Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

«2030 - Sanntidsquiz for Bærekraft»

«Utforsking av vurderinger ved valg av utviklingsmetodikk»

Systemdokumentasjon

Versjon 2.2

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
25/04/2022	1.0	Første versjon av systemdokumentasjonen	Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen
08/05/2022	2.0	Andre versjon av systemdokumentasjonen	Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen
14/05/2022	2.1	Iterasjon av systemdokumentasjon. Oppdatering av diagrammer	Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen
19/05/2022	2.2	Iterasjon av systemdokumentasjon. Oppdatering av diagrammer	Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen

INNHOLDSFORTEGNELSE

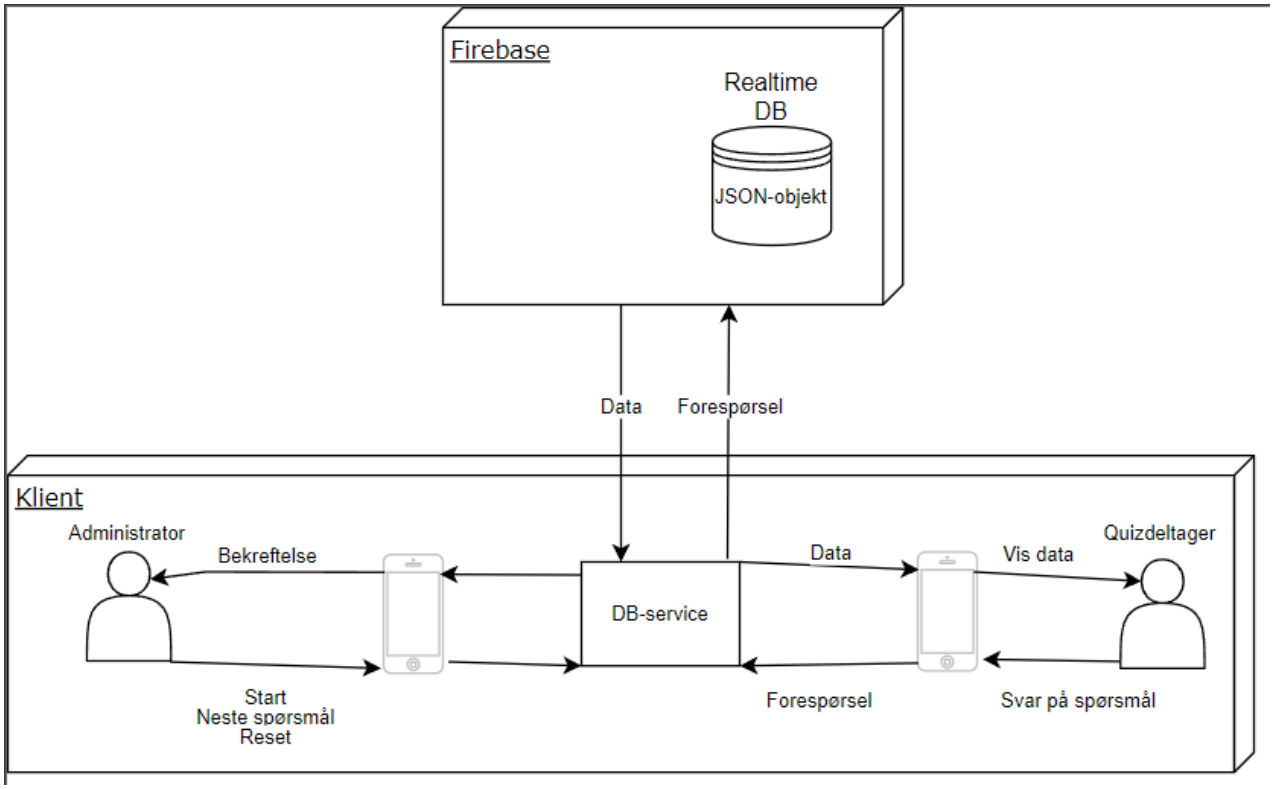
FIGURLISTE

1 INNLEDNING

Dokumentet er utformet i forbindelse med bachelorprosjektet til, og er skrevet av, Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen ved Høgskulen på Vestlandet, våren 2022.

