Android Programming Flipped Classroom: Firebase Authentication

Authentication is the process by which a system establishes identity. How does a computer system know I'm "witchel?" Because I know a secret that the person who set up the account with user name `witchel` knew. That secret is a password. Or there might be another secret shared with the entity who set up the account like a biometric identifier such as a fingerprint or a face scan.

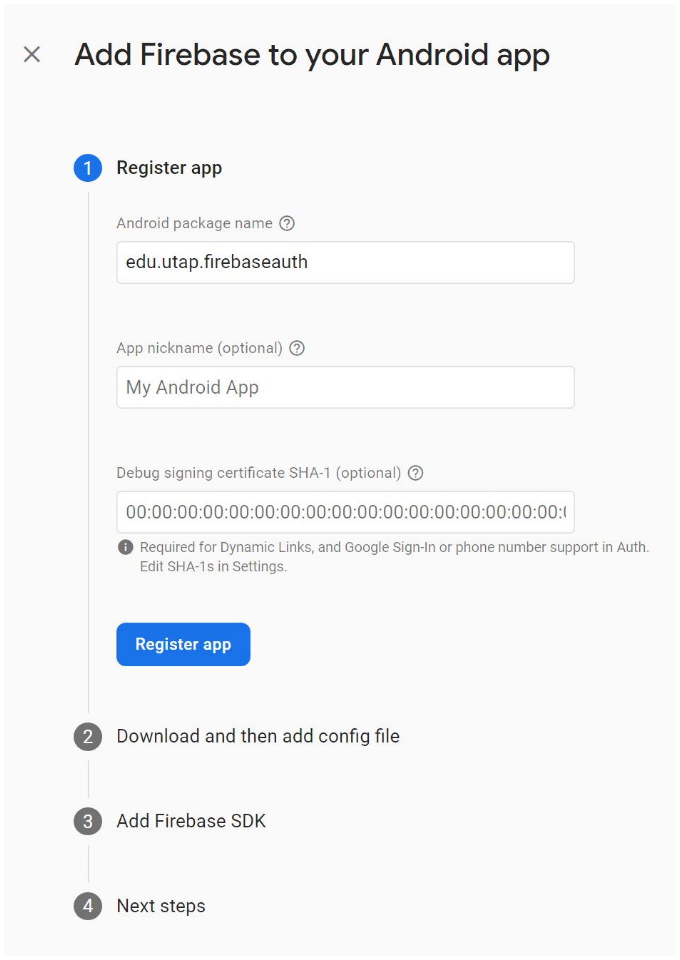
Authentication is a basic part of most networked apps because it allows people to have an identity within the app. In this FC we will use a cool "canned" login page that is provided by Google. That saves us work. Then we will see how that the authentication information gets distributed via a ViewModel and live data to an app.

There are three pieces of information being managed by the Firebase authentication system. The user's email, display name, and their user identifier (UID). The user signs up with an email (and password) and a display name. The system assigns the uid. The app allows the user to change their display name. When the user logs out and logs back in, the login screen should prompt the user just for their password (the login screen is implemented for you by firebaseui).

- You should probably start with this overview video. Your assignment layout is a little different, but this will get you started. `https://youtu.be/Rlk2Nsqr1Bc`. Here is some good Android documentation describing how to use FirebaseUI, which helps us do authentication while writing a minimal amount of code. `https://firebase.google.com/docs/auth/android/firebaseui`. These links are on the course web page.

- The first thing you need to do is get a Firebase account and set up your account to communicate with your app. Follow the instructions at the following web page. `https://firebase.google.com/docs/android/setup`. That page mentions two methods for adding Firebase to your project: console and Android Studio assistant. It recommends the console, though I walk through using the assistant in the video. Here is a link to the console `https://console.firebase.google.com/`. When I redid this process in 2022, I used the console. I've found the assistant to be buggy and unreliable, but it worked in the video demo. Either method is fine.

- In step two from the instructions above, you should see the "add firebase to your Android app" screen (see Figure below). Here it is asking for your application ID and SHA-1 hash. The application ID is the applicationId in the module's build.grade, likely edu.utap.firebaseauth.


