Mobile Application Development

Produced by

David Drohan (ddrohan@wit.ie)

Department of Computing & Mathematics
Waterford Institute of Technology
http://www.wit.ie

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Android & Firebase

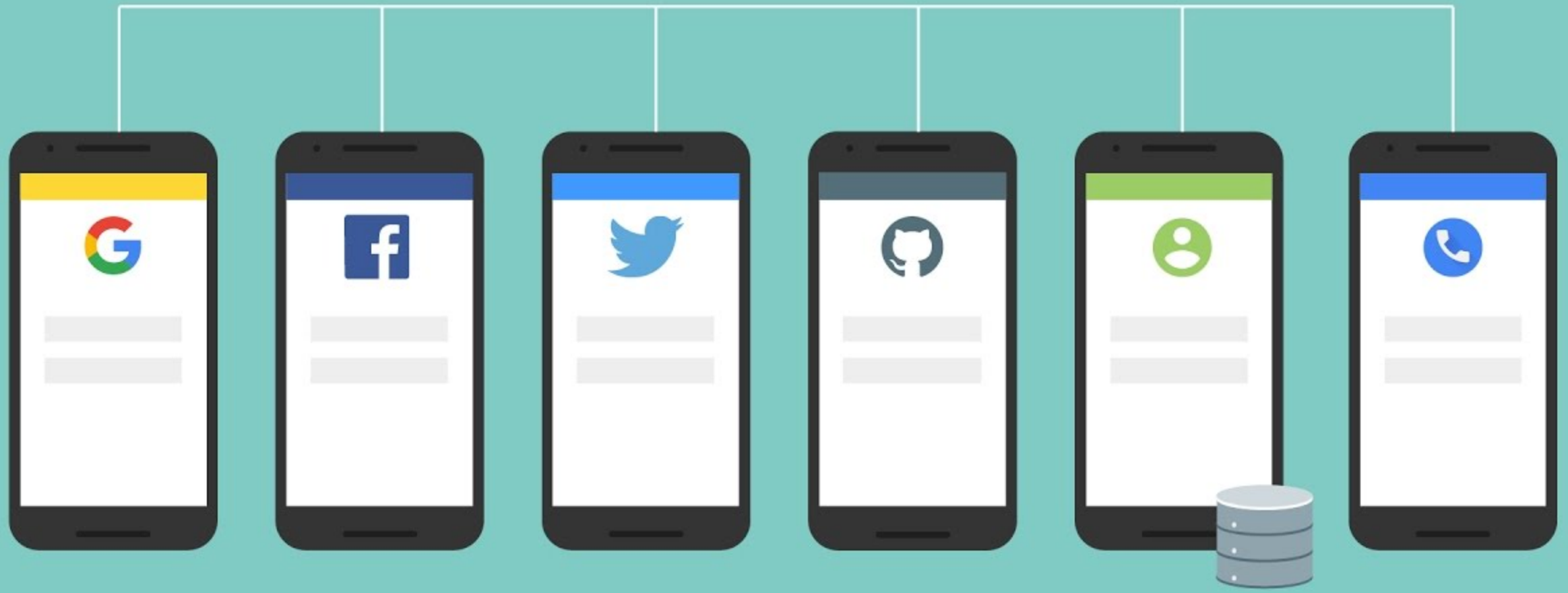
Firebase Integration

Agenda

- Firebase history
- The all new Firebase
- Real-time database
- Firestore
- Authentication
- Storage
- Remote config
- Hosting
- Crash reporting
- Test lab
- Firebase cloud messaging
- Dynamic links
- App indexing
- Analytics
- CoffeeMate Highlights & Demos along the way…

Agenda

- ❑ Firebase history
- ❑ The all new Firebase
- ❑ Real-time database
- ❑ Firestore
- ❑ Authentication
- ❑ Storage
- ❑ Remote config
- ❑ Hosting
- ❑ Crash reporting
- ❑ Test lab
- ❑ Firebase cloud messaging
- ❑ Dynamic links
- ❑ App indexing
- ❑ Analytics
- ❑ CoffeeMate Highlights & Demos along the way…

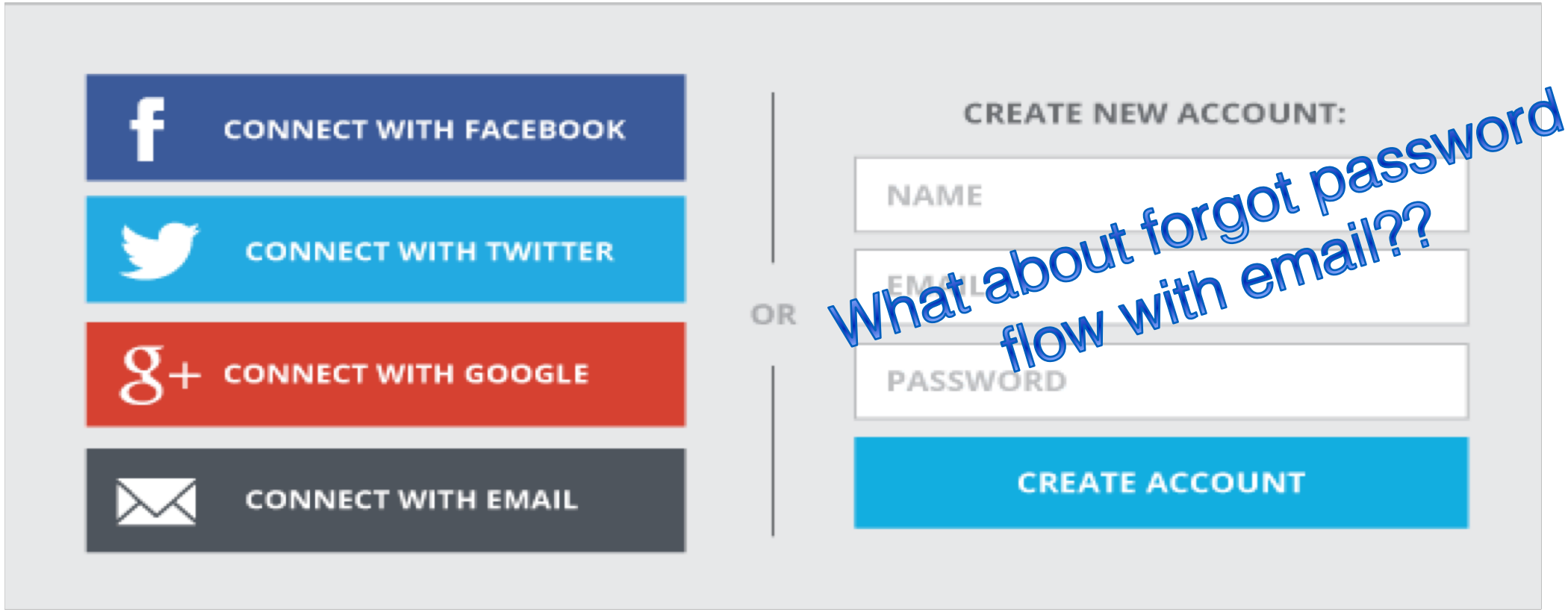


All goodness bundled as one…

Authentication

Firebase Authentication *

A type of screen present in almost all apps these days…



How long would it take you to develop this???

