Rebooked

Introduksjon

Denne løsningen er utviklet på macOS og har kun hatt mulighet til å teste på en iPhone. Oppgaven vi fikk utdelt gikk ut på å utvikle en teknisk løsning som skulle kunne brukes til å selge brukte fagbøker innenfor fagene vi var ferdige med på skolen. Applikasjonen skulle inkludere noen minstekrav når det kom til funksjoner i applikasjonen. Dette var funksjoner som bl.a. bruk av Firebase for databaseløsning, brukerhåndtering og lagringsplass for bilder.

Fokuset med oppgaven var å vise til kunnskap, som er tilegnet i løpet av semesteret med forelesninger, innenfor utvikling av en hybrid kryssplattform applikasjon i Ionic 3 ved å ta i bruk TypeScript, Sass, og Firebase som databaseløsning.

Fremgangsmåte

Da jeg skulle starte med utviklingen av løsningen var det først jeg gjorde å sette opp et skall for basiskravene som oppgaven spesifiserte, ved å generere tomme sider med knapper for å navigere seg på tvers av applikasjonen.
Jeg brukte litt tid på å sette meg inn i hvordan en tab-layout fungerer og hvordan man skal implementere denne løsningen på en god måte. I tab-baren på bunn av skjermen har jeg 4 ikoner, disse representerer sidene: Hjem, legg til bok, meldinger og profil.

Brukersystem

Deretter begynte jeg på brukersystemet til applikasjonen, til dette brukte jeg metodene som vi hadde blitt vist i løpet av forelesningene ved å registrere en bruker ved å bruke epost/passord. I min løsning så ønsket jeg at brukeren også skulle kunne registrere seg med ett brukernavn(displayname) og ett profilbilde. Til dette vurdere jeg i starten å bruke en user-collection i databasen til å holde på denne informasjonen, men etter en del research fant jeg ut at Firebase's autentiseringstjeneste tilbyr denne funksjonaliteten ved å oppdatere informasjonen til en registrert bruker. Så løsningen min til dette ble at i funksjonen som håndterer registrering henter jeg ut den registrerte brukeren rett etter at den har blitt laget og legge inn brukernavnet som er en konkatinert string bestående av fornavn og etternavn samt profilbildet som brukeren har tatt. Når brukeren skal logge inn på en konto så brukes det den registrerte eposten og passordet.

Den innloggede brukeren skulle ha mulighet til å se brukeren sin på en profilside, derfor lagde jeg ProfilePage til å vise frem all brukerdataen som var lagret på Firebase, samt en liste over bøker som brukeren har publisert. Det er også mulighet for brukeren å logge ut av kontoen ved å trykke på en knapp øverst i høyre hjørne på denne siden.

Publisering av bøker

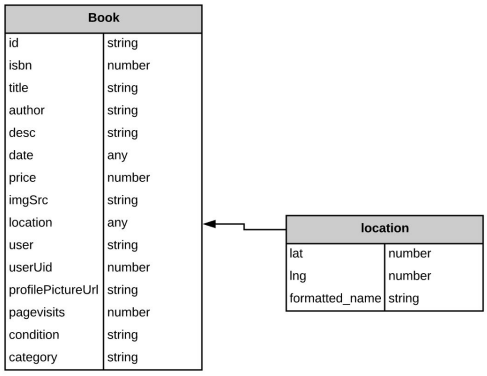
Etter at brukersystemet var fullført begynte jeg på databaseløsningen med å legge til bøker og vise en liste av bøker på hovedskjermen før designendringen.

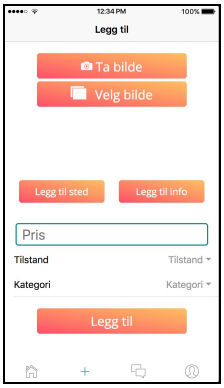
En bok har flere elementer (som vist i tabellen under) flere av disse blir satt til standardverdier ved opprettelsen av en bok, mens enkelte felter blir fylt ut enten ved at brukeren skanner bokens strekkode eller at hen manuelt fyller ut felter på siden AddBookPage. Brukeren kan også legge til ett bilde av boken de selger, eller velge ett bilde fra bildebiblioteket på deres telefon.

Som sagt har jeg implementert løsningen for at brukeren kan legge til informasjon om en bok ved å skanne bokens skrekkode. Her tok jeg i bruk ionic native's BarcodeScanner plugin til å lese strekkoden som returnerer ISBN-nummeret til boken. Ut ifra denne informasjonen bruker jeg en BookInfoProvider til å søke opp i Google's BookAPI for å hente ut all informasjonen om boken. Denne informasjonen (tittel, forfattere og beskrivelse) blir da satt inn i input-felt slik at brukeren kan dobbeltsjekke om den informasjonen stemmer eller om hen vi endre på den før publisering.