Dokumentet viser hvordan applikasjonen er bygget opp og teknologien som ligger til grunn. Det inneholder også en oversikt over systemarkitektur, kildekodestruktur med dokumentasjon, klassediagram, databasemodell med tilhørende beskrivelse, server-tjenester, installasjon og kjøring, og litt om sikkerhet.

2 ARKITEKTUR



Figur 1 - Arkitekturdiagram

Firebase:

- Firebase er en tjeneste fra google for å håndtere backend logikk. Vi bruker Firebase som «PaaS» plattform som en tjeneste for å hoste databasen som blir brukt til appen.

Realtime DB:

- Databasen for applikasjonen er en «Realtime database» (Google Developers, 2022) fra Firebase. Den er bygget opp som et enkelt JSON objekt.

DB-Service:

- En intern klasse hos klienten som håndterer dataflyt mellom klient og Firebase.

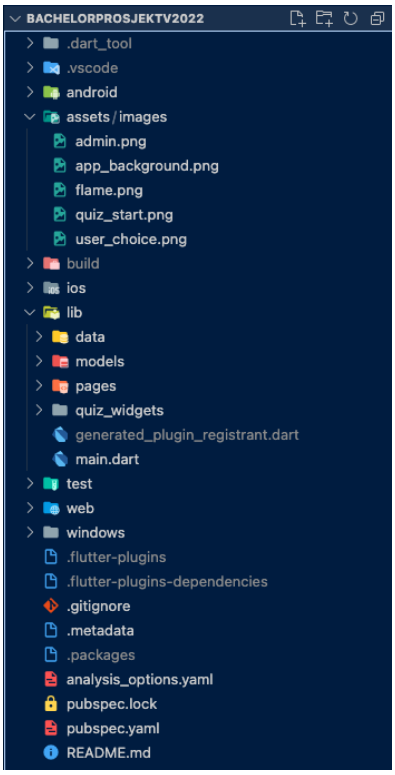
Quizdeltager:

- Det er en person som tar del i quizen. Brukere kan kun svare på spørsmål. Alle brukere kommuniserer med den samme databasen for henting av data.

Administrator:

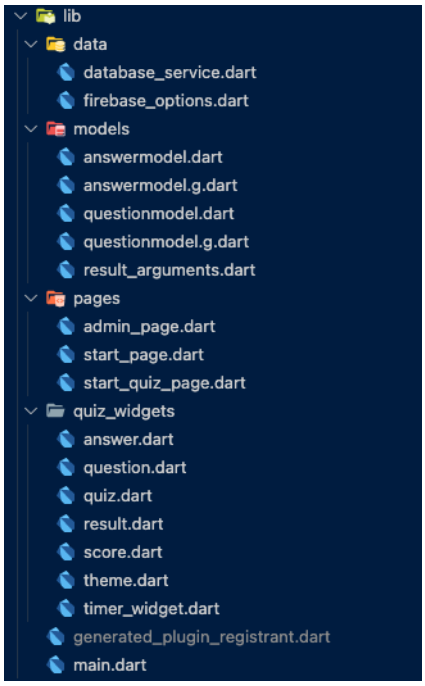
- Det er en person som styrer gjennomgangen av quizen. Administrator sender oppdateringer til Firebase databasen som endrer og sender verdier ut til quizdeltagere som sjekker på endringer av verdier.

3 PROSJEKTSTRUKTUR



Figur 2 - Prosjektstruktur

Den overordnede prosjektstrukturen er vist i Figur 2. Alt utenom /lib og assets/images er automatisk generert av Flutter. Siden rammeverket fungerer på flere plattformer ligger det også tilhørende programkode for hver plattform under iOS, Android og Web.



Figur 3 - Innhold i /lib

I /lib ligger all Flutterkode som er produsert av teamet, vist i Figur 3. Her har teamet delt opp koden i forskjellige mapper for å organisere en logisk mappestruktur. I mange Flutterprosjekt gitt av Googles Flutter Team ligger mye logikk i «main.dart» (Flutter.dev, 2022). Teamet ønsket å dele opp løsningen mer, slik at det er lettere å holde oversikt.

