Tredjepartsmoduler

For Nordic Semiconductor sine chiper, finnes det et bredt utvalg med tredjepartsmoduler. Slike moduler bygger på nRF-mikrokontrollere og har forskjellige utlegg for blant annet radiofrekvens (RF)-antenne, antall *General Purpose Input and Output (GPIO)*-pins og ulike størrelsesfaktorer. *aconno GmbH ACN52832* og *Insight SiP ISP1507* er to eksempler på slike moduler[20]. Begge disse bruker en nRF52832-chip, men de har ulik formfaktor og ulik plassering av RF-antennen. Se *Tabell 3* for en sammenlikning av disse to tredjepartsmodulene.

Spesifikasjon	ACN52832	ISP1507
Bilde	 <p><i>Figur 11: aconno GmbH ACN52832.[49]</i></p>	 <p><i>Figur 12: Insight SiP ISP1507.[50]</i></p>
Produsent	<p>Aconno GmbH</p>  <p><i>Figur 13: aconno GmbHs logo.</i></p>	<p>Insight SiP</p>  <p><i>Figur 14: Insight SiPs logo.</i></p>
Størrelse	20,2 × 25 × 3 mm[51]	8 × 8 × 1 mm[52]
RF-antenne	PCB[51]	Integrert[52]

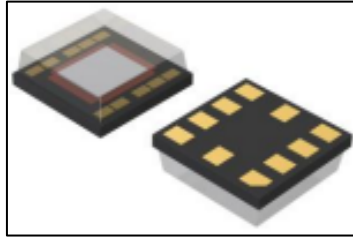
Tabell 3: Sammenlikning av to tredjeparts moduler (nRF52832).

2.4 Puls-sensor: Rohm Semiconductor BH1790GLC

2.4.1 Rohm Semiconductor

Rohm Semiconductor (eller bare *Rohm*) er et japansk selskap etablert i 1958 som utvikler av elektronisk deler[53]. Selskapet baserte seg lenge på levering av motstander, men leverer i dag en rekke sensorer for det kommersielle markedet[53].

2.4.2 BH1970GLC



Figur 15: Rohm Semiconductor BH1970GLC.[54]

I dette prosjektet ble det tatt i bruk en pulssensor fra Rohm Semiconductor som heter BH1790GLC[54] (se *Figur 15*). Denne pulssensoren har en egen virkemåte mens det finnes andre pulssensorer på markedet som fungerer annerledes. For eksempel finnes MAX30102 fra Maxim Integrated[55]. Den største forskjellen mellom disse to sensorene er at BH1790GLC bruker grønt lys[54] mens MAX30102 bruker infrarød til å gjennomføre pulsmålinger[55]. Grønt lys er oftere brukt enn infrarødt, selv om infrarødt lys har noen fordeler[56]. Blant annet kan infrarødt lys penetrere dypere i huden slik at man potensielt kan måle hydrering, muskelmetning, hemoglobin-nivå og mer[56]. Svakheten med infrarødt lys er at det lett kan komme støy på målingene[56], spesielt om sensoren plasseres på handledet hvor det kan forekomme mye bevegelse.

2.4.3 Tekniske spesifikasjoner

Se *Tabell 4* for tekniske spesifikasjoner knyttet til BH1790GLC-pulssensoren. Alle spesifikasjoner er hentet fra det offisielle databladet til sensoren.

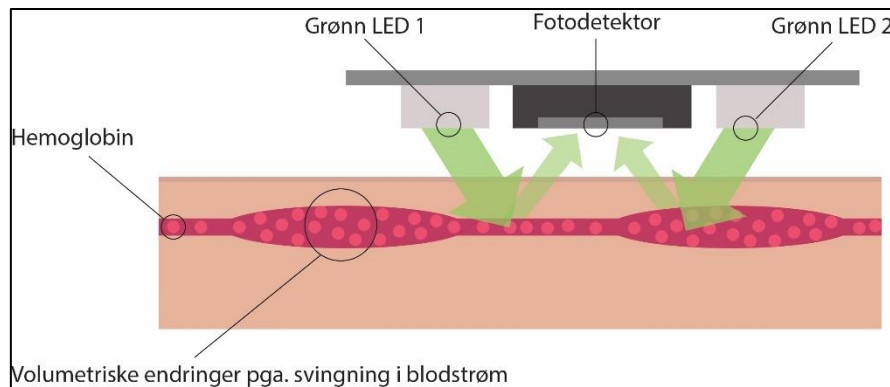
Spesifikasjon	Verdi
Spenningsrekkevidde på VCC 1	2,5V – 3,6V
Spenningsrekkevidde på VCC 2	1,7V – 3,6V
Typisk strømforbruk	200 μ A
Typisk strømforbruk i ventemodus	0,8 μ A
Operativt temperaturområde	-20°C til + 85°C
Fysisk chip-størrelse	2,8 mm \times 2,8 mm \times 1,00 mm
Grensesnitt (dataoverføring)	1,8 V I ² C/TWI

Tabell 4: Tekniske spesifikasjoner knyttet til Rohm Semiconductor BH1790GLC.[54]

BH1790GLC har dedikerte utganger for LED-drivere (*LED1* og *LED2*). Disse utgangene driver de grønne LED-ene som brukes av sensoren. Kommunikasjon med sensoren kan gjøres via grensesnittet som heter *Inter-Integrated Curcuit (I²C)* av *Philips Semiconductor*[57]. En lik implementasjonen av dette grensesnittet kalles også gjerne for *Two-Wire Interface (TWI)*[57].

2.4.4 Virkemåte

BH1790GLC fungerer ved at en fotodetektor fanger opp reflektert lys fra to grønne LED-er som plasseres på siden av sensoren (se *Figur 17*). En god porsjon av det grønne lyset blir absorbert i huden, men deler av lyset vil også penetrere huden og bli reflektert av vev[56]. *Hemoglobin* er det røde fargestoffet i blodet som bidrar til å frakte oksygen rundt i blodet[58]. Dette stoffet absorberer grønt lys. Med pulsslag fra hjertet vil det oppstå volumetriske endringer i blodstrømmen[59]. Dette gjør at mengden reflektert lys vil variere med pulsslage[59]. Fotodetektoren på sensoren bruker analogt til digital konvertering (*ADC*) på mengde absorbert lys og lagrer denne dataen i interne registre[54]. Basert på denne dataen må det utvinnes logikk (gjern i en mikrokontroller) for å kalkulere pulsen. De to LED-ene slås av og på under målinger enten med en frekvens på 128 Hz eller 64 Hz[54]. Fotodetektoren måler reflektert lys både når LED-ene er slått av og når de er slått på. Denne dataen legges i registrene *DATAOUT_LED OFF* (0x54/0x55) og *DATAOUT_LED ON* (0x56/0x57)[54]. Lysmengden som registreres når LED-ene er avslått kan potensielt brukes til å fjerne avvik eller støy på sensoren.



Figur 16: Pulssensorens virkemåte (BH1970GLC).[59]

2.4.5 Initialisering¹

Dersom det antas at oppkoblingen mellom mikrokontroller og pulssensoren (BH1970GLC) er riktig, vil man kunne skrive eller lese data på enhetens registre via TWI (eller I²C). For å kontrollere at kommunikasjonen fungerer, kan man lese ut registrene *MANUFACTURER ID* og *PART ID* og sjekke at disse stemmer overens med det som står oppført i databladet (se *Tabell 5*).

Navn	Adresse	Skrive-/lesetilgang (R/W)	Standard
MANUFACTURER ID	0x0F/0x92	R	0xE0
PART ID	0x10	R	0x0D
MEAS_CONTROL1	0x41	RW	0x00
MEAS_CONTROL2	0x42	RW	0x00
MEAS_START	0x43	RW	0x00
DATAOUT_LEDOFF	0x54/0x55	R	0xFF
DATAOUT_LEFON	0x56/0x57	R	0xFF

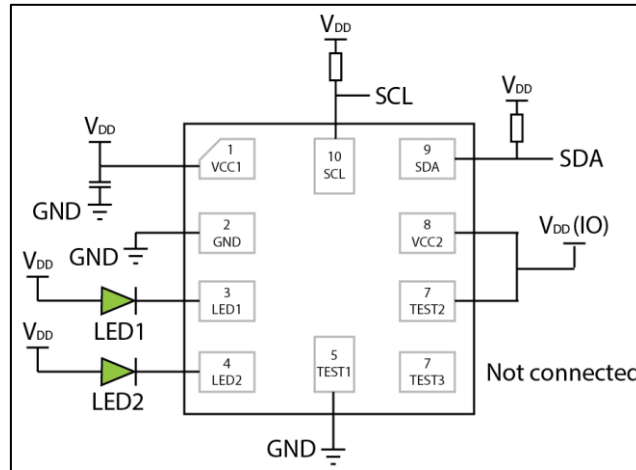
Tabell 5: Oversikt over registre i BH1970GLC.

En måte å initialisere enheten, slik at den utfører målinger, er å skrive 0x82 til *MEAS_CONTROL1* og 0x0D til *MEAS_CONTROL2*. Da settes oscillatoren (OSC-en) til å stå på. LED-ene vil blinke med en frekvens på 128 Hz og LED-strøm blir satt til 20 mA. Samtidig blir lesefrekvensen mellom mikrokontroller og pulssensor 32 Hz[54]. Når *MEAS_CONTROL2* er satt til å være 0x0D, lyser LED-ene når man leser data fra registret *DATAOUT_LEDON* (0x57) eller når første bit i *MEAS_START* (0x43) settes til verdien 1. Dersom dette er gjort kan det gjennomføres kontinuerlig målinger på *DATAOUT_LEDOFF*- og *DATAOUT_LEDON*-registrene. I TWI-oppsettet kan man lese ut 4 byte samtidig og hente ut all dataen fra disse registrene i én *lesning*.

¹ Navn på registre er ikke oversatt til norsk, slik at de skal være enklere å finne igjen i datablader.

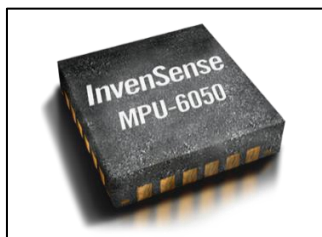
2.4.6 Typisk krets

En typisk krets for BH1790GLC står beskrevet i databladet. Se *Figur 17* for en forenklet versjon av den typiske kretsen.



Figur 17: Typisk krets for BH1790GLC.[54]

2.5 Akselerometer: TDK InvenSense MPU-6050



Figur 18: TDK InvenSense MPU-6050.[60]

MPU-6050 er navnet på en bevegelsessensor² utviklet av TDK InvenSense[60] (se Figur 18). MPU-6050 har akselerometer, gyroskop og temperatursensor innebygget[60]. Tekniske spesifikasjoner til MPU-6050 er hentet fra det offisielle databladet til enheten[61]. Noen sentrale tekniske spesifikasjoner er listet i tabellene nedenfor (se Tabell 6, Tabell 7 og Tabell 8).

Spesifikasjon	Verdi
Påkrevd spenningsforsyning	2,375 – 3.46 V (1,71 V for VLOGIC)
Grensesnitt (dataoverføring)	I ² C/TWI
Fysisk størrelse	4 mm × 4 mm × 0,9 mm

Tabell 6: Sentrale tekniske spesifikasjoner for MPU-6050.[61]

Sensor	Sensitivitetsmodus	Typisk oppløsning
Akselerometer	± 2 g	16,384 LSB/g
	± 4 g	8,192 LSB/g
	± 8 g	4,096 LSB/g
	± 16 g	2,048 LSB/g
Gyroskop	± 250 °/s	131 LSB/(°/s)
	± 500 °/s	65,5 LSB/(°/s)
	± 1000 °/s	32,8 LSB/(°/s)
	± 2000 °/s	16,4 LSB/(°/s)
Temperatursensor	N/A	340 LSB/°C

Tabell 7: Oppløsning for sensorer i MPU-6050.[9]

² I denne rapporten refereres MPU-6050 gjerne til som *akselerometeret*. Denne betegnelsen er noe upresist, men i dette prosjektet ble kun akselerometeregenskapene til MPU-6050 tatt i bruk.

Sensor	Arbeids tilstand	Typisk strømforbruk
Akselerometer	Typisk	500 μ A
	1,25 Hz	10 μ A
	5 Hz	20 μ A
	20 Hz	60 μ A
	40 Hz	110 μ A
Gyroskop	Arbeidsmodus	3,6 mA
	Ventemodus	5 μ A

Tabell 8: Typisk strømforbruk i MPU-6050. [61]

2.5.1 Initialisering

I²C/TWI-adressen til MPU-6050 er 0x68 dersom AD0-linjen er trukket til jord (GND)[62]. Dersom AD0-linjen er trukket til strømforsyningen (V_{DD}) brukes derimot adressen 0x69[62]. MPU-6050 initialiseres via TWI. Det er flere metoder for dette som står beskrevet i databladet[61]. En metode er å skrive 0x00 til PWR_MGMT_1 slik at målinger startes.

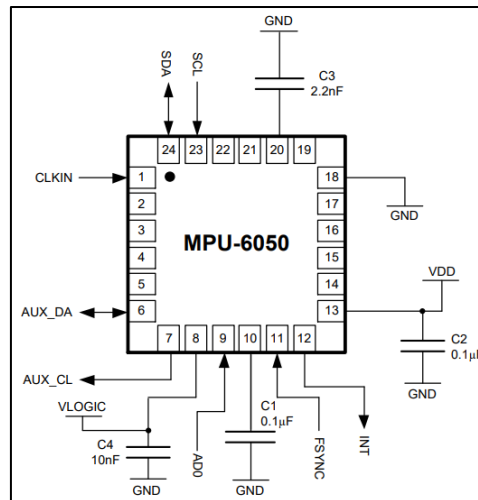
Register	Adresse	Funksjonsnavn	Funksjon	Aktiver	
PWR_MGMT_1	0x5B	CYCLE	Sov i sykluser	0x20	0x28
		TEMP_DIS	Deaktiverer temperatursensor	0x08	(aktiverer begge)
PWR_MGMT_2	0x6C	LP_WAKE_CTRL	Sovesyklus på 40 Hz	0xC0	0xC7
		DISABLE_XG	Deaktiverer X-gyro	0x04	(aktiverer alle fire)
		DISABLE_YG	Deaktiverer Y-gyro	0x02	
		DISABLE_ZG	Deaktiverer Z-gyro	0x01	

Tabell 9: Noen registre i MPU-6050.[62]

Ved å skrive 0x28 til PWR_MGMT_1, veksler MPU-6050 inn og ut av sovemodus, en oscillator klokke på 8 MHz benyttes og temperatursensoren deaktiveres. Ved å skrive 0xC7 til PWR_MGMT_2, deaktiveres gyroskopet og oppvåkningsfrekvensen settes til 40 Hz. Med et slikt oppsett kan kun akselerometer-verdier leses. Man kan lese akselerometerdata for X-, Y- og Z-aksene med en *lesning*. Dette gjøres ved å utføre en lesning av register *ACCEL_XOUT_H* (0x3B) og setter ønsket lengde på RX-data til 6 byte i TWI-oppsettet.

2.5.2 Typisk krets

Se *Figur 19* for en typisk kretsimplementasjon for MPU-6050.



Figur 19: Typisk krets for MPU-6050.[61]

2.6 Bluetooth

2.6.1 Bluetooth (IEEE 802.15.1)



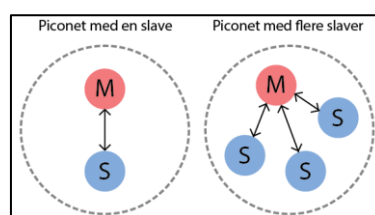
Figur 20: Bluetooths logo.

Bluetooth er en trådløs *personlig datanett* (PAN)-kommunikasjonsstandard[63]. Bluetooth og *IEEE 802.15.1*-standarden blir forvaltet av *Bluetooth Special Interest Group* (SIG). Bluetooth opererer innenfor frekvensene 2402 og 2480 MHz eller 2400 og 2483.5 Mhz og bruker en modulasjons-teknikk som kalles for *Gaussian Frequency Shift Keying* (GFSK)[63]. Vanlig FSK vil si at bæreølge-frekvensen altereres hyppig[64]. Denne frekvensaltereringen kalles vanligvis for *frekvenshopping* (*frequency hopping*). I GFSK blir datastrømmen bestående av nuller og enere sendt gjennom et *gaussian* (gaussisk) filter for å glatte overgangene mellom symbolene slik *out-of-band*-spektrumet kraftig reduseres[64]. 2.4 Ghz-båndet er et ikke-lisensiert frekvensbånd (*ISM-bånd*)[65] som benyttes av flere standarder enn Bluetooth, slik som f.eks. *WiFi*[66]. Bluetooth benytter seg derfor av noe som kalles adaptiv hopping (*adaptive frequency hopping*), som vil si at frekvenser som allerede okkuperes av andre nettverk blir ekskludert i *hopping-sekvensen*[67]. Bluetooth har mange bruksområder hvor trådløs dataoverføring er praktisk, slik som lydoverføring, *fitness*, helsevern, lokasjon-tjenester og mesh-nettverk i større skala[68].

2.6.2 Bluetooth Low Energy

Bluetooth Low Energy (*Bluetooth LE* eller *BLE*) (tidligere kalt *Bluetooth Smart*) sikter mot å bevare kommunikasjonsrekkevidden til klassisk Bluetooth men med et betydelig redusert strømforbruk[69]. Maksimalt *peak* (topp) strømforbruk spesifikk for BLE er 15 mA[69]. BLE er derfor velegnet for *IoT*-applikasjoner med små batteri-celler[70]. Blant annet brukes BLE innen helsevern, *fitness* og til hjemme-underholdning[69]. Til motsetning til klassisk Bluetooth, er BLE uegnet for overføring av tale eller lyd[71]. BLE har lavere gjennomstrømnings-hastighet (*throughput*) enn klassisk Bluetooth[71].

2.6.3 Piconet



Figur 21: Illustrasjon av to typer piconet.[72]

Bluetooth er en protokoll som bruker en master og slave-arkitektur[63]. Et nettverk bestående av flere Bluetooth-enheter kalles for et *Piconet*[63] (se *Figur 21*). Et piconet er et type *ad hoc*-nettverk[73]. *Ad hoc* vil si at nettverket er adaptivt uten predefinert infrastruktur[74], altså at enheter kan koble seg til og fra dynamisk. Et piconet har en master-enhet som kan kommunisere med syv slaver maksimalt[63]. En slave kan kun være slave i et piconet av gangen, men en slave-enhet kan fremdeles være master for et eget piconet (et slikt nettverksoppsett kalles gjerne for et *scatternet*[75]). Innen BLE definerer man ofte master som en sentral-enhet (*central*) eller *klient* (*client*) mens en slave typisk kalles for perifer-enhet (*peripheral*) eller *server*[69], [76]. BLE enheter oppdages gjennom en prosess som kalles *advertising* eller annonsering på norsk[69].

2.6.4 GATT



Figur 22: Et eksempel på en GATT-profil.

Videre i denne rapporten vil det bli brukt flere begreper innen Bluetooth. Noen av disse begrepene vil bli utdypet i dette avsnittet. Majoriteten av BLE-applikasjoner benytter seg av noe som kalles *Generic Attribute Profile (GATT)*[69]. *Figur 22* viser hvordan GATT-profiler typisk er strukturert. GATT brukes for overføring av korte datasegmenter kjent som *attributter* (*attributes*) over en lav-energi-link mellom klient og server[69]. Blant annet utveksles karakteristikker (*characteristics*) som er dataverdier[69]. En karakteristikk kan for eksempel holde på navnet til en enhet eller batteriprosent[69]. En tjeneste (*service*) er en samling av tilknyttede eller nærliggende karakteristikker[69]. Som et eksempel vil pulsdata og blodtrykk kunne ligge under samme tjeneste. En *descriptor* er tilleggsinformasjon som tilhører en bestemt karakteristikk[69]. En descriptor kan for eksempel holde på hva slags format/enhet en karakteristikk regnes i[69]. En karakteristikk kan ha så mange descriptorer den trenger[69].

2.6.5 Nordic Bluetooth LE GATT

I *Nordic Semiconductors Software Development Kit (SDK)* ligger et eksempelprosjekt som heter *ble_app_uart*. Dette eksempelprosjektet viser hvordan en nRF52-mikrokontroller kan kobles til mobiltelefon med datautskrift på PC via *Universal asynchronous receiver-transmitter (UART)*[77]. Prosjektgruppen tok utgangspunkt i dette eksempelprosjektet for å løse oppgaven. GATT-oppsettet i dette eksempelprosjektet har tre servicer. Den tredje servicen heter *Nordic UART Service*. Under denne servicen ligger to karakteristikk, hvor en er for RX (mottakelse) og den andre er for TX (sending). Data som sendes fra mikrokontrolleren legges i en descriptor med verdi 0x2902 på TX-karakteristikken. For å kunne lese data sendt fra nRF52-mikrokontrolleren må man dermed sette notifikasjon på denne descriptoren.

2.7 MQTT (Message Queuing Telemetry Transport)



Figur 23: MQTT.ORG sin logo.

Message Queuing (MQ) Telemetry Transport (MQTT) er en publiser-og-abonner-basert maskin-til-maskin (M2M)/IoT-protokoll som virker oppå TCP/IP-protokollen[78], [79]. MQTT versjon 3.1 støtter brukernavn og passord[80]. I tillegg kan *Secure Socket Layer (SSL)*[81] brukes for kryptering, selv om dette medfører mer databelastning på systemet[80]. MQTT er designet for å være ekstremt *lettvektig* og kan brukes i applikasjoner hvor båndbredden er begrenset eller hvor det kan oppstå høye tidsforsinkelser[78]. Man vil kunne distribuere korte meldinger til mange mottakere raskt og ressurseffektivt ved å bruke MQTT-protokollen[78]. Dette gjør at MQTT er praktisk å bruke i for eksempel IoT-nettverk og mobilapplikasjoner[80]. Potensielt kan et slikt system brukes for varsling fra en enhet til en annen. En MQTT-oppkobling benytter en *message broker* eller meldings-mekler[79]. Dette er en dataprogram-modul som håndterer meldingsvalidering, transformasjon og ruting av meldinger fra avsender til mottaker[82]. Meldinger som sendes via MQTT blir ikke lagret på server, men blir kun videresendt *her og nå*. Når en melding sendes til meldingsmekleren kalles dette gjerne for publisering (publishing). Meldingene publiseres på et emne (topic) som gjerne settes opp slik: *hovedemne/underemne*. Andre brukere tilknyttet tjenesten kan abonnere (subscribe) på slike emner. For å koble seg til en MQTT-broker, må man vite servernavn og portnummer og eventuelt ha brukernavn eller passord.

2.7.1 CloudMQTT

Et eksempel på en nettside som er vert for en MQTT-broker er *CloudMQTT*. CloudMQTT har flere abonnementer med ulike prislapper og begrensninger[83]. Gratis-abonnementet til CloudMQTT heter *Cute Cat*, som tillater opptil fem tilkoblede brukere og en dataoverføring på 10 Kbit/s[83]. På CloudMQTT sitt nettsted kan man legge til brukere som kan gis lese/skrive-tilgang. Ønsker man å gi alle brukere lese/skrive-tilgang kan man legge til en *ACL-pattern* (ACL-mønster) med verdi # og som har lese/skrive-tilgang. Man kan kontrollere CloudMQTT-klienten fra terminalvinduet dersom et programtillegg kalt *Eclipse Mosquitto*[84] er installert. Dokumentasjon/API-referanse for CloudMQTT ligger tilgjengelig på nett³.

³ Dokumentasjon for CloudMQTT finner du her: docs.cloudmqtt.com/cloudmqtt_api.html

2.8 App-utvikling

2.8.1 Operativsystemer og plattformer

Dersom det er ønskelig å utvikle en mobilapplikasjon som kan brukes av de fleste, kan det være hensiktsmessig å vurdere hvilke plattformer som brukes. I dag er de to største operativsystemene *Google Android OS* (forkortelse: *Android*) og *Apple iOS* (forkortelse: *iOS*). I mars 2019 er Android størst rundt om i verden og har en markedsandel på 54.3% i USA og 63.8% i Storbritannia, mens iOS har en markedsandel på 45.5% i USA og 35.9% i Storbritannia[85]. Operativsystemet fra *Microsoft* kalt *Windows Phone* har blitt offisielt kansellert[86] og hadde sin siste oppdatering 3. juni 2015[87]. I mars 2019 i USA har Windows Phone en markedsandel på 0.1% og i Storbritannia: 0.3%[85]. Basert på statistikk fra *Norsk Mediebarometer (Statistisk Sentralbyrå, SSB)*, hadde 95% av den norske befolkningen tilgang til smarttelefon i 2018[88]. Dersom vi antar tilnærmet lik markedsandel i Norge som i USA og Storbritannia, kan det antas at om en mobilapp gjøres tilgjengelig for Android og iOS, vil 99% av smarttelefon-brukere, eller ca. 93% av befolkningen ha mulighet til å bruke appen.

