Dokumentasjon

Dette er hvordan vi har løst kravene til oppgaven.

I webapplikasjonen skal det inngå en backend database som kjøres på gruppas virtuelle maskin. Type database og hvordan denne brukes er opp til dere å bestemme, men grensesnittet til databasen skal være godt designet ihht. god praksis (bruk av REST ea).

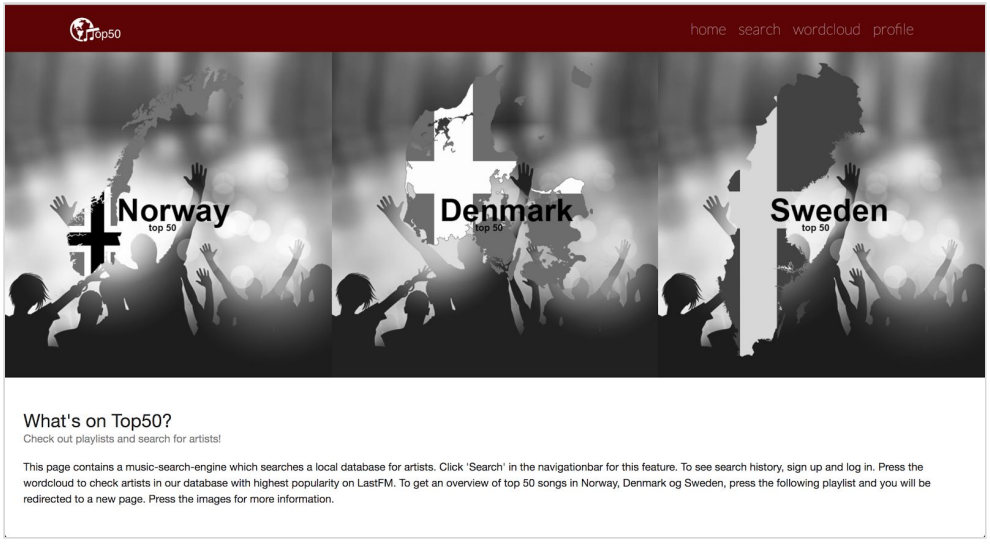
Vi har valgt å benytte oss av mongoDB som database på grunn av måten den er satt opp på. mongoDB er en såkalt NoSQL database, det vil si at dataen ikke er lagret i tabeller, men i JSON objekter. Dette gjør det meget enkelt å hente ut spesifikk data fra objektene, uten å bruke lange SQL kommandoer.

Dere skal demonstrere både skriving og lesing til databasen fra webapplikasjonen inklusive en form for søk (i praksis dynamisk brukerdefinert utvalg av det som skal vises). Generelt er det mye artigere å jobbe med en datamengde som gir et realistisk inntrykk (eksempelvis mulig å søke på forskjellige ting og få resultatsett som er forskjellige og har forskjellig antall). Bruk data dere finner på web, eller lag egne data.

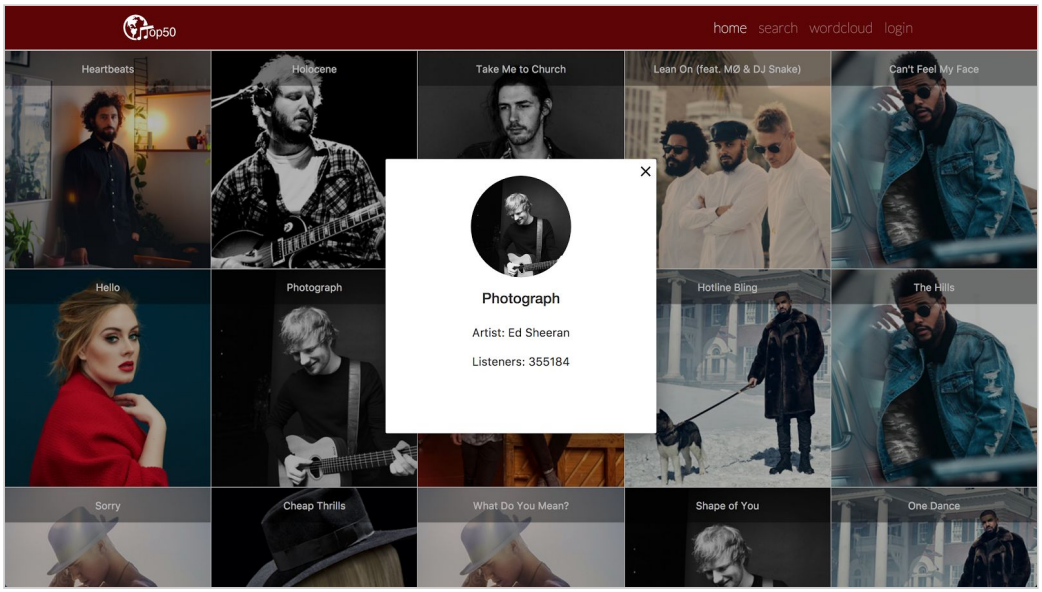
Vi har valgt en løsning hvor søk først ser etter data i vår database og returnerer det den finner der. Hvis det ikke finnes data der som matcher med søket så hentes data fra LastFM. All data som hentes fra LastFM blir lagret i vår database for senere søk. Dataen blir formatert etter modeller vi har definert i koden, både som schema for databasen og som en klasse i webappen.

