Cosc 5/4735

Firebase

greenwich.edu.vn

UNIVERSITY of GREENWICH

Alliance with FPT Education

UNIVERSITY of GREENWICH
Alliance with FPT Education

What is firebase

- It's basically middle ware that allows you do a number of things cross platform
  - Analytics
    - is a free and unlimited analytics tool to help you get insight on app usage and user engagement. No extra code needed, only console.
  - Cloud Messaging (covered in another lecture)
    - Firebase Cloud Messaging lets you deliver and receive messages across platforms reliably.
  - Notifications
    - helps you re-engage with users at the right moment. No extra code needed, only console
  - Authentication
    - a key feature for protecting the data in your database and storage.
  - Realtime Database
    - lets you sync data across all clients in realtime and remains available when your app goes offline.
  - Cloud Firestore (new database, just out of beta)
    - Combines cloud database and functions together. Uses a scalable NoSQL cloud database to store and sync data.
  - Storage
    - lets you store and serve user-generated content, such as photos or videos. Firebase Storage is backed by Google Cloud Storage

UNIVERSITY of GREENWICH
Alliance with FPT Education

What is firebase (2)

- Functions
  - Run your mobile backend code without managing servers
- Hosting (not covered here)
  - provides fast and secure **static** hosting. No backend, but delivers .html,.css, etc files.
- Remote Config
  - change the behavior and appearance of your app without publishing an app update.
- App Indexing (not covered here)
  - With Firebase App Indexing, you can drive organic search traffic to your app, helping potential users of your app become your app's biggest fans.
- Dynamic Links (not covered here)
  - lets you pull users right to the content they were interested in, keeping them engaged and increasing the likelihood that they will continue to use the app.
- Invites (not covered here)
  - Firebase Invites helps your users share your app with others. referral codes or share content from the app. Uses the Dynamic Links so links survive the install of the app. And do most of the work automatically.
- AdWords (not covered here)
  - help you search potential users with online ads.
- AdMob (covered in another lecture)
  - provides easy and powerful ad monetization with full support in Firebase.

UNIVERSITY of GREENWICH
Alliance with FPT Education

What is firebase (3)

- Beta products
  - ML Kit
    - "use machine learning to solve problems." Currently, it is just the Vision APIs for text recognition, face detection, barcode scanning, image labeling, and landmark recognition. But this would use the cloud based systems instead of just the device cpu.
  - A/B Testing
    - Let you optimize your app experience based on analytics. Change it for different demographics, make it different experience for say a teenage group verses a retired group. Without having to deploy two version of the app. Needs remote config.
    - Deploy a new feature to a group and evaluate the effect of the changes on app use.
  - Predictions
    - Uses remote config, providing a custom experience based on each of your users' predicted behavior.
    - You can use Predictions with the Notifications composer to deliver the right message to the right user groups.

UNIVERSITY of GREENWICH
Alliance with FPT Education

Firebase console

- Google has a number of "consoles".
  - These are webpages to control varying things.
    - Other Examples:
      - https://partner.android.com/things/console#/ android things
      - https://console.developers.google.com/apis/ for apis like maps
      - https://developers.google.com/beacons/dashboard/ for beacons
- Firebase has it's own as well.
  - https://console.firebase.google.com/

UNIVERSITY of GREENWICH
Alliance with FPT Education

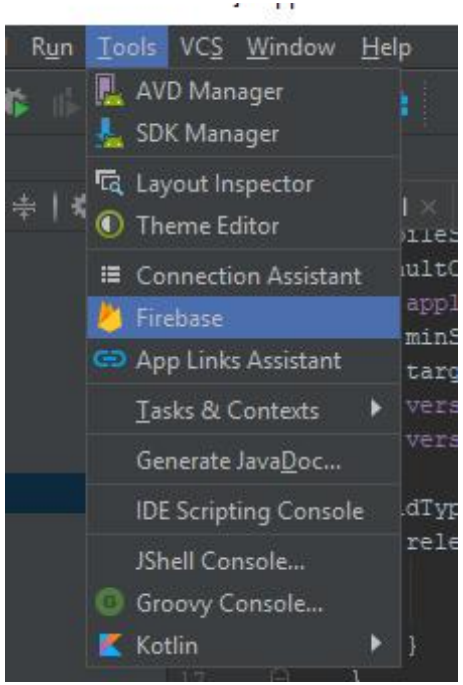
Firebase console (2)

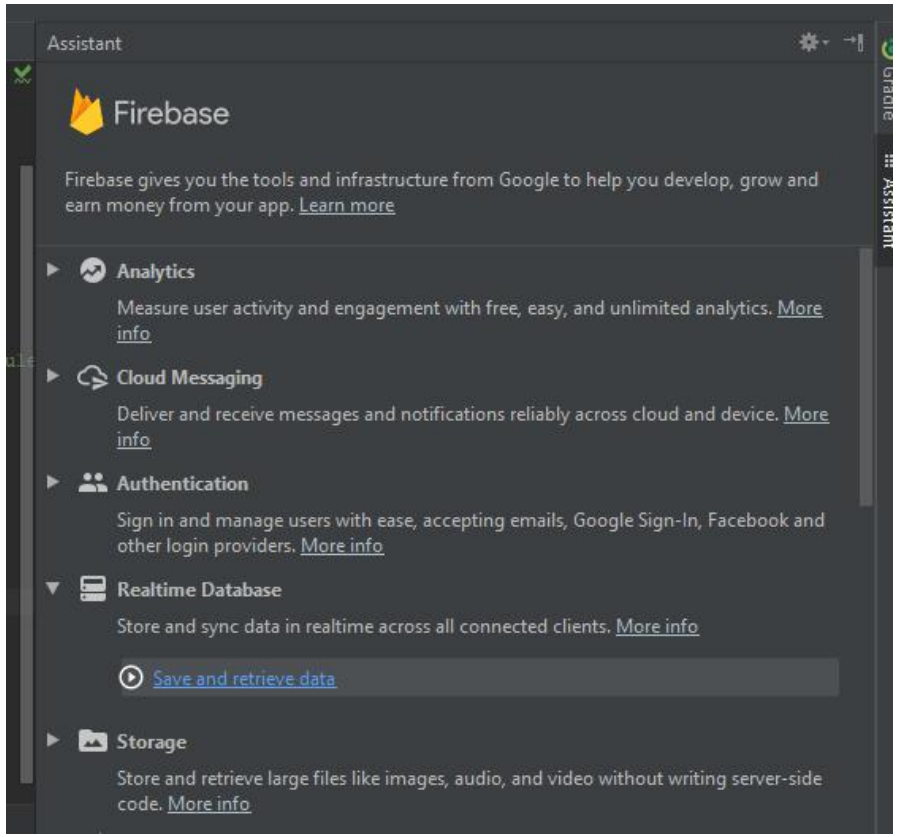
- You will need to log into the console
  - And setup a project. As primer guide, I'm skipping all this.
    - It can be done pretty easy with android studio.