Firebase Authentication

- Integrate easily with popular identity providers like google, twitter, facebook and more

Firebase Authentication

- Of course you will need to register your app with individual service providers.
- Minimal client side handling, integrates seamlessly with firebase
- Ready made ‘forgot password’ flow with customizable email template

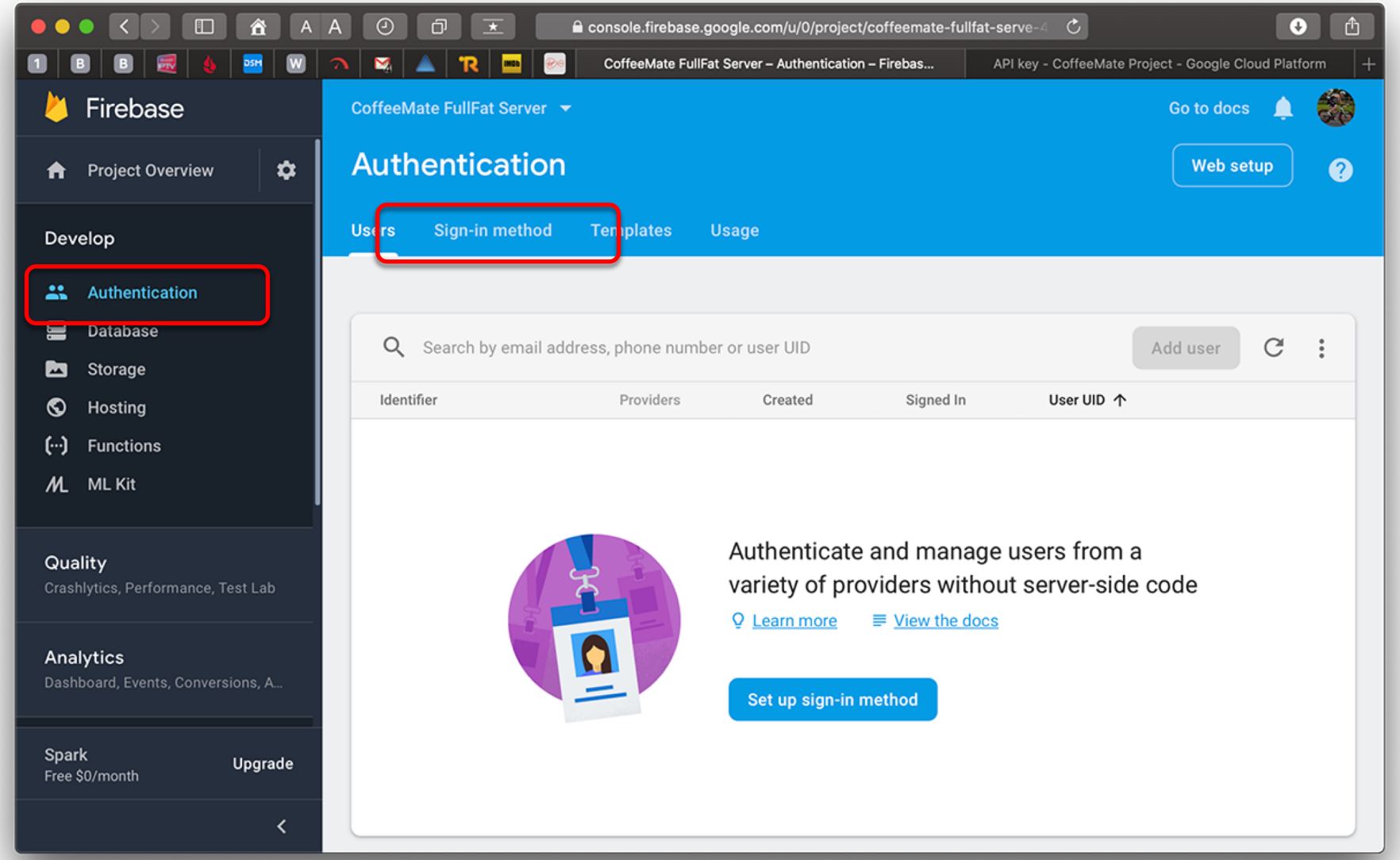
