

Android

With Firebase

Serverless

With firebase

What is Serverless?

- Serverless is a new paradigm of computing that abstracts away the complexity associated with managing servers for mobile and API backends, ETL, data processing jobs, databases, and more.
- No upfront provisioning - Just provide your code and data, and Google dynamically provisions resources as needed.
- No management of servers - Get out of the repetitive and error-prone task of managing or automating server management like scaling your cluster, OS security patches, etc.
- Pay-for-what-you-use - Because of the dynamic provisioning and automatic scaling, you only pay for what you use.

Why Serverless

- Applications with rapid time-to-market and unpredictable scale requirements benefit the most from Serverless. Here are some benefits experienced by Google Cloud Customers:
- Time-To-Market Improvement - Infrastructure management takes time, so eliminating it means you can get new code to production faster.
- Infrastructure Cost Reduction - Paying only for what you use means lower costs.
- Ops Cost Reduction - Automating repetitive provisioning and management tasks means you get to do higher-value devops tasks.

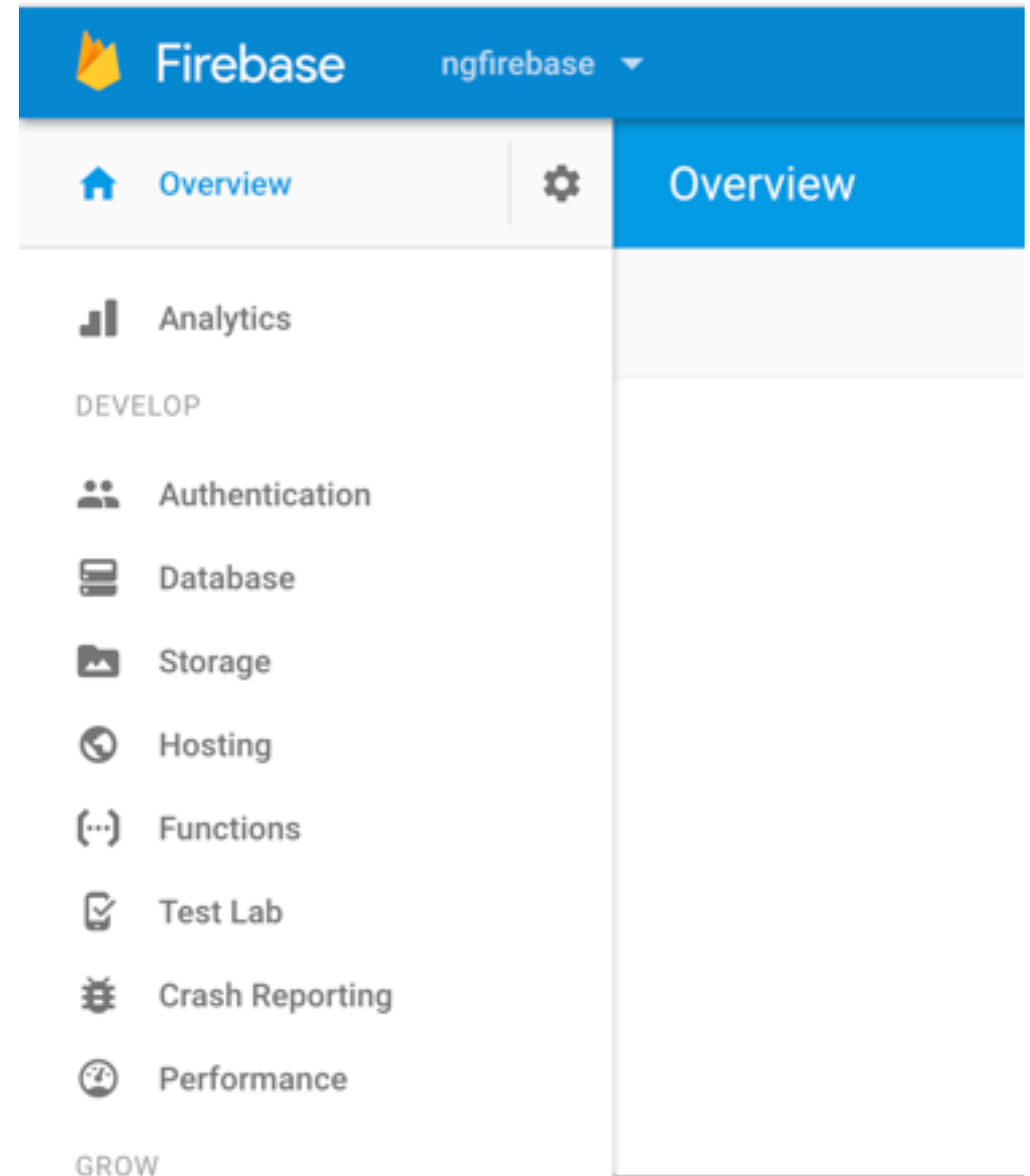
Serverless - Microservices Built Right

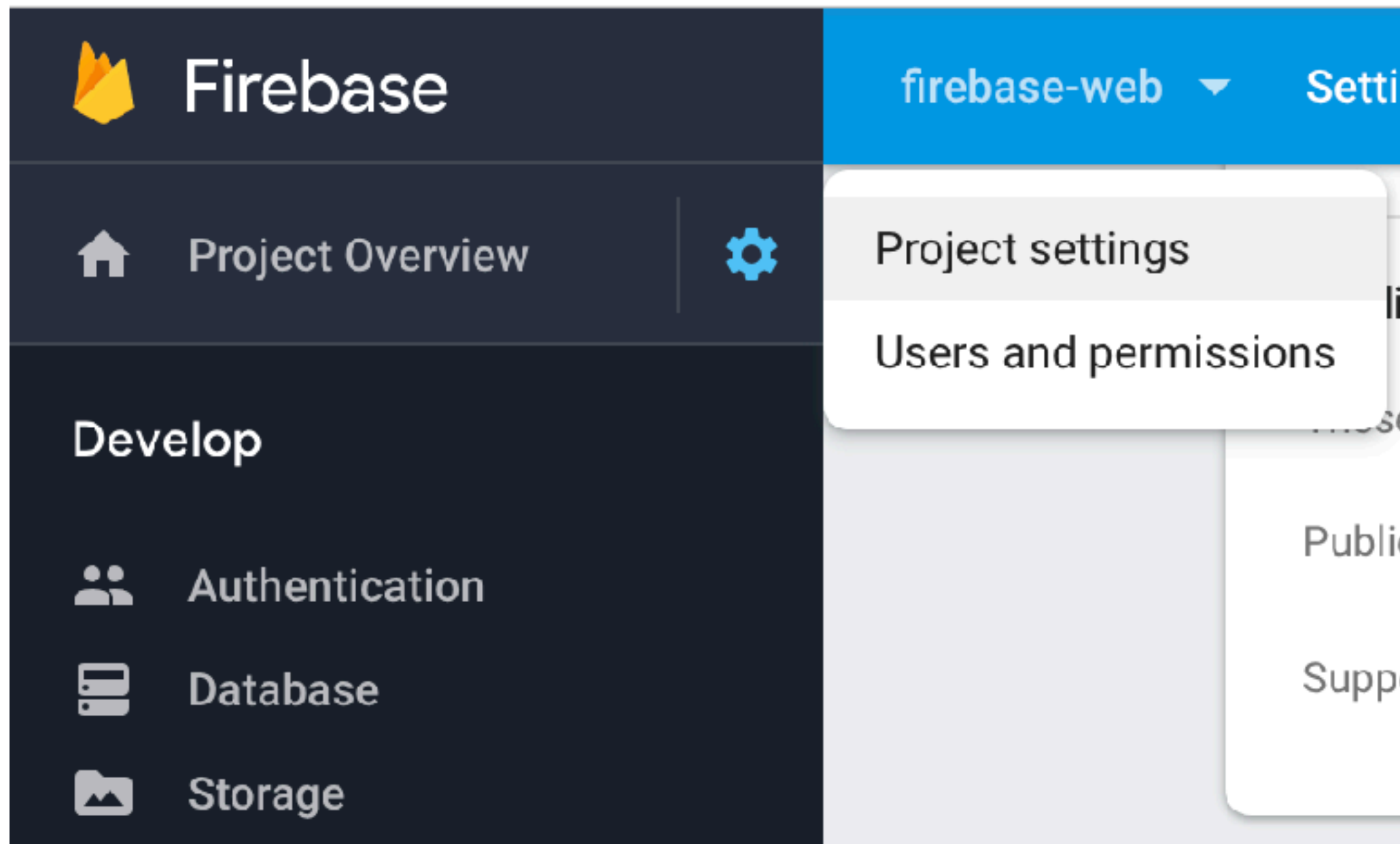
Properly designed microservices have a single responsibility and can independently scale. With traditional applications being broken up into 100s of microservices, traditional platform technologies can lead to significant increase in management and infrastructure costs. Google Cloud Platform's serverless products mitigates these challenges and help you create cost-effective microservices.

