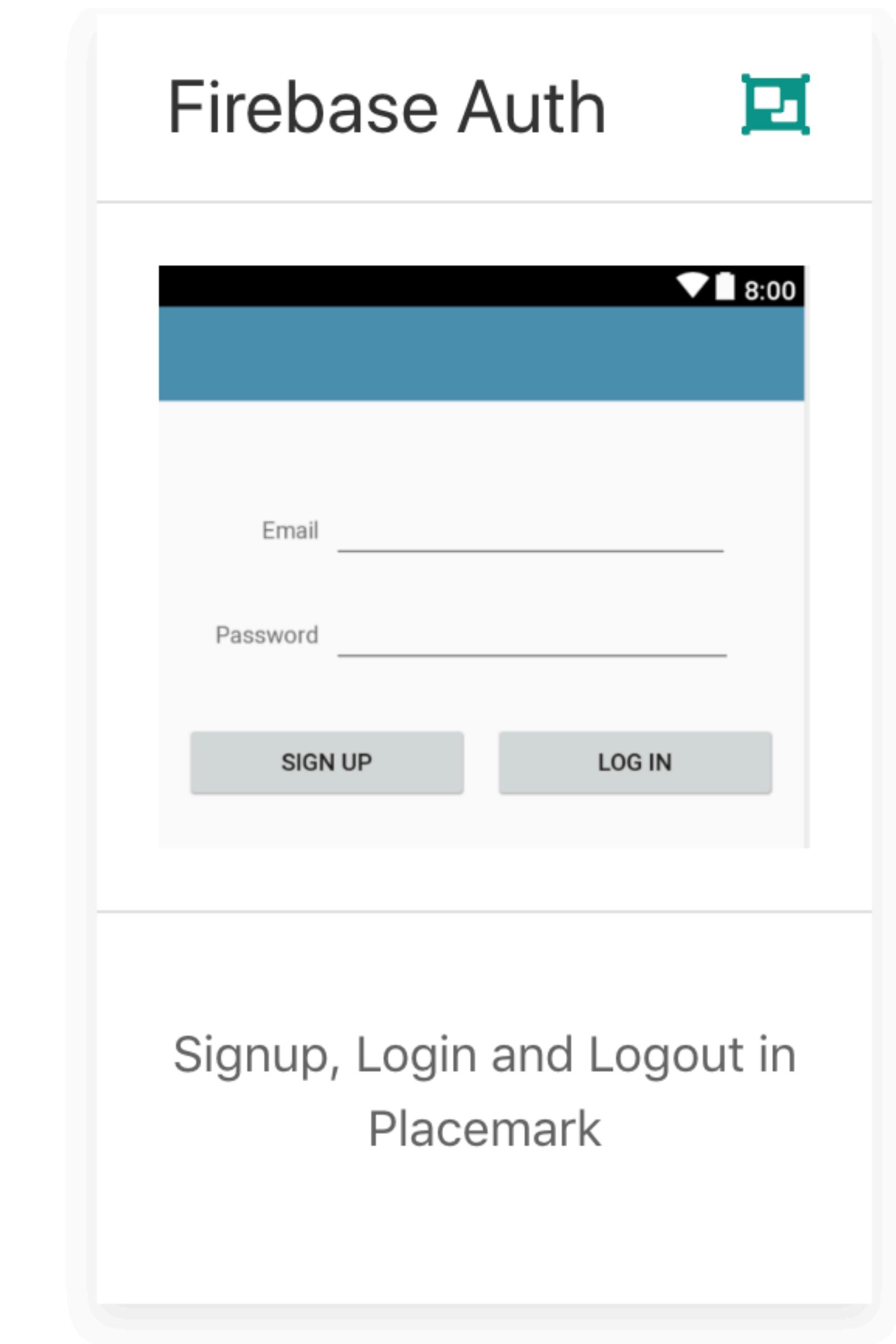


Firebase Auth



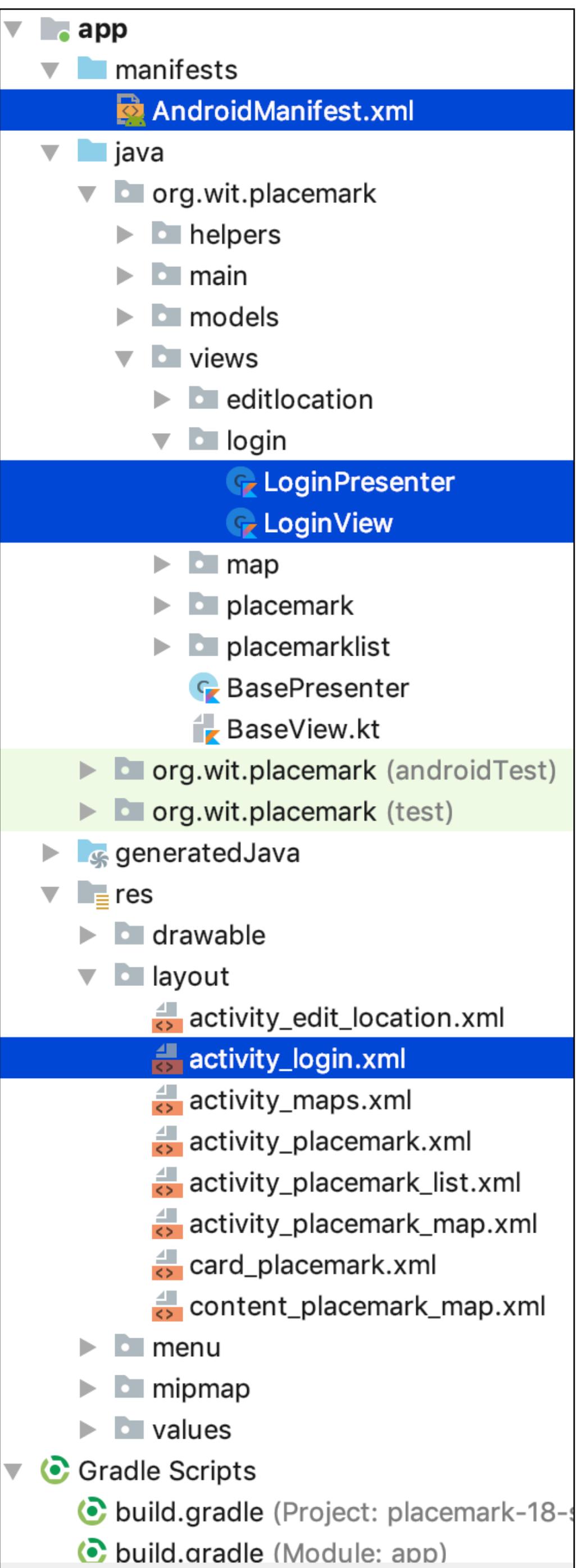


Authentication

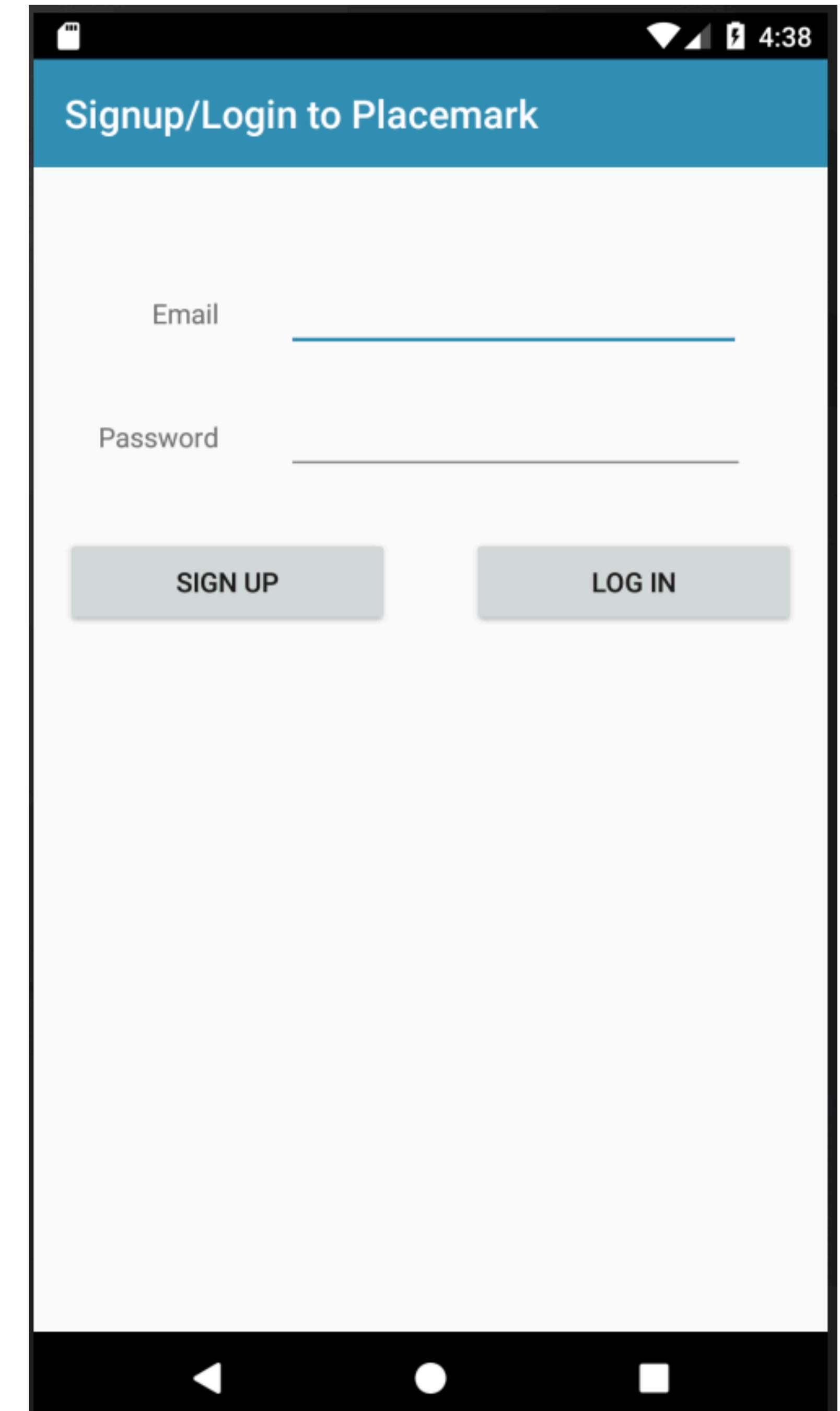
iOS

Manage your users in a simple and secure way.

Firebase Auth offers multiple methods to authenticate, including email and password, third-party providers like Google or Facebook, and using your existing account system directly. Build your own interface, or take advantage of our open source, fully customizable UI.



Introduce new View/ Presenter/ Layout



Presenter/View

LoginPresenter

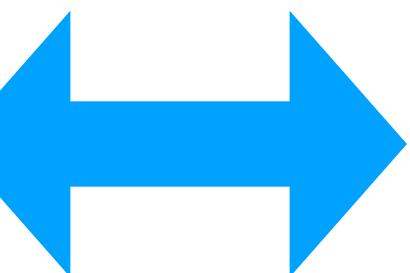
```
package org.wit.placemark.views.login

import org.wit.placemark.views.BasePresenter
import org.wit.placemark.views.BaseView
import org.wit.placemark.views.VIEW

class LoginPresenter(view: BaseView) : BasePresenter(view) {

    fun doLogin(email: String, password: String) {
        view?.navigateTo(VIEW.LIST)
    }

    fun doSignUp(email: String, password: String) {
        view?.navigateTo(VIEW.LIST)
    }
}
```



LoginView

```
package org.wit.placemark.views.login

import android.os.Bundle
import kotlinx.android.synthetic.main.activity_login.*