Jeg har også implementert muligheten for en bruker å registrere plasseringen på hvor boken har blitt lagt til. Dette var noe vi lærte i.l.a. forelesninger så dette var sett på som et krav, men jeg valgte å putte dataen i en array for at det ikke skulle bli for mange rader i bok-entiteten.

Når brukeren ønsker å legge til boken og trykker på "Legg til"-knappen på bunn vises det en toast på bunn av skjermen og brukeren blir sendt tilbake til skjermen med oversikt over alle bøkene i databasen.



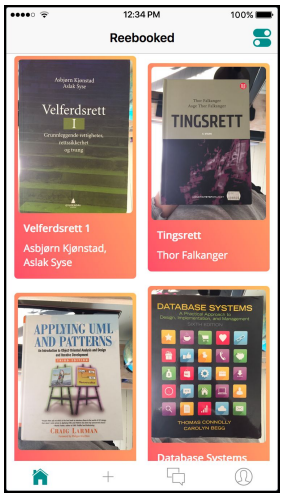


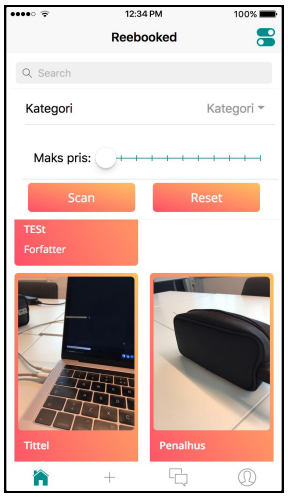
Hjem

Hjemskjermen består av en grid-visning av bøkene i databasen med informasjon om bokens tittel og hvilken/hvilke forfattere som har skrevet den respektive boken i listen. Denne er stylet ved å bruke flex-box.
Denne listen kan filtreres ved å trykke på ikonet øverst i høyre hjørne av skjermen. Da vil en liste over filtreringsalternativer dukke opp øverst på siden uansett hvor i listen man befinner seg. Her kan brukeren velge mellom filtreringsmåter som: Søk etter tittel, kategori på boken, minste pris, eller søk etter en bok men ISBN ved å bruke strekkodeskanneren. Listen vil da oppdateres automatisk med resultatet av filteringen.

Svakheter/restriksjoner

Pga mangel på kunnskap om hvordan firebase håndterer filtrering og at jeg ikke fikk til å bruke server-side filtering med variabler i where-spørringen når jeg hentet ut dataen fra databasen valgte jeg å bruke en clitent-side filtrering. Dette er en mindre skalerbar løsning som etterhvert vil knele når antall bøker i listen begynner å bli stort.
Det er også kun mulig å gjøre en filtrering av elementene av gangen, f.eks. Hvis en bruker har skrevet inn en tittel og prøver å filtrere resultatet etter pris, så vil tittelfilteret nulstilles. Dette skyldes min mangel på kunnskap innen en såpass avansert typescript funksjonalitet, i tillegg til at mangelen på tid så kunne jeg ikke bruke så alt for lang tid på å perfeksjonere alle funksjoner.



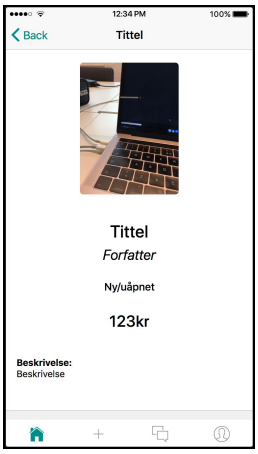


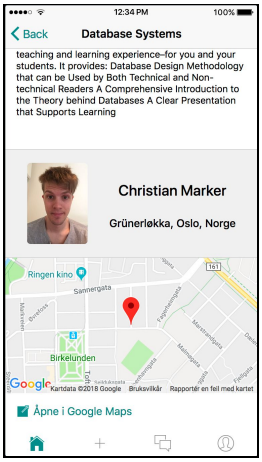
Detaljer

Når en bruker ønsker å se detaljene til en bok åpnes resultatet i en ny og separat side som heter BookDetailsPage. Denne siden inneholder all informasjon nødvendig om boken som er valgt, pluss to knapper: en for å starte en samtale med selgeren av boka, og en for å gå til en tom side som var ment til å bruke en betalingstjeneste eller liknende, men kjøp var en funksjon som jeg ikke rakk å implementere. Hvis den personen som er logget på appen er den samme som eier boken vil ikke disse to knappene dukke opp.

På bunn av detaljsiden ligger informasjon om hvilken bruker som har publisert annonsen samt ett kart over hvor boken ble publisert. Til dette brukte jeg Google MapsAPI som bruker dataen som ligger i lokasjons-arrayet til boken både til å sentere- og til å plassere en markør på kartet. Kartet blir rendret i en div som igjen blir stylet ved å bruke Scss.