Firebase Authentication

Quick DEMO…

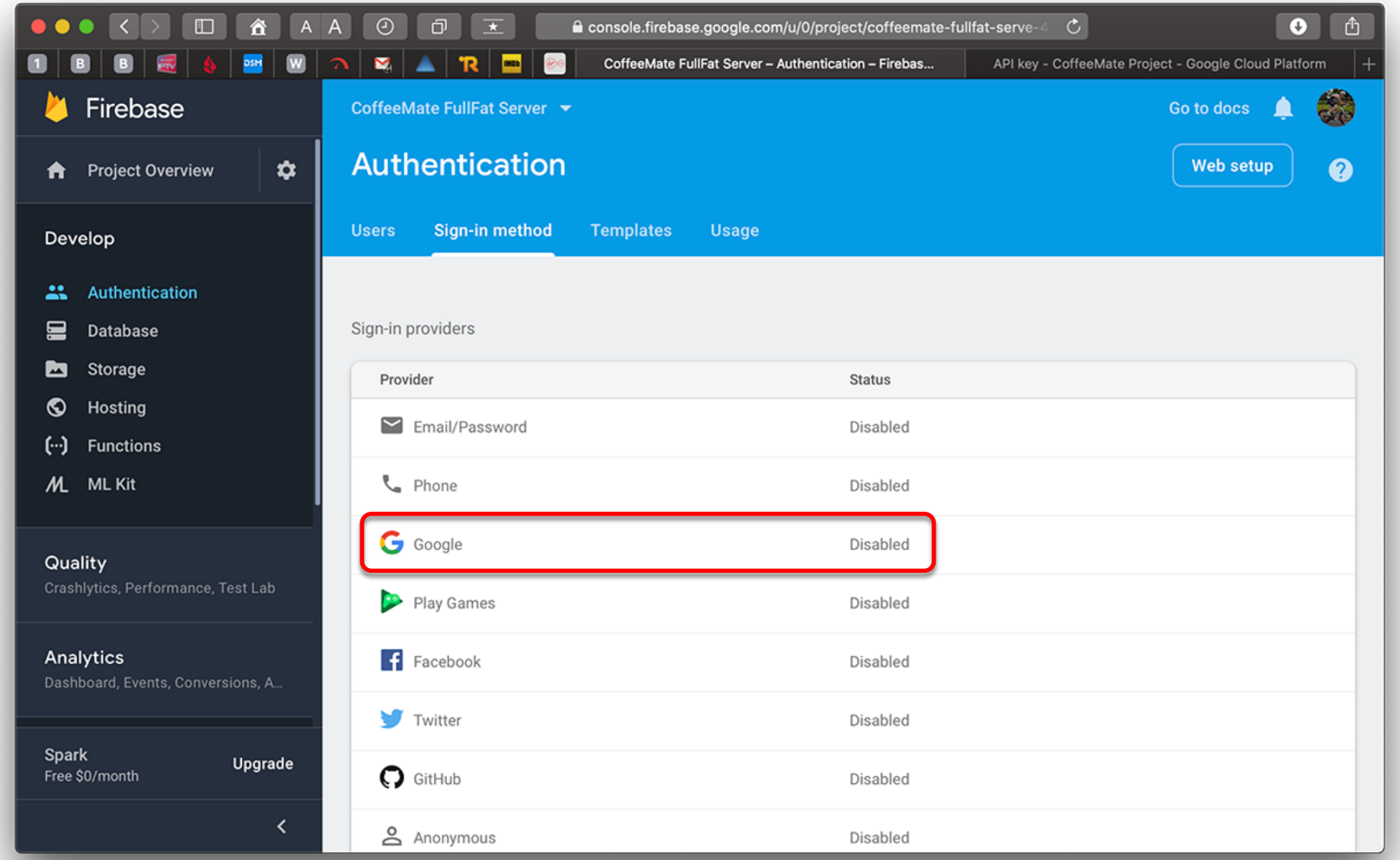
CoffeeMate.9.0

Setup
&
Code
Highlights

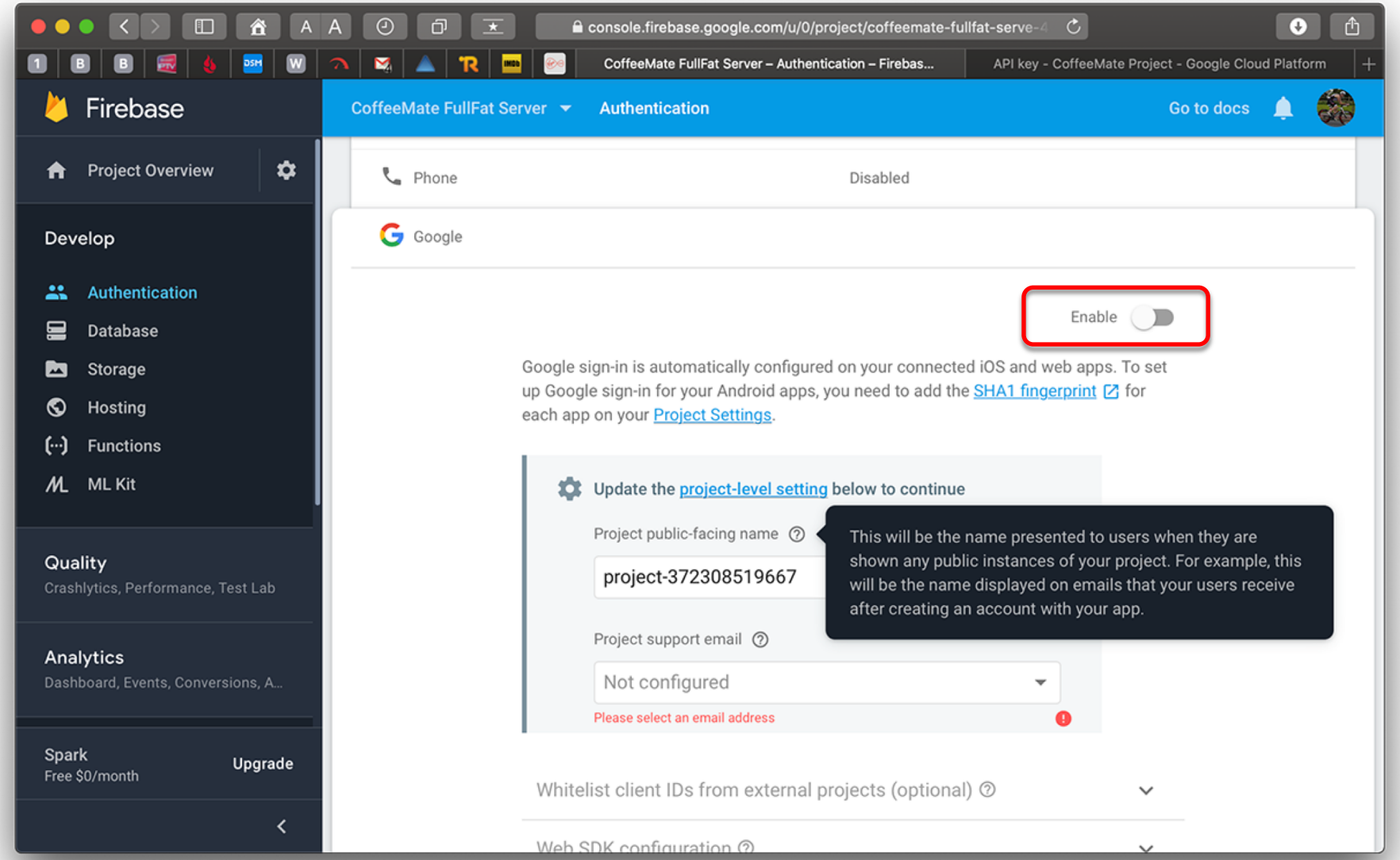
1. Setup your Sign-In Method *



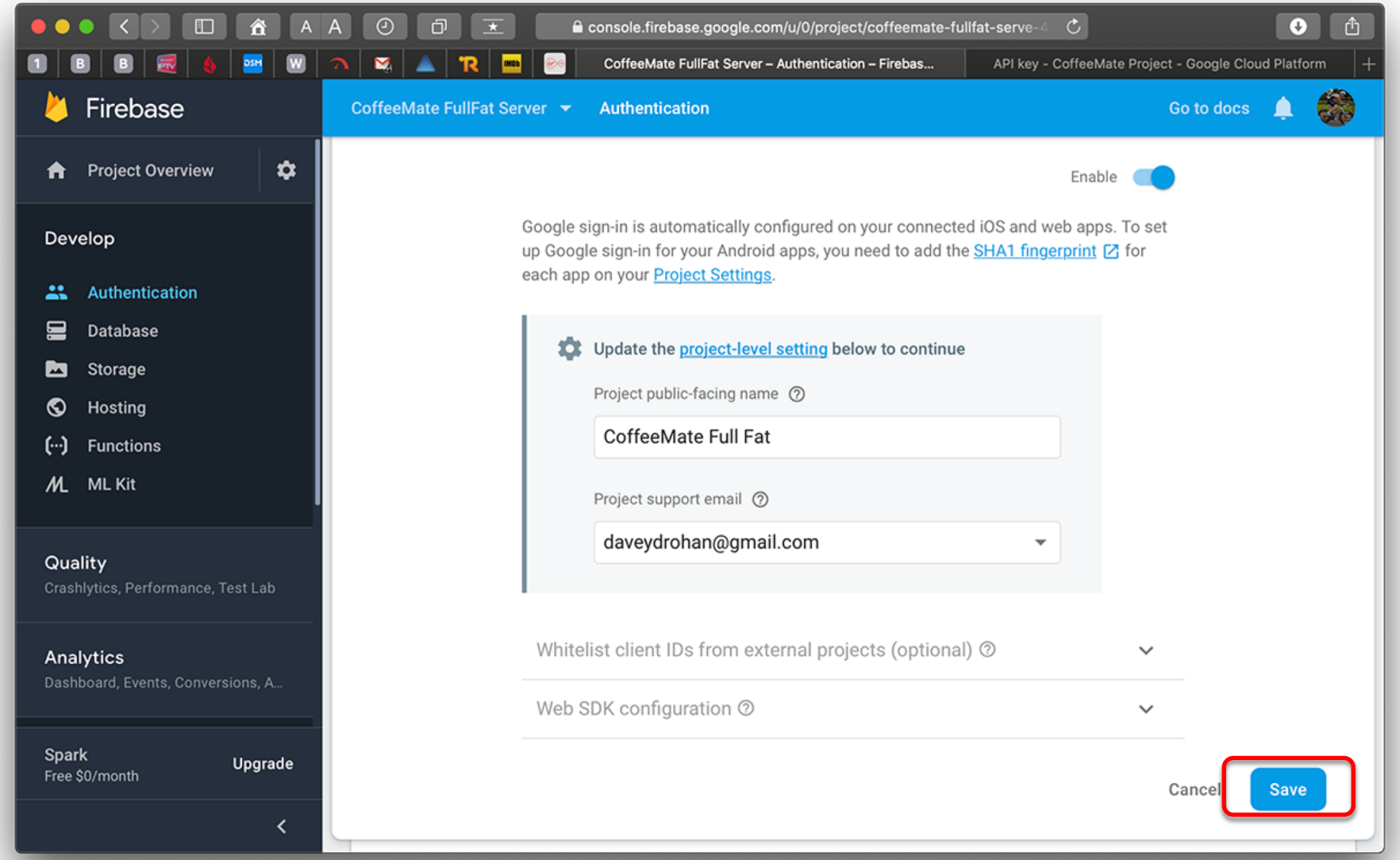
1. Setup your Sign-In Method *



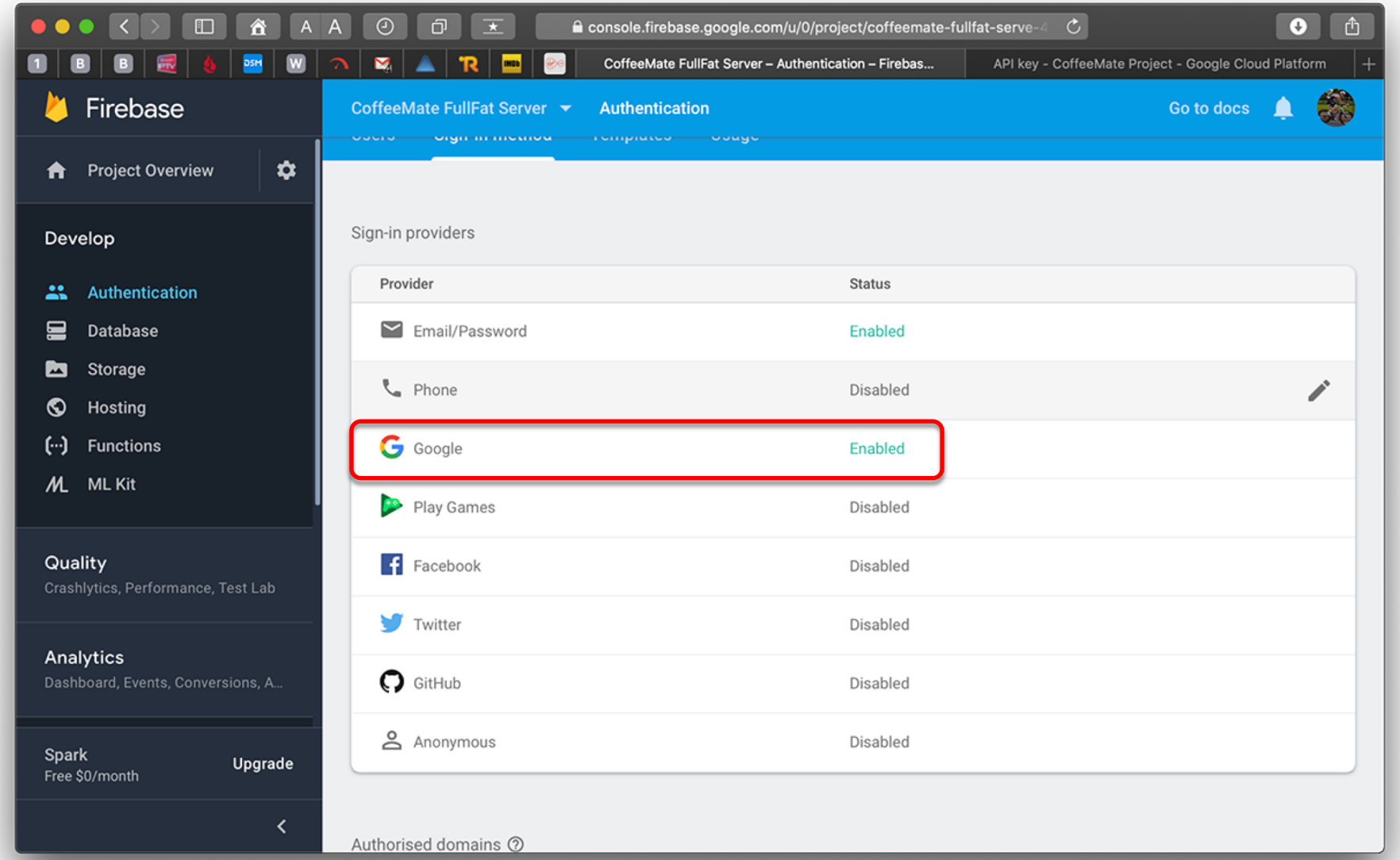
1. Setup your Sign-In Method *



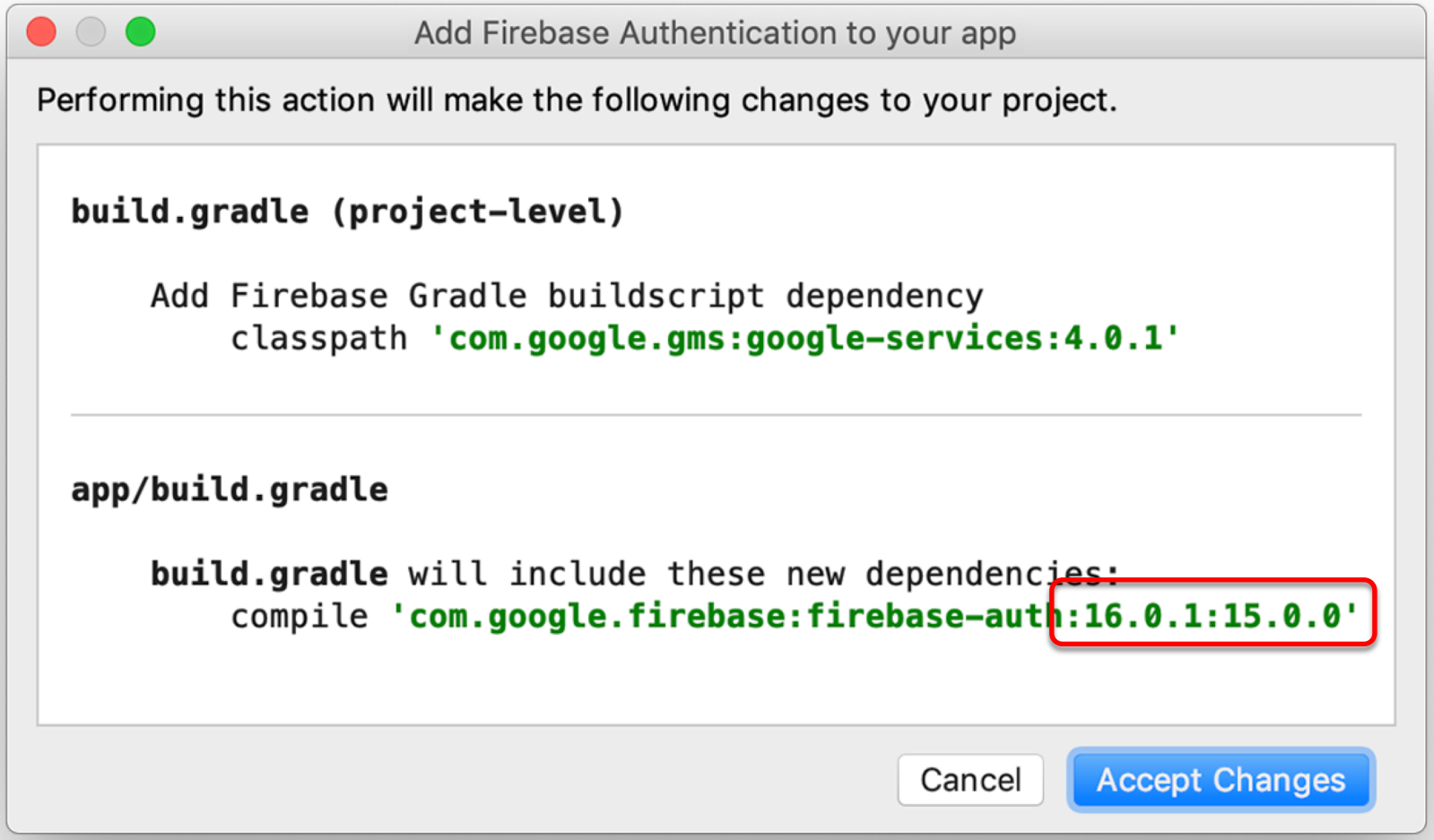
1. Setup your Sign-In Method *



1. Setup your Sign-In Method *



1. Still and issue (as of December 2018) *



1. Still and issue (as of December 2018) *

```
implementation 'com.google.firebase:firebase-core:16.0.1'
implementation 'com.google.firebase:firebase-auth:16.0.1'
```

```
}
dependencies {
    classpath 'com.android.tools.build:gradle:3.2.1'
    classpath 'com.google.gms:google-services:3.2.0'
```

