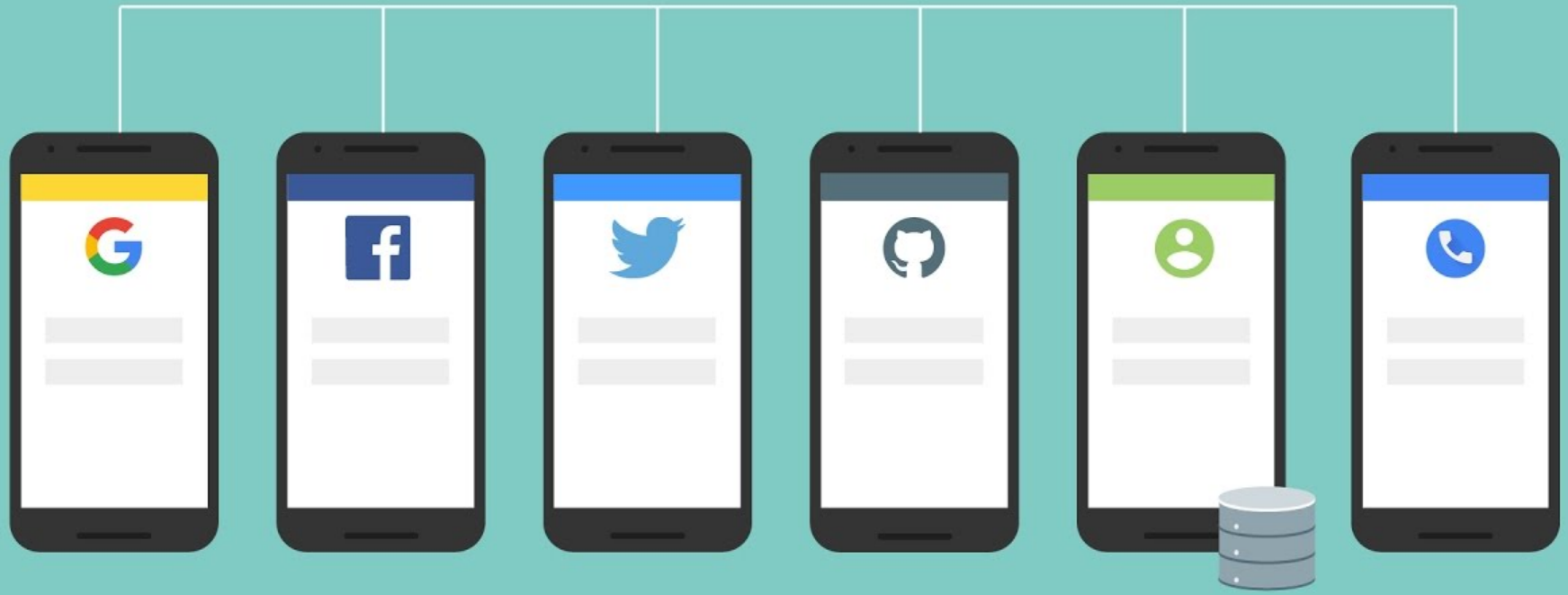
Mobile Application Development

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



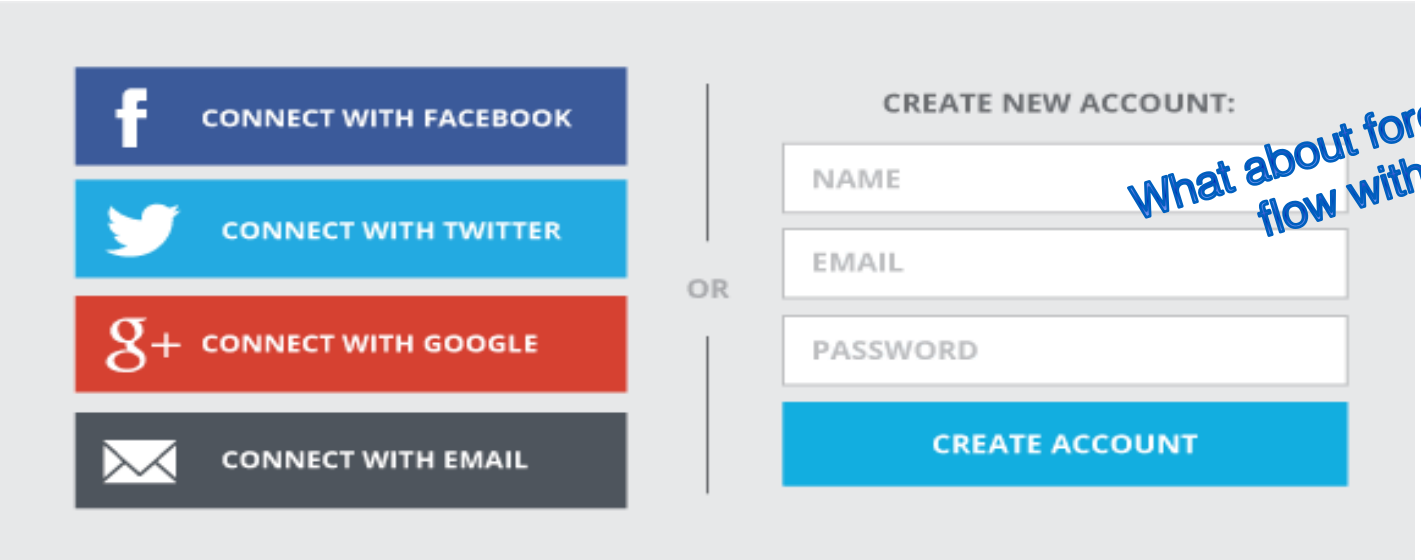
Authentication

Overview

- Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user's devices
- **Firebase Authentication** provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers and popular federated identity providers (later)

Overview

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



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

Overview

- Integrate easily with popular identity providers like Google, Facebook, Twitter and more

