Høgskulen på Vestlandet

BACHELOROPPGAVE

Redesign / implementering av applikasjonen Program.no

Redesign / implementation of the application Program.no

Gruppe D16

Martin Aas Eliassen

Steffen Lampe

DAT191

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

Dataingeniør

Veileder: Richard Kjepso

Innleveringsdato: 23.05.2022

Jeg bekrefter at arbeidet er selvstendig utarbeidet, og at referanser/kildehenvisninger til alle kilder som er brukt i arbeidet er oppgitt, *jf. Forskrift om studium og eksamen ved Høgskulen på Vestlandet, § 10.*

Rapportens tittel:	*Dato:*
Redesign / implementering av applikasjonen Program.no	*23.05.2022*
Forfatter(e):	*Antall sider u/vedlegg: 42*
Martin Aas Eliassen Steffen Lampe	*Antall sider m/vedlegg: 44*
Studieretning: Dataingeniør	*Antall disketter/CD-er: 0*
Kontaktperson ved studieretning: Richard Kjepso	*Gradering:* Ingen
Merknader:	

Oppdragsgiver:	*Oppdragsgivers referanse:*
Styreportalen AS	
Oppdragsgivers kontaktperson:	*Telefon:*
Stian Sømoen	91319131

Sammendrag:

Program.no er en mobilapplikasjon for forskjellige kulturarrangementer, og det er i dag over 10 000 aktive brukere av applikasjonen. I dette prosjektet ble det utviklet en mobilapplikasjon i React Native med en Firebase Firestore database. Utgangspunktet for prosjektet var at Styrepotalen AS ønsket en mer moderne mobilapplikasjon enn hva den daværende løsning var. Produktet som har blitt utviklet har gitt en prototype som gir et godt grunnlag for videreutvikling. Gjennom tett dialog med oppdragsgiver har gruppen lagt til ønskede funksjoner og design.

Stikkord:

React Native	Design	Mobilapplikasjon

FORORD

Denne rapporten beskriver bachelorprosjektet «Redesign / implementering av applikasjonen Program.no» og arbeidet rundt det. Prosjektet omhandler en mobilapplikasjon bygget fra grunn av med utgangspunkt i mobilapplikasjonen, Program.no, til Styreportalen AS. Prosjektgruppen består av Martin Aas Eliassen og Steffen Lampe.

Gruppen ønsker å rette en stor takk til veileder Richard Kjepso for gode og konstruktive tilbakemeldinger, og oppfølging gjennom prosjektperioden.

Gruppen ønsker også å takke oppdragsgiver Stian Sømoen for godt samarbeid og for muligheten til å jobbe med dagsaktuelle teknologier.

INNHOLDSFORTEGNELSE

1 INNLEDNING

1.1 Kontekst

Grunnlaget for bachelorprosjektet er at Styreportalen ønsker en mer moderne løsning for applikasjonen Program.no. Dette er en mobilapplikasjon hvor organisasjoner kan annonsere sine arrangementer i en aktivitetskalender, som brukere kan se og lagre som favoritt. Det har blitt gjort større fremsteg i teknologi og rammeverk for utvikling av applikasjoner de siste årene, noe som legger bedre til rette for videre arbeid. Dagens løsning er utdatert, og på grunnlag av dette er det et ønske om å ta i bruk nyere teknologi som er mer oppdatert funksjonelt sett, og optimalisert for dagens bruk. Videre vil dette også gi bedre verdi for brukerne.

1.2 Motivasjon

Motivasjonen for prosjektet er at dagens løsning ikke tilrettelegger for å enkelt kunne utvide applikasjonen med flere og mer moderne funksjoner og løsninger. Dette kan gi dårligere brukeropplevelse og skape frustrasjon.

Dagens løsning har begrenset med funksjoner og et gammelt design, noe Styreportalen ikke er tilfreds med. Det er derfor et behov for å redesigne og implementere applikasjonen med nyere teknologi og rammeverk. Gruppen vil derfor gi applikasjonen et mer moderne design med forbedrede funksjoner, som gir brukerne en bedre opplevelse med applikasjonen og nytteverdi for bedriften.

1.3 Prosjekteier

Prosjekteier er Styreportalen, representert av Stian Sømoen. Styreportalen er en liten bedrift som utvikler programvarer for frivillige organisasjoner, hvor blant annet Program.no er ett av hovedproduktene deres. Styreportalen har fullt eierskap for applikasjonen og står for blant annet databasen.

1.4 Problembeskrivelse og mål

Program.no er en aktivitetskalender hvor organisasjoner kan få annonsert sine arrangementer, som konserter, foredrag og andre typer arrangementer. Designet på dagens løsning er over 5 år gammelt, og inneholder få muligheter for brukerne. Applikasjonen er ikke godt tilrettelagt for brukere med iPhone-enheter, hvor noen brukere opplever at applikasjonen slutter å fungere. Brukeren må da avinstallere applikasjonen og laste den ned på nytt. Applikasjonen skalerer seg ikke korrekt i forhold til utforming av skjermen til de nyere utgavene av iPhone, da de har sensorer plassert i midten på øverste del av skjermen.

Målet for dette prosjektet er å lage en applikasjon som gir et mer moderne inntrykk av Program.no ved bruk av rammeverket React Native. Styreportalen ønsker en applikasjon som gir en rask respons når det utføres handlinger fra brukeren, og et design som er mer elegant og moderne enn dagens løsning. Applikasjonen skal gi en god oversikt over tilgjengelige arrangementer, og det skal være mulig å søke etter arrangementer. En bruker skal ha mulighet til å opprette en bruker-profil ved bruk av mailadresse. Brukeren kan da lagre sine favorittarrangementer, og vil kunne motta varsler om for eksempel endringer og påminnelser for de arrangementene.

1.5 Oppbygging av rapporten

Rapporten er bygget opp av flere kapitler som går gjennom prosessen ved å utvikle og implementere løsningene for applikasjonen Program.no.

Kapittel 1 gjør rede for hvem Styreportalen er, og bakgrunnen for prosjektet. Kapittelet inneholder også problembeskrivelsen og målet for applikasjonen som skal utvikles.

I Kapittel 2 vil prosjektet ta for seg det mer praktiske for utviklingen av applikasjonen. Her blir det presentert funksjonalitet og design i dagens løsning, samt prosjekteiers opprinnelige ønsker og krav til applikasjonen. Videre inneholder kapittelet informasjon om tilgjengelige ressurser som skal brukes, og begrensninger for prosjektet.

Kapittel 3 gir forskjellige diskusjoner for valgte løsninger og tilnærminger til prosjektet. Det diskuteres også om hvilken prosjektmetodikk gruppen har valgt å benytte seg av.

Kapittel 4 gir en bedre forståelse rundt løsningens framgangsmåte og løsningsdesign. Her vil det bli gått mer i detalj om løsningen som gruppen har utviklet.

Kapittel 5 beskriver og drøfter metodikken som er benyttet for å evaluere resultatet.

Kapittel 6 inneholder diskusjon rundt prosjektets gang, hva som ble oppnådd og om det står til forventningene ut ifra prosjektbeskrivelsen. Her vil avvik fra planer og oppsatte mål bli gjennomgått.

Kapittel 7 trekker fram konklusjoner i forhold til prosjektets problemstilling, og gruppens forslag til videre arbeid for applikasjonen.

Kapittel 8 inneholder alle referanser som er benyttet i forbindelse med rapportskrivingen.

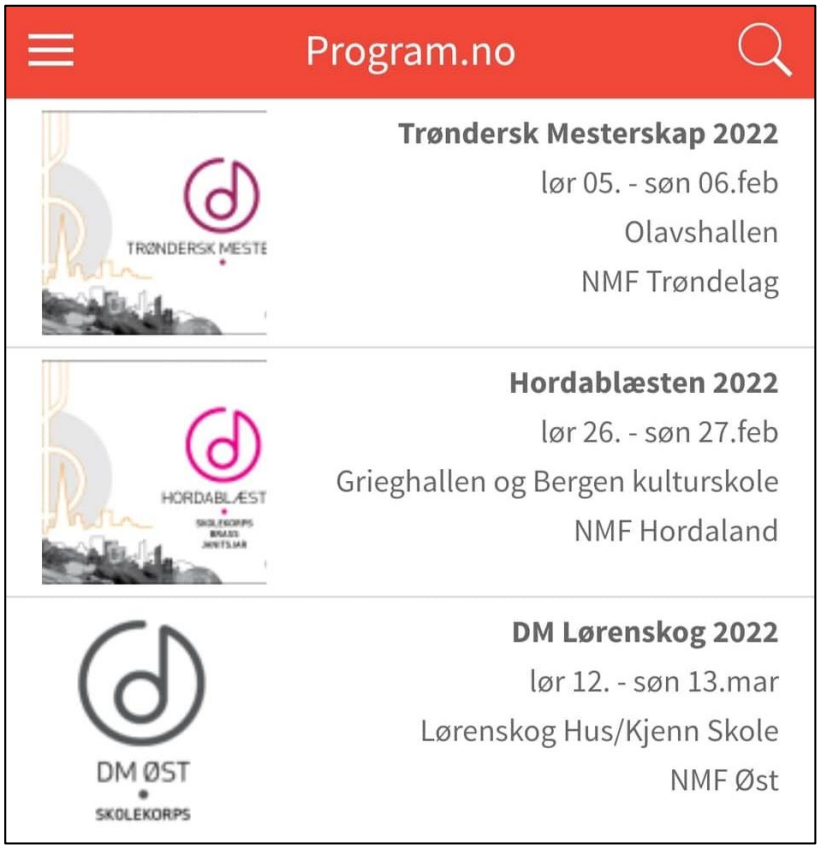
Kapittel 9 lister vedlegg som tilhører rapporten.

2 PROSJEKBESKRIVELSE

2.1 Praktisk bakgrunn

2.1.1 Tidligere arbeid

Bedriften har allerede en versjon av applikasjonen Program.no, som dette prosjektet vil bygge videre på og forbedre. Som nevnt tidligere har applikasjonen noen begrensninger og problemstillinger som prosjektet vil ta hensyn til og finne en bedre løsning på, for eksempel at applikasjonen slutter å fungere for enkelte brukere med iPhone.



Figur 1 - Dagens listeløsning over arrangement

Applikasjonen har også en webapplikasjon som fungerer på samme måte som den mobile applikasjonen, men med forskjellige brukergrensesnitt. Den mobile- og webapplikasjonen bygger på samme utforming, som farger og logo.

På grunnlag av egeninteresse og tilbakemeldinger fra brukere har derfor bedriften et ønske om å finne nye løsninger på tidligere nevnt problemer med applikasjonen og implementere

et nytt design. Tilbakemeldingene og forslag fra oppdragsgiver danner hovedgrunnlaget for prosjektets utviklingsdel.

2.1.2 Initielle krav

De initielle kravene er at brukere skal kunne se en liste over arrangement og deretter kunne gå inn på enkelte arrangement for å se programmet og mer detaljer rundt arrangementet. Det vil også være fokus på å finne en bedre løsning på skaleringen og utvikle en bedre meny. Som nevnt tidligere er dette et problem hos nyere smarttelefoner i dagens løsning. Det er også satt et mål om å forbedre noen av de eksisterende funksjonene i applikasjonen. Videre er det også ønskelig at brukere skal kunne ta i bruk applikasjonen uten internett for å se arrangement de har lagret/favoritt merket. Brukere skal også kunne opprette en profil i applikasjonen slik at en kan se favorittmerket arrangement på forskjellige enheter, uten at en mister data dersom en tar i bruk en ny enhet.

Med designet er kravet at det skal se mer morderne ut, siden det gamle designet er over 5 år gammelt. Det er også et krav at designet skal være likt som webapplikasjonen, slik at det det blir et samspill i hvordan de to applikasjonene fungerer og ser ut. Brukervennlighet er også et viktig krav, slik at brukere ikke må bruke tid på opplæring i bruk av applikasjonen.

Som nevnt tidligere vil også det å migrere deler av dagens databaseløsning til en ny databaseløsning være et viktig krav, siden dagens løsning er utdatert og setter begrensninger for funksjoner som kan implementeres.