2. Introduce Authentication Flow *

```
private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
    Log.v(TAG, "firebaseAuthWithGoogle:" + acct.getEmail());

    AuthCredential credential = GoogleAuthProvider.getCredential(acct.getIdToken(), null);
    app.mFirebaseAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    Log.v(TAG, "signInWithCredential:onComplete:" + task.isSuccessful());
                    validateFirebaseUser();
                    // If sign in fails, display a message to the user. If sign in succeeds
                    // the auth state listener will be notified and logic to handle the
                    // signed in user can be handled in the listener.
                    if (!task.isSuccessful()) {
                        Log.v(TAG, "signInWithCredential", task.getException());
                        Toast.makeText(Login.this, "Authentication failed.",
                                Toast.LENGTH_SHORT).show();
                    }
                }
            });
}
```

2. Introduce Authentication Flow – Check for User *

```
public void validateFirebaseUser() {
    if(app.mFirebaseUser == null)
        app.mFirebaseUser = FirebaseAuth.getInstance().getCurrentUser();

    Log.v(TAG, "checkUser ID == " + app.mFirebaseUser.getUid());
    app.mFirebaseManager.mFirebaseDatabase.child("users")
            .child(app.mFirebaseUser.getUid()).addListenerForSingleValueEvent(
            new ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {...}

                @Override
                public void onCancelled(DatabaseError databaseError) {...}
            }
    );
}
```

2. Introduce Authentication Flow – Check for User *

```
app.mFirebaseManager.mFirebaseDatabase.child("users")
        .child(app.mFirebaseUser.getUid()).addListenerForSingleValueEvent(
        new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {...}

            @Override
            public void onCancelled(DatabaseError databaseError) {...}
        }
);
```

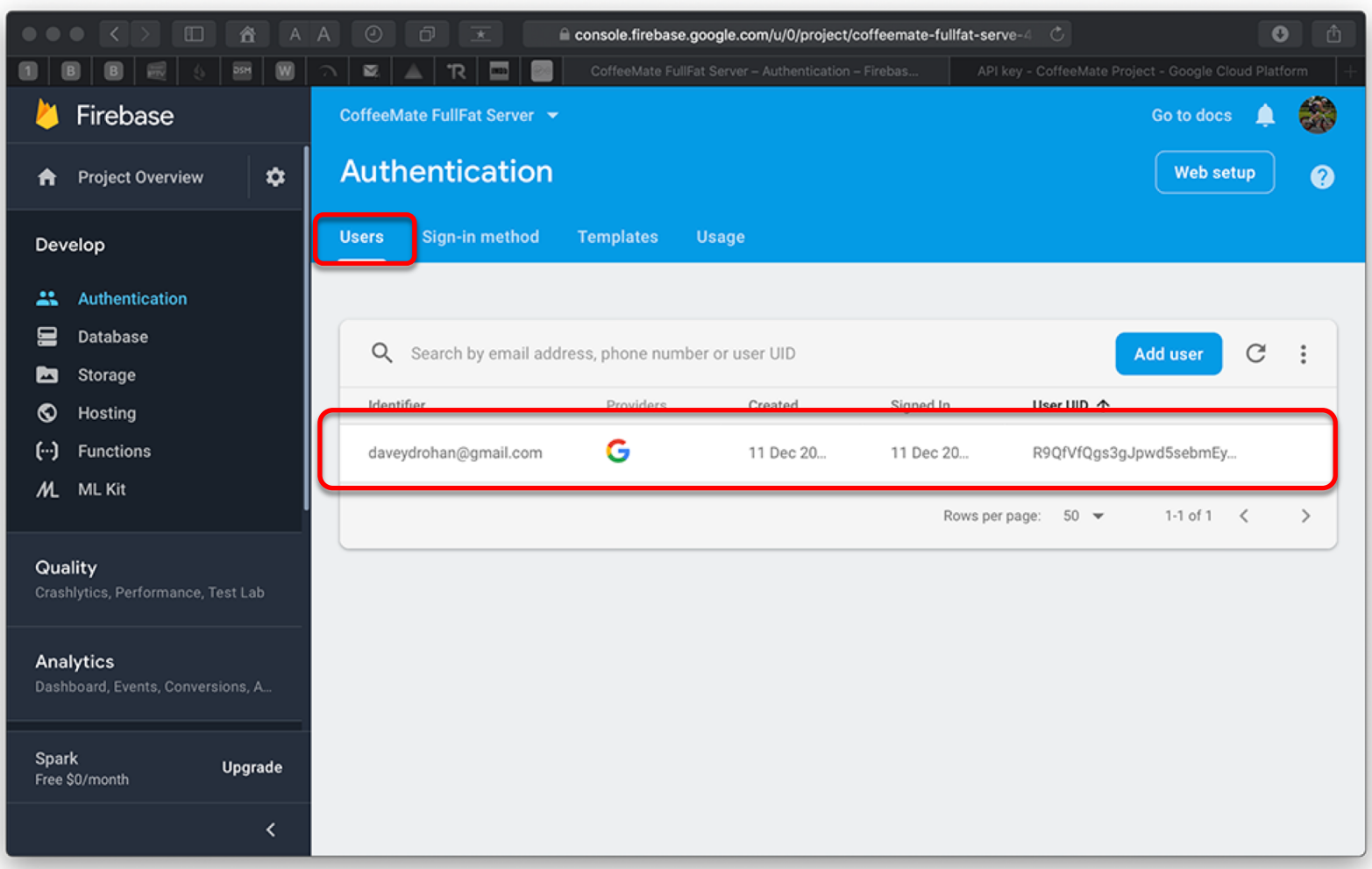
2. Introduce Authentication Flow – Success *

