Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

Redesign / implementering av applikasjonen Program.no

Systemdokumentasjon

Versjon <1.2>

Dokumentet er basert på Systemdokumentasjon utarbeidet ved NTNU. Revisjon og tilpasninger til bruk ved IDER, DATA-INF utført av Carsten Gunnar Helgesen, Svein-Ivar Lillehaug og Per Christian Engdal. Dokumentet finnes også i engelsk utgave.

REVISJONSHISTORIE

Dato	Versjon	Beskrivelse	Forfatter
01/mai/2022	1.0	Innledning	Steffen Lampe
11/mai/2022	1.1	Fylt ut kapitler 2-8	Martin Aas Eliassen Steffen Lampe
19/mai/2022	1.2	Resterende kapitler og justeringer	Steffen Lampe

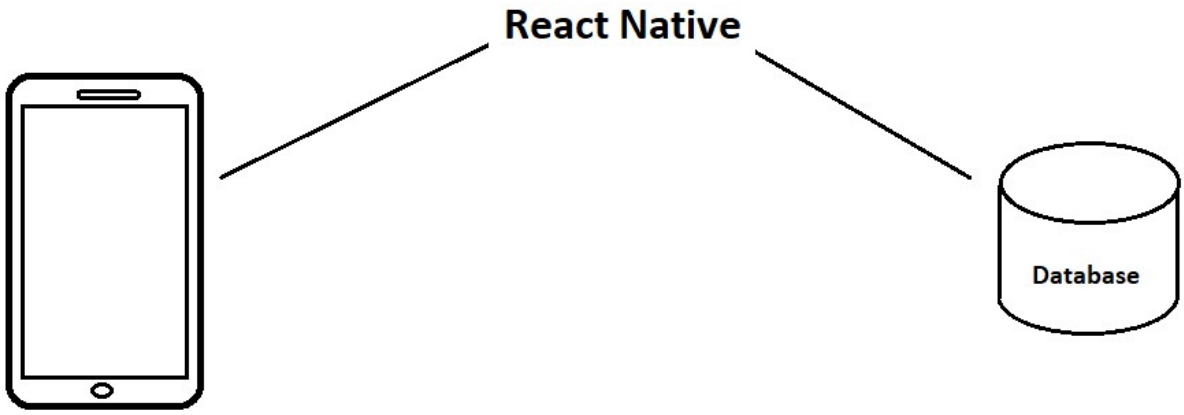
INNHOLDSFORTEGNELSE

1 INNLEDNING

Systemdokumentasjonen skal gi en oversikt over hvordan produktet er utviklet og de ulike delene den består av. Dokumentet er ment som en generell oversikt og for å gi en mer teknisk innsikt i produktet og arkitekturen. Systemdokumentasjonen vil bli ta for seg litt om databasen, arkitekturen og kodestruktur.

2 Arkitektur

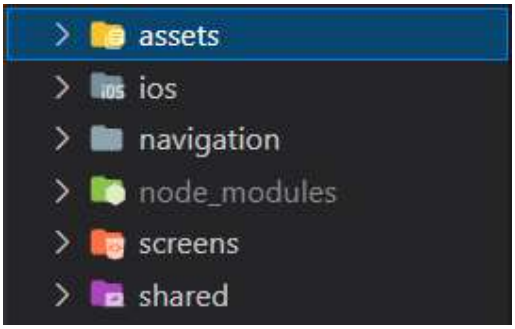
Mobilapplikasjonen Program.no er bygget på rammeverket React Native. Her blir det utviklet ulike komponenter, noen for det funksjonelle i applikasjonen og noen for det grafiske til applikasjonen, som settes sammen. Videre er det samspillet mellom React Native og databasen Firestore fra Firebase som henter ut nødvendig informasjon som komponentene kan ta i bruk.



