

LabW07 – User Management

Objectives:

1. Understand how to use Firebase for user management
2. Understand login using social account

Tasks:

1. Setup Facebook Login for Firebase app
2. Add Facebook authentication to Firebase app

One benefit of storing data in the cloud is that every user of the app can see the same content. We have learned how to enable basic user authentication in Firebase app using “Email/Password” (refer Lab Week 4). In addition, we have also learned how to set basic Security Rules on our data to restrict access only to users who are signed in, so that we can prevent unauthenticated users from reading or writing to the Cloud Firestore.

Supporting user login and imposing data access permission is important so different users can only view and modify their own data. As login using social account is very common nowadays, in this tutorial we will learn how to enable authentication in Firebase app using Facebook account. Login with other types of social accounts such as Google and Twitter utilise a similar approach like this tutorial.

The following are required in order to complete this tutorial:

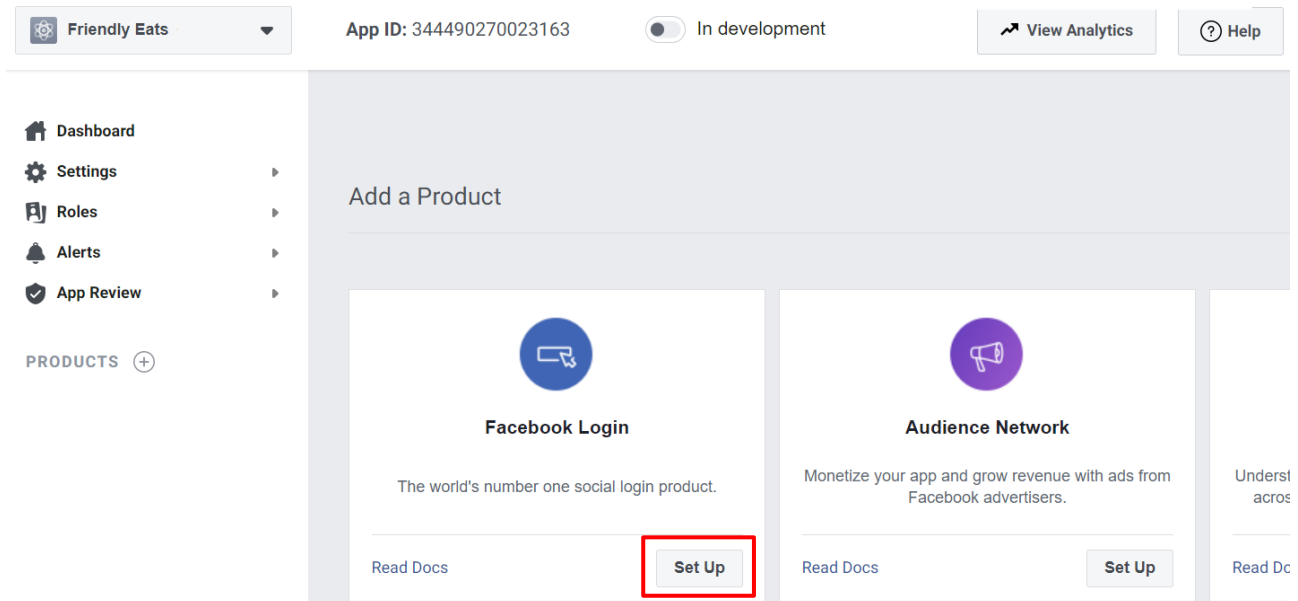
- A valid Facebook account. To create a new Facebook account, go to <https://www.facebook.com/> and sign up for one.
- A Firebase project. We will use a functional Friendly Eats app previously built with Cloud Firestore in Lab Week 4.