2.8.2 IDE (Integrert Utviklingsmiljø)

Det finnes flere forskjellige programmer og verktøy tilgjengelig for å utvikle apper, avhengig av hvilken plattform man ønsker å rette appen mot. Dersom man ønsker å rette en applikasjon for flere plattformer, slik som Windows Phone, Android og iOS kan man eksempelvis bruke *Microsoft Visual Studio* med tillegg for *Xamarin.Forms*, som er en UI-verktøykasse for flerplattform-apputvikling[12].

2.8.3 Android Studio



Figur 24: Android Studios logo.

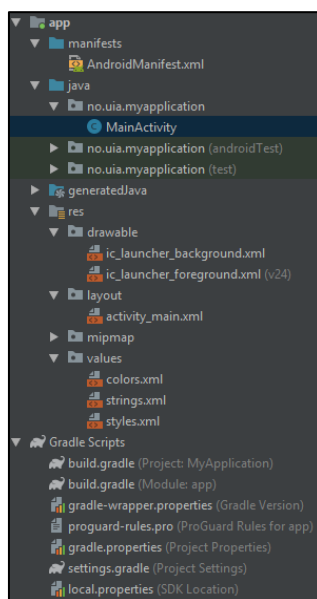
App-utvikling som primært rettes for *Google Android OS*, kan gjøres i den offisielle IDE-en til android-utvikling som heter *Android Studio*. Denne IDE-en er tilgjengelig for *Windows*, *MacOS* og *Linux*[13]. Android Studio bygget opp på *JetBrains' IntelliJ IDEA* og kodespråkene som støttes er *Java*, *C++*, *Kotlin* (Android Studio versjon 3.0 og oppover) i tillegg til programtillegg, slik som f.eks. støtte for programmeringsspråket *Go*[13]. Med Android Studio, kan man *enkelt* laste ned nødvendige SDK-versjoner og API-er. Offisiell dokumentasjon med veiledning og eksempellprosjekter ligger tilgjengelig på nett⁴. Android Studio bruker bygg-verktøyet *Gradle* og inkluderer enn innebygget *Android-emulator*.

⁴ Offisiell dokumentasjon for Android apputvikling ligger tilgjengelig her: developers.android.com/docs

2.8.4 Aktiviteter og tjenester

I Android Studio forholder man seg til *aktiviteter* (activities) og *servicer* (services). Enkelt forklart er aktiviteter en side i appen[89], mens servicer kan være bakgrunnsprosesser som kjører til og med når appen er lukket[90]. Servicer startes eller stoppes fra aktivitetene. Aktivitet og service kan sende signaler til hverandre via noe som kalles for *BroadcastReceiver*-e[91] eller *putExtra()*- og *getExtra()*-funksjonskall[92]. Selve programlogikken skrives i Java-filer eller alternativt Kotlin-filer, mens designet for de ulike sidene/aktivitetene, også kalt *Layout*, utformes i *Extensible Markup Language (XML)*-filer[89], [93]. Grafiske elementer, slik som tekst, knapper og bilder, kan legges til visuelt i IDE-en eller skrives direkte inn i layout-filene. Det er typisk at en aktivitet har en dedikert layout-fil som kobles til aktiviteten, slik at front-enden blir interaktiv og håndterer kommandoer for sluttbrukeren[89]. Det kan legges til push-notifikasjoner eller *Toast* (skåling) i appen for å gi sluttbrukeren varsler. Push-notifikasjoner dukker opp i oppgavelinjen mens en *Toast* er en liten tekstboks som vises i en begrenset tidsperiode nede på skjermen for gjeldende aktivitet[94].

2.8.5 Prosjekthierarki i Android Studio



Figur 25: Prosjekt-hierarki i Android Studio.

Se Figur 12 for et skjermbilde av et typisk prosjekt-hierarki i Android Studio. I *AndroidManifest.xml* listes app-tillatelser, aktiviteter og servicer i appen. *MainActivity.java* er en aktivitet som er tilkoblet layout-filen *activity_main.xml* i layout-mappen. I manifestet kan *MainActivity* defineres til å være start-aktivitet når appen åpnes. I *drawable*-mappen legges bilder, figurer og bilder. Konstanter for dimensjoner, farger, stiler og strenger legges i *values*-mappen. Disse verdiene kan bli referert til fra java-filene samt XML-filene. I *build.gradle (Mobile app)* defineres kompilasjonsinnstillinger og referanser til eksterne biblioteker.

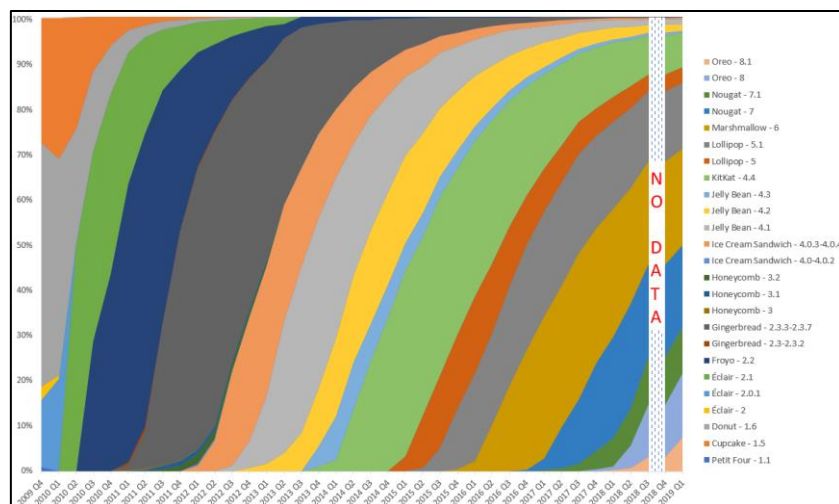
2.8.6 Nordic Semiconductor

Nordic Semiconductor legger ut åpen kildekode for sine mobilapper på *GitHub*⁵. For eksempel finner man kildekode for *nRF Toolbox*, *nRF Connect*-appen og *nRF Blinky*-appen. Disse mobilapplikasjonene retter seg mot bruk av Bluetooth Low Energy for kommunikasjon med mikrokontrollere utviklet av Nordic Semiconductor. Kildekoden for disse applikasjonene er noe avanserte, men innehar fullstendig funksjonalitet for oppkobling med *nRF52*-mikrokontrollere programmert med tilsvarende eksempelkode (tilgjengelig i Nordic Semiconductors SDK). På lenken finner man også kildekode for Nordic Semiconductors eget Bluetooth LE prosjekt; *Android-BLE-Library*[95] som kan brukes i Android-prosjekter hvor kommunikasjon over Bluetooth LE er ønskelig[95]. Appene nevnt tidligere ligger også tilgjengelig for nedlastning i *Google Play Store*.

2.8.7 Versjonshistorikk (Google Android OS)

Ønsker man å utvikle en mobilapplikasjon som flest mulig skal kunne benytte, må man også ta hensyn til at ikke alle har den nyeste versjon av operativsystemer. Se *Figur 26* for en grafisk fremstilling av distribusjonen av Android-versjoner over tid. Ser man på global distribusjon av Android-versjoner, ser vi at under 10% av Android-brukere i 1. kvartal 2019 har den nest-nyeste Android-versjonen kalt *Oreo* (lansert 21. august 2017)[96]. Et nyere Android OS vil være bakover-kompatibelt med apper utviklet for tidligere API-versjoner. Ifølge IDE-en *Android Studio*, må man bruke API 15 med Android-versjon 4.0.3 *IceCreamSandwich* for at en app skal kunne kjøres på tilnærmet 100% av Android-enhetene som brukes i dag[96]. Til sammenlikning er nyeste lanserte versjon av Android OS Android-versjon 9.0 *Pie* med API-nivå 28 (lansert 6. august 2018)[96]. Dersom man ønsker å utvikle en app som *alle* Android-brukere kan benytte, vil man ideelt sett bruke API 15, men det kan forekomme at noen biblioteker krever høyere API-nivå. Et eksempel på dette er *Bluetooth Gatt*-biblioteket som krever API 18 med Android-versjon 4.3 *Jelly Bean* eller høyere for flere funksjonsskall. Altså må man benytte seg av API 18, for å lage en Android-mobilapp med Bluetooth-funksjonalitet.

⁵ Nordic Semiconductors åpne kildekode ligger tilgjengelig her: github.com/NordicSemiconductor



Figur 26: Google Android OS versjonshistorikk.[96]

2.9 Kretskort-utvikling

2.9.1 Kretskort:

Kretskort eller *PCB (Printed circuit board)* er et brett/kort som består av laminat og et ledningsmateriale. På kretskortet blir det dannet ledningsmønstre som danner kretser med utlegg for at komponenter kan loddes på[97]. Et kretskort består typisk av glassfiber-laminat som danner selve kortet og kobberfolie på en eller begge sider som kan formes til ledningsmønstre gjennom etsning eller fresing[98]. Ofte består kretskort av flere lag med laminat og kobberfolie slik at kretsen kan utformes på flere nivåer. Kretskortet er skjelettet i et elektrisk produkt da dette fungerer som en sammenkobling mellom de separate komponentene, samt tilbyr en felles jording for disse[98].

2.9.2 Low-dropout regulator (LDO)

Low-dropout (LDO)-regulator er en form for lineær spenningsregulator[99]. Funksjonen til en spenningsregulator er å levere en konstant spenning[100]. Spenningsregulering brukes ofte når det er ønskelig med en stabil utgangsspenning selv om inngangsspenningen varierer[101]. LDO-regulatorer har særdeles gode egenskaper for å levere en utgangsspenning som er marginalt mindre enn inngangsspenningen[101]. Dette gjør de optimale for å regulere spenning fra et batteri, hvor spenningen vil falle etter hvert som batteriet blir utladet. En LDO-regulator har også typisk meget lav mengde støy sammenlignet med andre spenningsregulatorer. Denne egenskapen gjør den spesielt egnet for bruk i mindre kretser som er mer sensitive[99]. I kretser hvor batteri brukes, vil det kunne være ekstra viktig å ta hensyn til effekttap i en LDO. Effekttapet kan regnes med formelen nedenfor (se *Formel 1*).

$$P_{loss} = (V_{in} - V_{out}) \times I_{out} + V_{in} \times I_Q$$

Formel 1: Effekttap i en LDO-regulator.[99]

I formelen brukes inngangsspenning (V_{in}), utgangsspenning (V_{out}), utgangsstrøm (I_{out}) og hvilestrømmen (I_Q eller *quiescent current*). Hvilestrømmen pleier å stå beskrevet i databladet til komponenten.

2.9.3 Jordplan

Under utvikling av kretskort er det viktig at hele kretsen har en stabil, felles jording. For å oppnå en felles jord av høyest mulig kvalitet, brukes jordplan[102]. Et jordplan består av en flate som er elektrisk ledende. Optimalt vil jordplanet bestå av resterende ledningsflate etter at ledningsmønstre er utformet på et plan. Det er ønskelig at alle komponentene i en krets kobles mot felles jordplan[102]. Det er også vanlig å lage et helt plan for jord dersom kretskortet har flere lag hvor ikke alle lagene trenger ledningsmønstre. I en krets vil et bra jordplan forbedre kretsen ved å redusere støy. I tillegg vil kretsen bli mer stabil[103].

2.9.4 Avkoblingskondensator

En moderne elektrisk krets vil ofte være utsatt for støy som følge av lave referansespenninger og forskjellige potensielle støykilder[104]. Den største kilden for støy finnes ofte i strømmettet som blir lagt ut i et design[104]. For å redusere støy i en slik krets, kan avkoblingskondensator være et effektivt verktøy. En avkoblingskondensator er en kondensator som blir koblet direkte mellom strømkilden og jord (helst et jordplan). Gjennom en slik kobling kan støy reduseres betraktelig dersom kondensatoren er plassert i en god posisjon. For optimal avkobling burde kondensator plasseres nærmest mulig den komponenten hvor støy skal reduseres[104].

2.9.5 Digitalt kretskortdesign

For å produsere et kretskort, er det nødvendig å lage en skisse med utlegg for komponenter, ledningsmønstre og jord. Slike skisser blir typisk tegnet i digitale programmer som er laget for nettopp dette. *Altium Designer* er et eksempel på et slikt program. I Altium Designer samles designfilene for alle komponentene som skal benyttes. Hver komponent trenger en design-fil som inneholder to komponenter: skjematikk og fotavtrykk (*footprint*)[105]. Skjematikken utvikles først. Her sammenkobles kretsen i en skjemategning. Når skjemategningen til kretsen er komplett, overføres den skjematikken inn i kretskort-modus. Her kommer fotavtrykket til alle komponentene opp, slik at de kan kobles sammen med ledningsmønstre som tilsvarer skjemategningen[105], [106]. Et fotavtrykk beskriver den fysiske plasseringen til loddelandene (*soldering pads*) for en komponent[107]. Det er viktig at alle komponentene er tilkoblet et stabilt jordpunkt, og helst et jordplan. Optimalt vil dette jordplanet bestå av all overflate som ikke blir brukt til komponenter eller ledningsmønstre. Innad i Altium Designer kan et slikt jordplan designes ved hjelp av et verktøy som heter *Polygon Pour*[108].

2.9.6 Fresing

Når komponentene er festet sammen med ledningsmønstre og jordplan, kan kretskortet produseres. For å produsere kretskortet, eksporteres *Gerber*-filene⁶ til prosjektet. Gerber-filene kan lastes opp til en maskin for kretskortproduksjon[105]. Eksempelvis kan kretskort freses ved hjelp av en *ProtoMat S103*[110] (se *Figur 27*). ProtoMat S103 er en maskin som er utviklet av *LPKF Laser & Electronics*[111]. Den er designet for å produsere kretskort i en *relativt liten* skala. ProtoMat S103 bruker designfiler til å frese det ønskelige designet på et kort som er dekket med ledningsmateriale. Hele prosessen går på egenhånd ved hjelp av ulike frese-verktøy som maskinen selv er i stand til å velge og bytte mellom.

⁶ Gerber er et filformat som vanligvis brukes innen kretskortutvikling[109]. Filformatet beskriver et 2D bilde av kretskortdesign[109].



Figur 27: LPKF ProtoMat S103.[111]

3 Løsning

Dette kapitlet omhandler valgt løsning og resultatet til hovedproblemstillingen. Først blir kravene til produktet spesifisert i kapittel 3.1. Deretter blir nødvendige designspesifikasjoner for å innfri kravene gjennomgått i kapittel 3.2. Selve løsningen og resultatet står beskrevet i kapittel 3.3.

3.1 Krav

Opgaven fra oppdragsgiveren Bouvet sin side, er å utforske muligheter rundt pasientarmbåndet som inngår i konseptet *Mitt Sykehusopphold*. Bouvet har krav til hvilken funksjonalitet som skal inngå i *Mitt Sykehusopphold* for å kunne bedre pasientopplevelsen på sykehus. Funksjonalitet som inngår i konseptet, ble utdypet i kapittel 1.1 *Bakgrunn*. Bouvet har stort sett gitt prosjektgruppen frie tøyler til å sette sine egne krav til produktet som utvikles. Hovedkravet til Bouvet er at produktet skal ha en form for sensormåling (f.eks. puls) med overføring av data til en dedikert mobilapp.

Krav fra Bouvet:

- Utforske muligheter for utvikling av et pasientarmbånd som kan overføre data direkte til en dedikert mobilapplikasjon.

Fra universitetet sin side, ga veileder og universitetslektor Lei Jiao krav om at en fysisk elektronisk enhet skal utvikles i løpet av prosjektperioden, slik at prosjektet skulle ha god elektronikk-faglig relevans. Lei Jiao forespurte også gruppen å ta en titt på tids-synkronisering mellom IoT-enheter som inngår i et felles Bluetooth-nettverk. Dette er praktisk dersom man for eksempel har et bånd rundt armen og ett rundt magen og ønsker å synkronisere dataen mellom disse båndene.

Krav fra UiA:

- Produktet som utvikles må ha god elektronikk-faglig relevans.
- Gruppen skal undersøke synkronisering mellom to eller flere IoT-enheter som inngår i samme Bluetooth-nettverk (piconet).

Kravene til produktet ble i stor grad satt av prosjektgruppen i forprosjektperioden. Gruppen satte som mål å utvikle en kretskort-prototype som er i stand til å kommunisere med en egenutviklet mobilapp over *Bluetooth Low Energy*. Prototypen skal utstyres med ulike sensorer, deriblant puls-sensor og akselerometer/fallsensor. Data fra puls-sensoren skal vises direkte i mobilappen. Kretskortet skal freses og komponenter/moduler skal loddess på, slik at gruppen har en fysisk produktprototype som kan fungere på egenhånd. Flere krav er listet nedenfor.

Krav satt av prosjektgruppen til prototypen:⁷

- Et egenutviklet prototype-kretskort mindre enn 4 x 4 cm i fysisk størrelse.
- Skal ha pulssensor og fallsensor (akselerometer).
- Skal ha varsel-knapper, reset-knapp og LED-indikatorer.
- Skal kunne sende varsler til mobilappen.
- Kalkulasjon av pulsverdier med $\pm 10\%$ nøyaktighet.
- Fallsensor som kan detektere *store* bevegelser.
- Trådløs funksjonalitet: Prototypen skal gå på batteri.
- Batteriet skal være oppladbart.
- Strømbesparing: Mikrokontrolleren skal håndtere avbrudd.
- Strømbesparing: Pulssensoren skal deaktiveres dersom armbåndet tas av.

Krav satt av prosjektgruppen til appen:⁸

- En egenutviklet mobilapplikasjon (app) for Android-plattformen.
- Dataoverføring mellom prototype og app via *Bluetooth Low Energy*.
- Pulsdata kan leses av direkte i appen (med graf over tid).
- Indikasjon på om måleverdier ser normale ut.
- Helsepersonell skal se en liste over pasienter.
- Helsepersonell skal kunne lese data fra flere armbånd samtidig.
- Helsepersonell skal kunne legge inn pasientnotater, som skal kunne leses av pasienten.

⁷ Både funksjonelle og ikke-funksjonelle krav er listet.

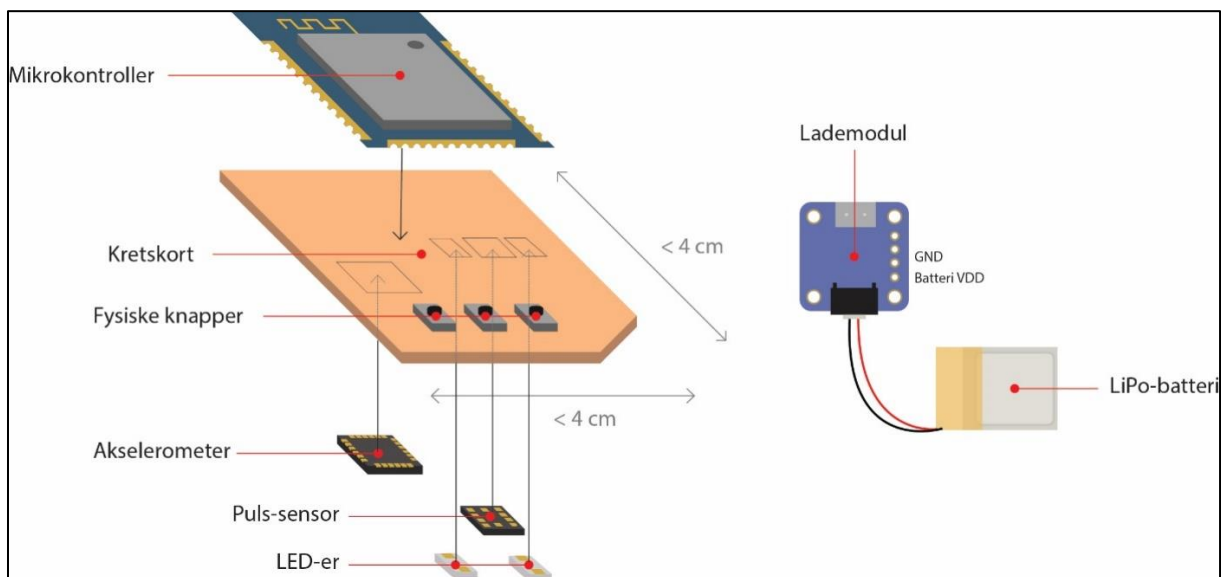
⁸ Både funksjonelle og ikke-funksjonelle krav er listet.

3.2 Designspesifikasjoner

Dette kapitlet omhandler design-spesifikasjonene som ble satt for armbånd-prototypen og appen før utvikling og ferdigstilling. Altså beskriver dette delkapitlet hvordan vi hadde tenkt til å løse oppgaven. Først blir design-spesifikasjonene for prototype-kretskortet beskrevet i kapittel 3.2.1, deretter blir design-løsningen for appen utdypet i kapittel 3.2.2.

3.2.1 Armbånd

For å utvikle armbånd-prototypen var det planlagt å utvikle et kretskort og lodde på nødvendige komponenter. Prosjektgruppen anså å utvikle en egen ladekrets som vanskelig. Det ble derfor bestemt at en ferdigutviklet modul for dette skulle benyttes for prototypen. Lademodulen som ble valgt heter *Adafruit 1904*. Denne lademodulen har micro-USB-inngang. Resten av komponentene består hovedsakelig av mikrokontroller, batteri, fysiske (*taktile*) knapper, akselerometer og pulssensor med to supplerende grønne LED-er. For å spare plass, og fordi det ville være en logisk plassering, var det tenkt å plassere puls-sensoren på undersiden av kortet, mens knapper plasseres på oversiden. Se *Figur 28* for en illustrasjon av hvordan det var tenkt å montere på komponenter. Illustrasjonen er en forenkling, og utelukker eventuelle avkoblingskondensatorer, spenningsregulatorer og pin-headere som kommer i tillegg. Design-spesifikasjoner til armbånd-prototypen er listet på neste side.



Figur 28: Forenklet illustrasjon av den planlagte prototypen.

Tekniske spesifikasjoner knyttet til armbåndet:

- Dimensjoner (kretskort): mindre enn 4 x 4 cm.
- Mikrokontroller-modul/hovedmodul (Nordic Semiconductor nRF52832 SoC).
 - Enten aconno GmbH ACN52832 eller Insight SiP ISP1507.
- Pulssensor (Rohm Semiconductor BH1790GLC).
 - To grønne LED-er dedikert til pulslesning.
- Puls kalkuleringer med $\pm 10\%$ målenøyaktighet.
- Fallsensor/akselerometer (TDK InvenSense MPU-6050).
- 3.7 V, 70 mAh oppladbart litium-polynomer-batteri (TinyCircuits ASR00011).
- LiPo-lademodul med micro-USB-inngang (Adafruit 1904, 16 x 15 mm).
- Batteritid: Minst 5 timer.
- 2 taktile knapper for å aktivere alarmer.
- Reset-knapp skjult på undersiden/siden.
- Minst én LED-indikator (RGB).
 - aconno GmbH ACN52832 har en RGB-LED tilgjengelig.
- Ingen isolering og dermed heller ikke vannavstøtende.

3.2.2 App

Appen skal i størst grad utvikles for å fungere som et supplement til armbånd-prototypen som utvikles, hovedsakelig for demonstrasjonsformål. Hovedmålet var å lage en egenutviklet app som klarer å kommunisere med armbånd-prototypen via Bluetooth Low Energy og vise data direkte på skjermen. Det valgte designet tar utgangspunkt i at helsepersonell skal ha mulighet til å se en liste over tilgjengelige pasienter i sin avdeling, mens pasientene kun skal kunne lese data fra sitt personlige armbånd. Det måtte derfor være mulig å separere helsepersonell fra pasienter. Løsningen for dette ville være å legge til en innloggingsskjerm hvor man kan velge tilknytning og eventuelt logge på med brukernavn eller passord. Etter innlogging vil man få opp UI-elementer avhengig av om man er helsepersonell eller eventuelt pasient. Det ble satt noen design-spesifikasjoner for det påkrevde designet. Disse er listet nedenfor.

Tekniske spesifikasjoner knyttet til appen:

- Appen utvikles for Android-plattformen.
- Pasienter og helsepersonell skal bruke samme app.
- Skal fungere for Android-versjon 4.3 *JellyBean* (med API-nivå 18) eller høyere.
- Enkelt grafisk brukergrensesnitt (UI) med få farger og god kontrast.
- Appen bruker Bluetooth Low Energy for kommunikasjon med pasientarmbånd.
- Pulsdata skal ha et dedikert felt på skjermen.
- Graf som viser puls over tid.
- App-språk: Norsk.