Under kartet er det en knapp som åpner kartet i Google Maps lokalt på telefonen, dette er for at brukeren skal kunne navigere seg til plasseringen til boken ved å ta i bruk en ekstern applikasjon. Når kartet blir åpnet settes det automatisk en markør på kartet på samme lokasjon som boken.





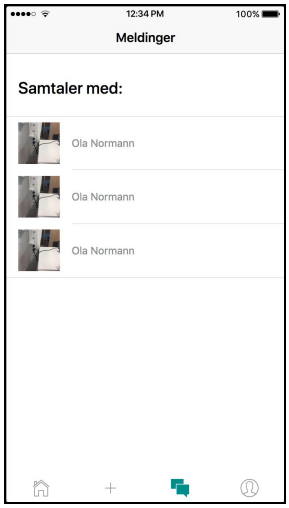
Meldinger

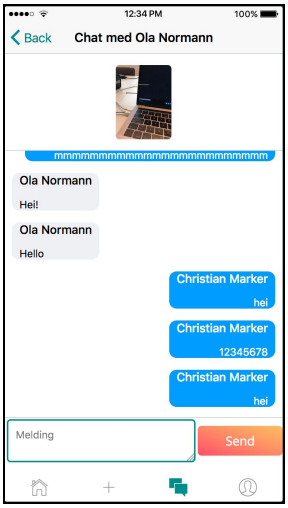
Meldinger er en chat-funksjonalitet som jeg implementerte sist i prosessen. Dette var fordi at det var en ekstrafunksjon av en signifikant størrelse som ville ta lang tid å implementere.
Jeg startet med å lage en meldingstjeneste som fungerte som mer som en mail-tjeneste, med at man kun kunne sende separate meldinger til hverandre, etter jeg fikk til dette gikk jeg over til en chat-tjeneste som hadde tråder med samtaler.
Meldingssiden består av en array som inneholder alle chater hvor brukeren enten er sender eller mottaker i, sortert etter hvilken chat som har den nyeste meldingen

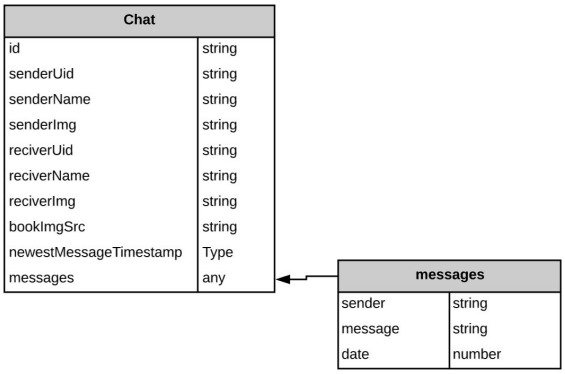
Når man starter en samtale med personen som selger boken man er interessert i så lages det en ny chat i databasen (vist under), dette dokumentet i databasen blir fylt med “metadata” om begge brukerne som er deltakere i samtalen, hvilken bok det snakkes om, når siste melding ble sendt samt en array over meldinger.
Når en mottakeren svarer på chaten som kommer opp i deres meldingsliste blir meldingsobjektet appendet på den eksisterende arrayen over meldinger, samt at feltet i samtalen som representerer timestampet på når den siste meldingen som ble sendt blir oppdatert til tiden svaret blir sendt.