Firebase setup

Create firebase project

- បើក url <https://firebase.google.com/>
- ចុចលើ GO TO CONSOLE
- ចុចលើ Add project
- បញ្ចូល Project name
- ចុច Create project





ตั้งค่า project

กดที่ Project Overview / Project settings

There are currently no apps in the project firebase-web



Add Firebase to
your iOS app



Add Firebase to
your Android app



Add Firebase to
your web app

เลือก android

เพื่อตั้งค่าให้ android สามารถเชื่อมกับ firebase ได้

Android package name ⓘ

com.company.appname

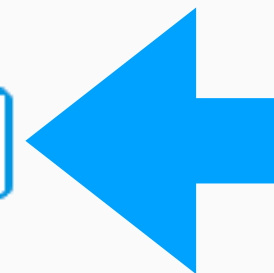
App nickname (optional) ⓘ

Freemium Android App

Debug signing certificate SHA-1 (optional) ⓘ

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

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




ระบุชื่อ package

ของ android ที่เราจะเชื่อมต่อกับ

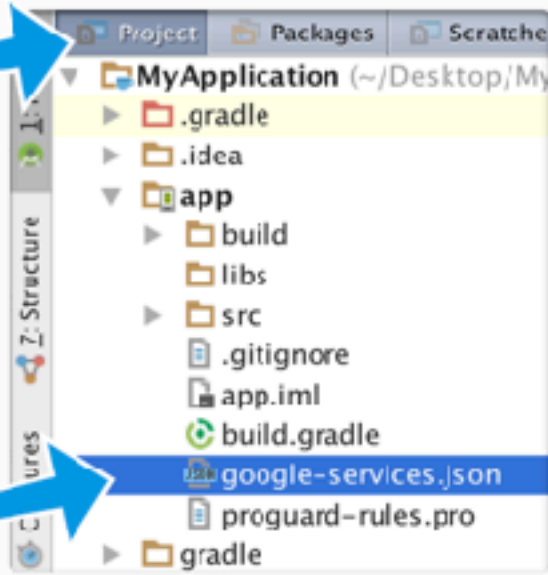
2 Download config file

Instructions for Android Studio below | [Unity](#) [C++](#)

 Download google-services.json

Switch to the **Project** view in Android Studio to see your project root directory.

Move the google-services.json file you just downloaded into your Android app module root directory.



Previous **Next**

กดปุ่ม download google-service.json

แล้วนำ file ไปวางไว้ใน project ดังภาพ

downloaded. Modify your build.gradle files to use the plugin.

Project-level build.gradle (<project>/build.gradle):

```
buildscript {  
    dependencies {  
        // Add this line  
        classpath 'com.google.gms:google-services:4.0.1'  
    }  
}
```



App-level build.gradle (<project>/<app-module>/build.gradle):

```
dependencies {  
    // Add this line  
    implementation 'com.google.firebase:firebase-core:16.0.1'  
}  
...  
// Add to the bottom of the file  
apply plugin: 'com.google.gms.google-services'
```



Includes Analytics by default (?)