Task 1: Setup Facebook Login for Firebase app

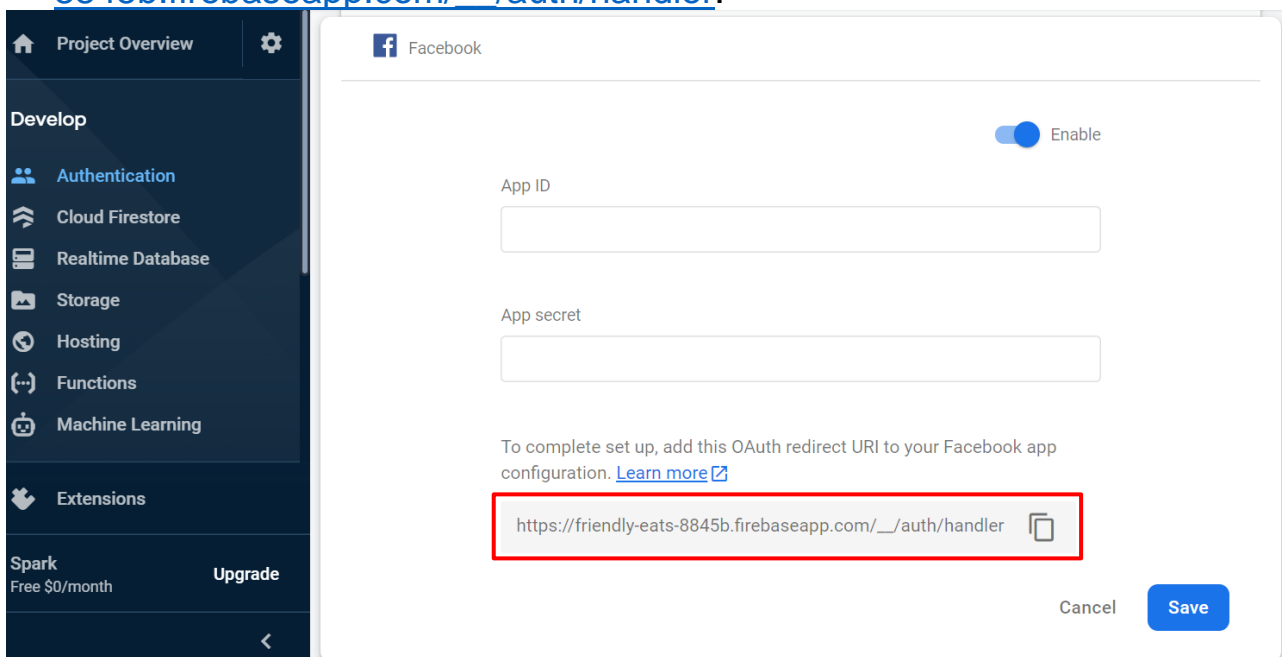
Authentication is critical to applications success. Look at your app, and usually the very first thing that your users will do is to sign up or log in to their account. This will probably leverage some form of social or other types of authentication. Each social authentication provider is a little bit different, and some, such as Facebook, offer a native SDK to simplify the login process and offer additional functionalities specific to their service.

1. To integrate Facebook authentication into our app, we must register a new application on the Facebook for Developers site: <https://developers.facebook.com/>. Log in using your Facebook account.
2. Click “My Apps” on the top right hand corner -> “Create App”.
3. You will be asked to Create an App ID. Since we will be integrating Facebook Login, choose “For Everything Else”. Enter “Friendly Eats” as the App Display Name, and your email address as the App Contact Email.

4. Click “Create App ID”, complete the security check and click Submit.
5. Next, we want to add a proper OAuth redirect URI for our app, which can be done by adding a “Facebook Login” product. Click the “Set Up” button under the “Facebook Login” section:



6. Click Settings to add your Valid OAuth Redirect URI, which can be obtained from your Firebase project.
Go to your Firebase console for “Friendly Eats” app -> Authentication -> “Sign-in method” -> enable Facebook authentication.
Copy the OAuth redirect URI as displayed e.g. https://friendly-eats-8845b.firebaseio.com/_/auth/handler.



7. Add the URI to your Facebook app's "Valid OAuth Redirect URIs" configuration. Make sure that you are using the HTTPS scheme. Click "Save Changes" to save your changes.