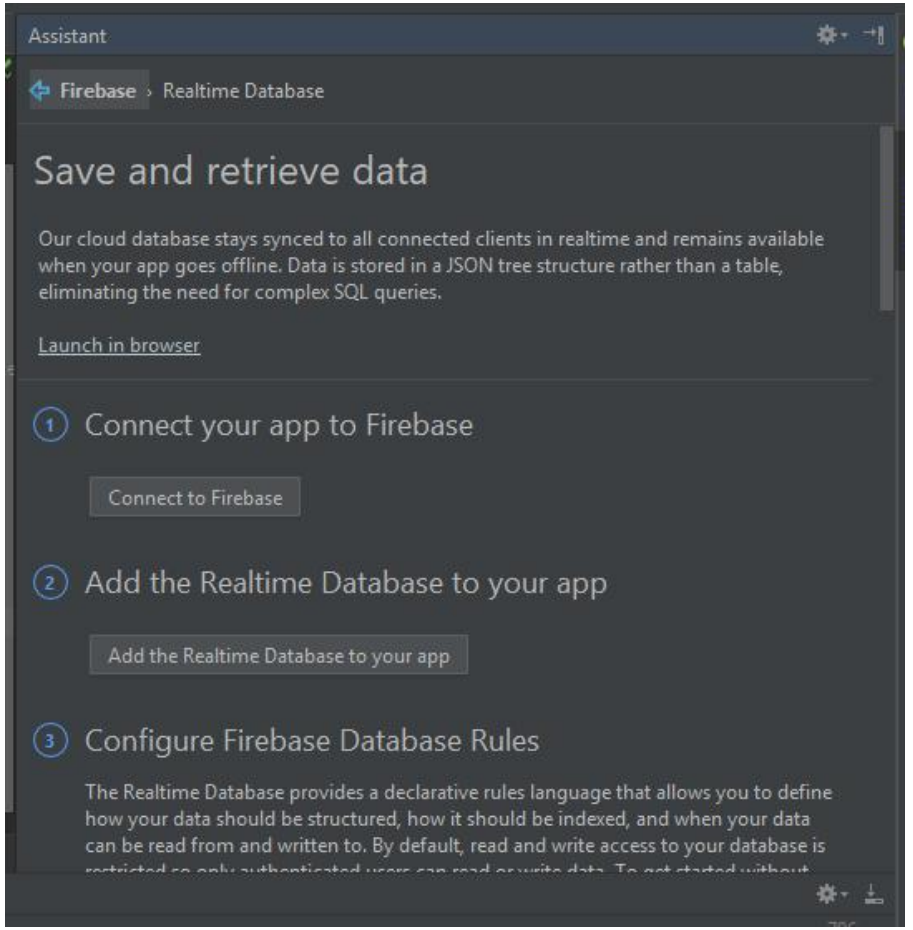
- Last part to remember, this is platform independent.
  - Can be used on IoS, Android, Unity, C++, and on the Web as well.

- Firebase versions will be listed in the following slides, but see https://firebase.google.com/docs/android/setup#available_libraries for the current versions.

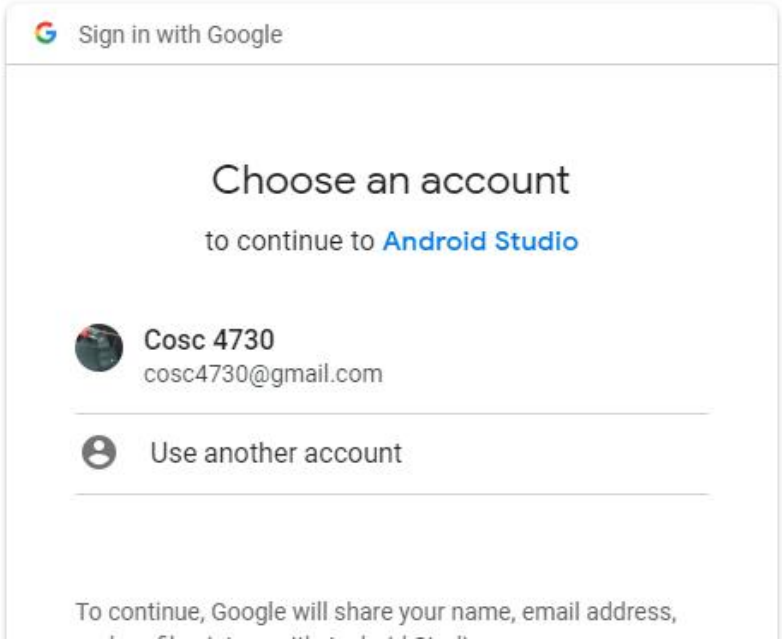
In Studio (this is setup the the Realtime Database)