Figur 1 – Arkitektur for mobilapplikasjon

3 PROSJEKTSTRUKTUR

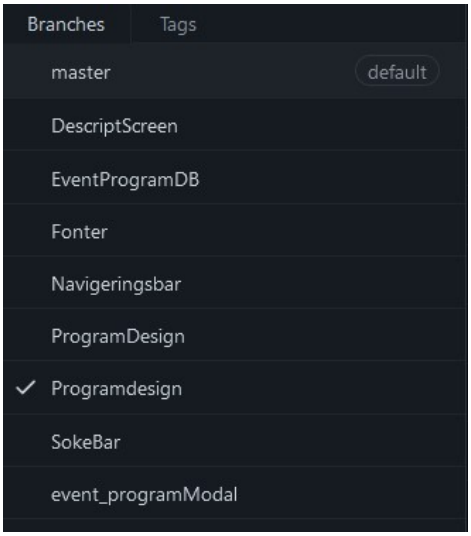
I applikasjonens kildekode er det opprettet ulike mapper til ulike formål. I figur 2 kan man se eksempel på hvordan dette er tatt i bruk for Program.no. I mappen assets finner man ulike fonter og statiske bilder applikasjonen har tatt i bruk. Ser man i screens-mappen er det opprettet en komponent for å vise fram de ulike skjermene i applikasjonen, hvor disse skjermene også kan hente innhold fra mappen shared.



Figur 2 – Struktur av kode

Versjonskontroll

Underveis i utviklingen har gruppen benyttet versjonskontrollsystemet Github. I Github har gruppen opprettet ulike grener for utvikling av de ulike komponentene av applikasjonen. Ved enden av prosjektet vil alt settes sammen i master-grenen. I figur 3 kan man se hvordan gruppen har anvendt dette systemet.



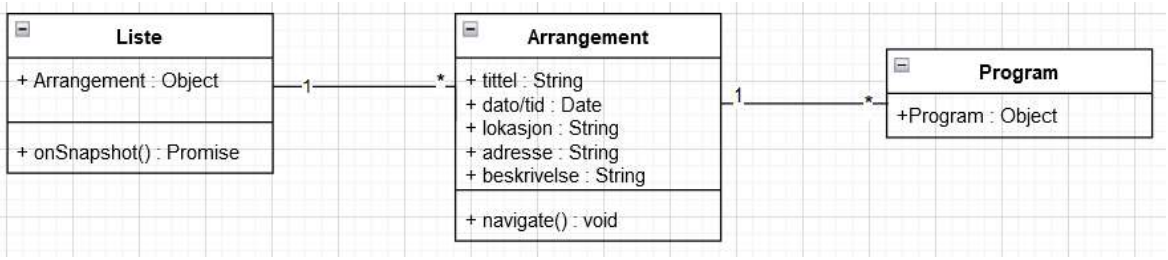
Figur 3 – Versjonskontroll i Github

4 KLASSEDIAGRAM

Ikke relevant for applikasjonen som er utviklet. Dette på bakgrunn av at React Native funksjoner som returnerer JSX (det grafiske), og ikke er klasser. For modellering av applikasjonen, se domenemodell.

5 Domenemodell

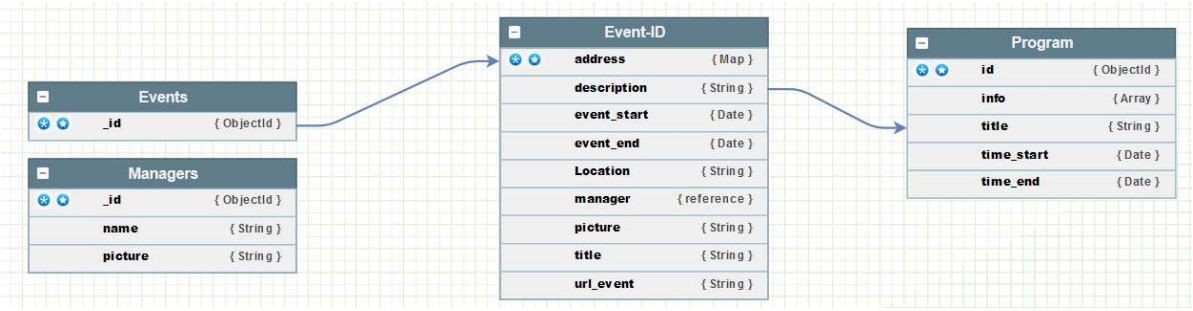
Domenemodellen for applikasjonen viser sammenhengen fra hva som vises i listen av arrangementer fram til programmet for et arrangement. Hvert arrangement med tilhørende program hentes fra databasen med relevante variabler, som så blir vist fram i en listevisning i applikasjonen. Forholdet mellom liste og arrangement, og mellom arrangement og program vises i figur 4.