Add Firebase to your Android app

1 Register app

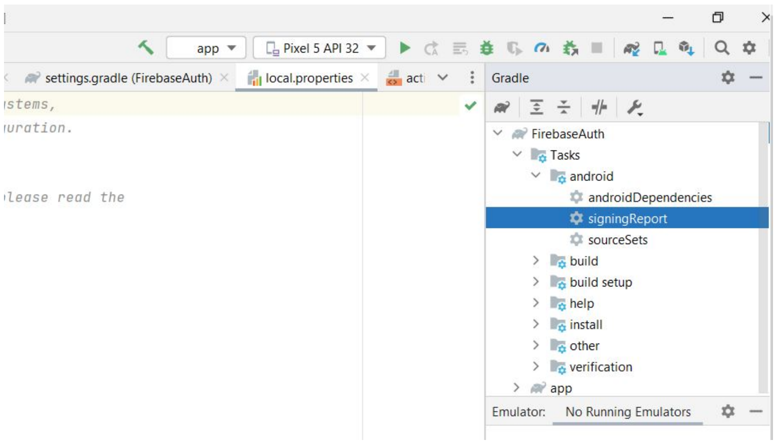
Android package name: edu.utap.firebaseauth

App nickname (optional): My Android App

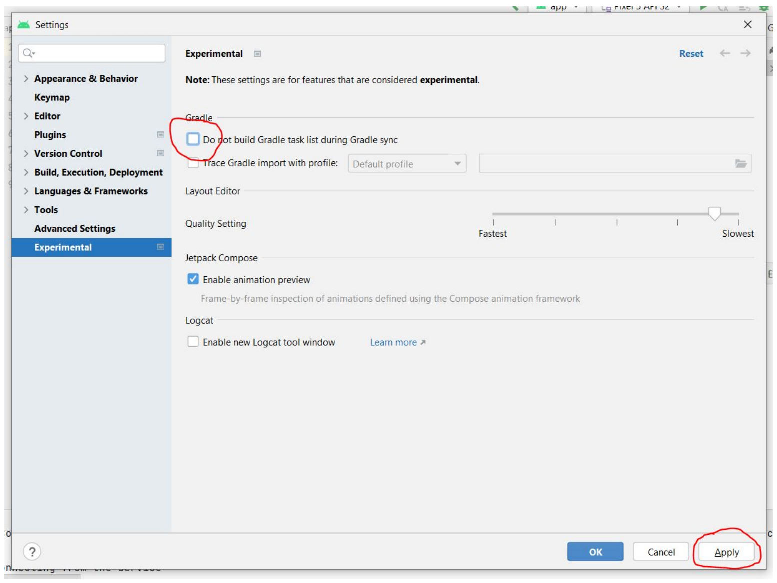
Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

- To get the SHA-1 hash of your project, you need to generate a signing report. Make sure your gradle panel is enabled (on the far right of Android Studio). Under FirebaseAuth – tasks – android you should run "gradle signingReport" either by typing it out or by double clicking it. Look for the output in your run console. Copy the string next to **SHA-1:** that is a sequence of two hex digits separated by colons.



- If you do not see "signingReport", go to File – Settings – Experimental in the Android Studio title bar. Uncheck the box indicated below and click apply (see figure below).



Experimental

Note: These settings are for features that are considered **experimental**.

Gradle

- [ ] Do not build Gradle task list during Gradle sync
- [ ] Trace Gradle import with profile: Default profile