Key Capabilities

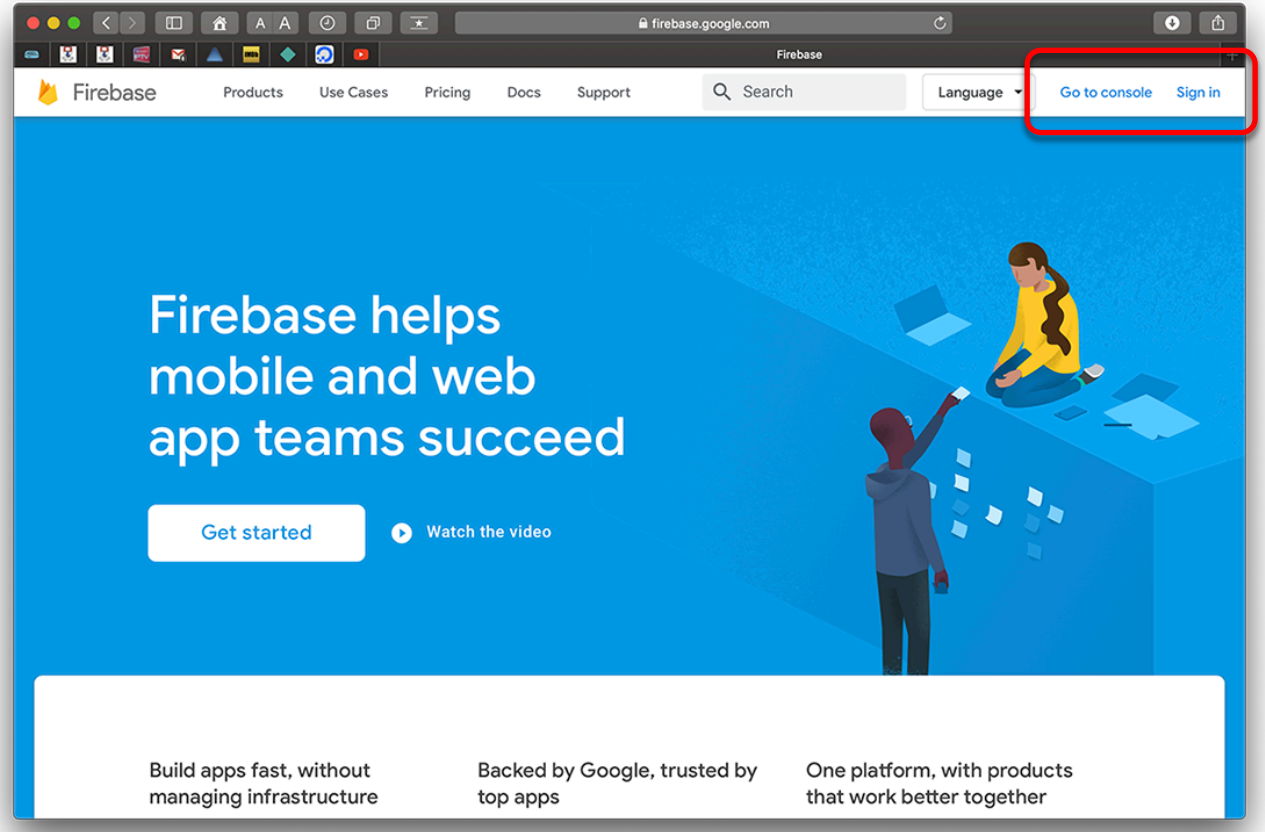
- You can sign in users to your Firebase app either by using **FirebaseUI** as a complete drop-in auth solution or by using the Firebase Authentication SDK to manually integrate one or several sign-in methods into your app
- The **Firebase Authentication SDK** provides methods to create and manage users that use their email addresses and passwords to sign in
- Firebase Authentication also handles sending password reset emails

Firebase Authentication

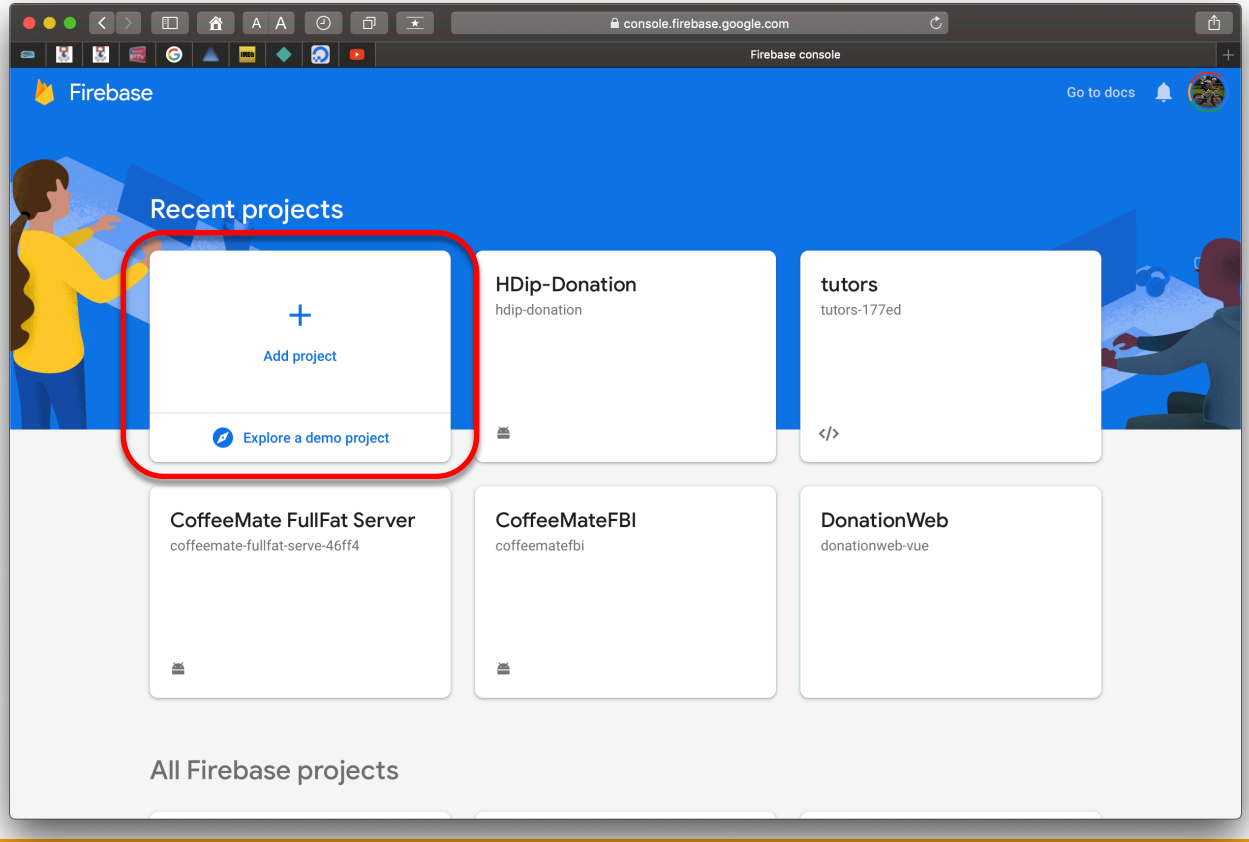
- Of course you will need to register your app with individual service providers.
- Minimal client side handling, integrates seamlessly with firebase
- Ready made ‘forgot password’ flow with customizable email template

