Emir Derouiche
Nikolai Roede Dokken
Ian-Angelo Roman Evangelista
Kasper Vedal Gundersen

Bacheloroppgave

Developing Mobile Applications for Healthcare Professionals

Utvikling av mobilapplikasjoner for helsepersonell

Bacheloroppgave i ingeniørfag, data

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektronikk
Institutt for datateknologi og informatikk

Veileder: Elise Klæbo Vonstad
Mai 2021

NTNU
Kunnskap for en bedre verden

Forord

Denne bacheloroppgaven er gjennomført i samarbeid med Infiniwell AS, våren 2021. Oppgaven regnes som et avsluttende prosjekt for studiet Bachelor i ingeniørfag, data ved Norges teknisk-naturvitenskapelige universitet i Trondheim.

Gjennom studiet har vi lært både systemutvikling og webutvikling, men utvikling for mobiltelefoner var nytt. Det har derfor vært en spennende, utfordrende og ikke minst en lærerik prosess å utvikle en mobilapplikasjon. Det har også vært givende at applikasjonen kan bidra til å redde liv og gjøre hverdagen til helsepersonell lettere.

Vi vil takke Infiniwell AS og spesielt daglig leder Odd Sandbekkhaug for et utrolig godt samarbeid. Odd har stilt seg og teamet hans disponible til oppfølging og veiledning. Vi vil også rette en stor takk til den faglige veilederen vår, Elise Klæbo Vonstad fra NTNU, for den gode oppfølgingen og de gode rådene hun ga oss gjennom prosjektet.

Denne oppgaven er en videreføring av et prosjekt vi startet på høsten 2020 i emnet TDAT3022 Systemutviklingsprosjekt ved NTNU. Da var målet å vise data strømmet fra Infiniwells pasient-monitorer, samt å vise en enkel oversikt over pasienter. I dette prosjektet utvider vi produktet fra høstsemesteret med en detaljert pasientjournal, et omfattende varslingssystem, og et meldingssystem som lar helsepersonell kommunisere med hverandre, både direkte og i grupper. Odd forklarte det godt ved å si at høstprosjektet var en prototype, mens bacheloroppgaven skulle etterstrebe et fullverdig produkt.

Trondheim, 20.05.2021

Sted, Dato

Emir Derouiche

Nikolai Roede Dokken

Ian-Angelo Roman Evangelista

Kasper Vedal Gundersen

Oppgavetekst

Bakgrunnen for oppgaven er Infiniwell AS sitt eksisterende system som gir helsepersonell tilgang til pasientdata via nettleser. Oppgaven vår var å utvikle en mobilapplikasjon som tar i bruk enkelte funksjonaliteter fra deres eksisterende system, samt legge til nye og utvidede funksjonaliteter. Applikasjonen skal kunne brukes på både iOS og Android mobiltelefoner. Slik lød oppgaveteksten vi fikk tildelt:

Teamet vil:

- *Implementere og teste back-end infrastruktur for push-notifications. ... Dette muliggjør to-veis kommunikasjon imellom back-end server og mobil app.*
- *Utvikle en mobil app (cross-platform for Android og iPhone/iPad) som viser pasient informasjon (blodtrykk, puls, temperatur, ...), alarm-meldinger og har en enkel chat-funksjon.*
- *Utvikle en enkel måte å autorisere brukere for at de kan se bare de pasienter de har rettighet til. ... Vi planlegger å bruke on-screen QR kode for enkel pairing med pasient id (dvs. mobil kamera leser QR kode på skjermen for å autorisere), men vi trenger også en måte å "invitere" brukere som skal kunne se pasient data.*
- *Om det er nok tid i prosjektet, integrere prosjektet inn i Infiniwell's systemer for live deployment.*

Når prosjektet er ferdigstilt, vil helsepersonell ha en mobilapplikasjon som lar dem se all data om deres pasienter, samt motta notifikasjoner og sende/motta meldinger. Produktet vil bli en viktig del av Infiniwells eksisterende pasientovervåkning- og diagnostikkplattform.

Sammendrag

Helse- og omsorgsminister Bent Høie har uttalt på vegne av regjeringen at de ønsker å styrke digitaliseringen i helsesektoren og sørge for bedre utnyttelse av helsedata [Regjeringen.no, 2020]. Infiniwell planlegger å bidra til digitaliseringen ved å automatisere pasientdiagnostisering ved hjelp av kunstig intelligens og pasientmonitorer som strømmer data direkte til helsepersonell. I denne oppgaven vil vi beskrive vårt arbeid med å tilby Infinwell en mobilapplikasjon som tar i mot data fra pasientmonitorer gjennom en sky-basert løsning. Målet til Infiniwell er å effektivisere helsepersonells hverdag ved å minke tiden brukt på administrative oppgaver, og å bistå med å ta kritiske valg. For å nå dette målet, trenger helsepersonell en nyttig og brukervennlig mobilapplikasjon.

I denne rapporten reflekterer vi over utviklingen av en app som har som mål å være nyttig og brukervennlig for helsepersonell. Vi vil forsøke å svare på hvilke teknologier som kan være gode for formålet, og hva som gjør en applikasjon brukervennlig. For å gjøre dette vil vi se på relevant teori tilknyttet vår utvikling av applikasjonen og sammenlikne vårt resultat med teori samt tilbakemeldinger fra helsepersonell som har brukertestet applikasjonen vår. I tillegg til anerkjent teori om brukeropplevelse og design, vil du også kunne lese tilbakemeldinger fra helsepersonell i Norge, India og USA.

Scrum-metodikken ble benyttet for prosjektet. Vi vil begrunne hvorfor metodikken ble valgt og reflekterer rundt hvordan den har påvirket produktet. Mobilapplikasjonen er skrevet i JavaScript med rammeverkene React Native og Expo. For tjenerkoden benyttet vi Python med Django-rammeverket og for databasen benyttet vi PostgreSQL. Vi diskuterer årsakene til at vi valgte å benytte disse språkene og rammeverkene, og hva det har hatt som betydning for produktet.

Gjennom å diskutere hva vi har gjort bra og hva som kunne vært bedre, ønsker vi til slutt å konkludere med hvordan man skal lage en nyttig og brukervennlig mobilapplikasjon for helsepersonell.

Innholdsfortegnelse

Figurer

Tabeller

Akronymer og forkortelser

API: Application Programming Interface — Programmeringsgrensesnitt

CD: Continious delivery/deployment — Kontinuerlig levering/utrulling

CI: Continious integration — Kontinuerlig integrasjon

EKG: Elektrokardiogram

GUI: Graphical User Interface — Grafisk brukergrensesnitt

HTTPS: Hypertext Transfer Protocol Secure

IBP1: Invasive Blood Pressure

IDL: Interface Descriptive Language

IKT: Informasjons-og kommunikasjonsteknologi

ISO: Den internasjonale standardiseringsorganisasjonen

IxD: Interaction Design — Interaksjonsdesign

KP: Kryss-plattform

MMI: Menneske-maskin-interaksjon

NEWS2: National Early Warning Score 2

ORM: Object-Relational Mapping

PWA: Progressive Web Application — Progressiv Web Applikasjon

REST: Representational State Transfer

SpO2: Oksygenmetning

UI: User Interface — Brukergrensesnitt

UU: Universell utforming

UX: User Experience — Brukeropplevelse

VCS: Version Control System — Versjonskontrollsystem

W3C: World Wide Web Consortium

WAI: Web Accessibility Initiative

WCAG: Web Content Accessibility Guidelines

1 Introduksjon og relevans

1.1 Bakgrunn for oppgaven

Infiniwell AS er et helse- og teknologiselskap som har utviklet et telemedisin-system som har fått navnet Sentio. Systemet bruker kunstig intelligens i sine analyser for å stille diagnoser til pasienter. Døgnet rundt strømmes og registreres vitale data i sanntid til helsepersonell. Disse dataene har man tilgang til i ettertid og man kan se ulike diagnoser som har blitt stilt avhengig av pasientens tilstand. I tillegg får man tilgang journalen til en pasient. I pasientjournalen finner man data tilknyttet verdier målt fra pasienten som alarmer, notater og observasjoner. Det er også mulig å kommunisere med kollegene sine gjennom en chat.

Begrensningen med det nåværende systemet er at man bare har tilgang til Sentio gjennom nettleser. Infiniwell ønsker derfor at vi skal utvikle en mobilapplikasjon for å gi deres brukere enda raskere tilgang til systemet deres. Gjennom den nye mobile plattformen vil brukeren slippe å ha tilgang til en datamaskin for å monitorere pasientene sine. Samtidig åpner mobilapplikasjonen for at leger og helsepersonell får varslinger på sine mobile enheter dersom tilstanden til en pasient er kritisk. I et slikt tilfelle kan brukere organisere assistanse til pasienter i samkjør med sine kolleger.

St. Olavs hospital sier at helsepersonell har to overordnete behov: samle nøyaktig data raskt og lette kognitiv belastning [St. Olavs, 2021]. Ved å utvikle en mobilapplikasjon for Infiniwells webløsning vil vi tilfredstille disse behovene. Appen vil tillate brukeren å observere data i sanntid og tilbake i tid, den vil la deg dokumentere og dele data, samt gjøre brukeren oppmerksom på viktig informasjon. I tillegg vil appen minimere veksling mellom ulike systemer og ikke minst legge til rette for enkel og effektiv kommunikasjon mellom brukerne.

1.2 Problemstilling

Ettersom mobilapplikasjonen skal brukes av leger, sykepleiere og annet helsepersonell, er det viktig at den er enkel og intuitiv å bruke. Dette er fordi helsepersonell trenger rask tilgang til pasientene sine, men også andre kolleger. Det kan eksempelvis være utfordrende for en bruker dersom applikasjoner er rotete og inneholder mange unødvendige funksjonaliteter som ikke er relevant for brukeren. Derfor ønsker vi å ta for oss problemstillingen:

Hvordan utvikle en brukervennlig og nyttig app for helsepersonell med fokus på brukergrensesnitt og brukeropplevelse?

Vi vil forske på ulike måter å designe en applikasjon for å gjøre den mest mulig brukervennlig. Vi vil utforske om kjente prinsipper innenfor brukeropplevelse og interaksjonsdesign kan bidra til å utvikle en brukervennlig app. Spesielt ønsker vi å avdekke hvordan man utvikler en app som kan være et nyttig verktøy for helsepersonell.

1.3 Struktur

Kapittel 1: Introduksjon og relevans, viser til bakgrunnen for oppgaven, problemstillingen og strukturen på rapporten.

Kapittel 2: Teori, forklarer teorien som er brukt i rapporten. Inneholder teori om data vi strømmer til applikasjonen vår, brukeropplevelse og interasjonsdesign, og utviklingsmetodikken vi har benyttet, Scrum.

Kapittel 3: Valg av teknologi og metode, presenterer de ulike teknologiene som er blitt brukt og hvorfor disse er valgt. Kapittelet tar også for seg hvordan vi har valgt å utvikle og hvorfor vi valgte å jobbe på den måten.

Kapittel 4: Resultater, viser til resultatene fra prosjektet.

Kapittel 5: Diskusjon, diskuterer problemstillingen sett i sammenheng med resultatene vi fikk i kapittel 4 og relevant teori.

Kapittel 6: Konklusjon og videre arbeid, svarer på problemstillingen og tar for seg hva som kan gjøres dersom prosjektet skal videreutvikles.

Kilder, viser alle kildene som er brukt i rapporten.

Vedlegg, viser til alle vedlegg som hører til dette dokumentet.

2 Teori

Dette kapittelet vil beskrive den teorietiske bakgrunnen for arbeidet som har blitt gjort for å løse både problemstillingen og for å utvikle produktet. Til å begynne med vil vi ta for oss det teoretiske grunnlaget som ble valgt høsten 2020 i forbindelse med emnet TDAT3022 Systemutviklingsprosjekt. Deretter vil vi presentere teorien knyttet til problemstillingen og vise til teori som begrunner valgene våre rundt design og brukskvalitet. I siste del av kapittelet vil vi ta for oss det teoretiske grunnlaget for den valgte utviklingsmetodikken.

2.1 Videreføring av systemutviklingsprosjekt til bacheloroppgave

Høsten 2020 tok vi faget TDAT3022 Systemutviklingsprosjektet, der vi valgte en oppgave fra Infiniwell som vårt prosjekt. Oppgavens omfattende arbeidsmengde tilsa at den kunne videreføres fra systemutviklingsprosjekt til bachelorprosjekt, dersom ønskelig. Dette betød at ved oppstart av høstprosjektet la vi til grunn en del avgjørelser som vi ble nødt til å videreføre til bachelorprosjektet. Her vil vi beskrive teorien som har hatt en direkte betydning for arbeidet som er lagt ned under bachelorprosjektet.

2.1.1 Utvikling av mobilapplikasjoner

Det finnes flere måter å utvikle mobilapplikasjoner på. De tre vanligste framgangsmåtene er native utvikling, Progressiv Web Applikasjon og kryss-plattform.

Native utvikling

Native utvikling bruker programmeringsspråk som kompileres til binære filer og kjøres direkte på enheten. Native kode er lavere nivå og dermed vanligvis raskere og med flere funksjoner, men til gjengjeld må de utvikles i operativsystemets respektive språk. Dette kan for eksempel være Kotlin eller Java for Android og Objective-C for iOS. Native apper har fordelen at man kan tilpasse design og funksjonalitet til det enkelte operativsystemet. På den annen side krever det flere separate kodebaser dersom man ønsker at en applikasjonen skal støttes av flere ulike operativsystemer [Clearbridge, 2020].

Progressiv Web Applikasjon - PWA

PWA er en responsiv web-løsning med egenskaper som muliggjør notifikasjoner og posisjonssporing. PWA kan hentes fra nettet og lagres som en applikasjon eller bokmerke på mobile enheter. Fordelen med PWA er at man slipper å distribuere applikasjonen gjennom en app-butikk. Oppdateringer kommer umiddelbart uten at man må vente på godkjenninger fra butikken. Ulempen med PWA er at det er en mindre anerkjent form for mobilapplikasjon, og ikke alle enheter tilbyr funksjonalitet for dem. PWA kan også oppleves som tyngre på nettverksbruken, som kan være en ulempe om man vil begrense mobildatabruk. PWA krever mindre lagringsplass på enheten, men har også treg lastetid i forhold til installerte applikasjoner [Richard and LePage, 2020].

Kryss-plattform

Et **kryss-plattform**-rammeverk utnytter én enkelt kodebase som kompileres, bygges og kjøres på flere forskjellige operativsystemer [Danielsson, 2016]. Dette gjør at man slipper å ta hensyn til ulike operativsystemer og dermed slipper man å utvikle ulike versjoner av samme produkt. I likhet med native applikasjoner, bygges og distribueres KP-applikasjoner gjennom app-butikken til mobilen.

2.2 Helsetjenesten

2.2.1 Pasientmonitorering

På sykehus og andre helseinstitusjoner benyttes pasientmonitorer som har som formål å registrere pasienters vitale data. Data vises kontinuerlig på skjermen så lenge monitoren er koblet til pasienten. En monitor kobles til en pasient ved bruk av slanger og ledninger. Data registreres og overføres til monitoren, og fra monitoren til en overvåkingssentral der dataen lagres. På overvåkningssentralen sitter det helsepersonell og monitorerer data som kommer inn samtidig fra flere pasienter. Data fra en pasient kan også sees fra en helsepersonells arbeidsstasjon. Lagringstiden på dataen varierer, men som regel er det mellom en dag og en uke. Dataen fra vitale tegn brukes ikke bare til å sees i sanntid, men også for å studere trendkurver, altså variasjonen i dataen over tid [Fanebust, 2014].

2.2.2 Vitale tegn

I følge St. Olavs hospital er det seks fysiologiske parametere som er standard målinger for en pasient [St. Olavs, 2021]:

- **Puls:** "En trykkstigning som forplanter seg gjennom blodet i arteriene hver gang hjertet trekker seg sammen. Trykkstigningen utvider de elastiske arteriene, som straks etter trekker seg sammen igjen" [Arnesen, 2020].
- **Oksygenmetning:** "Et mål for hvor stor andel av hemoglobinet i blodet som har inngått en kjemisk forbindelse med oksygen. Andelen angis i prosent av hemoglobinets maksimale oksygenbindingsevne". Normal oksygenmetning er 97-98% [Åshild Odden Miland, 2018].
- **Blodtrykk:** "Det trykket som blodet utøver mot blodåreveggen. Hos en ung, voksen person i hvile er 120/70 mmHg et vanlig blodtrykk." [Hisdal, 2021].
- **Respirasjonsfrekvens**: "Antall innåndinger per minutt. Respirasjonsfrekvensen for en voksen person i hvile vil vanligvis være i området 10–18" [Åshild Odden Miland, 2021].
- **Temperatur:** "Temperaturen i kroppens indre hos mennesket varierer under vanlige forhold i hvile mellom 36,5 °C og 37,5 °C" [Hauge, 2019].
- **Bevissthet og bevissthetsnivå:** "Menneskets evne til å oppleve, registrere og sanse hva som hender i ens omgivelser og med en selv" [Hansen, 2020].

2.2.3 Elektrokardiogram - EKG

Elektrokardiogram er en metode som registrerer de elektriske impulsene som sendes gjennom hjertet [NHI, 2019]. Den måler de elektriske spenningsforskjellene i hjertemuskulaturen når hjertet jobber [Arnesen, 2018]. EKG gir en god indikasjon på hjertets tilstand. I EKG ser man eksempelvis etter endringer i hjerterytmen, overbelastning i hjertet og tegn til nye eller eldre hjerteinfarkt.

2.2.4 Typer pasientmonitorer

Det finnes ulike typer pasientmonitorer. Forskjellige sykehus bruker forskjellige typer, men på St. Olavs hospital [St. Olavs, 2021] er det tre standarder:

- Pasientmonitor: brukes av og plasseres ved hver pasient og måler alle vitale tegn. Data fra monitoren strømmes til en sentral.
- Bærbar pasientmonitor: en bærbar enhet som kan kobles fra en stasjonær pasientmonitor. Brukes når en pasient må flyttes og kobles til en ny stasjonær pasientmonitor etter at pasienten er flyttet. Data lagres i monitoren og når den kobles til en stasjonær pasientmonitor igjen, vil data strømmes til sentralen.

- Sentral pasientmonitor: gir overblikk over all data som strømmes fra ulike pasientmonitorer. Helsepersonell kan ved bruk av dette, monitorere flere pasienter samtidig.

2.2.5 NEWS2-score

NEWS2-score beregnes ut i fra de seks fysiologiske parametere nevnt i delkapittel 2.2.2. Hver enkelt parameter får en score. Deretter beregnes en total NEWS2-score basert på de ulike parameterne [Royal College of Physicians, 2017].

2.3 Systemutvikling

2.3.1 Versjonskontrollsystem

Versjonskontrollsystem gjør det mulig for flere personer å endre samme prosjekt eller fil samtidig. Systemet holder styr på ulike versjoner av filene. Dette gjør det mulig å gå tilbake til en tidligere versjon av et prosjekt eller fil. Man kan dermed holde styr på hvem som har gjort hva, endringer over tid og sammenligne forskjeller [Chacon and Straub, 2014].

2.3.2 Kontinuerlig integrasjon - CI

Kontinuerlig integrasjon er en praksis innenfor systemutvikling. Når utviklerne laster opp koden sin til den sentrale kodebasen kjøres automatiske tester som verifiserer at koden fungerer som den skal. Målet ved CI er å finne feil raskere, forbedre systemkvaliteten og redusere tiden det tar å godkjenne og slippe nye systemoppdateringer [Amazon Web Services, 2021b].

2.3.3 Kontinuerlig levering/utrulling - CD

Kontinuerlig levering er en praksis innenfor systemutvikling. Etter at koden er bygget fra CI, vil all kode som er endret bli levert til et testmiljø eller produksjonsmiljø. CD lar utviklerne automatisere flere typer tester og gjør det mulig å finne feil raskere. Forskjellen på kontinuerlig levering og kontinuerlig utrulling er at ved kontinuerlig levering må man manuelt velge å trigge utrullingen. Ved kontinuerlig utrulling er ikke manuell trigging nødvendig [Amazon Web Services, 2021a].

2.3.4 REpresentational State Transfer

REpresentational State Transfer eller REST er en type arkitektur som brukes for web-tjenester for uthenting, fjerning og endring av data. En web-tjeneste anses som RESTful dersom man følger REST sine seks retningslinjer [restfulapi.net, 2020]:

- **Klient-tjener arkitektur:** separere brukergrensesnittet fra datahåndteringen for å øke fleksibiliteten til systemet.
- **Tilstandsløs:** hver forespørsel fra klienten har all nødvendig informasjon for at tjeneren skal skjønne hva den skal gjøre. Tjeneren skal ikke være avhengig av noe annet lagret informasjon.
- **Hurtigbuffer:** responser lagres i en hurtigbuffer dersom klienten forsøker å gjøre samme forspørsel flere ganger.
- **Uniformt grensesnitt:** standardisere informasjonen som blir sendt mellom ulike komponenter.
- **Lagdelt system:** klienten vet ikke om den kommuniserer med slutt-tjeneren eller andre tjenere. Tjeneren kan bestå av flere lag som for eksempel et lag med sikkerhet.
- **Kode på kommando (valgfri):** gir muligheten for tjeneren å returnere kjørbar kode til klienten.

Med REST gjør man HTTPS-kall og får data returnert i formater som JSON, XML, YAML eller andre formattyper basert på hva klienten ønsker i retur. Fordelen med REST er at det er tilgjengelig for alle typer klienter. Det vil si at den fungerer hvis klienten er web-basert eller mobil-basert. En annen fordel er at det brukes et hurtigbuffer dersom det er mange forspørsler av samme type ressurs. Da returnerer hurtigbufferen ressursen til andre klienter dersom det allerede har blitt forespurt om ressursen i forkant.

2.4 Brukskvalitet og brukeropplevelse

Det er ønskelig at alle typer systemer skal være brukervennlig og gi en god brukeropplevelse. Under vil vi beskrive teori som omhandler brukskvalitet og brukeropplevelse.

2.4.1 Brukskvalitet

Den internasjonale standardiseringsorganisasjonen (ISO) har en standard der de beskriver at brukskvalitet defineres som: "The extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [International Organization for Standardization, 2018]. Et systems brukskvalitet blir i følge ISO målt opp mot anvendbarhet, effektivitet og tilfredshet. Disse tre faktorene kan bare måles når man vet hvem brukerne av systemet er, hva systemet skal brukes til, og hvor og i hvilken sammenheng det skal brukes. I Jakob Nielsens bok, *Usability Engineering*, skriver han at man oppnår god brukskvalitet dersom systemet er [Nielsen, 1993]:

- Lett å lære, så brukere kan gå raskt fra ikke å kjenne systemet til å gjøre noe arbeid.
- Effektivt, lar ekspertbrukeren oppnå en høy grad av produktivitet.
- Lett å huske, så brukere med lav brukshyppighet kan returnere etter en periode med inaktivitet uten å måtte lære alt på nytt.
- Relativt feilfritt og feiltolerant, slik at brukere ikke gjør mange feil, og at disse feilene ikke er katastrofale (og at man lett kan ta seg inn igjen).
- Behagelig å bruke, tilfredsstiller brukerne subjektivt, slik at de liker å bruke systemet.

2.4.2 Interaksjonsdesign - IxD

Interaksjonsdesign er kjent som den fagdisiplinen som har i oppgave å designe digitale produkter, systemer og tjenester som mennesker kan samhandle med [Cooper et al., 2007]. IxD knyttes ofte til utformingen av grafiske grensesnitt for nettsider og applikasjoner, og er derfor også tett relatert til vitenskapen om menneske-maskin-interaksjon (MMI). Selv om hensikten bak interaksjonsdesign er å muliggjøre en god brukeropplevelse, finnes det ikke konsensus for hvilke kriterier som utgjør et godt interaksjonsdesign. Den danske informatikeren Jakob Nielsen publiserte i 1990 ti prinsipper for godt design, som fremdeles blir brukt den dag i dag.

Nielsens 10 heuristikker [Nielsen, 1990]:

- **Synlighet av systemstatus**: Systemet bør alltid holde brukeren oppdatert om hva som skjer gjennom passende tilbakemeldinger til riktig tid.
- **Likhet mellom systemet og den virkelige verden**: Systemet bør snakke brukerens språk, med termer og ord som brukeren forstår
- **Brukerkontroll- og frihet**: Brukere vil ofte gjøre feil og trenger å enkelt kunne komme tilbake til den tilstanden systemet var i før feilen ble gjort.