Appen skal ha tre hovedsider (også kalt *aktiviteter* innen app-utvikling): 1: Innloggingside, 2: Pasientside og 3: Helsepersonell-side. Disse ulike aktivitetene er videre beskrevet.

3.2.2.1 Innloggingside



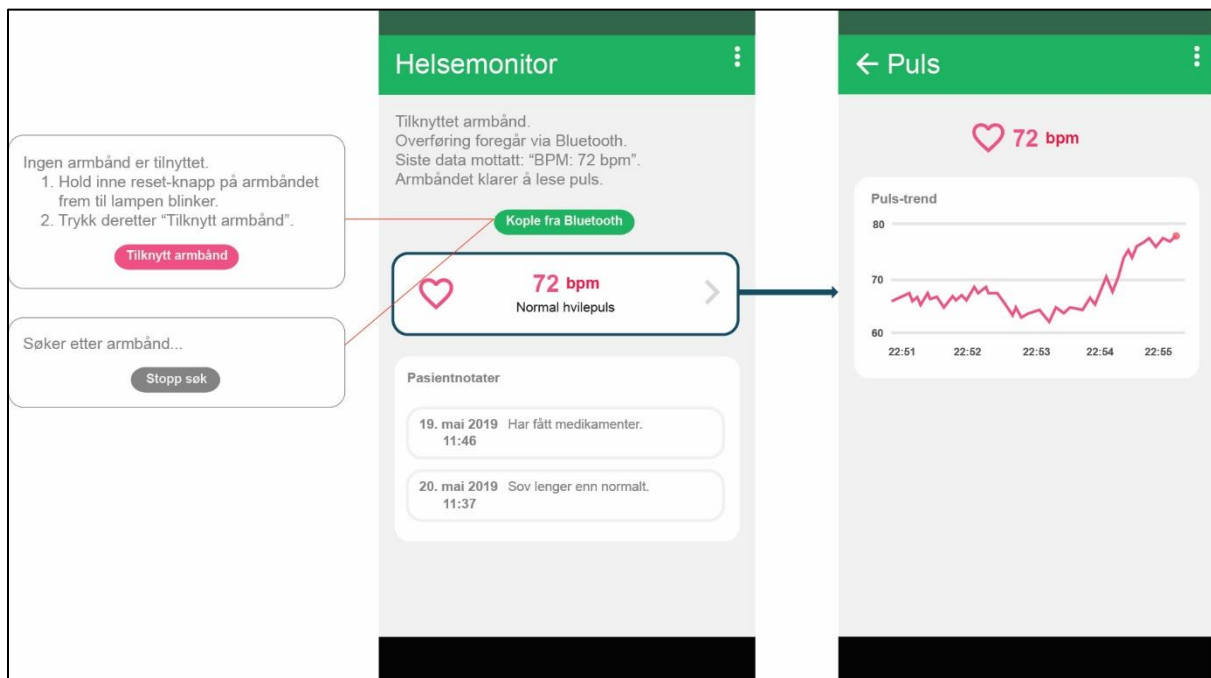
Figur 29: Skisse av innlogging-side i «Mitt Sykehusopphold»-appen.

Se Figur 29 for en skisse av innloggingssiden til appen. På innloggingside velger man om man er pasient eller helsepersonell. Potensielt kan det her tilføyes brukernavn og passord. Ettersom fokuset i dette prosjektet ikke var oppkobling mot server, ble det sett bort ifra et reelt innlogging-system som skiller helsepersonell fra pasienter med brukernavn og passord. Prosjektgruppen så bort ifra behovet av et sikkerhetssystem. Alle kan aksessere helsepersonellsiden ved å trykke på en knapp. Denne konstruksjonen er ikke ideell i en reell situasjon, men er praktisk i en demo-setting slik som i dette prosjektet.

Design-spesifikasjoner rettet mot innloggingsiden:

- Separering av pasienter og helsepersonell.
- «Mitt Sykehusopphold»-logoen vises øverst.

3.2.2.2 Pasientside/Helsemonitor



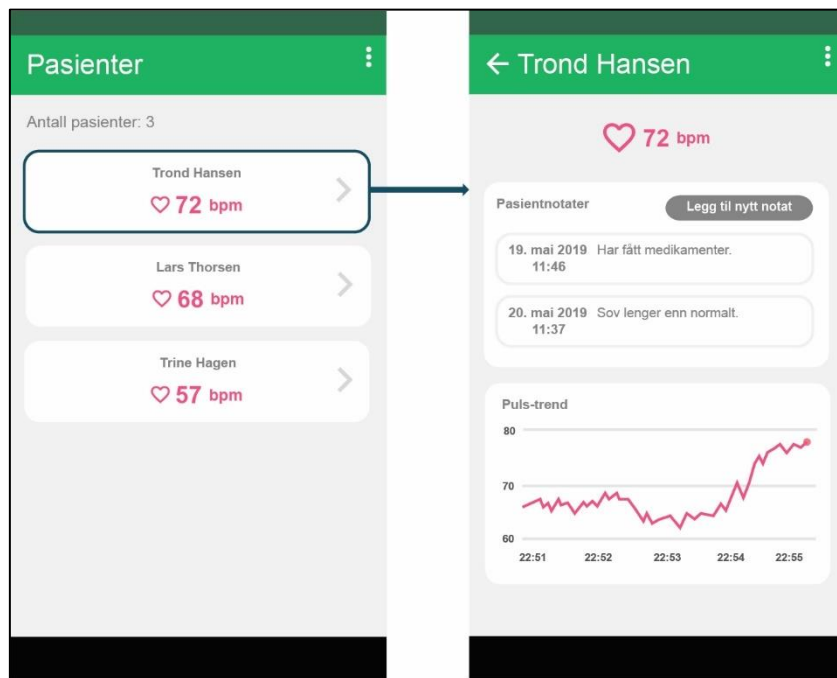
Figur 30: Skisse av pasientenes «hovedside» i appen, også navngitt «Helsemonitor».

Se Figur 30 for en skisse av pasientenes hovedside i appen. På pasientsiden kan man som pasient, administrere kommunikasjon med sitt eget personlige armbånd. Man kan *enkelt* koble seg til armbåndet ved å trykke en enkelt knapp for dette eller eventuelt koble seg fra igjen. Det meste av tilkoblingsprosessen bør foregå i bakgrunnen, slik at det blir enklest mulig for pasienten å forstå hva han eller hun skal gjøre. Når man er tilkoblet vil man kunne motta pulsdata som vises direkte på skjermen. På denne siden bør det også vises informasjon som guider pasienten rundt bruk av armbåndet, slik som for eksempel hvordan man tilknytter mobil og armbånd sammen. Dersom man trykker på den hvite boksen med puls, kommer man til en side som heter *Puls*. På denne siden kan man eventuelt kunne se pulsen over tid gjennom en graf som oppdaterer seg samtidig med pulsen.

Design-spesifikasjoner rettet mot pasientsiden:

- Enkel og intuitiv Bluetooth-administrasjon.
 - Kobler seg automatisk til tilgjengelig armbånd ved søk.
 - Forklarende og hjelpende tekstelementer.
- Puls vises i et eget felt/en egen boks.
 - Man kan klikke på puls-feltet for å se graf over pulsen.

3.2.2.3 Helsepersonellside



Figur 31: Skisse av hovedsiden til helsepersonell i «Mitt Sykehusopphold»-appen.

Se Figur 31 for en skisse av hovedsiden til helsepersonellet. På helsepersonalets side, skal man kunne se en liste over tilgjengelige pasienter med oppdaterte pulsverdier. Dersom man trykker seg inn på en eventuell pasient får man opp flere detaljer. Blant annet kan helsepersonell se og eventuelt legge til nye pasientnotater. I tillegg vises en graf som oppdaterer seg med pasientens puls. Når man er innlogget som helsepersonell, er det ønskelig at viktige notifikasjoner dukker opp på skjermen, enten som push-notifikasjoner eller som en *Toast* (skåling). Dette kan blant annet være varsler om at en pasient har falt, eller at en pasient tilkaller hjelp. I pasientlisten bør denne pasienten utheves og plasseres i toppen av listen dersom en slik notifikasjon forekommer.

Design-spesifikasjoner rettet mot personell-siden

- Helsepersonell skal se en liste over tilgjengelige pasienter med oppdaterte pulsverdier.
- Man skal kunne klikke på en pasient i listen og se flere detaljer.
 - Her kan man legge inn pasientnotater og se graf over pulsen til pasienten.

3.3 Implementering

3.3.1 Kretskortdesign

For dette prosjektet ble det satt noen begrensninger for designet basert på utstyret som gruppen hadde tilgjengelig. Designspesifikasjonene måtte ta hensyn til begrensningene til *ProtoMat S103* som ble brukt til fresing av kretskort. Noen av disse begrensningene er listet nedenfor.

Begrensninger ved bruk av ProtoMat S103:

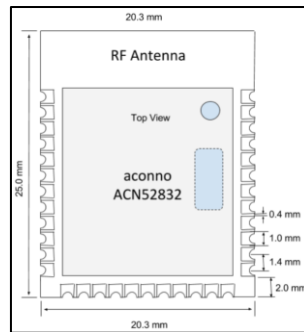
- Maksimal bredde på ledningsmønstre: 0,3 mm
- Minimum avstand mellom objekter: 0,2 mm
- Minimum diameter for hull: 0,4 mm
- Ingen loddelander skal bli plassert mindre enn 1 mm fra kanten på kretskortet.

3.3.1.1 Plassering av via-huller

Dersom via-hull skulle bli tatt i bruk, måtte gruppen være bevisst på at det ikke blir kontakt mellom de to sidene av kretskortet (planene) uten at det manuelt ble loddet ledningsmateriale mellom dem. Dersom et via-hull plasseres under en komponent, vil loddetinnet kunne heve komponenten litt. Da kan komponenten få dårlig kontakt eller bli vanskelig å lodde på riktig. På grunn av dette ville det ikke være lurt å plassere via-hull direkte under komponenter.

3.3.1.2 Utvikling av designfiler for ACN52832

Kretskortene i dette prosjektet har blitt utviklet ved hjelp av Altium Designer. Ikke alle komponentene hadde tilgjengelige designfiler for digitalt kretskortdesign. Mikrokontrollermodulen (ACN52832) manglet designfiler som var compatible med Altium Designer. For å kunne bruke denne modulen, måtte gruppen dermed lage disse designfilene selv. For å lage disse, ble dimensjoner fra databladet til komponenten brukt (se *Figur 32*). I figuren beskrives størrelsen på loddelandene, hvor langt fra hverandre de befinner seg og hvor stor selve komponenten er.



Figur 32: Størrelsesforhold for ACN52832 hentet fra datablad.[51]

Når designfilene til en komponent skal lages, brukes *New PCB Library*-funksjonen til Altium Designer[112]. Her plasseres loddelandene etter målene fra datablad. Resultatet vil være fotavtrykket (*footprint*'en) til modulen (se Figur 33).



Figur 33: Egenutviklet fotavtrykk for ACN52832.

Videre er det nødvendig å lage skjematikk-komponenten for designet. Her brukes *New Schematic Library*-funksjonen i Altium Designer. Med dette verktøyet lages en firkantet modell av komponenten hvor det fylles inn med pins som matcher pins listet i databladet (se Figur 34). For å fullføre designet, kobles fotavtrykket og skjematikken sammen ved å trykke på *Add Footprint* på oppgavelinjen som er plassert nederst i brukergrensesnittet. Deretter velger man fotavtrykk-filen man laget.

1	*C			19
2	GND	P0.10/NFC2		20
3	P0.25	P0.11		21
4	P0.26	P0.12		22
5	P0.27	P0.14		23
6	P0.28	VDD		24
7	P0.29	VDD		25
8	P0.30	P0.15		26
9	P0.31	P0.16		27
10	P0.02/AIN0	P0.17		28
11	P0.03/AIN1	P0.18		29
12	P0.04/AIN2	P0.19		30
13	VDD	P0.20		31
14	GND	P0.21		32
15	P0.05/AIN3	VDD		33
16	P0.06	SWDIO		34
17	P0.07	SWDCLK		35
18	P0.08	GND		
	P0.09/NFC1			
	IC?			

Figur 34: Skjematikk-komponent for ACN52832.

3.3.1.3 Antenne

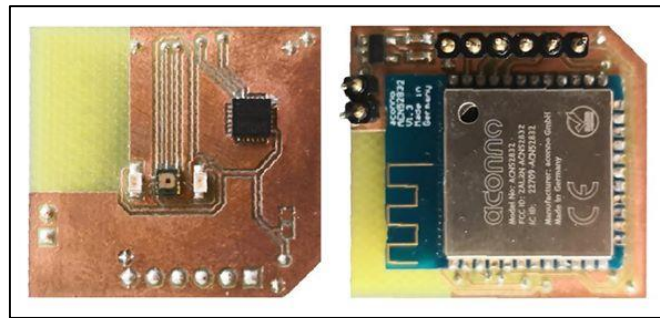
ACN52832 har en Bluetooth-antenne øverst på komponenten. En slik antenne kan bli påvirket av støy dersom det befinner seg jordplan eller ledningsmønster direkte under denne. For å unngå dette har det blitt fjernet kobberfolie fra området rundt komponenten i designene. Dette er synlig på prototypene (se *Figur 35*).

3.3.1.4 Prototyper

Det ble utviklet totalt to prototype-kretskort i dette prosjektet (*Prototype 1* og *Prototype 2*). Prototypene som har blitt designet, har jordplan på alle ledige flater hvor dette er tilgjengelig. Det er enkelte flater som er innestengt mellom ledningsmønstre og som ikke har en stabil jording. Disse flatene er ikke tilkoblet og bidrar dermed ikke til å utvikle støy. Kretskortdesignene som ble produsert i dette prosjektet ble loddet for hånd under mikroskop⁹. *Prototype 1* og *Prototype 2* vil bli vist i de neste to delkapitlene.

⁹ Hovedsakelig ble komponenter loddet for hånd på Durapart av Tore Myrene Rislåa med mikroskop.

3.3.2 Prototype 1



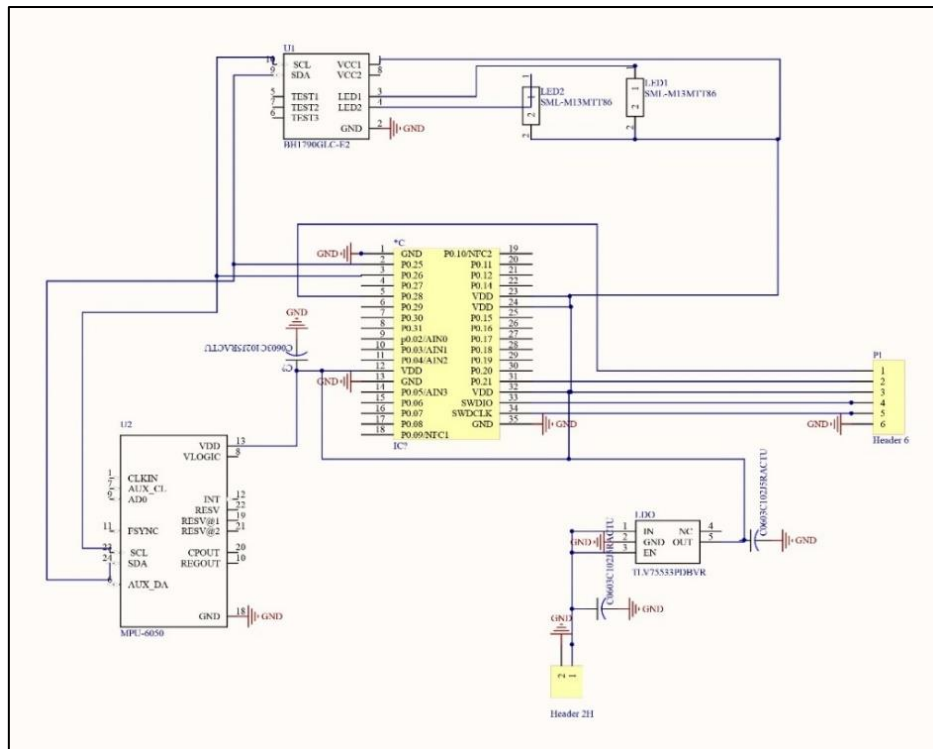
Figur 35: Prototype 1.¹⁰

Se Figur 35 for et bilde av *Prototype 1*. Komponentene som inngår i designet er listet i *Tabell 1*. Figur 36 viser kretsskjematikken for denne prototypen. prototypen hadde fundamentet på plass, men manglet taktile knapper. I tillegg førte noen feilkoblinger til at mikrokontrolleren ikke fikk kontakt med noen av sensorene.

Modul	Komponent	Produsent	Montering	Antall
Mikrokontroller	ACN52832	aconno GmbH	SMD/SMT	1
Akselerometer	MPU-6050	TDK Invensense	SMD/SMT	1
Pulssensor	BH1790GLC-E2	Rohm Semiconductor	SMD/SMT	1
LED (grønn)	SML-M13MTT86	Rohm Semiconductor	SMD/SMT	2
Spenningsregulator (LDO)	TLV75533PDBVR	Texas Instruments	SMD/SMT	1
Kondensatorer	1 μ F	AVX	SMD/SMT	2

Tabell 10: Komponenter som ble brukt i Prototype 1.

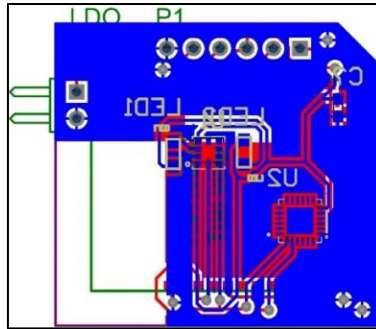
¹⁰ Se Vedlegg J Kretskortdesign for større bilder av prototypen, kretsskjematikken og kretskortdesignet.



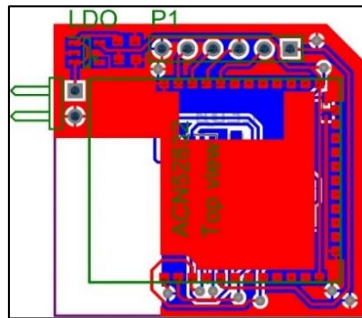
Figur 36: Kretsskjematikk for Prototype 1.

3.3.2.1 Oppkobling

I begge prototypene (*Prototype 1* og *Prototype 2*) leveres 3,7 V fra et litiumpolymerbatteri. Spenningen reduseres ned til 3,3 V ved å bruke en spenningsregulator (*LDO*). Utgangen fra *LDO*-en på 3,3 V brukes som en felles strømforsyning (V_{DD}) for alle komponentene i kretsen. Akselerometer (*MPU-6050*) og pulssensoren (*BHI790GLC-E2*) er koblet til mikrokontroller (*ACN52832*) med felles *SCL*- og *SDA*-linjer samt V_{DD} og jord. I *TWI* er alle slave koblet til samme *SCL*- og *SDA*-linje og det er disse linjene som brukes til utveksling av data. Pulssensoren bruker to grønne LED-er som er koblet mellom pulssensorens LED-driver og V_{DD} . For å programmere mikrokontrolleren etter den har blitt loddet på kretskortet, er det lagt til egne programmeringspinner i designet. Disse befinner seg på oversiden av kretskortet (se *P1* øverst i *Figur 35* og *Figur 38*). Hullene til disse programmeringspinnene er også synlige fra undersiden av kortet.



Figur 37: Undersiden av kretskortdesignet (Prototype 1).



Figur 38: Oversiden av kretskortdesignet (Prototype 1).

3.3.2.2 Feil eller mangler

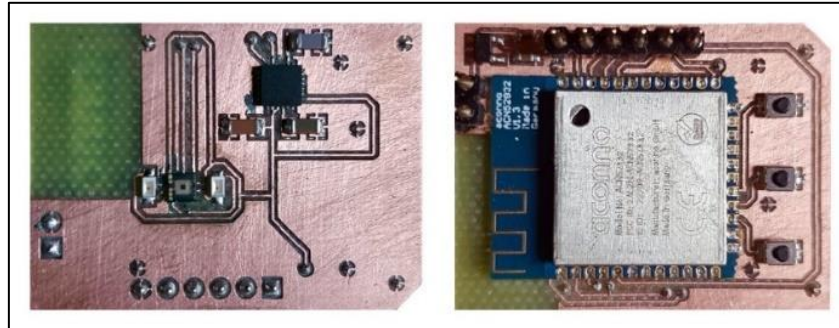
Det var ikke all funksjonalitet som kom på plass i den første prototypen. Mikrokontrolleren fikk ikke hentet ut data fra noen av sensorene (akselerometer og pulssensor). De grønne LED-ene tilknyttet pulssensoren lyste heller ikke. Spenningsregulatoren og mikrokontrolleren (i tillegg til Bluetooth) ble testet. Disse to fungerte. Dermed ble det antatt at feilen med sensorene skyldtes lodding eller kretskoblinger. Databladene for sensorene ble nøyer undersøkt. Det viste seg at noen pins som skulle vært tilkoblet V_{DD} eller jord, ikke var blitt tilkoblet. Akselerometeret manglet også noe kretslogikk. Dette skyldtes at sensorene var koblet likt som for utviklingskortene. Gruppen gikk derfor inn for å utvikle en ny prototype for å rette på disse feilene. Prototype 1 hadde heller ikke implementert noen fysiske/taktile knapper. Disse ble først implementert i Prototype 2.

Feil og mangler i Prototype 1:

- Ingen fysiske/taktile knapper.
- Ingen kommunikasjon med sensorer (pulssensor og akselerometer) pga. feilkoblinger.
- Pulssensorens assosierte LED-er ville ikke lyse.

3.3.3 Prototype 2¹¹

Se Figur 39 for et bilde av *Prototype 2*. Komponentene som inngår i designet er listet i Tabell 11. Figur 40 viser kretsskjematikken for denne prototypen. Figur 41 og Figur 42 viser kretskortdesignet.



Figur 39: Prototype 2.¹²

Modul	Komponent	Produsent	Montering	Antall
Mikrokontroller	ACN52832	aconno GmbH	SMD/SMT	1
Akselerometer	MPU-6050	Invensense	SMD/SMT	1
Puls sensor	BH1790GLC-E2	Rohm Semiconductor	SMD/SMT	1
LED	SML-M13MTT86	Rohm Semiconductor	SMD/SMT	2
Spenningsregulator (LDO)	TLV75533PDBVR	Texas Instruments	SMD/SMT	1
Knapp	B3U-1000p	OMRON	SMD/SMT	3
1 μ F	Kondensator	AVX	SMD/SMT	2
10 μ F	Kondensator	N/A	SMD/SMT	1
2,2 μ F	Kondensator	N/A	SMD/SMT	1
100 nF	Kondensator	N/A	SMD/SMT	1

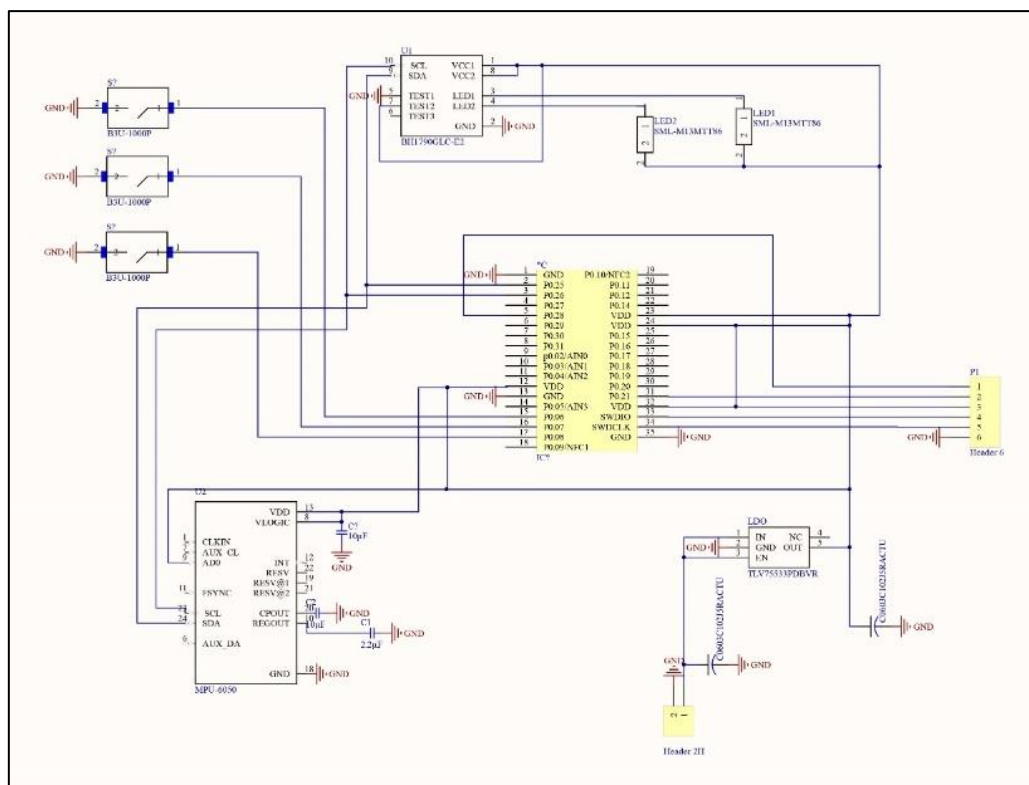
Tabell 11: Komponenter brukt i Prototype 2.