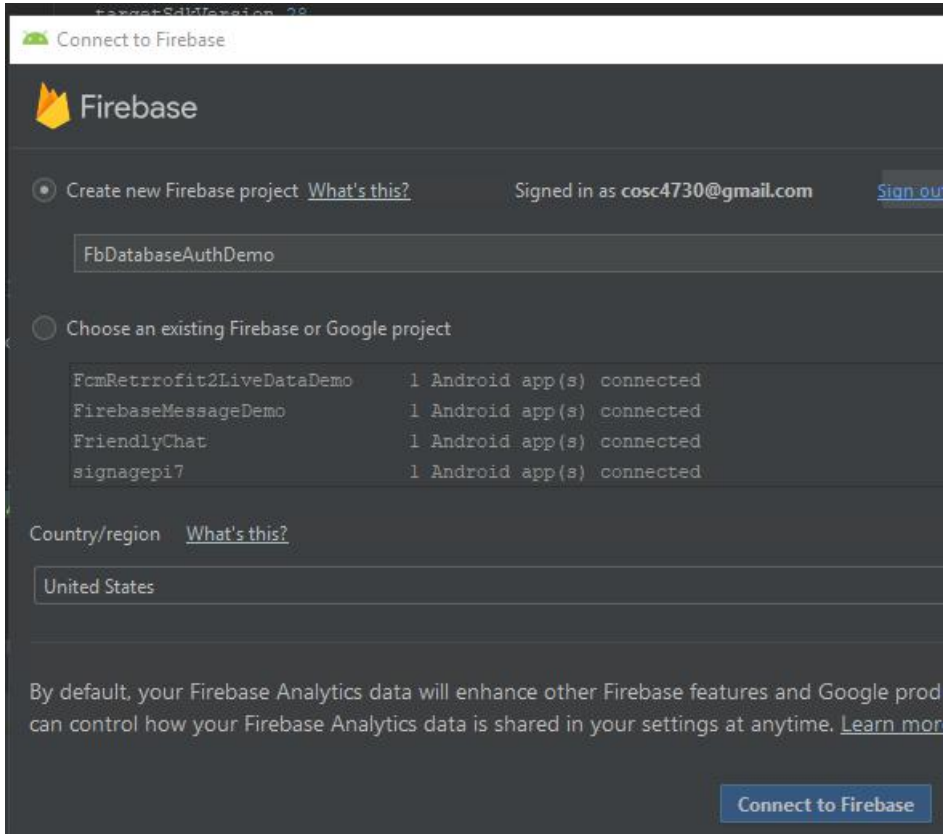


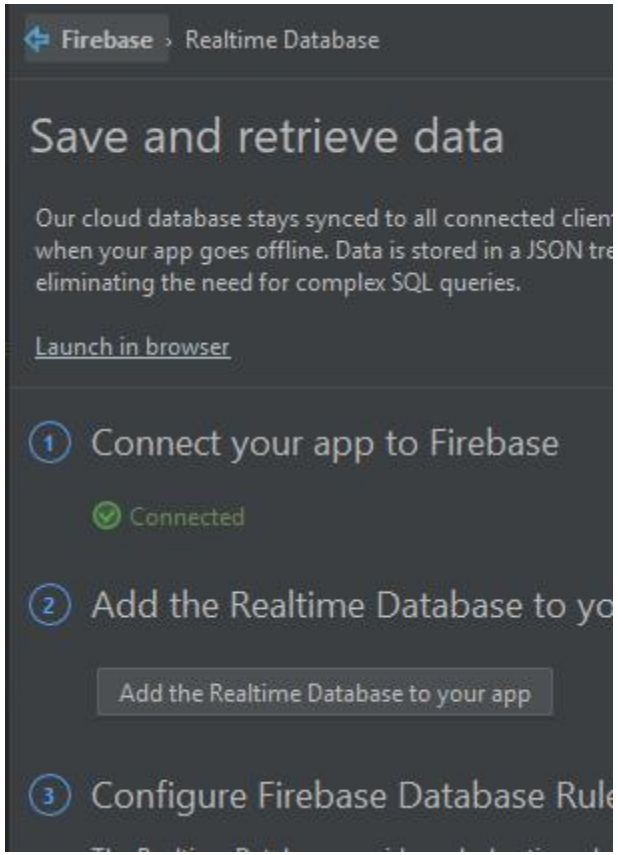


In the browser



Back in Studio

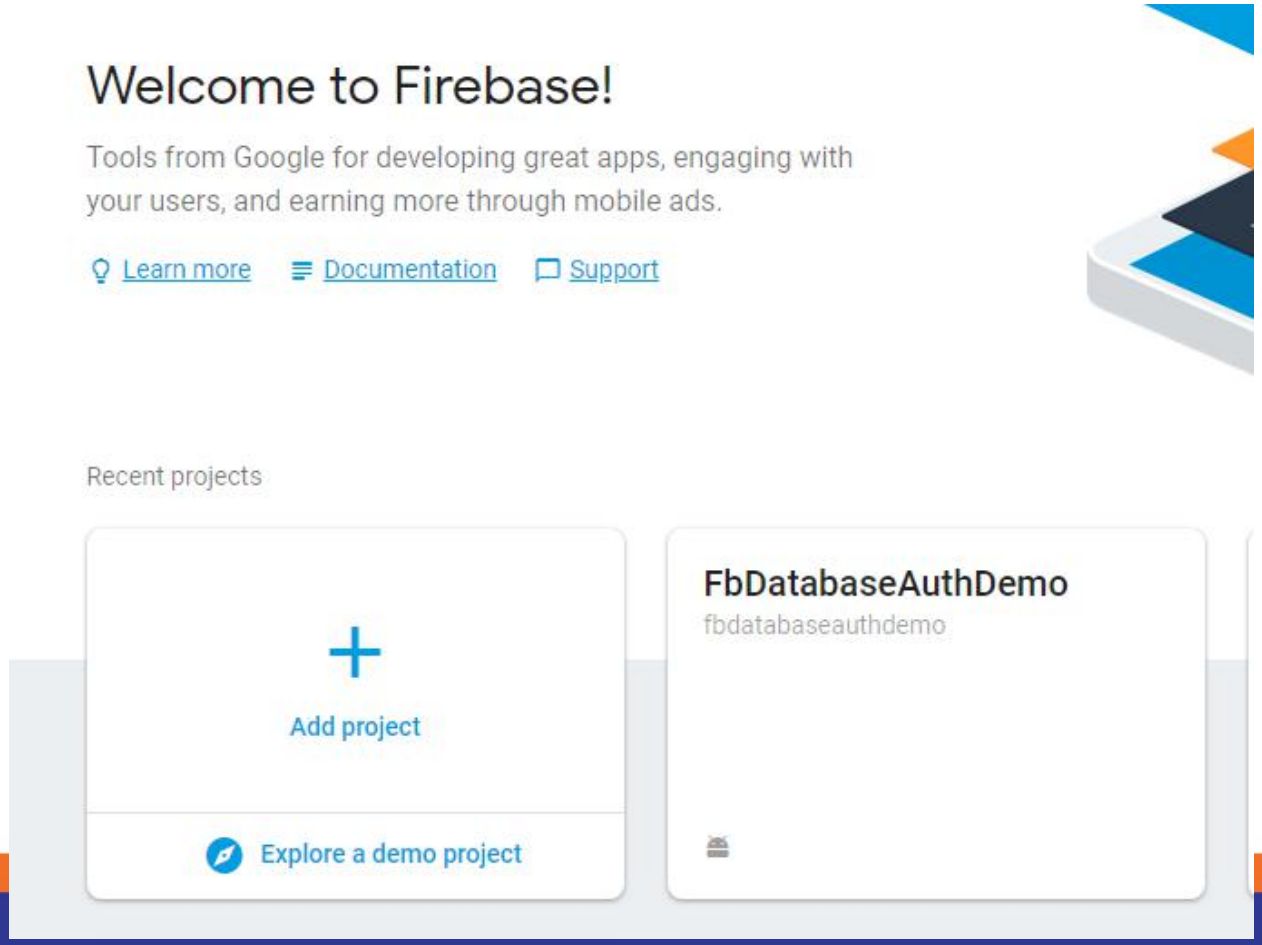




UNIVERSITY of GREENWICH
Alliance with FPT Education

Basic Setup (3)