Figur 4 – Domenemodell for Program.no

6 DATABASEMODELL

Databasemodellen i figur 5 gir et overblikk over hvordan de ulike delene er satt opp mot hverandre. Firestore er laget slik at man har collections, også kalt grupper, som kan inneholde dokumenter eller undergrupper. Ser man i Events, inneholder den flere dokumenter, som har flere attributter. Dokumentet er vist i Event-ID i figuren. Et dokument inneholder også en undergruppe *Program.* Her lagres informasjon om hvem som skal opptre på et arrangement, når de skal opptre, og annen informasjon som for eksempel dirigent for framføringen.



Figur 5 – Databasemodell for Firestore NoSql

7 Server-tjenester

Ikke relevant for dette prosjektet.

8 SIKKERHET

I løsningen til gruppen er det kun søkefeltet som kunne potensielt gitt et sikkerhetshull i applikasjonen. Dette er tatt hånd om ved at teksten som skrives inn kun blir brukt til å sammenligne med titler av arrangementer. Det er altså tekststrenger som blir sammenlignet med tekststrenger, og kan ikke gi tilgang til noe annet i koden eller databasen.

Om man ønsker å implementere løsninger som for eksempel brukerprofil vil det være nødvendig med flere type sjekker, og benytte seg av ulike autoriseringssystemer.

I databasen er det en API-nøkkel som også er med i kildekoden for prosjektet. Om man ønsker å lansere løsningen, og publisere den på for eksempel Google Play vil det være nødvendig å skjule denne for å forhindre uønsket tilgang i databasen.

9 INSTALLASJON OG KJØRING

Gjør telefonen klar for installering av APK-fil (Android)

1. Gå til innstillinger
2. Velg Biometri og Sikkerhet
3. Trykk på installer ukjente apper
4. Tillat fra «Mine filer» eller lignende

Deretter vil man ha bruk for APK-filen. Denne legges som egen fil i innleveringen. Denne laster man så ned på enheten. Når man åpner filen på enheten velger man «installer» og applikasjonen installeres.

10 DOKUMENTASJON AV KILDEKODE

Kildekode legges ved som Zip-fil i innleveringen.

11 KONTINUERLIG INTEGRASJON OG TESTING

Underveis i utviklingen har applikasjonen gått gjennom manuell testing og testing via Jest som er integrert i React Native.

Jest kjører tester samtidig som emulatoren kjører, og eventuelle feilmeldinger vil dukke opp om noe uventet skulle oppstå. Dette vil være for eksempel om en bildekomponent viser det den skal, og faktisk returnerer en gyldig verdi.

Applikasjonen som er utviklet leser kun verdier fra en database. Det har da vært nødvendig å teste at de verdiene som er lest stemmer overens med forventet resultat. Her har gruppen testet selv manuelt, og lagt inn håndteringer dersom verdiene ikke eksisterer.

12 REFERANSER

jestjs.io (n.å). *Testing React Native Apps · Jest*. Tilgjengelig fra: https://jestjs.io/docs/tutorial-react-native. (Hentet: 19.mai 2022)