¹¹ Prototype 2 regnes som den endelige prototypen for dette prosjektet. Videre i teksten refereres Prototype 2 til som prototypen, pasientarmbåndet eller PasArm (forkortelse for pasientarmbånd).

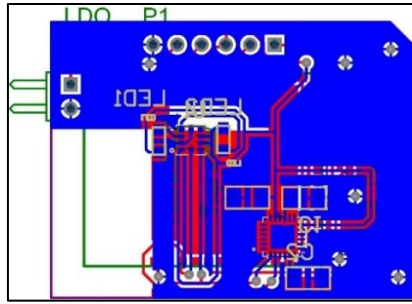
¹² Se Vedlegg I Kretskortdesign for større bilder av prototypen, kretsskjematikken og kretskortdesignet.

3.3.3.1 Forbedringer fra Prototype 1

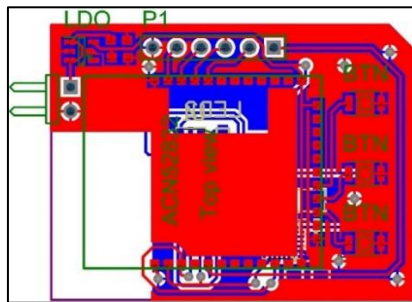
I *Prototype 2* er det forsøkt å rette opp feilene ved *Prototype 1*. Sensorene har dermed blitt koblet på nytt. I tillegg er det lagt til tre fysiske knapper for å gi prototypen ekstra funksjonalitet. En av knappene er dedikert til å være en reset-knapp, mens de to andre er varselknapper for tilkalling av hjelp. I tillegg til koblingene fra *Prototype 1*, er V_{DD} koblet til inngang 8 og 7, og jord til inngang 5 på pulssensoren (BH1790GLC-E2). MPU-6050 har i tillegg til den opprinnelige koblingen, fått V_{DD} til inngang 8 og 9. MPU6050 har også en 100nF kondensator fra pinne 10 til jord og en 2,2 nF kondensator fra pinne 20 til jord. En 10 nF avkoblingskondensator har også blitt plassert like ved V_{DD} -inngangen til akselerometeret for å redusere støy inn til denne komponenten. Forandringene som ble gjort, følger typisk applikasjon i databladene til de respektive komponentene. Korrigeringene er implementert i kretsskjematikken i *Figur 40*. Det endelige kretskortdesignet for *Prototype 2* vises i *Figur 41* og *42*.



Figur 40: Kretsskjematikk for Prototype 2.



Figur 41: Undersiden av kretskortdesignet (Prototype 2).



Figur 42: Oversiden av kretskortdesignet (Prototype 2).

3.3.3.2 Feil eller mangler

Denne prototypen fungerte stort sett slik den skulle. Mikrokontrolleren fungerte og var i stand til å kommunisere med mobiltelefon via Bluetooth. MPU6050 og BH1790GLC-E2 var i stand til å lese og videresende data til mikrokontrolleren. Prototypen klarte å oppnå greie pulsmålinger, selv om én av de grønne LED-ene ikke lyste. To av knappene fungerte heller ikke i denne prototypen.

Feil og mangler i Prototype 2:

- Én av pulssensor-LED-ene lyser ikke.
- To av knappene virket ikke (*Knapp 1* og *Knapp 3* i *Figur 43*).

3.3.4 Armbåndets funksjonalitet

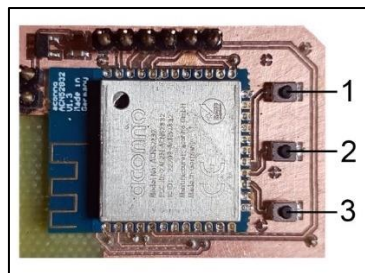
3.3.4.1 Annonseringsmodus

Når prototypen slås på, starter den Bluetooth-reklamering og er klar til å koble til mobiltelefon. I denne tilstanden blinker indikasjonsslyset med blått lys. Reklameringen fortsetter frem til enheten tilknyttes eller når enheten settes i *dyp søvnmodus* (ved å holde inne fysisk knapp på prototypen).

3.3.4.2 I tilkoblet modus

I tilkoblet modus vil indikasjonslampen blinke med rødt lys i takt med pulsslagene (følger firkantpuls fra *Figur X*). I tillegg vil fallsensoren aktiveres og lete etter potensielle fall. Dersom sensoren fjernes fra håndledd eller en annen overflate vil pulssensoren detektere dette. Da stoppes pulskalkuleringen og falldeteksjonen. I denne tilstanden blinker de grønne LED-ene én gang hvert femte sekund for å kontrollere om den er plassert på en overflate eller ikke. Dette gjøres for å spare strøm.

3.3.4.3 Knapper på armbåndet:



Figur 43: Knapper på prototypen nummerert.

Knapp	Kort trykk	Langt trykk
1 ¹³	Forespør akutt nødhjelp	N/A
2	Forespør assistanse	Slår RBG av/på
3 ¹⁴	Dyp søvnmodus	Koble fra enhet

Tabell 12: Liste over knapper på prototypen og deres funksjoner.

Man kan deaktivere indikatorlyset ved å holde inne *Knapp 2* i en liten periode. Man kan slå de på igjen ved å igjen holde inne *Knapp 2*. Dersom enheten tilbakestilles aktiveres alltid indikatorlysene, slik at man vet at enheten ikke er tilkoblet. For å sette enheten i dyp søvnmodus trykkes *Knapp 3* en gang. I dyp søvnmodus kan man vekke opp enheten med å trykke på hvilken som helst av knappene. Dersom *Knapp 1* trykkes, kan man forespørre akutt nødhjelp. Alternativt kan man forespørre etter assistanse med *Knapp 2*.

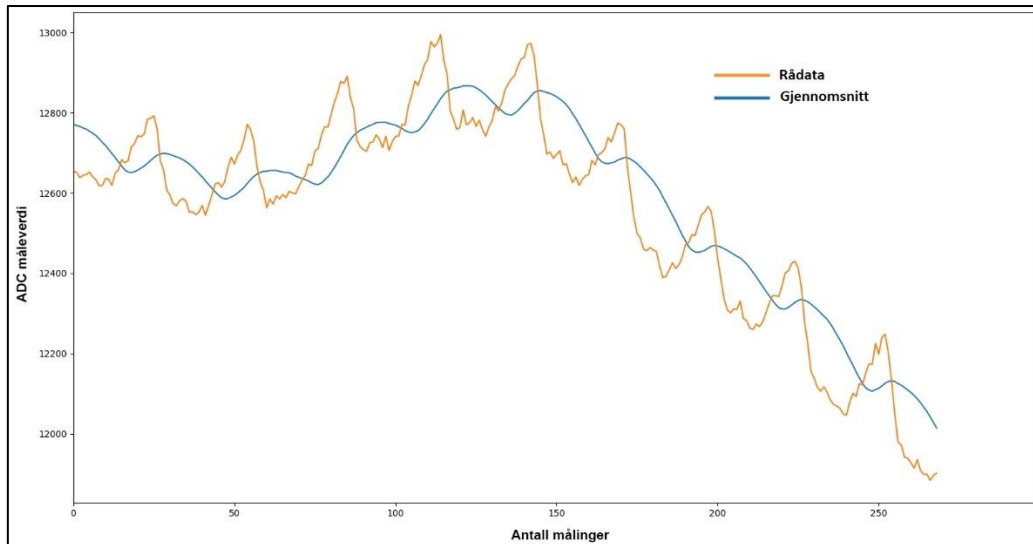
¹³ Fungerte ikke på prototypen.

¹⁴ Fungerte ikke på prototypen.

3.3.5 Kalkulering av pulsverdier

Dette avsnittet forklarer valgt implementasjon for å kalkulere puls. Pulsen kalkuleres av fastvaren i mikrokontrolleren (ACN32832) basert på rådata fra pulssensoren (BH1790GLC). Pulsdataen som mottas fra sensoren er vist i den oransje grafen i *Figur 44*.

3.3.5.1 Implementasjon



Figur 44: Graf som viser rådata (oransje) sammenliknet med utglattet data (blått).

Et gjennomsnitt kalkuleres av de 64 siste verdiene av rådataen fra pulssensoren. Gjennomsnittet av rådataen er vist i den blå grafen i *Figur 44*. Logikken sammenlikner disse to verdiene (rådata og gjennomsnitt) for å detektere pulsslag. Når rådata er høyere enn gjennomsnittet er logikken i tilstand høy og når rådata er lavere enn gjennomsnittet er logikken i tilstand lav. Slik kan man på en enkel måte detektere om et pulsslag har forekommet. Det detekteres én puls hver gang tilstanden går fra lav til høy.

3.3.5.2 Matematikk

Pulssensormålinger (og kalkuleringer) skjer med en frekvens på 32 Hz eller hvert 31,25 millisekund. Tiden som har gått de x antall siste pulsslagene bestemmer et anslag for hjerterate (HR). Formelen som brukes er vist nedenfor (se *Formel 2*). I formelen kalles en av faktorene *antall målinger (...)*. Dette er antall målinger som har blitt gjort i perioden x antall pulser har blitt detektert. Feilmarginen her vil ligge på $\pm 31,25$ millisekunder.

$$HR \approx \frac{(x \text{ antall pulsslag}) \times (\text{antall millisekunder i et minutt})}{(\text{antall målinger i løpet av de } x \text{ siste pulsslagene}) \times 32}$$

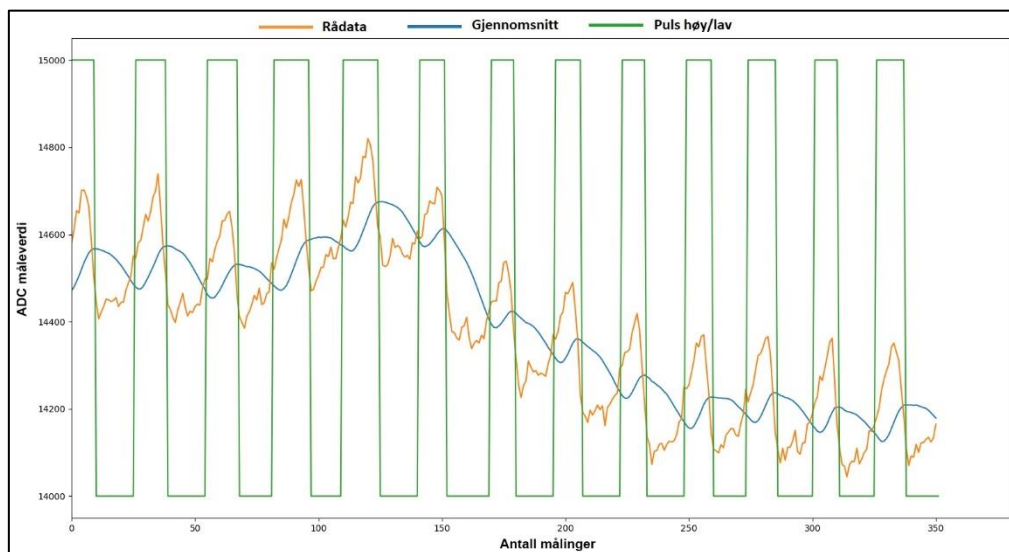
Formel 2: Generell kalkulering av puls.

Formel 3 viser kalkulasjon av hjerterate (HR) spesifikt for nåværende implementasjon av pasientarmbåndet. Her er antall pulsslag satt til fire.

$$HR \approx \frac{4 \times 60\,000}{(\text{antall målinger i løpet av de 4 siste pulsslagene}) \times 32}$$

Formel 3: Kalkulering av puls i nåværende implementasjon av prototypen.

3.3.5.3 Eksempel



Figur 45: Graf med rådata, gjennomsnitt og firkantpuls.

Figur 45 viser tilsvarende data som i *Figur 44*, men i denne figuren har det blitt tilføyd en firkantpuls som indikerer om pulslogikken er *høy* eller *lav*. Basert på informasjonen i figuren kan det kalkuleres et anslag for HR. Første overgang fra tilstand lav til høy i figuren skjer ved måling 25. Den fjerde overgangen skjer rundt måling 140. *Antall målinger (...)* blir dermed differansen mellom 25 og 140 som er 115 målinger. Vi kan plassere dette inn i *Formel 2*. Resultatet vil da bli 65 slag i minuttet (BPM). Denne verdien er innenfor vanlig hvilepulsområde.

3.3.5.4 Marginer

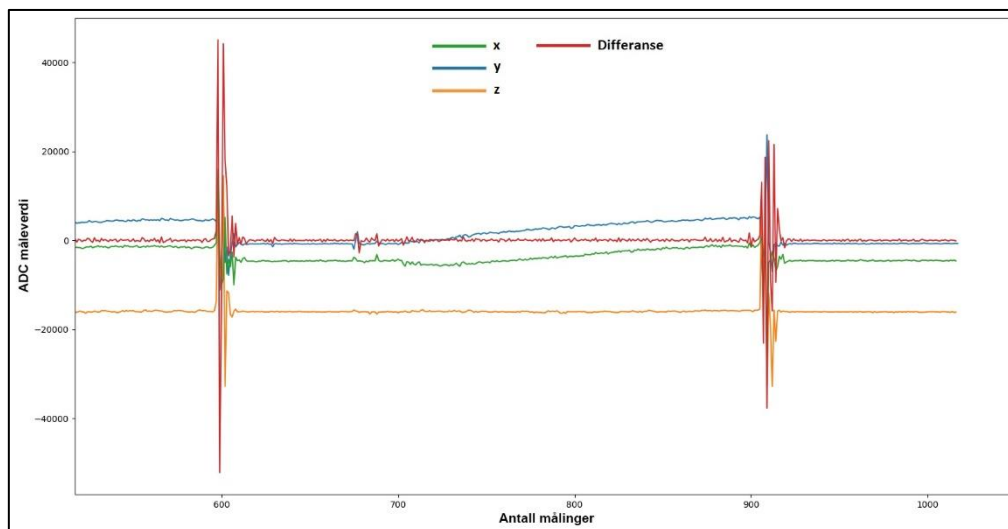
Tilstanden i logikken går ikke høy umiddelbart når rådataen er høyere enn gjennomsnittet (se *Figur 45*). Logikken setter krav om at rådataverdien minst skal være 10 høyere enn gjennomsnittet for å detektere en puls. I tillegg må rådataen være 10 mindre enn gjennomsnittet for at tilstanden går lav. Dette gjøres for å unngå feil hvor støy kan påvirke rådataen. Uten denne ekstra logikken ville støy kunne føre til at det ble detektert overganger rett etter hverandre, noe som ville ha gitt store avvik på HR. Det er også gjort slik at logikken ikke kan bytte tilstand før etter 4 nye målinger. Noe negativt med dette vil være at puls maksimalt vil kunne være 234 slag i sekundet. Allikevel vil det være vanskelig å kalkulere HR når pulsen er veldig høy med valgt pulssensor. Den lave målefrekvensen (32 Hz) vil kunne føre til større avvik dess høyere pulsen er. 234 BPM er også en del høyere enn vanlig makspuls[38]. Det antas at implementasjonen vil kunne gi en god antakelse for vanlig HR opp til en viss HR.

3.3.6 Falldeteksjon

I dette avsnittet forklares implementasjonen for å detektere fall ved hjelp av akselerometeret på *TDK InvenSense MPU-6050*. Implementasjonen for falldeteksjon er *relativt enkel*. Man har akselerometerdata fra tre akser (*X*, *Y* og *Z*) som baserer seg på g-krefter. Først kalkuleres differansen mellom forrige og nåværende måling for hver av aksene. Deretter summeres differansen fra alle aksene til en felles verdi. De 64 siste verdiene av denne verdien lagres. Disse verdiene representerer endringer i akselerasjon ca. de siste to sekundene (ettersom nye målinger gjennomføres hvert 31,25 sekund).

3.3.6.1 Eksempel

I *Figur 46* vises differansen for hver av aksene (*x* er grønn, *y* er blå og *z* er oransje). I tillegg vises summen av disse i rødt. I denne figuren ble enheten sluppet fra ca. 20 centimeters høyde to ganger. Fallene kommer tydelig frem i figuren. Verdien som holder på summen går drastisk opp når man slipper enheten, og i det den treffer bakkeflate går lenger ned. Ved å sammenlikne differansen mellom *topp-* og *bunnpunkt*, vil man kunne gi en indikasjon på om enheten har falt. Dersom fall detekteres, blir det sendt et varsel til mobiltelefon via BLE.



Figur 46: Graf som illustrerer falldetektering.

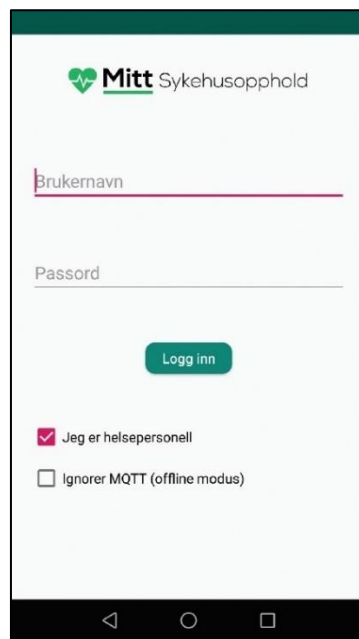
3.3.7 «Mitt Sykehusopphold» (mobilapplikasjon)

I denne seksjonen presenteres den endelige implementasjonen av mobilapplikasjonen *Mitt Sykehusopphold*, som vi har valgt å forkorte til *MSO*. Både funksjonelle og ikke-funksjonelle implementasjoner for de ulike app-sidene/aktivitetene vil bli utdypet i dette delkapitlet¹⁵.



Figur 47: Logo for appen «Mitt Sykehusopphold».

3.3.7.1 Innloggingside



Figur 48: Implementert innloggingside i MSO-appen.

Se *Figur 48* for et skjermbilde av den endelige implementasjonen for innloggingsiden i MSO-appen. På toppen av siden vises en logo prosjektgruppen har laget for Mitt Sykehusopphold. Sluttbrukere kan huke av en avkryssingsboks for å velge tilhørighet (pasient eller helsepersonell). Det kan også hukes av om man ønsker å logge på i *offline*-modus (avkoblet modus). Ønsker man å bruke *online*-funksjonalitet legger man inn brukernavn og passord for å koble seg til en MQTT-broker. Denne prosessen håndteres av en bakgrunns-tjeneste (*MqttService.java*) som bruker inntastet brukernavn og passord til å koble til en CloudMQTT-instans. MQTT-tilkoblingen brukes videre i appen for å sende og motta MQTT-meldinger via megleren/broker-en. Dersom brukernavn eller passord er feil, får man beskjed om dette.

¹⁵ Kildekoden til MSO-appen kan lastes ned fra github.com/robino16/mso. Kildekoden er også lagt som vedlegg til rapporten (se *Vedlegg I Kildekode mobilapp*). Prosjektet *mso-login* kan åpnes og kjøres via Android Studio.

3.3.7.2 Innlogging: Eclipse Paho

For MQTT-tilkoblingen benyttes et bibliotek som heter *Eclipse Paho* av *Eclipse Foundation*[113]. Det ble tatt utgangspunkt i et offisielt eksempelprosjekt for å bruke dette biblioteket i Android Studio¹⁶. Eksempelprosjektet ligger tilgjengelig på GitHub. Her står det også beskrevet hvordan man inkluderer Eclipse Paho-biblioteket inn i Android Studio-prosjekter. Eksempel-prosjektet tar utgangspunkt i at MQTT-funksjonaliteten skal fungere i én enkelt aktivitet. I MSO-appen fikk man bruk for MQTT-tjenesten også etter innloggingsiden. For å bevare MQTT-tilkoblingen, ble det laget en egen tjeneste (*service*) navngitt `MqttService.java` som håndterer tilkoblingen i bakgrunnen[90]. Tjenesten blir startet når man trykker på *Logg inn*-knappen. Brukernavn og passord gis til servicen via det som kalles `putExtra()`- og `getExtra()`-funksjonskall via en klasse som kalles *Intent*[92].

3.3.7.3 Innlogging: CloudMQTT

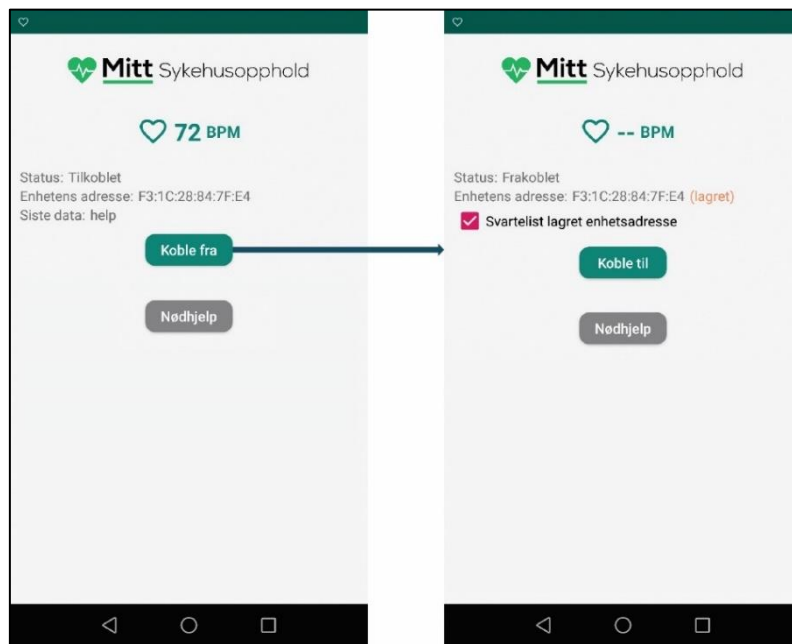
For å koble til *CloudMQTT* med *Eclipse Paho* må man definere i koden at *MQTT versjon 3.1* skal brukes[114]. I tillegg må navn og passord være lagt inn i CloudMQTT-klienten fra før av. Disse kan legges inn manuelt på CloudMQTT sin hjemmeside eller via terminal-vindu på PC med et programtillegg som kalles *Eclipse Mosquitto*[84]. For å kunne motta eller sende meldinger via CloudMQTT må brukeren gis riktige tillatelser inne i CloudMQTT-klienten. Man kan gi alle brukere skrive/lese-rettigheter ved å legge til en *ACL-pattern* (mønster) med verdi `#` og gi dette mønsteret lese/skrive-tilgang (se *Figur 49*). Etter dette kan man abonnere på emner (*topics*) eller publisere meldinger fra Android-appen. Fordi det i MSO-appen brukes en felles tjeneste for all MQTT-funksjonalitet, kan man abonnere på eller publisere meldinger i flere aktiviteter i koden. *MqttService.java* signaliserer aktivitetene og motsatt via det som kalles for *BroadcastReceiver*[91].

ACLs		
Note: <ul style="list-style-type: none"> There are two types of ACL rules, topic and pattern. Topic ACLs is applied to a given user. Pattern ACLs is applied to all users. Use # for multi level wildcard acl. Use + for single level wildcard acl. Creating and deleting users and ACLs are asynchronous tasks and may take up to a minute. Poll list APIs to see when ready. 		
For API docs look at HTTP API		
Type	Pattern	Read/Write
pattern	#	true/true
		<button>Delete</button>

Figur 49: ACL-mønster for å gi alle MQTT-brukere lese- og skrivetilgang.

¹⁶ Eksempelprosjektet ligger tilgjengelig her: github.com/eclipse/paho.mqtt.android

3.3.7.4 Pasientside



Figur 50: Implementert hovedside for pasienter i MSO-appen.