import org.jetbrains.anko.toast
import org.wit.placemark.R
import org.wit.placemark.views.BaseView

class LoginView : BaseView() {

    lateinit var presenter: LoginPresenter

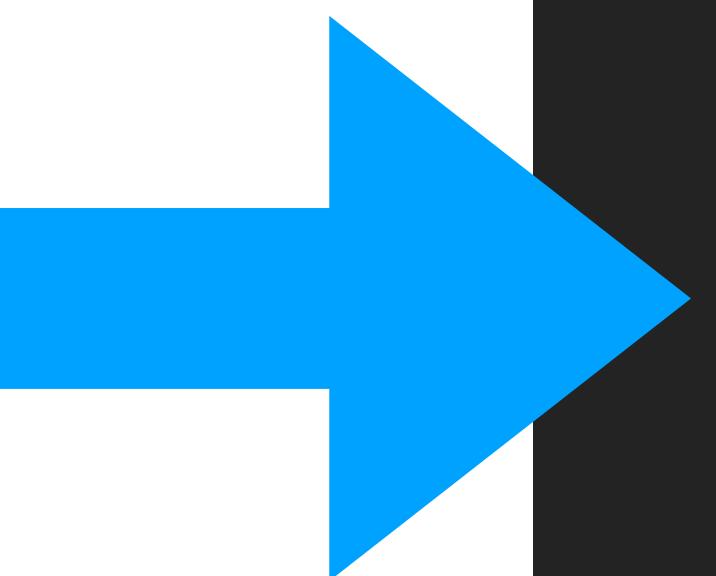
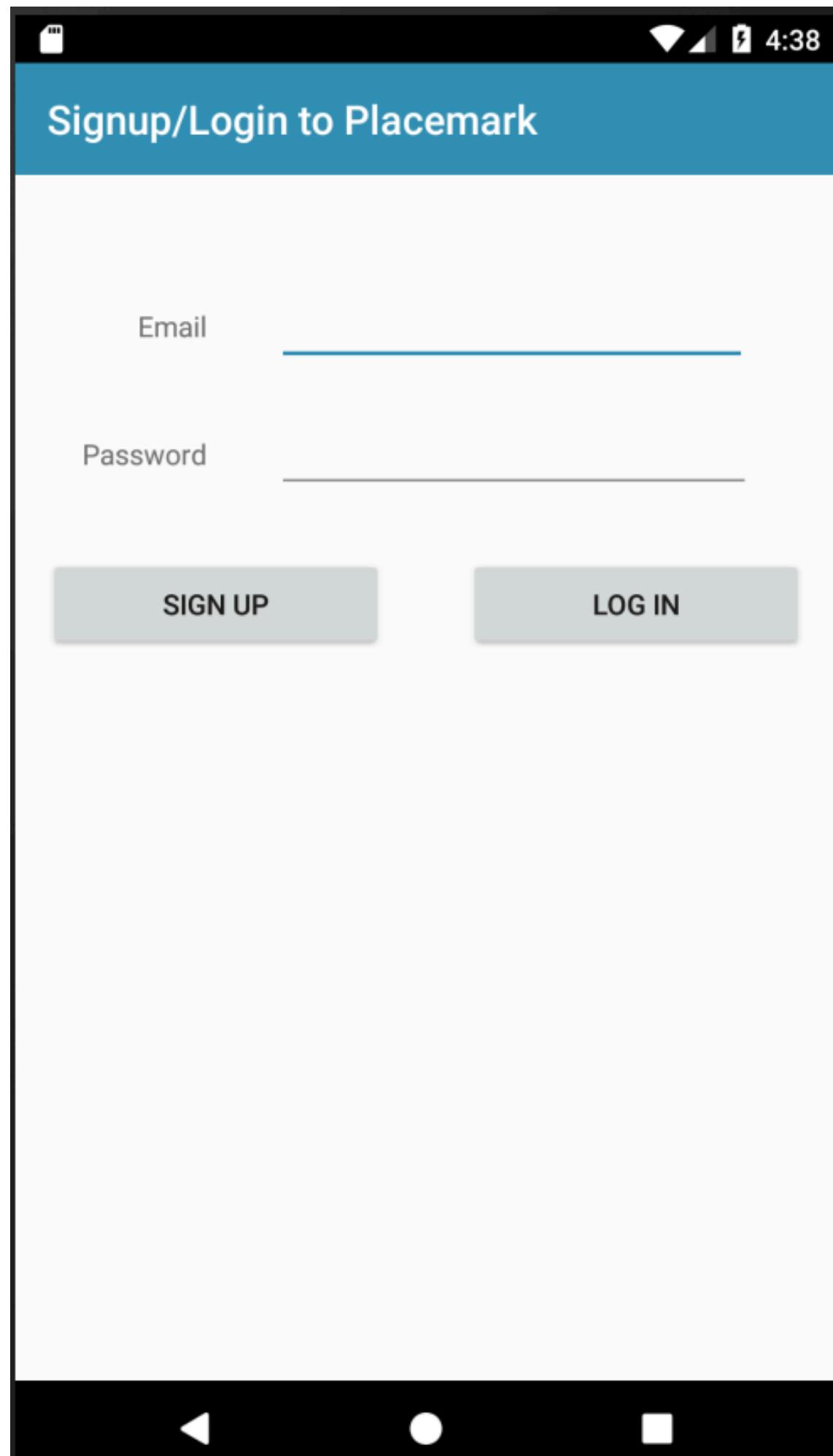
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)
        init(toolbar, false)

        presenter = initPresenter(LoginPresenter(this)) as LoginPresenter

        signUp.setOnClickListener {
            val email = email.text.toString()
            val password = password.text.toString()
            if (email == "" || password == "") {
                toast("Please provide email + password")
            } else {
                presenter.doSignUp(email, password)
            }
        }

        logIn.setOnClickListener {
            val email = email.text.toString()
            val password = password.text.toString()
            if (email == "" || password == "") {
                toast("Please provide email + password")
            } else {
                presenter.doLogin(email, password)
            }
        }
    }
}
```