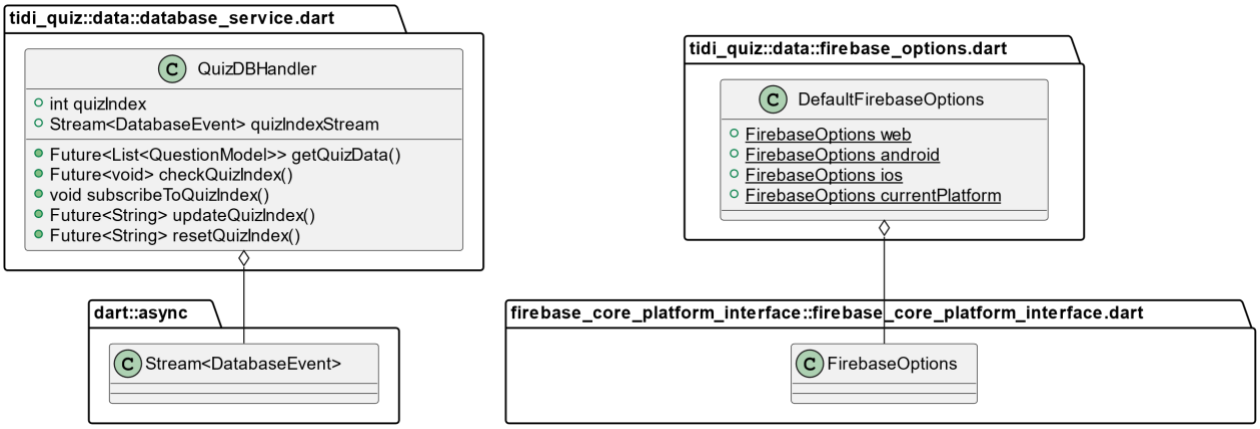
4 KLASSEDIAGRAM

Flutter er et rammeverk som bygger på programmeringsspråket Dart. I Flutter representerer alle UI-elementer en Widget. Dette er en standardklasse som alle synlige komponenter arver fra og mange av klassene i applikasjonen er derfor Widgets. To viktige subklasser av Widgets er Stateless og Stateful. Stateless er en klasse som bare viser frem UI-komponentene. Stateful er en klasse som kan ta vare på data og kalle på setState()-metoden for å fortelle Flutter at vi ønsker å vise Widgeten på nytt med oppdatert data. Det er bare komponenten som kaller setState() som blir vist på nytt.

De ulike sub-systemene er delt opp på følgende måte, vist i Figur 4, Figur 5, Figur 6 og Figur 7:

Data:

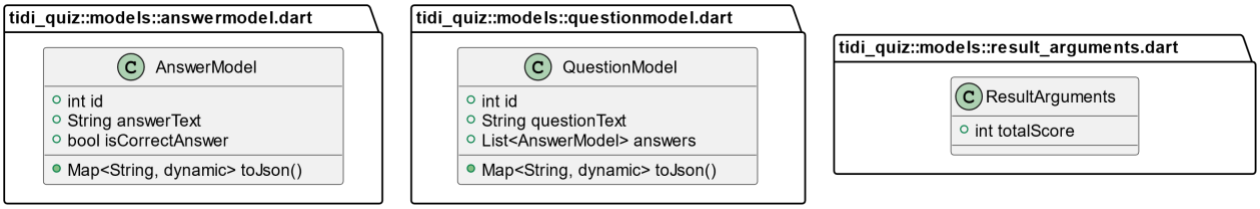
QuizDBHandler blir brukt for å hente og sende data til databasen. Hver widget som på en eller annen måte jobber med data har et objekt av denne typen som håndterer interaksjon med databasen hos firebase. FirebaseOptions holder konfigurering av verdier for tilkobling til firebase.