- **Konsekvens og standardisering**: Ikke endre betydningen av ord eller handlinger på tvers av kontekst. Følg konvensjoner.
- **Forebygge feil**: Design for å hindre brukeren i å gjøre feil
- **Gjenkjenning foran hukommelse**: Gjør systemet intuitivt å bruke. Synliggjør objekter og valgene brukeren har tilgjengelig til enhver tid så hen ikke trenger å huske i detalj hvordan prosesser foregår.
- **Fleksibilitet og effektivitet ved bruk**: Gjør det mulig for drevne brukere å foreta kjappe snarveier som er usynlige for den utrente bruker. For eksempel ved hjelp av tastekombinasjoner i stedet for klikking.
- **Estetisk og minimalistisk**: Fokuser på relevant informasjon og dialog. Jo flere elementer som legges til, jo mindre synlig blir hvert element.
- **Hjelp brukere til å gjenkjenne, diagnostisere og gjenopprette feil**: Feilmeldinger bør være presise og indikere hva problemet er og eventuelle løsningsforslag. Unngå kodespråk.
- **Hjelp og dokumentasjon**: Ha hjelp og dokumentasjon lett tilgjengelig selv om det optimale er at brukeren ikke får behov for det.

2.4.3 Brukergrensesnitt - UI

Brukergrensensitt er grensesnittet som lar en bruker kommunisere med maskiner [Rossen, 2020]. For datamaskiner delte man tidligere kun opp i *grafisk brukergrensesnitt* og *tekstlig brukergrensesnitt*, men det har i senere tid også blitt vanlig med *stemmestyrt brukergrensesnitt* og *haptisk brukergrensesnitt*. Målet ved design av brukegrensesnitt er å produsere et grensesnitt som gjør det enkelt og effektivt å betjene en maskin, med andre ord brukervennlig.

2.4.4 Brukeropplevelse - UX

Brukeropplevelse handler om hvordan en bruker kommuniserer og opplever et produkt og dens funksjon. Dette inkluderer brukerens oppfatninger, følelser og reaksjoner. Selv om brukeropplevelsen er subjektiv, er attributtene som legger til grunn for opplevelsen objektive. Sentralt i bruken av begrepet brukeropplevelse er å definere kravene eller kriterier for hva en god brukeropplevelse er. Ettersom fagbegrepet er bredt er det ikke like lett å formalisere et sett med kriterer, men det er blitt forsøkt gjort. Ofte blir en god brukeropplevelse knyttet til de internasjonale standardene for kvalitet ISO 8402 [Digitaliseringsdirektoratet, 2013][Nielsen Norman Group, 2021].

SINTEF på sin side bruker begrepene brukervennlighet (brukskvalitet), tillit, engasjement og estetikk for å forklare brukeropplevelse [SINTEF, 2007].

En av personene om har forsøkt å lage et sett med kriterier for god brukeropplevelse er Don Norman. I sin bok *The Design of Everyday Things* beskriver han 6 prinsipper for godt design [Norman, 2013]:

- **Synlighet (Visibility)**: Jo mer synlig funksjoner er, jo enklere vil det være for brukere å vite hva de skal gjøre.
- **Tilbakemelding (Feedback)**: Systemet bør si i fra om hvilken handling som har blitt utført og hva dette har resultert i. Dette kan gjøres visuelt, med lyd eller ved følelse.
- **Begrensninger (Constraints)**: I noen tilfeller kan det være nødvendig å begrense brukeren fra å utføre visse handlinger. Årsaken kan være mange, men i mange tilfeller gjelder det sikkerhet.
- **Overføring (Mapping)**: Sammenhengen mellom en kontroll - og dens funksjon. Nesten alle gjenstander trenger en form for overføring, enten det er lommelykter eller tv-fjernkontroller.

- **Å være konsistent (Consistency)**: Grensesnitt som bruker liknende design og liknende komponenter for å oppnå liknende resultater.
- **Tilbydelse (Affordance)**: Egenskap som gjør at en gjenstand kommuniserer hvordan den skal brukes. Et eksempel er et dørhåndtak på en dør, som signaliserer at døren skal trekkes i for å åpnes.

2.4.5 W3C, WCAG og UU

The World Wide Web Consortium - W3C

The World Wide Web Consortium er en internasjonal organisasjon der medlemsorganisasjoner, heltidsansatte og publikum jobber sammen for å utvikle webstandarder. Ledet av nettoppfinner og direktør Tim Berners-Lee og administrerende direktør Jeffrey Jaffe. W3Cs hovedmål er å utvikle protokoller og standarder for teknologien som brukes på World Wide Web[w3c, 2021].

Web Content Accessibility Guidelines - WCAG

En av undergruppene til W3C er Web Accessibility Initiative (WAI) som utarbeider retningslinjer for hvordan nettsider skal kodes og legges opp slik at de er best mulig tilrettelagt for personer med nedsatt funksjonsevne. Disse retningslinjene er samlet under Web Content Accessibility Guidelines (WCAG), og består av 12-13 retningslinjer samlet under 4 prinsipper[W3C, 2021]:

- Mulig å oppfatte
- Mulig å betjene
- Forståelig
- Robust

Universell utforming - UU

Universell utforming handler om å planlegge og utvikle produkter, omgivelser, programmer og tjenester slik at de kan brukes av så mange mennesker som mulig på en likeverdig måte. Hensikten er å legge til rette for like muligheter for samfunnsdeltagelse og minimere diskriminering på grunn av nedsatt funksjonsevne [Lid, 2020].

I Norge er universell utforming lovfestet innenfor flere sektorer, deriblant IKT. Den alminnelige plikten til universell utforming av IKT følger av diskriminerings- og tilgjengelighetsloven § 13, og gjelder etter en oppdatering i 2014 nå også for apper i tillegg til nettsider. For nettløsninger har Norge valgt standarden WCAG 2.0, for å sikre at grensesnittet gjør innholdet på nett tilgjengelig for flest mulig. Til tross for at WCAG er utviklet for web-innhold sier uutilsynet at:
Dersom en applikasjon formidler eller tilgjengeliggjør innhold fra nettet på den måten som er beskrevet i forskriften § 3 bokstav c, vil det være tale om en nettløsning i forskriftens forstand. I praksis antar vi at alle web-baserte applikasjoner vil måtte anses som en nettløsning [uutilsynet, 2014].

2.4.6 Menneske-maskin-interaksjon - MMI

Brukergrensesnitt (UI) er som sagt betegnelsen på kontaktflaten mellom brukeren og maskinen [Rossen, 2020]. Når dette grensesnittet brukes kalles det for menneske-maskin-interaksjon (MMI). MMI er derfor disiplinen om interaksjonen mellom mennesker og maskiner, som består av design, evaluering og implementasjon av maskinvare og programvare [SIGCHI, 2014].

2.5 Utviklingsmetodikk

2.5.1 Smidig/agil utvikling

Smidig eller agil utvikling er en generell betegnelse på metoder der utviklingsprosessen til et prosjekt gjennomføres ved hyppige del-leveranser. Hver del-leveransene er en iterativ prosess der man syklisk går gjennom de ulike utviklingsfasene. I Scrum kaller man disse syklusene for sprinter. Hver av disse del-leveransene leveres til kunden som et inkrement. Utviklerne og kunden har et tett samarbeid og før hver iterasjon vil begge parter diskutere og bestemme et del-mål. Målet med smidig utvikling er å øke kvaliteten på systemet som utvikles, og å reagere hurtig på endrede krav uten unødig dobbeltarbeid [Jaatun, 2019].

2.5.2 Scrum

Scrum er en smidig utviklingsmetodikk som er strukturert for å støtte kompleks produktutvikling [Scrum.org, 2020]. Scrum baserer seg på de fire verdiene fra det agile manifestet [Beck et al., 2001]:

- Personer og samspill fremfor prosesser og verktøy
- Programvare som virker fremfor omfattende dokumentasjon
- Samarbeid med kunden fremfor kontraktsforhandlinger
- Å reagere på endringer fremfor å følge en plan

Ved gjennomføring av Scrum-metodikken vil man behøve de tre rollene: produkteier, Scrum-master og utviklerne. Disse tre rollene er likestilte og skal sammen jobbe for å oppnå produktmålet. En produkteier er ansvarlig for å fastsette produktmålet, styre prioriteten til oppgavene for å løse produktmålet og bevisstgjøre alle parter om hva oppgavene går ut på. En Scrum-master har som rolle å veilede utviklerne og rollen skal sørge for å ivareta og gjennomføre Scrum-prinsippene. Utviklerne har som ansvar å fullføre oppgavene gitt av produkteier. De innehar som regel en bred og variert kunnskap for å oppnå produktmålet. Utviklerne organiserer arbeidet selv og skal holde hverandre ansvarlig for å oppnå hvert del-mål [Scrum.org, 2020].

Scrum består av tre de tre artefaktene: produktkø, sprintkø og inkrement. Disse representer arbeid og verdi for å skape transparens og muligheter for inspeksjon og adapsjon. Artefaktene gir informasjon til alle rollene for å sikre at man får levert et *ferdig* produktinkrement. En produktkø er en liste med funksjonaliteter og arbeidsoppgaver som er nødvendig for produktet som skal utvikles. Listen er sortert etter prioritert, og alt er bestemt av produkteier i samråd med utviklerne. Produktkøen vil kontinuerlig endres ved justeringer og tilføyelser av nye funksjonaliteter. En sprintkø er et bestemt utvalg av elementer fra produktkøen som skal implementeres for å oppnå målet for den bestemte sprinten. Et inkrement er summen av alle fullførte elementer fra sprintkøen i løpet av en sprint integrert med inkrementene fra alle tidligere sprinter [Scrum.org, 2020].

I Scrum finner man fem ulike aktiviteter som skal gi rutine og minimere unødvendige møter som ikke er definert i Scrum. De fem aktivitetene er: sprint, sprintplanlegging, daglig Scrum-møte, sprint-review og sprint-retrospektiv. Under en sprint følger man et sprintmål hvor man utvikler et brukbart og leveringsklart inkrement av produktet. Varigheten til en sprint varierer mellom en uke til en måned. I enhver sprint begynner man med en sprintplanlegging. Alle parter involveres der man legger til rette for sprintmålet, hva som skal leveres og hvordan det skal gjennomføres. Fra produktkøen vil et enkelt utvalg elementer som samsvarer med sprintmålet flyttes over til sprintkøen under sprintplanleggingen. Ved begynnelsen av hver arbeidsdag vil utviklerne ha et Scrum-møte der man synkroniserer arbeidet sitt og planlegger arbeidet for dagen. Hver enkelt deltaker tar opp hva man har oppnådd, hva som skal gjøres til neste gang og eventuelle problemer man har møtt. Det er Scrum-master sitt ansvar for å organisere møtet, men utviklerne har ansvaret for å utføre møtet.

Etter hver sprint avholdes det en sprint-review der man gjennomgår inkrementet som er levert. Her er det mulighet for alle parter og andre interessenter til å delta for å gi tilbakemeldinger. Ved behov er det også mulig å endre produktkøen. Etter gjennomgangen vil utviklerne gjennomføre et sprint-retrospekt. Dette er et tilbakeblikk på hva som ble utført under sprinten. Hver utvikler deler sitt synspunkt på hva som gikk bra, hva som gikk dårlig og tiltak for forbedringer. Utviklerne forplikter seg til å implementere de diskuterte tiltakene. Som et resultat av dette vil hver sprint forbedres og dette vil i teorien øke produktkvaliteten [Scrum.org, 2020].

3 Valg av teknologi og metode

I dette kapittelet vil vi ta for oss de ulike teknologiene og metoden som er brukt i prosjektet. Vi vil begrunne valgene som er gjort relatert til problemstillingen, og til slutt skrive om arbeids-og rollefordelingen.

3.1 Videreføring av høstprosjekt til bacheloroppgave

Fra høsten 2020 i emnet TDAT3022 Systemutviklingsprosjekt videreførte vi en rekke teknologier som ble brukt i løpet av prosjektet. Her vil vi forklare hvorfor vi valgte å videreføre disse teknologiene.

3.1.1 Videreførte valg av teknologi - Klient

Vi valgte å bygge videre på klientapplikasjonen vår fra høstprosjektet. Applikasjonen ble riktignok restrukturert, men teknologien var hovedsaklig den samme. De neste delkapitlene vil oppsummere våre valg av teknologi for klientapplikasjonen.

3.1.1.1 React Native

Som nevnt i teoridelen kan man bruke KP-rammeverk (kryss-plattform) for å utvikle mobilapplikasjoner for flere operativsystemer med én kodebase. React Native er et slikt KP-rammeverk og har gjort det mulig og raskt for oss å utvikle en mobilapplikasjon til både Android- og iOS-enheter. Koden skrives i JavaScript og kompileres ved bygging til operativsystemets foretrukne språk. Vi valgte å viderføre bruken av dette rammerverket da vi var godt kjent med det, og både teamet og oppdragsgiver var svært fornøyde med løsningen.

3.1.1.2 Expo

Expo er et rammeverk laget for å forenkle utviklingen- og byggingen av applikasjonen. Expo tilbyr ulike verktøy og servicer for utvikling, bygging, distribuering og testing. På lik linje med React Native tilbyr Expo integrasjon mot iOS, Android og webapplikasjoner fra én og samme JavaScript-kodebase. Under utviklingen av applikasjonen har Expos funksjonalitet latt oss teste koden gjennom ulike mobilenheter i sanntid, og dermed økt effektiviteten av utviklingen [Expo, 2020].

3.1.2 Videreførte valg av teknologi - Tjener

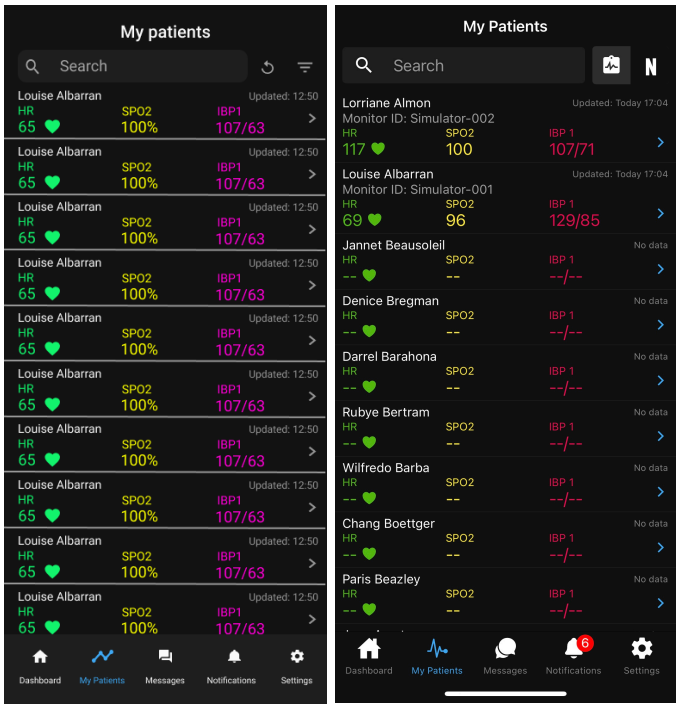
Til dette prosjektet visste vi at vi kom til å måtte gjøre mye mer utvikling i tjenerkoden enn vi hadde gjort i høstprosjektet for å tilrettelegge for alle de nye funksjonene. Vi valgte naturlig nok å videreføre bruken av Django og PostgreSQL ettersom vi ville bygge videre på Infiniwell sin kode. Vi investerte en del tid i starten av prosjektet på å lære oss disse nye teknologiene. Deriblant hadde vi en workshop med Infiniwell slik at vi ble godt innført i kodebasen deres og fikk satt opp lokale tjenere på maskinene våre.

3.2 Wireframes - Figma

For og tidligst mulig komme til enighet med oppdragsgiver om hvordan produktet vårt skulle se ut, valgte vi å lage en prototype av produktet ved å utvikle wireframes. Dette er ikke-funksjonelle skisser av hvordan de forskjellige skjermene skal se ut. Disse gjorde at teamet og oppdragsgiver fikk en felles forståelse av appen, og at vi kunne unngå å måtte gjøre store endringer sent i utviklingsprosessen.

For å lage disse skjermbildene brukte vi Figma, et web-verktøy som lot oss lage skisser samtidig, fra hvor som helst. Fordelen med å bruke Figma er at man lager en detaljert prototype av hvordan appen kan se ut. Dette gjøres ved å gi knapper og liknende komponenter funksjoner. Figma støtter

ulike biblioteker som inneholder kjente komponenter, og dette resulterer i at designprosessen blir enklere. I tillegg tilbyr Figma muligheten til å ilegge funksjonalitet til ulike komponenter som lar deg navigere deg rundt i prototypen slik det er ment i det ferdige produktet.



Prototypen fra Figma Det endelige produktet

Figure 3.1: Forskjellen mellom prototype og endelig produkt

3.3 Versjonskontrollsystem - Git og GitLab

Git ble brukt som versjonskontroll sammen med GitLab for lagring og samtidig samhandling av kode. Kildekoden for klienten ble holdt separert fra kildekoden til tjeneren da Infiniwell allerede hadde en eksisterende kodebase for tjeneren.

3.4 Valg av teknologi - Klient

3.4.1 React Native Paper

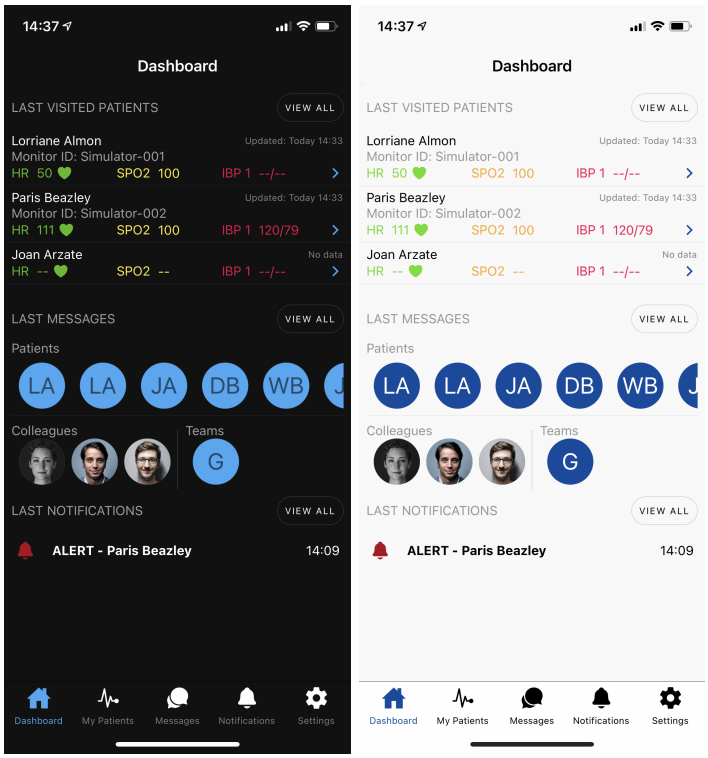
React Native Paper er en pakke for React Native som tilgjengeliggjør responsive og produksjonsklare komponenter som følger Googles Material Design sine retningslinjer. På denne måten sikrer vi at komponentene vi bruker følger WCAG 2.0 nivå AA blant annet når det gjelder fargekontraster [Google, 2020]. I applikasjonen vår brukte vi disse komponentene for å bruke ett standard fargetema, og for å bruke komponenter brukere er vant til å se andre steder. Bruk av React Native Paper vil i følge delkapittel 2.4.1 oppnå god brukskvalitet på grunn av pakkens gjenkjennelighet og gode lærbarhet. Samtidig følger den viktige design-retningslinjer, blant annet Don Normans designprinsipper, som er beskrevet i delkapittel 2.4.4.

3.4.1.1 Farger

React Native Paper lot oss også tilpasse fargetema for hele appen. Dette gjorde at vi kunne implementere lyst og mørkt tema. Dette er noe som har blitt vanligere i utvikling av mobilapplikasjoner og operativsystem, og gjør at brukeren kan velge etter egne preferanser. Vårt mørke tema reduserer lysstyrken samtidig som vi oppfyller minimumkrav for kontrastforhold. Ved å bruke mørkegrå kontra helsvart bakgrunn på det mørke temaet skaper vi muligheter for større dybde, i tillegg til at vi forhindrer forsinkelse på mobile enheter som bruker OLED skjerm. Ettersom OLED skjermer kan skru av skjermen der det er sorte piksler, fører dette ofte til at tekst og bilder blir utydelige når man blar siden pikslene skrur seg av og på istedenfor å endre farge [Google, 2020].

Temaene gir en uniform fargepalett gjennom hele applikasjonen. Fargene vi bruker i appen er hovedsakelig fra et dokument vi mottok fra oppdragsgiver som beskrev selskapets grafiske profil. Dette dokumentet finner du i *Vedlegg D: Systemdokumentasjon, Infiniwell Baseline Design Guidelines*. Fargepaletten er basert på Material Design sin standardpalett, i tillegg til at det er tatt hensyn til korrekte kontraster og metning på elementer og tekst.

Fargene på pasientmonitoren er svært nøye valgt ut med tanke på at monitoren skal likne pasientmonitorene helsepersonell er vant til å se på sykehus (ref. delkapittel 2.2.1). Her har vi også fulgt oppdragsgivers ønsker om å bruke samme farger som web-løsningen bruker. Dette skal også gjøre det enklere for brukerne å bytte mellom web-løsningen og appen. Pasientmonitorer har alltid sort bakgrunn og i appen er monitoren det eneste elementet av brukergrensesnittet som ikke blir påvirket av brukerens valg av fargetema.



Mørkt tema Lyst tema

Figure 3.2: Forskjellige fargetemaer i appen

3.4.1.2 Gjenkjennbare komponenter

En del av komponentene fra React Native Paper er knapper og brytere, samt andre komponenter som lar brukeren interagere med applikasjonen. Fordelen med disse knappene kontra hjemmelagde er at de bruker Googles Material Design-retningslinjer, men også at de likner knapper og brytere som brukes på Android og blant annet Googles egne sider og applikasjoner. Dette gjør at brukere raskt skjønner deres funksjon og hvordan man bruker komponenten. I tillegg til gjenkjennbare knapper og brytere gir React Native Paper oss tilgang til lister, tekstfelt, søkefelt, ikoner, hjelpetekst, pop-up-vinduer og andre navigasjons- og input-komponenter.

3.4.1.3 Hjelp-knapper

Ved hjelp av React Native Bottom Sheet, som er anbefalt av React Native Paper, har vi implementert flere "hjelp"-knapper i appen vår. Disse skal fungere som instruksjoner for de som måtte være i tvil om appens funksjonalitet. Knappene er strategisk plassert i nærheten av det som skal forklares, slik at brukeren enkelt skal forstå hva som forklares. Dette kan for eksempel være funksjonaliteten til en knapp eller et begrep i appen. Et eksempel på dette er forklaringen av NEWS som dukker opp om du trykker på info-knappen ved siden av NEWS i pasientoversikten.

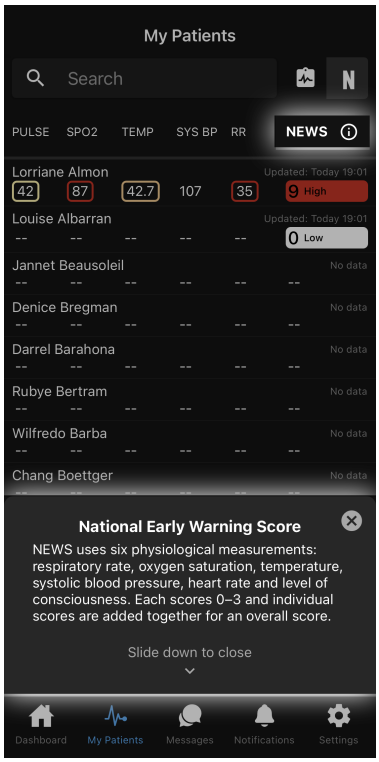
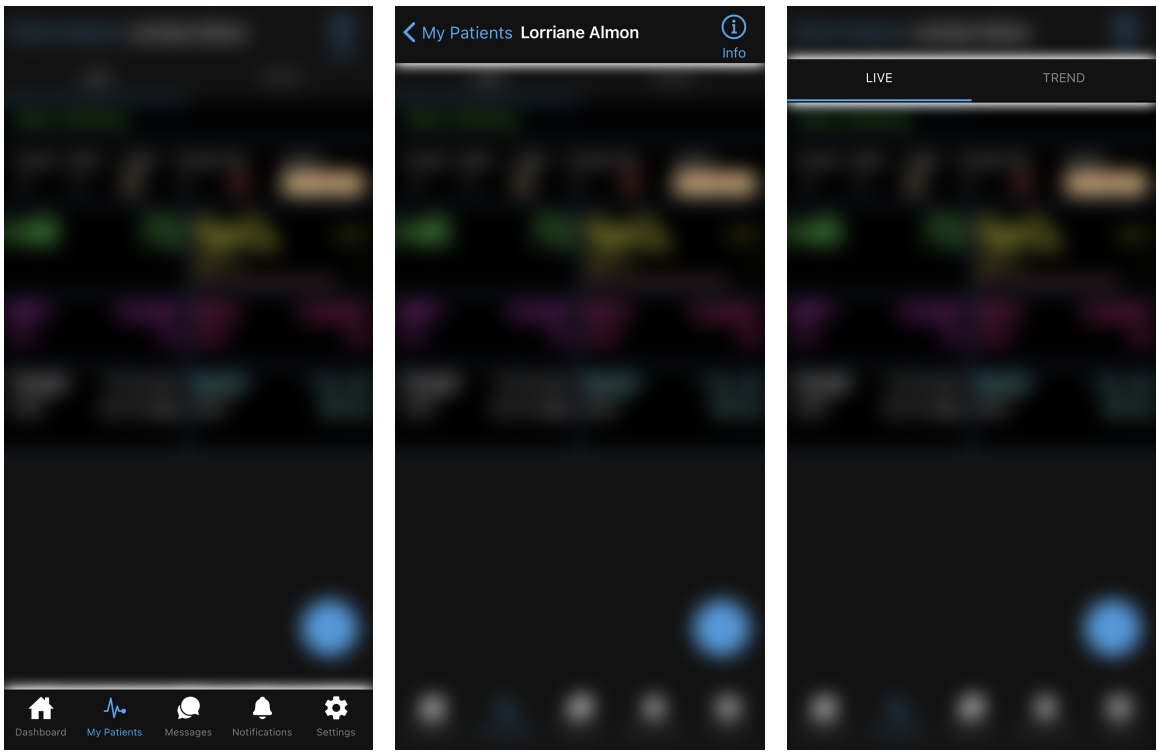


Figure 3.3: Pop-up som forklarer NEWS

3.4.2 React Navigation

For navigasjonen i appen, valgte vi å bruke React Navigation. Navigasjon er en viktig del av enhver mobilapplikasjon, og dens UI/UX. Vi har derfor valgt å bruke velkjente komponenter for dette. For navigasjon mellom enkle skjermer er det knapper som lar deg bytte skjerm, og en tilbakeknapp i headeren som tar deg tilbake til forrige side. Overordnet har vi en navigasjonsmeny på bunnen av skjermen som alltid er synlig, og som tar brukeren til de forskjellige delene av applikasjonen. På noen sider har vi også flere faner som lar brukeren bytte til ønsket fane. Ikke bare har React Navigation gitt oss navigasjonskomponenter som brukeren kjenner igjen visuelt (UI), men den tilbyr også en rekke interaksjoner som brukeren kjenner igjen fra mobilens eget operativsystem (UX). På iOS kan en bruker for eksempel sveipe fra venstre på skjermen for å gå tilbake til forrige side, og på Android kan brukere bruke mobilens egne tilbakeknapp. Sveiping kan også brukes for å navigere mellom faner.