AndroidManifest.xml



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.wit.placemark">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:name=".main.MainApp"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".views.login.LoginView"
            android:label="@string/title_activity_login"
            android:launchMode="singleTop">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name=".views.placemarklist.PlacemarkListView"
            android:label="@string/title_activity_placemark_list"
            android:launchMode="singleTop">
        </activity>
    
```

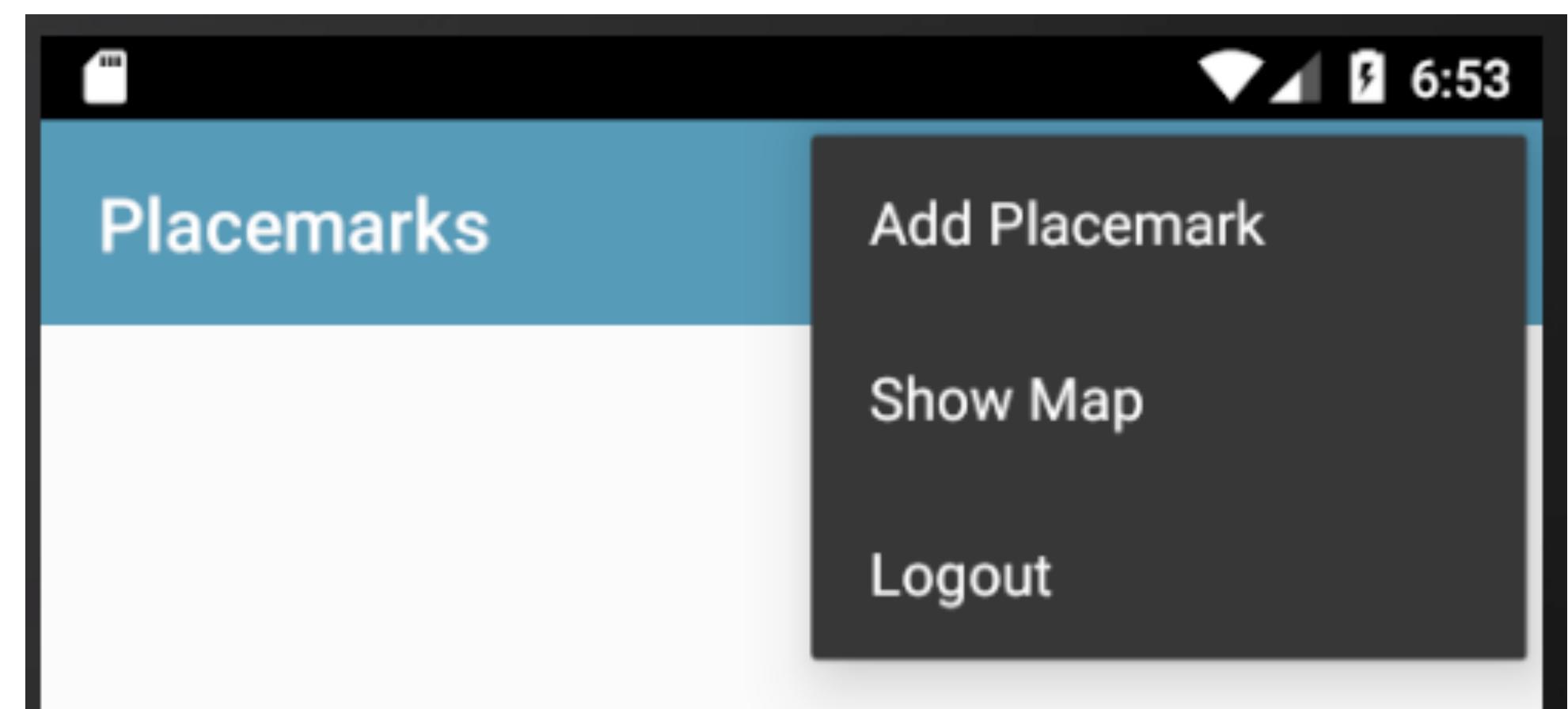
memu_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/item_add"
        android:icon="@android:drawable/ic_menu_add"
        android:title="@string/menu_addPlacemark"
        app:showAsAction="never"/>

    <item
        android:id="@+id/item_map"
        android:icon="@android:drawable/ic_menu_mapmode"
        android:title="@string/menu_showMap"
        app:showAsAction="never"/>

    <item
        android:id="@+id/item_logout"
        android:title="@string/menu_logout"
        android:visible="true"
        app:showAsAction="never" />
</menu>
```