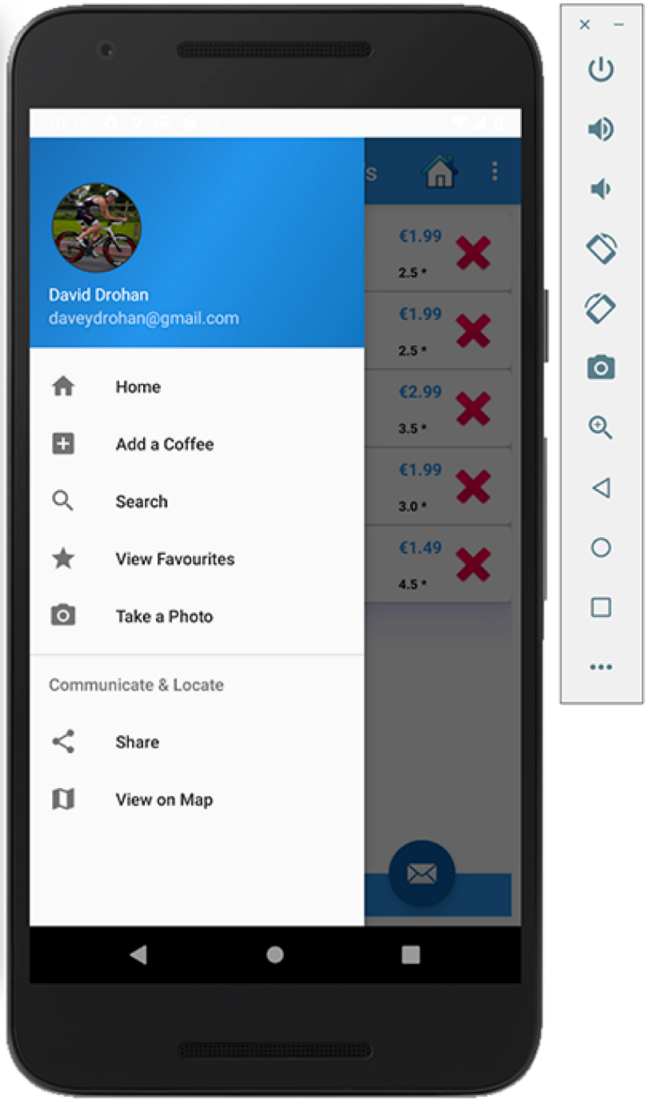
```
app.mFirebaseManager.mFirebaseDatabase.child("users")
        .child(app.mFirebaseUser.getUid()).addListenerForSingleValueEvent(
        new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                Log.v(TAG, "mFirebaseListener.onSuccess " + dataSnapshot);
                if(dataSnapshot.exists()){
                    Log.v(TAG, "User found : " + app.mFirebaseUser.getEmail());
                }
                else{
                    Log.v(TAG, "User not found, Creating User on Firebase");
                    User newUser = new User(app.mFirebaseUser.getUid(),
                            app.mFirebaseUser.getDisplayName(),
                            app.mFirebaseUser.getEmail(), null);
                    app.mFirebaseManager.mFirebaseDatabase.child("users")
                            .child(app.mFirebaseUser.getUid())
                            .setValue(newUser);
                }
                app.mFirebaseManager.mFirebaseUserId = app.mFirebaseUser.getUid();
                startHomeScreen();
            }

            @Override
            public void onCancelled(DatabaseError databaseError) { }
```

2. Introduce Authentication Flow – Failure *

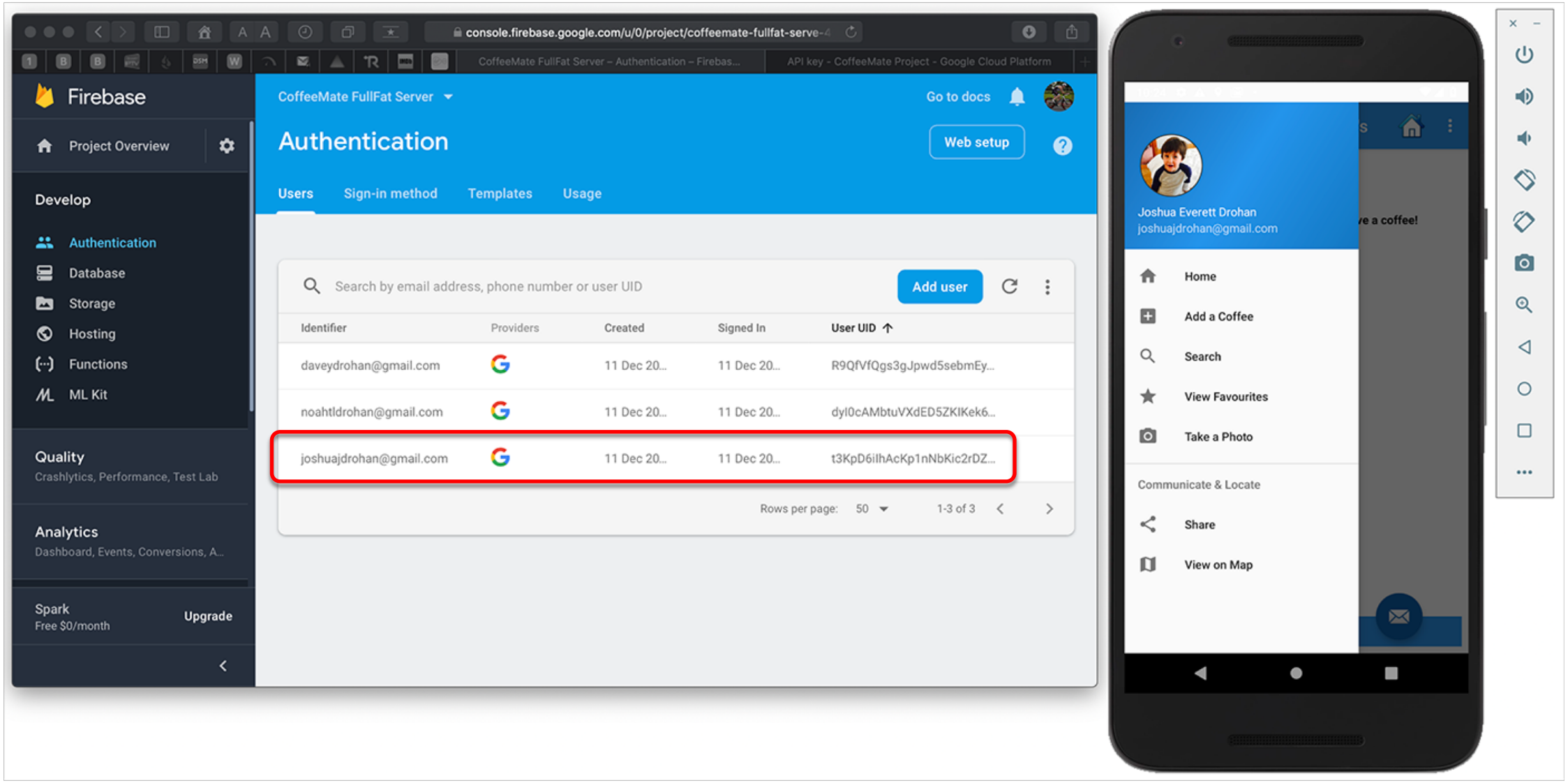
```
app.mFirebaseManager.mFirebaseDatabase.child("users")
        .child(app.mFirebaseUser.getUid()).addListenerForSingleValueEvent(
        new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {...}

            @Override
            public void onCancelled(DatabaseError databaseError) {
                Log.v(TAG, "mFirebaseListener.onFailure " + databaseError);
                Log.v(TAG, "Unable to Validate Existing Firebase User: "
                        + app.mFirebaseUser.getUid());
                Toast.makeText(Login.this,"Unable to Validate Existing Firebase User:",
                        Toast.LENGTH_LONG).show();
            }
        }
);
```