Bottom Tab Navigator for overordnet navigasjon

Stack Navigator for navigasjon frem og tilbake

Tab Navigator lar brukeren bla eller trykke mellom flere faner

Figure 3.4: Forskjellige navigasjonskomponenter i appen

3.4.3 ESLint

Vi valgte å sette opp ESLint i prosjektet vårt. ESLint er et statisk kodeanalyseringsverkyøy. Dette betyr at vi får tilbakemeldinger på koden vi skriver, mens vi skriver den. Dette hjalp oss veldig underveis i prosjektet for å sikre at alle skriver god kode, og følger de samme reglene mens vi skriver kode. Eksempler på tilbakemeldinger ESLint kan gi er bruk av *const* i stedet for *var* de stedene en variabel ikke endrer seg. ESLint forteller oss også om ubrukte variabler og stiler for React komponenter, samt forslag til forkortelser av kode. Alt i alt gjorde ESLint at koden vår ble kortere, renere, og lettere å lese for alle på teamet.

3.4.4 Prettier

Vi brukte Prettier for å formatere kode i prosjektet vårt. Prettier gjør at man kan sette regler for blant annet om man skal bruke to eller fire mellomrom for innrykk, og hvor lange linjer koden maksimalt skal ha. I tillegg kommer Prettier med mange standardregler som gjør at koden automatisk blir fordelt over flere linjer dersom det øker leseligheten. Dette har vært veldig nyttig fordi all kode er formatert likt for alle medlemmer i teamet, og vi har derfor unngått konflikter ved sammenslåing av kode i Git.

3.4.5 Testing av klient - JEST

Når man skriver frontend-tester finnes det flere varianter av tester man kan skrive for å teste koden på forskjellige nivåer. Rammeverket JEST er en del av React Native applikasjoner og lot oss skrive forskjellige tester. Tester som er vanlige å gjennomføre for en klient kan være enhetstesting, der man tester små deler av koden eksempelvis individuelle funksjoner eller klasser. Mocking brukes dersom man er avhengig av eksterne leverandører for data. Da lager man sin egen variant av den eksterne løsningen for å simulere den ordentlige løsningen. På denne måten får man fokuset bort fra den eksterne leverandøren og sentrerer testen rundt egen kode. Har man større systemer er det også vanlig å ha integrasjonstester, der man tester samspillet mellom de ulike komponentene.

Snapshot testing lot oss teste rendering av forskjellige sider. Denne fungerer ved at testkoden rendrer siden det er snakk om, lager en snapshot av denne, og sammenlikner denne med en referansesnapshot lagret fra før av. En snapshot er et generert skjermbilde av skjermen som skal vises.

Testing av funksjoner lot oss sjekke at vi fikk det ønskede resultatet av forskjellige funksjoner. Funksjoner som ble testet var blant annet funksjoner for sortering og for å sjekke om trendgrafer var tomme. Her sendte vi inn test-data, og matchet resultatet fra funksjonen med et forventet resultat.

3.5 Valg av teknologi - Tjener

På tjenersiden hadde Infiniwell allerede gjort et par valg av teknologier på forhånd som teamet måtte sette seg inn i. Disse ulike teknologiene er beskrevet under.

3.5.1 Django

Django er et Python web-rammeverk som bidrar med rask og enkel utvikling av nettsider. Rammeverket skal ta seg av alt det komplekse med webutvikling, og lar utvikleren fokusere på produktet. Tjenesten er gratis og har en åpen kildekode. Rammeverket skal tilby utvikleren høy fleksibilitet. Det vil si at Django fungerer med alle typer klient-rammeverk og kan levere innhold i formater som HTML, JSON og XML. Det skal også være et sikkert rammeverk ved å advare utvikleren om enkle sikkerhetshull. Dette kan for eksempel være lagring av sesjonsid i informasjonskapsler eller direkte lagring av ikke-krypterte passord. Django har også viktige funksjoner som forebygger trusler som SQL-injeksjoner og Cross-Site-Scripting [Docs, 2021].

3.5.2 PostgreSQL

PostgreSQL er et databasesystem som er objekt-relasjonelt og som tar i bruk og utvider SQL-språket. De nye utvidelsene inneholder funksjonaliteter som er med på å skalere og sikkert lagre store datamengder [The PostgreSQL Global Development Group, 2021]. Django har et innebygd Object-Relational Mapper-system (ORM) som tar seg av oppretting av tabeller og håndtering av data i databasen. Infiniwells hoveddatabase benytter seg av Djangos ORM for databasetilgang, men de benytter seg også av andre databaser som kun bruker direkte SQL-kall. For vårt prosjekt benyttet vi både Django ORM og direkte SQL-kall for å håndtere data. Dette ble gjort fordi noen av kallene våre var for kompliserte for ORMen og vi tok i bruk de databasene som ikke var laget av ORMen.

3.5.3 REpresentational State Transfer

For kommunikasjon mellom klienten og tjeneren ble REST brukt som arkitektur for legge til, hente og endre data. Dette er nærmere beskrevet i delkapittel 2.3.4 om REST.

3.5.4 Swagger API

Swagger er et Interface Descriptive Language-IDL som brukes for å beskrive RESTful API ved bruk av JSON. Ved hjelp av Swagger har man et eget brukergrensesnitt for de ulike endepunktene man har laget i tjeneren. Endepunktene kan testes og brukes til å se, fjerne, endre og legge til data i databasen uten å være avhengig av klienten.

3.5.5 GitLab CI/CD og testing av tjener

GitLab har integrert støtte for CI og CD. Infiniwell sin tjener-kodebase har omfattende testing som kjøres automatisk gjennom GitLab CI. Om disse testene består, kjører den CD-kode som oppdaterer koden på tjeneren til den nyeste versjonen. CI/CD kjøres hver gang man dytter ny kode til GitLab. For alle endringene vi gjorde i koden sørget vi for å legge til tester som sjekket at den nye koden fungerte slik som forventet ved fremtidige endringer. Rammeverket vi brukte for testing er den som er innebygd i Django. Dette ble brukt fordi det allerede var i bruk av Infiniwell og det integreres sømløst med resten av Django-koden. Testing hjalp oss med å oppdage feil som oppstod andre steder i koden etter endringer. CD hjalp oss med å hyppig oppdatere tjener-koden vår for å ta i bruk ny versjon. Django har i tillegg et migrering system som overfører databaseendringer til databasen automatisk i CD.

3.6 Digitale samhandlingsverktøy

Til tross for Covid-19-pandemien jobbet teamet fysisk sammen. For våre ukentlige oppdateringsmøter med oppdragsgiver og fortløpende spørsmål vi hadde, benyttet vi Slack ettersom dette var oppdragsgivers preferanse. For felles sprint-møter med oppdragsgiver og veileder var vi innom forskjellige plattformer, men endte opp med å bruke Google Meet.

3.6.1 Slack

Slack er en kommunikasjonsplattform som vi brukte for kommunikasjon med oppdragsgiver. Der fikk vi tilgang til deres kanal og hadde digitale møter og diskusjoner gjennom denne. Her var det rask tilgang til dokumenter, slik at man enkelt kunne ha møteinnkalling og tidligere referater åpne ved siden av møtet. Dette ble brukt da Infiniwell bruker plattformen internt.

3.6.2 Microsoft Teams

Microsoft Teams er en kommunikasjonsplattform som vi brukte for kommunikasjon med veileder, da dette var veileders preferanse. Møter med bare veileder gikk gjennom denne plattformen.

3.6.3 Google Disk

Google Disk er en skybasert tjeneste for fillagring og synkronisering. Ved hjelp av denne tjenesten hadde alle teammedlemmer og veilder tilgang til diverse filer og dokumenter. Google Disk tilater også samskriving av dokumenter.

3.6.4 Google Meet

Google Meet er videokommunikasjonstjeneste fra Google som muliggjør videomøter gjennom nettleser. Her trenger man bare å opprette et rom og deretter dele ut en invitasjonslenke til de man ønsker å invitere. Dette ble brukt for å holde møter som gjaldt både oppdragsgiver og veileder.

3.6.5 Overleaf

Overleaf ble benyttet for å skrive dette dokumentet ettersom det tilater samskriving i Latex.

3.6.6 GitLab

GitLab ble brukt for å dele kode mellom teammedlemmene da det tillater CI/CD. I tillegg uttrykket Infiniwell et ønske om at koden til appen skulle ligge på samme plattform som deres tjener-kode.

3.6.7 Trello

Trello ble brukt som et digitalt Scrum-board. Vi valgte Trello ettersom det er en brukervennlig løsning som tilater oss å fargekode, kommentere og sette delmål til kortene våre. Det gir oss også muligheten til å se hvem som jobber med hva.

3.7 Valg av utviklingsmetodikk

3.7.1 Scrum

Fra høsten 2020 i emnet TDAT3022 Systemutviklingsprosjekt valgte vi en utviklingsmetodikk som liknet Scrum. Dette var både vi og oppdragsgiver fornøyde med, men vi ønsket å utføre bachelorprosjektet med full utnyttelse av Scrum-metodikken. Dette var fordi vi ønsket hyppig dialog med oppdragsgiver for å forsikre oss om at vi fulgte oppdragsgivers visjon av produktet. Vi visste at Infiniwell sin løsning var i utvikling samtidig som vi utviklet og vi ville være tilpasningsdyktige. Dette kunne medføre kodeendringer eller endringer av krav rettet mot prosjektet. Som nevnt i delkapittel 2.5.2 om Scrum, egner metodikken seg godt når kravene endres underveis i et prosjekt. Vi ønsket følge verdiene som ligger til grunn i Scrum-utvikling: å reagere på endringer fremfor å følge en plan [Beck et al., 2001]. Det ble holdt daglige og fysiske Scrum-møter innad i teamet fire dager i uka. Sprint-review og sprint-retrospektiv ble holdt digitalt med oppdragsgiver og veileder ved endt sprint.

3.7.2 Fordeling av arbeid og roller

I begynnelsen ønsket ikke teamet å tilegne hvert medlem en spesifikk rolle. Vi ønsket å fordele arbeidet likt og rettferdig slik at arbeidsmengden innenfor hvert arbeidsområde ble jevnt for alle. Resultatet ble å signere en arbeidskontrakt innad i teamet som er tilgjengelig i *Vedlegg F: Kontrakter, Arbeidskontrakt*. Etter at alle fikk prøvd seg på alle områder, fant vi raskt ut at det var enklere å dele teamet i to. To medlemmer spesialiserte seg på klientsiden og to medlemmer spesialiserte seg på tjenersiden. Ved ønske gjorde vi rulleringer, men vi fant fort ut at medlemmene i teamet jobbet bedre når de kunne sette seg dypere inn i arbeidet sitt kontra å få nye arbeidsoppgaver på nye arbeidsområder hver dag. Dette økte også effektiviteten i teamet da alle medlemmer visste hvilke ekspertiser de forskjellige medlemmene hadde og dermed kunne henvende seg til riktig person for hjelp.

4 Resultater

4.1 Vitenskapelige resultater

Dette delkapittelet vil beskrive de vitenskapelige resultatene fra prosjektet. Resultatene er underlag til svar på problemstillingen og er basert på relevant teori og erfaringer gjort.

4.1.1 Brukergrensesnittet i applikasjonen

Brukergrensesnittet i en app er alle elementene brukeren kan se og interagere med. Dette inkluderer sidestrukturen, navigasjonsvalg, pop-up bannere, menyer, gester og alt av knapper og tekst. For å produsere et grensesnitt som er brukervennlig er det viktig å fokusere på elementene som skaper en god brukeropplevelse (ref. delkapittel 2.4.4). Don Norman, som er en av de mest profilerte personene innenfor fagområdet brukeropplevelse, beskriver i sin bok *The Design of Everyday Things* seks prinsipper for godt design som kan være avgjørende for å produsere et godt brukergrensesnitt.

For å opprettеholde Don Normans prinsipp om *å være konsistent* fokuserte vi på å beholde så mange elementer som mulig fra Infiniwells webløsning. Blant de viktigste og mest gjenkjennelige elementene fra webløsningen er pasientmonitoren og fargebruken i løsningen. I appen vår finner man selskapets grafiske profil, samtidig som appens fargetema er basert på samme design som nettsiden. For å skape et behagelig brukergrensesnitt for helsepersonell som jobber døgnet rundt, bestemte vi oss for å gi appen både et lyst og mørkt tema. Dette tillater brukeren selv å velge hvilken stil de foretrekker, enten det kun baseres på preferanse eller fordi de jobber i mørke omstendigheter på kveldsskift. Fargene for vitale-parametere ble nøye valgt ut for å representere hva helsepersonell er vant til. For de ulike fargetemaene var det viktig å skape gode nok kontraster mellom bakgrunnsfargen og de andre komponentene. Don Norman redegjør også for verdien av *synlighet* i brukergrensesnittet. Dette har vi sikret ved bruk av primær og sekundær farger i fargetemaet vårt. Knapper og andre elementer som symboliserer primærfunksjonaliteten i appen er uthevet med temaets primærfarge. Dette skaper et hierarki i appen som indikerer hvilke elementer som er viktigst, og hvilke man kan interagere med og ikke.

Ulike komponenter ble nøye plassert der brukeren er vant til å ha dem og i rekkevidde for hvordan man holder enheten sin. Ettersom 90% av verdens befolkning er høyrehendte er det normalt å plassere appens viktigste handlinger mot høyre side av appen, slik at de er lett tilgjengelig [Scharoun and Bryden, 2014]. På samme måte plasseres mindre hastende funksjoner og sekundær komponenter, slik som tilbake-knapp øverst til venstre og søkefelt på toppen av en side. Likevel har vi, som mange andre mobilapplikasjoner, tatt i bruk operativsystemspesifikke løsninger for å gjøre navigasjon så enkelt som mulig. På iOS-enheter kan man bruke den innebygde sveipingen for å gå tilbake, mens på Android kan man bruke den innebygde tilbakeknappen som finnes nederst i høyre hjørne.

I stedet for å lage egne komponenter, valgte vi å bruke gjenkjennbare komponenter fra React Native Paper som er beskrevet i delkapittel 3.4.1. Komponentene ble gjenbrukt flere ganger for å skape kontinuitet i gjennom appen. Desginet ble holdt minimalistisk, samtidig som vi ønsket å gi brukeren mest mulig informasjon dersom det lot seg gjøre uten at det skulle ødelegge brukeropplevelsen.

4.1.2 Brukeropplevelsen i applikasjonen

Det er vanskelig å definere brukeropplevelsen i applikasjonen, men det ble lagt et stort fokus på å implementere Nielsens 10 heuristikker fra delkapittel 2.4.2 om interaksjonsdesign for å sikre en god brukeropplevelse.

Ved å vektlegge synlighet av systemstatus økte brukeropplevelsen i appen. Dette innebærer å hjelpe brukere til å gjenkjenne, diagnostisere og gjenopprette feil. Dette oppfylte vi ved at hver brukerhan-

dling ga brukeren gode tilbakemeldinger fra systemet. Dette kunne være alt fra om et parameter ikke er tilgjengelig for visning til om et varsel har blitt sendt til andre helsepersonell. For de fleste brukerhandlinger ble det informert både når det gikk galt, men også når handlingen var vellykket.

Videre sikret vi en god brukeropplevelse ved å oppfylle heuristikken om brukerkontroll og frihet, samtidig som vi jobbet for å forebygge feil. Flere steder i appen har man mulighet til å gjøre ikke-reversible handlinger. For å forsikre brukerkontroll og forebygge feil la vi inn et ekstra steg i brukerhandlingen. Dette kunne for eksempel være å spørre om brukeren er sikker på å slette et notat eller en varsling. På denne måten sikret vi at brukeren ikke utførte en handling fordi brukeren trykket feil.

Et annet viktig heuristikk-punkt vi implementerte for å innfri god brukeropplevelse var muligheten for hjelp og dokumentasjon. Hjelpetekster og hjelpekomponenter ble utviklet for å forklare elementer om de skulle være uklare. Alle knapper har en beskrivelse og flere har også beskrivende ikoner. Noen av knappene er også koblet til en pop-up boks som inneholder en lengre hjelpetekst. Dette brukes dersom det er behov for en lengre forklaring av sider eller funksjonaliteter og er nevnt i delkapittel 3.4.1.3.

4.1.3 Nyttigheten av applikasjonen

Fra delkapittel 2.2.1 om pasientmonitoriering er det beskrevet at overvåking av pasienter som regel er stasjonært. Det vil si at man overvåker pasienter gjennom en arbeidsstasjon eller en overvåkningssentral. Ved bruk av appen vår, vil man utvide mulighetene for hvordan man overvåker en pasient. Først og fremst, er det mulig å overvåke en pasient fra hvor som helst og når som helst så lenge man har nettverksforbindelse og at pasienten er koblet til en monitor-enhet. Dette betyr at en travel sykepleier på et sykehus ikke trenger å være i umiddelbar nærhet av en pasient for overvåkning. Videre åpner appen for muligheten til å monitorere pasienter som er tilknyttet hjemmesykepleien.

Appen fungerer også som en kommunikasjonskanal. Her samles alle relevante helsepersonell i en organisasjon, og gir enhver bruker muligheten til å kommunisere med en kollega eller en gruppe med kolleger. Det er ikke noe tvil om at kommunikasjon er viktig på et sykehus, og at en løsning som gjør det mulig å kommunisere med alle kolleger innenfor din avdeling er nyttig. Dette belyser også en annen fordel som oppfyller et av behovene for helsepersonell, nemlig minimal bytting av systemer [St. Olavs, 2021]. Ved at vi har inkludert en meldingsfunksjonalitet i appen, slipper brukerne å bruke andre løsninger for å kommunisere med hverandre.

Ved å integrere systemet til Infiniwell med en mobilapplikasjon, vil man kunne utnytte funksjonaliteter fra en mobil enhet som ikke finnes på stasjonære arbeidsstasjoner. En av disse funksjonalitetene er push-notifikasjoner. Dette er knyttet opp mot meldingssystemet, men også ulike hendelser knyttet til en pasient. Brukeren av appen vil få push-notifikasjoner i sanntid ved nye journalinnlegg, eller dersom annet helsepersonell sender ut et varsel om at de trenger assistanse med en pasient. Appen er plattform-uavhengig, noe som gjør den tilgjengelig for de aller fleste. Dette kan være enheter som mobiler og nettbrett og appen fungerer sømløst på tvers av disse enhetene.

Applikasjonen rettet mot helsepersonell

Applikasjonen er rettet mot helsepersonell ved bruk relevante fagbegreper, samt kjente elementer og farger fra pasientmonitorer. På vår live-visning av pasientmonitoren har vi valgt å benytte de samme fargene som brukes på pasientmonitorene man finner på Infiniwells webløsning. Forkortelser for puls, blodtrykk, oksygenmetning er også de samme som på man finner på fysiske pasientmonitorer. Ellers er applikasjonen laget slik at man skal kunne komme seg rundt med færrest mulig trykk. Helsepersonell er ofte presset på tid og vi gjorde derfor alt lett tilgjengelig ved hjelp av forskjellige navigasjonskomponter. Dashbordet vårt er et eksempel på dette, der de viktigste elementene fra appen er samlet på ét sted.

I og med at testobjektene våre i stor grad var helsepersonell, fra tre forskjellig kontinenter, er også applikasjonen vår et resultat av deres tilbakemeldinger, og den er på denne måten rettet mot helsepersonell som målgruppe.

4.2 Ingeniørfaglie resultater

Dette delkapittelet beskriver de ingeniørfaglige resultatene fra prosjektet. Kapittelet tar for seg målene og kravene fra *Vedlegg A: Visjonsdokument* og beskriver status for hvert av disse.

4.2.1 Brukertester

I løpet av prosjektet ble det gjennomført to brukertester. Brukertestene ble utført fjernt av testobjektene selv. De fikk tilsendt testmateriell som innholdt blant annet en installasjonsmanual og en testinstruks.

Ettersom vi hadde en KP-løsning ga vi testobjektene muligheten til å laste ned applikasjonen vår gjennom Apple sin TestFlight app (iOS) eller gjennom Google Play Store (Android). Begge tjenestene tilbyr enkel nedlasting på samme måte som brukere er vant med å laste ned andre applikasjoner. Ved bruk av begge tjenestene kunne vi begrense hvem som skulle få tilgang til applikasjonen.

Ved første brukertest ble det utsendt en detaljert testinstrukjon, med spørsmål som skulle svares på både før, under og etter testen. Ved andre brukertest ble en ny testplan sendt ut med samme format som første brukertest. Tesplanen for andre brukertest ga brukerne friere tøyler til å teste appen. Brukertestene ble sendt ut til de samme testobjektene for begge tester. Testobjektene ble valgt ut av både oppdragsgiver og av teamet. Alle testene ligger vedlagt i *Vedlegg E: Tester, Brukertester.*

Vi ønsket å kartlegge brukernes tanker rundt tilnærmingen vår til bacheloroppgavens problemstilling: *brukskvalitet med fokus på brukergrensesnitt og brukeropplevelse.* For å få sammenlignbare resultater avsluttet vi testen ved å la brukerene rangere en rekke elementer i appen vår ved bruk av Likert-skala. Skalaen går fra 1-5 der 5 er best. Elementene som ble rangert er:

- **Navigasjon**
  Navigasjon i appen. Hvor lett det er å navigere seg til forskjellige skjermer i appen.
- **Lesbarhet**
  Tydelighet, skriftstørrelse og tekstfarge.
- **Farger**
  Fargebruk i appen.
- **Tilbakemeldinger**
  Tilbakemeldinger fra systemet, f.eks: ”Note has been created”
- **Beskrivelser**
  Hvor beskrivende er ikoner og tekst brukt i appen.
- **Gester**
  Zoom, sveip, tapping

Testobjektene

Testobjektene var de samme i begge testene og varierte i alder, nasjonalitet, kjønn, akademisk bakgrunn og yrke. Appen ble testet av 12 testobjekter der gjennomsnittsalderen var 40 år, med den yngste på 22 år og den eldste på 64 år. 42% av testobjektene hadde medisinsk utdanning. Utdanningene deres strakk seg fra sykepleien til medisinprofesjon til PhD innen medisin.

4.2.1.1 Første brukertest

Den første brukertesten ble sendt ut etter andre sprint. Målet i den første sprinten var å endre navigasjonsstrukturen i appen, samt lage et fungerende notifikasjonssystem som skulle stå for push-notifikasjoner. Den andre sprinten besto av å lage hele pasientjournal-delen av appen, med både nye lese-og-skriveløsninger opp mot Infiniwells eksisterende pasientjournal. Målet for den første brukertesten var å få svar på om vår løsning på disse sprintmålene falt i smak hos brukerne.

