By: S189994

University of suffolk

Firebase based APplicatioN



Report

**Introduction**

The aim of this report is to explain implementation details of the elements of Firebase that were used for the completion of the application, along with wireframes which were created before the application was developed.

**What is Firebase?**

Firebase is a service founded in 2011 and acquired by Google in 2014. The Firebase Real-time Database is a cloud-hosted database.

Written in the Firebase docs (Firebase, n.d.) is said that:

“*Data is stored as JSON and synchronized in real-time to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all your clients share one Real-time Database instance and automatically receive updates with the newest data”*

Firebase sends the new data as soon as it’s updated, which is why refreshing the web page isn’t required to receive results. According to Chris Espin (Espin, 2016):

“When you connect your app to Firebase, you’re not connecting through normal HTTP. You’re connecting through a Web Socket.”

More information about the web sockets, we can read in Hałabuda’s blog (Hałabuda, 2016) about “How to use Firebase in Mobile applications:

“Communication between database and client application is handled via web sockets. That’s actually one of the tricks Firebase uses to provide the real time functionality to developers and end users. Of course, such approach forces you to handle your mobile application code more efficiently.”

There are many features of the Firebase, which integrate the database platform. These include, but aren’t limited to:

* Cloud Firestore
* Realtime Database
* Remote Config
* Firebase Authentication
* Google Analytics for Firebase
* Crashlytics
* Cloud Storage
* Cloud Pub/Sub
* HTTP

**Implementations**

In this section there will be given examples of the elements which were implemented into the Application.

The first thing that had to be done before everything else, was to configure Firebase in Android studio. Following the steps given in Android Studio helped me set the database up. Afterwards I started implementing:

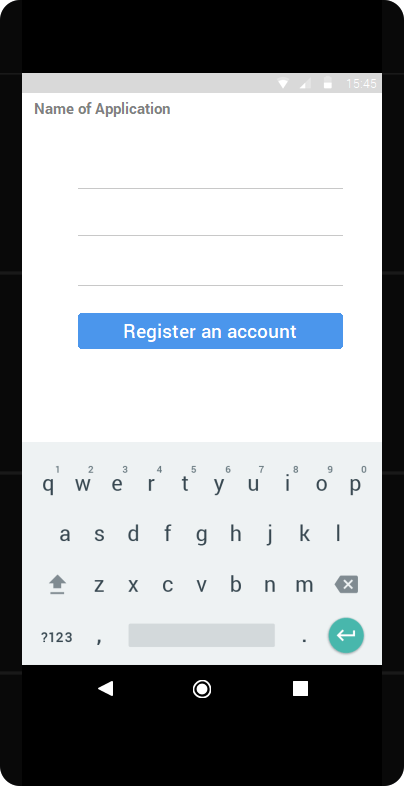
1. Cloud Firestore: As said in the official Firebase website (Firebase, n.d), the Firestore is flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.

“*It keeps your data in sync across client apps through real-time listeners and offers offline support for mobile and web, so you can build responsive apps that work regardless of network latency or Internet connectivity. “*

It was implemented because it is “expressive and efficient” (Firebase, n.d)

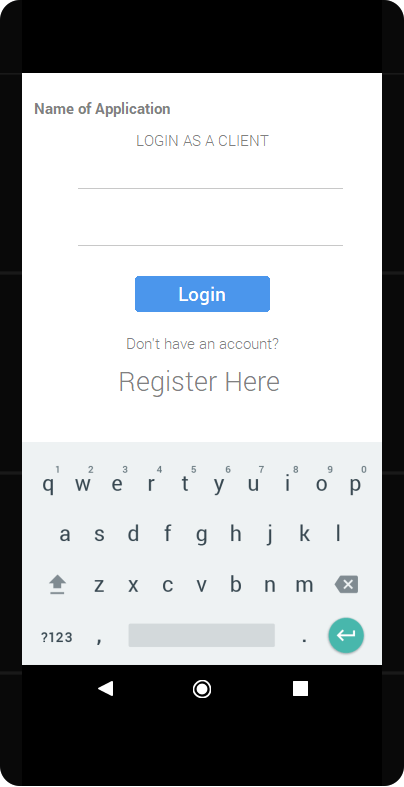
“*Following Cloud Firestore's NoSQL data model, you store data in documents that contain fields mapping to values. These documents are stored in collections, which are containers for your documents that you can use to organize your data and build queries.”*

Knowing the above, I created a collection with the title “MapEvents” and at first, I added test documents, which led to real ones in the process of development.