Svakheter/restriksjoner

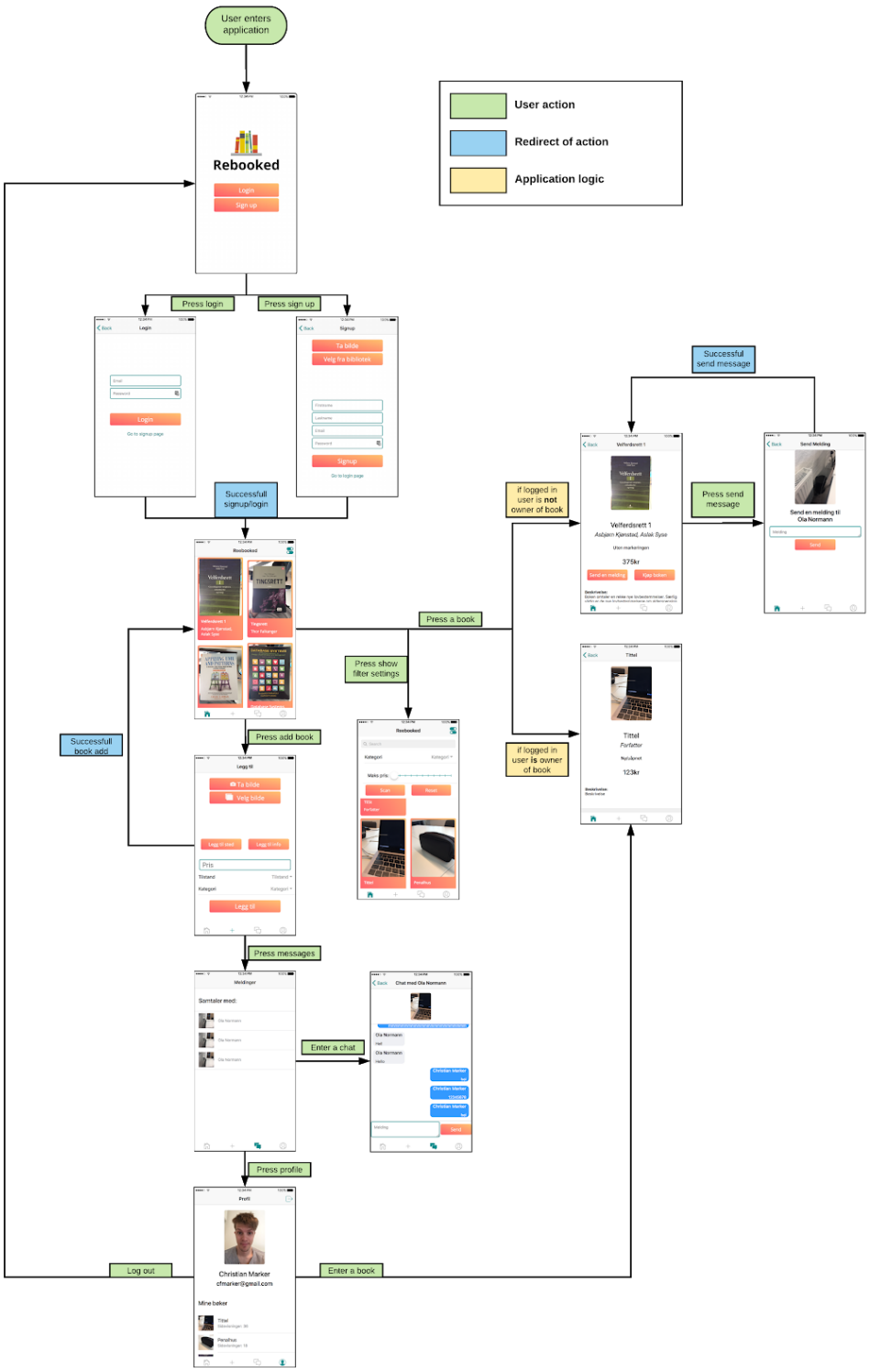
En bruker kan kun sende ett svar av gangen. Av en eller annen grunn så blir brukerens svar overskrevet hvis man prøver å sende flere meldinger på rad, men hvis man går ut og inn av samtalen så fungerer det å sende flere meldinger etter hverandre uten at de blir overskrevet.







Flytdiagram



Kilder

Jeg har generelt funnet løsninger på problemer ved å se på dokumentasjonene til Firebase og Ionic. Har ikke kopiert kode fra noen av stedene. Kun funnet svar på tekniske problemer.
Jeg har også funnet en del inspirasjon og kode fra koden som har blitt forelest i løpet av forelesningene.

Tab bar inspirert av ionic kode. Jeg har IKKE initialisert prosjektet med en tab layout

- https://ionicframework.com/docs/components/#tabs
- https://www.youtube.com/watch?v=vapQyFomRCo

Extra user info:

- https://firebase.google.com/docs/reference/js/firebase.User

post order by:

- https://firebase.google.com/docs/firestore/query-data/order-limit-data

Photo library

- https://github.com/apache/cordova-plugin-camera

Centering things

- https://www.w3.org/Style/Examples/007/center.en.html

Maps

- https://developers.google.com/maps/documentation/javascript/adding-a-google-map
- https://www.youtube.com/watch?v=0Ue6fNPOdB4

BookAPI

- https://developers.google.com/books/docs/v1/getting_started

LaunchNavigator

- https://ionicframework.com/docs/native/launch-navigator/
- https://github.com/dpa99c/phonegap-launch-navigator

Import json files

- https://hackernoon.com/import-json-into-typescript-8d465beded79

Firebase search

- https://firebase.google.com/docs/firestore/query-data/get-data

Style listview

- https://stackoverflow.com/questions/30581963/ionic-list-fixed-height-accordion

Sort array

- https://stackoverflow.com/questions/21687907/typescript-sorting-an-array

Lazy object initialisation

- https://basarat.gitbooks.io/typescript/docs/tips/lazyObjectLiteralInitialization.html