Under ser du en grafisk fremstilling av de seks spørsmålene vi ba brukerne våre rangere fra *misfornøyd (1)* til *fornøyd (5)*. Slik vi tolket resultatet hadde vi til en viss grad klart å utvikle god UI og UX. På tre av seks spørsmål svarte mer enn 50% av testobjektene at de var fornøyde med løsningen vår. Samtidig svarte noen på samme spørsmål at de var misfornøyde. Dette ga en indikasjon om at det eksisterte områder i appen som det var delte meninger om, og som etter vår mening måtte forbedres.

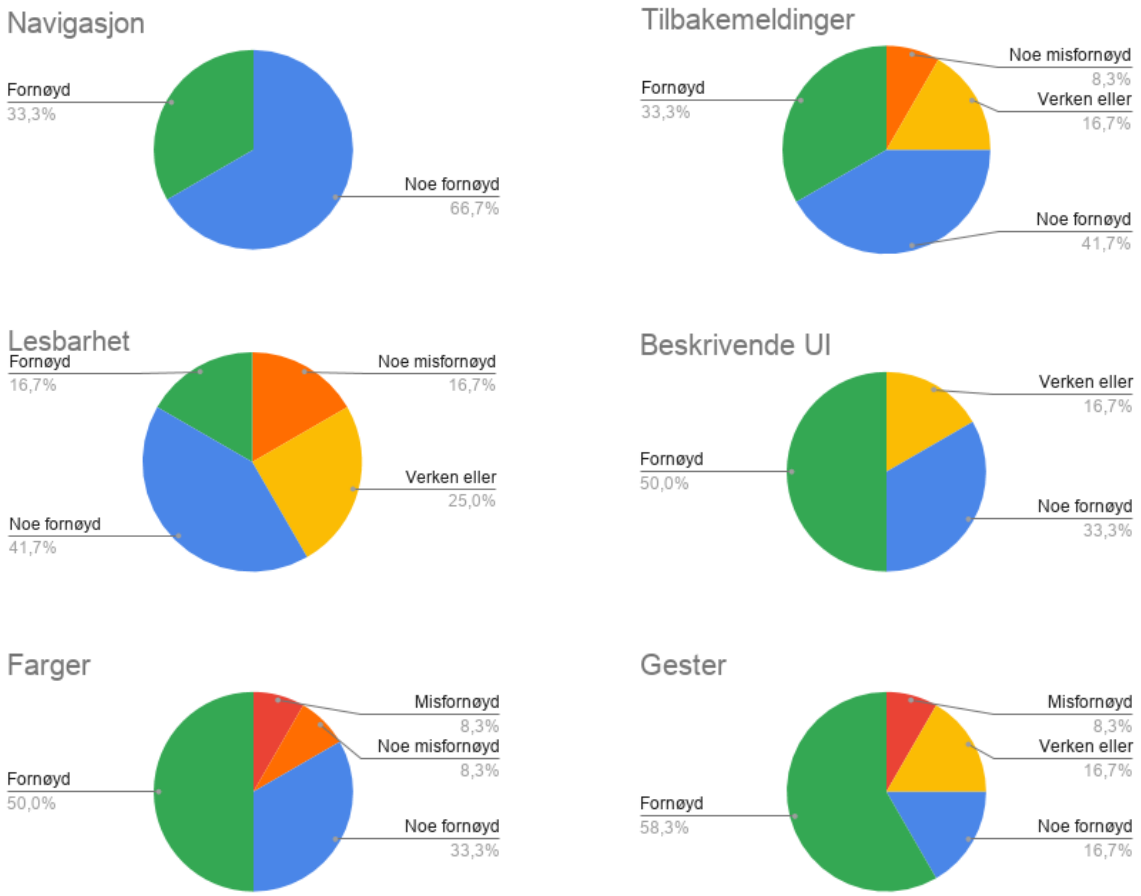


Figure 4.1: Statistikk fra første brukertest

Navigasjon og beskrivende UI var områdene som fikk best gjennomsnitt-score. På den andre siden var fargene i appen og bevegelser de områdene som fikk flest misfornøyde tilbakemeldinger, til tross for at bevegelser fikk den største andelen fornøyde. Konklusjonen ble umiddelbart at vi skulle endre fargebruken i appen, og gå for mer diffuse farger istedenfor sterke neon-lignende farger. Dette gjaldt ikke pasientmonitoren vel og merke. Når det gjaldt gestene planla vi å indikere dette på en bedre måte, i tillegg til å legge til muligheten for å utføre diverse handlinger uten å bruke gester. Eksempler

på dette er å legge til en "Delete all"-knapp i tillegg til muligheten for å slette en notifikasjon med sveip.

Flere av testebrukerne ønsket en forbedring også på trend-visningen. For det første, ønsket brukerne at grafene var litt høyere, slik at man tydeligere kunne se dataene som ble vist. For det andre ønsket brukerne en mulighet for å se grafene i fullskjerm eller landskapsmodus. På notifikasjonsonssiden mottok vi flere ønsker om en knapp som sletter alle notifikasjoner i notifikasjonsoversikten. Slik det var kunne man kun sveipe på en notifikasjon for å slette den. Brukerne likte dette fordi det er vanlig andre steder, men om man skal fjerne alle tar det for lang tid.

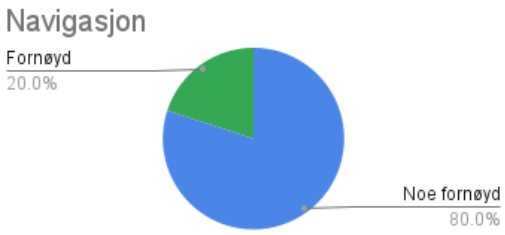
Av mindre endringer fikk vi tilbakemeldinger på at noen knapper ikke var beskrivende nok. Flere synes også at det var litt lite kontrast noen steder i appen hvis man hadde valgt lyst tema.

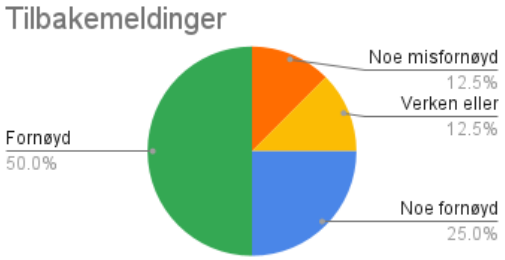
Ellers var brukerene fornøyde med pasientjournal og forstod seg på den. Dette var et av fokuspunktene under utviklingen som hadde foregått de tidligere ukene, så disse tilbakemeldingene var positive for teamet å få.

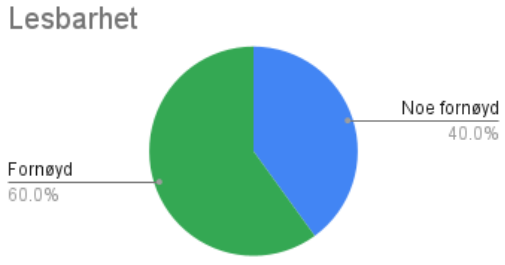
4.2.1.2 Andre brukertest

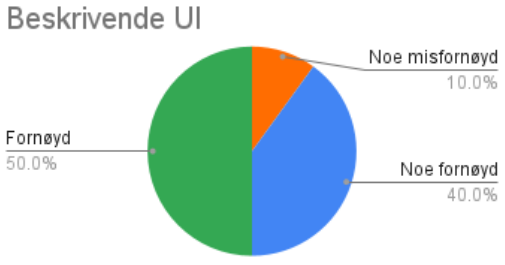
Den andre brukertesten ble sendt ut etter fjerde sprint. I denne testen ønsket vi å vise for testobjektene at tilbakemeldingene deres ble tatt på alvor og at vi hadde implementert flere av deres forslag. I tillegg hadde vi utvidet appen med nye løsninger som vi trengte tilbakemelding på. Målet for tredje sprint gikk ut på å implementere funksjonalitet for varslinger og ferdigstille meldingssystemet. Fra fjerde sprint lå fokuset på live-og-trendmonitorering av pasienter. Ønsket resultat for den andre brukertesten var derfor å få svar på om vår løsning på disse sprintmålene og endringer basert på forrige brukertest falt i smak hos brukerne.

Under ser du en grafisk fremstilling av de seks spørsmålene vi ba brukerne våre rangere fra *misfornøyd (1)* til *fornøyd (5)*. Kort fortalt sier grafene at appen har utviklet seg til det bedre på alle punktene utenom navigasjon og beskrivende brukergrensesnitt.









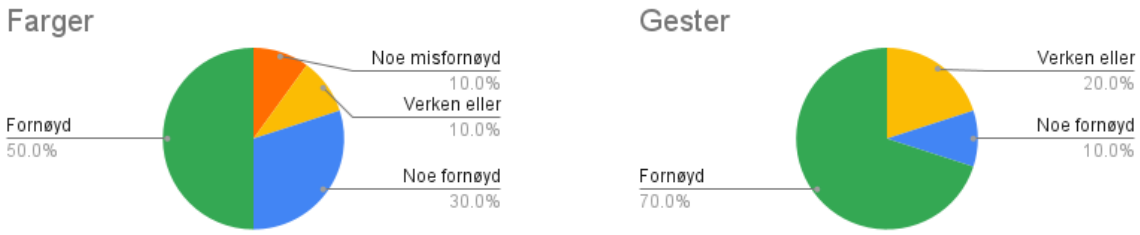


Figure 4.2: Statistikk fra andre brukertest

Statistikken for navigasjon viste at brukerene var mindre fornøyd til tross for at vi ikke hadde foretatt oss noen endringer med navigasjonen. Det samme gjelder statistikken for beskrivende UI, men her hadde vi gjort endringer siden første brukertest. Disse vi kommer vi tilbake til i diskusjonskapittelet.

På fire av de seks punktene så vi en forbedring ved at andelen fornøyde testbrukere økte samtidig som den var en minking av misfornøyde svar. I den andre brukertesten var det ingen som svarte at de var misfornøyde. Den største forbedringen var på punktet lesbarhet.

4.2.2 Akseptansetest

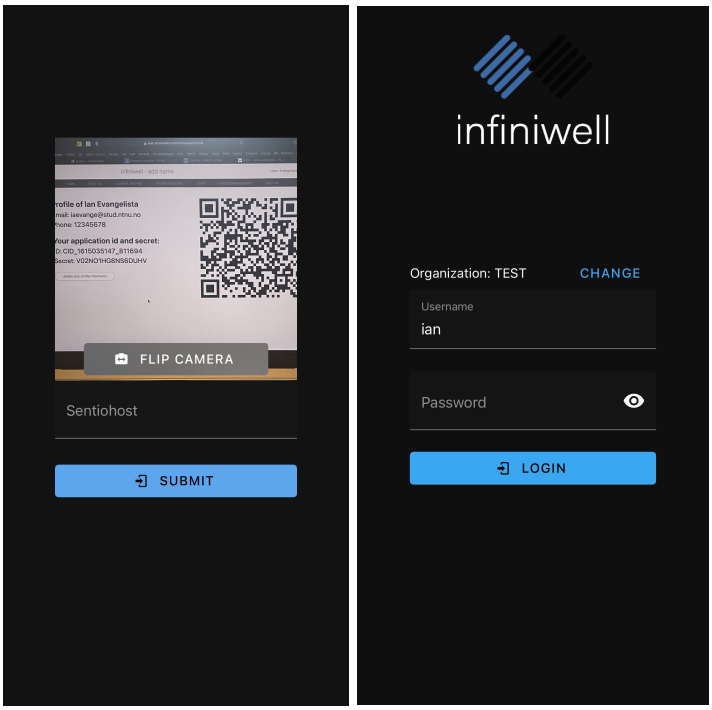
Som en avslutning for prosjektet utførte vi en akseptansetest sammen med oppdragsgiver. Testen ble gjennomført av produkteier i samarbeid med teamet på kontoret til Infiniwell AS. I denne testen ble kravene som er beskrevet i *Vedlegg A: Visjonsdokument, Produktets funksjonelle egenskaper* og *Vedlegg A: Visjonsdokument, Ikke-funksjonelle egenskaper og andre krav* gjennomgått for godkjenning. Dette gjaldt både funksjonelle- og ikke-funksjonelle krav. Funksjonelle krav er spesifikke funksjoner vi ønsker å tilby brukeren. Ikke-funksjonelle krav er ikke spesifikt knyttet til en funksjon, men som brukere og oppdragsgiver forventer av applikasjonen. De funksjonelle kravene fikk en grad av måloppnåelse fra 1 til 5 gitt av oppdragsgiver, der 1 er lav og 5 er høy. Noen av de funksjonelle kravene ble kategorisert fordi de gikk om hverandre.

4.2.2.1 Funksjonelle krav

Innlogging

For å logge inn, vil man kunne gjøre det gjennom en QR-kode eller ved å manuelt skrive inn alt av opplysninger. QR-koden er tilgjengelig på profilsiden i webløsningen. QR-koden vil fylle ut sentiohost, som er navnet på tjeneren man vil koble seg på, og brukernavnet til brukeren automatisk. Brukeren må selv skrive inn passordet sitt. Ved manuell innlogging, må man skrive inn sentiohost og deretter fylle inn brukernavn og passord.

Grad av måloppnåelse: 5

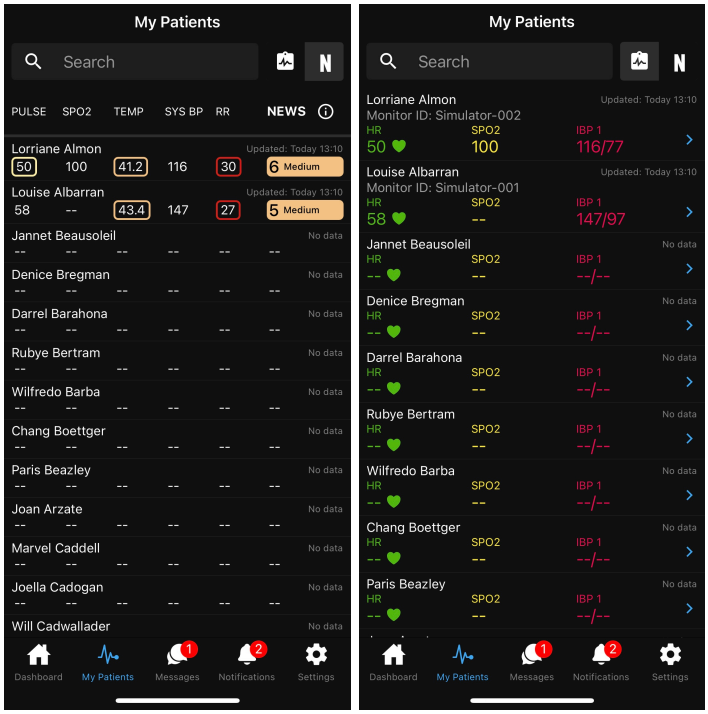


QR-skanning Innlogging etter QR-skanning

Figure 4.3: Innlogging

Pasientoversikt

Brukeren skal kunne se en overordnet oversikt over alle pasientene sine med de viktigste vitale tegnene. Utover prosjektet ble det utvidet til å inkludere NEWS2. Data blir automatisk oppdatert hvert halvminutt.
Grad av måloppnåelse: 5



Pasientoversikt - NEWS2 Pasientoversikt - General

Figure 4.4: Pasientoversikt

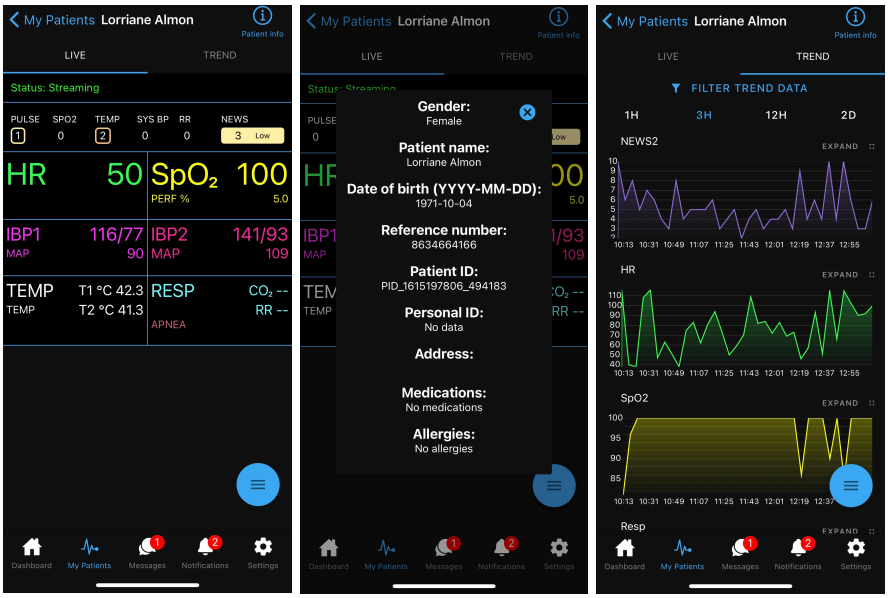
Pasientmonitorering

Innenfor pasientmonitorering vil man finne kravene:

- Se oversikt over en pasients informasjon
- Se detaljert informasjon om en pasients vitale tegn
- Livedata av en pasients status
- Vise trend over tid for pasientens status

For pasientmonitorering finner man ulike skjermer man kan analysere en pasient med. Først og fremst kan man enkelt finne frem til personlige opplysninger om en pasient etter å ha trykket seg inn på en. Dette vises tydelig i høyre hjørnet med et beskrivende ikon og en tilhørende hjelpetekst. På samme side finner man live-parametere som strømmes fra en websocket som er tilkoblet til monitoren og pasienten. Man kan enkelt bytte skjerm til trend-skjermen ved å sveipe seg til venstre eller trykke på trend på toppen. Her kan man velge hva slags trend-parametere man vil se ved å filtrere de man ønsker. Siden er en oversikt over valgte parametere. Man kan trykke seg inn på en parameter og enheten vil rotere til landskapsmodus for å gi et bedre bilde på variasjonen av parameteren.

Grad av måloppnåelse: 5



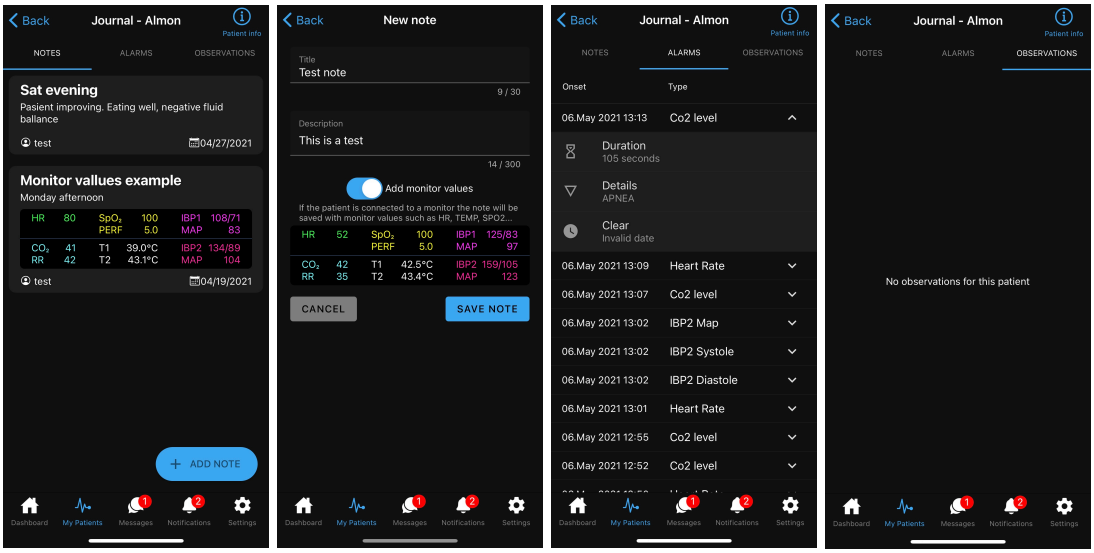
Live Pasientinformasjon Trend

Figure 4.5: Pasientmonitorering

Pasientjournal

I en pasientjournal har man mulighet til å se, legge til, slette og endre notater. Man har bare mulighet til å endre og slette notater man selv har laget. I tilegg vil man se ulike alarmer som har blitt utløst og observasjoner gjort av andre helsepersonell eller systemet.

Grad av måloppnåelse: 4



Notater Legge til notat Alarmer Observasjoner

Figure 4.6: Pasientjournal

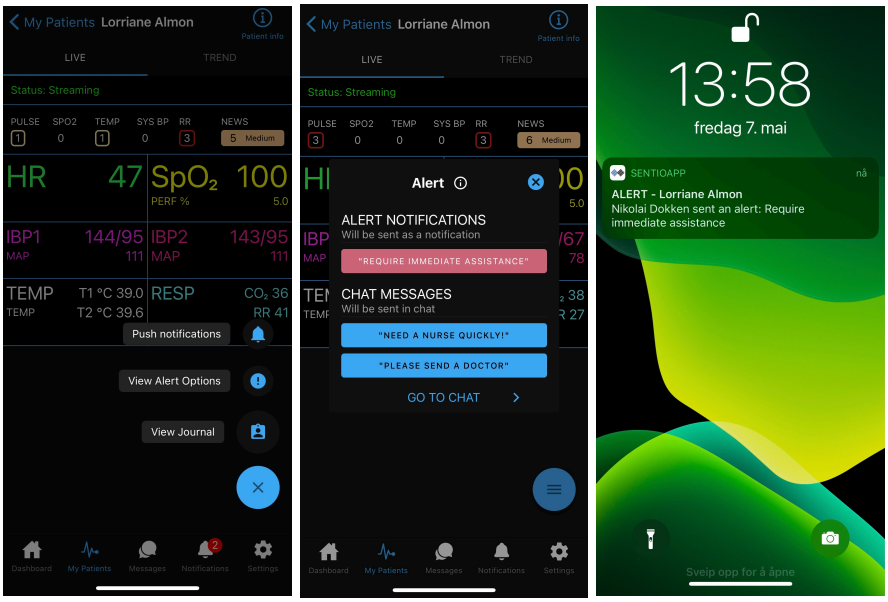
Varslinger

Innenfor varslinger vil man finne kravene:

- Sende varsel til en pasients tilknyttet helsepersonell om at noe er galt.
- Motta varslinger om pasienter i nød
- Skreddersy varslene sine - kunne velge hva man vil ha av varsler fra en pasient

Brukeren har mulighet til å sende en push-notifikasjon om at man trenger umiddelbar assistanse på grunn av en pasient. Dette blir sendt ut til andre helsepersonell som er knyttet til samme pasient. Brukeren vil også få push-notifikasjoner når det kommer nye notater inn i en pasientjournal. Man kan skru av push-notifikasjoner tilknyttet en pasient, men push-notifikasjonen som omhandler umiddelbar assistanse kan aldri skrus av.

Grad av måloppnåelse: 5



Pasient-meny Sende varsling Motta varsling

Figure 4.7: Varslinger

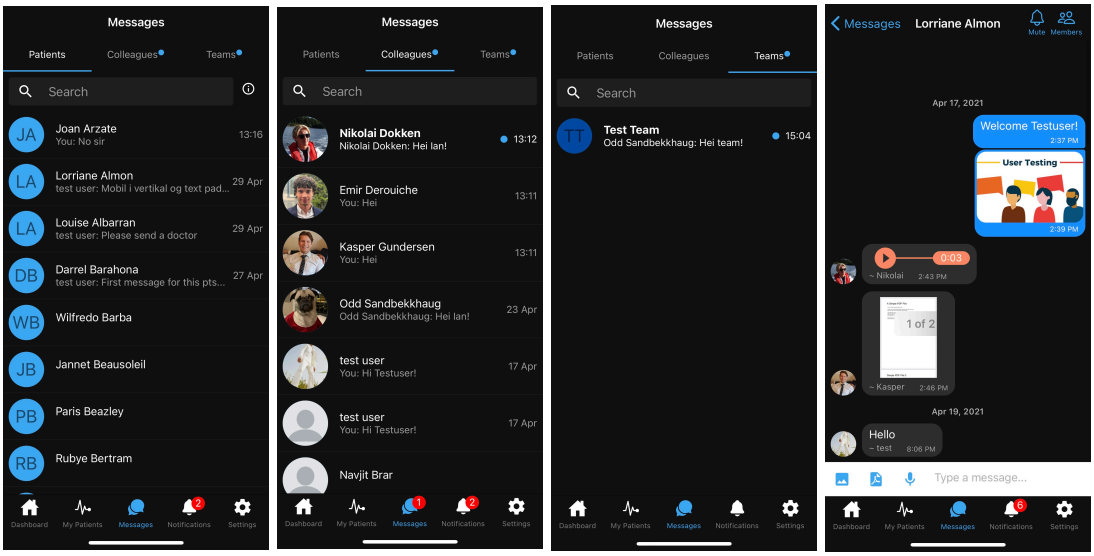
Chat

Innenfor chat vil man finne kravene:

- Egen chat for ansatte
- Egen chat team
- Egen chat om en pasient

En bruker kan sende meldinger til en bestemt kollega, et team man er med i og til andre kolleger som er tilknyttet samme pasient som seg selv. I hver chat kan man sende bilder og pdf-er, og spille av og ta opp lydopptak. I en team-og-pasient-chat kan man se hvilke kolleger som er med i chatten.