Friendly Eats APP ID: 368375533839504 OFF Status: In Development [View Analytics](#) [Help](#)

Client OAuth Settings

- ☒ **Client OAuth Login**
Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking down which token redirect URIs are allowed with the options below. Disable globally if not used. [\[?\]](#)
- ☒ **Web OAuth Login**
Enables web-based Client OAuth Login. [\[?\]](#)
- ☒ **Enforce HTTPS**
Enforce the use of HTTPS for Redirect URIs and the JavaScript SDK. Strongly recommended. [\[?\]](#)
- ☐ **Force Web OAuth Reauthentication**
When on, prompts people to enter their Facebook password in order to log in on the web. [\[?\]](#)
- ☐ **Embedded Browser OAuth Login**
Enable webview Redirect URIs for Client OAuth Login. [\[?\]](#)
- ☒ **Use Strict Mode for Redirect URIs**
Only allow redirects that use the Facebook SDK or that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [\[?\]](#)

Valid OAuth Redirect URIs

https://friendly-eats-8845b.firebaseio.com/_/auth/handler [×](#)

☐ Login from Devices Discard Save Changes

8. To enable Facebook authentication in Firebase, we need to get the App ID and App Secret. Click Settings -> Basic -> get the App ID and App Secret.

Friendly Eats APP ID: 368375533839504 OFF Status: In Development [View Analytics](#) [Help](#)

Basic

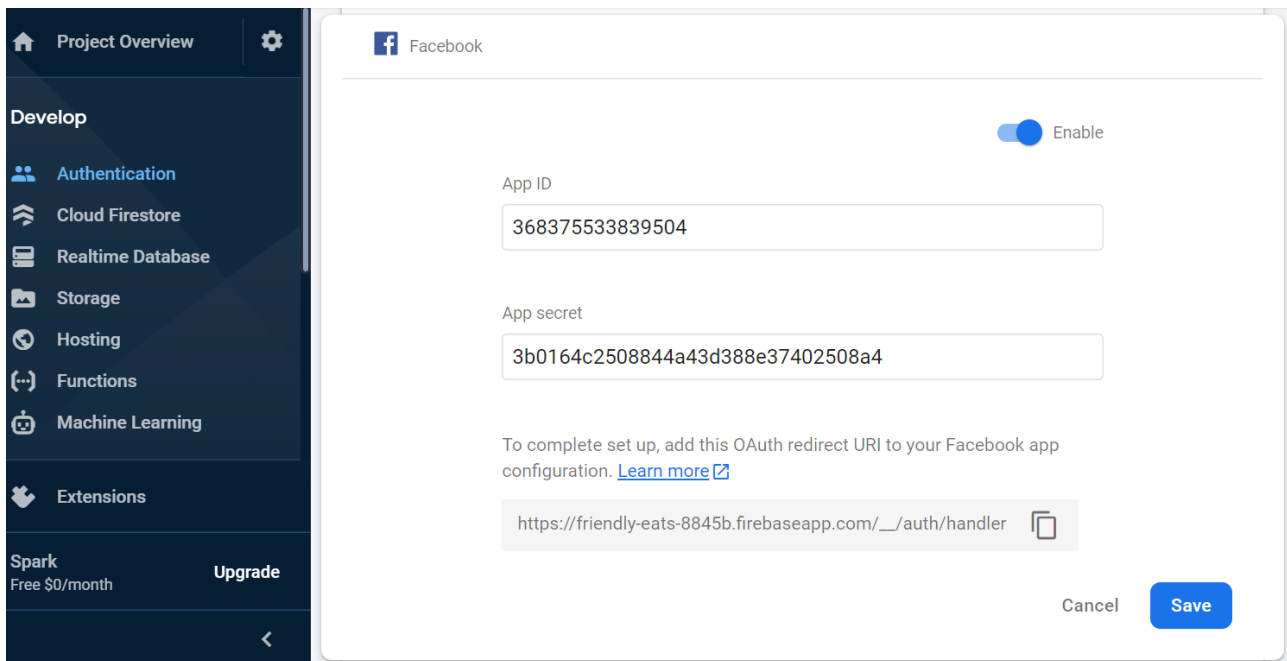
App ID: 368375533839504

App Secret: [Show](#)

Display Name: Friendly Eats

Namespace:

9. Enter the App ID and App Secret from step 8 into your Facebook authentication configuration for your Firebase project. Click Save.