Firebase Console – Authenticated Users *

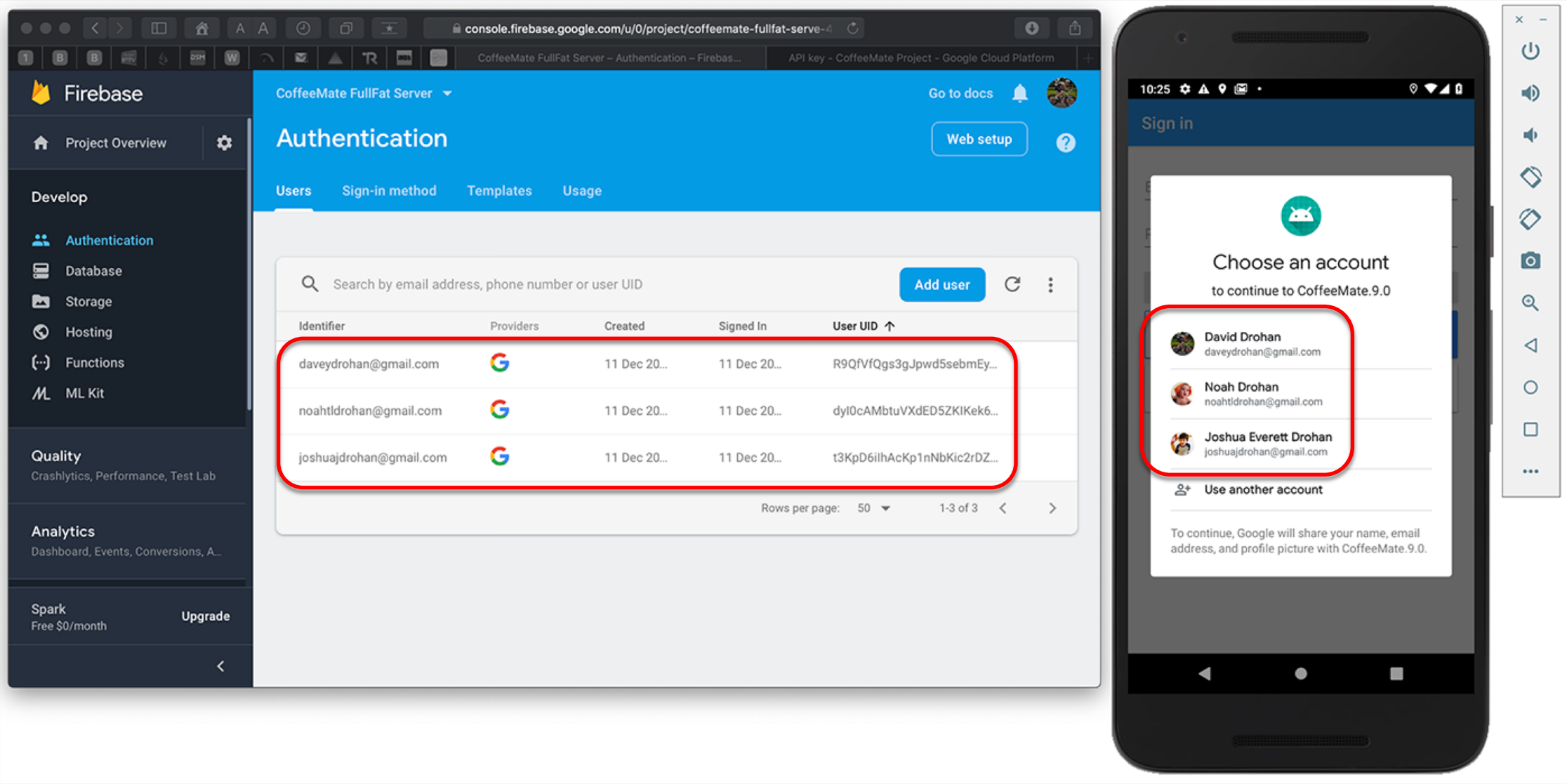




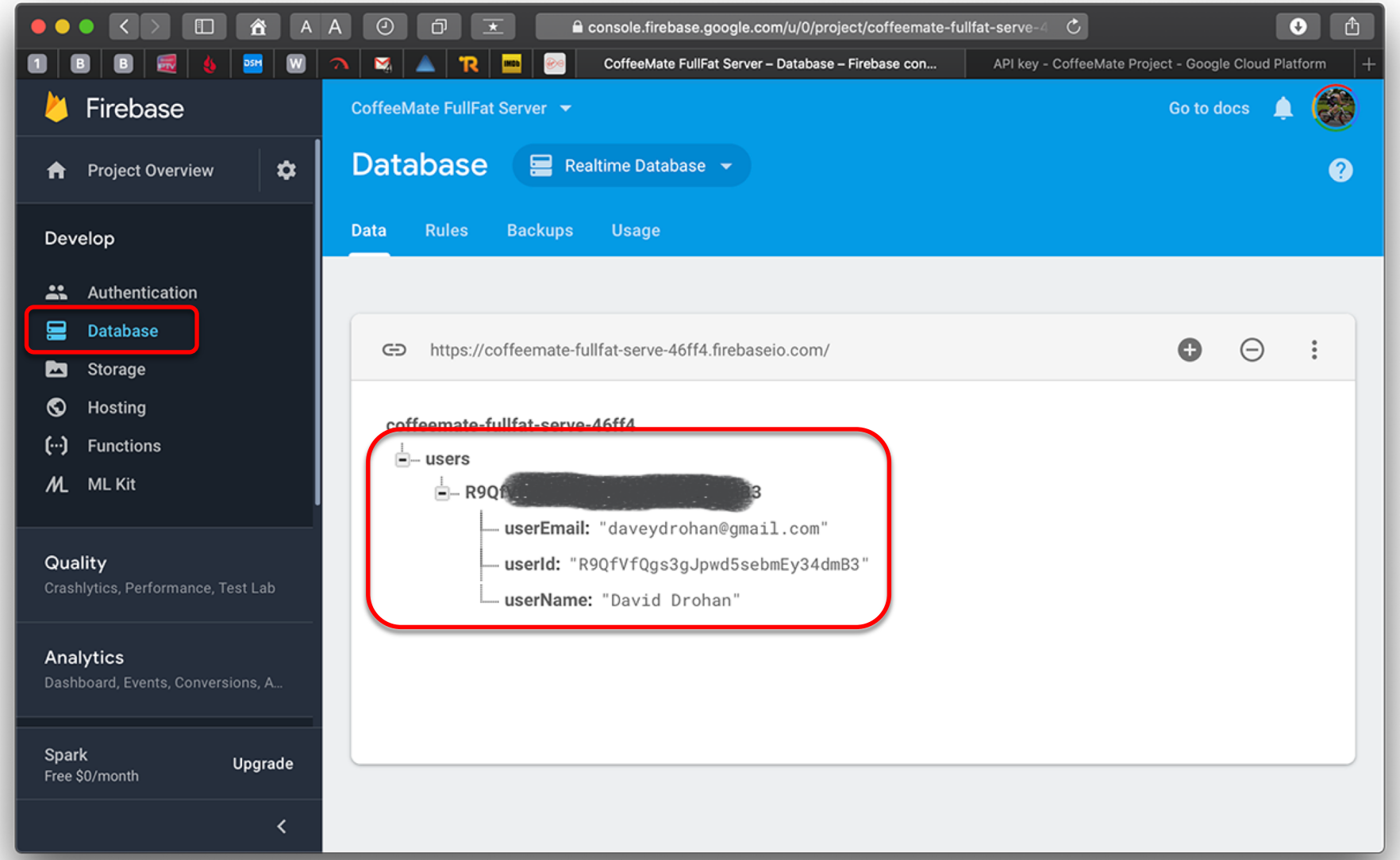
Firebase Console – Authenticated Users *