Grad av måloppnåelse: 5



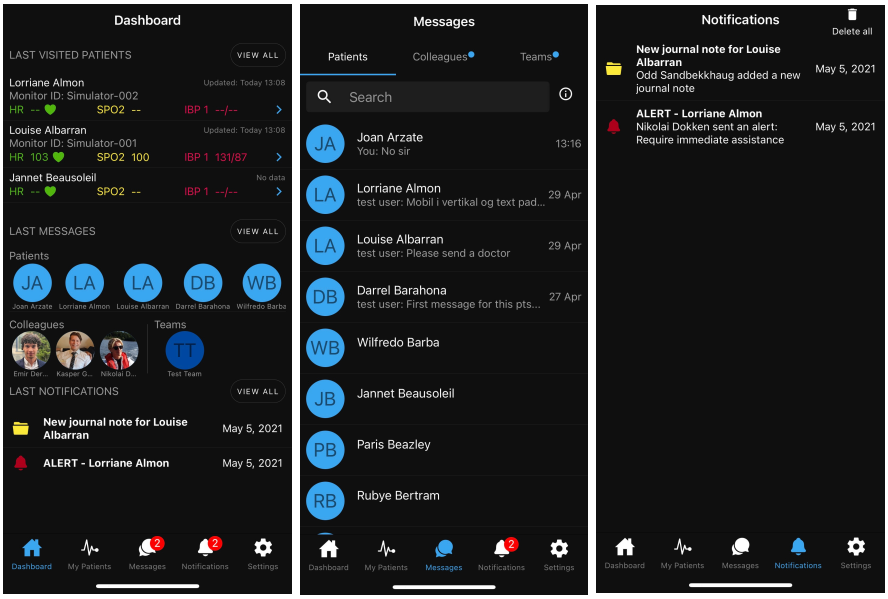
Pasient-meldinger Kollega-meldinger Team-meldinger Chat

Figure 4.8: Chat

Badges

Hver gang en bruker får en ny melding eller en ny notifikasjon, vil man få røde tall i navigasjonsbaren. Hvert tall vil gå bort når man går inn på tallets respektive side enten om det er meldingssiden eller notifikasjonssiden. På meldingssiden vil man også få en indikasjon på hvilke chatter som er lest. Dersom en chat er ulest, vil det markeres med en blå prikk på chatten. I tillegg vil chattens type i fanen: pasient-chat, kollega-chat eller team-chat få en blå prikk som indikerer at en chat innenfor den typen er ulest.

Grad av måloppnåelse: 5



Navigasjonsbar Meldinger Notifikasjoner

Figure 4.9: Badges

Dashboard

På dashboardet finner man de siste pasientene man har trykket seg inn på, siste meldinger innenfor hver chat-type og siste notifikasjoner. Dette er også startsiden man kommer inn på etter man har logget inn.

Grad av måloppnåelse: 5

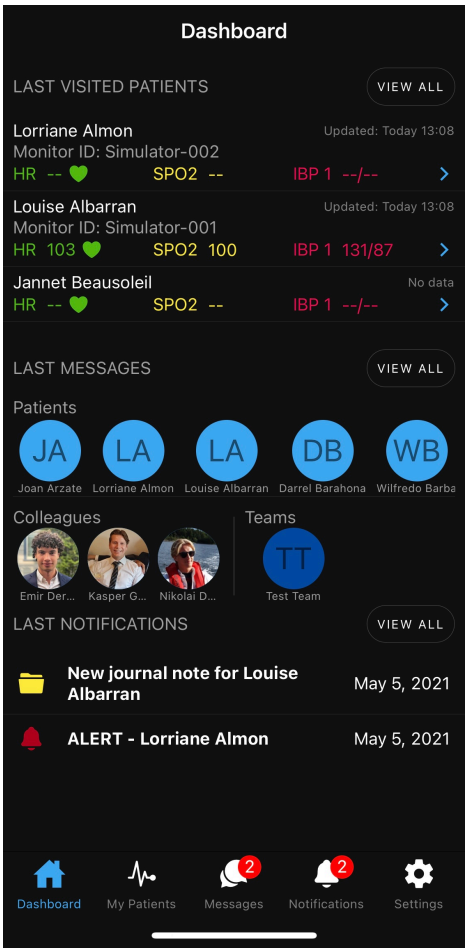


Figure 4.10: Dashboard

Profilside

Brukeren kan se sin egen profil og hente relevant personlige opplysninger. Det eneste man har mulighet til å endre på er profilbilde.

Grad av måloppnåelse: 5

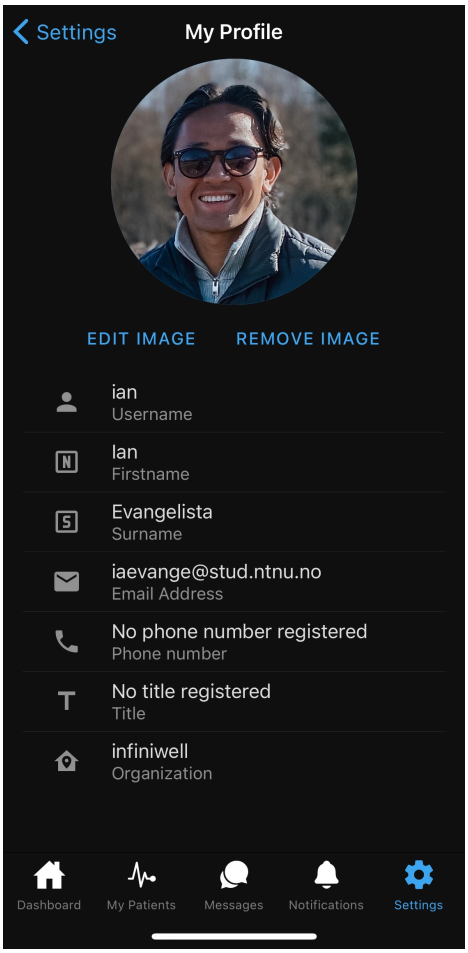
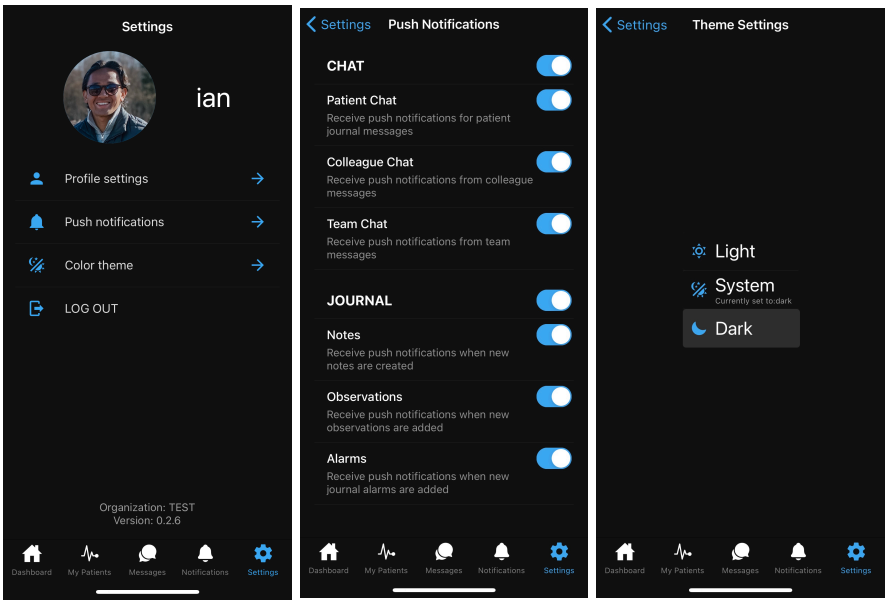


Figure 4.11: Profilside

Innstillinger

På innstillinger kan man endre fargetema i appen ved å enten ha et lyst tema eller et mørkt tema. Første gang man bruker appen vil den bruke enhetens system-tema. I tillegg kan man skru av push-notifikasjoner globalt gjennom innstillinger. Her kan man skru av varslinger for chat, ulike typer chatter og varslinger tilknyttet en pasient.

Grad av måloppnåelse: 5



Innstillinger Push-notifikasjoner Fargetema

Figure 4.12: Innstillinger

4.2.2.2 Utgåtte funksjonelle krav

Etter enighet mellom oppdragsgiver og teamet var det flere av de funksjonelle kravene som utgikk.

- AI gir tilbakemelding om statusen til en pasient
- Velge å sende varsel til personell høyere opp i hierarkiet.
- Videochat
- Flere språk

4.2.2.3 Ikke-funksjonelle krav

Brukeropplevelse og design

Ettersom brukeropplevelse er subjektivt er det derfor vanskelig å definere krav til dette. Likevel var kravene for brukeropplevelse og design at:

"Brukeren skal ha direkte tilgang til den mest aktuelle dataen med minimal klikking gjennom menyer."

Løsningen på dette ble å implementere snarveier flere steder i appen. Eksempelvis kan man fra den detaljerte siden om en pasient raskt navigere seg til pasientens chat. Fra pasientens chat kan man raskt navigere seg til kollega-chatten til alle som er tilknyttet pasienten. Vi lagde også et dashboard som har en samling av alle de nyeste og viktigste dataene man trenger i appen. På dashbordet ser man de siste notifikasjonene, de siste som har sendt deg melding og de siste pasientene du har besøkt i appen.

Kravet for design lød som følgende:

"Design skal være en nøyaktig representasjon av instrumentene som brukes til å måle en pasients vitale egenskaper."

Dette kravet ble oppfylt ved å etterligne kjente pasientmonitorer etter beste evne. Infiniwells webløsning hadde allerede en variant av en digital representasjon av en pasientmonitor. Ved å etterligne denne ville brukeren få en behagelig overgang fra webløsningen til appen, samtidig som vi forsikrer oss om at appens monitor ligner på instrumentene som brukes i den fysiske verdenen. Pasientmonitoren i appen er en nøyaktig representasjon av instrumentene som brukes til å måle en pasients vitale tegn ved at den bruker de samme forkortelsene på dataene, de samme fargene og samme oppsett.

Kompatibilitet

Appen er kompatibel med enheter fra iOS og Android. Under utviklingen har dette blitt testet med eldre og nyere enheter. Appen er tilpasset enhetens størrelse slik at alt er klart og tydelig. Det skal også være mulig å bruke appen med et nettbrett. Under er en oversikt over antall kompatible Android enheter. Oversikten er gitt av Google, under opplastning av appen til Google play store. Her ser vi at man i tillegg til å kunne bruke applikasjonen på mobiler, er den også kompatibel med klokker, TVer og til og med en bil.

Endringer i de støttede enhetene dine

Enheter som er ekskludert i enhetskatalogen, vises ikke

Enhetstype	Tidligere støttede enheter	Enheter som ikke støttes lenger	Enheter det er lagt til støtte for
Telefon	10 824	0	0
Nettbrett	3 592	0	0
TV	6	0	0
Til å ha på seg	2	0	0
Bil	1	0	0
Chromebook	51	0	0

Figure 4.13: Antall støttede Android-enheter (modeller)

Servicevennlighet

Kravet for servicevennlighet lød følgende:

"Koden bør være testbar, helst automatisk testet. Koden bør også være utvidbar og lettleselig for å rette opp i feil og mangler for systemet."

Koden vi utviklet i tjeneren er lettleselig med kommentarer. Tester har blitt laget for alle nye endepunkter, og blitt integrert i en automatisk test laget av Django. Testen kan kjøres lokalt, men kjøres hver gang man ruller ut en ny versjon av tjeneren gjennom GitLab CI/CD.

På klientsiden har vi en beskrivende filstruktur med beskrivende filnavn. Prettier og ESLint har blitt brukt for å skape en god og oversiktlig kode slik at den er lettleselig og det er lett å rette opp i feil. Klienten har snapshot-tester og tester som sjekker om ulike funksjoner fungerer som det skal. Koden for klienten er tydelig delt inn i navigasjonselementer, gjenbrukbare komponenter, service-klasser og views.

Pålitelighet

Dersom en feil skulle oppstå, vil man få tilbakemelding om dette. Et eksempel er uthenting av notater fra en pasientjournal. Dersom det ikke finnes noen notater, vil man få en tilbakemelding om dette. Dette gjør det tydeligere for brukeren å forstå hva som skjer. De fleste feilmeldinger tas hensyn til. Dersom appen ikke er koblet til tjeneren ordentlig vil brukeren få tilbakemelding om dette. Et eksempel er sanntidsdata for en pasient. Her vil man se om en pasientmonitor er tilkoblet eller ikke.

Ytelse

Data er hentet i sanntid og oppdateres automatisk. Vi har vært sparsommelige med bruken av antall åpne websockets for å minimere databruken og antall tilkoblede klienter. Nettverkshastighet er den største flaskehalsen i ytelsen. Slik vi nevnte i kompatibilitet delen har vi testet appen på eldre mobiltelefoner og ytelsen var god.

4.3 Administrative resultater

Dette delkapittelet beskriver de administrative resultatene fra prosjektet. Alle referanser er hentet fra *Vedlegg C: Prosjekthåndbok*. Vedlegget viser til fremdriftsplanen, dokumentasjon av timelister, utviklingsprossen, ukerapporter, møteinnkallinger og møtereferater.

4.3.1 Fremdriftsplan

Fremdriftsplanen er utformet som en milepælsplan og er beskrevet i *Vedlegg C: Prosjekthåndbok, Fremdriftsplan*. Siden prosjektet var videreført fra høstsemesteret hadde teamet allerede rådført seg med oppdragsgiver om hva som skulle bli prioritert. Dette gjorde det enklere for teamet å sette opp fremdriftsplanen. Planen bestod av 17 milepæler, og under er en overordnet beskrivelse av hvordan den ser ut.

Milepælnummer	Beskrivelse	Opprinnelig	Faktisk
1	Forarbeid	8 dager	8 dager
2	Prosjektinitiering	5 dager	5 dager
3	Forskningsspørsmål	10 dager - Sprint 1	10 dager - Sprint 1
4	Notifikasjoner til backend	10 dager - Sprint 1	10 dager - Sprint 1
5	Stack struktur/hierarki	10 dager - Sprint 1	10 dager - Sprint 1
6	Journal - Read	12 dager - Sprint 2	12 dager - Sprint 2
7	Journal - Write	12 dager - Sprint 2	12 dager - Sprint 2
8	Sende varsler	15 dager - Sprint 3	15 dager - Sprint 3
9	Meldingsfunksjon	15 dager - Sprint 3	*27 dager - Sprint 3/4*
10	Trend data	12 dager - Sprint 4	12 dager - Sprint 4
11	Live data	12 dager - Sprint 4	12 dager - Sprint 4
12	Brukertest 1	12 dager - Sprint 4	12 dager - Sprint 4
13	Settings	5 dager - Sprint 5	5 dager - Sprint 5
14	Brukertest 2	5 dager - Sprint 6	5 dager - Sprint 6
15	Akseptansetest	5 dager - Sprint 6	5 dager - Sprint 6
16	Avsluttende vedlegg	12 dager	12 dager
17	Hovedrapport	12 dager	12 dager

Table 4.1: Overordnet milepælsplan

Teamet var flinke på å strukturere og planlegge tiden godt. Dette resulterte i at det meste gikk etter planen. De eneste avvikene fra planen var at milepæl 9: Meldingsfunksjon gikk over sprint 3 og sprint 4. I tillegg ble milepæl 11: Live data, endret til å bli NEWS2.

4.3.2 Scrum

I løpet av prosessen hadde Odd Sandbekkhaug rollen som produkteier, Elise Vonstad hadde rollen som Scrum-master og teamet var utviklerne. Alle aktivitetene vi utførte i Scrum er dokumentert i *Vedlegg C: Prosjekthåndbok, Møter* og *Vedlegg C: Prosjekthåndbok, Scrum.* De fysiske daglige Scrum-møtene innad i teamet var kun holdt muntlig og ble ikke dokumentert annet enn at sprintloggen ble oppdatert og fremgang ble ført i et Burnup-chart for sprinten.

Det ble gjennomført seks sprinter som hadde en varighet på mellom én og tre uker. Grunnet andre emner, og opphold i utviklingen varierte sprintenes lengde. Hver sprints varighet ble satt med hensyn på deres respektive sprintmål. Arbeidsoppgavene som ble lagt til i sprintkøen var bestående av to punkter i tillegg til oppgavebeskrivelsen: prioritet og antall timer arbeid. Prioriteten kunne være høy, medium eller lav. Antall timer man trodde en arbeidsoppgave ville ta, ble endret fra dag til dag. Trello ble brukt for å vise hele produktkøen og sprintkøen. I løpet av sprintene testet teamet ut både Burndown-chart og Burnup-chart for å sette et bilde på hvordan utviklingen gikk. Etter endt sprint ble det holdt en sprint-review og sprint-retrospektiv sammen med oppdragsgiver og veileder, i tillegg til planlegging av neste sprint. Utenfor de obligatoriske møtene innenfor Scrum, hadde teamet og oppdragsgiver ukentlige møter. Disse møtene ble avholdt digitalt for å holde oppdragsgiver oppdatert på produktet.

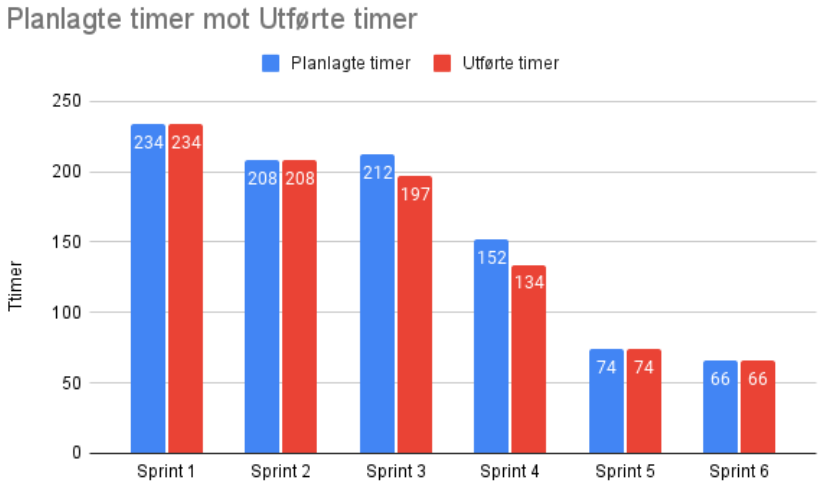


Figure 4.14: Planlagte timer mot Utførte timer per sprint

4.3.3 Timeforbruk

Hvert teammedlem dokumenterte antall arbeidstimer hver uke siden prosjektstart i egne timelister. Her finner man antall timer brukt per type arbeidsoppgave og en kommentar på hva som har blitt gjort. Etter hver uke skrev også hver person en ukerapport. Her skrev man ned hva som var gjennomført gjeldende uke og eventuelle problemer som hadde oppstått. I tillegg skulle man skrive ned tiltak for uken etter dersom man hadde møtt på problemer. Under er en oversikt over antall timer som er utført og hvordan disse er blitt disponert av teamet. Både timelister og ukerapporter finnes i *Vedlegg C: Prosjekthåndbok, Timelister og ukerapporter*.

Arbeidstype	**Planlagte timer**	**Utførte timer**	**Differanse**
Planlegging	200	233	33
Utvikling	1000	1073,5	73,5
Dokumentasjon	800	766	-34
Totalt	**2000**	**2072,5**	**72,5**

Table 4.2: Totalt timeforbruk av teamet per arbeidstype

De ulike arbeidstypene kan beskrives slikt:

Planlegging

Inkluderer alle typer aktiviteter som forarbeid til prosjektet, sprint-møter, møter med oppdragsgiver, veiledningsmøter, planlegging av brukertest og annet liknende.

Utvikling

Alle former for utvikling som er klient-utvikling, tjener-utvikling og testing.

Dokumentasjon

Alle former for dokumentasjon som er hovedrapport, systemdokumentasjon, møtereferater og andre vedlegg.

5 Diskusjon

5.1 Vitenskapelige resultater

Dette delkapittelet vil ta for seg diskusjonen rundt de vitenskapelige resultatene vi fikk i delkapittel 4.1.

5.1.1 Brukergrensesnittet i applikasjonen

Som vi har vært inne på finnes det ingen fasit på et godt brukergrensesnitt. Likevel finnes det flere retningslinjer som forsøker å gi et best mulig svar på hvordan man lager et godt brukergrensesnitt. I delkapittel 2.4 skrev vi om Nielsens 10 heuristikker og Normans designprinsipper som er eksempler på slike retningslinjer. Begge disse dreier seg om å oppnå godt interaksjonsdesign, som vil si et design som gjør det enkelt for brukeren å samhandle med systemet.

Nielsens første designprinsipp omhandler synlighet av systemstatus. Dette innebærer at systemet alltid skal holde brukeren oppdatert på hva som skjer ved hjelp av passende tilbakemeldinger til riktig tid. Måten vi prøvde å følge dette prinsippet var å bruke komponenter som lar brukeren samhandle med systemet, men også komponenter som samhandler med brukeren. Tekstfelt i applikasjonen kan gi feilmelding ved å bli røde, og gi en rød hjelpetekst på hva som er galt. Når det gjelder pasientmonitoren i appen, la vi også inn en status tekst for denne, som tydelig viste om monitoren mottok data eller ikke.

Et annet punkt knyttet tett opp mot brukergrensesnitt er Nielsens 10. heuristikk, som er hjelp og dokumentasjon. Dette punktet sier at man skal ha hjelp og dokumentasjon lett tilgjengelig selv om det optimale er at brukeren ikke får behov for det. I vårt brukergrensesnitt forsøkte vi så langt det lot seg gjøre å forklare enhver funksjon. Brukergrensesnittet vårt hadde beskrivende knapper og tekster, men noen steder la vi også inn egne knapper som kunne forklare en sides hensikt. Dette ble en veldig god løsning og brukergrensesnittet vårt kunne kanskje forbedres om vi la det inn enda flere steder. Heldigvis fikk vi gode tilbakemeldinger på at det gikk raskt å lære seg systemet, og at brukere enkelt husket hvordan de skulle bruke systemet, dersom de hadde brukt det før.

Don Norman nevner det *å være konsistent* som et viktig prinsipp når det kommer til interaksjonsdesign, nevnt i delkapittel 2.4. Vi vil komme tilbake til disse prinsippene i neste delkapittel om brukeropplevelse, men vi mener det å være konsistent er viktig også når man skal lage et brukergrensesnitt. Ved hjelp av React Native Paper har vi kunnet bruke knapper, brytere, og tekstfelt med liknende stiler og egenskaper gjennomgående i applikasjonen og dermed vært konsistente. Siden vi har utviklet en applikasjon for både Android og iOS med samme kodebase, skulle man tro at den så lik ut for begge operativsystemene. React Native Paper tilpasser komponentene sine ut i fra enheten som brukes. Det vil si at komponenter endrer design for å være konsistent med resten av operativsystemets brukergrensesnitt.

5.1.2 Brukeropplevelsen i appen

Som nevnt omhandler Nielsens første designprinsipp synlighet av systemstatus. Dette var et viktig fokusområde for oss gjennom hele utviklingsprosessen og noe vi spesifikt spurte om tilbakemelding på da vi sendte ut brukertester.

For å oppfylle dette designprinsippet implementerte vi flere kjente standardløsninger, som vi antok brukerne våre er kjent med fra før av. Eksempelvis vil man på alle tekstfelt, dersom man gjør en feil, få en rød ramme rundt feltet i tillegg til en rød hjelpetekst under som forklarer hva du har gjort feil. Flere steder i appen kan man utføre handlinger som medfører databasekall, her var det også viktig å informere brukeren om hva resultatet var. Her skjønte vi at det var like viktig å informere brukeren dersom en handling gikk bra, som det var å informere om når handlingen gikk galt. Når

man oppretter et notat i pasientjournalen vil man først få tilbakemelding fra tekstfeltet dersom noe er galt med tittelen eller beskrivelsen. Når man trykker på knappen for å opprette notatet, vil man se et informasjonsbanner som bekrefter at "Note was added".

Et av Don Normans prinsipper vi fokuserte på som er nevnt i forrige delkapittel var *å være konsistent*. Dette går ut på å bruke liknende elementer for å oppnå liknende resultater. En handling skal gi samme reaksjon, hver gang. Dette gjelder både innad i appen, men også sammenliknet med andre applikasjoner. Navigasjonen i applikasjonen er et eksempel på at vi har fulgt dette prinsippet, og vi har fått mange gode tilbakemeldinger på at navigasjonselementene er gjenkjennbare fra andre applikasjoner og at brukeropplevelsen med dem likner operativsystemets egne navigasjon. Alt i alt handler det om å kunne forvente samme reaksjon fra komponenter man er vant til fra andre systemer. Dette har vi oppnådd ved å unngå å finne opp hjulet på nytt og heller bruke gjenkjennbare komponenter.