The screenshot shows the Firebase console interface. On the left is a dark sidebar with navigation options: Project Overview, Develop (with sub-items: Authentication, Cloud Firestore, Realtime Database, Storage, Hosting, Functions, Machine Learning), Extensions, Spark (Free \$0/month), and an Upgrade button. The main panel is titled 'Facebook' and contains a toggle switch labeled 'Enable' which is turned on. Below this, there are two text input fields: 'App ID' containing '368375533839504' and 'App secret' containing '3b0164c2508844a43d388e37402508a4'. A paragraph of text instructs the user to add the OAuth redirect URI to their Facebook app configuration, with a link to 'Learn more'. Below this text is a text box containing the URI 'https://friendly-eats-8845b.firebaseio.com/_/auth/handler' and a copy icon. At the bottom right of the panel are 'Cancel' and 'Save' buttons.

10. Facebook authentication is now configured for your app.

Task 2: Add Facebook authentication to Firebase app

In this task, you will add authentication to the Friendly Eats Android app by using the Facebook login setup in the previous task.

Using FirebaseUI Auth is the recommended way to add a complete sign-in system to your app. It provides a drop-in auth solution that handles the UI flows for signing in users with email addresses and passwords, phone numbers, and with popular federated identity providers, including Facebook Login and Google Sign-In.

1. In your project-level build.gradle file, make sure to include Google's Maven repository in both your buildscript and allprojects sections.

```
buildscript {
    repositories {
        // Add this line
        google()
    }
    ...
}

allprojects {
    repositories {
        // Add this line
        google()
    }
    ...
}
```

2. Add the dependencies for FirebaseUI to your app-level build.gradle file. To support sign-in with Facebook, also include the Facebook SDK. The FirebaseUI Auth SDK has transitive dependencies on the Firebase SDK and the Google Play services SDK.

```
dependencies {
    ...

    // FirebaseUI (for authentication)
    implementation 'com.firebaseui:firebase-ui-auth:6.2.0'

    // Required only if Facebook login support is required
    // Find the latest Facebook SDK releases here: https://goo.gl/Ce5L94
    implementation 'com.facebook.android:facebook-android-sdk:5.0.0'

    ...
}
```

3. Add string resources to strings.xml that specify the identifying information required by Facebook.

```
<resources>
    ...

    <!-- Facebook application ID and custom URL scheme (app ID prefixed by 'fb'). -->
    <string name="facebook_application_id" translatable="false">YOUR_APP_ID</string>
    <string name="facebook_login_protocol_scheme"
translatable="false">fbYOUR_APP_ID</string>

    ...
</resources>
```

4. To kick off the FirebaseUI sign in flow, create a sign in intent with your preferred sign-in methods. Because our Friendly Eats app has previously

supported login using “Email/Password”, we can simply add Facebook authentication provider as another sign-in method to `startSignIn()` method.

```
private void startSignIn() {
    // Sign in with FirebaseUI
    Intent intent = AuthUI.getInstance().createSignInIntentBuilder()
        .setAvailableProviders(Arrays.asList(
            new AuthUI.IdpConfig.EmailBuilder().build(),
            new AuthUI.IdpConfig.FacebookBuilder().build()))
        .setIsSmartLockEnabled(false)
        .build();

    startActivityForResult(intent, RC_SIGN_IN);
    mViewModel.setIsSigningIn(true);
}
```