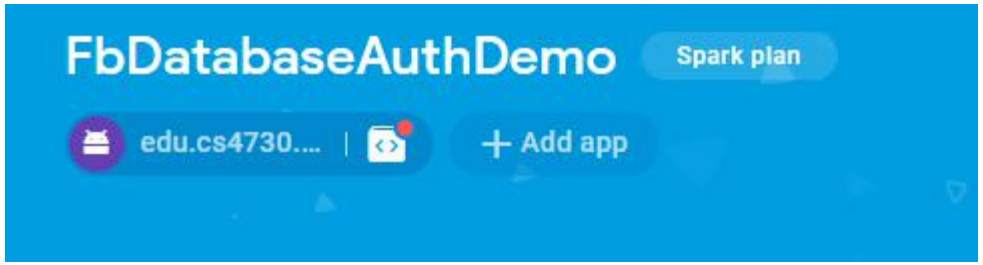
- You should now have the project in the firebase console as well

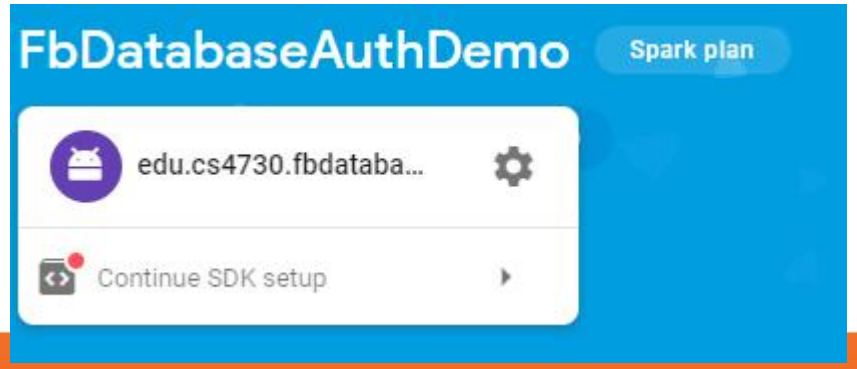


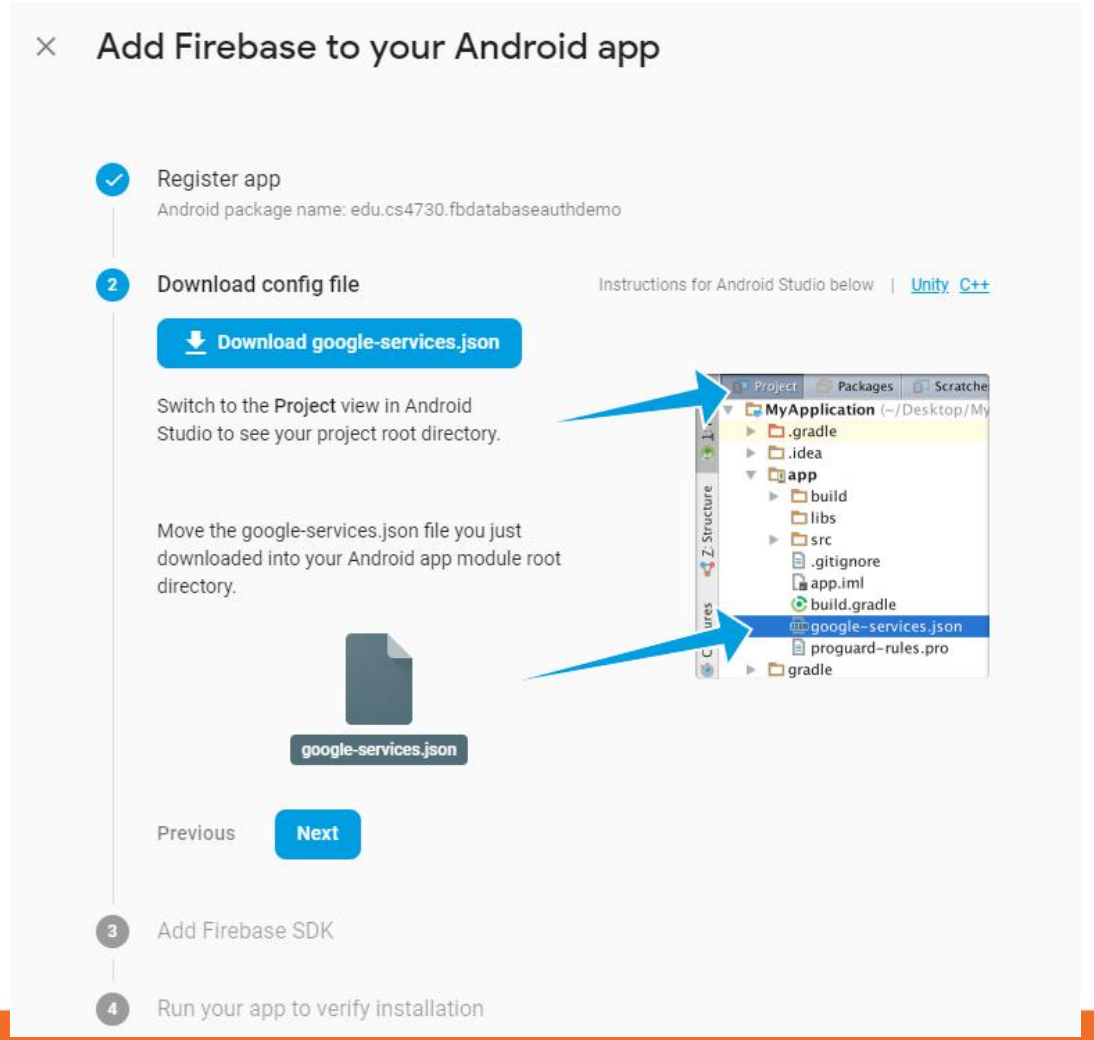
UNIVERSITY of GREENWICH
Alliance with FPT Education