Brukergrensesnittet skal ha listebasert visning med få detaljer for hver enhet, og hvor målet er å vise brukeren hva som er i databasen eller hva som er resultatet av et søk. Brukeren skal ha mulighet til å se flere detaljer for hver enhet enten i et eget vindu, eller ved at listen enheten i lista har expand/collapse egenskap.

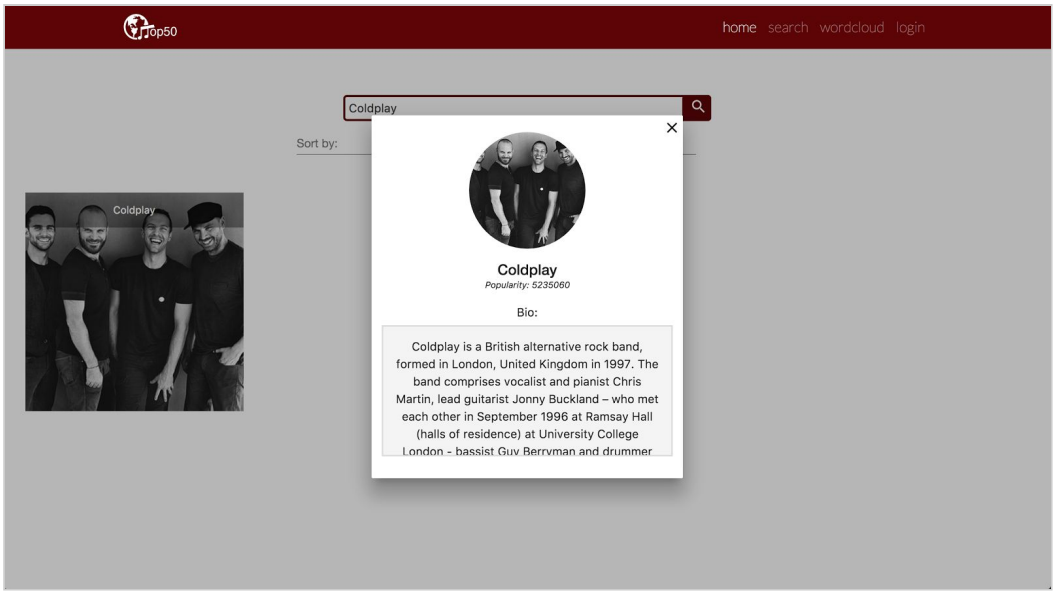
På forsiden kan brukeren velge mellom tre topp 50-spillelister.



Dersom bruker trykker seg inn på en spilleliste, vil hun/han få opp en gridlist med bilder av artist og navn på sang. Denne er sortert etter popularitet, da listen skal vise top 50. Personen kan trykke på bildet for å få opp mer informasjon om sangene i et eget pop-up vindu.

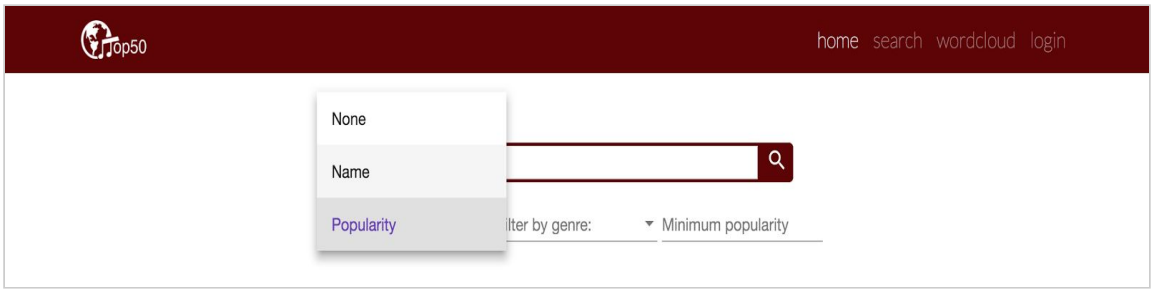


Dersom bruker trykker på “Search” i nav-baren vil han/hun kunne søke etter artister. Slik som på spillelistene, kan man trykke på bildet for mer informasjon, men her da altså om artist.