2.1.3 Initiell løsnings-idé

Løsnings-ideen vil ha bakgrunn i forslag som oppdragsgiver har utarbeidet i forkant. Oppdragsgiver har uttrykt at de ikke er tilfreds med dagens applikasjon og verktøyet den er bygget på. På bakgrunn av det skal den nye løsningen utvikles helt fra grunn av i et nytt verktøy, og kommunisere med en annen database.

Det legges vekt på at designet i applikasjonen skal gjenspeile designet som vil bli brukt på nettsiden deres. Nettsiden er det en annen bachelorgruppe som skal utvikle, så det arrangeres felles møter for å oppnå lignende design både i applikasjon og på nettsiden.

Videre er det ønsket at applikasjonen kan ta i bruk mer eller mindre samme kode både for iPhone- og Android-enheter, slik at applikasjonen kan nå ut til flest mulig.

Funksjonene i dagens løsning er ønsket videre til den nye versjonen, med noen unntak. Om en bruker skal gjøre et arrangement til favoritt, vil dette knyttes til brukerprofilen og ikke til enheten slik som i dag. Dette er for at brukeren kan logge seg inn på en annen enhet og fremdeles ha oversikt over sine favorittarrangementer.

2.2 Avgrensninger

Det vil oppstå noen avgrensninger i prosjektet, og noen allerede fra starten. De avgrensningene som vil prege prosjektet mest er kunnskapen om teknologien som oppdragsgiver har satt som krav, samt testing av produkt ut mot reelle brukere.

2.2.1 Teknologien

Oppdragsgiver har satt krav om React Native skal brukes under utvikling av applikasjonen, og Firestore fra Firebase som database. Ved prosjektets start er ikke gruppens kunnskap om teknologiene tilstrekkelig for å kunne sette i gang utviklingen. Det er helt nye teknologier for begge medlemmene i gruppen, og det vil da brukes en del tid i starten av prosjektet for å tilegne seg kunnskap om teknologien og verktøyene. Selv om det er ny teknologi for gruppen, vil tidligere erfaringer fra studiet gi et godt utgangspunkt da logikken og forståelsen har mange likheter.

Det vil også bli gitt en startkode til gruppen fra oppdragsgiver. Denne inneholder de funksjonene som gir brukerne mulighet til å opprette en brukerprofil. Dette gjøres på bakgrunn av at oppdragsgiver ønsker at tiden skal fokuseres mer på design og andre funksjonelle egenskaper.

2.2.2 Evaluering

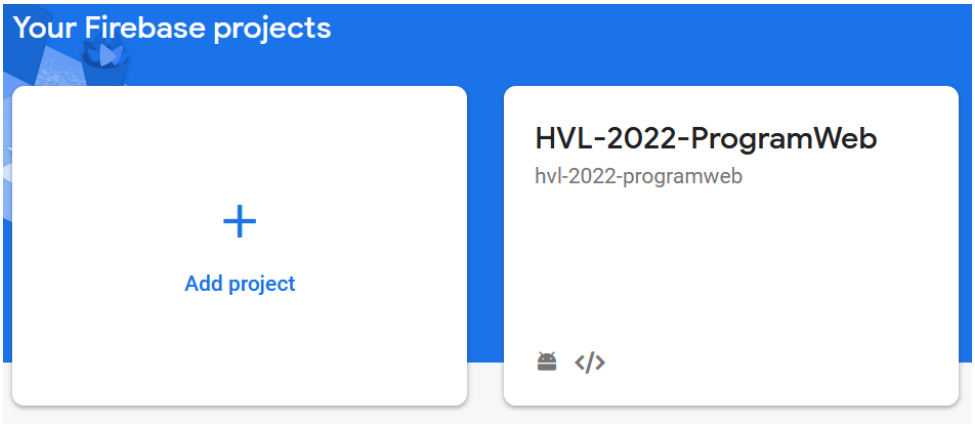
Applikasjonen vil ikke bli testet av de reelle brukerne til oppdragsgiver i første omgang. Senere i utviklingen kan det oppstå muligheter for dette, men det er uttrykt at gruppen generelt sett må forholde seg til oppdragsgiver. Det er og mulighet for gruppen å laste ned prototyper på egen telefon og få tilbakemelding fra andre ved å vise fram applikasjonen, for eksempel til intern veileder eller medstudenter.

2.3 Ressurser

Under prosjektet er det hentet ressurser både for å kunne utvikle applikasjonen, men og for å få bedre forståelse av det mer teoretiske.

2.3.1 Ressurser for utvikling

Gruppen har fra før av hver sin bærbare PC som er kraftig nok til å utvikle på. Det blir kodet i rammeverket React Native, med Visual Studio Code som teksteditor, og Android Studio som emulator for å kunne se i sanntid hvordan applikasjonen vil se ut. Bedriften stiller med tilgang til deres database i Firebase Firestore som gruppen kan knytte applikasjonen opp mot, og deretter hente ut data som bedriften har lagt inn.



Figur 2 – Firebaseprosjekt

For informasjon og veiledning fra bedriften er det også opprettet en egen kanal via kommunikasjonsplattformen Slack. Her har gruppen mulighet til å søke veiledning og stille spørsmål til oppdragsgiver. For å få hjelp til mer tekniske ting, kan det her også stilles spørsmål som ville bli henvendt videre til utviklere som jobber hos oppdragsgiver.

2.3.2 Litterære ressurser

Som nevnt tidligere vil det være behov for gruppen å tilegne seg kunnskap om teknologiene. Her vil det bli innhentet kunnskap via dokumentasjon på nett for kodespråket og databasen. Til det mer teoretiske for rapporten vil gruppen anvende kunnskap som er gitt fra forelesninger og emner tidligere i studiet.

2.4 Litteratur om problemstillingen

Problemstillingen for prosjektet er å lage et mer moderne design samt forbedre dagens løsninger. Derfor er det en del å forholde seg til med tanke på brukeropplevelsen. Hoekman Jr (2011, s.36) hevder at bare fordi en applikasjon har mange flere funksjoner enn andre applikasjoner i et marked, så er det ikke et bedre produkt og flere funksjoner kan også forvirre brukere. Det vil være viktig for både oppdragsgiver og brukere at applikasjonens funksjoner er nødvendige og at gruppens løsninger ikke er utviklet bare for å «fylle» ut applikasjonen.

Designet til applikasjonen vil også ha en stor betydning for brukere. Ifølge Hoekman Jr (2011, s.36) vil en bruker raskt knytte seg til den første nettsiden eller løsningen som de kan tolerere og ta i bruk, uten mye frustrasjon. Derfor vil det være viktig å finne en enkel og oversiktlig løsning på designet, slik at brukere ikke blir frustrert over hvordan de kommer seg fram i applikasjonen. Et mer moderne og oversiktlig design gjør da at brukere kanskje vil foretrekke å bruke applikasjonen fremfor konkurrenters løsning.

Prosjektets løsning på design og funksjoner burde ut ifra dette være fokusert på oversiktlighet, responstid, og kun relevante funksjoner. Dette vil også gjenspeile tilbakemeldinger som oppdragsgiver har fått fra brukere av dagens løsning. Et viktig mål for gruppen er å løse de forskjellige problemstillingene som ble diskutert i dette delkapittelet, og gi oppdragsgiver noe som de enkelt kan videreutvikle og muligens ta i bruk.

3 DESIGN AV PROSJEKTET

3.1 Forslag til løsning

Når det skal utvikles en applikasjon til mobiltelefoner finnes det ulike alternativer. I punktene under blir det gjennomgått noen av de alternativene som er relevant for en applikasjon som Program.no.

3.1.1 Alternativ løsning 1 – Webapplikasjon

En webapplikasjon er ganske lik som en tradisjonell nettside, men er mer interaktiv og funksjonell. Det utvikles med HTML, CSS og JavaScript, slik som vanlige nettsider. Webapplikasjoner er en enkelt løsning når man ikke vil sette krav for hvilke operativsystem det skal utvikles for, og man trenger ikke å laste ned eller installere noe for å få tilgang til applikasjonen.

Webapplikasjoner er også lettere å utvikle i forhold til andre alternativer, og tar dermed som regel kortere tid å utvikle sammenlignet med native mobilapplikasjoner. Om applikasjonen trenger en oppdatering kreves det ingen handling fra brukerens side da dette skjer automatisk når man laster webapplikasjonen inn på nytt. Det er også enklere å lansere slike applikasjoner da de ikke må gjennom noen form sikkerhetssjekk hos for eksempel Google Play eller App Store.

3.1.2 Alternativ løsning 2 – Native applikasjoner

Native applikasjoner menes som en applikasjon som er installert på en enhet og kan åpnes via et bildeikon på skjermen. Det er vanlig at disse bli installert via et applikasjonsmarked som for eksempel Google Play (Android) eller App Store (iPhone). En native applikasjon kan ta i bruk alle sensorer og egenskaper mobiltelefonen har, slik som kamera, GPS, kontaktlister og så videre.

Native applikasjoner oppleves ofte som veldig raske og responsive for brukerne. Siden applikasjonen er lastet ned lokalt hos brukeren, er det mindre som trenger å lastes inn via internett når man åpner applikasjonen. I gruppens prosjekt vil det kun

være nødvendig å laste ned, og/eller oppdatere listen med arrangementer, noe som vil føre til at applikasjonen vil være rask.

Det kreves derimot at man må ha to ulike kodedatabaser, da det må gjøres noen justeringer av kildekoden for at applikasjonen skal kunne kjøre på tvers av ulike operativsystemer. Dette gjør at utviklingen kan bli mer tidkrevende. Det samme vil også gjelde når man ønsker å oppdatere applikasjonen.

3.1.3 Diskusjon av alternativene

For å oppnå best mulig resultat av applikasjonen Program.no er det naturlig for gruppen å gå for alternativ løsning 2. Dette er både på bakgrunn av ønsker fra oppdragsgiver og for at Program.no skal fungere best mulig etter gitte krav. Ved å velge en native applikasjon gir det mulighet for å ta i bruk mobiltelefonen sine egenskaper slik som GPS for å filtrere etter nærmeste arrangementer. Om brukeren har lastet ned arrangementer i applikasjonen gir det også muligheten til å bruke applikasjonen på en god måte selv når man ikke har internett. Som nevnt tidligere vil applikasjonen også oppleves som være rask og responsiv.

Det finnes dog noen ulemper med native applikasjoner. For eksempel vil oppdatering av applikasjonen være mer tidkrevende da det må tas hensyn til at applikasjonen skal ha mulighet for å kjøre på ulike operativsystemer. Dette ser gruppen og oppdragsgiver på som en liten pris å betale for å få en applikasjon som vil møte kravene og brukeropplevelsen en native applikasjon kan by på.

3.2 Valgt løsning

I dette prosjektet ble en native applikasjon valgt. Bakgrunn for valget er både et ønske fra oppdragsgiver og at det gir et gunstig forbedringspotensial for en del av funksjonene som allerede eksisterer i dagens løsning. Med tanke på gruppens tidligere erfaring fra studiet vil ikke læringskurven for å utvikle native applikasjoner være like bratt som ved andre løsninger. Dette er ideelt for et prosjekt med relativt kort tid.

3.3 Valg av verktøy

Visual Studio Code

Som teksteditor vil gruppen bruke Visual Studio Code. Denne har innebygd terminal, og snakker godt sammen med de andre verktøyene.

Android Studio

Android Studio vil bli brukt som emulator for å opprette en virtuell enhet. Her kan man kjøre koden i sanntid, og endringer oppdateres i applikasjonen fortløpende.

React Native

React Native er et rammeverk for JavaScript, som er designet for å lage native applikasjoner. Det er et stort brukersamfunn, og det finnes mange moduler som kan gjøre utviklingsprosessen mindre tidkrevende. Rammeverket har også delt kodebase, som da gir mulighet for å anvende koden på både Android og iPhone uten store justeringer av kildekoden. Dette er et sterkt ønske fra bedriften.

Firebase Firestore

Som database vil Firebase Firestore bli brukt, da dette er ønske fra oppdragsgiver. Firebase har blant annet en del funksjoner for brukerstatistikk, og lokal lagring på telefonen selv når brukeren ikke har internett.