Layout Editor

Quality Setting: Fastest – Slowest

Jetpack Compose

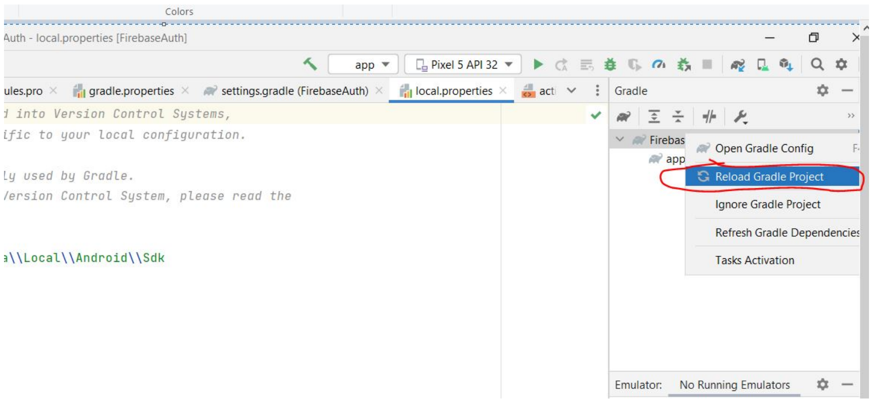
- [x] Enable animation preview

Frame-by-frame inspection of animations defined using the Compose animation framework

Logcat

- [ ] Enable new Logcat tool window — Learn more ↗

Then right click on your app in the gradle panel and "Reload Gradle Project". You should be able to generate a signing report now.

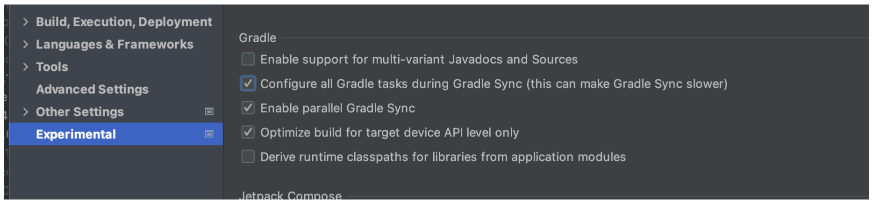


```
d into Version Control Systems,
ific to your local configuration.

ly used by Gradle.
Version Control System, please read the

a\\Local\\Android\\Sdk
```

- If you still don't see "signingReport" you might need to check "Configure all Gradle tasks during Gradle sync" in File – Settings – Experimental. You might then have to sync gradle again.



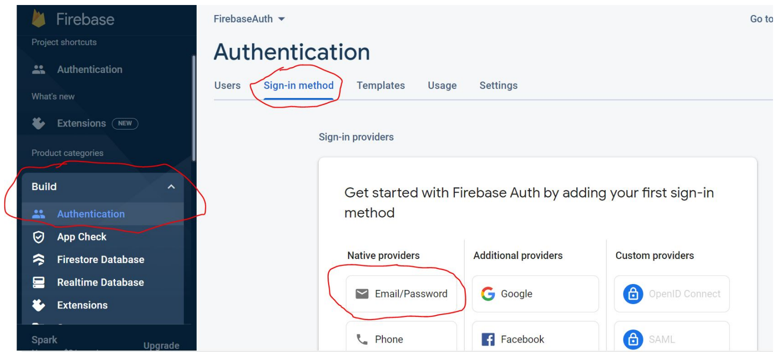
Gradle

- [ ] Enable support for multi-variant Javadocs and Sources
- [x] Configure all Gradle tasks during Gradle Sync (this can make Gradle Sync slower)
- [x] Enable parallel Gradle Sync
- [x] Optimize build for target device API level only
- [ ] Derive runtime classpaths for libraries from application modules

Jetpack Compose

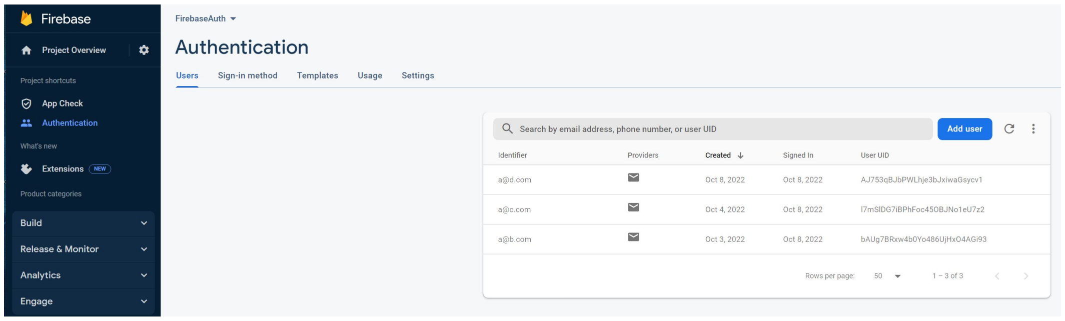
- As the web instructions say, download the google-services.json file and put it in your app directory. THIS IS VERY IMPORTANT! Without downloading this .json file, you cannot communicate with your firebase account. If you missed this step, click the gear in your project settings and scroll down under General, you will see a download button for google-services.json. You also should ENABLE email/password authentication in the firebase console.