Normans prinsipp om *tilbydelse* handler om relasjonen mellom hvordan noe ser ut og hvordan det brukes. Her nyter vi igjen fordelene av å bruke komponenter som allerede er brukt i enhetenes operativsystem og andre kjente applikasjoner. Eksempler på komponenter med god *tilbydelse* er brytere vi bruker for å slå av eller på varslinger i applikasjonen.

Figure 5.1: Bryterne som er brukt i applikasjonen (i andre farger), med god *tilbydelse*

Bryterne over illustrerer også et annet viktig prinsipp Don Norman skriver om. *Begrensninger* er et prinsipp som skal begrense brukeren og fortelle at visse handlinger er låst. Dette benytter vi blant annet i varslingsinnstillingene. Dersom varslinger for en hel kategori er avslått, vil det ikke være mulig å slå av eller på varslinger for en underkategori av denne. Bryterene for disse underkategoriene vil derfor være låst (disabled) som man ser på bildet over. Bildet viser også hvordan bryterne tilpasser desginet sitt til det gitte operativsystemet, og dermed følger prinsippet om *å være konsistent*.

5.1.3 Nyttigheten av applikasjonen

Da applikasjonen tilgjengeliggjør data som normalt sett kun er tilgjengelig fra arbeidsstasjoner eller på pasientmonitorer, er teamet av den oppfatning at applikasjonen er høyst nyttig. Ut i fra førspørsmålene vi stilte i brukertestene, svarte flertallet at det viktiste en app for helsepersonell skulle bidra til var å effektivisere arbeidet og arbeidsdagen. Som et resultat av å digitalisere arbeidet til helsepersonell, ville appen skape mer tid for helsepersonell slik at de kunne strukturere arbeidsdagen bedre.

Meldingsløsningen skaper en kommunikasjonskanal for arbeid, og man trenger ikke å bruke SMS eller andre meldingsapper for å nå kollegene sine. Det er også en mer moderne løsning som i fremtiden kanskje kan overta for personssøkere.

Applikasjonen inneholder som sagt en integrasjon mot enhetens push-notifikasjonssystem. Brukeren vil få varsling fra appen dersom ulike hendelser i appen tilknyttet brukeren utspiller seg. Brukeren kan få varsel når man får en ny melding, om en annen bruker oppretter et notat i en pasientjournal

eller om noen ønsker å varsle alle brukerne av appen med en egenutviklet varslingsløsning. For alle disse hendelsene, utenom den kritiske varslingsløsningen, er det gitt muligheten for å skru av disse push-varslingene. På denne måten kan brukeren av appen skreddersy sine egne varslinger og unngå å bli distrahert av unødvendige varslinger. Ved at vi ikke tillater å skru av varselet for den kritiske varslingen vil man alltid bli gjort oppmerksom dersom det oppstår noe alvorlig. Varslinsgsfunksjonen tillater derfor at personer kan sitte på en sentral og følge med på mange pasienter om gangen, og dermed varsle relevant helsepersonell om at en pasient trenger umiddelbar assistanse.

Som tidligere nevnt er det to overordnete behov for helsepersonell: samle nøyaktig data raskt og lette kognitiv belastning. Innenfor hver av de to behovene finnes det flere punkter som er beskrevet under [St. Olavs, 2021].

Samle nøyaktig data:

- Observere data over tid
- Lese data i et hektisk miljø
- Dokumentere og dele data
- Bli oppmerksom på viktig informasjon

Lette kognitiv belastning:

- Minimal bytting av systemer
- Evne til å ta effektive og gode beslutninger
- Ha kontroll og ligge foran
- Samarbeide og kommunisere effektivt

Vi har tolket at appen vår er nyttig dersom den oppfyller flere av behovene St. Olavs hospital mener helsepersonell har. Ved appens pasientmonitorering i sanntid og muligheten for å se denne dataen som trendgrafer innfrir appen behovet for å *observere data over tid*. Journalen i appen lar helsepersonell *lese data i et hektisk miljø* og *dokumentere og dele data*. Med varslingssystemet, vil brukeren alltid være *oppmerksom på viktig informasjon*. Som forklart i et tidligere avsnitt vil meldingssystemet i appen legge til rette for både å *samarbeide og kommunisere effektivt* og *minimal bytting av systemer*.

Tidligere leder for oppbygging av Norsk senter for elektronisk pasientjournal ved NTNU Anders Grimsmo, uttalte i 2007 at det som vil endre forholdet mellom pasient og lege er teknologien [Johannessen, 2007]. Han påpeker at tilliten til pasienter svekkes når de må fortelle samme historie til fem-seks forskjellige leger eller når kirurger bruker så mye tid på å kommunisere at de nesten ikke rekker å operere. Han påpeker flere svakheter ved den norske helsetjenesten som vår app kan bidra på å minke og kanskje til og med eliminere.

Ettersom teamet har tatt på seg oppdraget om å lage applikasjonen for Infiniwell, har teamet ikke undersøkt konkurrenter nøye. Det kan selvfølgelig tenkes at det finnes konkurrenter som har lignende produkter. I så fall kan man stille spørsmål ved nyttigheten av akkurat vårt produkt, dersom markedet allerede er mettet med konkurrenter. Videre er det vanskelig å fastsette nyttigheten av applikasjonen uten en større test der helsepersonell på flere nivåer benytter applikasjonen over en lengre periode. Teamet undersøkte muligheten, men fant raskt ut at dette ikke lot seg gjøre under Covid-19-pandemien, grunnet strenge tiltak for helsepersonell og på sykehus. Dette har resultert i at vi har basert nyttigheten av produktet vårt på våre erfaringer fra brukertestene og behovsetterspørsel vi har lest om.

Applikasjonen rettet mot helsepersonell

Applikasjonen var til beste evne laget til bruk for helsepersonell. Vi ønsket at helsepersonell skulle kjenne igjen elementer fra pasientmonitorer, og at standard fagbegreper ble benyttet. Teamet er fornøyde med resultatet og har fått gode tilbakemeldinger fra helsepersonell av flere nivåer når det gjelder disse punktene. Fra brukertestene beskrevet i delkapittel 4.2.1, fikk vi tilbakemeldinger fra både sykepleiere og kirurger. Likevel er det vanskelig å fastslå om helsepersonell ved forskjellige institusjoner er enige, uten å foreta en storskala test av applikasjonen. Som nevnt i avsnittet over, undersøkte teamet denne muligheten, men det lot seg ikke gjøre grunnet Covid-19-pandemien.

Som tidligere nevnt i delkapittelet om nyttighet 5.1.3, er appen med på å dekke de to overordnete behovene til helsepersonell: samle nøyaktig data og lette kognitiv belastning. Appen samler nøyaktig data til helsepersonell ved at den gir brukeren tydelige og beskrivende målinger gjort av monitorene. Dersom det ikke er en måling kommer dette tydelig frem, og hvis en måling tilsier at en pasient er i faresonen blir dette fremhevet ved hjelp av NEWS2 og fargekoder.

Ved bruk av en NEWS2-score og Infiniwells AI, kan appen være med på å lette helsepersonells kognitive belastning. Den kan gi tilbakemelding på en pasients status og med det avgjøre om pasienten er i en kritisk tilstand i stedet for at brukeren selv må avgjøre det.

5.2 Ingeniørfaglie resultater

Dette delkapittelet vil ta for seg diskusjonen rundt de ingeniørfaglige resultatene vi fikk i delkapittel 4.2.

5.2.1 Brukertester

Det er mange måter å utføre en brukertest på. Vi var avhengige av å finne en løsning som var mest mulig virkelighetsnær med tanke på nedlasting og bruk. Vi måtte også ta i betraktning at testobjektene våre var spredd ut over hele verden, og vi kunne dermed ikke være fysisk tilstede ved utførelse av testene. Ulempen med dette var at vi ikke kunne følge opp testerne like tett som om vi hadde vært fysisk tilstede med dem. Fordelen er derimot at vi ikke legger noen føringer på dem, og at de kanskje tør å være ærligere enn om vi hadde vært i rommet med dem.

Løsningen ble et fyldig og redigerbart dokument som ble tilsendt alle testobjektene, som inneholdt bakgrunnsinformasjon om Infiniwell og oppgaven vår, installasjonsmanual, et testforløp og ulike spørsmål. Det ble i tillegg lagt ved en forklaring på testobjektenes bidrag i prosjektet og en bruker persona slik at testobjektene kunne sette seg inn i tankesettet til appens målgruppe. Det som var bra med at brukerene fikk tilsendt test-instrukser var at de kunne bruke så lang tid de selv ønsket. Ulempen var de kanskje måtte vente noe lengre på respons dersom de lurte på noe og måtte sende oss en epost.

Testobjekter

Ettersom problemstillingen vår belyser både brukskvalitet og helsepersonell, ønsket vi å teste appen vår på helsepersonell i tillegg til personer uten medisinsk bakgrunn. Det var også i vår interesse å teste appen på personer med forskjellige teknologiske kunnskaper, så derfor har noen av testobjektene våre ingeniørbakgrunn. Appen skulle være brukervennlig og gi en god brukeropplevelse til alle som skulle bruke den. Vi mente brukeropplevelse kunne testes av testobjekter uten medisinsk bakgrunn, så lenge de hadde erfaringer med mobile applikasjoner.

Som følge av at appen bygger på Infiniwells løsninger ble den ansett på som sensitiv informasjon, og det ble derfor anmodet at vi ikke skulle dele appen med for mange testobjekter. Det var heller ikke ønskelig at den skulle deles med personer innenfor helseteknologi-sektoren på grunn av fare for

konkurranse. Brukertestene ble derfor distribuert til bekjente av teamet og familie med variert alder og relevant utdanning, samt samarbeidspartnere av Infiniwell.

Totalt ble løsningen testet på 12 personer, der 25% var fra Infiniwell og deres samarbeidspartnere. Jakob Nielsen skrev i en forskningsartikkel at for mange testobjekter i en brukskvalitets-test er en sløsing av ressurser og at de beste resultatene kommer fra å teste ikke mer enn fem personer, men heller teste så mange ganger man har råd til [Nielsen, 2000].

Bruker persona

Det var viktig for oss at testobjektene klarte å sette seg inn tankesettet til de brukerene appen er designet for. Målet vårt var å lage en app som er nyttig og brukervennlig for helsepersonell og skal på grunn av dette effektivisere og gjøre arbeidsdagen deres enklere. For å få mest realistisk tilbakemeldinger fra testobjektene våre skrev vi en bruker persona slik at de kunne sette seg inn i tankesettet til en helsearbeider. Her presenterte vi en typisk sykepleier og nevnte typiske arbeidsoppgaver, vanlige problemer man kan møte på i hverdagen og kjennskap til teknologi.

Testforløp og spørsmål

Før brukeren skulle laste ned appen, ønsket vi informasjon om testobjektene. Eksempler på informasjon vi ønsket var alder, yrke og utdanning. Dette gjorde vi for å kartlegge de ulike brukerne og se hva ulike grupper hadde til felles. Samtidig ønsket vi å finne ut av hva slags mobiltelefon de har og hvor vandt de er med å bruke mobilapplikasjoner. Å vite hva slags enhet og operativsystem brukerne testet applikasjonen på var viktig for å feilsøke problemer brukerne oppdager. Til slutt ønsket vi å vite tankene deres rundt utfordringer helsepersonell møter i hverdagen og hva de forventet av appen. For de fullstendige brukertestene se *Vedlegg E: Tester, Brukertester*.

Etter nedlasting av appen, ønsket vi at brukerne skulle utføre spesfikke handlinger i appen. Vi ga brukerne beskrivende oppgaver uten å forklare hvor de skulle trykke. Dette var for å teste ut hvor intuitive, lesbare og beskrivende elementer var. I hver oppgave ønsket vi å høre hvordan brukerne gikk frem for å utføre den.

Til slutt stilte vi spørsmål angående hver oppgave brukerne gjennomførte og tanker de hadde rundt appen. Dette var for å sjekke om brukerne skjønte oppgaven, om noe gikk galt eller om de var fornøyde med løsningen. Denne delen av testen var mer åpen for tankene deres og eventuelle forslag de hadde for å forbedre appen.

Likert-skala

For måle fremgang og holdningene testobjektene hadde til de ulike elementene i appen, valgte vi å bruke Likert-skala. Skalaen er på ordinalnivå, som vil si at verdiene i skalaen er i en ordnet rekkefølge. Rekkefølgen er som regel fra lav til høy [Malt and Grønmo, 2020]. I brukertesten hadde vi verdier fra 1-5:

- 1: Misfornøyd
- 2: Noe misfornøyd
- 3: Verken fornøyd eller misfornøyd
- 4: Noe fornøyd
- 5: Fornøyd

Som regel pleier man å bruke påstander i en Likert-skal, men tema og emner vil også fungere. Fordelen med å bruke skalaen sett fra vår side, er at det er kvantitativ data som er lett å forstå og prosessere videre til et resultat. I stedet for å få vage og korte tilbakemeldinger, vil man få konkrete svar på fokusområdene. Sett fra brukerens side, er det enklere å forholde seg til en skala fremfor

å uttrykke sin egen mening, samtidig som at man kan forholde seg nøytral. Testobjektene hadde selv ansvar for å fullføre brukertestene, og hvis vi fikk korte eller vage svar tilbake, ville det vært uheldig å måtte be testobjektene svare på nytt og fyldigere. En ulempe med denne metoden er at testobjektene kan bli påvirket av tidligere spørsmål stilt. Det vil si at et negativt eller positivt utslag på et spørsmål kan påvirke det neste.

5.2.1.1 Første brukertest

Etter fem ukers utvikling brukertestet vi en versjon av systemet vårt klar til testing. Teamet fikk distribuert appen til relevante testobjekter i både Norge, India og USA. Testen ble gjennomført på egenhånd av testobjektene. De fikk noen spørsmål de skulle svare på før testen, for å kartlegge deres forventninger til appen før de i det hele tatt hadde åpnet den. Her spurte vi også om testobjektets kunnskap innenfor helsesektoren.

Testobjektene fulgte et testforløp som var laget slik at testobjektene fikk navigert seg gjennom de delene av appen vi hadde vektlagt så langt i utviklingsprosessen. Denne brukertesten fokuserte på pasientjournalen samt notifikasjoner. Her ble testobjektene bedt om å notere fremgangsmåte og tanker samtidig som de gjennomførte testen. Dette viste seg å være veldig nyttig da vi raskt så hvor testobjektene hadde hatt problemer med navigasjon og hvor det var forbedringspotensialer. Det var også interessant å se de ulike fremgangsmåtene de ulike testobjektene hadde.

Etter testforløpet kom vi med oppfølgingsspørsmål. Disse spørsmålene besto av både åpne spørsmål og spesifikke spørsmål. Dette var fordi vi hadde noen løsninger i appen vi selv var i tvil om, og hadde derfor åpne spørsmål der vi ba derfor om forslag til forbedringer. Andre steder ønsket vi mer konkrete svar på om løsningen vår falt i smak. Her fikk vi bekreftet at vi stort sett tenkte likt som testerne våre, samtidig som vi fikk flere gode tips for å øke brukskvaliteten i appen.

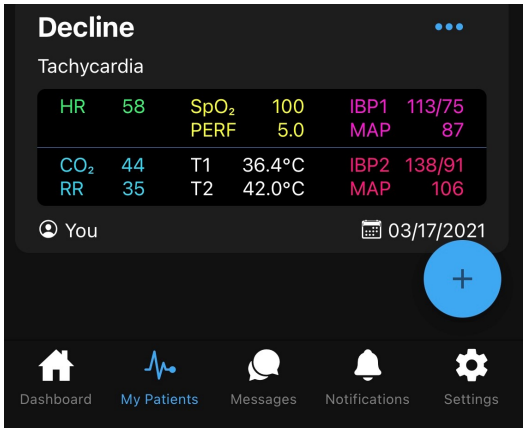
Mot slutten av testen spurte vi om brukerene savnet noe fra appen. Dette var gode tilbakemeldinger for oss ettersom vi både fikk bekreftet at vi ikke hadde oversett noe viktig, og vi fikk gode innspill til ny funksjonalitet som kunne forbedre appen.

5.2.1.2 Andre brukertest

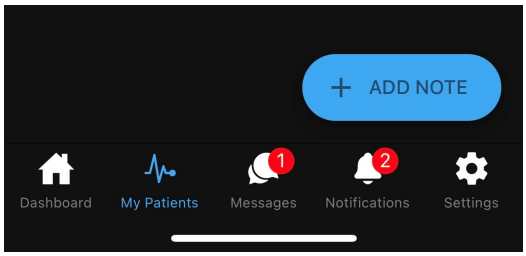
Andre brukertest ble sendt ut til de samme testobjektene, med stort sett likt testmateriell. Vi fjernet noen av de mer faglige delene, som bruker persona og informasjon om Infiniwell. Dette var fordi testobjektene allerede hadde lest dette og vi ønsket å holde fokuset på testingen av de nye funksjonene i appen.

Denne brukertesten var i korte trekk ganske lik som første. Den største forskjellen var at testobjektene fikk litt kortere tid til å sende tilbake testen. For oss var det viktig å se på de statistiske spørsmålene for å enklest se en utvikling fra første brukertest. De fleste punktene fikk bedre gjennomsnittsscore på andre brukertest, som er beskrevet i delkapittel 4.2.1.2. Likevel fikk vi lavere gjennomsnittsscore på beskrivende UI til tross for at vi hadde tatt bevisste valg for å forbedre beskrivelsen til diverse komponenter.

Under ser man to eksempler på endringer av beskrivende brukergrensesnitt fra første brukertest til andre. Flere testobjekter sa ved første test at de ikke skjønte hva alle knappene betød da de kun hadde ikoner, så flere steder valgte vi å legge til tekstlig beskrivelse på knappene(ref. figur 5.2).

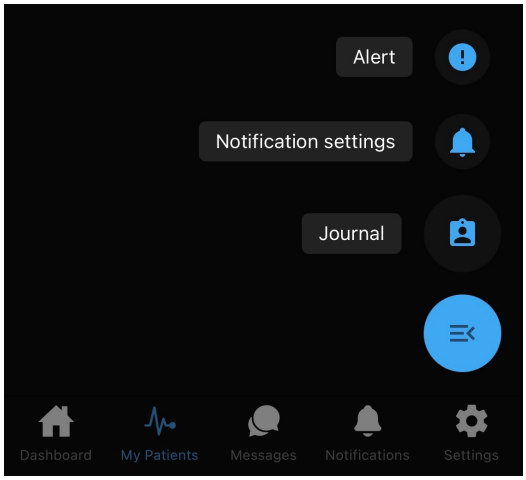


Nytt notat - første brukertest

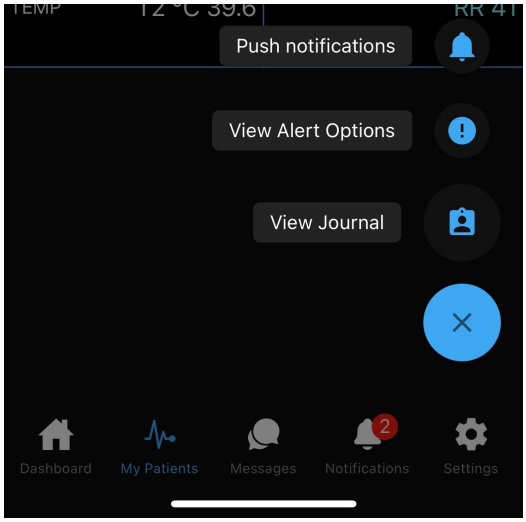


Nytt notat - andre brukertest

Figure 5.2: Hvordan opprette nytt notat - En sammenligning mellom brukertestene



Menyvalg fra første brukertest



Menyvalg fra andre brukertest

Figure 5.3: Menyvalg ved bruk av en FAB (Floating Action Button)

Gjennomsnittsscoren for lesbarhet var en av de som økte mest. Forbedringen kan ha en korrelasjon med endringen av farger i appen. Bedre valg av farger skapte tydeligere kontraster og dermed bedre lesbarhet. Flere steder ble fontstørrelsen økt for å forbedre lesbarheten, uten at det gikk på bekostning av appens kompakthet.

I tillegg til å legge til tekstlig beskrivelse på knappene var vi bevisste på bruken av ikoner. React Native Paper lot oss bruke ikoner fra Googles Material Design Icons som ga oss et utvalg på 26 000 forskjellige ikoner. Denne mengden av valgmuligheter gjør det enda viktigere å forstå karakteristikken til et ikon. Når man har bestemt seg for et passende beskrivende ikon kan man som regel velge mellom ulike design av det samme ikonet. De vanligste variantene er fylte ikoner og ikoner med kun kontur eller omriss.

Forskning viser at det er mer enn kun preferanser som skiller de to typene. Et forskningstudium kalt *Filled-in vs. Outline Icons: The Impact of Icon Style on Usability*, fant ut at ikonets stil påvirker ytelsen til utføring av oppgaver. Oppgaveytelsen ble målt av hastigheten og nøyaktigheten til å gjenkjenne og velge ikoner [Arledge, 2014]. Studiet konkluderte med at fylte ikoner generelt sett var raskere å kjenne igjen, men at det likevel fantes unntak. Vi var bevisste på dette fra begynnelsen av da vi lagde wireframes for applikasjonen, se *Vedlegg B: Kravdokumentasjon, Wireframes*.

5.2.2 Akseptansetest

Det endelige produktet samsvarer godt med de kravene som ble satt i *Vedlegg A: Visjonsdokument, Produktets funksjonelle egenskaper* og *Vedlegg A: Visjonsdokument, Ikke-funksjonelle egenskaper og andre krav*. Oppdragsgiver har vært med gjennom hele prosessen og var godt orientert over fremgangen vår ettersom vi benyttet en agil metodikk. Det var dermed ingen overraskelser da testen ble gjennomført. Grunnen til at vi gjorde det på denne måten var at vi ville formelt forsikre oss om at alle forventningene til oppdragsgiver var oppfylt. I tillegg til å si om et krav var godkjent eller ikke, ønsket vi også at oppdragsgiver skulle gi en grad av måloppnåelse. Dette var for å kunne gi oss en pekepinn på hvor godt løsningene vi implementerte falt i smak hos oppdragsgiver.

Vi samkjørte også akseptansetesten med en kode-overrekking. Der gikk vi igjennom all kode som var blitt produsert, sammen med oppdragsgiver og en av Infiniwells samarbeidspartnere. Dermed var vi også forsikret om at Infiniwell hadde innsikt i hvordan koden er skrevet slik at de kan videreutvikle den.

5.2.2.1 Funksjonelle krav

Sett i sammenheng med de funksjonelle kravene som ble satt i begynnelsen av prosjektet, samsvarer løsningen vi har utviklet godt med disse. Under vil vi diskutere den gitte graden av måloppnåelse fra oppdragsgiver.

Innlogging

Det funksjonelle kravet er **oppfylt**.

Innloggingssystemet vårt baserer seg mye på Infiniwells system, ettersom det er samme brukere som skal bruke webløsningen og appen. I tillegg til en ordinær innlogging med brukernavn og passord ønsket oppdragsgiver at man skulle kunne velge tilhørighet ved å oppgi en tjener-vert. Dette kommer av at Infiniwell planlegger å ha egne tjenere for de ulike kundene av systemet. Dette betyr i praksis at brukere ved eksempelvis St. Olavs hospital vil kunne oppgi sykehusets verts-id, og deretter logge inn. På denne måten avgrenser man systemet og oppnår bedre sikkerhet.

Det ble også etterspurt å implementere muligheten for å logge inn med QR-kode. Det gjorde vi ved å bruke Expo sin kamerafunksjon, som har innebygd QR-skanner. På denne måten kan brukere som allerede er innlogget på arbeidsstasjonen sin lettere komme i gang med å bruke appen. De kan så skanne QR-koden med mobilen sin og appen vil automatisk fylle ut verts-id og brukernavn. Av sikkerhetsmessige årsaker bestemte vi oss for at brukeren måtte oppgi passordet sitt selv ved QR-innlogging. Dette ble vi enige om med oppdragsgiver at var den beste løsningen.

Pasientoversikt

Det funksjonelle kravet er **oppfylt**.

Pasientoversikten var et av de viktigste kravene fra oppdragsgiver. Her skulle man raskt og enkelt kunne se alle pasientene samlet på en side og hente ut den viktigste og mest kritiske informasjonen for hver pasient. Vår primære løsning var en enkel liste med alle pasientene. Her ble det vist pasientens navn i tillegg til pasientens siste måling av puls, SpO2 og IBP1. For brukskvalitet la vi også til en høyrevendt pil for å indikere at man kan trykke seg videre for å komme til den detaljerte siden om

pasienten. Det ble også lagt til tydelige indikasjoner for når den siste dataen var hentet ut i form av et tidspunkt øverst i høyre hjørne. Dersom pasienten var sjekket inn, men ikke koblet til en monitor ville pasienten fremdeles befinne seg i listen, men da med "No data" der hvor tidspunktet normalt ville vært. I tillegg vil det være bindestreker hvor de ulike monitorverdiene ville vært. Dette er en normal måte å indikere at det er mangel på data, som brukertestene bekreftet.