Git

Git er et gratis versjonskontrollverktøy for å behandle og dele kildekode. Verktøyet har mulighet for å holde rede på endringer som er gjort. Man kan jobbe med ulike «branch» i Git, som da gjør at flere utviklere kan jobbe parallelt på samme prosjekt. Git er også gunstig for å kunne dele koden med oppdragsgiver underveis, og overføre eierskapet etter endt prosjekt.

Slack

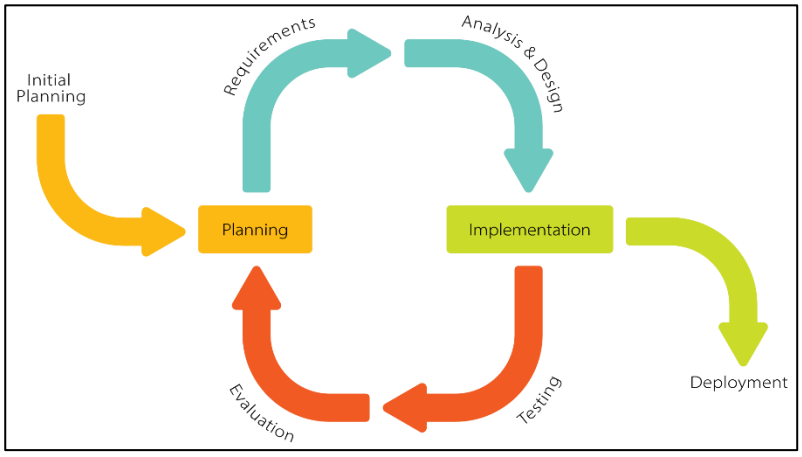
Slack er et verktøy som tilbyr kommunikasjon via meldinger og videosamtaler. Det er også mulighet for å dele relativt store filer (1GB) via Slack.

3.4 Prosjektmetodikk

3.4.1 Utviklingsmetodikk

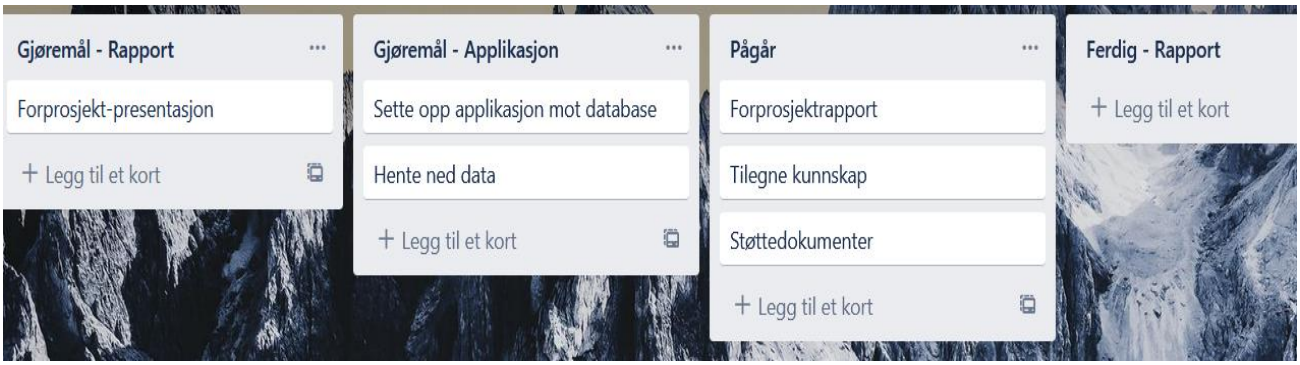
I dette prosjektet er iterativ og inkrementell utvikling sammen med Kanban valgt som utviklingsmetodikk. Dette er en veldig åpen utviklingsmetodikk, siden det er veldig få begrensninger, og kan enkelt tilpasse endringer og uforutsette hendelser, som er uunngåelig i et utviklingsprosjekt. Kanban bruker en såkalt «Kanban-tavle» som visualiserer arbeidet som skal gjøres, gjøres nå og er gjort (Radigan, u.å.). Det skal også være nevnt at det er vanlig å bruke en slik tavle i Scrum for å kartlegge oppgaver. Det er derfor begrenset for hvor mange oppgaver som kan gjøres samtidig, spesielt i et prosjekt med få teammedlemmer. I motsetning til utviklingsmetodikken Scrum, som krever flere roller og er ment for større grupper, så gir det mer frihet i teamet (Drumond, u.å.). En ulempe med Kanban er at det ikke er like godt planlagt over hvilke oppgaver en skal gjøre, siden en trekker en oppgave fra tavlen og ikke får tildelt. Dette kan også føre til at det oppstår ujevn fordeling av oppgaver i gruppen, siden oppgavene varierer noe i størrelse.

Iterativ og inkrementell utvikling går ut på at et team deler opp større oppgaver til mindre og mer håndterbare oppgaver til iterasjoner (Martin, 1999). Iterasjonene burde ikke gå over lang tid, men heller brutt ned til en til to uker per iterasjon (Martin, 1999). Etter hver iterasjon kan gruppen vise fram utførte oppgaver til oppdragsgiver, som da kan gi nyttige tilbakemeldinger og eventuelle forbedringer som ønskes. Denne utviklingsmetodikken er ideell for dette prosjektet, siden et nytt design er hovedfokuset.



Figur 3 – Iterativ og inkrementell utviklingsmodell (Krupadeluxe, 2021)

Kanban med iterativ og inkrementell utvikling har ikke ingen spesifikke krav om roller, slik som andre utviklingsmetodikker har, men heller mer åpent for å bruke roller som allerede er brukt i teamet (Radigan, u.å.). Som da er ideelt for dette prosjektet. I Scrum er det flere spesifikke roller å fylle, og det er ikke like åpent for hvilke roller en kan ha. Derfor er det ikke like ideelt å velge Scrum på grunnlag av dette (Drumond, u.å.). Åpne roller i Kanban kan også gi litt problem med tanke på hvem som styrer gruppen, og hvem som har ansvaret for hva.



Figur 4 – Kanban-tavle

Kanban sammen med iterativ og inkrementell utvikling som utviklingsmetodikk gjør det også mulig at når teamet er ferdig med en oppgave, for eksempel en søkefunksjon i applikasjonen, så kan det vises fram til oppdragsgiver fortløpende og få tilbakemeldinger på resultatet. I Scrum er dette også mulig, men det skjer gjennom såkalte «sprinter» (Drumond, u.å.).

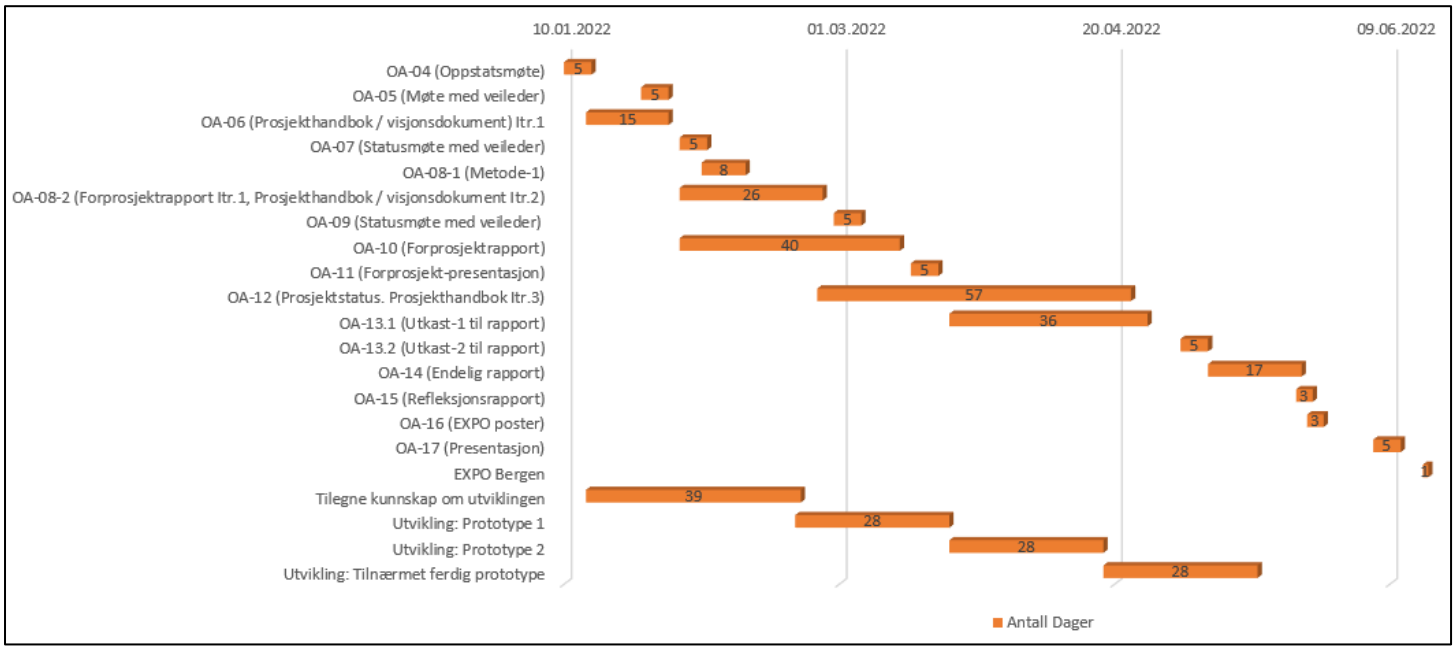
Scrum vil ikke bli brukt i dette prosjektet siden, som diskutert tidligere, den metodikken vil være mer egnet for større team og har mange flere roller å fylle, noe som ikke egner seg for denne gruppen. Scrum har også veldig spesifikke roller, noe som gjør at det vil være litt mer unaturlig dynamikk i gruppen, siden gruppen har brukt mer åpne roller under tidligere prosjekt i studiet.

3.4.2 Prosjektplan

Prosjektplanen ble utviklet som et GANTT-skjema i Microsoft Excel. Planen har som hensikt å gi en oversikt over hvilke oppgaver som skal utføres i ukene/dagene fra prosjektstart til prosjektslutt.

Aktivitet	Planlagt Start	Planlagt Slutt	Antall Dager	Ansvarlig Person
OA-04 (Oppst	10.01.2022	14.01.2022	5	D16
OA-05 (Møte	24.01.2022	28.01.2022	5	D16
OA-06 (Prosje	14.01.2022	28.01.2022	15	D16
OA-07 (Status	31.01.2022	04.02.2022	5	D16
OA-08-1 (Met	04.02.2022	11.02.2022	8	D16
OA-08-2 (Forp	31.01.2022	25.02.2022	26	D16
OA-09 (Status	28.02.2022	04.03.2022	5	D16
OA-10 (Forprc	31.01.2022	11.03.2022	40	D16
OA-11 (Forprc	14.03.2022	18.03.2022	5	D16
OA-12 (Prosje	25.02.2022	22.04.2022	57	D16
OA-13.1 (Utka	21.03.2022	25.04.2022	36	D16
OA-13.2 (Utka	02.05.2022	06.05.2022	5	D16
OA-14 (Endeli	07.05.2022	23.05.2022	17	D16
OA-15 (Reflek	23.05.2022	25.05.2022	3	D16
OA-16 (EXPO	25.05.2022	27.05.2022	3	D16
OA-17 (Preser	06.06.2022	10.06.2022	5	D16
EXPO Bergen	15.06.2022	15.06.2022	1	D16
Tilegne kunns	14.01.2022	21.02.2022	39	D16
Utvikling: Prot	21.02.2022	20.03.2022	28	D16
Utvikling: Prot	21.03.2022	17.04.2022	28	D16
Utvikling: Tiln	18.04.2022	15.05.2022	28	D16

Tabell 1 - Aktiviteter



Tabell 2 – Prosjektplan

I prosjektplanen jobbes det fram til forskjellige prototyper/versjoner av applikasjonen, noe som passer godt for iterativ og inkrementell utvikling hvor det er kontinuerlig arbeidsflyt og bruk av en Kanban-tavle. Selv om det jobbes fram til ulike prototyper så er det likevel ikke et krav at enkelte funksjoner skal være ferdigutviklet, men heller et mål å se fram til. Mellom hver prototype/versjon blir det tatt for seg flere oppgaver med applikasjonsutviklingen, så oppgavene blir tatt fra en «kø» av oppgaver. Etter hver prototype er det som mål å ha utviklet eller forbedret flest mulig funksjoner i applikasjonen.