Se Figur 50 for to skjermbilder av pasientenes hovedside i MSO-appen. Denne siden er dedikert til Bluetooth-administrasjon og direkte kommunikasjon med armbåndet. Når man først logger inn som pasient starter man i tilstanden *Frakoblet*. Når sluttbrukeren trykker *Koble til* vil appen søke etter Bluetooth-enheter. Appen prøver automatisk å koble seg til enheter med enhetsnavn *RTA*¹⁷ som leses ut fra GATT-karakteristikk fra tilgjengelige enheter. Dersom en RTA-enhet oppdages, skjer tilkoblingen automatisk. Når enhetene er tilkoblet er det klart for overføring av data. Eventuell pulsdata legges i toppen av skjermen i et dedikert felt for dette. Dersom armbåndet ikke er i stand til å lese puls på håndleddet, vil det dukke opp en oransje varsel-tekst under puls-feltet hvor det står; «Pulsverdien oppdateres ikke». Dersom sluttbrukeren kobler seg fra armbåndet blir enhetsadressen midlertidig lagret. Man kan deretter huke av et felt som svartelister den lagrede enhetsadressen. Dette kan være praktisk dersom appen knytter seg til feil armbånd. Det er også en dedikert knapp for *Nødhjelp*. Dersom man er tilkoblet via MQTT vil man kunne sende en forespørsel om akutt assistanse til helsepersonellet ved å trykke denne knappen.

3.3.7.5 Pasientside: android-BluetoothLeGatt

MSO-appen tar utgangspunkt i et Bluetooth LE-eksempelprosjekt fra *Google* med tittelen *android-BluetoothLeGatt*. I eksempelprosjektet er søk og håndtering av tilkoblet enhet separert i to ulike aktiviteter. I MSO-appen er disse slått sammen til én aktivitet som heter *PatientMainActivity*. I tillegg foregår tilkoblingen mer i bakgrunnen i MSO-appen. Når man trykker på *Koble til*-knappen sendes den

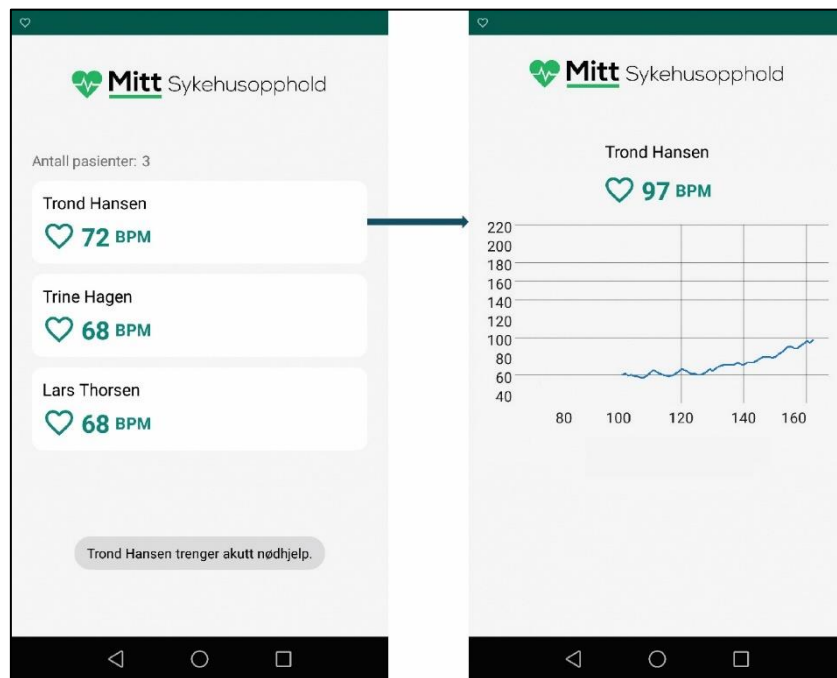
¹⁷ RTA er navnet gitt til pasientarmbåndet via fastvareprogrammet. RTA står for Robin, Thomas og Aksel.

boolske verdien sann (true) til en funksjon som heter *scanLeDevice()*. I denne funksjonen blir da en *callback*-funksjon kalt *mLeScanCallback* gitt til en Bluetooth-adapteren deklart i *PatientMainActivity*-klassen. Denne callback-funksjonen kalles hver gang en enhet oppdages. I MSO-prosjektet blir hver enhet som oppdages validert. Dersom enhetsnavnet er *RTA* prøver appen å tilknytte til enheten. Ved tilkobling går appen inn i tilstanden *Tilkoblet*. I en service kalt *BluetoothLeService* som følger med eksempelprosjektet, setter man opp notifikasjon på ønsket karakteristikk. For å motta TX-data fra nRF52-mikrokontrolleren, definerer man da en descriptor med verdi *0x2902* på TX-karakteristikken (*UUID: 6e400003-b5a3-f393-e0a9-e50e24dcca9e*). Ved mottakelse av data fra mikrokontrolleren sendes denne dataen fra *BluetoothLeService* til aktiviteten *PatientMainActivity* via en *BroadcastReceiver*[91]. Dataen kan deretter håndteres og eventuelt vises på skjermen.

3.3.7.6 Pasientside: MQTT

Pasientsiden tar i bruk MQTT-tilkoblingen fra tidligere. Data fra armbåndet blir publisert via den tilknyttede CloudMQTT-broker-en. I tillegg publiseres eventuelle forespørsler etter nødhjelp. Nødhjelp-forespørsler kan sendes enten ved å trykke fysiske knapper på armbåndet eller via *Nødhjelp*-knappen i appen. MQTT-meldingene som sendes kan inneha forskjellig informasjon, slik som pasientens brukernavn, navn eller pulsverdi. I aktuell implementasjon formateres MQTT-meldingene på denne måten: *[brukernavn][data]*, hvor *data* enten kan være *H* for *hjelp* eller oppdatert pulsverdi fra armbåndet. Meldingene publiseres på emnet *patient* og kan deretter mottas og dekodes fra helsepersonellsiden.

3.3.7.7 Helsepersonellsiden



Figur 51: Implementasjon av helsepersonell-siden i MSO-appen.

Se Figur 51 for to skjermbilder fra den implementerte helsepersonell-siden i MSO-appen. I skjermbildet til venstre ser man hovedsiden til personellet. På hovedsiden kan man se en liste over tilgjengelige pasienter. Listen er dynamisk og oppdaterer seg når nye pasienter kobler seg til MQTT-nettet og publiserer meldinger. I tillegg oppdateres pulsverdier direkte dersom pasienter er tilkoblet. Dialogboksen nede på venstre skjermbilde er det som kalles en *Toast*[94] eller skåling. En slik boks dukker opp når en pasient sender en forespørsel etter nødhjelp. Dersom sluttbruker trykker på en av pasientene kommer man inn til en ny aktivitet kalt *PatientActivity* (skjermbildet til høyre i Figur 51). I *PatientActivity*-klassen oppdateres fremdeles pulsverdien til pasienten basert på MQTT-meldinger. I tillegg vises en graf over pulsverdien. Grafen oppdateres direkte.

3.3.7.8 Helsepersonellsiden: MQTT

Helsepersonell-siden er knyttet til MQTT-servicen nevnt tidligere. Denne blir satt opp til å abonnere på MQTT-meldinger som publiseres på emnet (topic'en) *patient*. Ved mottatt melding (formatert *[brukernavn][data]*) overføres meldingen videre til *PersonnelMainActivity* via en *BroadcastReceiver*[91] (også nevnt tidligere). I *PersonnelMainActivity* valideres og tolkes meldingen. Brukernavnet definerer hvilken pasient som sendte meldingen. Er det første symbolet i dataen en *H*, vises *Toast*/skåling nederst på skjermen som sier: «*brukernavn* trenger akutt nødhjelp.» (se Figur 51). Dersom dataen er en tallverdi, antas det at dataen er en ny puls-verdi. Programmet finner da aktuell pasient i listen og oppdaterer puls-verdien som vises på skjermen. Cellene i listen er tilpasset MSO-applikasjonen, men andre verdier kunne også blitt vist i cellene, slik som f.eks. romnummer.

4 Validering og testing

I dette kapitlet blir prototypekretskortet samt mobilappen testet og validert på bakgrunn av målsetningen. Blant annet blir nøyaktigheten av pulssensoren og fallsensoren validert. Strømforbruket til prototypen måles slik at batteritid kan kalkuleres. Funksjonalitet i mobilappen blir også testet på ulike mobiltelefoner.

4.1 Målenøyaktighet

For å teste målenøyaktigheten til prototypen, ble det gjennomført sammenlikninger med to andre aktivitetsarmbånd (Samsung *Galaxy Watch 42 mm* og *POLAR H1*). Resultatet av sammenlikningene er vist i tabellene nedenfor (se Tabell 13, Figur 14 og Figur 15). Alle målingene ble gjort med intervall på ca. 2 minutter. Puls målinger ble målt i slag per minutt (BPM).

Målinger i sittende stilling:

Minutter siden start	Samsung Galaxy Watch (#1)	POLAR H1 (#2)	Prototype (PasArm)	Måleavvik sammenliknet med #1	Måleavvik sammenliknet med #2
1	86 BPM	86 BPM	84 BPM	2,3 %	2,3 %
3	81 BPM	84 BPM	78 BPM	3,7 %	7,1 %
5	84 BPM	86 BPM	81 BPM	3,6 %	5,8 %
7	83 BPM	80 BPM	79 BPM	4,8 %	1,3 %
9	83 BPM	95 BPM	82 BPM	1,2 %	3,5 %
Gjennomsnitt	83,4 BPM	84,2 BPM	80,8 BPM	3,12 %	4,00 %

Tabell 13: Sammenlikning av pulsmålinger i sittende stilling.

Målinger i stående stilling:

Minutter siden start	Samsung Galaxy Watch (#1)	POLAR H1 (#2)	Prototype (PasArm)	Måleavvik sammenliknet med #1	Måleavvik sammenliknet med #2
1	87 BPM	86 BPM	86 BPM	1,1 %	0,0 %
3	86 BPM	85 BPM	87 BPM	1,1 %	2,3 %
5	89 BPM	87 BPM	89 BPM	0,0 %	2,2 %
7	87 BPM	88 BPM	87 BPM	0,0 %	1,1 %
9	86 BPM	89 BPM	89 BPM	3,4 %	0,0 %
Gjennomsnitt	87,0 BPM	87,0 BPM	87,6 BPM	1,12 %	1,12 %

Tabell 14: Sammenlikning av pulsmålinger i stående stilling.

Målinger i gående tilstand:

Minutter siden start	Samsung Galaxy Watch (#1)	POLAR H1 (#2)	Prototype (PasArm)	Måleavvik sammenliknet med #1	Måleavvik sammenliknet med #2
1	96 BPM	103 BPM	105 BPM	8,6 %	1,9 %
3	98 BPM	111 BPM	109 BPM	10,1 %	1,8 %
5	107 BPM	108 BPM	98 BPM	8,4 %	9,3 %
7	109 BPM	112 BPM	106 BPM	2,8 %	5,4 %
9	107 BPM	112 BPM	109 BPM	1,8 %	2,7 %
Gjennomsnitt	103,4 BPM	109,2 BPM	105,4 BPM	6,34 %	4,22 %

Tabell 15: Sammenlikning av pulsmålinger i gående tilstand.

Ved å analysere tabellen konkluderes det med at prototypen avviker mindre enn ± 10 % sammenliknet med andre aktivitetsarmbånd (i sittende, stående eller gående tilstand). Avvik kan for eksempel komme av lys fra eksterne kilder som forstyrrer armbåndet når armen er i bevegelse. Store avvik forekom når prototypen ble gått med ute i mye lys. Prototypen måtte dermed dekkes til for å fungere optimalt.

4.1.1.1 Fallsensor

Fallsensoren anses å være noe unøyaktig. Det er ikke alltid fall detekteres. Det er i tillegg vanskelig å teste fallsensoren på prototypen ordentlig. Det er vanskelig å holde enheten mens man simulerer et fall. I tillegg kan man fremprovosere falske fall ved å riste på enheten.

4.2 Strømforbruk

I en ideell konstruksjon vil det være ønskelig at armbåndet benytter minimalt med strøm, slik at pasienter ikke behøver å lade/bytte eller ta av armbåndet så ofte. Prosjektgruppen satte som minimumskrav at batteritiden til prototypen minst skulle være 5 timer, selv om den ideelt sett helst burde være på minst en hel uke. Det ble gjennomført strømmålinger av prototypen (*Prototype 2*) for å undersøke strømforbruket.

4.2.1.1 I «deep sleep»-modus

Det ble først tatt i bruk et *multimeter* for å måle DC-strøm direkte. Når mikrokontrolleren ble satt i *deep sleep*-modus (dyp søvnmodus), lå strømforbruket relativt stabilt på ca. 0,5 mA, noe som anses som høyt ettersom det i databladet til mikrokontrolleren (ACN52832), står spesifisert at strømforbruket ligger i mikro-ampere-siktet (mellom 0,7 μ og 1,9 μ) når enheten står i *Ultra-low power*-moduser[51]. Det høye forbruket kan skyldes strømtap i spenningsregulatoren eller komponenter.

4.2.1.2 Målemetode med oscilloskop

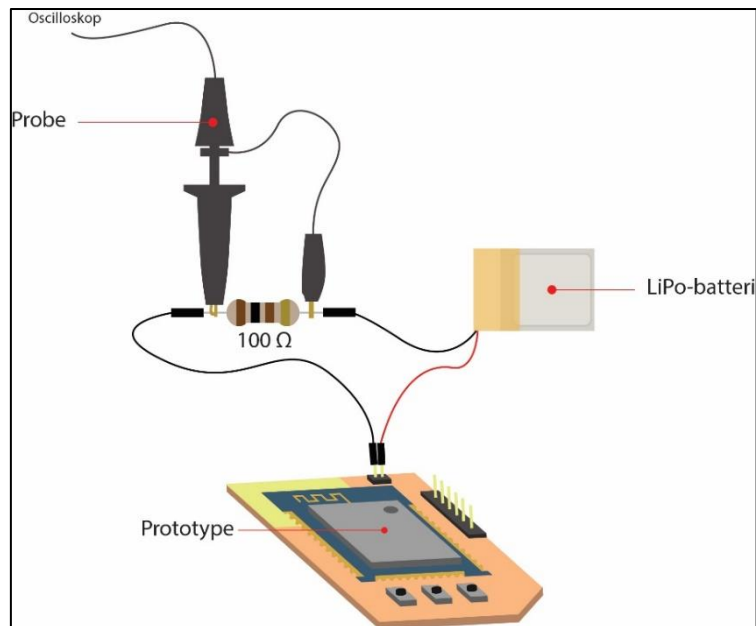
Når enheten var påslått kunne man visuelt lese at strømmen varierte mellom 0,5 mA og 5 mA (grovt anslag) på multimeteret. Strømtrekket varierte i pulser med ulik varighet og intensitet. På grunn av den høye variasjonen, ble det tatt i bruk et *oscilloskop* for å gjennomføre mer nøyaktige målinger av strømforbruket. Måleproben fra oscilloskopet ble satt over en 100 Ω -motstand mellom jord-pin-en på prototypen og jord på LiPo-batteriet (se *Figur 52*). Med oscilloskopet kan man da lese av spenningen over motstanden grafisk, og kalkulere et estimat for strømtrekket fra batteriet. Det må tas forbehold til at spenningen varierer mellom de ulike strømpulsene og at pulsene har ulik varighet. Klarer man å finne et godt estimat for gjennomsnittlig spenningsfall over motstanden, finner man gjennomsnittlig strømforbruk med ohms lov (se *Formel 4*). For å finne ut hvor lang batteritid man vil kunne få i timer (h), kan man dividere kapasiteten til batteriet i milliamperetime (mAh) med strømtrekket i milliampere (mA). For å ta hensyn til eksterne faktorer som kan påvirke batteritiden, kan resultatet multipliseres med en faktor på 0,7[115] (se *Formel 5*). Det endelige resultatet vil være et greit estimat for batteritid.

$$U = RI, \quad I = \frac{U}{R}$$

Formel 4: Ohms lov.

$$\text{Batteritid [timer]} = \frac{\text{Batterikapasitet [mAh]}}{\text{Gjennomsnittlig strømforbruk [mA]}} \times 0,7$$

Formel 5: Kalkulasjon av batteritid.[115]



Figur 52: Oppkobling av oscilloskop for å måle strømforbruk.

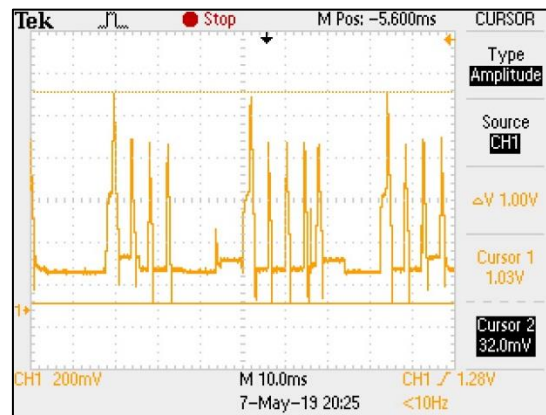
4.2.1.3 Scenarier

Med oscilloskop ble det gjort målinger av strømforbruket i tre scenarier. Det første scenarioet er når enheten klarer å lese pulldata og overfører til mobiltelefon over Bluetooth Low Energy. Det andre scenarioet var dersom mobiltelefon er tilkoblet via Bluetooth Low Energy, men pulsarmbåndet ikke klarer å lese puls. I dette scenarioet vil de to LED-ene assosiert med pulssensoren blinke en gang ca. hvert femte sekund. Det siste scenarioet er når enheten er i *advertising*-modus (annonserings-modus). Strømforbruk for de ulike scenarioene er diskutert videre i teksten.

Strømforbruk-scenarier:

- Scenario 1: Tilkoblet via Bluetooth Low Energy og klarer å lese/overføre pulldata til mobiltelefon.
- Scenario 2: Tilkoblet via Bluetooth Low Energy, men klarer ikke lese puls.
- Scenario 3: *Advertising*-modus (annonseringsmodus).

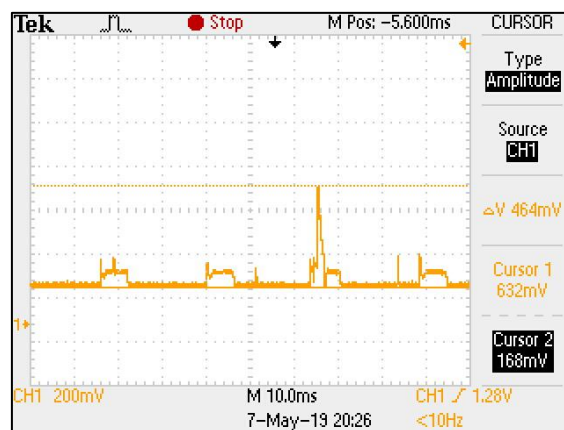
4.2.1.4 Scenario 1: Tilkoblet via BLE og klarer å lese puls



Figur 53: Oscilloskop-måling når BLE er tilkoblet og armbåndet klarer å lese puls.

I Figur 53 ser man hvordan spenningen over motstanden endrer seg når enheten er tilkoblet via Bluetooth Low Energy og leser pulsdata. I et estimat ved å studere dette skjermbildet (ved å telle piksler for å finne lengde og høyde på pulsene), ble det funnet at den gjennomsnittlige spenningen ligger på ca. 387 mV. Det vil si at gjennomsnittlig strømforbruk ligger på ca. 3,87 mA. Med et batteri med kapasiteten 70 mAh vil batteritiden være ca. 18 timer. Tar vi hensyn til feilmargin og eksterne faktorer (multipliserer resultatet med 0,7) finner vi at minimal batteritid ligger på ca. 12 timer. I Figur 53 ser vi at spenningen stort sett ligger over ca. 0,19 V, og at den i de høyeste pulsene kan ligge på ca. 1,00 V. Strømtrekket for scenario 1 ligger dermed mellom 1,9 mA og 10,0 mA.

4.2.1.5 Scenario 2: Tilkoblet via BLE, men klarer ikke å lese puls

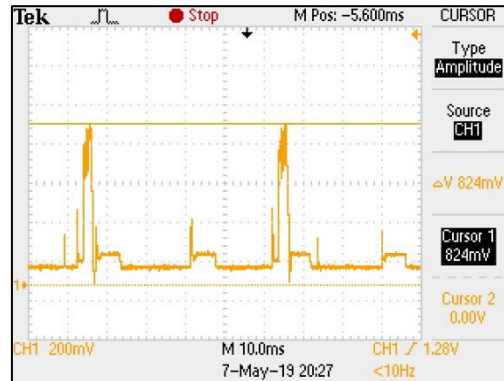


Figur 54: Oscilloskop-måling når BLE er tilkoblet, men puls ikke kan leses.

I Figur 54, ser vi hvordan spenningen over motstanden endrer seg når enheten er tilkoblet via Bluetooth Low Energy, men ikke klarer å lese puls. I et estimat ved å studere dette skjermbildet, ble det funnet at den gjennomsnittlige spenningen ligger på rundt 204 mV. Det vil si at gjennomsnittlig strømforbruk ligger på rundt 2,04 mA. Med et batteri med kapasiteten 70 mAh vil batteritiden være ca. 34 timer. Tar

vi hensyn til feilmargin, finner vi at enheten i alle fall bør ha en batteritid på 24 timer. Strømtrekket for scenario 2 ligger mellom 1,7 mA og 6,3 mA.

4.2.1.6 Scenario 3: «Advertising»-modus (annonseringsmodus)



Figur 55: Oscilloskop-måling når enheten er i «advertising»-modus.

I Figur 55, ser vi hvordan spenningen over motstanden endrer seg når enheten er i *advertising*-modus. *Advertising*-modus er tilstanden når enheten klar for tilknytning. I et estimat ved å studere dette skjermbildet, ble det funnet at den gjennomsnittlige spenningen ligger på rundt 185 mV. Det vil si at gjennomsnittlig strømforbruk ligger på rundt 1,85 mA. Med et batteri med kapasiteten 70 mAh vil batteritiden være i underkant av 38 timer. Tar vi hensyn til feilmargin, finner vi at enheten i alle fall bør ha en batteritid på 26 timer i denne modusen. Strømforbruket i dette scenarioet ligger mellom ca. 1,03 mA og 8,24 mA.

4.2.1.7 Oppsummert

I Tabell 16, ser vi en oversikt over strømforbruk i de ulike scenarioene. Vi ser også en sammenlikning av batteritid dersom kapasiteten til batteriet potensielt hadde blitt økt til 300 mAh eller 600 mAh.

Scenario	Gjennomsnittlig strømforbruk	Batteritid		
		70 mAh	300 mAh	600 mAh
1: BLE og pulsmåling	3,87 mA	> 12 timer	> 54 timer	> 4.5 dager
2: BLE uten pulsmåling	1,64 mA	> 30 timer	> 5.3 dager	> 10.6 dager
3: «Advertising»	1,85 mA	> 26 timer	> 4.7 dager	> 9.4 dager
«Deep Sleep»	0,5 mA	> 4 dager	> 17.5 dager	> 35 dager

Tabell 16: Strømforbruk og batteritid i ulike scenarioer/moduser.

4.3 Appens funksjonalitet

MSO-appen ble i utgangspunktet utviklet og testet på en HUAWEI P9-smarttelefon med Androidversjon 7.0 Nougat (API-nivå 24)[96]. Ved ferdig implementering av applikasjonen ble sentrale funksjoner (BLE og MQTT) testet på ulike mobiltelefoner (se *Tabell 17*).

Mobiltelefon	Androidversjon	BLE	MQTT
HUAWEI P8	6.0 Marshmallow	Appen krasjer etter velkomstbilde.	
HUAWEI P9	7.0 Nougat	Fungerer	Fungerer
HUAWEI Mate 10	9 Pie	Ikke testet	Fungerer ikke
HUAWEI Mate 10 lite	8.0.0 Oreo	Ikke testet	Fungerer
Motorola Moto G ⁴ Plus	7.0 Nougat	Fungerer ikke	Fungerer
Samsung S5	6.0.1 Marshmallow[116]	Ikke testet	Fungerer

Tabell 17: Sentral appfunksjonalitet testet på ulike mobiltelefoner.

Med utgangspunkt i *Tabell 17* kan det fastslås at MSO-appen i gjeldende tilstand ikke fungerer på alle smarttelefoner. Videre feilsøking må gjøres dersom appen skal kunne brukes på flere smarttelefoner.

4.3.1 Validering av konseptet

For å validere at konseptet som gruppen arbeider med er relevant, hadde gruppen et styringsmøte med *Sørlandet Sykehus HF (SSHF)* 6. mars 2019. På dette møtet stilte SSHF med flere avdelingsledere for å høre på en presentasjon av oppgaven. Tonje Mathilde Holand Salgado fra Bouvet presenterte deres konsept. Prosjektgruppen presenterte også prosjektoppgaven for SSHF i denne konteksten.