Firebase Authentication

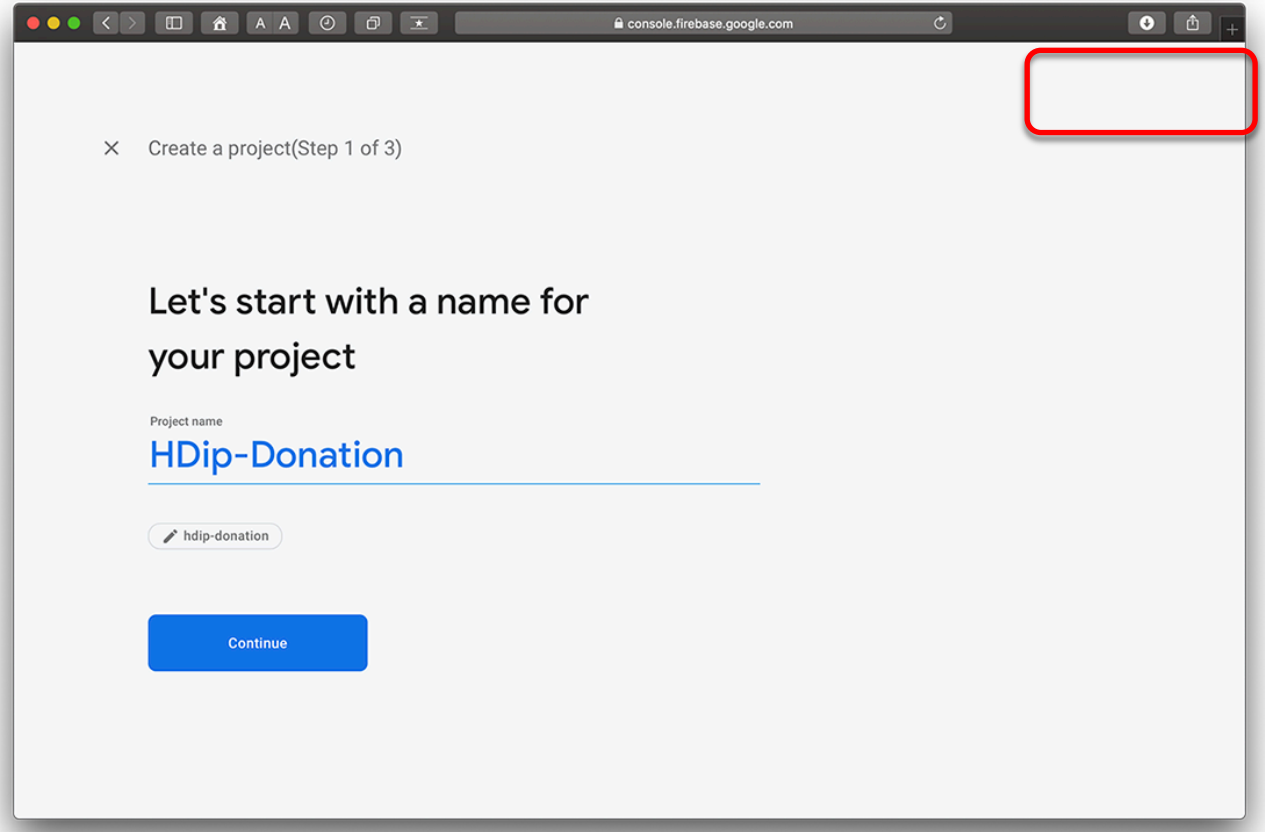
1. Create your Firebase Project



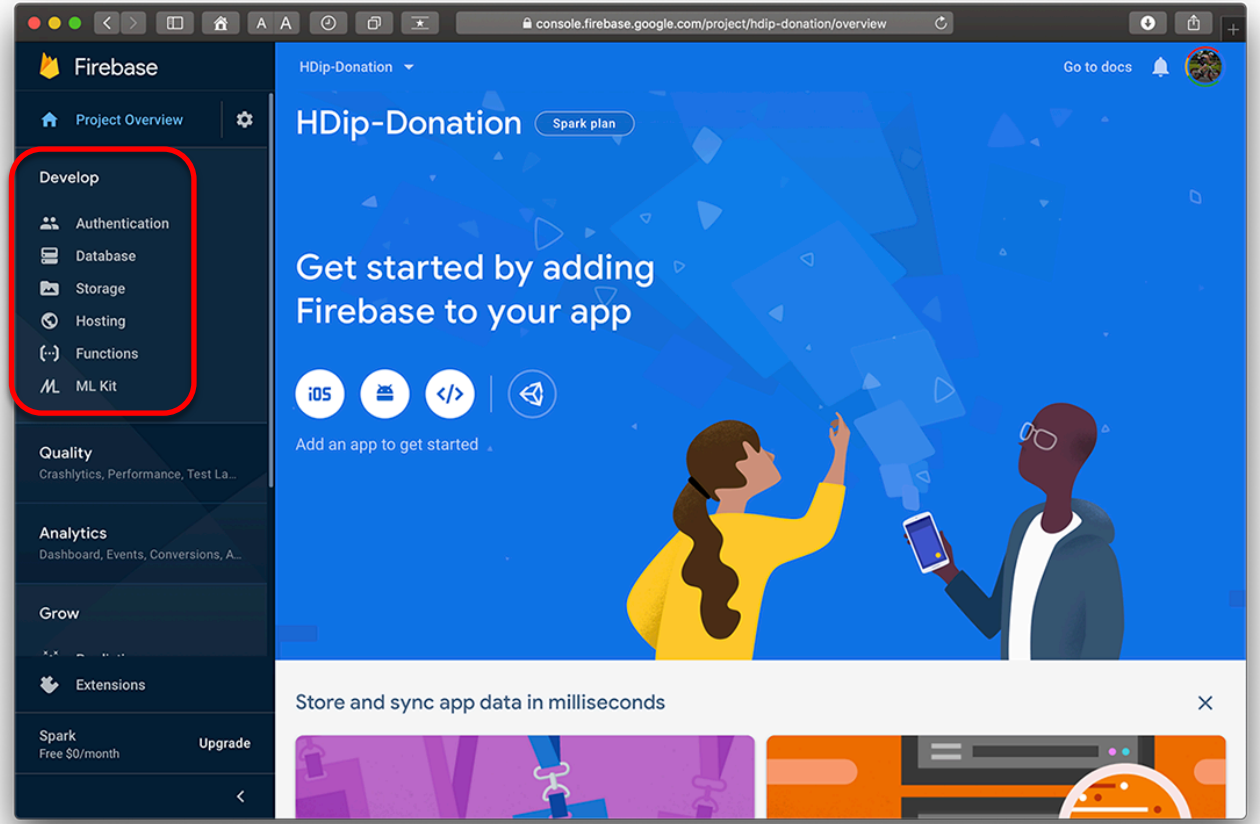
1. Create your Firebase Project



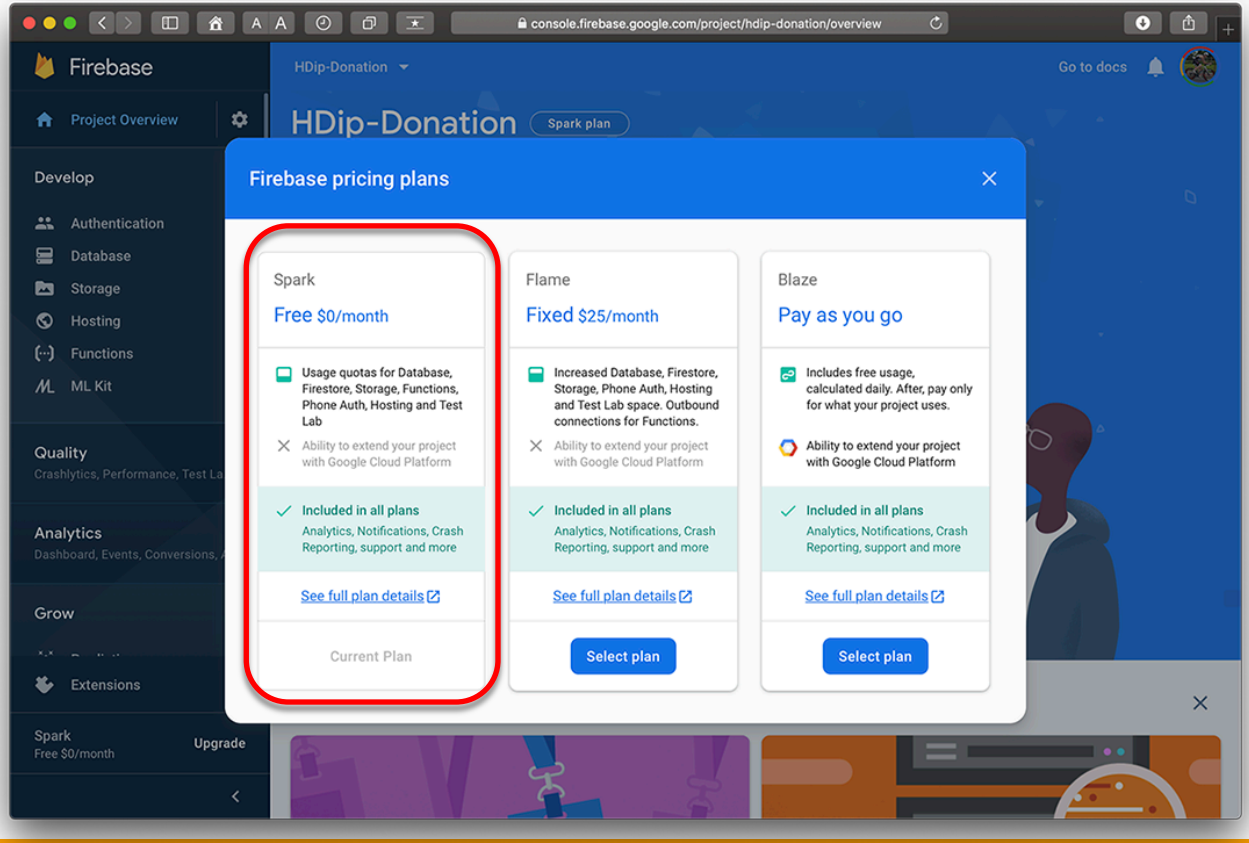
1. Create your Firebase Project



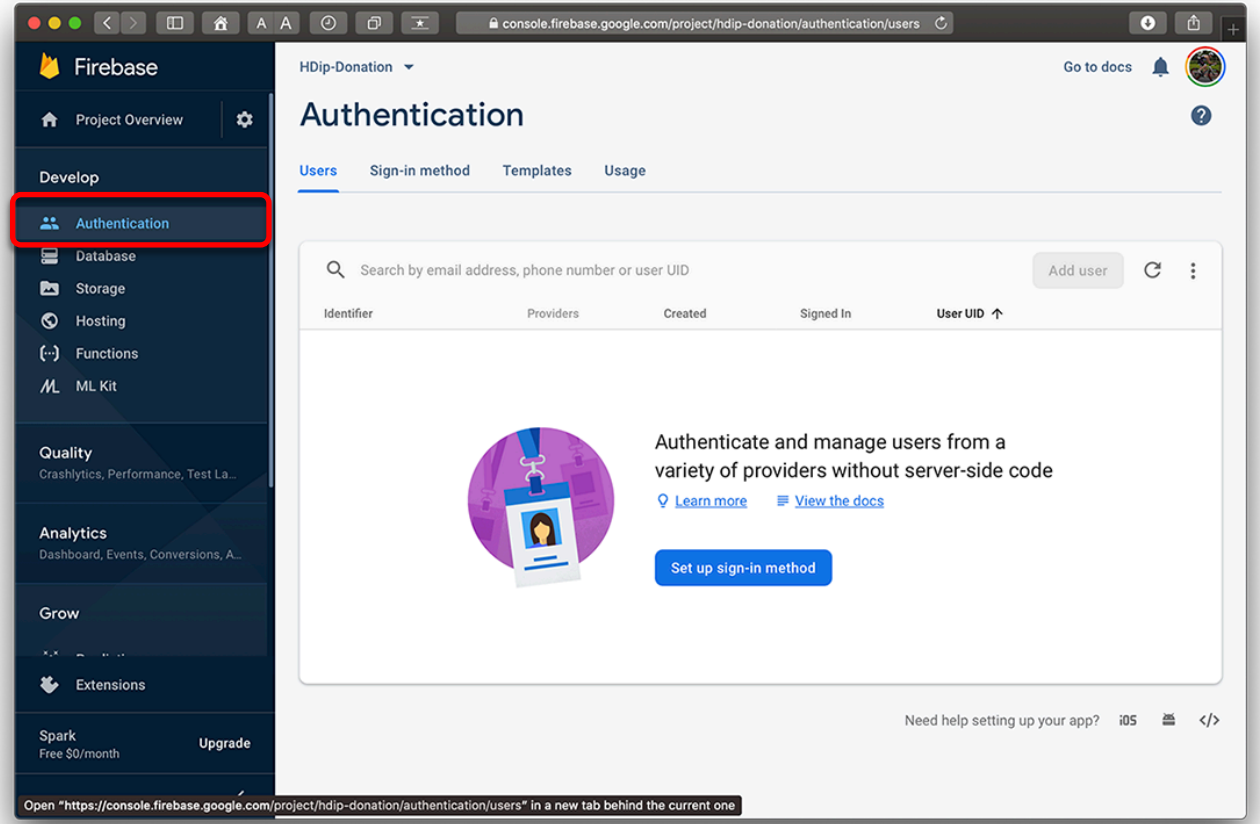
1. Create your Firebase Project



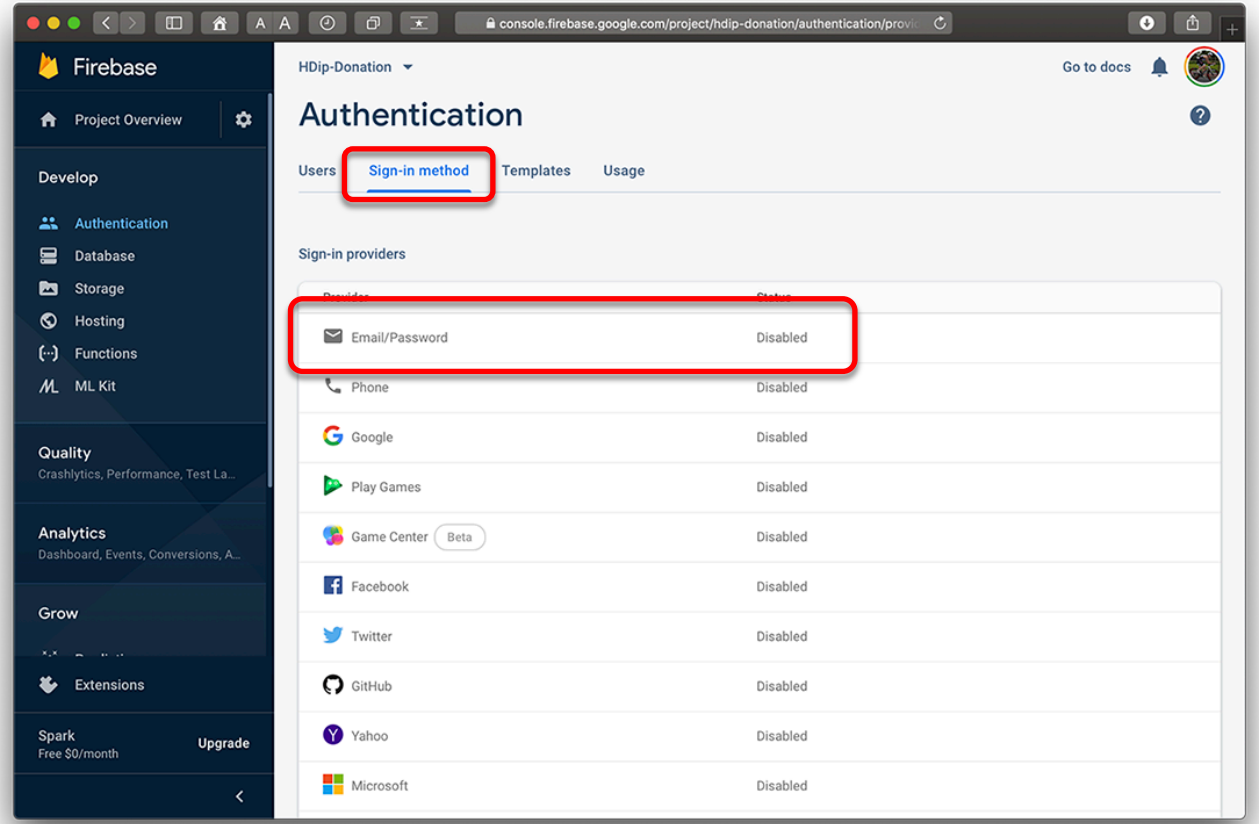
1. Create your Firebase Project



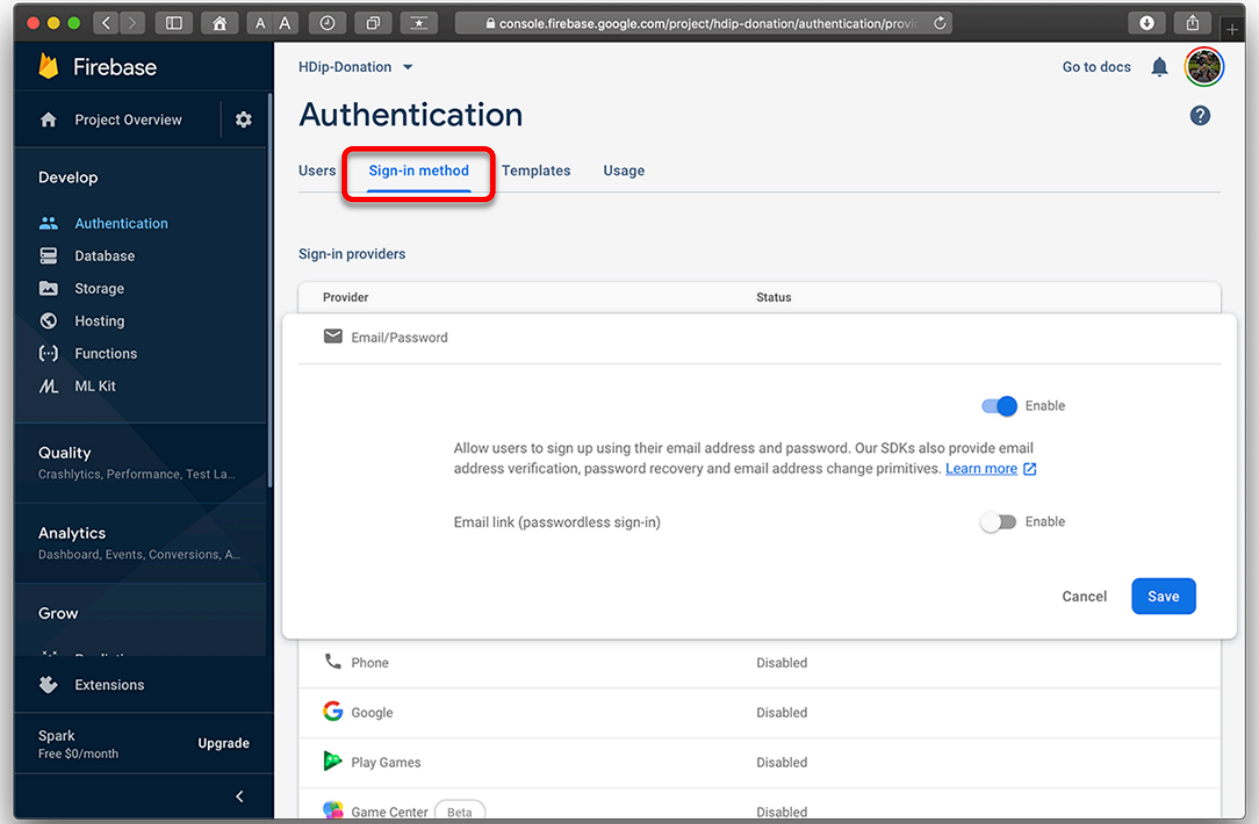
2. Setup your Sign-In Method



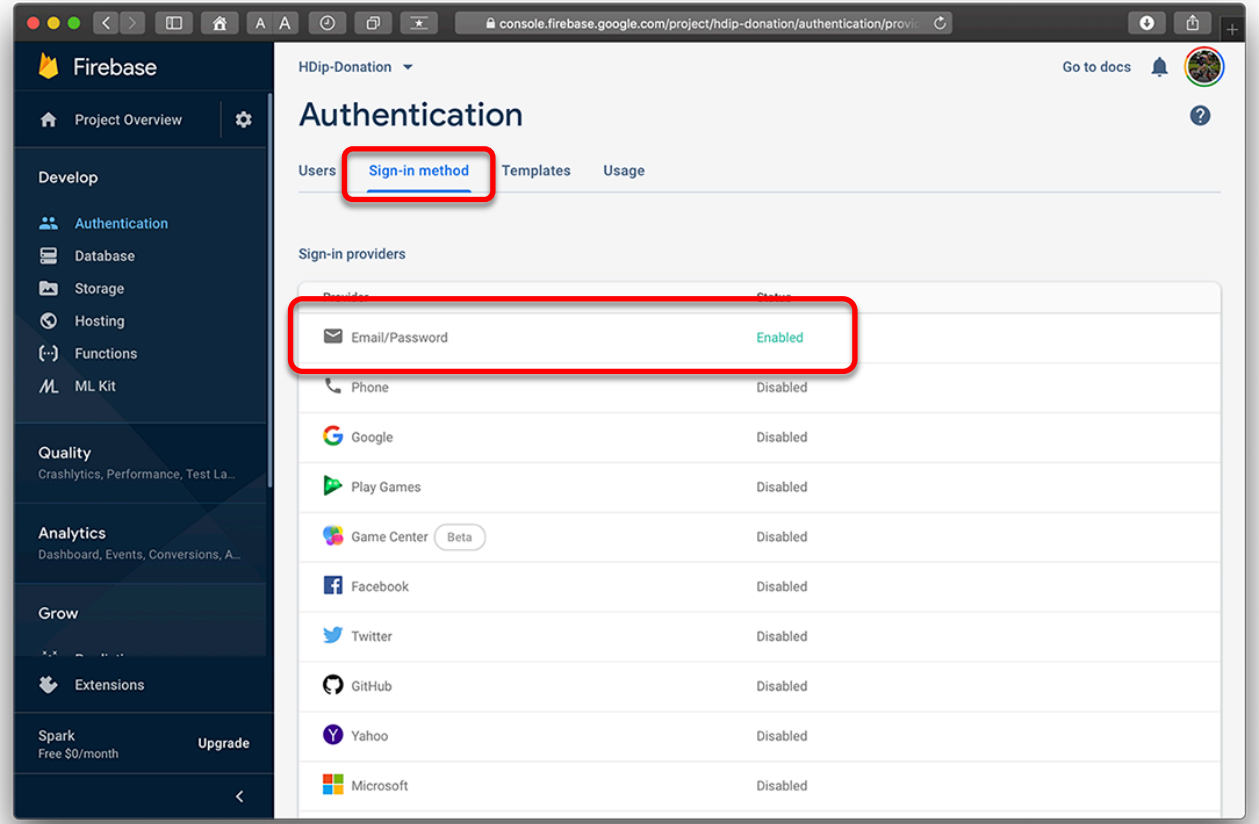
2. Setup your Sign-In Method



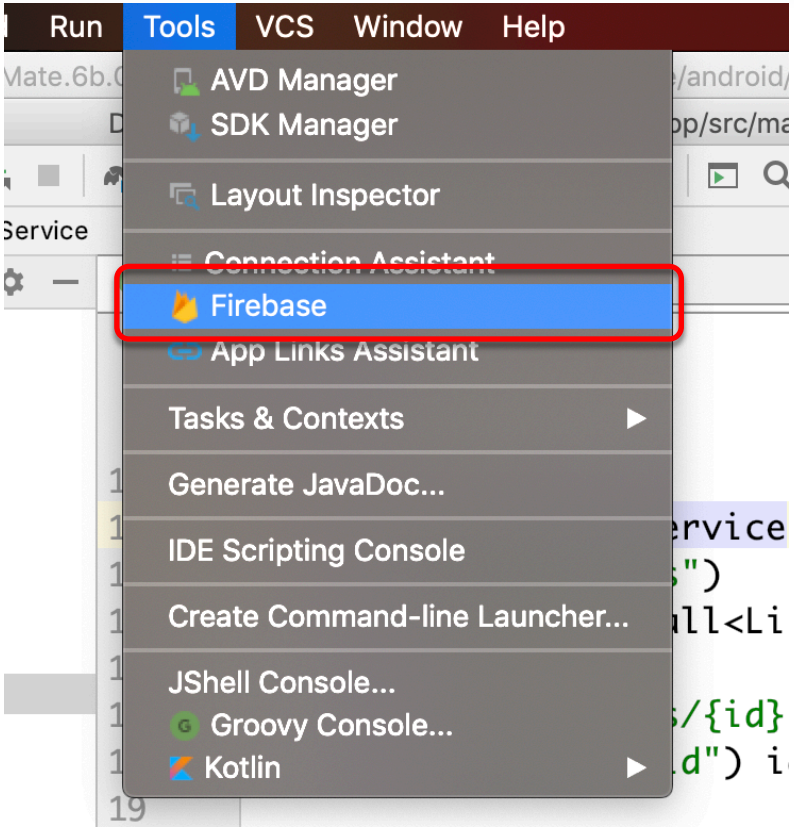
2. Setup your Sign-In Method



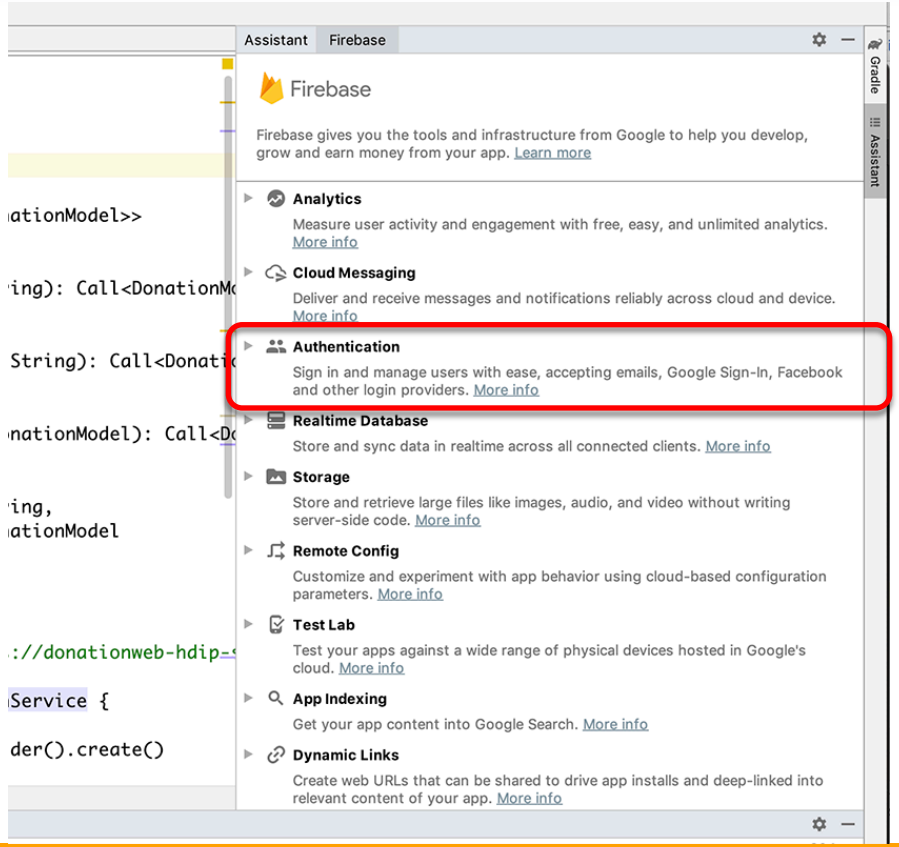
2. Setup your Sign-In Method



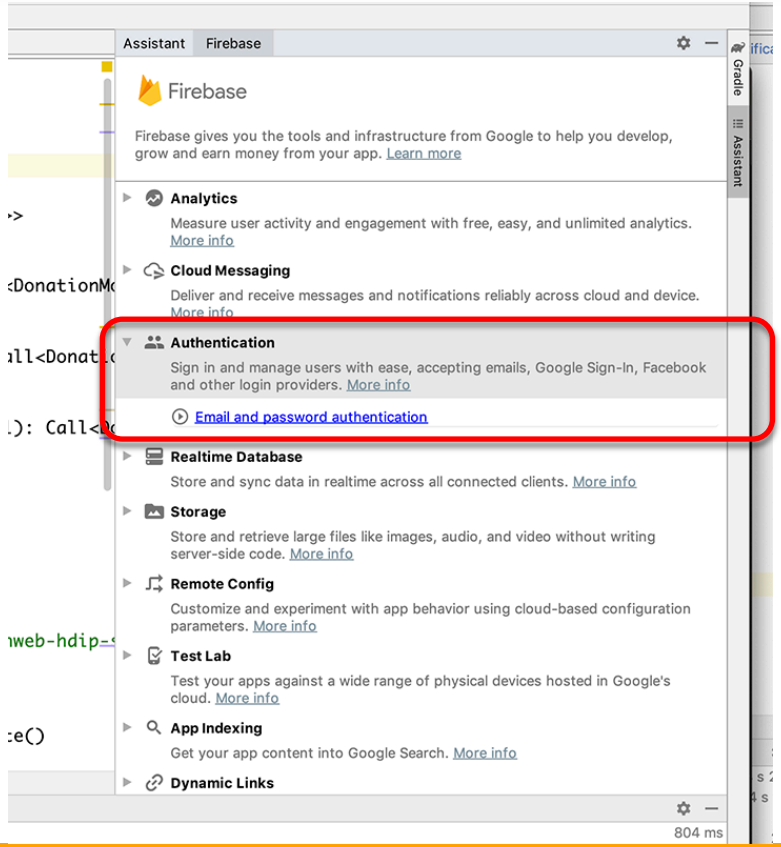
3. Setup Firebase in your App



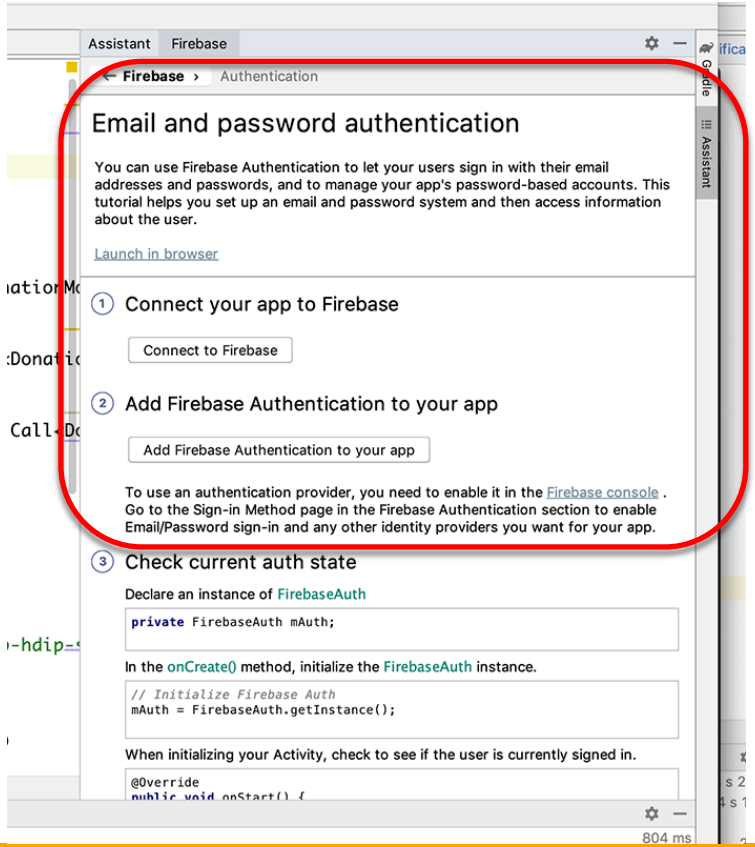
3. Setup Firebase in your App