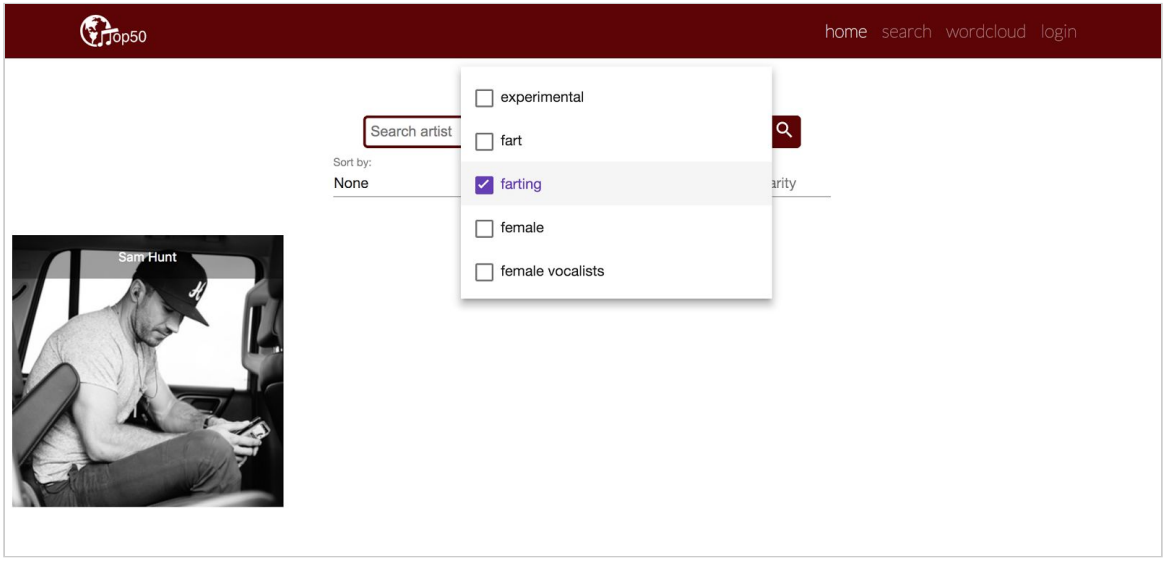
Den listebaserte visningen skal kunne sorteres på minimum to forskjellige egenskaper. Eksempel: etter at brukeren har fått returnert en liste etter et søk skal brukeren kunne bytte mellom forskjellige sorteringer.

På søk-siden kan brukeren sortere alfabetisk på artistnavn, popularitet, eller ingen.



Den listebaserte visningen skal kunne filtreres på minimum to forskjellige egenskaper. Eksempel: etter at brukeren har fått returnert en liste etter et søk skal brukeren kunne krysse av på en egenskap for å få begrenset antallet enheter i resultatsettet til kun de som har denne egenskapen.

Etter at man har søkt, kan man filtrere på artistenes sjanger. Man har også mulighet til å skrive inn minimum antall lyttere (popularitet). Denne informasjonen er hentet fra LastFM.