1. Authentication “FirebaseUI”( Fig1; Fig2)- This is a library built on top of the Firebase Authentication SDK that provides drop-in UI flows for use in your app.

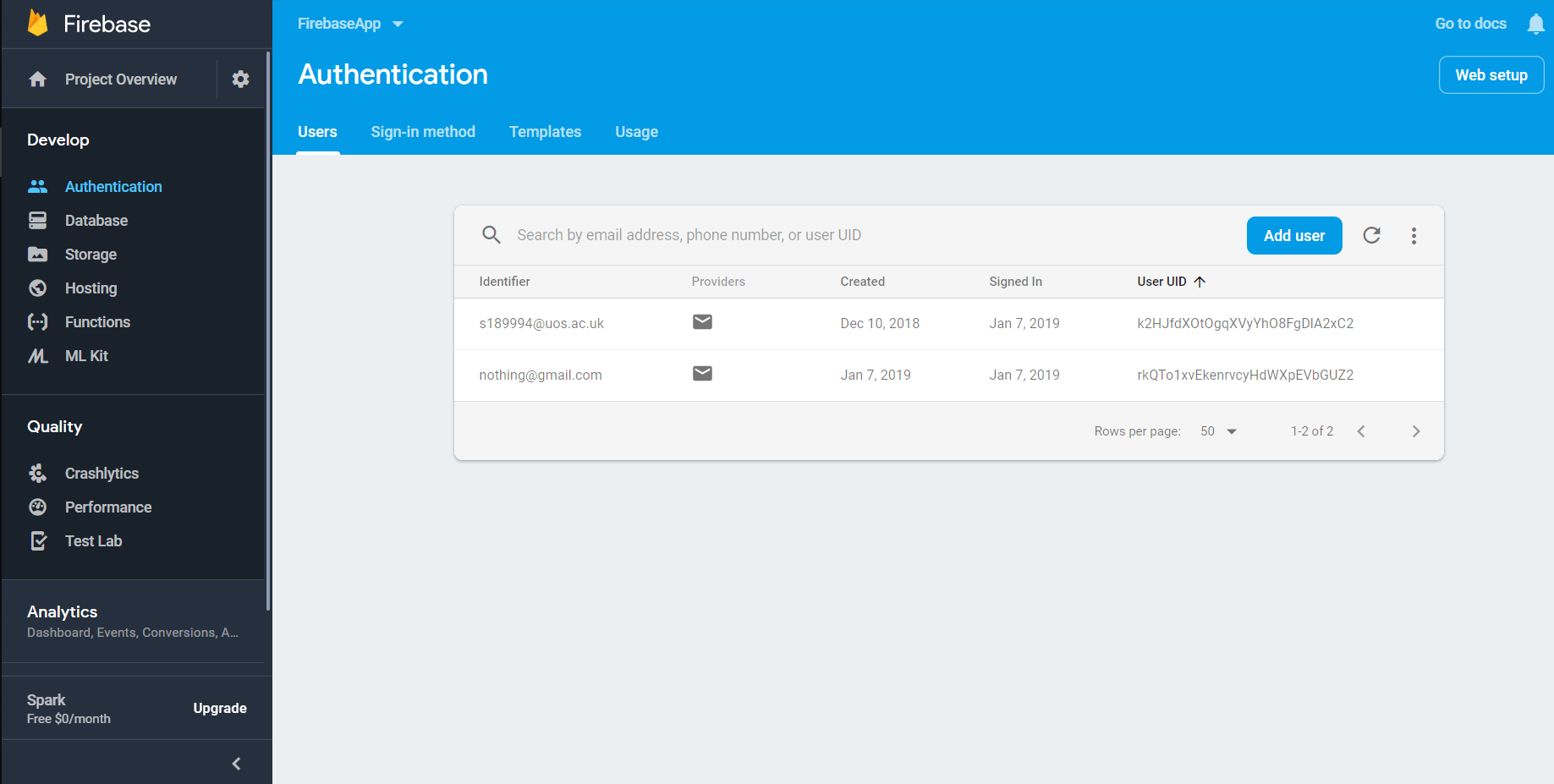
Figure

FirebaseUI provides the following benefits (Firebase, n.d)

* Multiple Providers - sign-in flows for email, phone authentication, Google Sign-In, Facebook Login, and Twitter Login.