Logout Menu Option

BaseView

```
...
import org.wit.placemark.views.login.LoginView
...
enum class VIEW {
    LOCATION, PLACEMARK, MAPS, LIST, LOGIN
}
...

fun navigateTo(view: VIEW, code: Int = 0, key: String = "", value: Parcelable? = null) {
    var intent = Intent(this, PlacemarkListView::class.java)
    when (view) {
        VIEW.LOCATION -> intent = Intent(this, EditLocationView::class.java)
        VIEW.PLACEMARK -> intent = Intent(this, PlacemarkView::class.java)
        VIEW.MAPS -> intent = Intent(this, PlacemarkMapView::class.java)
        VIEW.LIST -> intent = Intent(this, PlacemarkListView::class.java)
        VIEW.LOGIN -> intent = Intent(this, LoginView::class.java)
    }
    if (key != "") {
        intent.putExtra(key, value)
    }
    startActivityForResult(intent, code)
}
```

PlacemarkListView

```
override fun onOptionsItemSelected(item: MenuItem?): Boolean {
    when (item?.itemId) {
        R.id.item_add -> presenter.doAddPlacemark()
        R.id.item_map -> presenter.doShowPlacemarksMap()
        R.id.item_logout ->presenter.doLogout()
    }
    return super.onOptionsItemSelected(item)
}
```

PlacemarkListPresenter

```
fun doLogout() {
    view?.navigateTo(VIEW.LOGIN)
}
```

Logout Action

Signup to Firebase - Create Project

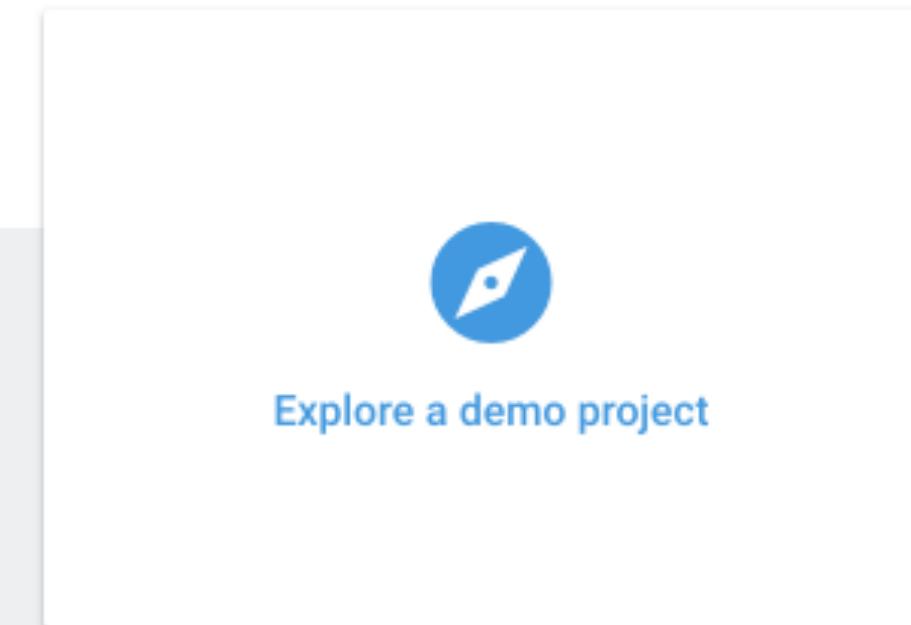
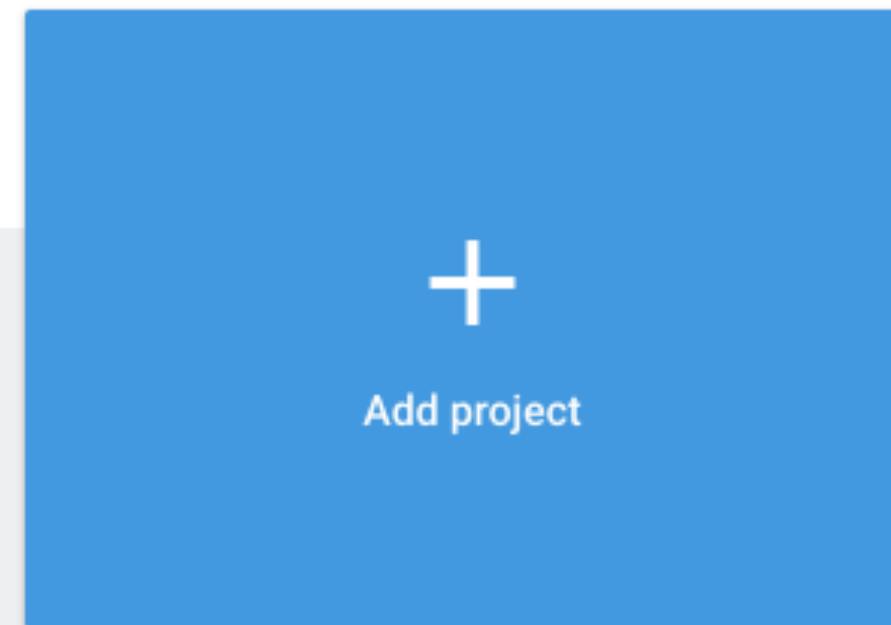


Welcome to Firebase!

Tools from Google for developing great apps, engaging with your users, and earning more through mobile ads.

[Learn more](#) [Documentation](#) [Support](#)

Your projects using Firebase



Your projects currently at Firebase.com

Don't see your projects from Firebase.com? Click [here](#) to link your account.

through mobile ads.

Add a project

Project name

placemark-demo

Tip: Projects span apps across platforms [?](#)

Project ID [?](#)

placemark-demo [edit](#)

Locations [?](#)

United States (Analytics) [edit](#)
us-central (Cloud Firestore) [edit](#)

Use the default settings for sharing Google Analytics for Firebase data

- ✓ Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable technical support
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google Account Specialists

I accept the [controller-controller terms](#). This is required when sharing Analytics data to improve Google Products and Services. [Learn more](#)

[Cancel](#) [Create project](#)

Firebase Console

Firebase

Project Overview | [Settings](#)

placemark-demo ▾ placemark-demo Spark plan

Get started by adding Firebase to your app

Our core SDK unlocks most Firebase features and includes Analytics for iOS and Android apps

iOS Android </>

Add an app to get started

Store and sync app data in milliseconds

Auth
Authenticate and manage users

Database
Store and sync data in realtime

[See all Develop features](#)

Spark Free \$0/month [Upgrade](#)

Authentication Console

Firebase

Project Overview |

DEVELOP

- Authentication
- Database
- Storage
- Hosting
- Functions

STABILITY

Crashlytics, Performance, Test Lab

ANALYTICS

Dashboard, Events, Audiences, Attrib...