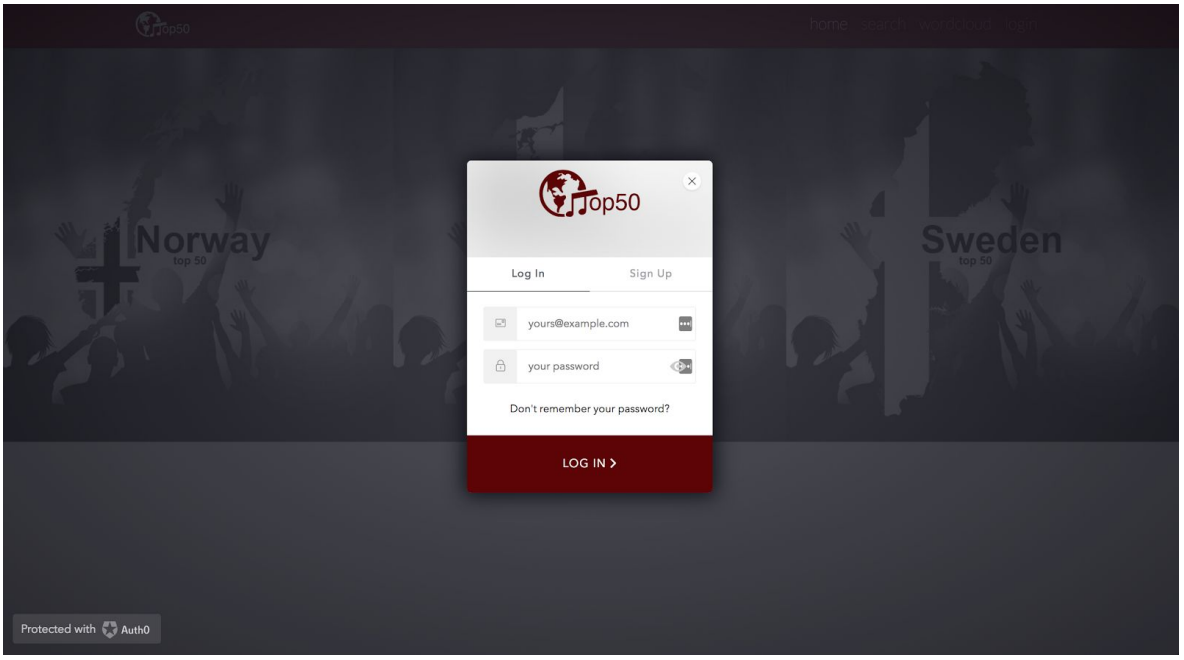


Den listebaserte visningen skal ha dynamisk lasting av data. Eksempel: etter et søk vises de 10 første treffene, men flere lastes når brukeren scroller eller ved blaing i sider.

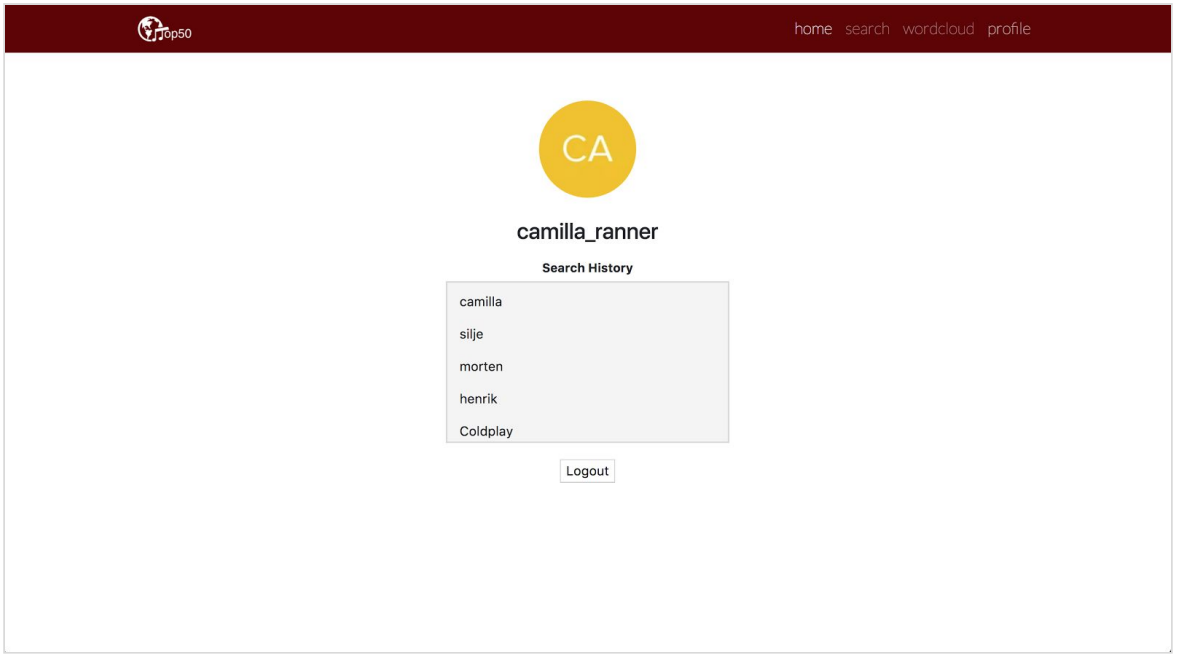
Vi benytter ngx-infinite-scroll. Først vises de 15 første bildene, deretter hentes de 5 av gangen fram til maks-grensen på 30 bilder nås. Vi har også brukt @angular/animations og lagt på en “ease in”-animasjon på bildene når de hentes inn.

Webapplikasjonen skal ha "min side"-funksjonalitet som i praksis betyr at en bruker skal kunne logge seg på og at det blir registrert noe fra brukerens søkeaktivitet f.eks. hva brukeren har sett på tidligere eller søkene som brukeren har brukt.