Figure

* Account Management
* Account Linking
  + - * Custom Themes
* Smart Lock for Passwords

The two accounts used for uploading information into the Database are shown below (Fig3)

Figure

Using an email with a created password specifically for the application log-in, a user can easily sign in or sign up with the application. Signing in provides the user with the ability to upload into the application.

1. Play services “MAPS” - needed for creating the Maps functions to work, as “The Google Play services APK contains the individual Google services and runs as a background service in the Android OS.” (Google Developers, n.d). The development process went through getting an API key from the official website which lead to implementing permissions “GET\_ACCOUNTS"; “READ\_PROFILE"; “READ\_CONTACTS";” READ\_EXTERNAL\_STORAGE";” WRITE\_EXTERNAL\_STORAGE";” CAMERA"; “ACCESS\_FINE\_LOCATION";”ACCESS\_COARSE\_LOCATION".

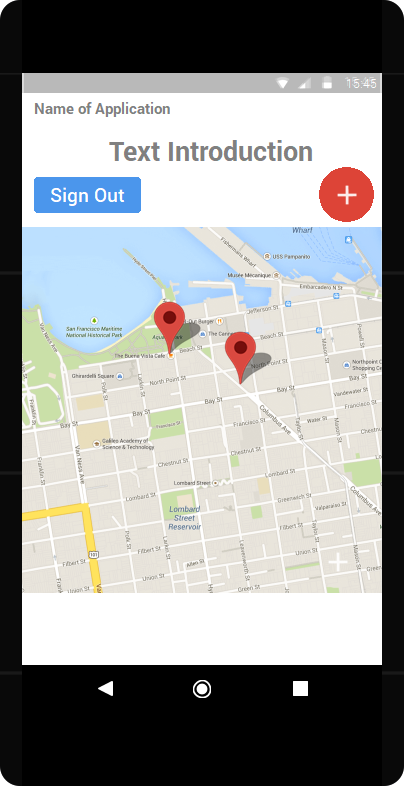
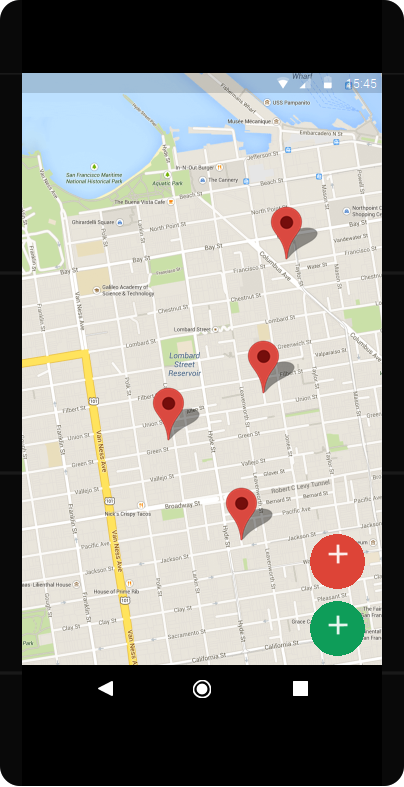
At first (Fig4), when I was creating the wireframe for the main page (the map) the floating buttons were in a different place, as well as the map - Map view. The red button is for login and uploading and the sign out button has an obvious function.

Figure 5

Figure 4

The changes can be seen on the right (Fig5). The “Map view” became a “Map activity”, the buttons went on the bottom right. This is so they can be

easily accessible to the user with their touch

input, while also be clearly visible. The

Green Floating action button is for

authentication if the user hasn’t signed in

already and then used for uploading into the

storage. The red button remains hidden until a user has been

accessed, it’s function is to solely sign them back out. There were

made other changes with the flow of development the

application.