I perioden med posten «Tilegne kunnskap om utvikling» vil det å sette seg inn i den nye teknologien og rammeverket være hovedfokuset. Dette er nye teknologier for begge i gruppen og må derfor settes det av en del tid for å kunne bruke de både individuelt og sammen. Dette skjer da parallelt med møter med oppdragsgiver for å få et bilde av hvordan den nye applikasjonen skal se ut og fungere. Perioden går over 42 dager, men tilegning av kunnskap skjer samtidig som en del andre poster og innleveringer i et annet emne.

I utviklingsperioden «Prototype 1» er fokuset på å sette kode opp mot database og få hentet ned data fra den, som gruppen videre kan begynne å bruke i designet til applikasjonen. Denne perioden vil det mest grunnleggende av koding på applikasjonen utføres, og ingen større funksjoner blir lagt til før dette er på plass.

I utviklingsperioden «Prototype 2» er det designet og de større funksjonene som å kunne søke etter arrangement, vise program for arrangement, og navigering mellom sider som er målet å bli ferdig med. Denne perioden vil være den som det blir lagt ned mest tid på og jevnlige møter med oppdragsgiver gir gruppen en god pekepinn, spesielt med tanke på design.

I utviklingsperioden «Tilnærmet ferdig prototype» vil det for det meste være å finjustere funksjoner og designet. Her vil det ikke bli gjort noen store implementeringer eller laget nye design, men heller fikse på det som allerede er lagt til eller legge til mindre funksjoner dersom gruppen har tid. I denne perioden er det også flere innleveringer og en eksamen gruppen må fokusere på, som gjør tiden veldig knapp.

3.4.3 Risikovurdering

I risikovurderingen gjøres det rede for ulike typer hendelser med negativ innvirkning som kan oppstå underveis i prosjektet. Tabell 3 viser hvordan en hendelse blir vektlagt, med hensyn på dens sannsynlighet, og dens konsekvens. Både sannsynlighet og konsekvens har en skala som går fra Svært Lav(1) til Svært Høy(5). Deretter multipliserer man de to tallene man får, og ender med en totalsum fra 1-25, sammen med en fargekode. I Tabell 4 finner man ulike hendelser med årsaker og tiltak. Hendelsene og risikoene som er vist i tabellen er kartlagt ved prosjektstart, og tar for seg det gruppen så på som reelle risikoer for prosjektet.

Fargen grønn indikerer at hendelsen vil ha liten grad av sannsynlighet og innvirkning på prosjektet. Her trengs ikke store tiltak for å rette opp i, og/eller forebygge en slik hendelse.

Fargen gul indikerer mindre problemer som kan oppstå, uten at det går så mye utover prosjektets gang. Her er det viktig å ligge litt i forkant med tanke på tid, slik at man har tid til å rette opp i ved for eksempel sykdom fra Hendelse/Risiko nummer 5 i Tabell 4.

Fargen oransje er av større alvorlighetsgrad. Slike hendelser vil mest sannsynlig oppstå på et eller annet tidspunkt i prosjektets gang. Ser vi i Tabell 4, handler disse hendelsene om ting som kan forebygges i forkant. Også her gjelder det å være føre var, og se kontinuerlig på tiltakene selv om hendelsen ikke har forekommet enda. Dette vil være med på å minimere innvirkningen en slik hendelse vil ha på prosjektet om den inntreffer.

Fargen rød er ting som vil ha svært høy konsekvens for prosjektet. Dette er kritiske ting som må unngås for å kunne levere et ferdigresultat ved slutten av prosjektets tidsramme.

Sannsynlighet	**Svært Høy (5)**	**5**	**10**	**15**	**20**	**25**
	Høy (4)	**4**	**8**	**12**	**16**	**20**
	Middels (3)	**5**	**6**	**9**	**12**	**15**
	Lav (2)	**6**	**4**	**6**	**8**	**10**
	Svært Lav (1)	**1**	**2**	**3**	**4**	**5**
		Svært Lav (1)	**Lav (2)**	**Middels (3)**	**Høy (4)**	**Svært Høy (5)**
	Konsekvens					

Tabell 3 – Risikomatrise

	Hendelse /Risiko	Årsak	Sannsynlighet	Konsekvens	Risiko-produkt	Tiltak
1	Misforståelse av funksjonelle egenskaper	Dårlig dokumentering av møte med oppdragsgiver	Lav (1)	Svært høy (5)	10	Kalle inn til ekstra møte med oppdragsgiver
2	Urealistiske mål	Overvurdering av egne evner	Middels (3)	Høy (4)	12	Være mer kritisk til planlegging. Dele målene opp i mindre delmål.
3	Ikke tilstrekkelig kunnskap	Nytt verktøy for å utvikle produktet	Høy (4)	Middels (3)	12	Sette seg inn i nye verktøy og bruke tilgjengelig ressurser
4	Ikke-kompatibel med iPhone	Gruppen har bare PC med Windows	Middels (3)	Lav (2)	6	Oppdragsgiver har mulighet til å kompilere applikasjonen for iPhone
5	Sykdom	Eksempelvis korona	Høy (5)	Lav (2)	8	Overholde nasjonale retningslinjer, og bruke sunn fornuft

Tabell 4 - Risikoanalyse

3.5 Evalueringsplan

Evalueringsplanen for prosjektet er ikke en fastsatt plan mellom gruppen og oppdragsgiver, men heller en kontinuerlig evaluering av resultatene gruppen har kommet fram til i utviklingen. Som nevnt tidligere i kapittel 2.2.2, så vil det ikke bli gjort noen testing på reelle brukere med det første. Det vil være en kontinuerlig dialog mellom gruppen og oppdragsgiver for å evaluere resultatene, og etter hvert også vise fram applikasjonen og dens funksjoner til andre for å få tilbakemeldinger på resultatene.

Resultatene vil bli evaluert forløpende med oppdragsgiver, som kan gi tilbakemeldinger om misnøye og tilfredshet med dem.

Gruppen vil også bruke et SUS-skjema, som vil bli beskrevet mer detaljert i kapittel 5. Dette skjemaet vil gi gruppen en mer generell oversikt for brukervennligheten til applikasjonen, og om løsningen oppfyller målene som omhandler dette.

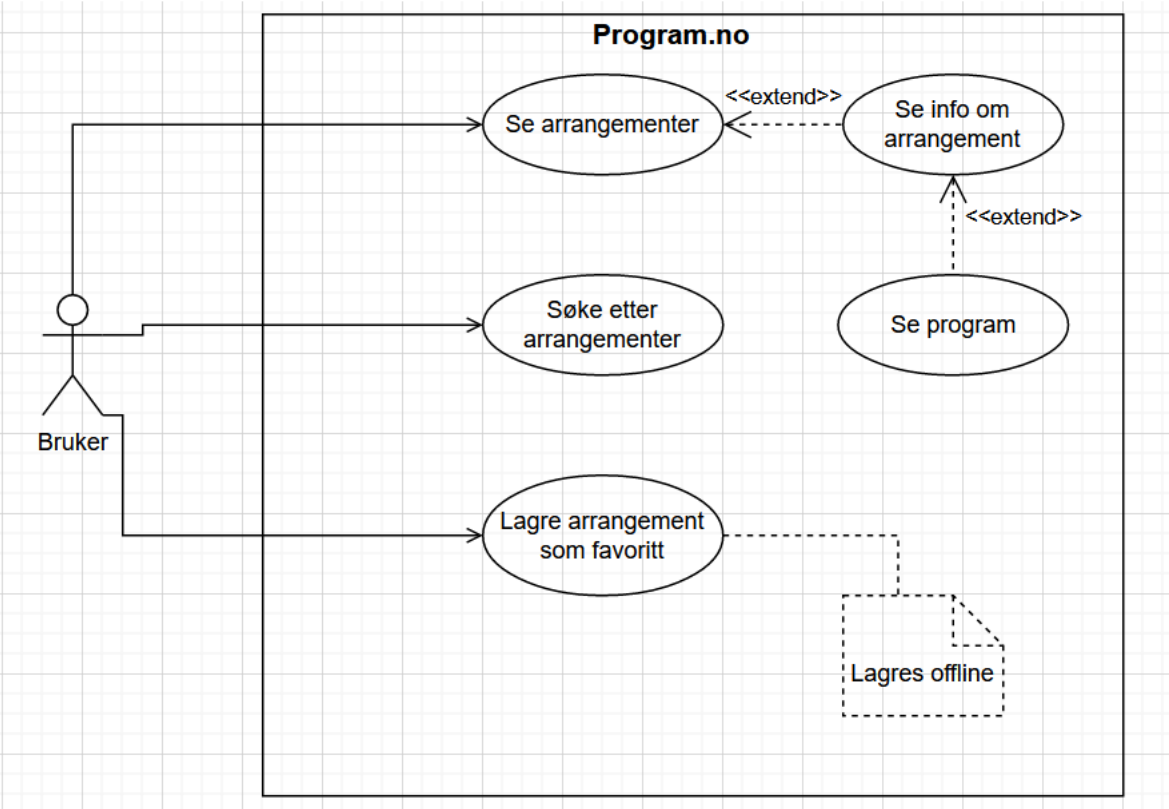
4 DETALJERT LØSNING

4.1 Innledning

I dette kapittelet beskrives løsningen og designet i detalj, og hvordan resultatene for prosjektet ble oppnådd. Den detaljerte beskrivelsen blir gitt hovedsakelig gjennom tekst, men også ved bruk av modeller og illustrasjoner for å gi bedre inntrykk i hvilken løsning som er tatt i bruk.

4.2 Brukstilfelle

Løsningen og designet til applikasjonen er basert på at en bruker kan bruke den fritt, uten at en «admin» eller lignende må gripe inn for at brukeren skal kunne ta i bruk de forskjellige funksjonene. Som tidligere nevnt er det ikke for mange funksjoner for brukeren, slik at det er mindre forvirring, men alle funksjonene er viktig for helheten til det endelige produktet. Dette kan man tydelig se på brukstilfellediagrammet i figur 5. Den gir en enkel oversikt på hva en bruker kan gjøre i applikasjonen.



Figur 5 – brukstilfellediagram

I figur 5 kan en se at applikasjonen ikke har for mange funksjoner, slik at en bruker ikke må sette av mye tid til opplæring i bruk av applikasjonen. Dette er ideelt med tanke på at brukergruppen til applikasjonen er av alle aldere med forskjellig teknisk kompetanse.

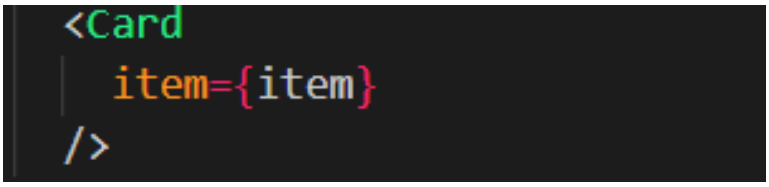
I figur 5 er det 3 hovedfunksjoner en bruker kan ta i bruk. De er, å se en liste over arrangement, søke på et arrangement og lagre arrangement som favoritt. I listen over arrangement kan en videre klikke inn på arrangementet for å få mer informasjon om det, for eksempel mer detaljert beskrivelse om arrangementet, og videre se programmet for et arrangement. Brukeren kan også lagre et arrangement som favoritt, som gjør det mulig å se det offline, altså uten internett.

4.3 Rammeverk og moduler

4.3.1 React Native

Som diskutert i kapittel 3 er det et ønske og fordelaktig å ha en native applikasjon for å oppfylle kravene for applikasjonen Program.no. React Native gir muligheten for å skrive kildekoden kun en gang, og rammeverket vil ta hånd om kompilering av koden som skal kjøre på de ulike plattformene uten at utviklerne må gjøre store endringer. Underveis i utviklingen har man også muligheten for å utføre en «hot reload». Det betyr at man ikke trenger å kompilere hele koden for hver gang man gjør endringer, og applikasjonen trenger kun en kjapp omstart for at endringer skal tre i kraft.