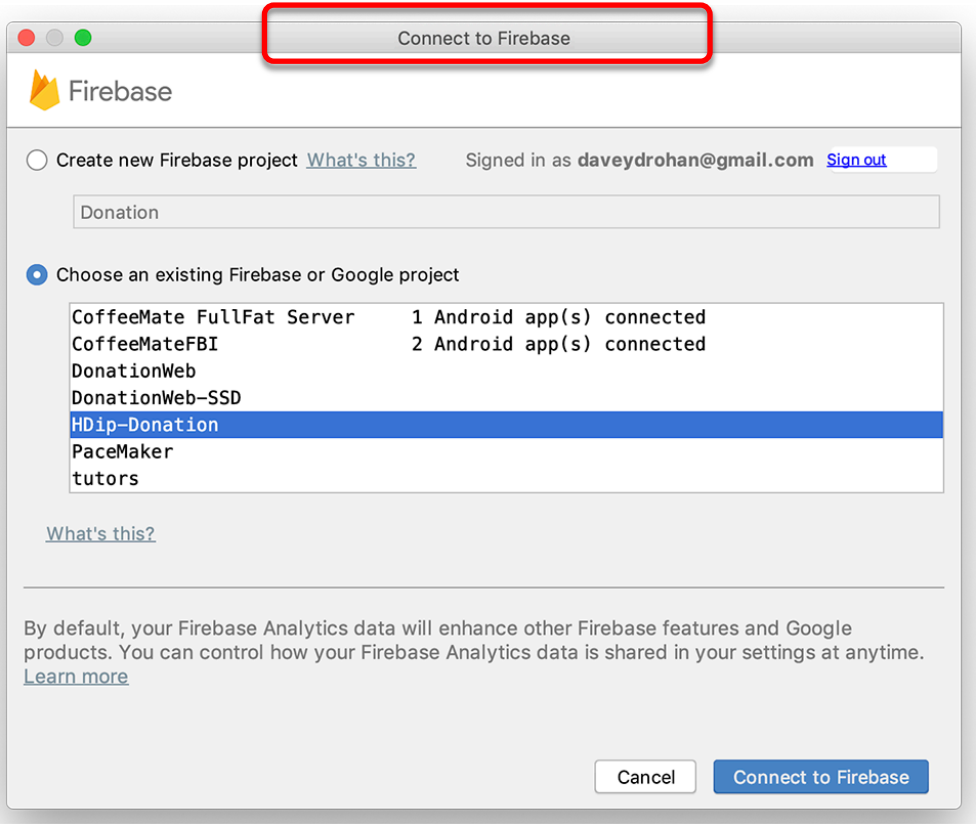
3. Setup Firebase in your App



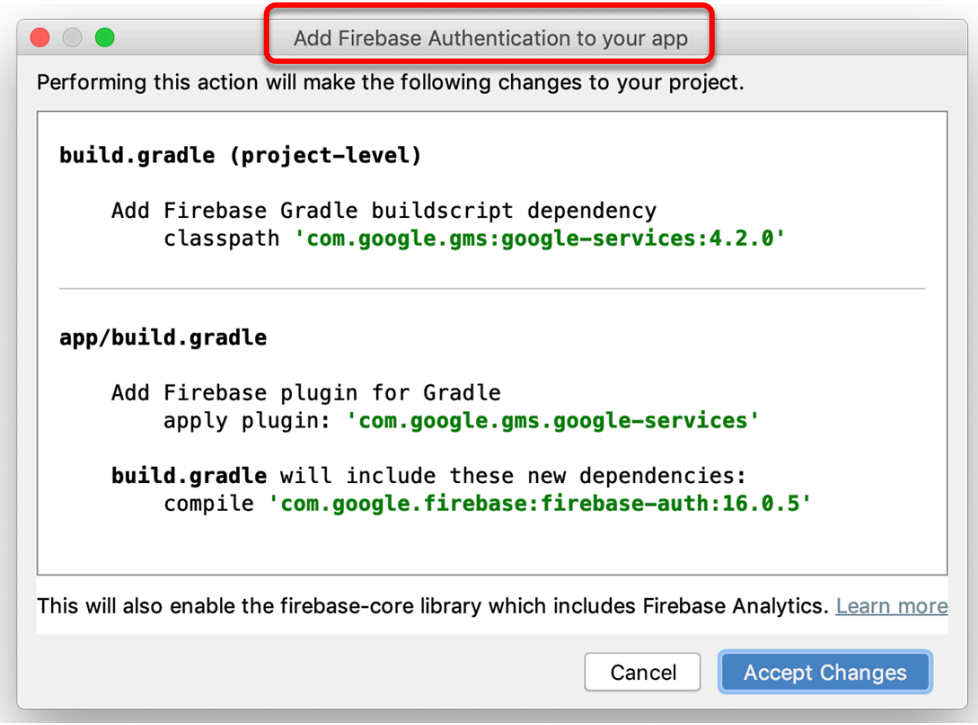
3. Setup Firebase in your App



3. Setup Firebase in your App



3. Setup Firebase in your App



4. Introduce Authentication Flow

https://github.com/firebase/quickstart-android

4. Introduce Authentication Flow – Create Account

To create a new user account with a password, complete the following steps in your app's sign-in activity:

1. In your sign-up activity's `onCreate` method, get the shared instance of the `FirebaseAuth` object:

Java Android | Kotlin Android

```
private lateinit var auth: FirebaseAuth
// ...
// Initialize Firebase Auth
auth = FirebaseAuth.getInstance()
```

EmailPasswordActivity

4. Introduce Authentication Flow – Create Account

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