Øverst i navbaren finner man innlogging. Trykker du på login kommer du til en side hvor man kan registrere seg/logge på.



Etter du har logget deg inn eller registrert deg, blir du videreført til forsiden. Da står det ikke lenger “login” i navbaren, men “profile”. Trykker du på “profile” vil du komme til en ny side med oversikt over egen profil og søkehistorikk.

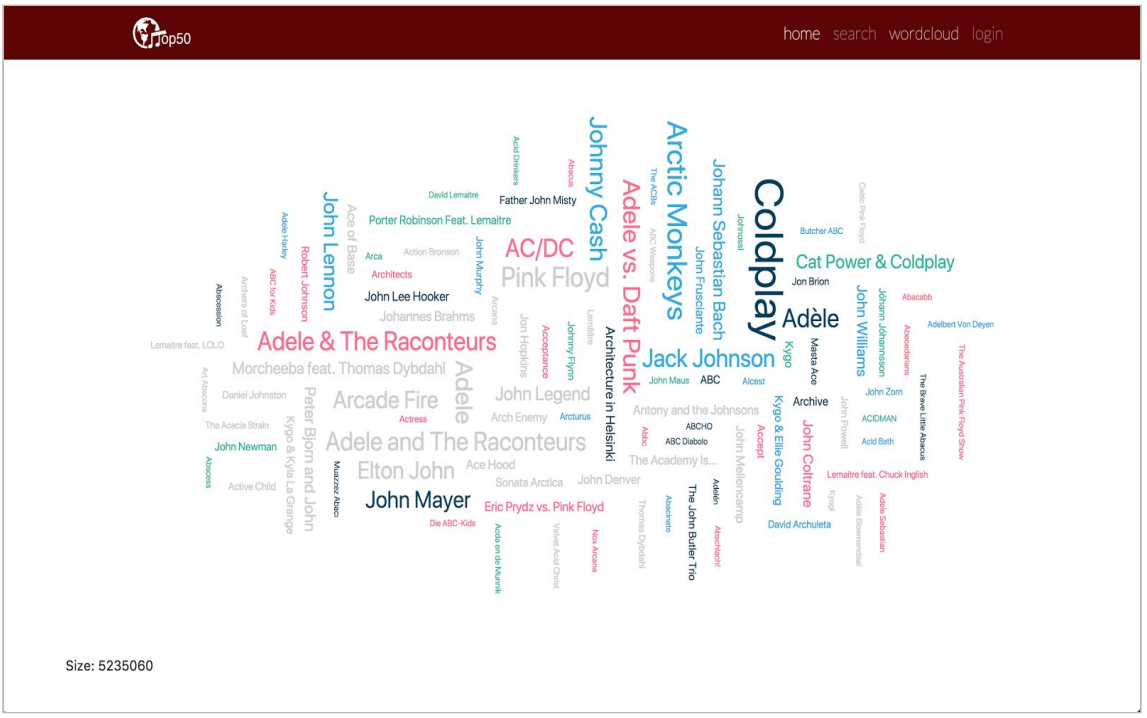


Webapplikasjonen må implementere "session"-håndtering (som du f.eks. trenger for å implementere dynamisk lasting, min side, og filtrering/sortering som skal fungere med sidevisning).

Ved hjelp av Auth0 har vi implementert en login funksjon som holder styr på brukeren er logget inn eller ikke. Dersom brukeren er logget inn, vil alle søk han/hun gjør bli lagret og vist på profilsiden.

Webapplikasjonen skal ha et litt "fancy" alternativ visning av listen f.eks. visning på kart eller visuell grafisk fremstilling av data, ordsky ea.

Ordskyen vår henter alle artistene i vår database, og rangerer størrelsen etter popularitetsrankingen de har fått i LastFM. Holder du pilen over et av artistnavnene vil du få opp size (antall lyttere) nede i venstre hjørne.



Kode skal være testet og funksjonaliteten skal være godt utprøvd og feilfri.

Hadde problemer med å sette opp testing som fungerte for komponenter, har skrevet tester for pipen og artist service. Den genererte testen for callback component var den eneste av de genererte testene som kjørte. Problemet vi hadde med å få kjørt tester er knyttet opp mot hvordan vi router/refresher sider. På grunn av dette må vi ha alle komponenter tilgjengelig for routing i testene for hver komponent som kan routes til/fra. Når en eller flere av disse har dependencies som ikke er så lett å få importert eller mocket så tar det for mye tid. Vi har dermed ikke fått skrevet nok tester for appen vår.