Etter ønske fra oppdragsgiver skulle vi teste ut en løsning der vi kunne beholde den gamle listen med monitorverdier, samtidig som man kunne velge å se NEWS2 dersom man ville det. Vårt forslag ble derfor en vippebryter, slik at brukeren selv kunne veksle mellom den originale listen med monitordata og den nye listen med NEWS2-score. Pasienten med høyest NEWS2-score skulle plasseres høyest for å indikere at det må holdes et ekstra øye til akkurat den pasienten. Oppdragsgiver ønsket også at NEWS2 listen skulle være standardlisten. Derfor vises NEWS2 for alle pasienter når man går inn på appen hver gang.

Pasientmonitorering

Det funksjonelle kravet er **oppfylt**.

Pasientmonitorering er appens fundament. Her var ønsket fra oppdragsgiver å ta mest mulig inspirasjon fra deres webløsning, som igjen har tatt mye inspirasjon fra de fysiske monitorene brukt på dagens sykehus. Dette vil gjøre det enklere for helsepersonell å bevege seg fra en fysisk hverdag over til en digital løsning.

Infiniwells webløsning SentioWeb har i tillegg til live-monitorering muligheten til å vise trendutvikling hos pasienten over lengre tidsperioder. Dette ønsket oppdragsgiver at vi inkluderte i appen. Fra høstprosjektet hadde vi en side i appen vi kalte for "Patient Details" som var den siden du kom til dersom du trykket på en pasients rad i pasientoversikten (ref. delkapittel 5.2.2.1). På denne siden fant man både live-monitorering i form av numeriske verdier og trend-monitorering i form av grafer. I dette prosjektet ønsket vi å utvide live-monitorering slik at denne kunne vises i form av EKG (Elektrokardiogram) i tillegg til numeriske verdier. Oppdragsgiver utrykket også et ønske om å minimalisere mobildataforbruket og åpne websockets i appen. Vi kom dermed fram til at det var hensiktsmessig at vi splittet Live og Trend i to separate sider. Dette var en hensiktsmessig løsning fordi da kunne vi holde en websocket åpen kun når brukeren spesifikt ville se live-data. Hvis brukeren kun ønsket å se langsiktig trend-data, som kan være inntil en megabyte, kunne vi holde websocketen lukket. Dermed ble vi kvitt to problemer ved å hindre at man både hadde en åpen websocket og hentet inn store mengder med data samtidig. Splittingen gjorde vi med React Navigations sin Tab Navigator (ref. delkapittel 3.4.2). Ved at vi splittet Live og Trend økte også brukskvaliteten ettersom hver av sidene ble enklere og mer oversiktlig.

Live-monitoreringen skulle som sagt bestå av to elementer, numeriske vitale målinger pasienten har i sanntid og en EKG som visualiserer rytmen til målingene. På sprint review 3 ble det diskutert mål for sprint 4 og der kom det fram et ønske fra oppdragsgiver om å inkludere NEWS2 i appen. Dette ønsket dukket plutselig opp fra oppdragsgiver etter at St. Olavs kom med en utlysning der Infiniwell ønsket å være kapable til å konkurrere om å inngå et innovasjonspartnerskap med St. Olavs. NEWS2 er som forklart i delkapittel 2.2.5 en skår som beregnes ut ifra flere fysiologiske parametere, som Infiniwell i lengre tid har planlagt å inkludere i sin webløsning. På grunn av utlysningen fra St. Olavs som omfattet 10 millioner kroner ble implementering av denne funksjonaliteten høyt prioritet, ettersom utlysningen nevnte NEWS2 spesifikt som et av behovskravene. Takket være vårt valg av en agil utviklingsmetodikk var vi tilpassningsdyktige og bestemte i samarbeid med oppdragsgiver og Scrum-master å fokusere på NEWS2 i utviklingen av Live monitoreringen i appen. Oppdragsgiver ga også utrykk for at EKG uansett ikke passet godt inn på en liten mobilskjerm. Vi ble dermed enige om at støtte for EKG utgikk og støtte for NEWS2 inngikk i dens plass. Resultatet ble at vi fikk implementert NEWS2 i løpet av kun en uke og oppdragsgiver fikk levert inn skjermutklipp fra appen i infiniwell sin søknad, noe oppdragsgiver var svært fornøyd med.

Pasientjournal

Det funksjonelle kravet er **oppfylt**.

I begynnelsen av prosjektet ble vi enige med oppdragsgiver om at pasientjournal var viktig å implementere. Appen skulle ikke i like stor grad som webløsningen, inneholde alt av opplysninger. Derfor satte vi oss som mål å implementere notater, alarmer og observasjoner i appen.

Gjennom utviklingen av hvordan vi skulle fremstille notater, tenkte teamet det var en gode ide å vise en pasients vitale mål ved notatets opprettelse. Dette vil kunne gi andre helsepersonell et overblikk over pasientens status da notatet ble laget. Derfor la vi det inn som et valg for brukeren som laget notatet å legge inn parametere i notatet. Dette var en ide oppdragsgiver likte svært godt og ønsket å ha med videre i appen. Etter to brukertester fikk vi gode tilbakemeldinger om småfeil rundt notater. Dette var at man kunne laget et notat med svært mange linjeskift, noe som gjorde notatet veldig langt. Dette ble raskt fikset, i tillegg til at vi la en maks grense på tittellengde og lengden på beskrivelsen. Det var også viktig at en bruker bare kunne redigere og slette sine egne notater. Dette blir sjekket både av klienten og tjeneren.

Etter akseptansetesten kom det fram at det kan være ulovlig i noen land å slette notater fra en pasientjournal. Denne funksjonen kan lett fjernes fra både klienten og tjeneren, og er et punkt å ta med seg i videre arbeid. I motsetning til notater skulle både alarmer og observasjoner være skrivebeskyttede. Det vil si at man bare kan lese data og ikke har mulighet til å modifisere noe. For alarmer valgte vi å fremstille det tabellarisk med en utvidbar rad for hver alarm for å vise mer informasjon. Dette gjelder også for observasjoner, bortsett fra at observasjoner har mindre informasjon og behøver ikke å være utvidbare. Dersom en pasient ikke har noen notater, alarmer eller observasjoner vil brukeren få tydelig tilbakemelding om dette.

Varslinger

Det funksjonelle kravet er **oppfylt**.

Varslinger eller push-notifikasjoner var et av de store kravene fra oppdragsgiver når det gjaldt appens funksjonaliteter. Dette er en effektiv måte å få umiddelbar kontakt og oppmerksomhet fra brukere. Teamet tenkte derfor at brukeren skal kunne slå av varsler ettersom man kan få flere push-notifikasjoner som omhandler mindre viktig informasjon. Derfor ble det lagt til rette for å kunne slå av varsler for alle typer push-notifikasjoner. Dette ble senere endret utover i prosjektet da vi innså at funksjonaliteten med å motta et nødvarsel om en pasient ikke burde kunne slås av. På klient-siden ble det lagt til rette for varsler når et vitalt mål som hjertepuls eller temperatur gikk over eller under visse grenser. Idéen kom fra teamet og ble presentert til oppdragsgiver som en ekstra funksjonalitet. Oppdragsgiver syntes det var en god idé, men teamet fant raskt ut at det krevdes store endringer på tjener-siden for å implementere det. Samtidig måtte man på forhånd vite normalintervallet på hvert vitalt mål for å vite grensene per vitale tegn. Derfor ble teamet enige med oppdragsgiver om å ikke fokusere på dette. Dette kan derfor ansees som videre arbeid.

Chat

Det funksjonelle kravet er **oppfylt**.

Chat var et annet prioritert krav fra oppdragsgiver. Den ble utviklet i samarbeid med andre utviklere fra Infiniwell, og med god kommunikasjon mellom begge parter fikk teamet implementert en chat-løsning i appen. Vi fikk implementert en fungerende chat-løsning i løpet av sprint 3 slik som planlagt, men løsningen oppdaterte seg ikke i sanntid med nye meldinger. Grunnen til dette var at vi fikk beskjed om at websockets for chat fortsatt var under utvikling på Infiniwell sin side. I løpet av sprint 4 fikk vi funksjonalitet for sanntid chat i tjenerkoden. Vi modifiserte tjenerkoden for å legge til støtte for vedlegg og push-notifikasjoner, og implementerte det i appen. Selv om implementasjonen av løsningen tok lenger tid en forventet, var oppdragsgiver svært fornøyd med

det endelige resultatet. Det er mulig å kommunisere med andre helsepersonell innenfor brukerens organisasjon. Dette kan gjøres ved direktemelding til kollega, pasient-chat eller team-chat. Det er også enkelt å søke opp en enkelt pasient, kollega eller team. Kravet om lydopptak er oppfylt i tillegg til at teamet la inn muligheten til å sende bilder og pdf-er. Chat-løsningen fungerer på tvers av både nettsiden og appen. Det vil si at brukere kan kommunisere med hverandre uavhengig av hvilken plattform de bruker.

Teamet skjønte at en pasient-chat kunne mistforstås. Det kan tenkes at en bruker snakker med en pasient, men i realiteten snakker man med annet helsepersonell som har ansvar for pasienten. Derfor ønsket teamet å plassere ekstra informasjon som forklarer hva en pasient-chat går ut på. Dette ble gjort ved å legge inn en "hjelp"-knapp som er beskrevet i delkapittel 3.4.1.3. Svakheter man kan finne i chat-løsningen er relatert til lydopptak. For iOS-brukere kan lyden i noen tilfeller være lav. For Android ble det besluttet å ikke inkludere en fremdrifts-linje som viser hvor langt i lydopptaket man er kommet. Grunnen til at det ikke ble inkludert er at animasjoner var krevende for visse modeller.

Badges

Det funksjonelle kravet er **oppfylt**.

Badges var en nyttig funksjon for brukskvalitet og oppdragsgiver var svært fornøyd med implementasjonen. I chattene valgte vi å skille mellom *lest* og *sett*. Om du har vært inne og *sett* at du har uleste meldinger i meldinger-skjermen fjernes badgene. Hver chat er forsatt uthevet fram til du går inn i chatten og *leser* de nyeste meldingene.

Vi lot oss inspirere av kjente meldingsapper da vi utviklet dette systemet. Badges er viktige for å få brukeren til å navigere til sider med ny informasjon. Likevel er det viktig at badges forsvinner når brukeren har registert den nye informasjonen, slik at de dukker opp igjen når det faktisk er ny informasjon brukeren ikke har sett.

Dashboard

Det funksjonelle kravet er **oppfylt**.

Oppdragsgiver ønsket en forside til appen som skulle gi brukeren rask oversikt og hurtignavigering til appens viktigste sider. Oppdragsgiver var svært fornøyd med resultatet.

På dashboardet valgte vi å vise de siste pasientene man hadde besøkt med en begrensning på tre pasienter. Vi valgte denne løsningen for å ha plass til snarveier for både meldinger og notifikasjoner. Siden skal sikre færrest mulig trykk for å komme seg dit man ønsker, og sørger forhåpentligvis for at helsepersonell raskt får utført den oppgaven de er inne på appen for å gjøre. Dermed kan de bruke mer tid på deres viktige arbeid. Faren med dashboardet kan være at brukere synes det er for mange elementer her, og dermed synes grensesnittet blir vanskelig fordi funksjonalitetene ikke kommer godt nok frem. Likevel var flertallet av test-objektene fornøyde med siden, og synes den ga verdi i form av færre trykk.

Profilside

Det funksjonelle kravet er **oppfylt**.

I starten planla vi å gi brukeren mulighet til å endre på all brukerinformasjonen hans/hennes, men etter diskusjon med oppdragsgiver kom vi frem til at appen ikke var et passende sted å utføre slike endringer. Muligheten til å endre profilbilde beholdt vi ettersom dette var noe vi selv hadde utviklet og webløsningen ikke hadde støtte for profilbilder enda. Det er i tilegg vanlig at brukere har bilder av seg selv på mobilen.

Innstillinger

Det funksjonelle kravet er **oppfylt**.

Brukeren har mulighet til å endre fargetema i appen og se profilinfo ved å enkelt trykke seg inn på innstillinger. På innstillinger kan brukeren også skreddersy hva man ønsker av varsler.

Et av målene fra visjonsdokumentet vårt var å inkludere flere språk i applikasjonen vår. Vi fikk tidlig beskjed av oppdragsgiver om å ikke prioritere dette, og senere i utviklingsprosessen kom vi frem til at vi skulle se bort i fra dette målet. Dette kom av at vi fikk beskjed om å fokusere på implementasjonen av NEWS2 score, som kom som et ønske sent i utviklingen. I tillegg fant vi ut at vi måtte ha oversatt mye fra tjeneren også, om vi skulle hatt støtte for flere språk. Det vil si at vi måtte implementert språk-funksjonalitet i både klient og tjener. Kort sagt fant vi ut at det var mye arbeid for en lite viktig arbeidsoppgave. Ved akseptansetesten kom det dog frem at systemet skal tas i bruk i Afrika, og at nettløsningen skal oversettes til fransk. Oppgaven var altså viktigere enn først antatt og vi har derfor skrevet om muligheten for språkbytte i delkapittel 6.2 om videre arbeid.

5.2.2.2 Utgåtte funksjonelle krav

AI gir tilbakemelding om statusen til en pasient

For AI integrasjonen planla vi å tilrettelegge notifikasjonssystemet slik at AI automatisk kunne sende ut notifikasjoner til brukere hvis den oppdager noe uregelmessig. Denne delen av systemet ble implementert men er imidlertid ikke i bruk ettersom AIen for dette ikke var helt klar fra Infiniwell sin side. Når dette er klart bør det være en lett sak å legge til støtte for notifikasjoner. Vi har imidlertid feilmeldinger på monitoren på live siden hvor AI brukes for å gi brukeren et hint om noe er galt.

Velge å sende varsel til personell høyere opp i hierarkiet

Tidlig i prosjektet hadde oppdragsgiver en ambisjon om å dele inn brukere i et hierarki med leger, sykepleiere og medisinske teknikere. For appen planla vi å tilrettelegge for at notifikasjoner først gikk til sykepleiere eller annet helse personell med tettere ansvar for færre pasienter. Deretter kunne personen sende en melding opp i hierarkiet til for eksempel en lege om det krevde oppfølging. Denne ideen utgikk ettersom at den krevde et sofistikert brukerhierarki fra Infiniwell sin side, noe de ikke rakk å implementere i løpet av prosjektet. En erstatning for denne funksjonen var teams og pasient-chatter.

Videochat

Videochat var en lavprioritets funksjon som ble nedprioritert ettersom vi sammen med oppdragsgiver identifiserte at det ikke er like nyttig som muligheten til å dele dokumenter, bilder og lydfiler. Vi innså at nytteverdien ble utveid av mengden tid som måtte investeres for å implementere videochat på en trygg og hensiktsmessig måte.

Flere språk

Endring av språk var også en lavprioritets funksjon som ble nedprioritert. Oppdragsgiver foreslo ideen for å tilrettelegge for brukere som ikke kan engelsk. Funksjonen ble nedprioritert ettersom vi innså at mye av det tekstlige som dukker opp i appen kommer fra tjener. Dette vil si at vi måtte tatt for oss oversettelse av både klient- og tjenerinnhold. Oppdragsgiver mente uansett at det er usannsynlig at brukere ikke forstår engelsk ettersom det er det webløsningen deres bruker. Dermed prioriterte vi heller finpuss av andre funksjoner i appen. Under kode-overrekkelsen kom det frem at systemet skal tas i bruk i Afrika, og at oversettelse til blant annet fransk vil være en del av det videre arbeidet med applikasjonen.

5.2.2.3 Ikke-funksjonelle krav

Brukeropplevelse og design

Under utviklingen har det blitt lagt et stort fokus på å gi brukeren en god brukeropplevelse. Ettersom brukeropplevelse er et subjektivt og derfor vanskelig å måle, ble det utarbeidet to akseptansekrav for å kunne måle oppnåelse av det ikke-funksjonelle kravet om brukeropplevelse og design. Selv om det er vanskelig å måle om noe er brukervennlig skriver Jakob Nielsen at man oppnår god brukskvalitet dersom systemet blant annet er lett å lære og effektivt (ref. delkapittel 2.4.1).

Et av kravene som ble utformet ble derfor muligheten for å *"ha direkte tilgang til den mest aktuelle dataen med minimal klikking gjennom menyer"*. Dette tolket vi som oppnåelig ved å lytte til Nielsen ved å utvikle et effektivt system. I følge Nielsen er et system effektivt dersom det lar ekspertbrukeren oppnå en høy grad av produktivitet. Ved å strukturere appen med ulike faner og skjermer fra React Navigation kunne vi sikre at de viktigste sidene i appen var enklest mulig tilgjengelig fra hvor som helst i appen. Ved å lage et dashbord som samlet all aktuell data på et sted kunne man enda raskere få tilgang til denne dataen med minimal klikking.

Dashbordet i appen var delt inn i tre seksjon der man finner sist besøkte pasienter, siste meldingene du har mottatt eller sendt og nyeste notifikasjoner. Alle disse elementene kunne man også nå ved å navigere seg til deres respektive sider fra navigasjons-baren på bunnen av skjermen, eller ved hjelp av dashbordet. I tillegg ble det implementert ulike snarveier i appen for å minimere antall klikk. Dersom snarveiene ikke eksisterte hadde man vært nødt til å bruke navigasjons-baren for å navigere seg til en hovedskjerm før man måtte bevege seg dypere i stabelen av skjermer. Snarveiene kan derfor redusere antall klikk gjennom menyer drastisk.

Da systemet for notifikasjoner i appen skulle utvikles ble interaktivitet vektlagt. På grunn av dette er både notifikasjoner i appen og push-notifikasjoner interaktive, slik at du kan navigere deg til det stedet i appen de omhandler kun ved å trykke på notifikasjonen. Eksempelvis vil du bli sendt direkte til den chatten du får varsling om, eller til den pasienten en pushvarsling omhandler.

Vi hadde i stor grad autonomi i designet av appen, men vi fikk et dokument som beskrev den grafiske profilen til Infiniwell. Oppdragsgiver ønsket at vi skulle bruke fargene som var i deres grafiske profil og at designet vårt nøyaktig representere *"instrumentene som brukes til å måle en pasients vitale egenskaper"*. Vi utforsket hvordan standardinstrumentene som brukes til å måle pasienters vitale egenskaper ser ut. Selskaper som Samsung og Philips leverer begge pasientmonitorer som kan måle flere vitale målinger samtidig og vise dette på samme skjerm. Infiniwell samarbeider med Clarity Medical som er en indisk-basert produsent av medisinsk utstyr. De leverer produktet Recobro Vigile som Infiniwell har integrert med sine tjenester. Dette er en miniatyrisert pasientmonitor som måler opptil syv vitale egenskaper og overfører disse dataene trådløst. Ved å ta inspirasjon fra designet av den fysiske monitoren ville vi sikre oss å nå kravet om å designe en nøyaktig representasjon av instrumentene som brukes til å måle vitale egenskaper.

Fargene brukt i pasientmonitoren i appen er hentet direkte fra SentioWeb sin digitale monitor. Disse fargene var originalt brukt andre steder i appen for å sikre en rød tråd i designet. Tilbakemeldingene fra brukertestene var ganske tydelige på at disse fargene ble for skarpe når de ble brukt utenfor pasientmonitoren, og vi ble derfor nødt til å finne en valør av de ulike fargene for å tilfredstille brukerne samtidig som vi opprettholdt den røde tråden i designet.

Kompatibilitet

React Native-kode kan kjøre på en lang liste av forskjellige enheter. Det var en av årsakene til at vi valgte rammeverket. Vi valgte å fokusere på Android- og iOS-enheter ettersom det var et krav fra oppdragsgiver, men koden kan kjøres på nettbrett, smartklokker, Windows, macOS og i nettleser [Facebook, 2021]. I tillegg viser Google Play Store oss at applikasjonen vår kan kjøres på

konsollen i én type bil(ref. figur 4.13). Det vil kreve mer testing og design for å forsikre oss om at brukeropplevelsen er like god på alle disse enhetene. Oppdragsgiver testet appen på et Android-nettbrett og alt fungerte fint uten at teamet spesifikt hadde utviklet for nettbrettet.

Servicevennlighet

Vi begynte prosjektet med å overhale hele strukturen på klientkoden. Målet var å rydde opp slik at koden var lettere å arbeide med både for vår del og for fremtidig videreutvikling. Et viktig moment var at kodens stuktur intuitivt skulle gjenspeile hvordan klienten fungerte, noe vi mener den gjør nå.

For tjenerkoden var det viktig for både oss og oppdragsgiver at den var i tråd med den etablerte stilen ettersom vi utvidet en eksisterende kodebase. Vi brukte dermed god tid på å bli kjent med kildekoden for tjeneren slik at koden vi produserte passet inn med systemene deres og ikke skapte konflikter med deres webløsning.

Til slutt hadde vi en grundig kodeoverrekking hvor vi gikk igjennom strukturen sammen med Infiniwell. Vi leverte også en rekke dokumenter og modeller som forklarer koden vår og endringene våre.

Pålitelighet

Vi har tatt høyde for mulige feil etter beste evne med gode tilbakemeldinger til brukeren når feil oppstår. Ved brukertesting benyttet vi verktøy for å overvåke hvor mange ganger appen krasjet. I løpet av førte brukertest oppstod det krasj i noen få tilfeller. Vi oppdaget noen feil i etterkant og fikk fikset dette til neste brukertest. Da rapporterte test-verktøyene ingen krasj. Når den ferdigbygde appen krasjer startes appen raskt opp igjen av seg selv og brukeren blir tatt tilbake til dashbordet og kan prøve på nytt. Vi fikk ingen tilbakemeldinger på krasjing selv om vi vet at krasjer oppstod, noe som tyder til at brukerne ikke la merke til at den krasjet.

Appen sin pålitelighet er nært knyttet til påliteligheten til Infiniwell sine tjenere. Derfor har vi sørget for at endringene vi gjorde i koden for tjeneren var testet godt. Django har også god feilhåndtering som sørger for at feil som oppstår ikke medfører fullstendig krasj.

Ytelse

Ytelse er essensielt for appen ettersom den skal kunne laste inn store mengder data i sanntid. Vi kunne fått bedre ytelse ved å utvikle to separate native applikasjoner for iOS og Android, men kom fram til at den største flaskehalsen i ytelsen kom til å bli nettverksforbindelsen og at ytelsesforskjellen dermed ville være minimal. Den mest ytelseskrevende funksjonen i appen er trenddata oversikten ettersom mengden datapunkter i Infiniwell sine systemer er enorm. Bare det å tegne grafene var meget ressurskrevende og vi måtte gjøre en rekke omfattende optimaliseringer for å gjøre brukeropplevelsen god. I våre møter med oppdragsgiver ble det diskutert tiltak som kunne gjøres for å øke ytelsen, men oppdragsgiver krevde at all data skulle vises nøyaktig uten noen form for utjevning og approksimasjoner for å unngå missvisende figurer. Oppdragsgiver bistod også med eldre iOS og Android mobiltelefoner da vi etterspurte det. Vi brukte disse mobilene for testing og fikk dermed bekreftet at ytelsen er god nok selv på gamle og trege mobiler.

5.2.3 Styrker og svakheter som en konsekvens av prosess og valg av teknologi

Prosess

Daglige Scrum-møter gjorde at vi til enhver tid visste hva alle teamets medlemmer jobbet med, og man fikk tatt opp eventuelle problemer man trengte hjelp med. Dette gjorde at vi jobbet effektivt, og unngikk dobbeltarbeid. Scrum-metodikken skal derfor ha æren for at vi fikk utviklet et såpass ferdig produkt og som oppdragsgiver kan lansere når han måtte ønske.

De hyppige møtene i Scrum kan også sees på som en svakhet. Møtene krever mye tid til planlegging og administrativt arbeid, og gjorde at vi fikk mindre tid til utvikling. Det er også viktig å gi et korrekt timesestimat på arbeidsoppgaver da under- eller overvurdering kan føre til mye dødtid, eller høyt press på utviklerne som igjen fører til mangelfulle løsninger og teknisk gjeld. Vi klarte å estimere timer nøyaktig nok til at produktet ikke led av mye dødtid eller for stor arbeidsmengde. En siste fare ved Scrum er at prosjektets omfang raskt kan bli uhåndterlig stort utover utviklingsperioden dersom produkteier ikke klarer å begrense krav og forventinger til sluttproduktet. Dette var ikke et problem for oss da produkteier hadde teknisk kompetanse og dermed hadde realistiske forventninger til sluttproduktet.

Valg av teknologi

Expo ga oss tilgang til mange av enhetens innebygde funksjoner som kamera, mikrofon og varslingssenter, uavhengig av operativsystem. I tillegg gjorde expo at vi enkelt kunne bygge applikasjonen til .aab og .ipa filer som vi videre kunne laste opp til App Store og Google Play Store sine testprogrammer.