Oppbyggingen av applikasjonen ved bruk av React Native skjer gjennom å sette sammen komponenter. En React-komponent er en gjenbrukbar og uavhengig kodebit, som kan sees på som en Javascript-funksjon som returnerer noe visuelt. I figur 6 kan vi se hvordan gruppen har definert en komponent *Card* som tar inn en parameter *item* som representerer et arrangement. I *Card* er det en del tekstfelt og bildehåndtering som blir returnert tilbake. Deretter brukes komponenten i en liste for å returnere hvert arrangement som et *Card*. Siden komponenten kun har et arrangement som parameter, kan denne gjenbrukes andre steder i koden hvor man ønsker å vise fram et arrangement på samme måte. I tillegg til dette har React Native også en god del komponenter innebygd, som for eksempel bildekomponent og listekomponenter.



Figur 6 – Egendefinert React-komponent

4.3.2 Moduler

Med modul menes det et sett med funksjoner som blir implementert i applikasjonen. Slike funksjoner kommer i en form for bibliotek, og er ofte lagd av brukersamfunnet. Gruppen har anvendt noen slike moduler for å oppnå ønsket design.

- **SVG –** SVG står for skalerbar vektorgrafikk. Dette har gruppen tatt i bruk for å kunne skalere ulike ikon uten at de skal bli kornete.
- **Vector-Icons –** Et bibliotek av ulike ikoner i SVG-format, med ulike type fonter.
- **React navigation –** Dette er modulen som tillater at man kan trykke seg fra en skjerm til en annen på en smidig måte. Modulen gir også muligheten for å kunne ha en navigeringsbar på skjermen, for å kunne veksle mellom ulike skjermer.
- **React-native-elements –** Denne modulen har blitt brukt for å kunne lage en søkefunksjon som kan oppdatere liste-visningen på en smidig måte.
- **Safe-area-context –** Et av kravene fra oppdragsgiver var at applikasjonen skulle skalere seg riktig i forhold til ulike modeller og plattformer. Safe-area-context brukes for å få tilgang til enhetens visualiseringsområde, slik at tekst og bilder ikke havner utenfor dette området.

Fordelen med slike moduler er at de kan spare utviklere for en del tid, de er ofte lett å tilpasse til ønsket bruk, og er ofte godt dokumenterte. Disse fordelene gjør at moduler er lett og effektivt å jobbe med.

4.4 Skjermer

4.4.1 Innlastingsside

Når en bruker åpner applikasjonen vil den bli møtt med en innlastingsskjerm, som vist i figur 7. Denne skjermen vises i et kort øyeblikk, og går automatisk over til hjemskjermen når all dataen er klar til å vises fram. Dette er gjort med tanke på at det kan ta noe tid å hente ut informasjon fra databasen om man for eksempel har dårlig internettforbindelse.

Å ha en slik innlastingsside gir brukeren et tegn på at applikasjonen fungerer, og ikke har krasjet. Jakob Nielsen nevner 3 hovedfordeler ved å ha en progresjonsindikator som brukeren kan se på mens systemet jobber, hvor 2 av fordelene også er relevant for prosesser hvor tiden er under 10 sekunder (Nielsen, 1993). Slike prosesser vil for eksempel være i en applikasjon som Program.no.

Første fordel er at indikatoren forteller brukeren at applikasjonen og/eller systemet ikke har krasjet, men at den holder på å jobbe med problemet til brukeren. Dette er en del av brukeropplevelsen, da man slipper å få opp en blank side før det plutselig popper inn masse informasjon.

Den andre fordelen er at det gir brukeren noe å se på, som igjen vil holde på oppmerksomheten til brukeren når neste skjerm er klar til å vises.

Figur 7 – innlastingsside

4.4.2 Startside

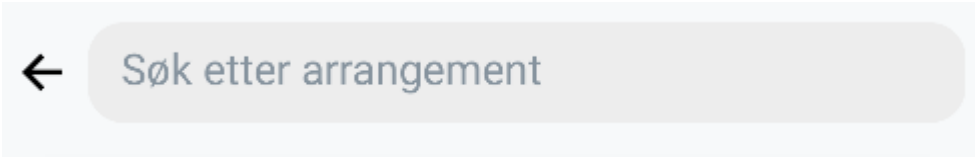
Når all relevant informasjon er hentet fra databasen og er klar til å bli vist fram, blir brukeren navigert til startsiden automatisk. Her vises alle tilgjengelige arrangement fram med relevant informasjon, slik som startdato og tidspunkt for arrangementet, lokasjon og tittel.

Startsiden er bygget opp av 3 ulike deler; header, selve listen og en navigeringsbar.

Header

Headeren finner vi øverst på startsiden. Som vist i figur 8, inneholder denne logoen til Progam.no, samt en søkeknapp. Når man trykker på søkeknappen vil figur 9 vises, hvor brukeren nå kan skrive inn hele eller deler av tittelen til et arrangement. Listen vil oppdatere seg fortløpende mens brukeren skriver. Dette vil si at om brukeren skriver inn bokstaven «i» vil både Innlandsmesterskapet og Fanafestivalen vises, da begge disse inneholder bokstaven «i». Om brukeren for eksempel skriver «inn», vil kun Innlandsmesterskapet vises.

Figur 8 – Header



Figur 9 – Søkefelt

Listen

Videre har man selve listen. Når gruppen skulle lage listekomponenten, ble det oppdaget at React Native hadde 3 ulike valg for dette.

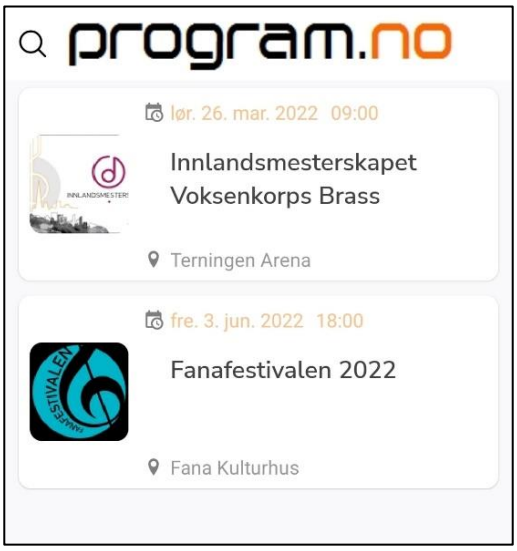
Den ene er Section List, som er en listekomponent som har egenskapen at ting kan deles inn i seksjoner. Dette så vi fort at ikke var bruk for i en slik applikasjon som gruppen

skulle utvikle. Denne er mer hendig for en applikasjon som for eksempel skal vise fram en meny, hvor man vil dele matretter inn i seksjoner som hovedretter, sideretter og forretter.

Den neste er ScrollView. ScrollView er en generisk komponent. Ulempen med denne komponenten er at man må hente ut all informasjonen på en gang, og alt må lastes inn og være klar til å vises (Meta Platforms, u.å). Dette vil si at om Program.no består av 1000 arrangementer, så må alle lastes inn, selv om brukeren ikke kan se alle samtidig på skjermen sin. Dette kan kreve mye plass lokalt på telefonen, og redusere responstiden på applikasjonen.

Det siste alternativet er FlatList. FlatList-komponenten er mer dynamisk, og laster kun inn de arrangementene brukeren faktisk ser på skjermen (Meta Platforms, u.å). De andre arrangementene ligger klar, og blir kun vist fram når brukeren blar nedover skjermen. Med denne komponenten kan man altså ha utallige arrangementer, uten at applikasjonen spiser opp for mye plass og ytelse. På bakgrunn av dette, valgte gruppen å gå for FlatList-komponenten for å laste inn arrangementene.

Som nevnt tidligere i kapitlet består listen av egendefinerte komponenter kalt *Cards*, som representerer arrangementene. Disse komponentene er laget dynamisk, og vil oppdatere seg i sanntid om det skjer endring for et arrangement i databasen. Med dette sikrer man at brukeren alltid får den siste informasjonen om et arrangement. Når en bruker trykker på et av arrangementene vil de tas med videre til neste skjerm som gir beskrivelse om det valgte arrangementet.



Figur 10 - Startside med visning av liste og cards

Navigeringsbar

Applikasjonen vil til slutt bestå av mange ulike skjermer. Noen av disse vil navigeres ut ifra hverandre, og noen er uavhengig av de andre skjermene. For å oppnå en god navigasjonsflyt har gruppen implementert en navigeringsbar nederst på skjermen. En slik tilnærming vil være godt kjent for de fleste brukere, da dette også finnes hos mange av de mest populære applikasjonene slik som Spotify, Facebook og Instagram. Denne navigeringsbaren vil alltid være synlig og tilgjengelig for brukeren.

I figur 1 i kapittel 2.1.1 kan man se dagens løsning hvor en såkalt hamburgermeny tas i bruk. Denne gjenkjennes ved bruk av tre streker, som ofte er plassert i et av de øverste hjørnene på skjermen. Denne menyen inneholder de samme mulighetene som er tiltenkt navigeringsbaren, men har noen begrensninger. Om man for eksempel går inn på et arrangement, vil denne menyen forsvinne. Desto lenger inn i applikasjonen man kommer, desto lenger vekk vil man komme fra menyen. En navigeringsbar er derfor implementert for å gi brukeren hurtig tilgang til de ulike skjermene. I figur 11 kan man se denne løsningen implementert i applikasjonen.

Figur 11 – Navigeringsbar

4.4.3 Arrangementsbeskrivelse

Fra startsiden sendes det med en «dokumentnøkkel», som angir ID-en til et dokument i databasen, til siden for arrangementsbeskrivelse. Denne dokumentnøkkelen gjør det mulig for brukeren å trykke seg inn på et spesifikt arrangement på startsiden, og tar brukeren videre til siden «Arrangementsbeskrivelse», som viser en mer detaljert side med informasjon om arrangementet. Figur 12 viser en teknisk løsning på dette.

```
useEffect(() => {
  const sub = firestore()
    .collection('events')
    .doc(key)
    .onSnapshot(doc => {
      setEvent({
        picture_normal: doc.data().picture_normal,
        title: doc.data().title,
        location: doc.data().location,
        description: doc.data().description,
```

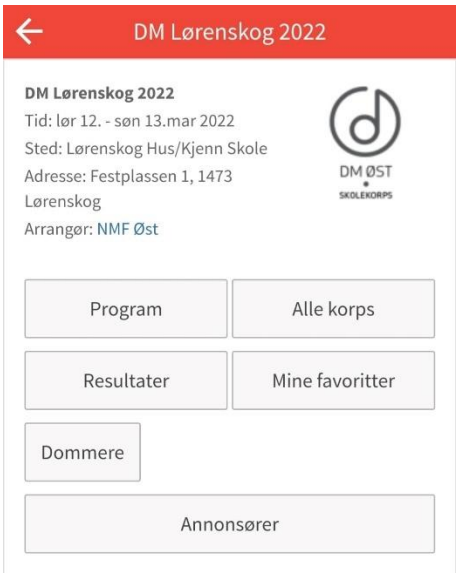
Figur 12 – Kode som viser bruk av en dokumentnøkkel

Her vil en bruker bli møtt med bilde/logo til arrangør, tittel for arrangementet, dato og tidspunkt for arrangement slutt og start, sted og adresse og til slutt en beskrivelse av arrangementet. Fra kapittel *2.4 – Litteratur om problemstilling*, ble det forklart at en bruker kan bli frustrert og i verstefall slutte å bruke applikasjonen dersom designet er uoversiktlig og rotete. Derfor viser denne siden bare informasjon som er nyttig for bruker, og dersom en ønsker kan man gå videre til en annen side som viser oversikt over programmet for et arrangement ved å trykke på «Se Program» knappen i midten, som vist på figur 13.