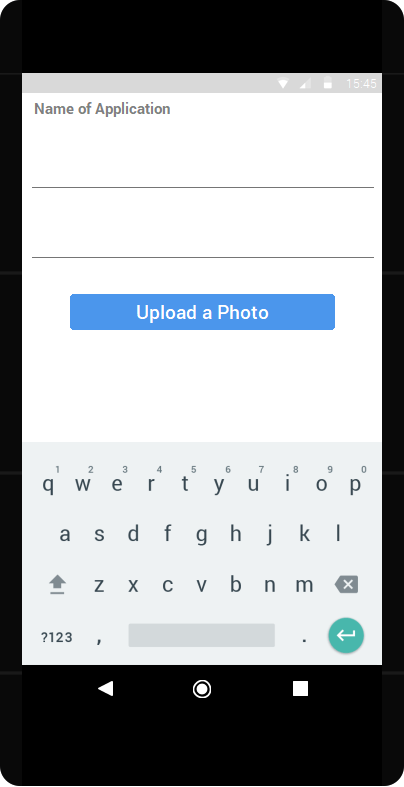
1. Firebase storage “lets you upload and share user generated content, such as images and video. It allows you to build rich media content into your apps” (Firebase, n.d). In this application’s case, the users can upload images of wild life/plants, along with their chosen title and description depicting the information they desire. As an example, they could provide a detailed description around the event, more details about the location or what was found and uploaded. Implementing it was one of the first steps in developing, because is one of the core function of the application. By Upload (Fig6), the information will go directly to the Firebase and it will show on the map as a marker. When tapped on, the marker leads to another dialog window, with the already uploaded information and picture enabling anyone to see and discover this information.

Figure 6

Other Implementations:

* Picasso (Square.github.io, n.d.) – used library for the image appearing in the marker’s dialog window.
* Concurrent using async (Prendota, 2018)
* Dialog windows – used “Creating a custom layout”. Code helped from Android Developers (Android Developers, n.d.)

**References**:

Android Developers. (n.d.). *Dialogs | Android Developers*. [online] Available at: https://developer.android.com/guide/topics/ui/dialogs [Accessed 11 Jan. 2019].

Android Developers. (n.d.). *Connect to Firebase  |  Android Developers*. [online] Available at: https://developer.android.com/studio/write/firebase [Accessed 11 Jan. 2019].

Esplin, C. (2016). *What is Firebase? – How To Firebase.* [online] How To Firebase. Available at: https://howtofirebase.com/what-is-firebase-fcb8614ba442 [Accessed 11 Jan. 2019].

Hałabuda, P. (2016). *How To Use Firebase In Mobile Apps Development.* [online] Whalla Labs. Available at: http://whallalabs.com/how-to-use-firebase-in-mobile-applications/ [Accessed 11 Jan. 2019]

Google Developers. (n.d.). *Set Up Google Play Services | Google APIs for Android* | Google Developers. [online] Available at: https://developers.google.com/android/guides/setup [Accessed 11 Jan. 2019].

Firebase. (n.d). *Cloud Firestore | Firebase.* [online] Available at: https://firebase.google.com/docs/firestore [Accessed 11 Jan. 2019].

Firebase. (n.d.). *Firebase Authentication  |  Firebase*. [online] Available at: https://firebase.google.com/docs/auth/ [Accessed 11 Jan. 2019].

Firebase. (n.d.). *Firebase Realtime Database | Firebase Realtime Database* | Firebase. [online] Available at: https://firebase.google.com/docs/database/ [Accessed 11 Jan. 2019].

Firebase. (n.d.). *Easily add sign-in to your Android app with FirebaseUI  |  Firebase*. [online] Available at: https://firebase.google.com/docs/auth/android/firebaseui [Accessed 11 Jan. 2019].

Prendota, A. (2018). *Kotlin/kotlinx.coroutines.* [online] GitHub. Available at: https://github.com/Kotlin/kotlinx.coroutines/blob/master/docs/composing-suspending-functions.md#concurrent-using-async [Accessed 11 Jan. 2019].

Square.github.io. (n.d.). *Picasso.* [online] Available at: http://square.github.io/picasso/ [Accessed 11 Jan. 2019].