Through the console only

- Create a project and
- add the app project



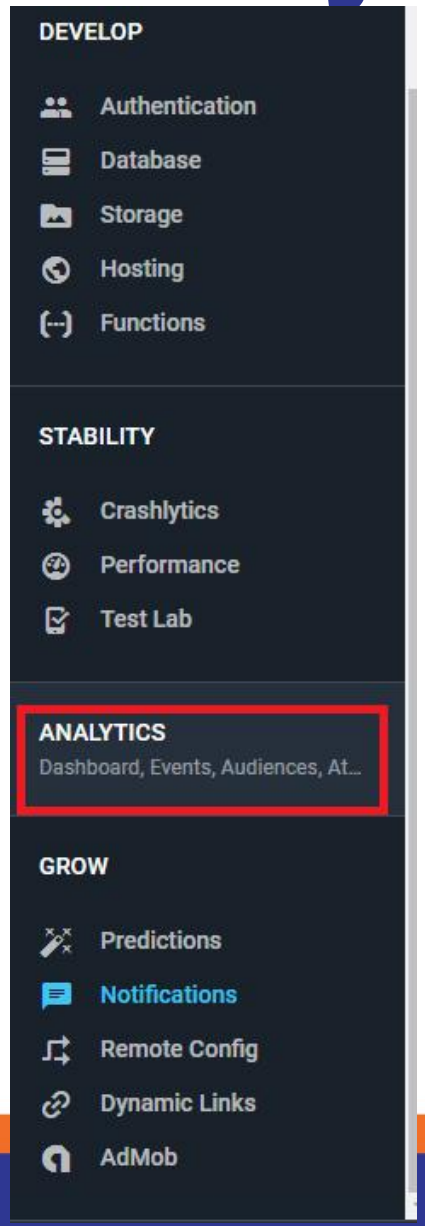




UNIVERSITY of GREENWICH
Alliance with FPT Education

Analytics

- You need to only add firebase to the dependencies.
  - implementation 'com.google.firebase:firebase-core:16.0.7'
  - The rest is done within the https://console.firebase.google.com/
  - You can look many different things, from first time usage, how long the app is up, demographics of users, etc.
  - Data updates once every 24 hours.



UNIVERSITY of GREENWICH
Alliance with FPT Education

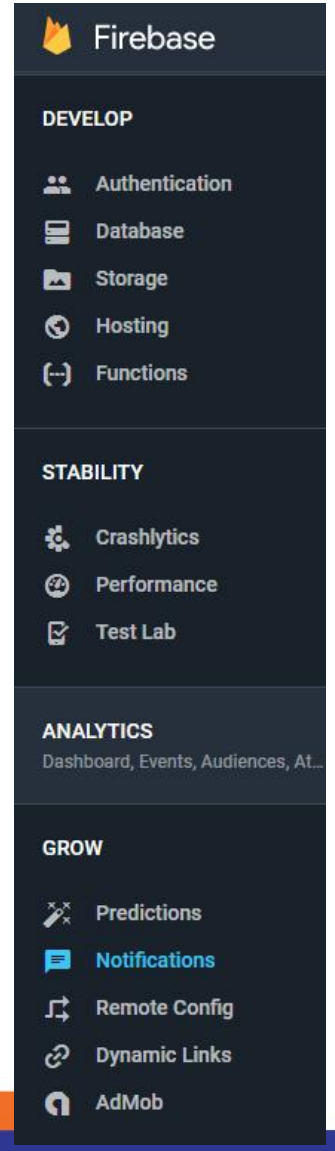
Notifications

- Not to be confused with cloud messaging
  - Covered in the next lecture.
- This allows you send a notification to the all (or some) of the devices where your app is instead.
  - You need on include firebase core and message in dependencies
    - implementation 'com.google.firebase:firebase-messaging:17.3.4'
- The rest is done in the console.

UNIVERSITY of GREENWICH
Alliance with FPT Education

Notifications (2)

- You create a message in the notifications section.
- Send the message.
- It will then show up on the device as a notification.
  - If they click the notification it will launch you app.
  - Note, if the app is up then the users won't see it (without extra code from cloud messaging).



UNIVERSITY of GREENWICH
Alliance with FPT Education

Realtime Database

- Is a cloud based database with live connections to the app.
  - The data is stored a json objects and is only intended for text, to allow for fast responses.
  - Authentication is needed.
  - Works even offline, with cached data then sync when online.
  - dependencies
    - implementation 'com.google.firebase:firebase-database:16.0.6'

UNIVERSITY of GREENWICH
Alliance with FPT Education

- The data is stored as a json object.
  - The L7W.. Is a unique key, that I didn't have to create.
  - You will need a POJO class that matches your data names

```
public class FriendlyMessage {

    private String text;
    private String name;
    private String photoUrl;

    public FriendlyMessage() {
    }

    public FriendlyMessage(String text, String name, String photoUrl) {
        this.text = text;
        this.name = name;
        this.photoUrl = photoUrl;
    }
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Database setup and use.

- Setup a connection and reference.
  - FirebaseDatabase database = FirebaseDatabase.getInstance();
  - DatabaseReference dbReference = database.getReference().child("messages");
- To add a new value use the push and setValue
  - dbReference.push().setValue(friendlyMessage);
    - Where friendlyMessage is a object holding the data.

UNIVERSITY of GREENWICH
Alliance with FPT Education

Database setup and use. (2)

- Listener for new data

```
ChildEventListener mChildEventListener = new ChildEventListener() {
    @Override
    public void onChildAdded(DataSnapshot dataSnapshot, String s) {
        FriendlyMessage friendlyMessage = dataSnapshot.getValue(FriendlyMessage.class);
      //now do something with the data
}
    public void onChildChanged(DataSnapshot dataSnapshot, String s) {          }
    public void onChildRemoved(DataSnapshot dataSnapshot) {          }
    public void onChildMoved(DataSnapshot dataSnapshot, String s) {          }
    public void onCancelled(DatabaseError databaseError) {          }
};
mMessagesDatabaseReference.addChildEventListener(mChildEventListener);
```

- Don’t forget to remove listeners in onPause, add them in onResume

UNIVERSITY of GREENWICH
Alliance with FPT Education

- Firebase Realtime Database allows nesting data up to 32 levels deep, but where possible avoid nesting deep.
- Also don't use json arrays.