Et av problemene oppdragsgiver ville finne en løsning på var at skaleringen til applikasjonen for brukere med en «busslomme» øverst på telefonen ikke var helt gunstig. Løsningen på dette ble derfor å ta i bruk dimensjonene til både telefonskjermen og «vinduet» til applikasjonen, og videre sette høyde og bredde til de variablene. Figur 13 viser løsningen gruppen tok i bruk, hvor bildet til et arrangement tar opp 25% av skjermen og 100% av bredden til applikasjonen.

Figur 13 – Arrangementsbeskrivelsessiden

I planleggingen av hvordan designet skal se ut ble det klart at løsningen dagens applikasjon tok i bruk, med tanke på navigering tilbake fra en side til startsiden, var ganske utdatert og tok mer plass enn nødvendig. Derfor ble en gjennomsiktig «tilbake header» en klar løsning på dette. Figur 14 viser den tidligere løsningen på navigeringen, med «DM Lørenskog 2022» som topptekst og en pil som navigering.

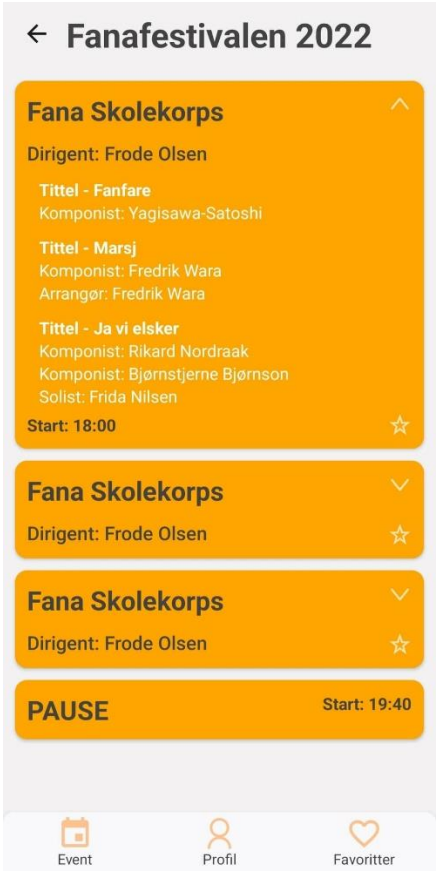


Figur 14 – Tidligere løsning

4.4.4 Programbeskrivelse

Etter en bruker har trykket på knappen «Se Program» i siden for beskrivelse av et arrangement, som vist på figur 13, vil man bli møtt med en side som har tittelen for det valgte arrangementet på toppen og flere bokser med info om programmet til det arrangementet.

I hver boks kan en bruker utvide og kollapse de, som vist på figur 15. Når en boks er utvidet vil en bruker kunne se informasjon for en del av et program. Dette er informasjon som for eksempel når et korps skal opptre og tiden de starter. Når en boks er utvidet eller kollapset er det en pil som indikerer det neste som skjer når en trykker på den, pil ned indikerer at en kan utvide boksen og pil opp indikerer at en kan kollapse boksen.



Figur 15 – Programbeskrivelse

I hver boks er det også lagt til rette for videre arbeid. I nedre høyre del av en utvidet boks er det en stjerne som indikerer at en bruker kan favorittmerke en del av et program. Dette blir indikert ved en stjerne som er fylt, som vil si at bruker har favorittmerket delen av programmet, og en stjerne som ikke er fylt.

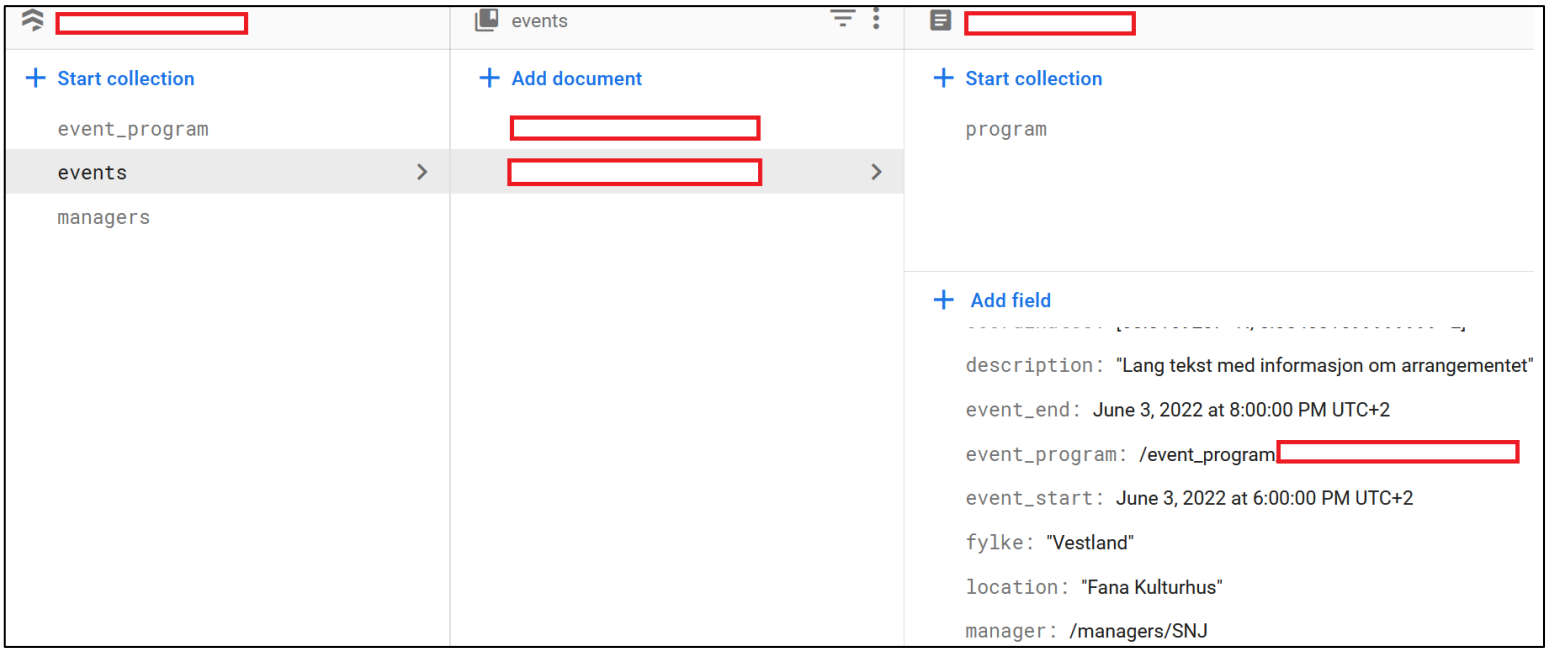
4.5 Database

For utviklingen er det tatt i bruk Firestore, som er en tjeneste levert av Firebase. Firestore er et NoSQL databasesystem som er fleksibelt og skalerbart for å lagre dokumentfiler. I underkapitlene skal vi se på hvorfor denne type database er godt egnet til bruk i native applikasjoner, slik som Program.no.

4.5.1 Arkitekturen

Firestore er en NoSQL-database, som vil si at databasen lagrer og henter data ved bruk av nøkkelverdier. Tradisjonelt er data lagret i rekker og kolonner, hvor en NoSQL-database lager unike nøklene for hvert objekt (Christensson, 2013). At data ikke lagres i en struktur med rekker og kolonner gjør at man kan håndtere større mengder data med stor fleksibilitet (SYSCO, u.å). Uten å måtte tenke på relasjonen mellom ulike tabeller, kan utviklingen utføres smidigere, og data kan hentes ut direkte ved bruk av de unike nøklene.

I figur 16 kan man se at det er opprettet to objekter, events og manager, som applikasjonen benytter for å hente ut informasjon. Disse objektene blir lagret i en collection, videre kalt grupper. Under hver gruppe kan man enten ha dokumenter, eller undergrupper. I databasen til Program.no inneholder events et dokument for hvert arrangement. For hvert arrangement skal det også være en undergruppe som igjen inneholder informasjon om programmet for arrangementet, i figur 16 kalt program. I tillegg til undergruppen er det lagt til en del felt som gir en mer generell beskrivelse av arrangementet. I figur 16 ser man at informasjon slik som start og slutt for arrangementet, lokasjonen, beskrivelse om arrangementet, med mer, lagres her.



Figur 16 – Oppsett i Firestore

4.5.2 Funksjonalitet

Oppdragsgiver hadde et ønske om at gruppen skulle ta i bruk Firestore med tanke på funksjonene som kan brukes mellom databasen og applikasjonen. Funksjonene som Firestore tilbyr gir en mer responsiv applikasjon, og kan samtidig redusere kostnadene for bedriften.

onSnapshot

En av nøkkelfunksjonene til Firestore er oppdatering av data i sanntid. Ved bruk av funksjonen onSnapshot vil man kunne hente ut data, for og så sette en «lytter» på dataen som er hentet ut. Dette vil si at applikasjonen vil lytte etter endringer. Om det skjer en endring av dataen i databasen, vil det sendes en beskjed til funksjonen onSnapshot, og endringen trer automatisk i kraft så lenge enheten er tilkoblet internett. Om man skulle være uten internett, vil funksjonen ligge klar til å sjekke etter oppdatering automatisk når tilkoblingen er tilbake. Applikasjonen trenger kun å hente ut dataen som er endret, og ikke hele gruppen, som fører til en smidig og responsiv applikasjon.

I figur 17 kan man se hvordan gruppen har anvendt funksjonen for å hente ut de ulike arrangementene. Applikasjonen går direkte inn i events, og henter ut informasjonen lagret i denne gruppen ved bruk av onSnapshot. Videre hentes dataen for hvert dokument, som er arrangementene, og settes inn i en liste. Om for eksempel startdato for et av

arrangementene endres i databasen, vil kun det berørte dokumentet bli hentet på nytt, og endringen blir umiddelbart synlig for brukerne av applikasjonen.

```
//Henter ut alle arrangementene fra databasen og legger de til i listen events
const sub = firestore()
  .collection('events')
  .onSnapshot(querySnapsot => {
    const events = [];

    //Legger til alle arrangementene i events-listen
    querySnapsot.forEach(documentSnapshot => {
      events.push({
        ...documentSnapshot.data(),
        key: documentSnapshot.id,
      });
```

Figur 17 – onSnapshot i Program.no

Lokal lagring

Ved å ta i bruk Firestore vil applikasjonen som standard sette av 10MB til lokal lagring av data som brukeren henter fra databasen. Dette skjer automatisk som en del av Firestore sin utviklerpakke. Denne grensen kan økes ytterlig, eller settes til 0 om ønskelig. Gruppen og oppdragsgiver er enig i at 10MB er mer enn nok for den type applikasjon Program.no er. Om brukeren mister tilgang til internett vil den lokale lagringen sørge for at data som er lastet inn er tilgjengelig, selv om applikasjonen blir lukket. Om man skulle overstige 10MB lagring vil de dokumentene som ble brukt sist være tilgjengelig, slik at gammel unyttig data ikke tar opp plass. Denne funksjonen er også en av hovedfunksjonene som gjør Firestore gunstig for utvikling av native applikasjoner. Det gjør at applikasjonen er mer tilgjengelig, selv uten tilkobling til internett.

I tillegg til dette er det også med på å redusere kostnaden for oppdragsgiver. Slik som løsningen deres er i dag betales det for hver gang en bruker leser et dokument. Dette skjer for hver gang brukeren går inn på et arrangement i dagens applikasjon. Ved en konsert hvor det er flere ting på programmet, sjekker gjerne brukeren programmet flere ganger i løpet av arrangementet. Med Firestore henter brukeren et dokument for et arrangement kun en gang, og ved eventuelle endringer. Om brukeren så går inn på samme arrangementet vil

dette leses fra den lokale lagringen og ikke databasen. Dermed vil det være færre avlesninger i databasen per bruker enn det er i dagens løsning.

5 RESULTATER

5.1 Evalueringsmetode

I prosjektperioden har gruppen hatt jevnlig dialog med oppdragsgiver, som fra første møtet av har kommet med ønsker for design og funksjonalitet. Møtene har gjort at gruppen har kunnet vise fram løsninger på design og funksjonalitet kontinuerlig gjennom prosjektets gang og fått tilbakemeldinger på hva som fungerer, og hva som ikke fungerer.