Figur 4 - Data klassediagram

Models:

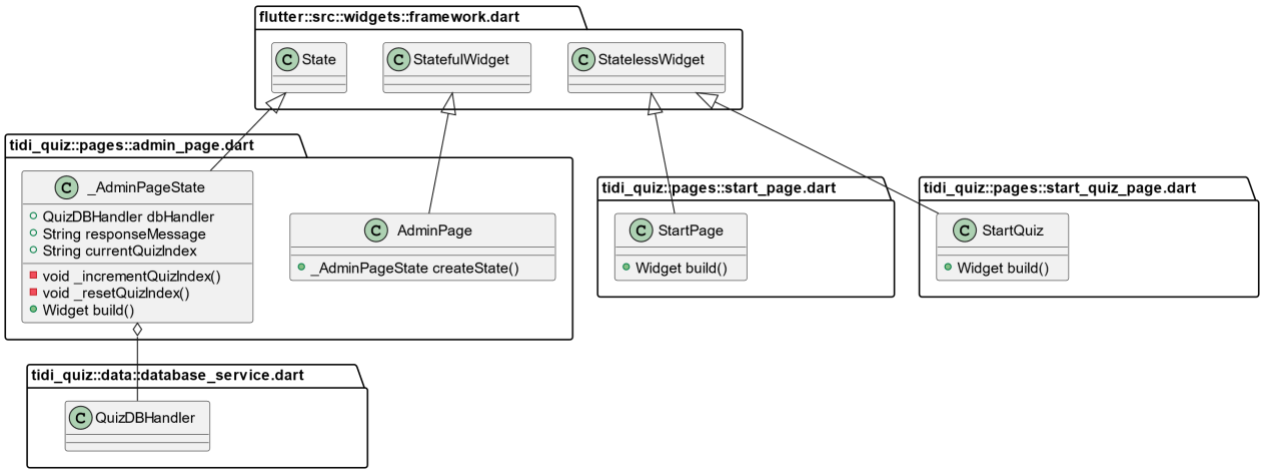
Modellene mapper JSON data fra databasen om til dart objekter som kan brukes direkte i koden i tillegg til å mappe fra et objekt til JSON for å sende data til databasen.



Figur 5 - Models klassediagram

Pages:

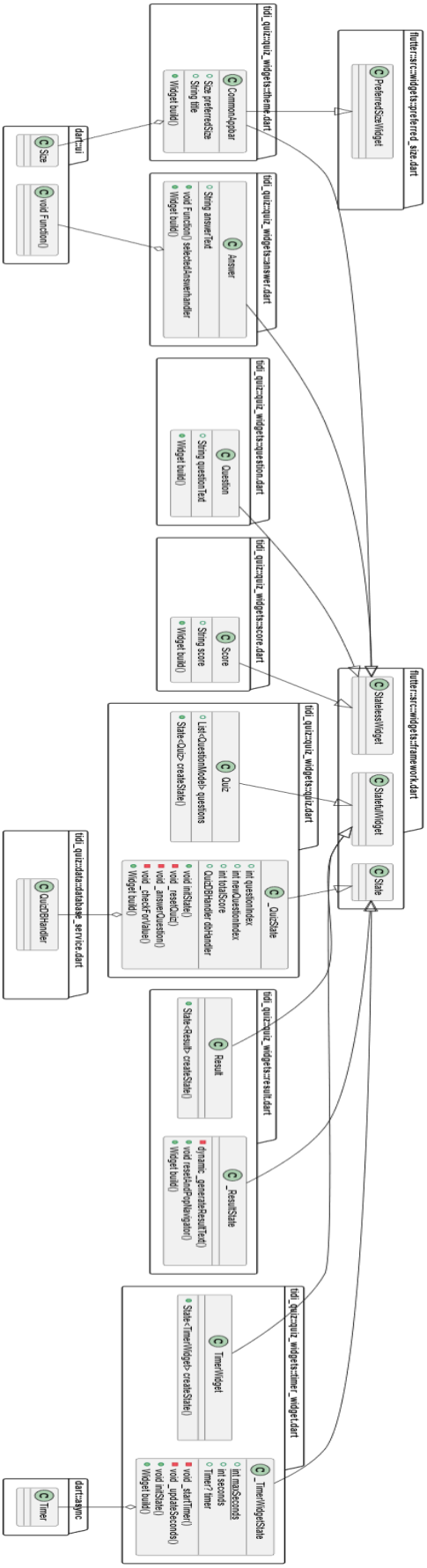
Dette er de ulike sidene som en bruker interagerer med. Det er da større widgets som utgjør hovedkomponenten for UI bygd opp av flere mindre widgets.