```
{
  "users": {
    "alovelace": {
      "name": "Ada Lovelace",
      // Index Ada's groups in her profile
      "groups": {
         // the value here doesn't matter, jus
         "techpioneers": true,
         "womentechmakers": true
      }
    },
    ...
  },
  "groups": {
    "techpioneers": {
      "name": "Historical Tech Pioneers",
      "members": {
        "alovelace": true,
        "ghopper": true,
        "eclarke": true
      }
    },
    ...
  }
}
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Database and security

- By default only authenticated users can read and write the database.
- Console:

UNIVERSITY of GREENWICH
Alliance with FPT Education

Database and security (2)

- It can be changed, but not recommend.
- You can setup very complex rules, where only some users can access data, etc. With cascading rules for the depth of nodes.

- See the documentation:
  - https://firebase.google.com/docs/database/security/quickstart

UNIVERSITY of GREENWICH
Alliance with FPT Education

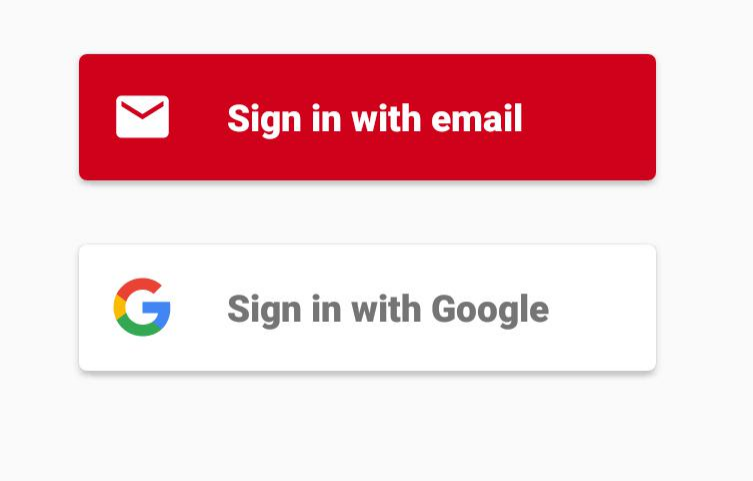
Authentication

- Allows the user to "login" to your app.
  - With any number of services using FirebaseUI
    - Currently: phone number, email and password, google, facebook, twitter, and github.
      - Note facebook and twitter require their api key in order for it to work.
    - You don't write the code, just call the method for sign in or sign out.
      - It keeps track, even the password for email.
  - dependencies
    - implementation 'com.google.firebase:firebase-auth:16.1.0'
    - implementation 'com.firebaseui:firebase-ui-auth:3.2.2'
      - See https://github.com/firebase/FirebaseUI-Android

UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use

- private FirebaseAuth mFirebaseAuth;
- private FirebaseAuth.AuthStateListener mAuthStateListener;
- mFirebaseAuth = FirebaseAuth.getInstance();



UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use (2)

- In onResume,

```
mFirebaseAuth.addAuthStateListener(mAuthStateListener);
```

- Where

```
mAuthStateListener = new FirebaseAuth.AuthStateListener() {
    @Override
    public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
        FirebaseUser user = firebaseAuth.getCurrentUser();
        if (user != null) {
            // User is signed in  use  user.getDisplayName() to get the name.
        } else {
            // User is signed out
            startActivityForResult(
                    AuthUI.getInstance()  //see firebase UI for documentation.
                            .createSignInIntentBuilder()  .setIsSmartLockEnabled(false)
                            .setAvailableProviders(Arrays.asList( new new AuthUI.IdpConfig.EmailBuilder().build(),
                                new AuthUI.IdpConfig.GoogleBuilder().build(),
                                new AuthUI.IdpConfig.PhoneBuilder().build()))
                            .build(),
                    RC_SIGN_IN);
        }
    }
};
```

- Again, don't forget to remove the listener in onPause()

```
mFirebaseAuth.removeAuthStateListener(mAuthStateListener);
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use (3)

- In onActivityResult

```
if (requestCode == MainActivity.RC_SIGN_IN) {
        if (resultCode == RESULT_OK) {
      mFirebaseUser = mFirebaseAuth.getCurrentUser();
      mUsername = mFirebaseUser.getDisplayName();
   } else {
        //failed login.
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Storage

- This allows for file storage in the cloud
  - It handles all the networking and syncing between cloud and device.
  - Again should use authentication.
  - Can specify any of rules on what the users are allowed to write to/read from the storage.

  - dependencies
    - implementation 'com.google.firebase:firebase-storage:16.0.5'

UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use.

- private FirebaseStorage mFirebaseStorage = FirebaseStorage.*getInstance*();
- private StorageReference mChatPhotosStorageReference = mFirebaseStorage.getReference().child("photos");
  - Where photos is directory in storage. See the console to create directories.

UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use (2)

- Add to storage

```
StorageReference photoRef = mPhotosStorageReference.child(filename);
```

  - Where filename is the name you want to the file to use in storage.

- Upload file to Firebase Storage

```
UploadTask uploadTask = photoRef.putFile(selectedImageUri);
```

- now we need the download url to add the db, but have do it with second task.

```
Task<Uri> urlTask = uploadTask.continueWithTask(new Continuation<UploadTask.TaskSnapshot, Task<Uri>>() {
    @Override
    public Task<Uri> then(@NonNull Task<UploadTask.TaskSnapshot> task) throws Exception {
        if (!task.isSuccessful()) {                    throw task.getException();                }
```

- Continue with the task to get the download URL

```
        return photoRef.getDownloadUrl();
    }
}).addOnCompleteListener(new OnCompleteListener<Uri>() {
    @Override
    public void onComplete(@NonNull Task<Uri> task) {
        if (task.isSuccessful()) {
            Uri downloadUri = task.getResult();
         } else {  // Handle failures}
    }
```

Download from

  - Use any method to download with a standard urll

UNIVERSITY of GREENWICH
Alliance with FPT Education

Security and authentication

- Storage by default doesn't have any security.
- It's suggested you use something like