Tilbakemeldingene fra avdelingslederne var positive. Flere så nytteverdien i pasientarmbåndet og mobilapplikasjonen. Det ble diskutert relevansen for diverse målingsalternativer og funksjoner som kan implementeres i fremtidige design. SSHF sa at de ville bidra til å bringe konseptet videre. De skulle be om ressurser slik at gruppen kunne få bedre innblikk i sykehushverdagen til personellet.

14. mai 2019 fikk Bouvet en ny tilbakemelding fra SSHF om at samarbeidet mellom SSHF og Bouvet måtte avsluttes. Begrunnelsen for dette var at et lignende tjenester allerede er under utvikling i regional regi (Helse Sør-Øst). Denne utviklingen var igangsatt av store etablerte aktører som blant annet *Diffia* og *Imatis*. Tilbakemeldingen som gruppen fikk tilsier at konseptet er av høyeste relevans. Dessverre er konseptet allerede i utvikling på et høyere nivå.

5 Diskusjon

I denne delen av rapporten, diskuteres problemstillingene, resultatene og videre arbeid for prosjektet.

5.1 Hovedproblemstilling

Å utvikle et e-helse-armbånd som aktivt overvåker helsedata og viser denne dataen direkte i en dedikert mobilapplikasjon.

Hovedproblemstillingen i dette prosjektet omhandler utvikling av et e-helse-armbånd som aktivt overvåker helsedata og viser denne dataen direkte i en dedikert mobilapplikasjon. I løpet av prosjektet, har det blitt utviklet et e-helse-armbånd som er i stand til å måle puls, registrere fall og som har mulighet for å tilkalle hjelp via fysiske knapper. Det har også blitt utviklet en applikasjon som kan kommunisere med e-helse-armbåndet gjennom Bluetooth Low Energy og motta helsedataene. Denne applikasjonen sender også helsedataene videre via MQTT slik at andre mobile enheter som er tilkoblet internett kan motta disse helsedataene.

For å oppfylle hovedproblemstillingen, finnes det forskjellige innfallsvinkler. Hvilke helsedata som er mest relevant kan for eksempel variere fra situasjon til situasjon. I denne oppgaven valgte gruppen å måle puls. Andre helsedata som kunne vært relevant å loggføre, ville vært blodtrykk og oksygenmetningen i blodet. Mobilapplikasjonen er designet for en Android plattform, noe som ville utelukket enkelte brukere i en sykehussetting. I kontekst av prosjektet var det urealistisk å utvikle applikasjonen for flere enn denne plattformen. Dersom konseptet skulle blitt tatt i bruk, ville det vært nødvendig å utvikle mobilappen for flere plattformer. Kommunikasjon mellom e-helse-armbåndet og telefon kunne også vært løst på andre måter. Informasjonen som armbåndet leverer kunne blitt mottatt av en inngangsport (*gateway*) og videresendt over internett til mobilapplikasjonen. Resultatet som gruppen kom frem til, baserte seg på hvilke ressurser som var tilgjengelige samt tiden som gruppen hadde til disposisjon. Kunnskap og tidligere erfaring spilte også en rolle i hvilke løsningsalternativer som ble valgt for å oppfylle problemstillingen.

5.2 Sekundær problemstilling

Vil kontinuerlig overvåking av pasienters helseverdier via et pasientarmbånd bedre pasientopplevelsen eller effektivisere arbeidsprosesser blant ansatte?

Baktanken med pasientarmbåndet er å forbedre hverdagen til både pasienter og helsearbeidere. Dersom pasientarmbåndet skal bli brukt i en sykehussetting, er det viktig at produktet har denne tiltenkte effekten. For at produktet skal ha denne tiltenkte effekten, er det essensielt at nøyaktigheten til sensorene er gode nok for en sykehussetting. Applikasjonen må også være intuitiv og brukervennlig, da man må ta hensyn til at enkelte pasienter og helsearbeidere er mindre innarbeidet i moderne teknologi.

Først og fremst vil et pasientarmbånd gi pasientene tilgang til sine egne helsedata. For mange kan det være betryggende eller behagelig å kunne forsikre seg om at helsedataene sine samsvarer med det normale. Dersom helsedataene ikke samsvarer med det normale, kan pasienten også varsle helsepersonell og få hjelp umiddelbart. Med det nåværende *rød snor*-systemet er det vanskelig å vite når en varsling er kritisk. Med *Mitt Sykehusopphold*-konseptet vil man derimot kunne aktivere alarmer med ulik alvorlighetsgrad. På denne måten kan helsepersonell selv prioritere hvilke varsler som er viktigst å håndtere først. Det er også mulig å legge til funksjonalitet hvor pasientarmbåndet sender ut en alarm dersom helsedataene ikke samsvarer med normale verdier. Med denne funksjonaliteten implementert kan manuell varsling reduseres betraktelig og dette burde skape trygghet blant utsatte pasienter.

En stor fordel som er tiltenkt produktet, er å kunne lese puls på pasienter som sover uten å vekke dem. Søvn av god kvalitet har store fysiske og psykiske meritter, og kan øke livskvaliteten over lange perioder[117]. Tilgangen til en pasients helsedata uten å komme inn i rommet kan også hjelpe helsearbeidere med å effektivisere arbeidsoppgavene sine. De behøver ikke nødvendigvis å gjennomføre manuelle målinger like ofte eller kanskje ikke i det hele tatt.

Fordelene som dette medfører for pasientene kan være mange. Viktigst av alt, burde et slikt pasientarmbånd skape ekstra trygghet for pasientene. For eldre pasienter vil fallsensor for eksempel være en trygget når de skal ut og bevege seg på sykehuset. Eldre kan pådra seg skader dersom de faller. Mange av disse fallene skjer på helt vanlige gåturer[118]. Frykt for å bli liggende uten å få hjelp kan reduseres betraktelig med et pålitelig varslingssystem. E-helse-armbåndet har også en knapp for varsel om hjelp. Tilgangen på en slik knapp gir pasientene mulighet til å be om hjelp selv om de ikke befinner seg i nærheten av den *røde snoren*. I nåværende prototype er det ikke lagt inn noen posisjonsinformasjon, men man vil kunne se hvilken pasient som trenger hjelp.

For noen pasienter kan det oppleves om ubehagelig at deres helsedata blir kontinuerlig loggført og gitt ut direkte til helsepersonellet. Dette er et aspekt som kan føre til at flere vil ta avstand fra produktet, men håpet er at de positive sidene utveier det negative i stor nok grad til at det er ønskelig å ta produktet i bruk.

5.3 Fysisk størrelse

For et produkt som skal sitte på håndleddet til pasienter, er fysisk størrelse noe som bør evalueres. Ønskelig skal Pasientarmbåndet erstatte plastikkarmbåndet som hver pasient får utdelt i løpet av et sykehusbesøk, og om dette skal oppnås må pasientarmbåndet være relativt liten.

Prototypen som gruppen har designet er laget med diverse begrensninger i baktanke. Utstyret gruppen har tilgang på setter noen begrensninger for designet. Det er begrensninger i størrelse på ledningsmønstre og på hvor tett disse ledningsmønstrene kan trekkes. Kretskortene gruppen er i stand til å lage kan heller ikke bestå av mer enn to plan eller sider, noe som fjerner muligheten for et eget dedikert jordplan. Dette gjør at jording og ledningsmønstrene enkelte steder må føres over større områder enn ellers nødvendig. En annen utfordring med utstyret som gruppen har tilgang på, er via-hull mellom plan. Hvert hull trenger at det loddes en leder fast i begge plan for å få kontakt. Konsekvensen blir dermed at via-hullene ikke kan befinne seg under noen komponenter. Dette tvinger også designet til å bli større enn ønskelig. Kapasitet for lodding er også en utfordring som gruppen ble påvirket av. Gruppen fikk hjelp av Tore Myrene Rislåa fra Durapart med lodding. Selv om Tore har stor ekspertise innen lodding, var enkelte av komponentene så små at de allikevel var vanskelige å lodde.

Den ferdige prototypen er 3,840 cm lang og 2,915 cm bred. Dette er en størrelse som er akseptabel for en prototype, men i en reell setting vil dette kunne være for stort. Den største faktoren for enheten sin fysiske størrelse, var mikrokontroller-modulen (ACN52832). Denne modulen har en fysisk størrelse på 2,50 cm × 2,02 cm. Dette var den minste mikrokontrolleren som gruppen var i stand til å få loddet på kortet, men det finnes mindre moduler tilgjengelig. En profesjonell bedrift vil ha bedre muligheter for å bruke en mindre modul i designet. En profesjonell bedrift ville sannsynligvis ha mulighet til å utvikle kretskort med flere plan. Noe som igjen også ville kunne redusert størrelsen til kretskortet.

5.4 Strømforbruk og batteritid

Dersom pasientarmbåndet skal brukes i alle døgnets tider av pasienter på sykehus, er strømforbruk en viktig faktor som må evalueres. Lengre batterilevetid betyr også lengre levetid for enheten. I tillegg blir den mer praktisk i bruk, da den ikke trenger å lades like ofte.

5.4.1.1 Batteritid

Prototypen har et gjennomsnittlig strømforbruk på 3,87 mA når den arbeider for fullt (med Bluetooth og pulsmålinger) og 0,5 mA når den står i dyp søvnmodus. Med et batteri med kapasitet på 70 mAh, vil batteritiden kun være 12 timer når den arbeider for fullt (med Bluetooth og pulsmålinger). Dette er ansett som *kort batteritid* for gjeldende applikasjon. Ideelt sett hadde det vært ønskelig at pasientarmbåndet hadde batteritid i minst én uke. De to mest effektive måtene å øke batterilevetiden på er å bruke et batteri med større kapasitet eller optimalisere fastvaren slik at enheten bruker minst mulig strøm. I tillegg kan kretsen modifiseres til å bruke andre komponenter. Noe som også vil kunne påvirke strømforbruket.

5.4.1.2 Fastvare

Det er ønskelig at mikrokontrolleren jobber så lite som mulig, slik at den trekker minst mulig strøm. Dette gjøres effektivt ved å sette mikrokontrolleren i en søvnmodus som heter *vent på hendelse* eller *Wait For Event (WFE)* så ofte som mulig. I denne modusen trekker mikrokontrolleren minimalt med strøm frem til den vekkes av et avbrudd fra *Nested Vectored Interrupt Controller (NVIC)*'en. Prototypen står i *WFE*-modus i ca. 29 av 32 ms. I de resterende 3 millisekundene jobber den. Målinger fra pulssensoren og akselerometeret gjøres på en frekvens på 32 Hz. Dersom en lavere frekvens hadde blitt brukt, ville nøyaktigheten av målingene blitt redusert. Slik ser vi at det kan være problematisk å begrense strømforbruket ved å sette mikrokontrolleren oftere i søvnmodus. En faktor som trekker store mengder strøm er radio-antennen som brukes til *Bluetooth Low Energy*[51]. I databladet til ACN52832 står det at mikrokontrolleren trekker opptil 9,6 mA når prosessoren og radioantennen jobber samtidig[51]. I motsetning trekker den nedmot 1,2 μ A når enheten venter på avbrudd/hendelser[51]. Noe som potensielt vil kunne spare store mengder strøm, er å bruke antennen sjeldnere og ikke overføre data direkte hele tiden. Kanskje man i stedet ville kunne gjennomført målinger like ofte, men kun overført data til mobil/gateway hvert femte sekund, eller hvert minutt? Så lenge man ikke mister muligheten for å varsle helsepersonell umiddelbart, vil dette med fordel kunne begrense strømforbruket.

5.4.1.3 Effekttap i komponenter

Det kan være flere årsaker til at strømforbruket er høyere enn ønskelig. Det vil for eksempel forekomme effekttap i flere av komponentene. I prototypen ble det tatt i bruk en LDO-regulator for å regulere spenningen fra batteriet på 3,7 V ned til ønsket V_{DD} på 3,3 V. Slike spenningsregulatorer vil alltid ha et effekttap som er vanskelig å unngå. I tillegg vil pulssensoren og akselerometeret sammen med LED-ene stå for en del av strømforbruket. Pulssensoren trekker vanligvis 1.6 mA for å drive begge disse LED-ene[54]. Det er mulig å begrense spenningen som gis til pulssensor-LED-ene, men dette kommer igjen på bekostning av lysstyrke og nøyaktighet i pulsmålingene. Derimot kan LED-indikatoren/RGB LED-en (innebygd i ACN52832-modulen) programmeres til å ikke stå på like ofte som den gjør i den nåværende prototypen.

5.4.1.4 Fallsensor

Det kan hende at løsningen av fallsensoren ikke er optimal. Måten den fungerer er ved å stadig måle akselerometerdata. Dette gjøres på en frekvens på 32 Hz. Verdiene blir deretter sammenliknet med kalkuleringer i fastvarelogikken. Det finnes andre løsninger på markedet i dag, som også sannsynligvis er mer strømeffektive. Blant annet har forskningsinstitusjonen *SINTEF*[42] og teknologileverandøren *Tellu*[28] (nevnt i teoridelen) utviklet en fallsensor som bruker endringer i lufttrykk til å detektere fall[39]. Med en slik type sensor, vil man ikke ha behov for å gjøre målinger med like høy frekvens. Kanskje man kan gjennomføre målinger hvert tiende sekund og fremdeles få en god indikasjon på om en person har falt.

5.4.1.5 Øking av batterikapasitet

Dersom kretskortet hadde blitt redusert i fysisk størrelse, ville man til gjengjeld få mer plass til et batteri. Det kan hende at et 70 mAh timers batteri er for lite for denne applikasjonen. I tillegg vil man potensielt kunne øke batteritiden om det var mulig å utvikle kretsen slik at V_{DD} var 3 V. Det står i databladene til mikrokontrolleren (ACN52832), pulssensoren (BH1970GLC) og akselerometeret (MPU-6050) at de kan drives med forsyningspenning (V_{DD}) på 3 V[51], [54], [61]. Ved å samtidig øke kapasiteten fra 70 mAh til eksempelvis 300 mAh, vil man betraktelig kunne øke batteritiden til pasientarmbåndet. Dessverre var ikke prosjektgruppen i stand til å finne et batteri som innfridde ønskede krav. Det er standard at oppladbare LiPo-batterier leverer 3,7 V. I batteriet til prototypen leverer batteriet opptil 4,2 V fulladet og ned til 3,0 V utladet[119]. Ved å bruke en spenningsregulator som regulerer denne spenningen til 3,3 V eller 3,0 V, vil man derfor uansett få en del effekttap i regulatoren. I tillegg var ikke pulssensorens utviklingskort i stand til å levere gode målinger når 3.0 V forsyningspenning ble brukt. Dette er grunnen til at prototypen bruker 3.3 V som V_{DD} .

5.5 Arbeidsinndeling og tidsbruk.

Prosjektarbeidet hadde tre hovedfokusområder; fastvareutvikling, kretskortutvikling og apputvikling. Arbeidsinndelingen kunne dermed enkelt splittes opp mellom gruppemedlemmene. Aksel tok ansvar for fastvareutvikling, Thomas André tok ansvar for kretskortutvikling og Robin tok ansvar for apputvikling. Se også *Vedlegg D: Prosjektplan*. Gruppen har jobbet sammen for å nå satte målbetingelser. Selv om prosjektet ble delt opp i tre deler, har alle gruppemedlemmene lært masse om alle de ulike prosjektdelene. Gruppen har hatt jevnlig møter og har hatt god kommunikasjon stort sett gjennom hele prosjektperioden. Det var mange elementer som måtte komme på plass på alle de ulike delene, men gruppen føler seg fornøyd med arbeidsmengden som ble tildelt i starten av prosjektet. Stort sett har arbeidsmengden for de ulike delene vært lik.

5.6 Videre arbeid

Appen *Mitt Sykehusopphold* og pasientarmbåndet er ansett til å ha et stort potensial for videre utvikling. Før prosjektet kan bli utviklet til et fullverdig produkt, er det nødvendig å legge til flere funksjoner, samt øke kvaliteten på målingene. Fysisk størrelse og batterilevetid er også punkter som trenger optimalisering.

5.6.1.1 Andre typer målinger

Pasientarmbåndet kan med fordel overvåke mer enn bare pulsmåling og falldeteksjon. På markedet finnes det komponenter som kan gjennomføre f.eks. blodtrykkmåling og målinger av oksygen i blodet (oksymetri). Dersom det er mulig å gjennomføre målinger innenfor en akseptabel feilmargin, vil disse kunne være nyttig for et fremtidig produkt. Det stilles høye krav til nøyaktighet av målinger som gjennomføres av pasienter på sykehus. Det er sannsynligvis en grunn til at fitnessarmbånd foreløpig ikke brukes til å overvåke pasienter på sykehus. Allikevel ser vi eksempler på at fitnessarmbånd testes ut innen helsesektoren i andre land[120]. Grunnen til det høye kravet av nøyaktighet har med pasientenes sikkerhet å gjøre. Å utvikle et produkt som er nøyaktig nok og samtidig overvåker mange verdier, vil være en stor utfordring.

Målinger som potensielt kan implementeres i et fremtidig produkt:

- Temperaturmåling.
- Blodtrykk.
- Oksygenmetning i blodet (oksymetri).
- EKG (Apple Watch klarer nå å gi EKG-måling[121]).

5.6.1.2 Pasientarmbåndet

Fysisk størrelse er et utrolig viktig punkt for videre design. Med redusert størrelse på kretskortet, kan mer plass gis til et større batteri med mer kapasitet. Et mindre produkt med bedre passform gjør også armbåndet mer komfortabelt å bruke over lengre tid. Det tenkes at trådløs ladning kan implementeres på pasientarmbåndet, dersom dette gjør det enklere å lade produktet[122]. Trådløs ladning kan helt klart implementeres på pasientarmbåndet[122]. Spørsmålet er om dette gjør det enklere for pasienter å lade armbåndet.

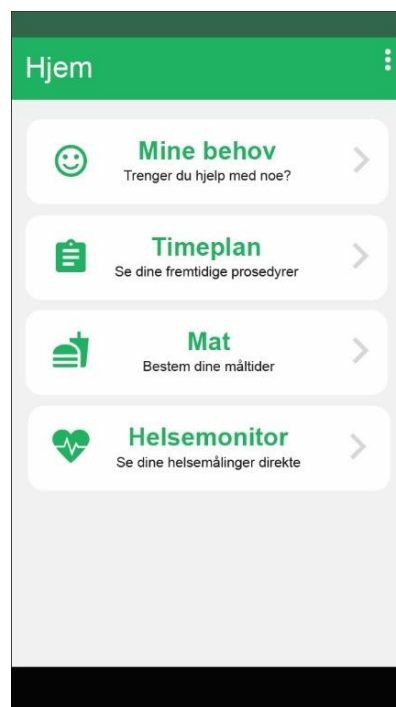
Potensielle forbedringer med den fysiske enheten:

- Økt batterikapasitet og batteritid.
- Utforming som et armbånd med god passform og liten størrelsesfaktor.
- Trådløs ladning.

- RFID (slik at armbåndet kan brukes som ID-tag).

5.6.1.3 Mobilapplikasjonen

Potensiale til videreutvikling av mobilapplikasjonen er ganske stort. MSO-appen er foretar seg kun en liten porsjon av det fullstendige konseptet til *Mitt Sykehusopphold*. I tillegg var det enkelte mål som ikke kom på plass i det ferdige produktet. Blant annet var et mål å gi en indikasjon på om måleverdier ser normale ut. I tillegg var det planlagt å gi helsepersonell mulighet til å legge inn pasientnotater. Disse to elementene ville blitt prioritert dersom applikasjonen skulle ha blitt videreutviklet. Mobilapplikasjonen bør utvikles for flere plattformer. Hovedsakelig bør den også utvikles for *Apple iOS* da det kan antas at mellom 20-40% av den norske befolkningen bruker dette operativsystemet[85], [88]. I tillegg er det mange flere funksjoner fra *Mitt Sykehusopphold*-konseptet som kan tas videre. Blant annet kan det implementeres en oversikt/timeplan over kommende prosedyrer. Det kan legges til dedikert side/aktivitet for *behover*, hvor man blant annet kan tilkalle hjelp fra helsepersonellet. Det er også tenkt å gi pasienter muligheter til å planlegge måltider. Se *Figur 56* for en skisse av hvordan pasientenes hovedside vil kunne se ut med tilføyd funksjonalitet.



Figur 56: Appskisse av pasientenes hovedside i «Mitt Sykehusopphold».

Fremtidige forbedringer for mobilapplikasjonen:

- Indikasjon på om måleverdier ser normale ut.
- Helsepersonell bør kunne legge inn pasientnotater.
- Utvikling av appen for andre plattformer, slik som *iOS*.

- Videreutvikling av konseptet *Mitt Sykehusopphold*.
 - Oversikt over fremtidige prosedyrer, gjøremål og medisinerer.
 - Egen side hvor man kan sende forespørsler/behov til helsepersonellet.
 - Planlegging av måltider i appen.

5.6.1.4 Bruk av markedsstandarder

Siste skrittet ville blitt å integrere produktet med markedsstandarder. Ved å integrere *Elektronisk pasientjournal (EPJ)* levert av *Distribuert Informasjons og Pasientdatasystem (DIPS)*[25] vil man gi pasienter tilgang til sin personlige journal rett fra appen. I tillegg bør appen samarbeide med *MetaVision Elektronisk kurveprosjekt* levert av *EVERY*[26]. Dette vil gi pasienter innsyn i sin medisinske kurve. Dersom appen skal videreutvikles, er det også viktig å ta i bruk *HL7 FHIR*[7]. Dette er en global standard for formatering av klinisk data. Alarmsystemet bør også integreres med *SCAIP*[6]. Det ville også vært hensiktsmessig å samarbeide med helsearbeidere for å utvikle en applikasjon som *de* kan være fornøyde med å bruke i sin arbeidssituasjon.

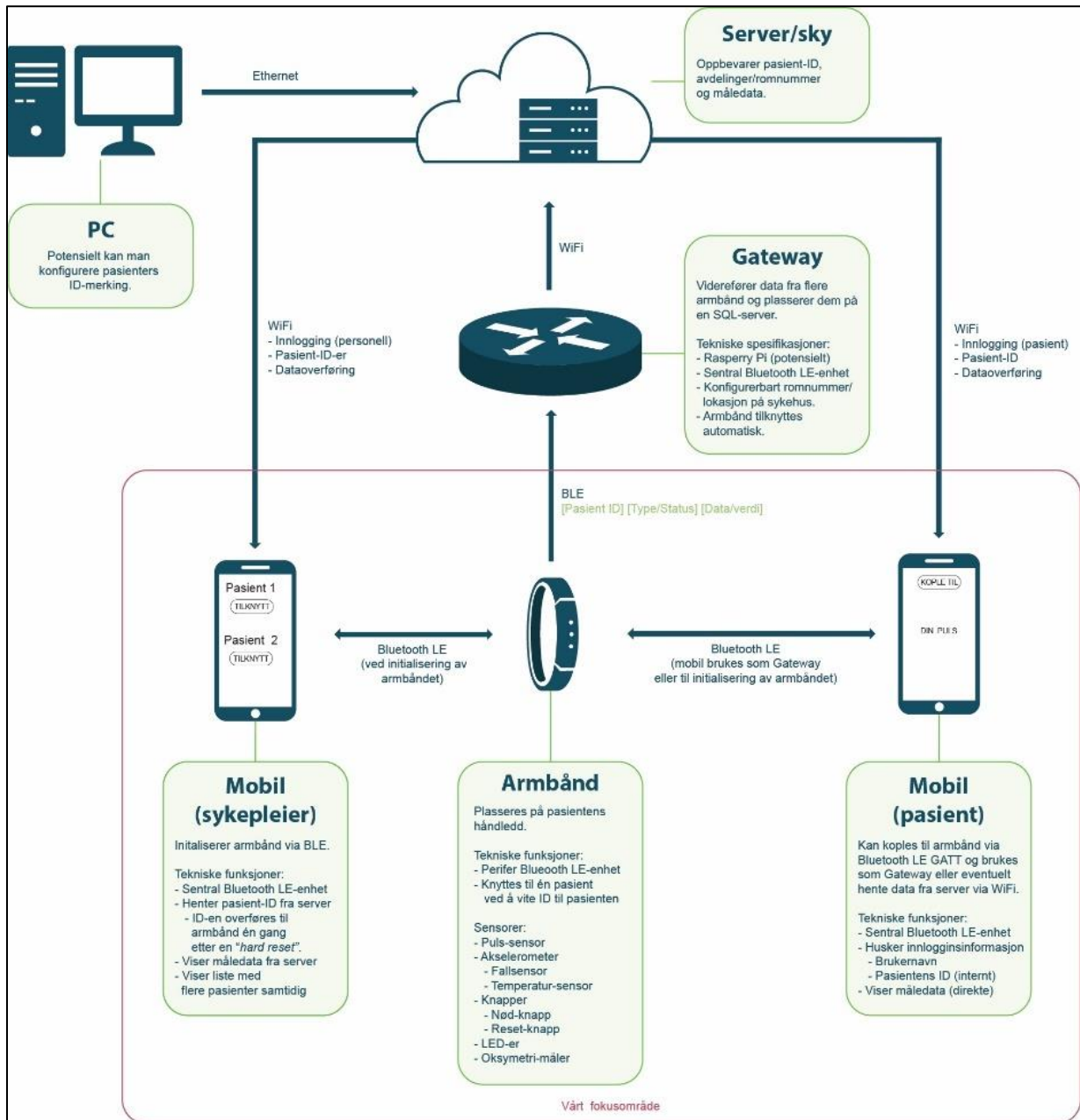
Relevante markedsstandarder som bør integreres:

- DIPS-systemet og EPJ.
- MetaVision/Elektronisk kurveprosjekt.
- HL7 FHIR-standarden.
- SCAIP-protokollen for alarmsystemer.

