# CMSC436: Programming Handheld Systems

# Introduction to Firebase

# Today’s Topics

- Google’s Firebase Platform
- Adding services to your android application
  - User Authentication
  - Realtime Database

# Firebase

**Firebase** is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, the Firebase platform has 18 products, which are used by 1.5 million apps - wikipedia

Example of mobile backend as a service design pattern

Similar to AWS for web applications

# Today’s Focus

Adding Firebase to an Android App

Firebase Email Authentication Service

Firebase Realtime Database

# Adding Firebase to an Android App

2 choices:

- manually add firebase through the firebase console (https://console.firebase.google.com/)

- directly from inside Android Studio (select Tools -> Firebase from the menu)

- second option requires you to select a service to add to your app

- Let’s add firebase authentication to an app

# Firebase Email/Password Authentication Service

What it does:

- Allows users to sign up with email and password
- Stores users login information separate from database
- Activities gain access to currently logged in user via api
- Combined with security rules can restrict data access to only the data of the logged in user
- (Note: this requires the data be structured in a particular pattern – more on this later)

API Reference:
https://firebase.google.com/docs/reference/android/com/google/firebase/auth/FirebaseAuth.html

# Firebase Email/Password Authentication Service

We’ve already added authentication via the Firebase assistant in Android Studio

Final Step – select Sign-in method in Firebase console and enable Email/Password under provider heading

# Firebase Email/Password Authentication Service

Let’s take a look at a sample app

Example: FirebaseEmailAuthExample

# Firebase Email/Password Authentication Service

You can view all of your users for your application from the firbase console

Let’s take a look and see who we have added

# Firebase Realtime Database

Firebase offers 2 types of database services

- Realtime Database (original)
- Cloud Firestore (Newer service)
- Differences between the 2 options
- https://firebase.google.com/docs/database/rtdb-vs-firestore?authuser=0

# Firebase Realtime Database

Our focus will be the original realtime database

What is it?

It's an efficient, low-latency solution for mobile apps that require synced states across clients in realtime.

# Firebase Realtime Database

How to add Realtime Database to your app:

Add the dependency for Realtime Database to your app-level build.gradle file

implementation 'com.google.firebase:firebase-database:19.1.0’

Configure Realtime Database Rules

https://firebase.google.com/docs/database/android/start?authuser=0

# Firebase Realtime Database

Let’s take a look at the security rules for our example app in the firebase console

Example Rules with no auth:

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

# Firebase Realtime Database

Let’s take a look at the security rules for our example app in the firebase console

Example Rules with  auth (note: this requires authentication to have been enabled to be useful)

```
{
  "rules": {
     ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

# Firebase Realtime Database

Realtime Database has a REST API

You can use any Firebase Database URL as a REST endpoint. All you need to do is append .json to the end of the URL and send a request from your favorite HTTPS client

Let’s see an example using Postman

https://firebase.google.com/docs/reference/rest/database

# Firebase Realtime Database

How is Data Structured?

JSON tree

When you add data to the JSON tree, it becomes a node in the existing JSON structure with an associated key

Let’s take a look

# Firebase Realtime Database

## Data Structure Best Practices

Avoid nesting data

Why?

Iterating through the data becomes problematic

Any data request returns the entire tree

# Firebase Realtime Database

## Data Structure Best Practices

Flatten the data structure

Split data into separate paths (also called denormalization)

Can efficiently download data in separate calls as needed

# Firebase Realtime Database

Reading and writing data:

https://firebase.google.com/docs/database/android/read-and-write?authuser=0

# Firebase Realtime Database

Working with Lists of Data:

https://firebase.google.com/docs/database/android/lists-of-data?authuser=0

# Firebase Realtime Database

Let’s look at a sample app:

MyHomeLibrary

Allows you to add the authors in your home library and corresponding titles

# Firebase Realtime Database

Let’s look at a sample app:

MyHomeLibrary

How can we make it better?

# Firebase Realtime Database

API Reference:

https://firebase.google.com/docs/reference/android/com/google/firebase/database/DatabaseReference.html

# Firebase

Sample Code Covered Today:

FirebaseRealtimeDatabaseExample

FirebaseEmailAuthExample