  The instructions might say that you need to change your gradle file. You don't! Please do not.



Get started with Firebase Auth by adding your first sign-in method

- Test your app by adding a user to your firebase account with email address `fake@example.com` and give them a password of 123456. Once you have successfully logged in, you can also test new users. Then you can see the list of current users in the admin view in the firebase console for your project. Here you can manually delete users.



Identifier	Providers	Created	Signed In	User UID
a@d.com	✉	Oct 8, 2022	Oct 8, 2022	AJ753qBJbPWLhje3bJxiwaGsycv1
a@c.com	✉	Oct 4, 2022	Oct 8, 2022	l7mSlDG7iBPhFoc45OBJNo1eU7z2
a@b.com	✉	Oct 3, 2022	Oct 8, 2022	bAUg7BRxw4b0Yo486UjHxO4AGi93

Hint: You might get a runtime error along these lines
`Targeting S+ (version 31 and above) requires that one of FLAG_IMMUTABLE or FLAG_MUTABLE be specified when creating a PendingIntent.`
You can google this error and resolve it in different ways. We use `.setIsSmartLockEnabled(false)` which means we don't have to change our gradle file.

Just for reference, I have an older video here with has some additional detail, `https://youtu.be/q-GoOCHLW-E`. Here is a final older video that goes over adding users (which is covered in the previous videos). `https://youtu.be/8G4dDZVcbag`.

- **Gradle.** Communicating with Firebase will require you to modify your gradle file. We have set up your gradle file to be correct. The Firebase instructions will tell you to add stuff, but you don't need to!

- **AuthInit.kt** This class handles all the interactions between Firebase and the app. The general flow is to update in firebase, and if successful update in the view model. The view model locally caches the authentication information in firebase, which is the source truth.
  - **init**: Create and launch a sign-in intent using Firebase AuthUI. Be sure to pass in the providers, which is just email/password.
  - **setDisplayName**: The user has requested to update their display name. Get the current user from Firebase and update the user with the new display name. Then update the view model if the update in Firebase was successful.
  - This web page has code that will help you with managing users in Firebase. `https://firebase.google.com/docs/auth/android/manage-users`. You may find some useful code in here.

- **MainActivity.kt**
  - **onCreate**: You need to get information from the ViewModel to display in the UI. We have given you the onclick listeners declarations necessary for you to login, logout, and set display name. You fill in what goes inside the callback.
    - You need to populate `@+id/userName`, `@+id/userEmail`, and `@+id/userUid` appropriately. Remember, this is all driven by the view model. `@+id/displayNameET` is the EditText box that a user can edit to change their display name. Do NOT update the display name if this box is empty. The layout defines this identifier for your use.
  - If the user is already logged in, the login button should do nothing.
- **MainViewModel.kt**
  - **updateUser**: Fill in the details for updating user info. Hint: you need to use FirebaseAuth for something. Look in AuthInit.

```
When your login/set display name succeeds you should see this (for example)

Jo Q. Public
fake@example.com
Sdkfo48slkdfjDodfkslfiI
```

- **activity_main.xml** No need to do anything. Use the names of text views and buttons we've defined.

Submission. Recall the rules for flipped classrooms: do the work alone or with a partner that is unique for the semester. You can find the github link on piazza.

Only one student needs to submit this code, but both students may do so if they wish. This assignment should be submitted through github (sorry, partners can't share the repository). Fill in a README. We didn't include one for you, please write your own in the same format as all previous FCs.