5.6.1.5 Kommunikasjonsprotokoll

I løpet av prosjektperioden diskuterte prosjektgruppen hva som ville vært et ideelt kommunikasjonsoppsett i en reell setting. I starten av prosjektet var det planlagt at både pasient og helsepersonell var tilkoblet pasientarmbåndet samtidig via Bluetooth og at helsepersonell ville kunne være tilkoblet flere armbånd samtidig. Dette viser seg å være problematisk da en slave (også kalt periferenhet eller server) innen Bluetooth Low Energy kun kan være del av et piconet (Bluetooth-nettverk) om gangen. En slave kan kun ha en master. Derfor kan ikke armbåndet, som her er slave, koble seg til flere smartelefoner samtidig. I prosjektet ble det gjort mulig å lese data fra flere armbånd samtidig ved å bruke MQTT. Pasient er tilknyttet pasientarmbånd via Bluetooth og videresender data til andre mobiltelefoner via en MQTT-tjeneste. Det var tenkt at en liknende løsning ville kunne implementeres dersom konseptet videreutvikles, men at hvert rom på sykehuset stasjoneres med en dedikert inngangsport/*gateway*. For eksempel kan en *Raspberry Pi* benyttes som *gateway*[123]. Denne *gateway*'en vil da kunne videresende data fra alle armbånd lokalisert i rommet til en dataservert (skylagring). Se *Figur 57* for en illustrasjon av hvordan et slikt system kan se ut. Det er tenkt at armbåndet tilknyttes pasienten enten ved at pasienten selv kobler seg til armbåndet via Bluetooth én gang, eller at helsepersonell gjør denne tilknytningen. Dette må gjøres slik at dataen fra armbåndet tilknyttes riktig pasient-ID. Deretter kobles mobilen fra

armbåndet. Armbåndet kobler seg deretter automatisk til gatewayen og overfører måledata. Måledataen kan så hentes fra dataserveren ned til mobilappen. I tillegg til dette systemet må et varslingsystem implementeres. Her kan for eksempel MQTT brukes.



Figur 57: Ideelt konsept for kommunikasjonssystem.

6 Konklusjon

I dette kapitlet konkluderes resultatene fra prosjektet. Blant annet kommer konklusjonen for problemstillingene som ble satt i problemdefinisjonen i innledningen. Hovedproblemstillingen for prosjektet var:

6.1 Hovedproblemstilling

Å utvikle et e-helse-armbånd som aktivt overvåker helsedata og viser denne dataen direkte i en dedikert mobilapplikasjon.

I løpet av denne oppgaven har hovedproblemstillingen blitt løst. Gruppen har klart å utvikle en fungerende prototype for et e-helse-armbånd som er i stand til å overvåke helsedata. Den utviklede prototypen er i stand til å lese informasjon fra en pulssensor og beregner puls med en feilmargin mindre enn $\pm 10\%$. Deretter sender den resultatet videre til en mobilapplikasjon via *Bluetooth Low Energy* slik at dataen kan vises direkte i appen. Prototypen leser også informasjon fra et akselerometer. Dersom akselerometerdataen som mottas oppfyller gitte krav (basert på fastvarelogikk) vil enheten sende ut et varsel til smarttelefon om at pasienten har falt.

6.2 Sekundær problemstilling

Vil kontinuerlig overvåking av pasienters helseverdier via et pasientarmbånd bedre pasientopplevelsen eller effektivisere arbeidsprosesser blant ansatte?

6.2.1 Hovedkonklusjon

Hovedkonklusjonen er at å bruke et pasientarmbånd med direkteoverføring av data til mobiltelefon kan være fordelaktig i enkelte sammenhenger. Det vil kunne gjøre det enklere for helsepersonell å overvåke helsetilstanden til pasienter og gjøre arbeidsflyten bedre. Eksempelvis kan et slikt pasientarmbånd potensielt redusere antall unødvendige besøk inn til pasientene.

6.2.2 Puls alene er ikke nok

Det å måle puls alene vil ikke kunne gi en god indikasjon på om noe er galt med en pasient. Det vil kun gi en indikasjon på om personen har normal eller unormal puls (i ekstreme situasjoner). Mange faktorer kan påvirke puls, og de er derfor vanskelig å diagnostisere pasienten på bakgrunn av puls alene. For å

gjøre sykehusoppholdet tryggere med et pasientarmbånd, må mer funksjonalitet tilføyes. Blant annet vil det være aktuelt å overvåke temperatur, oksygenmetning, EKG og blodtrykk. Ved å implementere en kunstig intelligens (AI) eller ved å bruke maskinlæring (ML), vil det kunne utvikles dataprogrammer som lærer seg pasientens normale helsetilstand. Et slikt program vil etter hvert kunne gi et varsel på om noe er galt med en pasient. Konklusjonen er at puls alene ikke er godt nok for å gjøre et pasientarmbånd hjelpsomt.

6.2.3 Trygghet

Videre konkluderes det med at det kan føles betryggende for enkelte pasienter å ha tilgang til sine egne helsedata på sykehus. Å få eierskap til sin egen helsetilstand kan gjøre oppholdet mer behagelig og beroligende, da pasienten selv kan få en generell indikasjon på om helsedataene sine befinner seg innenfor normale områder. På den andre siden konkluderer vi også med at enkelte pasienter kan finne den kontinuerlige overvåkningen ubehagelig. Det må tas hensyn til at alle mennesker er individer. En etisk evaluering av kontinuerlig overvåkning på sykehus kan dermed være svært aktuelt for konseptet.

6.3 Prosjektets måloppnåelse

Det konkluderes med at produktet oppfyller mesteparten av kravene som ble satt for designet. Både prototypen og applikasjonen har alle de mest sentrale funksjonalitetene på plass.

6.3.1 Fastvare

Konklusjonen for fastvare-programmeringen er at den behandler sensor-data på tilfredsstillende måte og videresender dette til mobilapplikasjonen på en god måte. Beregningene som blir gjort, leverer stabile og fine pulsmålinger som kan ha nytteverdi. Testene som gruppen har gjort tilsier at prototypen har en feilmargin på mindre enn $\pm 10\%$ av tilsvarende produkter på markedet. Videre er det implementert fallsensor som utløser en alarm ved store bevegelser. Denne funksjonaliteten kan utløses selv om ikke fall forekommer, noe som tyder på at det fremdeles er forbedringspotensial her. For strømsparing, er mikrokontrolleren programmert til å stå i WFE-modus frem til et avbrudd kommer.

6.3.2 Prototype-kretskort (fysisk enhet)

Gruppen konkluderer med at prototype-kretskortet oppfyller kravene som ble satt i forkant. Prototypen kommuniserer med både pulssensor og akselerometer. Den fysiske enheten har tre taktile knapper og mikrokontrolleren er utstyrt med en RGB-LED som brukes som indikasjonslys. Armbåndet har tilhørende oppladbart batteri (3,7 V og 70 mAh). Hele prototypen står på egne ben når batteriet er koblet til. Allikevel er batteritiden noe i det minste laget (>12 timer). Et større batteri og mer fokus på strømbesparing er svært aktuelt. Den fysiske størrelsen (3,840 cm \times 2,915 cm) er godt innenfor kravene som ble satt.

6.3.3 Mobilapplikasjon

Mobilapplikasjonen innfrir de sentrale kravene som ble satt på forhånd av oppgaven. Det ble implementert kommunikasjon med armbåndet via *Bluetooth Low Energy (BLE)*. Dataen fra armbåndet vises direkte i appen. Dersom man er helsepersonell kan man også lese data fra flere armbånd samtidig (så lenge pasienter er innlogget). Noen av målene ble ikke innfridd, men anses ikke som vanskelige å implementere senere. Blant annet blir det ikke gitt noen indikasjon på om måleverdier er normale i den nåværende versjonen av MSO-appen. Det er heller ikke mulig for helsepersonell å legge inn pasientnotater. Disse to funksjonene ble nedprioritert for å få på plass en god utveksling av data via *BLE*, samt overføring til flere smarttelefoner via *MQTT*.

6.4 Videre arbeid

Det konkluderes med at konseptet har en del arbeid som gjenstår før den kan implementeres på sykehus. Blant annet må et ordentlig kommunikasjonssystem bli designet (se *Figur 57*). Systemet må også integreres med nåværende markedsstandarder, slik som HL7 FHIR, SCAIP, DIPS, EPJ og MetaVision.

Referanser

- [1] Helse-og omsorgsdepartementet, «Helseteknologi for bedre og tryggere tjenester», *Regjeringen.no*, 03-feb-2014. [Online]. Tilgjengelig på: <https://www.regjeringen.no/no/aktuelt/helseteknologi-for-bedre-og-tryggere-tje/id750336/>. [Åpnet: 08-mai-2019].
- [2] H. omsorgsdepartementet, «E-helse og IKT i helsesektoren», *Regjeringen.no*, 14-jan-2016. [Online]. Tilgjengelig på: <https://www.regjeringen.no/no/tema/helse-og-omsorg/innsikt/e-helse-og-ikt-i-helsesektoren/id2356319/>. [Åpnet: 08-mai-2019].
- [3] Den Norske Legeforening *mfl.*, «Statusrapport 2014 Sykehus for fremtiden Innspill til nasjonal sykehusplan», apr-2014. [Online]. Tilgjengelig på: <https://legeforeningen.no/PageFiles/184482/Statusrapport%20-%20Sykehus%20for%20fremtiden.pdf>. [Åpnet: 01-apr-2019].
- [4] Sørlandet Sykehus, «Fremtidens sykehusteknologi og dens anvendelse», Høsten-2012. [Online]. Tilgjengelig på: <https://docplayer.me/1599330-Fremtidens-sykehusteknologi-og-dens-anvendelse.html>. [Åpnet: 22-apr-2019].
- [5] S. Radcliffe, «What Will Your Hospital Look Like in 5 Years?», *Healthline*, 15-feb-2018. [Online]. Tilgjengelig på: <https://www.healthline.com/health-news/future-of-hospitals-in-five-years#2>. [Åpnet: 01-apr-2019].
- [6] SIS, Swedish Standards Institute, «Digital social alarm - Social care alarm internet protocol (SCAIP) - Specification», *Swedish Standards Institute*, 2014. [Online]. Tilgjengelig på: <https://www.sis.se/en/produkter/health-care-technology/aids-for-disabled-or-handicapped-persons/general/ss911002014/>. [Åpnet: 25-apr-2019].
- [7] Health Level Seven International, «HL7 FHIR® R4», *HL7 International*. [Online]. Tilgjengelig på: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=491. [Åpnet: 25-apr-2019].
- [8] Wikipedia, «SEGGER Embedded Studio». [Online]. Tilgjengelig på: https://wiki.segger.com/SEGGER_Embedded_Studio. [Åpnet: 02-mai-2019].
- [9] Altium, «Altium Designer 19 - Easy, Modern and Powerful PCB Design Software». [Online]. Tilgjengelig på: <https://www.altium.com/altium-designer/>. [Åpnet: 08-mai-2019].
- [10] Wikipedia, «Altium Designer», *Wikipedia*, 17-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Altium_Designer&oldid=892941097. [Åpnet: 08-mai-2019].
- [11] Wikipedia, «Microsoft Visual Studio», *Wikipedia*, 18-jul-2018. [Online]. Tilgjengelig på: https://no.wikipedia.org/w/index.php?title=Microsoft_Visual_Studio&oldid=18702251. [Åpnet: 08-mai-2019].
- [12] Microsoft, «Xamarin.Forms». [Online]. Tilgjengelig på: <https://docs.microsoft.com/en-us/xamarin/xamarin-forms/>. [Åpnet: 03-mai-2019].
- [13] Wikipedia, «Android Studio», *Wikipedia*, 19-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Android_Studio&oldid=893154780. [Åpnet: 25-apr-2019].
- [14] Wikipedia, «Java (programming language)», *Wikipedia*, 06-mai-2019. [Online]. Tilgjengelig på: [https://en.wikipedia.org/w/index.php?title=Java_\(programming_language\)&oldid=895844807](https://en.wikipedia.org/w/index.php?title=Java_(programming_language)&oldid=895844807). [Åpnet: 08-mai-2019].
- [15] Wikipedia, «YouTube», *Wikipedia*, 07-mai-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=YouTube&oldid=895914717>. [Åpnet: 08-mai-2019].
- [16] conceptdev, «Xamarin Documentation». [Online]. Tilgjengelig på: <https://docs.microsoft.com/en-us/xamarin/>. [Åpnet: 08-mai-2019].
- [17] Android Developers, «Developer Guides | Android Developers». [Online]. Tilgjengelig på: <https://developer.android.com/guide>. [Åpnet: 08-mai-2019].
- [18] Quora, Rakshit G.L, «What is SoloLearn?» [Online]. Tilgjengelig på: <https://www.quora.com/What-is-SoloLearn>. [Åpnet: 08-mai-2019].

- [19] Nordic Semiconductor, «A brief overview of our wireless product portfolio». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Products>. [Åpnet: 08-mai-2019].
- [20] Nordic Semiconductor, «3rd party modules». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Software%20and%20Tools/3rd%20Party/3rd%20party%20modules>. [Åpnet: 08-mai-2019].
- [21] Wikipedia, «Digi-Key», *Wikipedia*, 28-mar-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Digi-Key&oldid=889913626>. [Åpnet: 09-mai-2019].
- [22] Wikipedia, «Mouser Electronics», *Wikipedia*, 30-jan-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Mouser_Electronics&oldid=881003158. [Åpnet: 09-mai-2019].
- [23] Wikipedia, «Premier Farnell», *Wikipedia*, 11-jan-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Premier_Farnell&oldid=877871870. [Åpnet: 09-mai-2019].
- [24] Direktoratet for e-helse, «Anbefaling om bruk av HL7 FHIR for datadeling», mar-2019. [Online]. Tilgjengelig på: <https://ehelse.no/Documents/Rapporter%20og%20dokumenter%20standardisering/Anbefaling%20om%20bruk%20av%20HL7%20FHIR%20for%20datadeling.pdf>. [Åpnet: 03-mai-2019].
- [25] Wikipedia, «DIPS», *Wikipedia*, 16-mar-2019. [Online]. Tilgjengelig på: <https://no.wikipedia.org/w/index.php?title=DIPS&oldid=19268771>. [Åpnet: 26-apr-2019].
- [26] T. H. Eide og H. M. Sporse, «MetaVision, elektronisk kurve: Et verktøy i forbedring av pasientsikkerhet?», *Farmatid*, 22-feb-2010. [Online]. Tilgjengelig på: <https://www.farmatid.no/artikler/metavision-elektronisk-kurve-et-verktoy-forbedring-av-pasientsikkerhet>. [Åpnet: 26-apr-2019].
- [27] Helse Nord, «Elektronisk kurve og medikasjon», *Helse Nord RHF*, desember-2017. [Online]. Tilgjengelig på: <https://helse-nord.no/helse-nord-fiks-n-journal-i-nord/elektronisk-kurve-og-medikasjon->. [Åpnet: 03-mai-2019].
- [28] TelluCloud, «TelluCloud», *TelluCloud*. [Online]. Tilgjengelig på: <https://www.tellucloud.com/tellucloud/>. [Åpnet: 26-apr-2019].
- [29] Wikipedia, «Fast Healthcare Interoperability Resources», *Wikipedia*, 03-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Fast_Healthcare_Interoperability_Resources&oldid=895287372. [Åpnet: 03-mai-2019].
- [30] Wikipedia, «Activity tracker», *Wikipedia*, 14-mar-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Activity_tracker&oldid=887676925. [Åpnet: 13-mai-2019].
- [31] C. Allison, «Fitbit heart rate monitoring explained», *Wearable*, 10-mai-2019. [Online]. Tilgjengelig på: <https://www.wearable.com/fitbit/fitbit-heart-rate-monitor-guide-330>. [Åpnet: 13-mai-2019].
- [32] Apple INC, «Check your heart rate on Apple Watch», *Apple Support*. [Online]. Tilgjengelig på: <https://support.apple.com/guide/watch/heart-rate-apda88aefe4c/watchos>. [Åpnet: 13-mai-2019].
- [33] K. Østvang, «Nå kan man ta EKG-måling med Apple Watch i Norge», *DinSide.no*, 27-mar-2019. [Online]. Tilgjengelig på: <http://www.dinside.no/a/70910421>. [Åpnet: 13-mai-2019].
- [34] Wikipedia, «Pulse oximetry», *Wikipedia*, 10-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Pulse_oximetry&oldid=896381623. [Åpnet: 13-mai-2019].
- [35] S. Sheps, «Wrist blood pressure monitors: Are they accurate?», *Mayo Clinic*. [Online]. Tilgjengelig på: <https://www.mayoclinic.org/diseases-conditions/high-blood-pressure/expert-answers/wrist-blood-pressure-monitors/faq-20057802>. [Åpnet: 13-mai-2019].
- [36] F. El-Amrawy og M. I. Nounou, «Are Currently Available Wearable Devices for Activity Tracking and Heart Rate Monitoring Accurate, Precise, and Medically Beneficial?», *Healthc. Inform. Res.*, bd. 21, nr. 4, s. 315–320, okt. 2015.
- [37] LifeSpan, «Your Resting Heart Rate: What Is Normal and Healthy?» [Online]. Tilgjengelig på: <https://www.lifespanfitness.com/fitness/resources/articles/your-resting-heart-rate-what-is-normal-and-healthy>. [Åpnet: 13-mai-2019].
- [38] Wikipedia, «Heart rate», *Wikipedia*, 18-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Heart_rate&oldid=893024002. [Åpnet: 25-apr-2019].

- [39] SINTEF, «Ny sensor vil øke tryggheten for eldre», *SINTEF*. [Online]. Tilgjengelig på: <http://www.sintef.no/siste-nytt/ny-sensor-vil-oke-tryggheten-for-eldre/>. [Åpnet: 13-mai-2019].
- [40] Tryggereliv, «Mobil trygghetsalarm m/GPS, 2-veis tale og fallsensor». .
- [41] Doro Care Norge, «Fallsensor». [Online]. Tilgjengelig på: <https://care.doro.no/produkter/fallsensor/>. [Åpnet: 13-mai-2019].
- [42] SINTEF, «Teknologi for et bedre samfunn». [Online]. Tilgjengelig på: <https://www.sintef.no/>. [Åpnet: 13-mai-2019].
- [43] Wikipedia, «Nordic Semiconductor», *Wikipedia*, 17-apr-2019. [Online]. Tilgjengelig på: https://no.wikipedia.org/w/index.php?title=Nordic_Semiconductor&oldid=19375967. [Åpnet: 12-mai-2019].
- [44] Nordic Semiconductor, «Low power short-range wireless products». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Products/Low%20power%20short-range%20wireless>. [Åpnet: 12-mai-2019].
- [45] Nordic Semiconductor, «nRF52832: Flexible, efficient Bluetooth 5 and Bluetooth mesh multiprotocol SoC». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Products/Low%20power%20short-range%20wireless/nRF52832>. [Åpnet: 13-mai-2019].
- [46] Nordic Semiconductor, «IDEs and Toolchains». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Software%20and%20Tools/Development%20Tools/IDEs%20and%20Toolchains>. [Åpnet: 12-mai-2019].
- [47] Nordic Semiconductor, «nRF52 DK: Bluetooth 5 and Bluetooth mesh Development Kit for nRF52810 and nRF52832 SoCs». [Online]. Tilgjengelig på: <https://www.nordicsemi.com/en/Software%20and%20Tools/Development%20Kits/nRF52%20DK>. [Åpnet: 12-mai-2019].
- [48] ARM, «Cortex-M4 Devices Generic User Guide», s. 277.
- [49] Mouser Electronics, «ACN52832 aconno», *Mouser Electronics*. [Online]. Tilgjengelig på: <https://no.mouser.com/ProductDetail/364-ACN52832>. [Åpnet: 14-mai-2019].
- [50] Mouser Electronics, «ISP1507-AX-RS Insight SiP». [Online]. Tilgjengelig på: <https://no.mouser.com/ProductDetail/Insight-SiP/ISP1507-AX-RS?qs=sGAEpiMZZMve4%2FbfQkoj%252bcQV2rmt7cIkWnZEKZXWtGc%3D>. [Åpnet: 14-mai-2019].
- [51] aconno GmbH, «Datasheet ACN52832 Fully integrated, ultra-low power, Bluetooth Smart module», 11-aug-2017. [Online]. Tilgjengelig på: https://1897079276.rsc.cdn77.org/wp-content/uploads/2018/08/ACN52832_data_sheet_v1.3.pdf. [Åpnet: 08-mai-2019].
- [52] Insight SIP, «ISP1507 Miniature Bluetooth Low Energy (BLE) Module». [Online]. Tilgjengelig på: <https://www.insightsip.com/products/bluetooth-smart-modules/isp1507>. [Åpnet: 13-mai-2019].
- [53] Wikipedia, «Rohm», *Wikipedia*, 27-mar-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Rohm&oldid=889765300>. [Åpnet: 14-mai-2019].
- [54] Rohm Semiconductor, «Optical Sensor for Heart Rate Monitor IC - BH1790GLC», 01-feb-2017. [Online]. Tilgjengelig på: <https://no.mouser.com/datasheet/2/348/bh1790glc-e-1139225.pdf>. [Åpnet: 13-mai-2019].
- [55] Maxim Integrated, «MAX30102 High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health». [Online]. Tilgjengelig på: <https://www.maximintegrated.com/en/products/sensors/MAX30102.html>. [Åpnet: 14-mai-2019].
- [56] Biostrap, «Red vs. Green: Does the Light Sensor in Your Wearable Matter?» [Online]. Tilgjengelig på: <https://blog.biostrap.com/posts/going-red-or-green>. [Åpnet: 14-mai-2019].
- [57] Wikipedia, «I²C», *Wikipedia*, 06-mai-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=I%2C2%B2C&oldid=895709225>. [Åpnet: 14-mai-2019].
- [58] S. A. Evensen, «hemoglobin», *Store medisinske leksikon*, 23-jan-2019. [Online]. Tilgjengelig på: <http://sml.snl.no/hemoglobin>. [Åpnet: 14-mai-2019].
- [59] Digikey, «BH1790GLC Optical Sensor - ROHM Semiconductor». [Online]. Tilgjengelig på: <https://www.digikey.com/en/product-highlight/r/rohm-semi/bh1790glc-optical-sensor-for-heart-rate-monitoring>. [Åpnet: 12-mai-2019].