GROW

Predictions, Cloud Messaging, Remo...

placemark ▾

Authentication

WEB SETUP

USERS SIGN-IN METHOD TEMPLATES USAGE

Search by email address, phone number, or user UID ADD USER

Identifier	Providers	Created	Signed In	User UID ↑
------------	-----------	---------	-----------	------------

 Authenticate and manage users from a variety of providers without server-side code

[Learn more](#) [View the docs](#)

[SET UP SIGN-IN METHOD](#)

Need help setting up your app? </>

Authentication

WEB SETUP 

USERS SIGN-IN METHOD TEMPLATES USAGE

Sign-in providers

Provider

Status

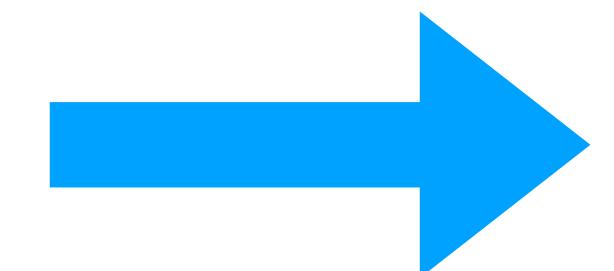
 Email/Password

Enable 

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. [Learn more](#) 

Email link (passwordless sign-in)

Enable 



Sign-in Method -> Email/Password

Authentication

WEB SETUP 

USERS

SIGN-IN METHOD

TEMPLATES

USAGE

Sign-in providers

Provider

Status

 Email/Password

Enabled

 Phone

Disabled

 Google

Disabled

 Play Games

Disabled

 Facebook

Disabled

 Twitter

Disabled

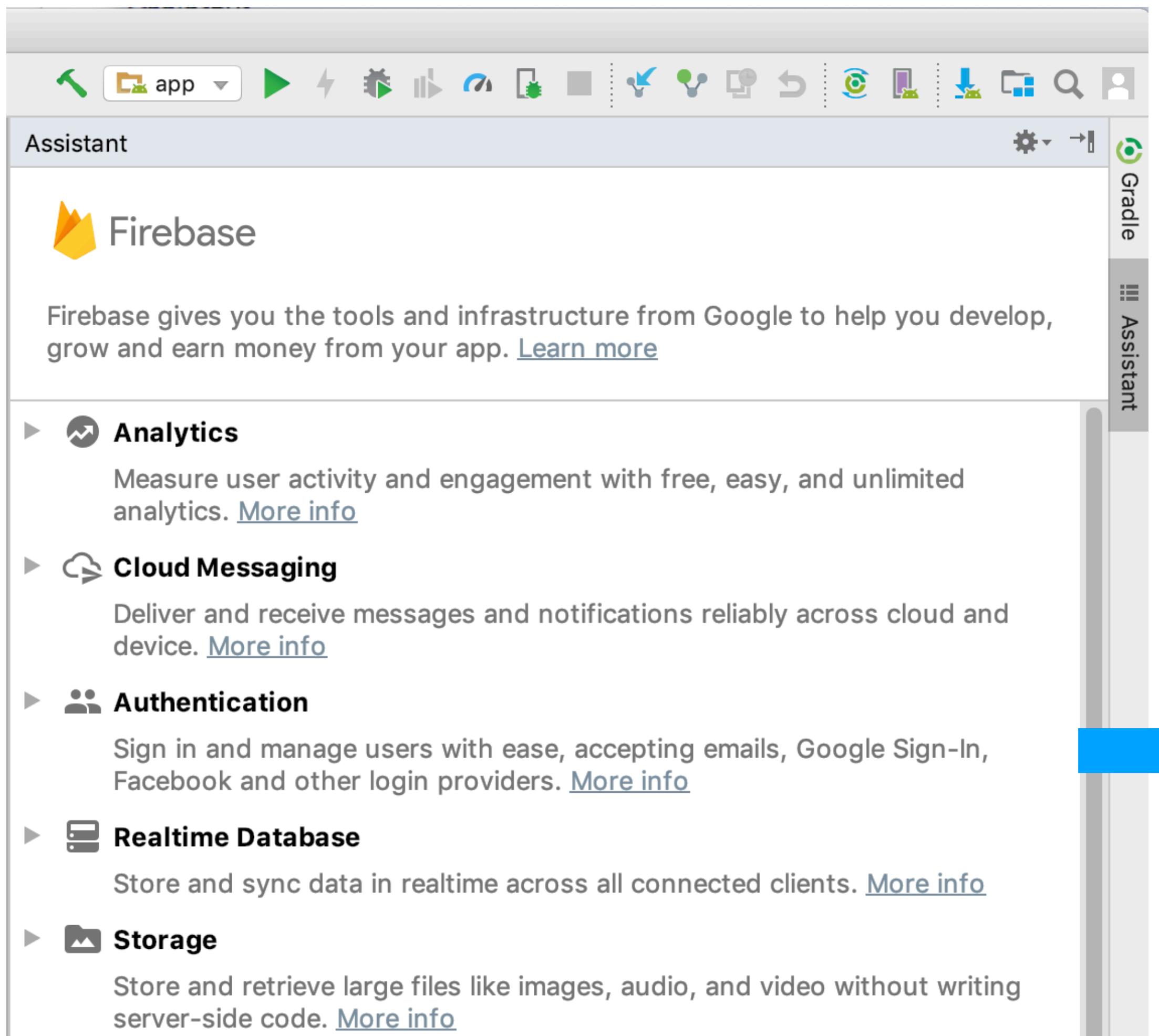
 GitHub

Disabled

 Anonymous

Disabled

Android Studio Firebase Assistant



Assistant
Firebase > Authentication

Email and password authentication

You can use Firebase Authentication to let your users sign in with their email addresses and passwords, and to manage your app's password-based accounts. This tutorial helps you set up an email and password system and then access information about the user.

[Launch in browser](#)

- 1 Connect your app to Firebase**
[Connect to Firebase](#)
- 2 Add Firebase Authentication to your app**
[Add Firebase Authentication to your app](#)

To use an authentication provider, you need to enable it in the [Firebase console](#). Go to the Sign-in Method page in the Firebase Authentication section to enable Email/Password sign-in and any other identity providers you want for your app.
- 3 Check current auth state**

Declare an instance of `FirebaseAuth`

```
private FirebaseAuth mAuth;
```

In the `onCreate()` method, initialize the `FirebaseAuth` instance.

```
mAuth = FirebaseAuth.getInstance();
```

When initializing your Activity, check to see if the user is currently signed in.

```
@Override
public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update UI accordingly!
    FirebaseUser currentUser = mAuth.getCurrentUser();
    updateUI(currentUser);
}
```

Assistant

← Firebase > Authentication

Email and password authentication

You can use Firebase Authentication to let your users sign in with their email addresses and passwords, and to manage your app's password-based accounts. This tutorial helps you set up an email and password system and then access information about the user.

[Launch in browser](#)

- 1 Connect your app to Firebase**
[Connect to Firebase](#)
- 2 Add Firebase Authentication to your app**
[Add Firebase Authentication to your app](#)

To use an authentication provider, you need to enable it in the [Firebase console](#). Go to the Sign-in Method page in the Firebase Authentication section to enable Email/Password sign-in and any other identity providers you want for your app.
- 3 Check current auth state**
Declare an instance of `FirebaseAuth`

```
private FirebaseAuth mAuth;
```

In the `onCreate()` method, initialize the `FirebaseAuth` instance.

```
mAuth = FirebaseAuth.getInstance();
```

When initializing your Activity, check to see if the user is currently signed in.

```
@Override  
public void onStart() {  
    super.onStart();  
    // Check if user is signed in (non-null) and update UI accordingly.  
    FirebaseUser currentUser = mAuth.getCurrentUser();  
    updateUI(currentUser);  
}
```

Connect Project to Firebase App

Connect to Firebase

 Firebase

Create new Firebase project [What's this?](#) Signed in as [edeleastar@gmail.com](#) [Sign out](#)

Choose an existing Firebase or Google project

placemark-origin

placemark

[What's this?](#)