Figur 6 - Pages klassediagram

QuizWidgets:

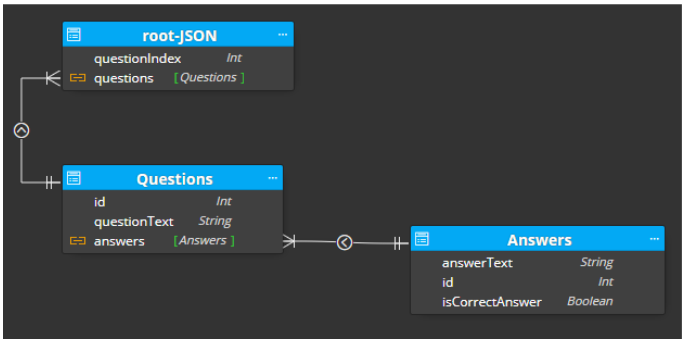
Quizwidgets er en samling av klasser som håndterer det meste av quiz-flyten. Det er alt fra visning av spørsmål, logikk for å vente på administrator sine styringer, poengutregning og videresending til resultat for quiz.



Figur 7 - QuizWidgets klassediagram

Det komplette klassediagrammet er vedlagt under Vedlegg 1 - Klassediagram. På grunn av bildekvalitet vil det komplette klassediagrammet også legges ved som en SVG-fil sammen med kildekoden.

5 DATABASEMODELL



Figur 8 - Databasemodell av JSON objekt

Databasen består av ett enkeltstående JSON-objekt som inneholder en liste av «questions» og en indeks som har kontroll på hvilket spørsmål som er det gjeldende til enhver tid, vist i Figur 8.

Hvert element i listen av «questions» inneholder en int «id», en string «questionText» og en liste av «answers». Hvert element i «answers»-listen inneholder en int «id», en boolean «isCorrectAnswer» og en string «answerText».

6 SERVER-TJENESTER

Firebase tar seg av server-tjenestene for applikasjonen. Prosjektet bruker «Realtime Database» som er en skyløsning fra Firebase som man samhandler med via Realtime Database API. I Flutter bruker man en programvareutvidelse «Firebase_core» sammen med en tredjeparts kodepakke «firebase_database» (Google, 2022) som lar et Flutterprosjekt koble seg opp til API-et. Logikken for samhandlingen ligger i QuizDbHandler klassen, se Figur 4.

7 SIKKERHET

Tilgangen til databasen i Firebase kan styres med egendefinerte regler. De definerte reglene bestemmer hvem som har lese- og skriverettigheter til databasen (Google Dvelopers, 2022). Ved bruk av «Firebase Authentication» kan man bestemme at det bare er autentiserte brukere som får lov til å gjøre databaseoperasjoner. Autentisering kan skje via Google- eller Facebook-konto.

Mobilapplikasjonen har ikke autentisering enda og databasen er satt opp som testmiljø der alle som ønsker har lese og skriverettigheter som er en generell sikkerhetsrisiko for dataen i databasen.

En annen svakhet med sikkerheten er at det er utviklingsteamet som er ansvarlig for å sette opp sikkerheten for databasen, ikke en databaseadministrator, men siden denne applikasjonen ikke lagrer sensitiv data er ikke dette en relevant problemstilling.

Applikasjonen er også avhengig av flere tredjeparts kodepakker og en bruker kan ende opp i en situasjon hvor man må vente med å gjøre en oppdatering til alle pakker er oppdaterte og kommuniserer med hverandre igjen. Dersom det dreier seg om en større sikkerhetsoppdatering, kan dette føre til at applikasjonen ikke kan brukes før alle de ulike pakkene er oppdatert. Mange av de kommer fra åpen-kildekode og det er ingen garanti for at oppdateringene kommer raskt nok.