- [60] TDK-InvenSense, «MPU-6050». [Online]. Tilgjengelig på: <https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/>. [Åpnet: 12-mai-2019].
- [61] TDK-InvenSense, «MPU-6000 and MPU-6050 Product Specification Revision 3.4». [Online]. Tilgjengelig på: <https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf>. [Åpnet: 13-mai-2019].
- [62] TDK-InvenSense, «MPU-6500 Register Map and Descriptions Revision 2.1». [Online]. Tilgjengelig på: <https://www.invensense.com/wp-content/uploads/2015/02/MPU-6500-Register-Map2.pdf>. [Åpnet: 14-mai-2019].
- [63] Wikipedia, «Bluetooth», *Wikipedia*, 07-mai-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Bluetooth&oldid=895876347>. [Åpnet: 08-mai-2019].
- [64] Wikipedia, «Frequency-shift keying», *Wikipedia*, 03-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Frequency-shift_keying&oldid=895254683. [Åpnet: 08-mai-2019].
- [65] Wikipedia, «ISM-bånd», *Wikipedia*, 17-mar-2016. [Online]. Tilgjengelig på: <https://no.wikipedia.org/w/index.php?title=ISM-b%C3%A5nd&oldid=15992736>. [Åpnet: 08-mai-2019].
- [66] Wikipedia, «Wi-Fi», *Wikipedia*, 06-mai-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Wi-Fi&oldid=895762337>. [Åpnet: 08-mai-2019].
- [67] «Adaptative Frequency Hopping». [Online]. Tilgjengelig på: <https://sites.google.com/site/nearcommunications/adaptative-frequency-hopping>. [Åpnet: 08-mai-2019].
- [68] Bluetooth SIG, «Solution Areas», *Bluetooth Technology Website*. [Online]. Tilgjengelig på: <https://www.bluetooth.com/bluetooth-technology/solutions/>. [Åpnet: 08-mai-2019].
- [69] Wikipedia, «Bluetooth Low Energy», *Wikipedia*, 06-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Bluetooth_Low_Energy&oldid=895821514. [Åpnet: 08-mai-2019].
- [70] Bluetooth SIG, «Bluetooth Low Energy», 10-mar-2017. [Online]. Tilgjengelig på: <https://web.archive.org/web/20170310111443/https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/low-energy>. [Åpnet: 08-mai-2019].
- [71] EverythingRF, «What is the difference between Bluetooth 5, Bluetooth 4.2 and Bluetooth 2.0?» [Online]. Tilgjengelig på: <https://www.everythingrf.com/community/what-is-the-difference-between-bluetooth-5-0-bluetooth-low-energy-bluetooth-v4-2-and-classic-bluetooth>. [Åpnet: 08-mai-2019].
- [72] R. Kumar, «The Digital Knight: What is difference between Piconet and Scatternet?», *The Digital Knight*, 23-okt-2016. [Online]. Tilgjengelig på: <http://rkspidey.blogspot.com/2016/10/what-is-difference-between-piconet-and.html>. [Åpnet: 09-mai-2019].
- [73] Wikipedia, «Piconet», *Wikipedia*, 27-mar-2017. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Piconet&oldid=772479120>. [Åpnet: 08-mai-2019].
- [74] Wikipedia, «Wireless ad hoc network», *Wikipedia*, 13-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Wireless_ad_hoc_network&oldid=892309772. [Åpnet: 08-mai-2019].
- [75] Wikipedia, «Scatternet», *Wikipedia*, 17-mai-2018. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=Scatternet&oldid=841749845>. [Åpnet: 09-mai-2019].
- [76] Nordic Semiconductor DevZone, Ole Morten, «What is a client and server in BLE?», *Nordic DevZone*. [Online]. Tilgjengelig på: <https://devzone.nordicsemi.com/f/nordic-q-a/71/what-is-a-client-and-server-in-ble>. [Åpnet: 08-mai-2019].
- [77] Wikipedia, «Universal asynchronous receiver-transmitter», *Wikipedia*, 06-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Universal_asynchronous_receiver-transmitter&oldid=895709636. [Åpnet: 08-mai-2019].
- [78] MQTT, «MQTT». [Online]. Tilgjengelig på: <http://mqtt.org/>. [Åpnet: 08-mai-2019].
- [79] Wikipedia, «MQTT», *Wikipedia*, 27-apr-2019. [Online]. Tilgjengelig på: <https://en.wikipedia.org/w/index.php?title=MQTT&oldid=894377639>. [Åpnet: 08-mai-2019].
- [80] MQTT, «FAQ - Frequently Asked Questions». .
- [81] digicert, «What Is SSL (Secure Sockets Layer)?», *DigiCert*. [Online]. Tilgjengelig på: <https://www.digicert.com/ssl/>. [Åpnet: 08-mai-2019].

- [82] Wikipedia, «Message broker», *Wikipedia*, 16-feb-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Message_broker&oldid=883665804. [Åpnet: 08-mai-2019].
- [83] CloudMQTT, «Plans & Pricing». [Online]. Tilgjengelig på: <https://www.cloudmqtt.com/plans.html>. [Åpnet: 08-mai-2019].
- [84] Mosquitto, «Eclipse Mosquitto: An open source MQTT broker», *Eclipse Mosquitto*, 08-jan-2018. [Online]. Tilgjengelig på: <https://mosquitto.org/>. [Åpnet: 09-mai-2019].
- [85] Kantar, «Android vs. iOS Smartphone OS sales market share evolution», *kantarworldpanel.com*, mar-2019. [Online]. Tilgjengelig på: <https://www.kantarworldpanel.com/smartphone-os-market-share/>. [Åpnet: 25-apr-2019].
- [86] C. Reilly, «Windows 10 Mobile gets its final death sentence», *CNET*, oktober-2017. [Online]. Tilgjengelig på: <https://www.cnet.com/news/windows-10-mobile-features-hardware-death-sentence-microsoft/>. [Åpnet: 25-apr-2019].
- [87] Wikipedia, «Windows Phone», *Wikipedia*, 23-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Windows_Phone&oldid=893792102. [Åpnet: 25-apr-2019].
- [88] Statistisk Sentralbyrå, «Fakta om Internett og mobil», *ssb.no*. [Online]. Tilgjengelig på: <https://www.ssb.no/teknologi-og-innovasjon/faktaside/internett-og-mobil>. [Åpnet: 25-apr-2019].
- [89] TheZachBales, «How to Create an Android App With Android Studio», *Instructables*. [Online]. Tilgjengelig på: <https://www.instructables.com/id/How-To-Create-An-Android-App-With-Android-Studio/>. [Åpnet: 08-mai-2019].
- [90] Android Developers, «Services overview», *Android Developers*. [Online]. Tilgjengelig på: <https://developer.android.com/guide/components/services>. [Åpnet: 08-mai-2019].
- [91] Android Developers, «BroadcastReceiver», *Android Developers*. [Online]. Tilgjengelig på: <https://developer.android.com/reference/android/content/BroadcastReceiver>. [Åpnet: 09-mai-2019].
- [92] L. KAPI, «How to Use getExtra and putExtra in Android for String Data». [Online]. Tilgjengelig på: <http://leylaKapi.github.io/blog/2014/11/08/how-to-use-getextra-and-putextra-in-android-for-string-data/>. [Åpnet: 09-mai-2019].
- [93] Wikipedia, «XML», *Wikipedia*, 15-okt-2018. [Online]. Tilgjengelig på: <https://no.wikipedia.org/w/index.php?title=XML&oldid=18902030>. [Åpnet: 08-mai-2019].
- [94] Google Developers og Android Developers, «Toasts overview (Android Documentation)», *Android Developers*. [Online]. Tilgjengelig på: <https://developer.android.com/guide/topics/ui/notifiers/toasts>. [Åpnet: 09-mai-2019].
- [95] GitHub, phillips77, *Android-BLE-Library*. Nordic Semiconductor, 2019.
- [96] Wikipedia, «Android version history», *Wikipedia*, 25-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Android_version_history&oldid=894034430. [Åpnet: 25-apr-2019].
- [97] P. B. Andersen og I. M. Liseter, «kretskort», *Store norske leksikon*, 18-des-2017. [Online]. Tilgjengelig på: <http://snl.no/kretskort>. [Åpnet: 25-apr-2019].
- [98] T. Lynghaug, «Intro til PCB - Printed Circuit Board - Kretskort». Universitetet i Agder, sep-2016.
- [99] Wikipedia, «Low-dropout regulator», *Wikipedia*, 18-apr-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Low-dropout_regulator&oldid=893076905. [Åpnet: 09-mai-2019].
- [100] Wikipedia, «Voltage regulator», *Wikipedia*, 11-mar-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Voltage_regulator&oldid=887257560. [Åpnet: 12-mai-2019].
- [101] J. Patoux, «Ask The Applications Engineer—37: Low-Dropout Regulators». [Online]. Tilgjengelig på: <https://www.analog.com/en/analog-dialogue/articles/low-dropout-regulators.html>. [Åpnet: 25-apr-2019].
- [102] R. Keim, «PCB Layout Tips and Tricks: Use a Ground Plane Whenever Possible». [Online]. Tilgjengelig på: <https://www.allaboutcircuits.com/technical-articles/pcb-layout-tips-and-tricks-use-a-ground-plane-whenever-possible/>. [Åpnet: 03-mai-2019].

- [103] Wikipedia, «Ground plane», *Wikipedia*, 04-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Ground_plane&oldid=895510685. [Åpnet: 09-mai-2019].
- [104] Haihua Su, S. S. Sapatnekar, og S. R. Nassif, «Optimal decoupling capacitor sizing and placement for standard-cell layout designs», *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, bd. 22, nr. 4, s. 428–436, apr. 2003.
- [105] Altium, «Generate Gerber Files in Altium Designer Step-by-Step from Schematic to PCB». [Online]. Tilgjengelig på: <https://resources.altium.com/pcb-design-blog/generate-gerber-files-in-altium-step-by-step-from-schematic-to-pcb>. [Åpnet: 11-mai-2019].
- [106] Altium, «How to Create a PCB Layout from a Schematic in Altium Designer». [Online]. Tilgjengelig på: <https://resources.altium.com/pcb-design-blog/how-to-create-a-pcb-layout-from-a-schematic-in-altium-designer>. [Åpnet: 08-mai-2019].
- [107] Wikipedia, «Footprint (electronics)», *Wikipedia*, 12-feb-2016. [Online]. Tilgjengelig på: [https://en.wikipedia.org/w/index.php?title=Footprint_\(electronics\)&oldid=704661069](https://en.wikipedia.org/w/index.php?title=Footprint_(electronics)&oldid=704661069). [Åpnet: 11-mai-2019].
- [108] S. Riege, «Polygon Pour». [Online]. Tilgjengelig på: [https://www.altium.com/documentation/18.0/display/ADES/PCB_Obj-PolygonPour\(\(Polygon+Pour\)\)_AD](https://www.altium.com/documentation/18.0/display/ADES/PCB_Obj-PolygonPour((Polygon+Pour))_AD). [Åpnet: 11-mai-2019].
- [109] Wikipedia, «Gerber format», *Wikipedia*, 10-mar-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Gerber_format&oldid=887076806. [Åpnet: 12-mai-2019].
- [110] «ProtoMat S103». [Online]. Tilgjengelig på: https://forms.zohopublic.com/ataylor/form/ProtoMatS103/formperma/tqIF_Z0B9cfd3GTYOCT2N1VZUPcJXW3-GwjIPq1wz2k. [Åpnet: 12-mai-2019].
- [111] amtest, «LPKF ProtoMat S103», *Amtest SMT d.o.o.* [Online]. Tilgjengelig på: <http://www.amtest-smt.com/hr/proizvodi/izrada-prototipova-tiskanih-plocica/uredaji-za-iscrtavanje-elektronickih-plocica/lpkf/lpkf-protomat-s103,181.html>. [Åpnet: 12-mai-2019].
- [112] PCB 3D, «Create a new Integrated Library, add Schematic and PCB library», *PCB 3D*.
- [113] Eclipse, «Eclipse Paho». [Online]. Tilgjengelig på: <https://www.eclipse.org/paho/>. [Åpnet: 09-mai-2019].
- [114] S. Kock, «MQTT Client Library Encyclopedia – Paho Android Service». [Online]. Tilgjengelig på: <https://www.hivemq.com/blog/mqtt-client-library-encyclopedia-paho-android-service/>. [Åpnet: 09-mai-2019].
- [115] Nordic Semiconductor, «Battery life calculation for NRF52 and NRF51(Beacon kit)», *Nordic DevZone*. [Online]. Tilgjengelig på: <https://devzone.nordicsemi.com/f/nordic-q-a/21571/battery-life-calculation-for-nrf52-and-nrf51-beacon-kit>. [Åpnet: 07-mai-2019].
- [116] Wikipedia, «Samsung Galaxy S5», *Wikipedia*, 05-mai-2019. [Online]. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=Samsung_Galaxy_S5&oldid=895548031. [Åpnet: 10-mai-2019].
- [117] National Heart Lung, and Blood Institute, «Sleep Deprivation and Deficiency». [Online]. Tilgjengelig på: <https://www.nhlbi.nih.gov/health-topics/sleep-deprivation-and-deficiency>. [Åpnet: 10-mai-2019].
- [118] Norsk Helseinformatikk, «Fall og fallskader hos eldre», *NHI.no*. [Online]. Tilgjengelig på: <https://nhi.no/sykdommer/eldre/diverse-problemstillinger/fall-og-fallskader-hos-eldre/>. [Åpnet: 10-mai-2019].
- [119] TinyCircuits, «Lithium Ion Polymer Battery - 3.7V 70mAh», *TinyCircuits*. [Online]. Tilgjengelig på: <https://tinycircuits.com/products/lithium-ion-polymer-battery-3-7v-70mah>. [Åpnet: 13-mai-2019].
- [120] J. Comstock, «21 clinical trials that are using Fitbit activity trackers right now», *MobiHealthNews*, 16-mar-2016. [Online]. Tilgjengelig på: <https://www.mobihealthnews.com/content/21-clinical-trials-are-using-fitbit-activity-trackers-right-now>. [Åpnet: 13-mai-2019].
- [121] K. Østvang, «Nå kan man ta EKG-måling med Apple Watch i Norge», *DinSide.no*, 27-mar-2019. [Online]. Tilgjengelig på: <http://www.dinside.no/a/70910421>. [Åpnet: 13-mai-2019].

Referanser

- [122] M. Valle, «Alt du trenger å vite om trådløs lading», *Tu.no*, 25-okt-2013. [Online]. Tilgjengelig på: <https://www.tu.no/artikler/test-alt-du-trenger-a-vite-om-tradlos-lading/226806>. [Åpnet: 13-mai-2019].
- [123] StackExchange, Txugo, «Use Pi as internet gateway», *Raspberry Pi Stack Exchange*. [Online]. Tilgjengelig på: <https://raspberrypi.stackexchange.com/questions/42260/use-pi-as-internet-gateway>. [Åpnet: 13-mai-2019].

Vedlegg

Vedleggliste

Se *Tabell 18* for en liste over vedlegg som medfølger denne rapporten.

Vedlegg	Tittel	Plassering
Vedlegg A	Ordliste og forkortelser	Side 90
Vedlegg B	BOM (Bill of Materials)	Side 92
Vedlegg C	Pressemelding	Side 94
Vedlegg D	Prosjektplan	Side 95
Vedlegg E	Gruppektrakter	Side 96
Vedlegg F	Møtereferater	Side 98
Vedlegg G	Timelister	Side 119
Vedlegg H	Kildekode fastvare	Egen mappe ¹⁸
Vedlegg I	Kildekode mobilapp	Egen mappe ¹⁹
Vedlegg J	Kretskortdesign	Egen mappe ²⁰

Tabell 18: Liste over vedlegg.

¹⁸ Kildekoden for fastvaren ligger kun som eget vedlegg. Se mappe *vedlegg_H_kildekode_fastvare* i mappen *vedlegg*. Prosjektet må pakkes ut. Deretter kan prosjektfilen *PasArm_pca10040_s112.emProject* åpnes i *SEGGER Embedded Studio*. Kildekoden kan også lastes ned fra github.com/robino16/mso.

¹⁹ Kildekoden for mobilappen ligger kun som eget vedlegg. Se mappe *vedlegg_H_kildekode_mobilapp* i mappen *vedlegg*. Prosjektet må pakkes ut. Deretter kan prosjektet *mso-login* åpnes i *Android Studio*. Kildekoden kan også lastes ned fra github.com/robino16/mso.

²⁰ *Vedlegg J* er bildefiler av kretskortdesign og kretsskjematikk for Prototype 1 og Prototype 2. Disse ligger kun som et eget vedlegg. Se mappe *vedlegg_J_kretskortdesign* i mappen *vedlegg*.

Vedlegg A: Ordliste og forkortelser

Se *Tabell 20* for en liste over ord og forkortelser som fremkommer i denne rapporten. Listen er sortert i alfabetisk rekkefølge.

Forkortelse	Begrep
AC	Vekselstrøm / Alternating Current
AI	Kunstig Intelligens / Artificial
API	Programmeringsgrensesnitt / Application Programming Interface
Bluetooth LE / BLE	Bluetooth Low Energy
DC	Likestrøm / Direct Current
DK	Utviklingskort / Development Kit
EKG	Elektrokardiogram
FHIR	Fast Healthcare Interoperability Resources
GATT	General Attribute Profile
GFSK	Gaussian Frequency Shift Keying
GPIO	General Purpose Input and Output
IC	Integrert krets / Integrated Curcuit
IDE	Integrert utviklingsmiljø / Integrated Development Environment
IoT	Internett av ting / Internet of Things
I ² C	Inter-Integrated Curcuit
LDO	Low Dropout Regulator
LED	Lysemitterende diode / Light Emitting Diode
LiPo	Litium-polynomer / Lithium Polynomer
MCU	Mikrokontroller / Microcontroller Unit
ML	Maskinlæring / Machine Learning
MQTT	Message Queuing Telemetry Transport

NVIC	Nested Vectored Interrupt Controller
PCB	Kretskort / Printed Curcuit Board
SCL	Seriellklokke / Serial Clock
SDK	Software Development Kit
SMT	Surface-mount Technology
SoC	System On Chip
TWI	Two Wired Interface
THR	Target Heart Rate
TWI	Two Wire Interface
UI	Brukergrensesnitt / User Interface
WFE	Wait For Event
WFI	Wait For Interrupt
XML	Extensible Markup Language

Tabell 19: Ordliste og forkortelser.

Vedlegg B: Regning av materialer/BOM (Bill of Materials)

Se Tabell 21 for en regning av påkrevde komponenter til å utvikle én prototype av pasientarmbåndet.

Prototype 1

Produsent	Serienummer	Antall	Beskrivelse	Montering
Aconno GmbH	ACN52832	1	Mikrokontroller-modul, nRF52832 chip, 20,3 x 25 x 3 mm	SMD/SMT
Rohm Semiconductor	BH1790GLC-E2	1	Puls-sensor, 2,8 x 2,8 x 1,0 mm	SMD/SMT
Rohm Semiconductor	SML-M13MTT86	2	Grønn LED	SMD/SMT
TinyCircuits	ASR00011	1	Oppladbart litium-polynomer-batteri, 3.7 V, 70 mAh	SMD/SMT
Adafruit	1904	1	Lade-modul	SMD/SMT
TDK InvenSense	MPU-6050	1	Akselerometer	SMD/SMT
AVX	06035D105KAT2A	2	Kondensator 1uF, 0603 størrelse	SMD/SMT
Texas Instruments	TLV75533PDBVR	1	Spenningsregulator (LDO)	SMD/SMT

Tabell 20: Regning av materialer for Prototype 1.

Prototype 2

Produsent	Serienummer	Antall	Beskrivelse	Montering
Aconno GmbH	ACN52832	1	Mikrokontroller-modul, nRF52832 chip, 20,3 x 25 x 3 mm	SMD/SMT
Rohm Semiconductor	BH1790GLC-E2	1	Puls-sensor, 2,8 x 2,8 x 1,0 mm	SMD/SMT
Rohm Semiconductor	SML-M13MTT86	2	Grønn LED	SMD/SMT
TinyCircuits	ASR00011	1	Oppladbart litium-polynomer-batteri, 3.7 V, 70 mAh	SMD/SMT
Adafruit	1904	1	Lade-modul	SMD/SMT
TDK InvenSense	MPU-6050	1	Akselerometer	SMD/SMT
Texas Instruments	TLV75533PDBVR	1	Spenningsregulator (LDO)	SMD/SMT
AVX	06035D105KAT2A	2	Kondensator 1uF, 0603 størrelse	SMD/SMT
N/A	N/A	1	Kondensator 10uF, 1206 størrelse	SMD/SMT
N/A	N/A	1	Kondensator 2.2uF, 1206 størrelse	SMD/SMT
N/A	N/A	1	Kondensator 100uF, 1206 størrelse	SMD/SMT

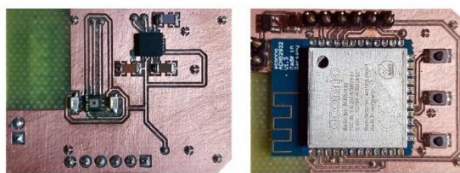
Tabell 21: Regning av materialer for Prototype 2.

Vedlegg C: Pressemelding

Pressemelding (15.05.2019)

Dette studentprosjektet skal gjøre din pasientopplevelse bedre

I samarbeid med Bouvet ASA har ingeniørstudenter på Universitetet i Agder utviklet et e-helse-armbånd som skal overvåke helsedataen til pasienter på sykehus.



Figur 58: Prototypen til e-helse-armbåndet.

Tre studenter fra Elektronikkingeniør linjen på Universitetet i Agder, har utviklet en prototype som skal effektivisere hverdagen til helsepersonell og gi pasientene eierskap til egen helsedata. Aksel Melby Haugen, Robin Omslandseter og Thomas André Rislå har i løpet av vårsemesteret 2019 hatt i oppgave å utvikle et armbånd som kan bedre pasientopplevelsen på sykehus.

En forbedring av pasientopplevelsen

Hverdagen til pasienter på norske sykehus kan ofte fremstå kaotisk. Helsepersonell som løper frem og tilbake for å gjøre målinger, men måleresultatene har du som pasient lite innsyn i. Visjonen å gi pasientene eierskap til sin egen helsesituasjon gjennom bruk av et e-helse-armbånd. Armbåndet skal gjøre enkelte målinger som helsepersonellet trenger og fremviser disse målingene i en mobilapplikasjon for både pasient og personell. Med dette er det ønskelig å gi pasienten en trygghet og oversikt over egen helsesituasjon. I tillegg til å løsrive helsepersonellet fra enkelte målinger som tidligere har blitt gjort manuelt.

Et større konsept

Pasientarmbåndet som blir utviklet skal være en byggekloss i et større prosjekt med tittelen «Mitt Sykehusopphold». «Mitt Sykehusopphold» er et system som skal ta for seg hele sykehusoppholdet fra før pasienten ankommer til etter han/hun har reist hjem. Dette prosjektet skal ta for seg alt av logistikk som en pasient opplever på daglig basis. Oversikt over prosedyrer og behandlinger, planlegging av måltider og testresultater er eksempler på funksjoner som skal samles i én og samme app. På denne måten vil pasienter ha bedre innsyn i sitt sykehusopphold.

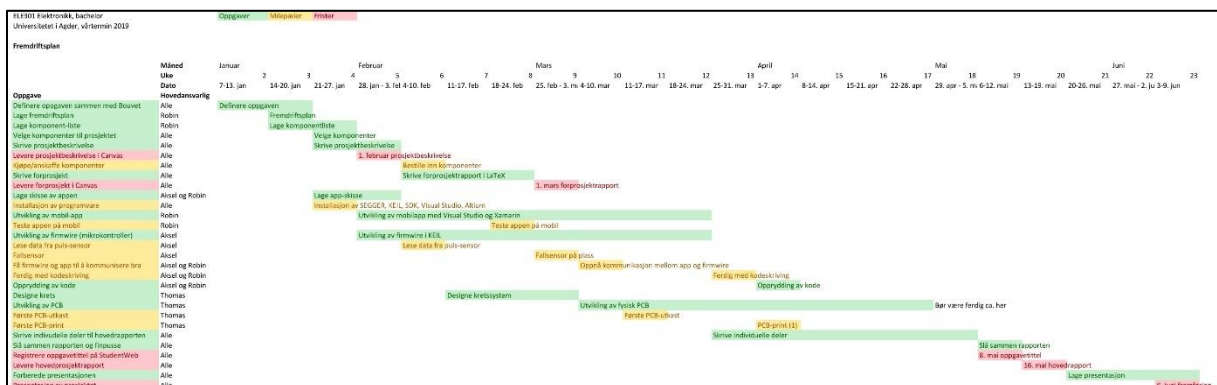
Vedlegg D: Prosjektplan

Navn	Rolle	Selskap
Aksel Melby Haugen	Student	UiA
Robin Omslandseter	Student	UiA
Thomas André Rislåa	Student	UiA
Ken Henry Andersen	Veileder	UiA
Lei Jiao	Veileder	UiA
Tonje Holand Salgado	Administrasjon og oppfølging/Styringsgruppe	Bouvet
Ivar Sønstabø	Avdelingsleder Arendal/Styringsgruppe	Bouvet
Eivind Auestad	Teknisk oppfølging	Bouvet

Tabell 22: Prosjektdeltakere og roller.

Oppgave	Hovedansvarlig
Kretskort-design (PCB-design)	Thomas André Rislåa
Programmering av mikrokontroller og sensorer	Aksel Melby Haugen
Mobil-applikasjon og kommunikasjon	Robin Omslandseter

Tabell 23: Inndeling av ansvarsområder.



Figur 59: Gnatt-chart/fremdriftsplan.