Dialogen med oppdragsgiver har vært sentralt for å kunne kontinuerlig evaluere mobilapplikasjonen og sikre at sluttproduktet stemmer overens med oppdragsgiverens ønsker og krav.

Mot slutten av prosjektperioden har gruppen sendt en såkalt «APK-fil» eller Android Package File, og kildekoden som lar oppdragsgiver installere applikasjonen på egen telefon for testing (Google, 2021). Ut fra testingen av applikasjonen kan oppdragsgiver gi tilbakemeldinger for hva som fungerer, og eventuelt hva de ville gjort annerledes. Videre kan det vurderes om sluttproduktet samsvarer med de tilbakemeldingene som ble gitt underveis i prosjektet. Gruppen får da tilbakemeldinger om helheten av applikasjonen.

Etter oppdragsgiver har testet applikasjonen vil et «SUS-skjema», forkortelse for «System Usability Scale», bli brukt til evaluering. Dette skjemaet er brukt til å undersøke brukervennligheten til et system og gir gruppen en god måte evaluere resultatet og kvaliteten til applikasjonen (Usability.gov, u.å). Resultatet til SUS-skjemaet vil bli presentert i delkapittel 5.2. Det skal også være sagt at et slikt skjema egner seg best dersom det er flere testere av applikasjonen, men gruppen vurderte skjemaet som en god evalueringsmetode for å gi en helhetlig oversikt for hvordan applikasjonen fungerer.

5.2 Evalueringsresultat

Generelt sett fikk gruppen positive tilbakemeldinger og evalueringer på applikasjonen, men også noen forbedringsområder.

Den var grei å installere og oppdragsgiver kom seg lett inn på applikasjonen. Han syntes startsiden var litt rotete, men det gjaldt plassering av innholdene i hver boks for et

arrangement. Dette kunne enkelt fikses med noen små justeringer. Logoen til Program.no hadde også litt feil oppløsning, noe som enkelt kan fikses.

På siden for beskrivelse av et arrangement ville han at man skal kunne scrolle hele siden og ikke bare beskrivelsen om et arrangement. Han kunne også ha tenkt seg at dersom en bruker trykker på kart/plasserings-ikonet, så blir en tatt videre til et kart eller videre til Google Maps som viser hvor arrangementet skjer.

På siden som gir informasjon om et program syntes oppdragsgiver at siden så bra ut. Det kunne bli litt for mye oransje, men det gjald mer personlig preferanse. Han likte også godt at gruppen hadde lagt til stjernemerking slik at det kunne bli brukt til videreutvikling.

Gruppen brukte en del av de tilbakemeldingene til å raskt gjøre små justeringer som ikke tok for mye tid, slik at oppdragsgiver ble mer tilfreds med det endelige produktet. Alt i alt syntes oppdragsgiver at applikasjonen er et godt utgangspunkt for videreutvikling, noe som stemmer godt overens med hva som var ønsket resultat med oppgaven.

Som del av evalueringen ble det også sendt ved et SUS-skjema til oppdragsgiver, som nevnt i delkapittel 5.1. Dette skjemaet ga gruppen en god oversikt over hvordan applikasjonen fungerer generelt sett, og om det hadde vært nødvendig med store endringer dersom gruppen skulle ha utviklet applikasjonen videre. Spørsmål og svar vises i tabell 5.

På spørsmål 1 og 5 er det gitt en 4-er. Dette stemmer overens med gruppens tanker om at enkelte deler av applikasjonen kunne vært bedre integrert dersom noen kunnskaper hadde vært bedre. Selv om en 4-ere er gode poeng, så er det fortsatt alltid et mål om å nå 5-eren på de to spørsmålene.

Spørsmål 2 og 3 ble vurdert med beste poeng for hva de spurte etter. Systemet var ikke unødvendig vanskelig, og det var enkelt å bruke for oppdragsgiver. Dette er noe som gruppen har strebet etter å oppnå, og er et viktig krav for applikasjonen.

Spørsmål 4, 7, 8, 9 og 10 henger veldig mye sammen, fordi de vurderer om applikasjonen har en læringskurve å komme seg over dersom en vil ta den i bruk. Dette ble ikke tilfelle for oppdragsgiver og scoret høyest i henhold til ha de spurte om. Siden det er en bred aldersgruppe som bruker dagens løsning, så ble dette et viktig krav å oppfylle.

På spørsmål 6 ble det spurt om det var mange uregelmessigheter i applikasjonen. Dette gjelder da som nevnt tidligere at enkelte sider ikke hadde best mulig oppsett. For eksempel har siden som beskriver programmet veldig mye oransje i forhold til de andre sidene. Et annet eksempel var at siden for beskrivelse av et arrangement, så kunne en bare scrolle den nederste boksen, men andre sider kan man scrolle hele siden.

Alt i alt ga evalueringsresultatene gruppen en god oversikt på hva som var bra og ikke fullt så bra. Det gir også et inntrykk om sluttproduktet er et godt grunnlag for videreutvikling, noe som oppdragsgiver var positivt til.

1 = Strongly disagree

5 = Strongly agree

	1	2	3	4	5
1. I think that I would like to use this system frequently.				X	
2. I found the system unnecessarily complex.	X				
3. I thought the system was easy to use.					X
4. I think that I would need the support of a technical person to be able to use this system.	X				
5. I found the various functions in the system were well integrated.				X	
6. I thought there was too much inconsistency in the system.			X		
7. I would imagine that most people would learn to use this system very quickly.					X
8. I found the system very cumbersome to use.	X				
9. I felt very confident using the system.					X
10. I needed to learn a lot of things before I could get going with this system.	X				

Tabell 5 – SUS-skjema (Usability.gov, u.å)

5.3 Prosjektresultat

Gjennom prosjektet har gruppen fått utviklet en fungerende applikasjon som er i stand til å hente ned informasjon i sann tid fra en database. Denne informasjonen er blitt presentert på en oversiktlig måte og applikasjonen er lett å navigere. Det er likevel ting fra visjonsdokumentet og kravene som ikke er implementert, for eksempel det å kunne lagre favorittarrangement. Kravene som ikke er implementert har underveis gjennom prosjektperioden blitt nedprioritert til fordel for andre krav. På bakgrunn av tilbakemeldinger fra oppdragsgiver fra kapittel 5.2 har gruppen likevel inntrykk av at den nye løsningen har nytteverdi for bedriften.

5.4 Prosjektgjennomføring

I start av prosjektet ble det laget en overordnet plan for gjennomføring av prosjektet gjennom bruk av Gantt-diagram fra kapittel 3.4.2. Gruppen har fulgt denne planen nøye fra start til slutt, men det har også vært behov for noen endringer underveis som vist i vedleggene tabell 6 og 7. En av endringene i planen som påvirket prosjektet mest var at tidsrammen for utviklingen måtte justeres ned. I starten var det tenkt at gruppen skulle programmere helt fram til siste uken av prosjektperioden, men det ble klart at dette ikke lot seg gjøre da flere kapitler handler om tilbakemelding på produktet. Istedenfor ble det prioritert å sette av tid for sluttevaluering fra oppdragsgiver.

I kapittel 3.4.1 er det diskutert rundt om utviklingsmetodikken iterativ og inkrementell utvikling, og hvordan dette kan være en fordel for mindre grupper. Denne metodikken har fungert svært bra for gruppen, da metodikken har vært gitt fleksibilitet til prosjektgjennomføringen. Dette har vært positivt for gruppen, da de avvikene og endringene som har oppstått har blitt oppdaget tidlig og tatt tak i med en gang.

Planen for prosjektet har vært ganske stabil gjennom hele prosessen. Det har vært mindre endringer i krav til produkt, men jevn dialog med oppdragsgiver har vært med på å minimere konsekvensene av dette. Jevnlige møter med veileder har også vært med på å hjelpe gruppen til å finne de neste stegene i prosessen for prosjektet.

6 DISKUSJON

I starten av prosjektperioden ga oppdragsgiver uttrykk for at det ikke er forventet at alle de ønskede funksjonene skal være med i sluttproduktet. Det ble heller et fokus på at gruppen skal løse så mange oppgaver som tiden strekker til, og da gi Styreportalen et godt grunnlag for videre arbeid.

En viktig problemstilling var at den tidligere løsningen på designet var utdatert og lite moderne, i forhold til andre mobilapplikasjoner. Gruppen valgte derfor å se på flere populære applikasjoner, som for eksempel Spotify og Instagram. Deretter finne ut hva de gjør riktig når det kommer til design og samtidig bruke litteratur som nevnt i delkapittel 2.4, for å gi en bedre brukeropplevelse som viser de ulike delene i applikasjonen på en oversiktlig måte. Det ble brukt nettsider som material.io og annen litteratur, for å finne hvilke designfeil en burde unngå i utviklingen og hva som er gode løsninger. Mot slutten av prosjektperioden fikk gruppen tilbakemeldinger på applikasjonens helhet, som ble beskrevet i delkapittel 5.2.

For å gjøre applikasjonen mer oversiktlig valgte gruppen å ta i bruk flere ikon for å vise forskjellig informasjon. Som løsning ble det brukt ikon for å vise fram informasjon, for eksempel brukt et kalenderikon etterfulgt av dato og tid for et arrangement som vist på figur 18. Det ble også brukt skygger på enkelte områder, som fremhever at området er klikkbart. Dette er noe som er ikke er brukt i den tidligere løsningen. Som resultat ble det mer oversiktlig og enklere å finne fram informasjon i applikasjonen.

Figur 18 – Ikoner og informasjon

En svakhet i løsningen ble at gruppen har lite tidligere kunnskaper med design, og derfor ble det mye å sette seg inn i. Dette førte til at mye tid ble satt av til å få tilbakemeldinger fra oppdragsgiver på fargevalg og oppsett i applikasjonen, og avvik i forhold til fargekombinasjoner og tekststørrelser. En tilbakemelding gruppen fikk etter oppdragsgiver testet og evaluerte applikasjonen var at i programsiden, vist på figur 15 i delkapittel 4.4.4, kunne fargekombinasjonene vært bedre. Dette er også noe som gruppen kunne ha løst på en bedre måte dersom kunnskapene om applikasjonsdesign hadde vært bedre eller mer tilstrekkelig tid i utviklingsperioden. Her var det som nevnt i kapittel 5.2 noen elementer i applikasjonsdesign som ikke fungerte fullt så bra, for eksempel litt rotete startside. Det ble også gitt positive tilbakemeldinger som at programsiden generelt sett så bra ut og oppdragsgiver syntes informasjonen ble vist på en god måte.

Som nevnt i delkapittel 1.4, så skalerer applikasjonen seg ikke korrekt. Dette ble løst, som nevnt i 4.3.2, ved å bruke safe-area-context og som resultat av dette kan nyere smarttelefoner med forskjellig skjermløsninger bruke applikasjonen uten at det oppstår forskjellige resultater.

I starten av prosjektperioden ble det valgt å nedprioritere flere funksjoner, mye fordi oppdragsgiver ville at fokuset skulle ligge på å forbedre designet og legge til rette for videre arbeid. Gruppen baserte da noen av valgene rundt funksjonene, på at de skulle være tilrettelagt for videreutvikling. Derfor valgte gruppen å bruke siste perioden av prosjektet til å legge til det mest grunnleggende for noen av de funksjonene som ikke ble prioritert. Eksempel på dette er profil og favoritter i navigeringsbaren som vist på figur 11 i delkapittel 4.4.2. Favorittmerking på programsiden ble også lagt til rette for videre utvikling, som vist på figur 19.

Figur 19 – Favorittmerking i et program

Gruppen valgte også å bruke versjonskontroll verktøyet Git sammen med et oppbevaringssted på GitHub. Dette var et valg som gruppen tok da dette verktøyet er blitt brukt aktivt gjennom studiet, og slipper da å lære flere teknologier, som for eksempel BitBucket. Her kunne også gruppen bruke såkalte «branches», for å alltid ha en versjon av applikasjonen som fungerer. Dette valget fungerte veldig godt, siden kunnskaper om verktøyene var noe som gruppen allerede satt på.