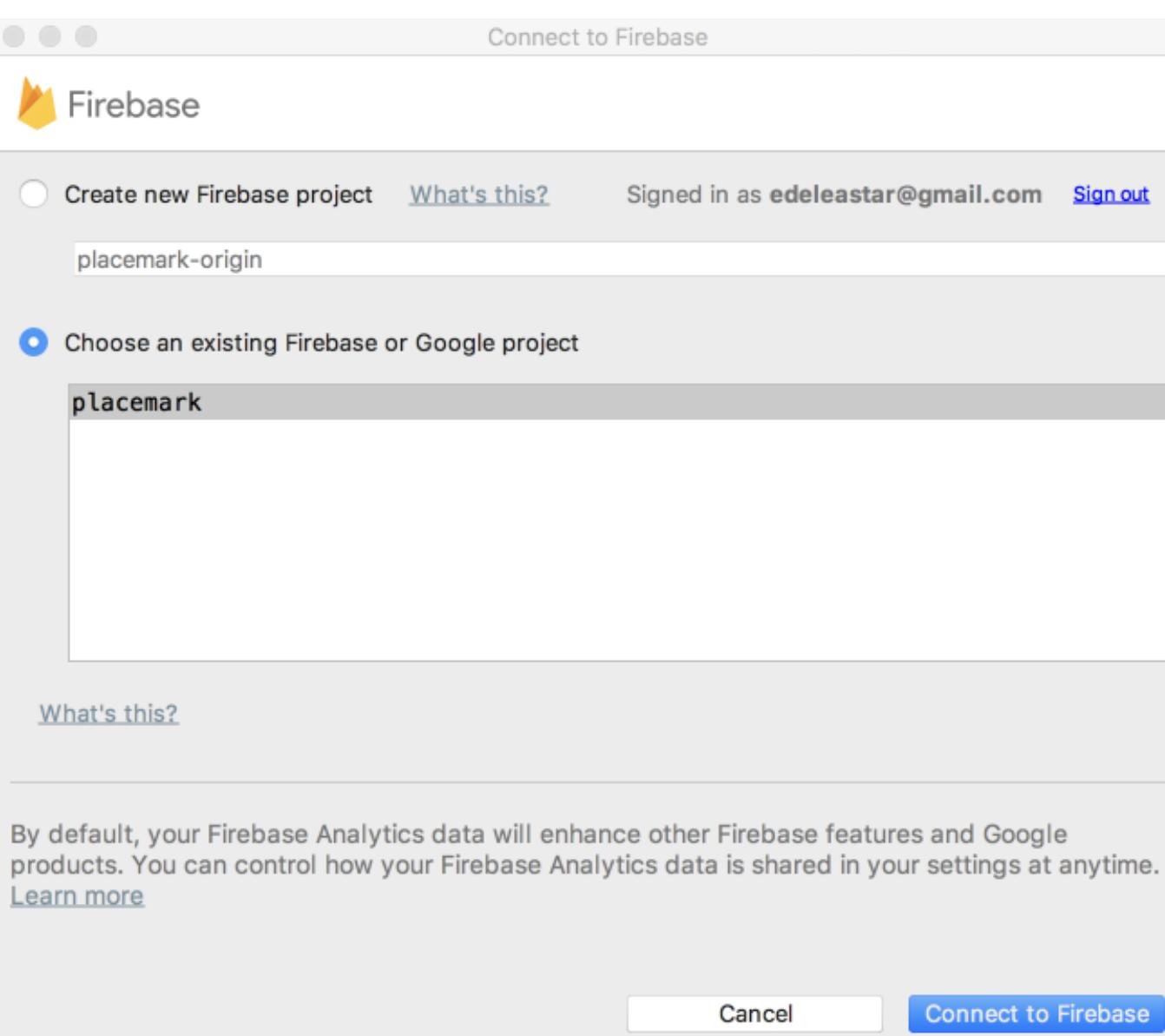
By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime. [Learn more](#)

[Cancel](#) [Connect to Firebase](#)

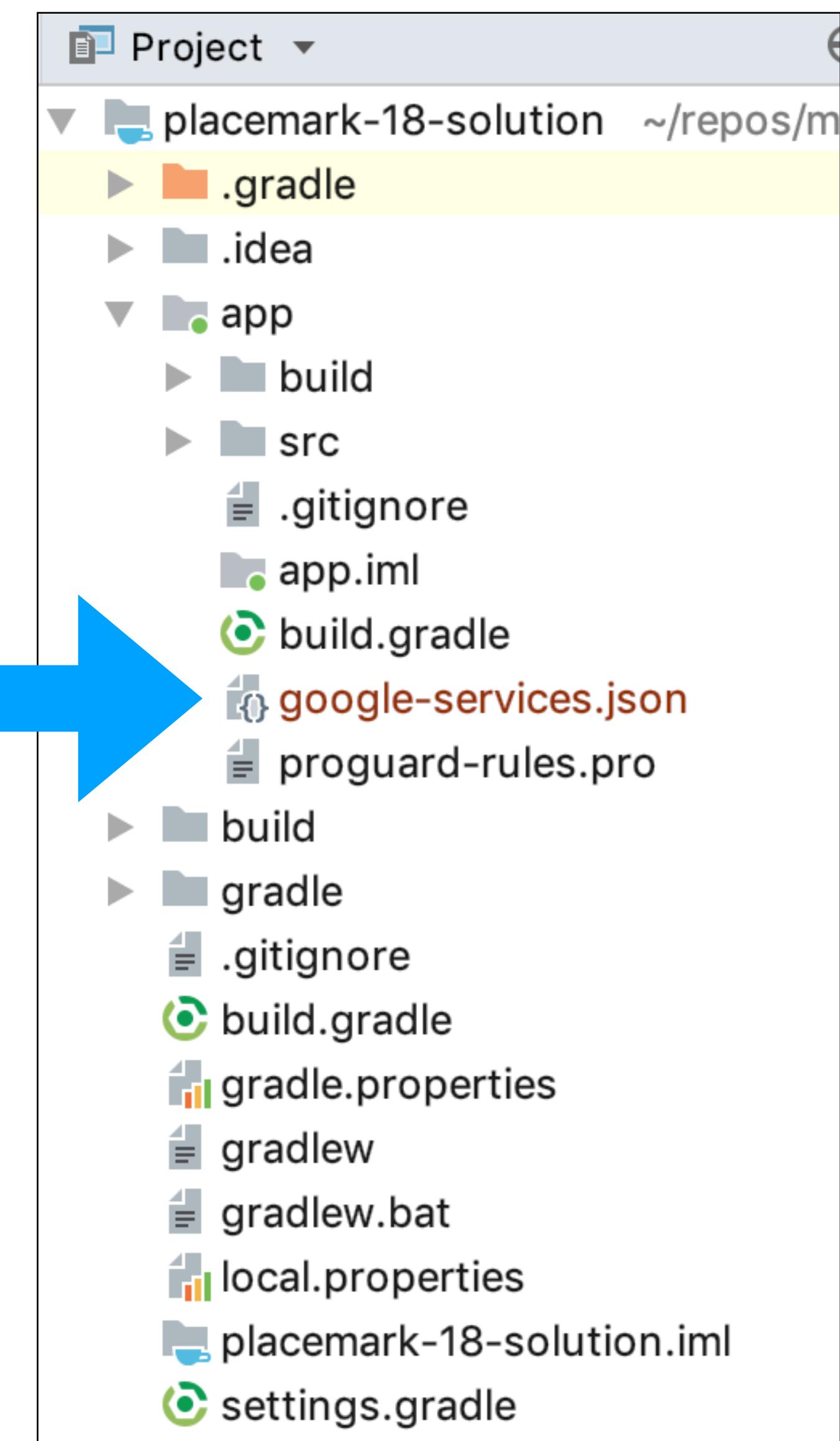
```
...
apply plugin: 'com.google.gms.google-services'
...
    firebase_version = '16.0.5'
    ...
implementation "com.google.firebaseio:firebase-auth:$firebase_version"
implementation "com.google.firebaseio:firebase-database:$firebase_version"
...
```