Java Android | Kotlin Android

```
public override fun onStart() {
    super.onStart()
    // Check if user is signed in (non-null) and update UI accordingly.
    val currentUser = auth.currentUser
    updateUI(currentUser)
}
```

EmailPasswordActivity

3. When a new user signs up using your app's sign-up form, complete any new account validation steps that your app requires, such as verifying that the new account's password was correctly typed and meets your complexity requirements.

4. Create a new account by passing the new user's email address and password to `createUserWithEmailAndPassword`:

Java Android | Kotlin Android

```
auth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                // Sign in success, update UI with the signed-in user's information
                Log.d(TAG, "createUserWithEmail:success")
                val user = auth.currentUser
                updateUI(user)
            } else {
                // If sign in fails, display a message to the user.
                Log.w(TAG, "createUserWithEmail:failure", task.exception)
                Toast.makeText(baseContext, "Authentication failed.",
                        Toast.LENGTH_SHORT).show()
                updateUI(null)
            }

            // ...
        }
```

EmailPasswordActivity.kt

If the new account was created, the user is also signed in. In the callback, you can use the `getCurrentUser` method to get the user's account data.

4. Introduce Authentication Flow – Create Account

4. Introduce Authentication Flow – Sign In a User

1. In your sign-in activity's `onCreate` method, get the shared instance of the `FirebaseAuth` object:

Java Android | Kotlin Android

```
private lateinit var auth: FirebaseAuth
// ...
// Initialize Firebase Auth
auth = FirebaseAuth.getInstance()
```

EmailPasswordActivity

4. Introduce Authentication Flow – Sign In a User

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

Java Android | Kotlin Android

```kotlin
public override fun onStart() {
    super.onStart()
    // Check if user is signed in (non-null) and update UI accordingly.
    val currentUser = auth.currentUser
    updateUI(currentUser)
}
```

EmailPasswordActivity

3. When a user signs in to your app, pass the user's email address and password to `signInWithEmailAndPassword`:

Java Android | Kotlin Android

```kotlin
auth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                // Sign in success, update UI with the signed-in user's information
                Log.d(TAG, "signInWithEmail:success")
                val user = auth.currentUser
                updateUI(user)
            } else {
                // If sign in fails, display a message to the user.
                Log.w(TAG, "signInWithEmail:failure", task.exception)
                Toast.makeText(baseContext, "Authentication failed.",
                        Toast.LENGTH_SHORT).show()
                updateUI(null)
            }

            // ...
        }
```

EmailPasswordActivity.kt

If sign-in succeeded, you can use the returned `FirebaseUser` to proceed.

We'll look at what we do with the authenticated user in the Code walk through and the Lab

References

https://console.firebase.google.com/
https://firebase.google.com/
https://github.com/firebase/quickstart-android


Thanks.