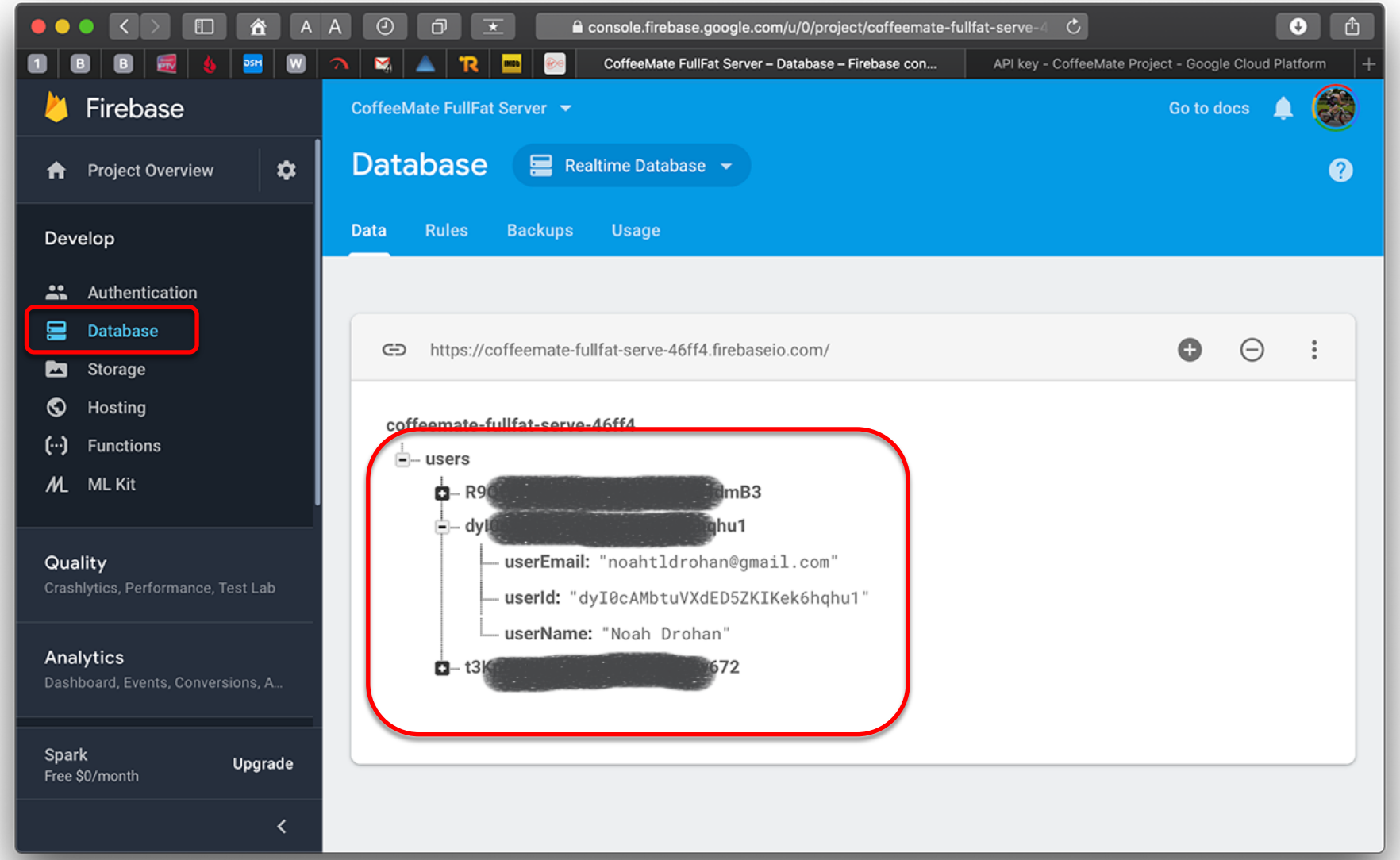
Firebase Console – Authenticated Users *



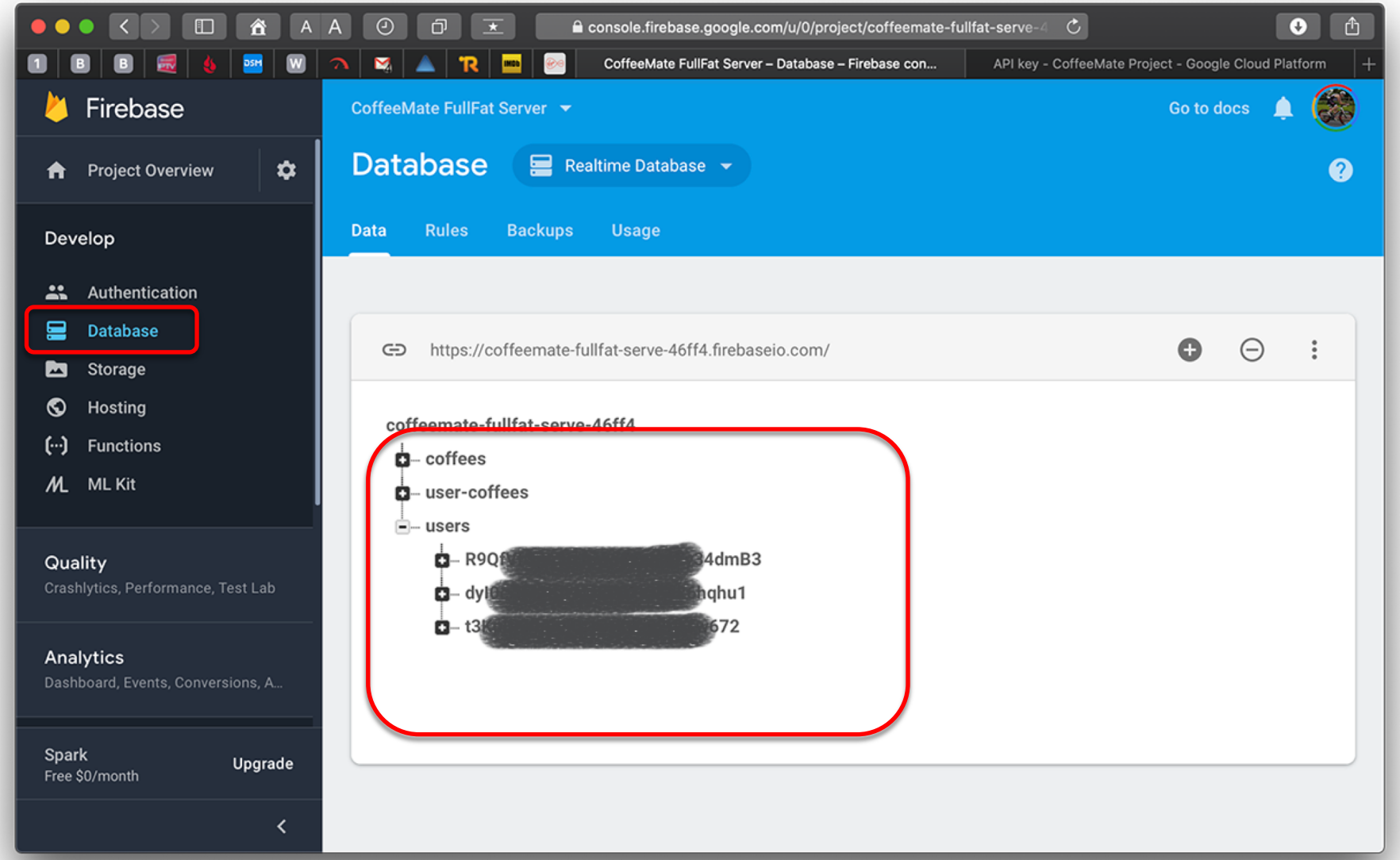
Firebase Console – User Data in db *



Firebase Console – User Data in db *



Firebase Console – User Data in db *



Some important points though…

- Do not think RDBMS, think JSON. How data should be structured is very important.
- Firebase has a recycler view, that integrates with real time database smoothly without any listeners. (FirebaseUI)
- Test lab which is available in paid plan (Blaze), is an amazing feature for testing your app on different real and virtual devices (next section)
- Set developer mode to true when testing Remote Config (next section).

References & Links

- Presentation by Kaushal Dhruw & Shakti Moyal 2016
- https://firebase.google.com
- Demo app available at https://goo.gl/WBP5fR

Questions?


Thanks!