Mot midten av prosjektperioden valgte oppdragsgiver, etter samtaler med en prosjektgruppe som utvikler webapplikasjonen av Program.no, å forandre på databasen. Dette førte til at vår prosjektgruppe måtte sette av en del mer tid på å finne en løsning på å bruke den nye delen av database, og da måtte avvike litt fra planen. Dersom prosjektet skulle gjøres på nytt, ville gruppen hatt samtaler tidlig med oppdragsgiver om å gi en mer fullstendig database.

Et av målene for prosjektet var at applikasjonen skulle bli et godt utgangspunkt for videreutvikling. Som nevnt tidligere ble det gjort en endelig evaluering av applikasjonens helhet mot slutten av prosjektperioden. Her fikk gruppen en tilbakemelding om at applikasjonen var et godt utgangspunkt for videreutvikling, og at det meste stemte overens med det som var forventet.

7 KONKLUSJON OG VIDERE ARBEID

7.1 Måloppnåelse

Ser man på resultatet fra evalueringen satt opp mot målene prosjektet hadde i begynnelsen, ser man at det mest sentrale for applikasjonen er implementert i løsningen. Gruppen anser prosjektet som tilfredsstillende, selv om løsningen er ikke fullstendig.

Fra kapittel 1.4, hvor målet for prosjektet ble presentert, kan man med stikkord oppsummerer hva målet for prosjektet var ved starten.

- Moderne inntrykk
- Rask respons
- God oversikt
- Brukerprofil
- Lagring av favorittarrangement

På bakgrunn av evaluering fra oppdragsgiver, beskrevet i kapittel 5, har gruppen oppnådd de viktigste målene som omhandler moderne inntrykk, rask respons og god oversikt. Som nevnt tidligere var det ikke et forventet av gruppen å kunne levere en fullverdig applikasjon, men heller fokusere på nytt design og de mest grunnleggende funksjonene. Prosjektet kan da sees på som en suksess, da oppdragsgiver har uttrykt at løsningen stemmer overens med kravene, og bygger et godt grunnlag for videre arbeid.

7.2 Videre arbeid

Tidsrammen for prosjektet, samt kunnskapen ved prosjektstart, gjorde at det ikke ble mulig for gruppen å innfri alle ønskene fra oppdragsgiver. Under vil det ble gjennomgått hvordan gruppen tenker de ville fortsatt arbeidet for de avvikene som ikke ble oppnådd for applikasjonen.

Lagring av program

For videreutvikling av denne funksjonen ville gruppen fokusert på å opprette et innloggingssystem, hvor brukerprofilen blir lagret i databasen. Deretter vil man kunne gjøre det mulig for en bruker å trykke på et stjernemerke på programmet den ønsker å gjøre til favoritt, og brukeren vil da kunne motta varsler om resultat og annen nyttig info.

Her ville gruppen også knyttet brukerprofil og favorittprogram til navigeringsbaren, slik at de er lett tilgjengelig.

Resultat for et arrangement

I Program.no finnes det en del arrangementer innenfor konsertkonkurranse. Etter et slikt arrangement er ferdig, ville det vært gunstig at en bruker kan se resultatene. For å implementere dette i applikasjonen må det først opprettes en undergruppe i databasen på samme måte som program er implementert. Deretter vil man kunne ta i bruk store deler av koden gruppen har lagd for å hente program, når man skal hente ut resultater. Her vil det for det meste være å endre variabelnavn og litt design for å kunne vise resultatene fram på en god måte.

Filtrering av arrangement

For applikasjonen var det også tenkt et filter hvor brukeren kan sortere listen basert på lokasjon, alfabetisk rekkefølge av tittel, eller annen relevant info. Denne funksjonaliteten eksisterer i dagens webløsning til Program.no. Dette ville vært nyttig å ha i applikasjonen også, da dette gir en likhetsfølelse mellom webløsning og mobilapplikasjonen. Filtreringen av lokasjon var også tenkt å bli knyttet sammen med et kart.

8 REFERANSER

Budiu, R. (2013). *Mobile: Native Apps, Web Apps, and Hybrid Apps.* Tilgjengelig fra: https://www.nngroup.com/articles/mobile-native-apps/ (Hentet: 25. februar 2022)

Chacon, S. og Straub, B. (2014) *Pro Git.* 2. utg. New York City: Apress.

Christensson, P. (2013). *NoSQL Definition*. Tilgjengelig fra: https://techterms.com/definition/nosql (Hentet: 23.april 2022)

Drumond, C. (u.å.) *What is Scrum?* Tilgjengelig fra: https://www.atlassian.com/agile/scrum (Hentet: 25. februar 2022)

Firebase (u.å) *Firebase Documentation* Tilgjengelig fra: https://firebase.google.com/docs (Hentet: 10. mars 2022)

GeeksforGeeks (2021). *Difference between Native Apps and Web Apps.* Tilgjengelig fra: https://www.geeksforgeeks.org/difference-between-native-apps-and-web-apps/. (Hentet: 25. februar 2022)

Google (2021) *Application Fundamentals*. Tilgjengelig fra: https://developer.android.com/guide/components/fundamentals (Hentet: 04.05.2022)

Hoekman Jr, R. (2011). *Designing the Obvious: A Common Sense Approach to Web & Mobile Application Design.* 2nd Edition. Berkeley: Peachpit

Jordan, P., Thomas, B., McClelland, I. og Weerdmeester, B. (1996). *Usability Evaluation In Industry*. 1. utg. London: CRC Press.

Krupadeluxe. (2021) [Bilde]. Tilgjengelig fra: https://en.wikipedia.org/wiki/Iterative_and_incremental_development#/media/File:Iterative_Process_Diagram.svg (Hentet: 26.april 2022)

Martin, R.C. (1999) *Iterative and Incremental Development (IID),* april. Tilgjengelig fra: https://condor.depaul.edu/~dmumaugh/readings/handouts/SE477/IIDII.pdf (Hentet: 16. april 2022)

Meta Platforms (2022) *React Native. Learn once, write anywhere.* Tilgjengelig fra: https://reactnative.dev/ (Hentet: 10. mars 2022)

Meta Platforms (u.å) *Using a ScrollView.* Tilgjengelig fra: https://reactnative.dev/docs/using-a-scrollview (Hentet: 12.april 2022)

Meta Platforms (u.å) *Using List Views.* Tilgjengelig fra: https://reactnative.dev/docs/using-a-listview (Hentet: 12.april 2022)

Nielsen, J. (1993) *Response Times: The 3 Important Limits.* Nielsen Norman Group. Tilgjengelig fra: https://www.nngroup.com/articles/response-times-3-important-limits/ (Hentet: 12.april 2022)

Radigan, D. (u.å.) *What is Kanban?* Tilgjengelig fra: https://www.atlassian.com/agile/kanban (Hentet: 25. februar 2022)

SYSCO (u.å) SYSCO Ordbok: *Hva er No SQL.* Tilgjengelig fra: https://sysco.no/ordbok/no-sql/ (Hentet: 23.april 2022)

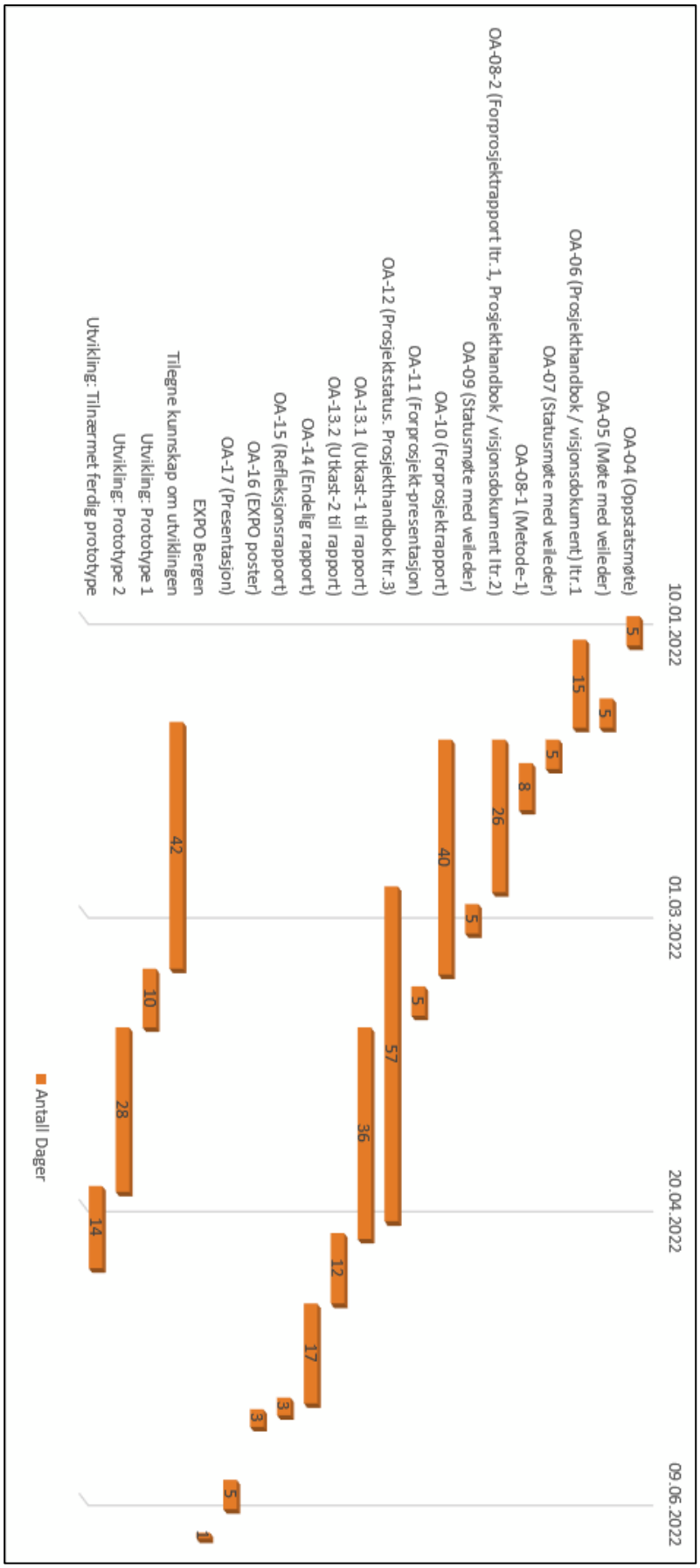
Usability.gov (u.å). *System Usability Scale (SUS).* Tilgjengelig fra: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html (Hentet: 04.05.2022)

9 VEDLEGG

9.1 Gantt

Aktivitet	Planlagt Start	Planlagt Slutt	Antall Dager	Ansvarlig Person
OA-04 (Oppst	10.01.2022	14.01.2022	5	D16
OA-05 (Møte	24.01.2022	28.01.2022	5	D16
OA-06 (Prosje	14.01.2022	28.01.2022	15	D16
OA-07 (Status	31.01.2022	04.02.2022	5	D16
OA-08-1 (Met	04.02.2022	11.02.2022	8	D16
OA-08-2 (Forp	31.01.2022	25.02.2022	26	D16
OA-09 (Status	28.02.2022	04.03.2022	5	D16
OA-10 (Forprc	31.01.2022	11.03.2022	40	D16
OA-11 (Forprc	14.03.2022	18.03.2022	5	D16
OA-12 (Prosje	25.02.2022	22.04.2022	57	D16
OA-13.1 (Utka	21.03.2022	25.04.2022	36	D16
OA-13.2 (Utka	25.04.2022	06.05.2022	12	D16
OA-14 (Endeli	07.05.2022	23.05.2022	17	D16
OA-15 (Reflek	23.05.2022	25.05.2022	3	D16
OA-16 (EXPO	25.05.2022	27.05.2022	3	D16
OA-17 (Preser	06.06.2022	10.06.2022	5	D16
EXPO Bergen	15.06.2022	15.06.2022	1	D16
Tilegne kunns	28.01.2022	10.03.2022	42	D16
Utvikling: Prot	11.03.2022	20.03.2022	10	D16
Utvikling: Prot	21.03.2022	17.04.2022	28	D16
Utvikling: Tiln	17.04.2022	30.04.2022	14	D16

Tabell 6 – Oppdaterte aktiviteter



Tabell 7 – Oppdatert prosjektplan