FILES RULES

```
1  service firebase.storage {
2    match /b/{bucket}/o {
3      match /{allPaths=**} {
4        allow read: if request.auth != null
5        allow write: if request.auth != null && request.resource.size < 3 * 1024 * 1024;
6      }
7    }
8  }
```

https://firebase.google.com/docs/storage/security/start

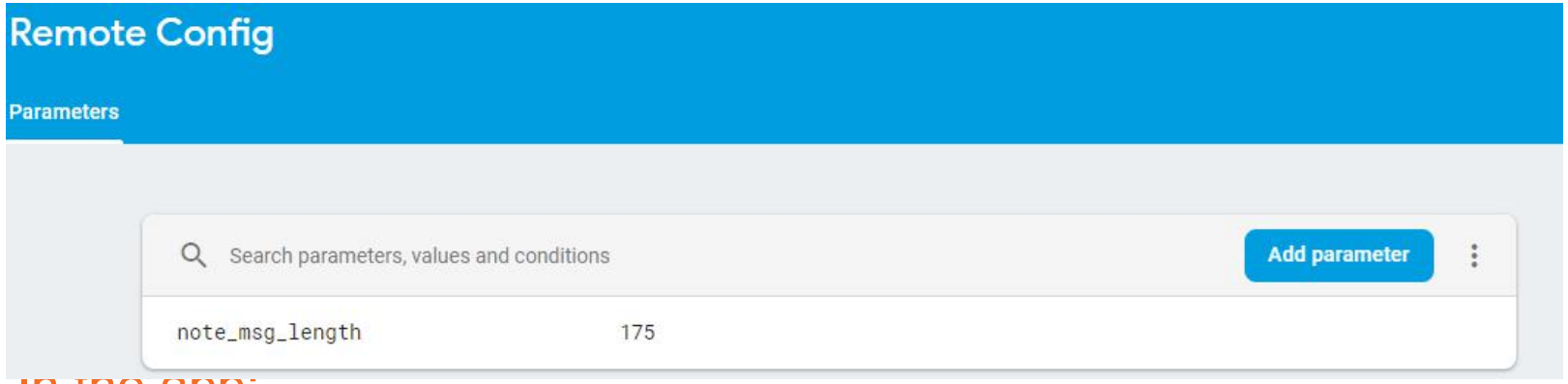
UNIVERSITY of GREENWICH
Alliance with FPT Education

Remote Config

- Allows you to put some variables in the cloud
  - This allows you change them in the console and push them to the app.
    - You can push different values to groups of devices as well. (maybe based on what you see in the analytics)
  - dependencies
    - implementation 'com.google.firebase:firebase-config:16.3.0'

Setup and use.

- First in the console add the parameters and values



- In the app:
  - Have default value for all parameters
  - private FirebaseRemoteConfig mFirebaseRemoteConfig = FirebaseRemoteConfig.*getInstance*();
    FirebaseRemoteConfigSettings configSettings = new FirebaseRemoteConfigSettings.Builder()
    .setDeveloperModeEnabled(BuildConfig.*DEBUG*)
    .build();
    mFirebaseRemoteConfig.setConfigSettings(configSettings);

UNIVERSITY of GREENWICH
Alliance with FPT Education

Setup and use (2)

- Define default config values. Defaults are used when fetched config values are not available. Eg: if an error occurred fetching values from the server.

```
Map<String, Object> defaultConfigMap = new HashMap<>();
defaultConfigMap.put(NOTE_LENGTH_KEY, DEFAULT_MSG_LENGTH_LIMIT);
mFirebaseRemoteConfig.setDefaults(defaultConfigMap);
```

- Now fetch the parameters.

```
mFirebaseRemoteConfig.fetch(cacheExpiration)
        .addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {        mFirebaseRemoteConfig.activateFetched();}
        })
        .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                // An error occurred when fetching the config.
                Log.w(TAG, "Error fetching config", e);
            }
        });
```

- Somewhere else called likely in success and failure (remember defaults!)

```
long note_msg_length = mFirebaseRemoteConfig.getLong(NOTE_LENGTH_KEY);
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Functions

- You can write the "functions" you need (via Node.js) and use them on google cloud systems.
- Low maintenance and it keeps your logic private and secure (not within the app itself)
- Allows for you to use more cpu power then a device has.
- These can connect to storage and database.
  - https://www.youtube.com/watch?v=bpFAdhNkA6c
  - You will need Node.js, npm (included in node), and firebase-tools. This all command line, not studio. See https://firebase.google.com/docs/functions/get-started

UNIVERSITY of GREENWICH
Alliance with FPT Education

Using Google Authentication with firebase

- If you app already uses google authentication, (ie requires a google signin)
  - Then you can use that to login to firebase as well.
- First sign in with Google Authentication
- Pass the credentials off to firebase.

– https://developers.google.com/identity/sign-in/android/sign-in

UNIVERSITY of GREENWICH
Alliance with FPT Education

Google SignIn

- In FireBase UI, we let it do the heavy lifting.
- Here we will do it manually.

```
 GoogleSignInOptions gso = new
GoogleSignInOptions.Builder
( GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken(getString(R.string.default_web_client_id))
            .requestEmail()
            .build();
```

- get the sign in client and start the activity for it.

```
mGoogleSignInClient = GoogleSignIn.getClient(getContext(), gso);
Intent signInIntent = mGoogleSignInClient.getSignInIntent();
startActivityForResult(signInIntent, MainActivity.RC_G_SIGN_IN);
```

Google SignIn (2)

- In onActivityResult

```
 if (requestCode == MainActivity.RC_G_SIGN_IN) {
            Task<GoogleSignInAccount> task =
GoogleSignIn.getSignedInAccountFromIntent(data);
            try {
                GoogleSignInAccount account = task.getResult(ApiException.class);
                // Google Sign In was successful, authenticate with Firebase
                firebaseAuthWithGoogle(account);
            } catch (ApiException e) {
            //failed to sign in.
        }
          }
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

firebaseAuthWithGoogle part.

- Get the credential from the account.

```
AuthCredential credential = GoogleAuthProvider.getCredential(acct.getIdToken(), null);
```

- Now sign in to the firebase.

```
mFirebaseAuth.signInWithCredential(credential)
 .addOnCompleteListener(getActivity(), new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
```

  - 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.w(TAG, "signInWithCredential", task.getException());
            } else {
```

  - SignIn Success"

```
                mFirebaseUser = mFirebaseAuth.getCurrentUser();
                mUsername = mFirebaseUser.getDisplayName();
           }    }   });
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

FireStone Database.