5. When the sign-in flow is complete, you will receive the result in `onActivityResult`:

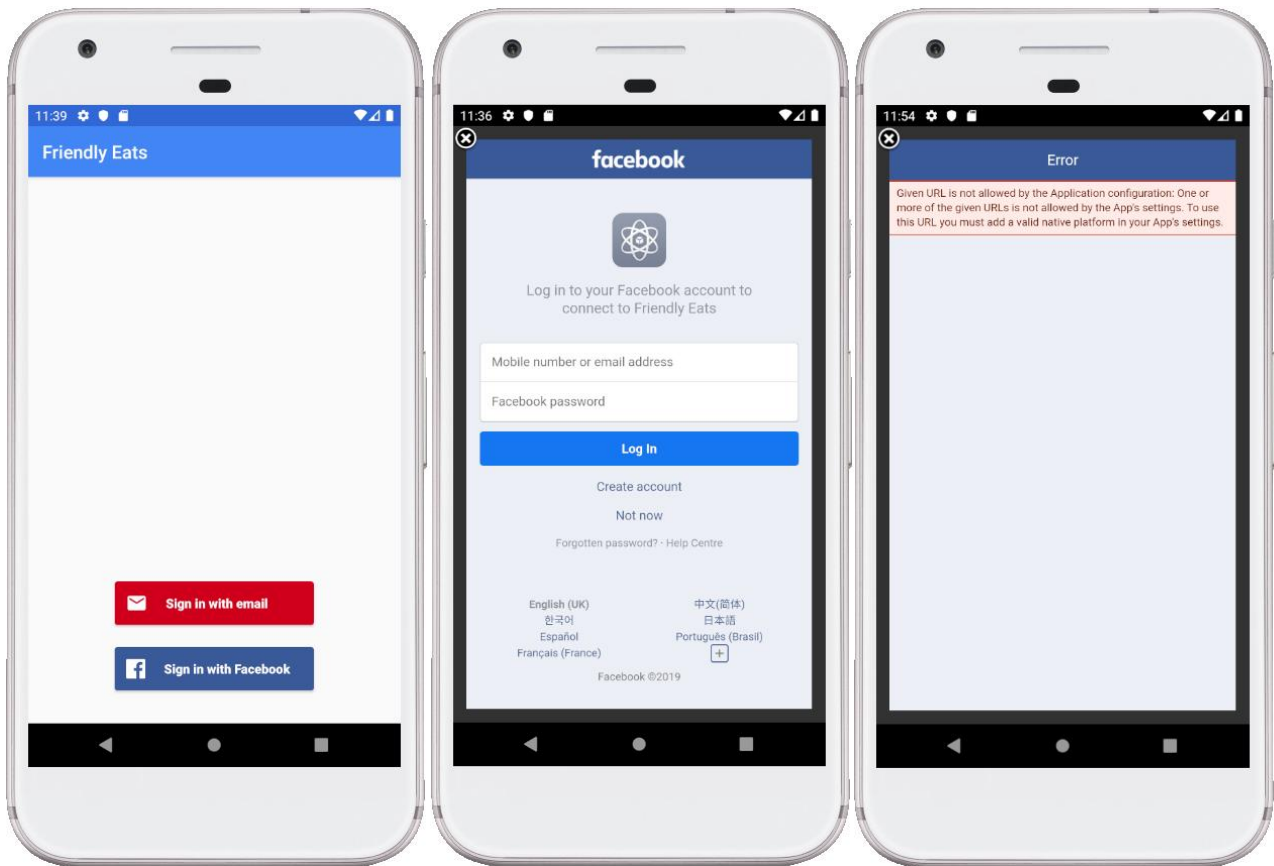
```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == RC_SIGN_IN) {
        mViewModel.setIsSigningIn(false);

        if (resultCode != RESULT_OK && shouldStartSignIn()) {
            startSignIn();
        }
    }
}
```

6. FirebaseUI provides convenience method `signOut()` to sign out of Firebase Authentication as well as social identity provider like Facebook:

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menu_add_items:
            onAddItemsClicked();
            break;
        case R.id.menu_sign_out:
            AuthUI.getInstance().signOut(this);
            startSignIn();
            break;
    }
    return super.onOptionsItemSelected(item);
}
```

7. Build and run your app. You should get another option to “Sign in with Facebook” in addition to “Sign in with email”. Clicking “Sign in with Facebook” button will prompt you to sign in to the app using Facebook account. Use your Facebook account to login and you should notice an error message “... URL is not allowed ...” – refer to screenshots below.



8. The error is because we have not completely integrated our Firebase app with Facebook. Open Facebook for Developers site and add an Android platform. Click Settings -> Basic -> +Add Platform -> Android.

Friendly Eats APP ID: 368375533839504 OFF Status: In Development View Analytics Help

Dashboard

Settings

Basic

Advanced

Roles

Alerts

App Review

PRODUCTS (+)

Facebook Login

Analytics

Apt/Suite/Other (Optional)

City/District

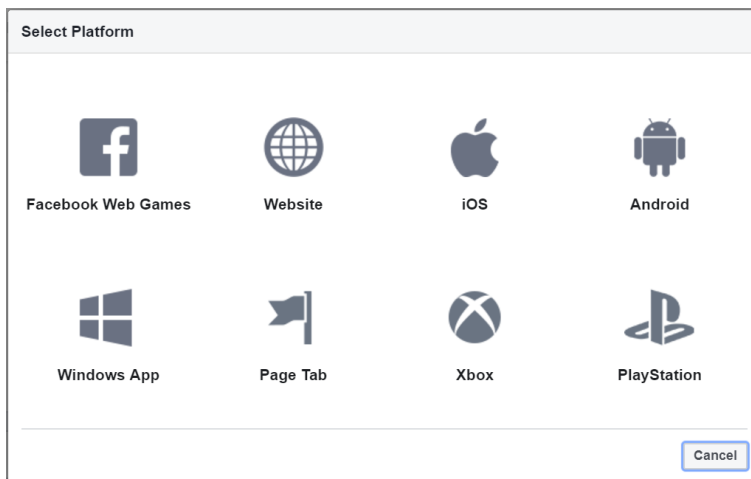
State/Province/Region

ZIP/Postal Code

Country

United States

+ Add Platform



9. Enter your app package name and default activity class name:
 - Google Play Package Name: com.google.firebase.example.fireeats
 - Class Name: com.google.firebase.example.fireeats.MainActivity

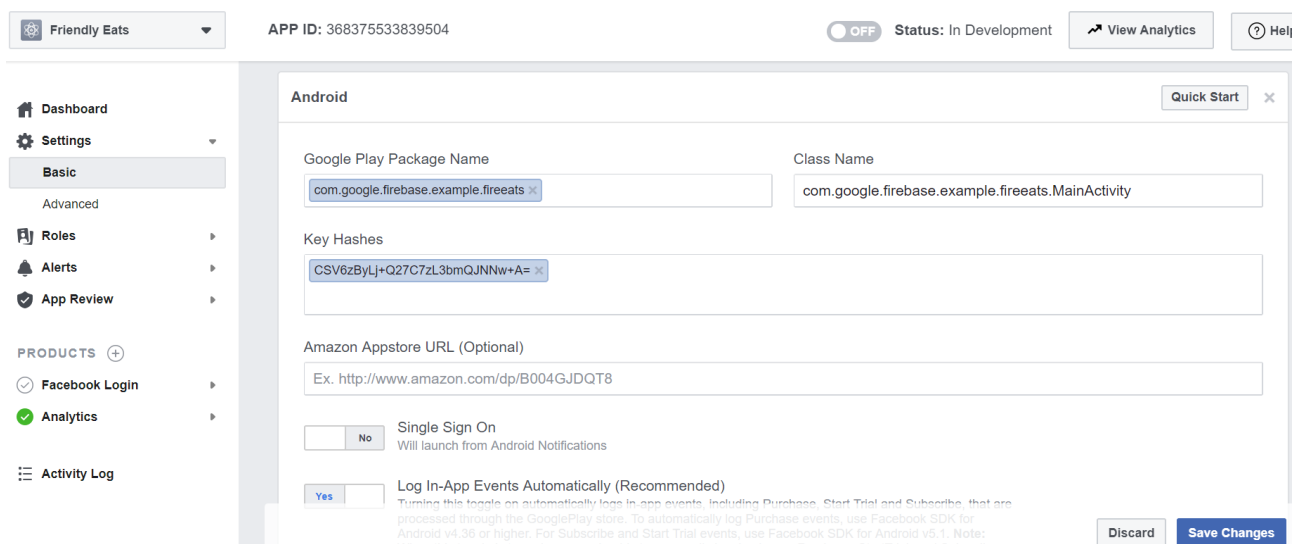
10. Generate a development key hash to ensure the authenticity of the interactions between your app and Facebook. You'll have a unique development key hash for each Android development environment. To generate a development key hash on Windows, run this command:

```
"C:\Program Files\Android\Android Studio\jre\bin\keytool" -exportcert -alias androiddebugkey -keystore %HOMEPATH%\android\debug.keystore | "C:\openssl-0.9.8k_X64\bin\openssl" sha1 -binary | "C:\openssl-0.9.8k_X64\bin\openssl" base64
```

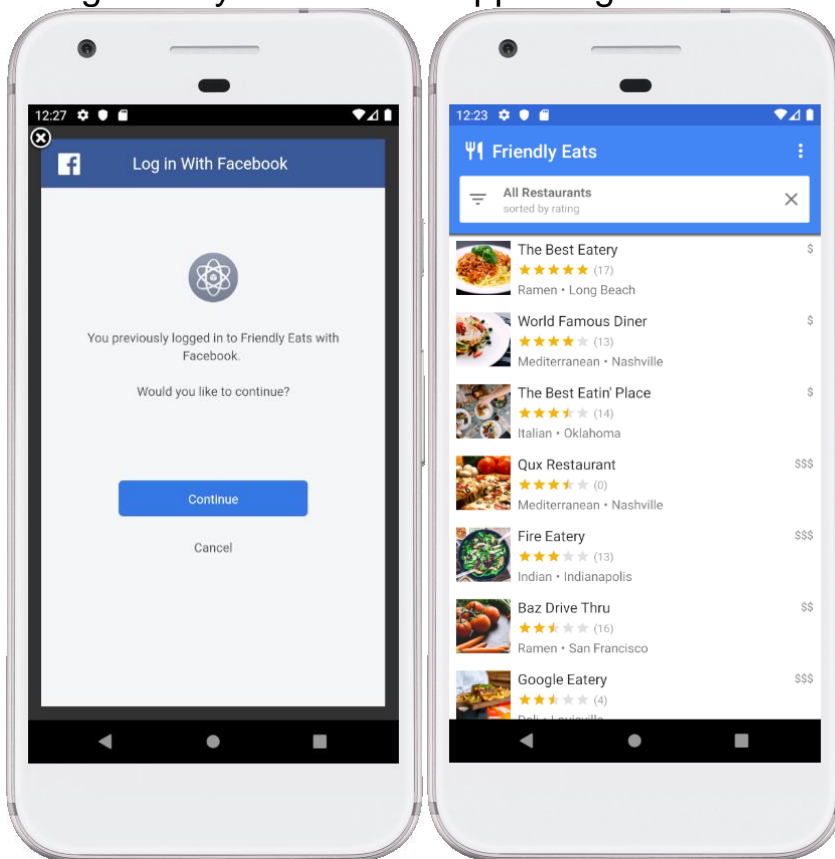
Make sure you have downloaded OpenSSL for Windows from this link:

<https://code.google.com/archive/p/openssl-for-windows/downloads>

11. Add the hash to the “Key Hashes”. Save Changes.



12. Rebuild and run your app. This time you should be able to successfully sign in to your Firebase app using Facebook account.



13. [Optional] By default FirebaseUI uses AppCompat for theming, which means it will naturally adopt the color scheme of your app. Customize the theme and logo of Facebook Sign In by passing a theme and a logo to the sign-in Intent builder.

References:

- Firebase Authentication - <https://firebase.google.com/docs/auth/>
- Easily add sign-in to your Android app with FirebaseUI - <https://firebase.google.com/docs/auth/android/firebaseui>
- Authenticate Using Facebook Login on Android - <https://firebase.google.com/docs/auth/android/facebook-login>
- OpenSSL for Windows - <https://code.google.com/archive/p/openssl-for-windows/downloads>