React Native har en stor brukerbase, og det har vært lett å finne løsninger på problemer som oppstod. Det finnes også et veldig godt utvalg av tredjeparts-biblioteker som har hjulpet oss med utviklingen, spesielt når det kommer til navigasjon og design. Likevel måtte vi i noen tilfeller benytte tredjeparts-biblioteker som det kunne være vanskelig å finne dokumentasjon på.

Django og PostgreSQL leverte all funksjonalitet vi hadde behov for i APIet og sørget for at vi kunne levere på alle krav oppdragsgiver hadde.

5.3 Administrative resultater

Dette delkapittelet vil ta for seg diskusjonen rundt de administrative resultatene vi fikk i delkapittel 4.3.

5.3.1 Fremdriftsplan

Fremdriftsplanen var strukturert og planlagt på en ryddig og hensiktsfull måte i form av en milepælplan. Vi valgte å gjøre det på denne måten da vi visste at det kunne komme endringer underveis. Siden én milepæl var avhengig av en annen, var rekkefølgen på milepælene nøye valgt ut. Det fremste eksempelet er notifikasjonssystemet som var et av målene i sprint 1. Denne milepælen la grunnlaget for alt av varslinger gjennom hele appen og var svært viktig å få unnagjort i begynnelsen.

Fra delkapittelet om fremdriftsplanen 4.3.1 var det et par avvik fra den opprinnelige planen. Det største avviket var milepæl 9: Meldingsfunksjon som gikk over både sprint 3 og sprint 4. Opprinnelig var det bare satt av tid til å gjennomføret milepælen i løpet av sprint 3, men på grunn av arbeid som ikke ble ferdig fra Infiniwells side ble milepælen utsatt. Da arbeidet ble ferdiggjort fra Infiniwells side, krevdes det små endringer fra vår side for å bli ferdig med milepælen. Samtidig ble det også gjort en liten endring av milepæl 14: Live data. Hovedfokuset i milepælen var å få implementere live-bølgeformer, men NEWS2-score ble i stedet lagt fokus på.

5.3.2 Scrum

Scrum var metodikken som ble brukt under utviklingen av produktet. Hovedgrunnen til at teamet ønsket å ta i bruk Scrum var fordi vi ønsket å være tilpasningsdyktige. Dette viste seg å være et godt valg da det utover i prosjektet skjedde uventede hendelser og endringer som måtte gjøres.

Selv om Covid-19-pandemien bød på flere utfordringer, klarte teamet å tilpasse seg uten at det gikk på bekostning av Scrum-metodikken. For best mulig utvikling av produktet valgte teamet å jobbe fysisk sammen da dette er normalt i et Scrum-prosjekt. Da var det enklere å spørre og få hjelp av

hverandre, samtidig som at man fikk følelsen av at man jobbet sammen som et team. Resten av de ulike Scrum-aktivtitene ble holdt digitalt da det var mer praktisk både for veileder og oppdragsgiver.

Teamet utførte Scrum slik det er beskrevet i delkapittel 2.5.2. Som nevnt i resultater (delkapittel 4.3.2) ble det avholdt ukentlige møter med mellom teamet og produkteier utenfor Scrum. Siden teamet i all hovedsak styrte produktkøen ble likevel arbeidsoppgavene i køen godkjent av produkteier under de ukentlige møtene. De ukentlige møtene med produkteier ble avholdt fordi vi ønsket å inkludere produkteier mest mulig under utviklingen, samtidig som at det forsikret at teamet hadde riktig fremgang.

Figur 4.14 viser et overordnet bilde på utførelsen av hver sprint. Timene planlagt i hver sprint var estimater teamet antok at ulike arbeidsoppgaver ville ta. Enkelte sprinter hadde større mål og som et resultat av dette hadde de et høyere antall planlagte timer. Man ser også at antall planlagte timer har en synkende trend. Dette kommer av at figuren kun viser timer planlagt til utvikling. Underveis ble flere og flere timer satt av til å dokumentere utviklingsprosessen, og å skrive denne rapporten. Flertallet av sprintene ble utført med stor suksess med unntak av sprint 3 der meldingssystemet ble overført inn i sprint 4. Dette er en av fordelene med å bruke Scrum, da metodikken tillater teamet å tilpasse seg situasjonen. Et annet godt eksempel som viser fordelen med Scrum er innføringen av NEWS2-score midt under sprint 4. Ved å bruke en generell plan med mulighet for store endringer underveis føler vi at vi har dratt alle fordelene fra Scrum. Trello var et viktig verktøy som holdt oversikt over alle arbeidsoppgavene for alle i teamet og fungerte som et Scrum-board. Dette gjorde det enkelt å se de prioriterte arbeidsoppgavene, hvem som gjorde hva og hvilke oppgaver som var igjen.

5.3.3 Timeforbruk

Teamet prøvde gjennom hele prosjektet å holde et jevnt timeforbruk. Det var uker der timeforbruket var lavere enn gjennomsnittet grunnet ferie eller annet opplegg fra NTNU. Dette ble tatt høyde for i begynnelsen av prosjektet og vi føler at vi har disponert tiden vår godt. Vi er svært fornøyde med hvor effektive vi har vært, selv om vi gikk over planlagte timer for planlegging og utvikling.

Prosjektet var videreført fra høstsemesteret og dette var en viktig bidragsyter til at oppstartsperioden og forarbeidet til prosjektet tok mindre tid enn forventet. Dette gjorde at planlegging og dokumentasjon i oppstartsperioden gikk bedre enn planlagt. En av hovedgrunnene til at arbeidstypen *planlegging* gikk 33 timer over det som var planlagt var planleggingen av de forskjellige brukertestene. Fra høstsemesteret ble appen testet på en enklere måte enn fra dette prosjektet. I dette prosjektet ble appen distribuert på en profesjonell måte ved å legge appen ut på både App Store (iOS) og Google Play Store (Android). Samtidig gikk det også mye tid på hvordan vi skulle planlegge oppsettet av brukertestene. En annen grunn er de hyppige møtene vi har hatt både med oppdragsgiver og veileder.

Vi er godt fornøyde med hvordan vi planla og strukturerte tiden til de ulike sprintene. De prioriterte sprintmålene ble utført først og ble satt av mest tid til. Til tider måtte vi også jobbe utenfor de vanlige arbeidstidene for å oppnå sprintmålene, og derfor var det enkelte uker der man gikk over det gjennomsnittlige timeantallet per uke.

Resten av timene som ble utført etter at applikasjonsutviklingen var ferdig gikk til dokumentasjon av systemet, hovedrapport og andre vedlegg. Grunnet godt kjennskap til oppgaven og systemet på forhånd, har tidsforbruket på dokumentasjon vært lavere enn forventet.

5.3.4 Refleksjon av gruppearbeidet

Som et team er vi svært fornøyde med hvordan vi har arbeidet sammen. Samarbeidet innad i teamet har vært profesjonelt og høyst effektivt. Hvert teammedlem tok ansvar fra start og arbeidsoppgavene ble fordelt likt og rettferdig. De administrative oppgavene var jevnt fordelt utover prosjektet. Selv om det var to ulike roller ved at det var en klientside og en tjenerside i teamet, mener vi at dett var en god og effektiv praksis for prosjektet. Ettersom vi delte oss i to gikk man ikke i veien for hverandre og man fikk fordypet seg i sitt respektive arbeidsområde.

Alle kommuniserte godt med hverandre og hjalp hverandre dersom det oppstod problemer. Terskelen for å spørre om hjelp fantes ikke. Ingen avgjørelser ble tatt uten at alle i teamet var enige. Dersom det var uenigheter ble disse løst gjennom diskusjoner og kompromisser. Teamet hadde flere teambuildingaktiviter i løpet av semesteret som styrket moral og samhold. Gjennom prosjektet har alle tilegnet seg ny kunnskap, lært nye teknologier og sist, men ikke minst hatt det gøy sammen.

6 Konklusjon og videre arbeid

6.1 Konklusjon

I løpet av semesteret har vi fått utviklet en ferdig applikasjon som komplimenterer den eksisterende løsningen til Infiniwell. Appen tar i bruk kjente elementer fra andre mobilapplikasjoner og bruker design som er kjent fra pasientmonitorer på sykehus.

Ved hjelp av React Navigation (ref. delkapittel 3.4.2) fikk vi benyttet kjente navigasjonsstrukturer som blant annet faner, navigasjons-barer og tilbakeknapper brukt i både iOS og Androids operativsystemer. Dette bidrar til å senke terskelen for å bruke appen for nye brukere, da mye vil være kjent fra andre applikasjoner. Disse navigasjonskomponentene kom ferdiglaget med navigasjonshandlinger som sveiping mellom sider og sveiping for å gå tilbake, som er med på å bedre brukeropplevelsen. Slike handlinger er også vanlige i både iOS og Android operativsystem.

React Native Paper (ref. delkapittel 3.4.1) lot oss bruke ferdiglagde komponenter designet etter Googles Material Design retningslinjer. Dette er komponenter som kan tilpasses og som skaper et helhetlig inntrykk når det kommer til brukergrensesnittet. Knapper og brytere fra React Native Paper er gjenkjennelige fra flere av Googles egne løsninger og skaper et kjent inntrykk for brukere.

For å svare på problemstillingen vi nevnte innledningsvis, må vi dele opp spørsmålet i flere deler. Når man lager en brukervennlig mobilapplikasjon handler det først og fremst om å ha et godt brukergrensesnitt og skape en god brukeropplevelse. Det finnes ingen fasit på hvordan man skaper god brukeropplevelse, men det finnes flere anerkjente retningslinjer og teorier som kan hjelpe oss godt på vei. For å utvikle et godt brukergrensesnitt har vi erfart at det er lurt å følge retningslinjer som Jakob Nielsens 10 heuristikker og Don Normans designprinsipper. Det vi har kommet frem til gjennom vår utvikling er at det er viktig å bruke gjenkjennbart design for å oppnå godt brukergrensesnitt. Farger og kontraster spiller også en viktig rolle for å sikre god lesbarhet og forståelse av knapper og beskrivelser. Når det gjelder brukeropplevelse har vi også her kommet frem til at det er viktig å bruke elementer brukere er vant med fra andre applikasjoner. Dette gjelder hovedsaklig handlinger og komponenter tilknyttet navigasjon.

For å lage en nyttig mobilapplikasjon må man ta hensyn til målgruppens behov. I vårt tilfelle har applikasjonens nytte hovedsaklig handlet om å tilgjengeliggjøre data. Mobilapplikasjonen vår har vist seg å gi god nytte da det er raskt og enkelt å åpne applikasjonen på mobilen og se de dataene man måtte ønske å se om en eller flere pasienter. For å oppnå denne nytten har det også vært viktig å gjøre dataene synlig og lettleselig som igjen dreier seg om et godt brukergrensesnitt. I tillegg handler det å skape nyttighet om å implementere etterspurt funksjonalitet. I vårt tilfelle var noe av grunnen til å lage en mobilapplikasjon, at helsepersonell skulle kunne motta notifikasjoner på sin mobil, dersom det skulle være noe viktig. Disse varslingene har åpnet for at systemet kan brukes på en overvåkningssentral, der det kan sendes ut varsel til relevant helsepersonell ved nødstilfeller, og dermed økt nyttigheten av appen.

Vår applikasjon er rettet mot helsepersonell. En brukervennlig applikasjon er viktig for enhver målgruppe, men for å rette oss mot helsepersonell har det vært viktig å vite hvordan personer innen helsesektoren er vant til å lese data om pasienter. Dette gjelder både farger, design og fagbegreper. Det har vist seg å være viktig å benytte samme farger og design som leger er vant til å se på pasientmonitorer på sykehus. Samtidig har det vært viktig å bruke de samme begrepene for pasientjournaler, målinger og feilmeldinger som leger er vant til å se og bruke til vanlig. For å rette en applikasjon mot helsepersonell er det altså kritisk å sette seg inn i deres domene.

6.2 Videre arbeid

6.2.1 Utvide testing av klienten

Til tross for at vi implementerte testing frontend i prosjektet vårt, vil en viktig del av videre utvikling være å utvide denne, samt å automatisere den. I tjenerkoden er all kode dekket av tester, men teamet oppdaget at det var langt mer krevende å teste mobilklienten. Testing er sentralt når man videreutvikler en løsning, for å sørge for at gammel funksjonalitet fortsatt virker. Automatisering av dette medfører at man kontinuerlig får sjekket at ny kode ikke skaper konflikt med gammel funksjonalitet.

Forskjellige typer tester burde implementeres. Vi implementere tester for flere av funksjonene brukt i prosjektet, og noen tester for å se at det visuelle rendres som det skal. Videre kan det være lurt å implementer tester som virkelig stresstester applikasjonen, samt tester som tester UI'en.

6.2.2 Automatisk utlogging

En mulig forbedring når det kommer til sikkerhet i applikasjonen er automatisk utlogging etter et gitt tidsintervall. Slik det er nå vil appen huske brukeren frem til brukeren logger ut manuelt. Infiniwells system bruker token, men serveren mangler endepunkter for å verifisere et tokens gyldighet. Et slikt endepunkt vil gjøre det mulig å kun lagre token på enheten som kjører applikasjonen. Tokenets gyldighet kan deretter verifiseres ved kjøring av appen. Dersom tokenet er gyldig, blir man logget inn og tokenet fornyes. Dersom tokenet har utgått vil man måtte logge inn på nytt. Oppdragsiver mente dette var relevant for ekstra sikkerhet når appen skal begynne å bli tatt i bruk, men at det nå var greit å la brukeren forbli innlogget ettersom det gir en mer sømløss brukeropplevelse.

6.2.3 Støtte for ulike pasientmonitorerings enheter

Oppdragsgiver påpekte at Infininwell i framtiden har lyst til å støtte flere forskjellige typer pasient-monitorerings enheter. Enhetene kan variere i målingsfrekvens og målingsparametre. Derfor lagde vi en guide som vedlegg i systemdokumentasjonen hvor vi forklarer hvor man må gjøre endringer i appen for å støtte nye enheter. Parametere for enheten kan endres relativt lett og kan endres slik at de varierer for hver pasient. Endring i målingsfrekvens behøver ingen endringer i appen for å fungere, men i ekstreme tilfeller kan være ønskelig å øke eller senke tidsrammene man velger i trend oversikten, noe som også er en svært simpel endring.

6.2.4 Fjerne muligheten for å endre og slette notat

Som nevnt i delkapittel 5.2.2.1 om pasientjournal, kan det være ulovlig i noen land å endre eller slette notater angående en pasient. Vi ga brukeren denne muligheten fordi vi så for oss at dersom en bruker hadde glemt å skrive noe eller skrevet noe feil, ville man lett kunne endre eller slette notatet. Som videre arbeid kan man enkelt fjerne denne funksjonaliteten i klient-koden og tjener-koden. Dette vil ikke påvirke annen funksjonalitet.

6.2.5 Push-notifikasjoner når parametere når farlige nivåer

Som nevnt i delkapittel 5.2.2.1, var det ønskelig å kunne få varslinger dersom et vitalt mål til en pasient går under eller over visse grenser. Dette krever endringer i tjener-koden og man trenger å vite normalintervallet for hvert vitale tegn. Å finne normalintervallet til et vitalt mål vil avhenge av pasienten og det vitale tegnet. Eksempelvis vil pulsen til enkelte pasienter variere når man tar pasientens alder og fysiske helse i betraktning. Derfor kan det være vanskelig å definere et bestemt intervall. Man kan benytte maskinlæring her for å detektere uregelmessige fall eller hopp. En løsning er også å la helsepersonell slå på varslinger for vitale tegn, og å la dem definere ett intervall.

6.2.6 Språk

Et av kravene som utgikk under utviklingen av applikasjonen var språk. Oppdragsgiver synes ikke språk var like kritisk sammenliknet med andre oppgaver, og denne oppgaven utgikk derfor. Ved akseptansetest kom det derimot frem at systemet skal tas i bruk i Afrika. Her vil det være nødvendig å ha franske oversettelser i både tjener og klient, da mye tekstlig informasjon hentes fra tjener.

6.2.7 SentioWeb

Etter akseptansetesten kom det frem at Infiniwell ønsket å begynne å bruke React for webapplikasjonen deres, SentioWeb. Hensikten med dette var å gjøre det sømløst å bytte mellom de ulike plattform-løsningene. Dette var aldri en del av oppgaven, men fordelen med produktet vi har utviklet er at den er skrevet i JavaScript slik at mye av koden vår er overførbar til webapplikasjoner.

6.2.8 Pasient app

Etter akseptansetesten ble muligheten for en kompanjong app som er rettet mot pasienter diskutert. Tanken var at man skulle tilby pasienter en oversikt over deres informasjon. Helsepersonell skulle også få muligheten til å kommunisere med pasientene sine på tvers av de to appene. Dette vil kreve en omstrukturering av APIet slik at pasienter kun har tilgang på et begrenset utvalg av endepunktene. Den sekundære appen kan eksistere innad i appen vi har utviklet og endre GUI utifra om brukeren som logger inn er helsepersonell eller pasient. Vi foreslo at en ryddigere løsning ville vært å heller lage to separate apper.

Referanser

[Amazon Web Services, 2021a] Amazon Web Services (2021a). What is continuous delivery? https://aws.amazon.com/devops/continuous-delivery/. Hentet 28.04.2021.

[Amazon Web Services, 2021b] Amazon Web Services (2021b). What is continuous integration? https://aws.amazon.com/devops/continuous-integration/. Hentet 28.04.2021.

[Arledge, 2014] Arledge, C. (2014). Filled-in vs. outline icons. https://cdr.lib.unc.edu/concern/masters_papers/6w924g35w.

[Arnesen, 2018] Arnesen, H. (2018). Ekg. https://sml.snl.no/EKG. Hentet 14.05.2021.

[Arnesen, 2020] Arnesen, H. (2020). Puls. https://sml.snl.no/puls. Hentet 14.05.2021.

[Beck et al., 2001] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001). Manifestet for smidig programvareutvikling. https://agilemanifesto.org/iso/no/manifesto.html.

[Chacon and Straub, 2014] Chacon, S. and Straub, B. (2014). Pro Git. Apress.

[Clearbridge, 2020] Clearbridge (2020). What is native mobile app development. https://clearbridgemobile.com/benefits-of-native-mobile-app-development/. Hentet 03.05.2021.

[Cooper et al., 2007] Cooper, A., Reimann, R., and Cronin, D. (2007). About Face 3: The Essentials of Interaction Design. Wiley.

[Danielsson, 2016] Danielsson, W. (2016). React native application development. https://www.diva-portal.org/smash/get/diva2:998793/FULLTEXT02.

[Digitaliseringsdirektoratet, 2013] Digitaliseringsdirektoratet (2013). Kvalitet på nett frå 2007 til 2011. https://www.digdir.no/media/472/download. Hentet 18.05.2021.

[Docs, 2021] Docs, M. W. (2021). Django introduction. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction. Hentet 07.04.2021.

[Expo, 2020] Expo (2020). Introduction to expo. https://docs.expo.io. Hentet 29.04.2021.

[Facebook, 2021] Facebook, I. (2021). Out-of-tree platforms. https://reactnative.dev/docs/out-of-tree-platforms. Hentet 10.05.2021.

[Fanebust, 2014] Fanebust, R. (2014). Hjerteovervåkning. https://www.legeforeningen.no/contentassets/00070974436841f6ac9f75262a5a6f6c/metodebok-2014-19-hjerteovervakning.pdf. Hentet 04.04.2021.

[Google, 2020] Google (2020). Dark theme. https://material.io/design/color/dark-theme.html#properties. Hentet 07.04.2021.

[Hansen, 2020] Hansen, M. K. (2020). Bevissthet. https://snl.no/bevissthet. Hentet 14.05.2021.

[Hauge, 2019] Hauge, A. (2019). Kroppstemperatur. https://sml.snl.no/kroppstemperatur. Hentet 14.05.2021.

[Hisdal, 2021] Hisdal, J. (2021). Blodtrykk. https://sml.snl.no/blodtrykk. Hentet 14.05.2021.

[International Organization for Standardization, 2018] International Organization for Standardization (2018). Ergonomics of human-system interaction — part 11: Usability: Definitions and concepts. https://www.iso.org/standard/63500.html. Hentet 04.04.2021.

[Jaatun, 2019] Jaatun, M. G. (2019). Ergonomics of human-system interaction — part 11: Usability: Definitions and concepts. https://infosec.sintef.no/informasjonssikkerhet/2019/10/smidig-utvikling-og-programvaresikkerhet/. Hentet 05.04.2021.

[Johannessen, 2007] Johannessen, M. (2007). Å leve med ansvar for andres liv. https://nhi.no/rettigheter-og-helsetjeneste/helsetjenesten/a-vare-lege/. Hentet 14.05.2021.

[Lid, 2020] Lid, I. M. (2020). Universell utforming. https://snl.no/universell_utforming. Hentet 06.04.2021.

[Malt and Grønmo, 2020] Malt, U. and Grønmo, S. (2020). Likert-skala. https://snl.no/Likert-skala. Hentet 12.05.2021.

[NHI, 2019] NHI (2019). Ekg. https://nhi.no/sykdommer/hjertekar/undersokelser/ekg/. Hentet 14.05.2021.

[Nielsen, 1990] Nielsen, J. (1990). Heuristic evaluation of user interfaces. https://dl.acm.org/doi/10.1145/97243.97281. Hentet 07.04.2021.

[Nielsen, 1993] Nielsen, J. (1993). Usability Engineering. Morgan Kaufmann Publishers Inc.

[Nielsen, 2000] Nielsen, J. (2000). Why you only need to test with 5 users. https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/. Hentet 13.05.2021.

[Nielsen Norman Group, 2021] Nielsen Norman Group (2021). Brukeropplevelse. https://www.nngroup.com/articles/definition-user-experience/. Hentet 06.04.2021.

[Norman, 2013] Norman, D. A. (2013). The Design of Everyday things. Basic Books.

[Regjeringen.no, 2020] Regjeringen.no (2020). Bedre digitale løsninger i helsesektoren. https://www.regjeringen.no/no/aktuelt/bedre-digitale-losninger-i-helsesektoren/id2696597/. Hentet 19.05.2021.

[restfulapi.net, 2020] restfulapi.net (2020). What is rest. https://restfulapi.net. Hentet 07.04.2021.

[Richard and LePage, 2020] Richard, S. and LePage, P. (2020). What are progressive web apps? https://web.dev/what-are-pwas/. Hentet 03.05.2021.

[Rossen, 2020] Rossen, E. (2020). Brukergrensesnitt. https://snl.no/brukergrensesnitt. Hentet 18.05.2021.

[Royal College of Physicians, 2017] Royal College of Physicians (2017). National early warning score (news) 2. https://www.rcplondon.ac.uk/projects/outputs/national-early-warning-score-news-2. Hentet 06.05.2021.

[Scharoun and Bryden, 2014] Scharoun, S. M. and Bryden, P. J. (2014). Hand preference, performance abilities, and hand selection in children. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3927078/.

[Scrum.org, 2020] Scrum.org (2020). What is scrum? https://www.scrum.org/resources/what-is-scrum. Hentet 05.04.2021.

[SIGCHI, 2014] SIGCHI (2014). Menneske-maskin-interaksjon. https://web.archive.org/web/20140817165957/http://old.sigchi.org/cdg/cdg2.html#2_1. Hentet 07.04.2021.

[SINTEF, 2007] SINTEF (2007). Brukeropplevelse. https://web.archive.org/web/20071222103843/http://www.sintef.no/content/page1____3070.aspx. Hentet 06.04.2021.

[St. Olavs, 2021] St. Olavs (2021). Measurement of vital signs in the emergency department. https://stolav.no/seksjon/innovasjon/Documents/Komprimert_Measurement%20of%20Vital%20Signs%20in%20the%20Emergency%20Department_%20Overview%20of%20needs_%20Autosk%C3%A5r.pdf.

[The PostgreSQL Global Development Group, 2021] The PostgreSQL Global Development Group (2021). Postgresql - about. https://www.postgresql.org/about/. Hentet 07.04.2021.

[uutilsynet, 2014] uutilsynet (2014). Apper er omfattet av kravene. https://www.uutilsynet.no/regelverk/apper-er-omfattet-av-kravene/763. Hentet 06.04.2021.

[w3c, 2021] w3c (2021). W3c - about. https://www.w3.org/Consortium/. Hentet 06.04.2021.

[W3C, 2021] W3C (2021). Web content accessibility guidelines (wcag). https://www.w3.org/WAI/standards-guidelines/wcag/. Hentet 06.04.2021.

[Åshild Odden Miland, 2018] Åshild Odden Miland (2018). Oksygenmetning. https://sml.snl.no/oksygenmetning. Hentet 14.05.2021.

[Åshild Odden Miland, 2021] Åshild Odden Miland (2021). Respirasjonsfrekvens. https://sml.snl.no/respirasjonsfrekvens. Hentet 14.05.2021.

Vedlegg

Vedlegg A: Visjonsdokument

Vedlegg B: Kravdokumentasjon

Vedlegg C: Prosjekthåndbok

Vedlegg D: Systemdokumentasjon

Vedlegg E: Tester

Vedlegg F: Kontrakter

NTNU

Kunnskap for en bedre verden