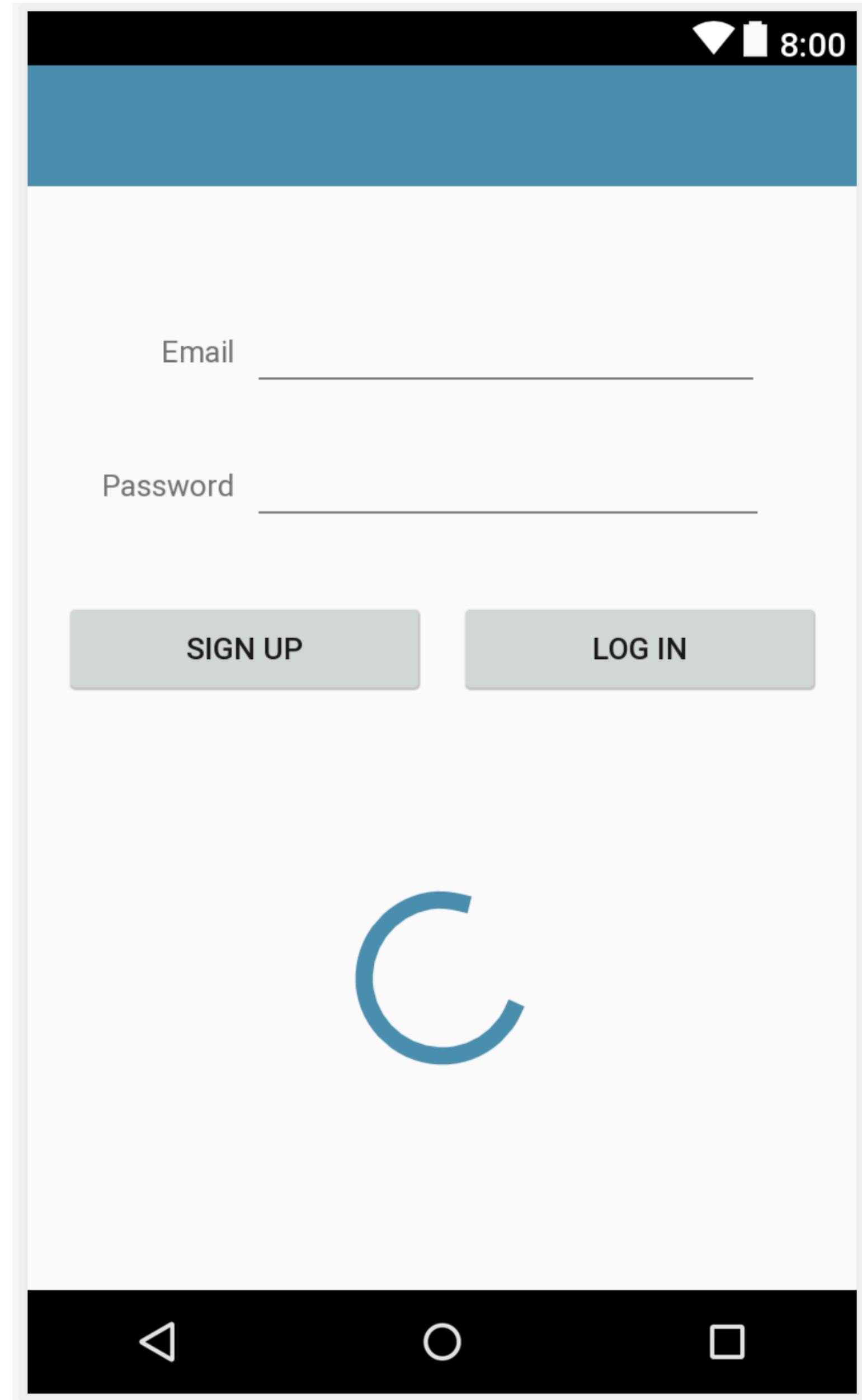
build.gradle

google-service.json



```
{  
  "project_info": {  
    "project_number": "1062442537261",  
    "firebase_url": "https://placemark-222108.firebaseio.com",  
    "project_id": "placemark-222108",  
    "storage_bucket": "placemark-222108.appspot.com"  
  },  
  "client": [  
    {  
      "client_info": {  
        "mobilesdk_app_id": "1:106XXXXXXXXX61:android:634c4d908a4ce143",  
        "android_client_info": {  
          "package_name": "org.wit.placemark"  
        }  
      },  
      "oauth_client": [  
        {  
          "client_id": "1062442537261-elh1gh5mnvh8h3uqo1q320l17eu0pdv.apps.googleusercontent.com",  
          "client_type": 1,  
          "android_info": {  
            "package_name": "org.wit.placemark",  
            "certificate_hash": "368eadXXXXXXXXXXXXXXdc4f2123a7ed96"  
          }  
        },  
        {  
          "client_id": "1062442537261-u7qr2nfjbgvvq3gtvq20avhhp4l5eqc4.apps.googleusercontent.com",  
          "client_type": 3  
        }  
      ],  
      "api_key": [  
        {  
          "current_key": "AIzaSyBtXXXXXXXXXXXXXX52I95o"  
        }  
      ],  
      "services": {  
        "analytics_service": {  
          "status": 1  
        },  
        "appinvite_service": {  
          "status": 2,  
          "other_platform_oauth_client": [  
            {  
              "client_id": "10624425XXXXXXXXXXXXXhp4l5eqc4.apps.googleusercontent.com",  
              "client_type": 3  
            }  
          ]  
        },  
        "ads_service": {  
          "status": 2  
        }  
      }  
    },  
    "configuration_version": "1"  
  }  
}
```





ProgressBar

activity_login.xml

```
...<ProgressBar  
    android:id="@+id/progressBar"  
    style="@style/Widget.AppCompat.ProgressBar"  
    android:layout_width="196dp"  
    android:layout_height="96dp"  
    android:layout_marginStart="8dp"  
    android:layout_marginTop="8dp"  
    android:layout_marginEnd="8dp"  
    android:layout_marginBottom="8dp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toBottomOf="@+id/signUp" />  
...
```

LoginView

```
override fun showProgress() {  
    progressBar.visibility = View.VISIBLE  
}  
  
override fun hideProgress() {  
    progressBar.visibility = View.GONE  
}
```

Revised LoginPresenter

```
class LoginPresenter(view: BaseView) : BasePresenter(view) {

    var auth: FirebaseAuth = FirebaseAuth.getInstance()

    fun doLogin(email: String, password: String) {
        view?.showProgress()
        auth.signInWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
            if (task.isSuccessful) {
                view?.navigateTo(VIEW.LIST)
            } else {
                view?.toast("Sign Up Failed: ${task.exception?.message}")
            }
            view?.hideProgress()
        }
    }

    fun doSignUp(email: String, password: String) {
        view?.showProgress()
        auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
            if (task.isSuccessful) {
                view?.navigateTo(VIEW.LIST)
            } else {
                view?.toast("Sign Up Failed: ${task.exception?.message}")
            }
            view?.hideProgress()
        }
    }
}
```