8 INSTALLASJON OG KJØRING

8.1 Kjøring av applikasjonen som utvikler

Applikasjonen kjøres ved hjelp av Visual Studio Code og en Android eller iOS emulator fra Android Studio eller Xcode. Man kan også kjøre applikasjonen i Chrome nettleser. Flutter SDK kommer via en programvareutvidelse som lastes ned til Visual Studio Code. Dart SDK er inneholdt i Flutter-utvidelsen.

8.2 Deployering av applikasjonen

Applikasjonen er ikke klar for å deployeres og sendes til godkjenning hos App Store eller Google Play. For å gjøre dette var tanken å følge guiden til Flutter.dev (Flutter.dev, 2022). Hovedoppgavene vil da være å signere appen med ID, bygge en bundle (Developer.apple, 2017), få testet applikasjonen og deretter publisere den.

Applikasjonen kan publiseres i App store og Google Play når den er ferdigutviklet og testet. Dette kan gjøres ved å følge oppskrifter laget av App Store og Google Play selv.

9 DOKUMENTASJON AV KILDEKODE

Systemdokumentasjonen var tenkt å genereres ved hjelp av DartDoc (tools.dart.dev, 2022) men på tidspunktet den ble forsøkt brukt, virket ikke pakken slik den skulle.

Kildekoden er i vedlagt zip-fil.

10 KONTINUERLIG INTEGRASJON OG TESTING

Til tross for at testing kan være veldig nyttig for å utelukke feil i koden har teamet ikke prioritert gjennom utviklingen av applikasjonen å lage enhetstester. Det teamet har gjort derimot var mye brukertesting ved kjøring av applikasjonen. Å skrive enhetstester ble vurdert til å stjele tid fra å få på plass viktig funksjonalitet. Siden teknologien var ny for teamet kunne det gått mye tid på å utvikle tester istedenfor å utvikle produktet.

Kontinuerlig integrasjon ble heller ikke gjennomført. Teamet hadde kontinuerlig integrasjon av kildekoden på en «master-branch» via kodegjennomganger, men applikasjonen var ikke satt opp i et isolert system med automatisk bygging og testing av prosjektet.

11 REFERANSER

Google Developers, 2022. *Firebase Documentation.* [Internett]
Available at: https://firebase.google.com/docs/database
[Funnet 25 April 2022].

Google Dvelopers, 2022. *Understand Firebase Realtime Database Rules.* [Internett]
Available at: https://firebase.google.com/docs/database/security
[Funnet 26 April 2022].

tools.dart.dev, 2022. *Dart documentation generator.* [Internett]
Available at: https://pub.dev/packages/dartdoc
[Funnet 26 April 2022].

Flutter.dev, 2022. *simplistic_editor/lib/main.dart.* [Internett]
Available at: https://github.com/flutter/samples/blob/master/simplistic_editor/lib/main.dart
[Funnet 16 Mai 2022].

Flutter.dev, 2022. *flutter/_maps_firestore/lib/main.dart.* [Internett]
Available at:
https://github.com/flutter/samples/blob/master/flutter_maps_firestore/lib/main.dart
[Funnet 16 Mai 2022].

Flutter.dev, 2022. *Build and release an Android App.* [Internett]
Available at: https://docs.flutter.dev/deployment/android
[Funnet 16 Mai 2022].

Developer.apple, 2017. *About Bundles.* [Internett]
Available at:
https://developer.apple.com/library/archive/documentation/CoreFoundation/Conceptual/CFBundles/AboutBundles/AboutBundles.html#//apple_ref/doc/uid/10000123i-CH100-SW1
[Funnet 16 Mai 2022].

tools.dart.dev, 2022. *Dart documentation generator.* [Internett]
Available at: https://pub.dev/packages/dartdoc
[Funnet 03 Mai 2022].

Google, 2022. *Firebase core for Flutter.* [Internett]
Available at: https://pub.dev/packages/firebase_core
[Funnet 14 Mai 2022].

Google, 2022. *Firebase database plugin.* [Internett]
Available at: https://pub.dev/documentation/firebase_database/latest/
[Funnet 26 April 2022].

12 VEDLEGG

12.1 Vedlegg 1 - Klassediagram