- Just released from Beta, Dec 2018.
- For flexibity, query the database, and scalable.
  - Like realtime db, realtime updates and offline support.
- Uses a NoSQL data model.
- You have a collections, which contain documents, which can contain subcollections to build hierarchical data structures.
- Allows for efficient questions, by creating shallow queries or nested questions.
- https://firebase.google.com/docs/firestore/

UNIVERSITY of GREENWICH
Alliance with FPT Education

Getting started.

- In the console turn on firestone.
- First add firestone to gradle
  - implementation 'com.google.firebase:firebase-firestore:18.0.1'
  - implementation 'com.google.firebase:firebase-core:16.0.7'
- Access your database

```
FirebaseFirestore db = FirebaseFirestore.getInstance();
```

- Adding data, create a HashMap

```
Map<String, Object> data = new HashMap<>();
data.put("back in 5 minutes", et_msg.getText().toString());
data.put("Arrow", "none");
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

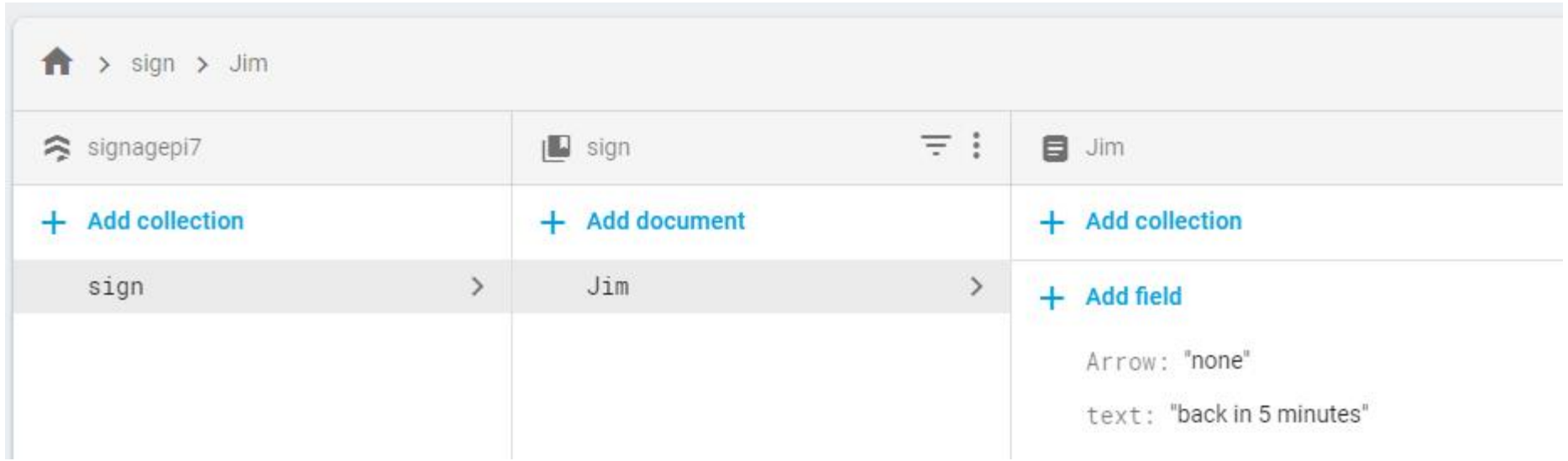
Adding data.

- Now Add the data to the level you want it. In this case, /sign/jim/

```
db.collection("sign").document("Jim")
        .set(data)  //hashmap previously created.
        .addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void aVoid) {
                Log.d(TAG, "DocumentSnapshot successfully written!");
            }
        })
        .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Log.w(TAG, "Error writing document", e);
            }
        });
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

In the console it looks like this



UNIVERSITY of GREENWICH
Alliance with FPT Education

Receive the data

```
db.collection("users").document("Jim").get()
        .addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {
            @Override
            public void onComplete(@NonNull Task<DocumentSnapshot> task) {
                if (task.isSuccessful()) {
                    DocumentSnapshot snapshot = task.getResult();
                     Map<String, Object> data = snapshot.getData();
                     TestText = String.valueOf(data.get("text"));
                     Arr = String.valueOf(data.get("Arrow"));
                }
            }
        });
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

To get multiple documents from a collection

```
db.collection("users")
        .get()
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {  //iterate over the documents.
                    for (QueryDocumentSnapshot document : task.getResult()) {
                 //using document.getData() to get each one.
                    }
                } else {
                    Log.w(TAG, "Error getting documents.", task.getException());
                }
            }
        });
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Retrieve via a listener

```
DocumentReference docRef = db.collection("sign").document("Jim");
        docRef.addSnapshotListener(new EventListener<DocumentSnapshot>() {
            @Override
            public void onEvent(@Nullable DocumentSnapshot snapshot, @Nullable
FirebaseFirestoreException e) {
                if (e != null) {  Log.w(TAG, "Listen failed.", e);    return;          }
                //retrieve the data if it exists.
                if (snapshot != null && snapshot.exists()) {
                    Map<String, Object> data = snapshot.getData();
                    TestText = String.valueOf(data.get("text"));
                     Arr = String.valueOf(data.get("Arrow"));
                  }
                }        });
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Example code

- The FbDatabaseAuthDemo
  - Has code to firebase sign example
  - DBlistFragment and DBSimpleFragment use RealTime database example.
  - RCFragment has remoteconfig
  - StorageFragment has all the code for storage
  - MessingService will catch notification message (not cloud) even when the app is running.
  - InviteAntFragment is an attempt at invites, but doesn't seem to work.

UNIVERSITY of GREENWICH
Alliance with FPT Education

References

- https://console.firebase.google.com/
- https://firebase.google.com/docs/
  - Many of the sub doc's where listed on slides.

- Code lab: FriendlyChat app (about 2ish hours)
  - https://codelabs.developers.google.com/codelabs/firebase-android/#0
    - Covers database, auth, invites, remote config, admob, analytics, and firebase notifications.
- A short course (about 8 hours) Firebase in a weekend
  - https://classroom.udacity.com/courses/ud0352 (free course)
    - It's dated, but still pretty good.

UNIVERSITY of GREENWICH

Alliance with FPT Education

Q&A