API: createUserWithEmailAndPassword()

```
var auth: FirebaseAuth = FirebaseAuth.getInstance()

fun doSignUp(email: String, password: String) {
    view?.showProgress()
    auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
        if (task.isSuccessful) {
            view?.navigateTo(VIEW.LIST)
        } else {
            view?.toast("Sign Up Failed: ${task.exception?.message}")
        }
        view?.hideProgress()
    }
}
```

API: createUserWithEmailAndPassword()

```
var auth: FirebaseAuth = FirebaseAuth.getInstance()

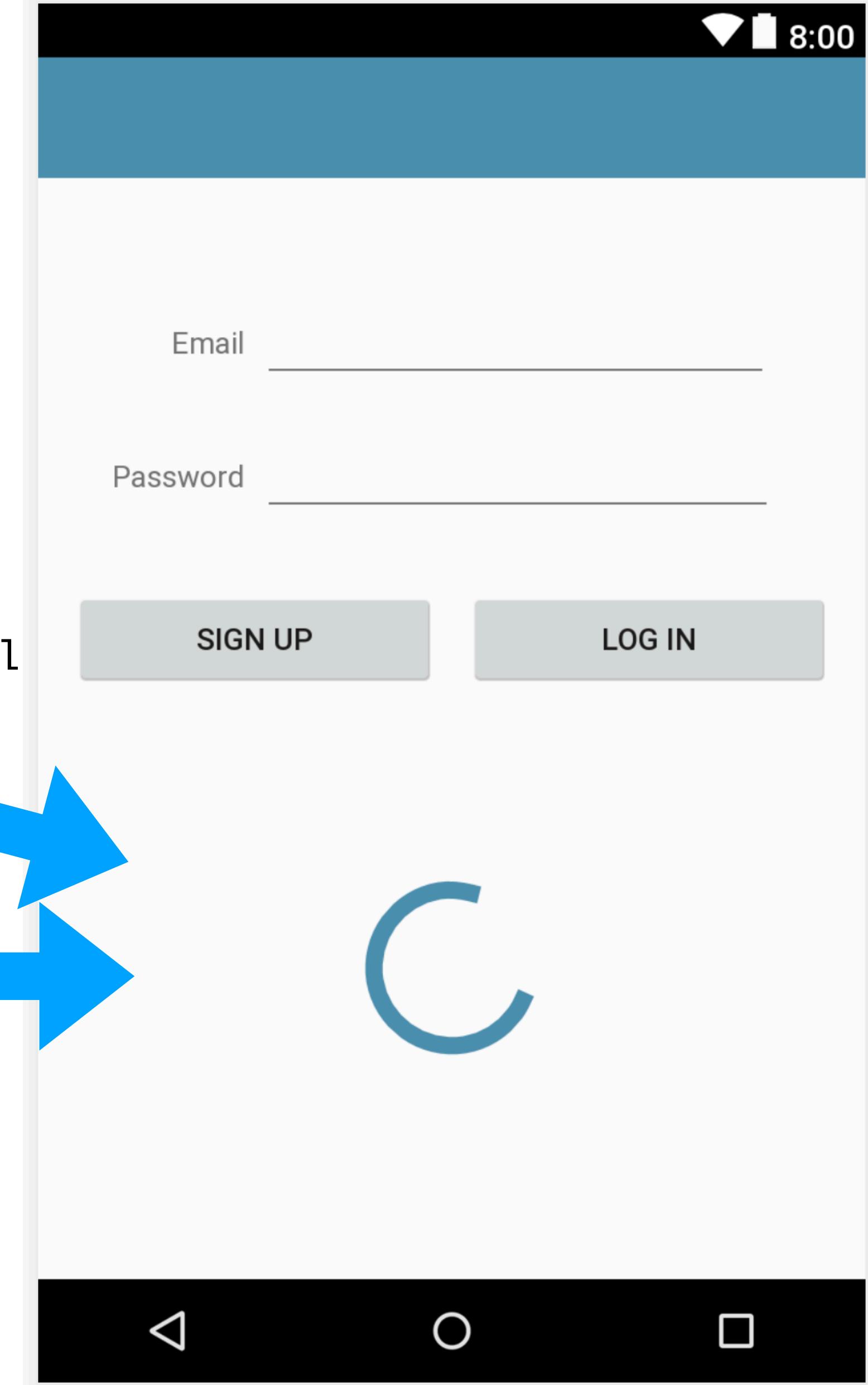
fun doSignUp(email: String, password: String) {
    view?.showProgress()
    auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
        if (task.isSuccessful) {
            view?.navigateTo(VIEW.LIST)
        } else {
            view?.toast("Sign Up Failed: ${task.exception?.message}")
        }
        view?.hideProgress()
    }
}
```

Lambda: task.isSuccessful => signed up and authenticated
Navigate to PlacemarkList

showProgress() / hideProgress()

```
var auth: FirebaseAuth = FirebaseAuth.getInstance()

fun doSignUp(email: String, password: String) {
    view?.showProgress()
    auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener { task ->
        if (task.isSuccessful) {
            view?.navigateTo(VIEW.LIST)
        } else {
            view?.toast("Sign Up Failed: ${task.exception?.message}")
        }
    view?.hideProgress()
    }
}
```



API: singInWithEmailAndPassword()

```
class LoginPresenter(view: BaseView) : BasePresenter(view) {

    var auth: FirebaseAuth = FirebaseAuth.getInstance()

    fun doLogin(email: String, password: String) {
        view?.showProgress()
        auth.signInWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
            if (task.isSuccessful) {
                view?.navigateTo(VIEW.LIST)
            } else {
                view?.toast("Sign Up Failed: ${task.exception?.message}")
            }
            view?.hideProgress()
        }
    }
    ...
}
```

API: singInWithEmailAndPassword()

```
class LoginPresenter(view: BaseView) : BasePresenter(view) {

    var auth: FirebaseAuth = FirebaseAuth.getInstance()

    fun doLogin(email: String, password: String) {
        view?.showProgress()
        auth.signInWithEmailAndPassword(email, password).addOnCompleteListener(view!!) { task ->
            if (task.isSuccessful) {
                view?.navigateTo(VIEW.LIST)
            } else {
                view?.toast("Sign Up Failed: ${task.exception?.message}")
            }
            view?.hideProgress()
        }
    }
    ...
}
```



Lambda - callback when user authenticated

Android

[Sign in with a pre-built UI](#)

Get Started with the Firebase SDK

Manage Users

Password Authentication

Email Link Authentication

Google Sign-In

Facebook Login

Twitter

GitHub

Play Games Sign-in 

Phone Number

Use a Custom Auth System

Anonymous Authentication

Link Multiple Auth Providers

Passing State in Email Actions

Easily add sign-in to your Android app with FirebaseUI

[FirebaseUI](#) is a library built on top of the Firebase Authentication SDK that provides drop-in UI flows for use in your app. FirebaseUI provides the following benefits:

- **Multiple Providers** - sign-in flows for email, phone authentication, Google Sign-In, Facebook Login, and Twitter Login.
- **Account Management** - flows to handle account management tasks, such as account creation and password resets.
- **Account Linking** - flows to safely link user accounts across identity providers.
- **Custom Themes** - customize the look of FirebaseUI to match your app. Also, because FirebaseUI is open source, you can fork the project and customize it exactly to your needs.
- **Smart Lock for Passwords** - automatic integration with [Smart Lock for Passwords](#) for fast cross-device sign-in.