Finally, press "Sync now" in the bar that appears in the IDE:

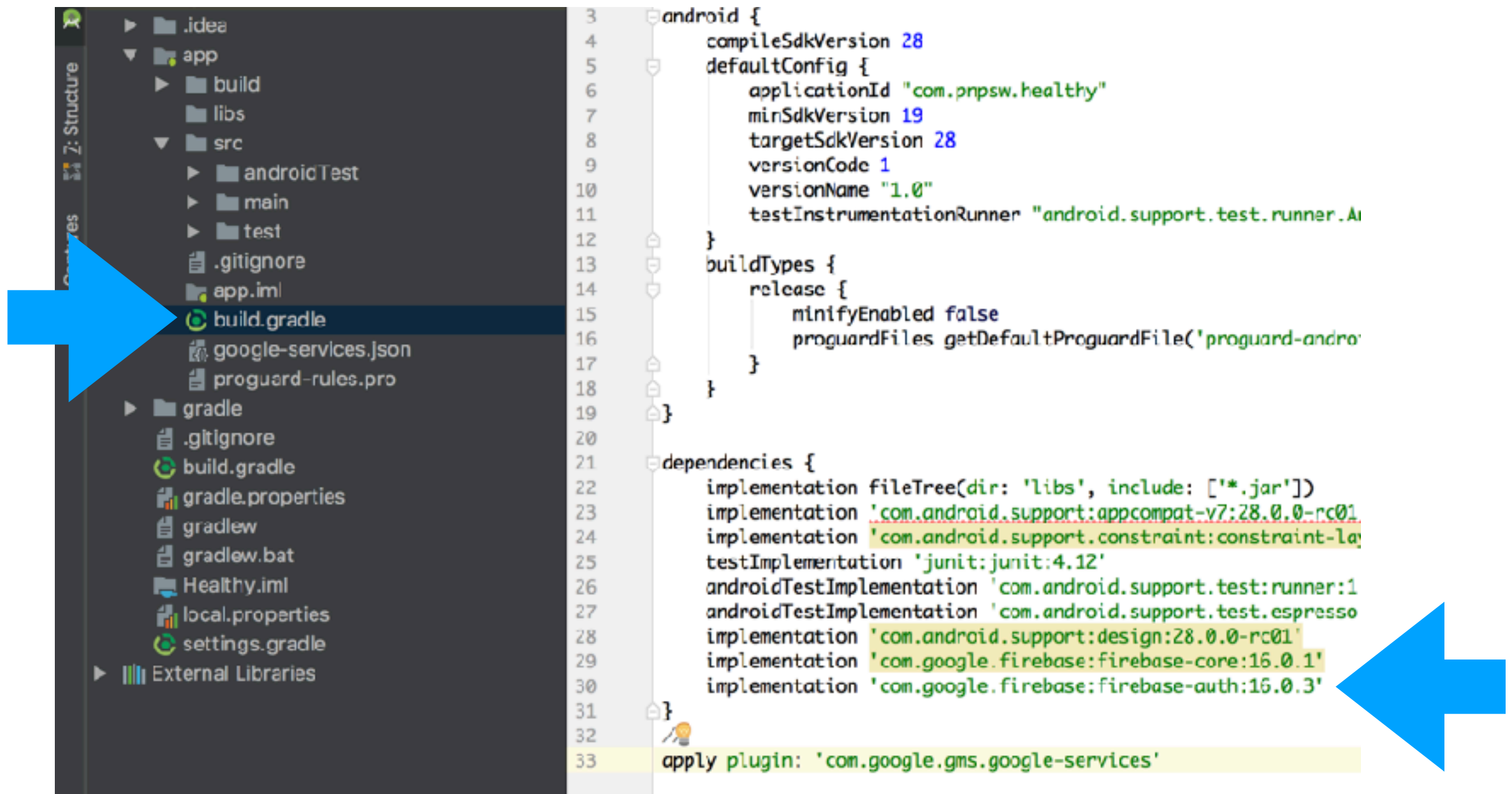
Gradle files have changed since

[Sync now](#)

ตั้งค่า classpath

เข้าไปทั้งสอง file แล้วกดปุ่ม sync now


Firebase Auth



เพิ่ม lib

com.google.firebase: firebase-auth:16.0.3

เข้าไปที่ file build.gradle

 **Firebase**

ngfirebase ▾

🏠 Overview

⚙️

📊 Analytics

DEVELOP

👤 Authentication

🗄 Database

🖼 Storage

🌐 Hosting

⌘ Functions

📝 Test Lab

🐛 Crash Reporting

📈 Performance

GROW

💬 Notifications

🔮 Predictions

Authentication

USERSSIGN-IN METHODTEMPLATESUSAGE

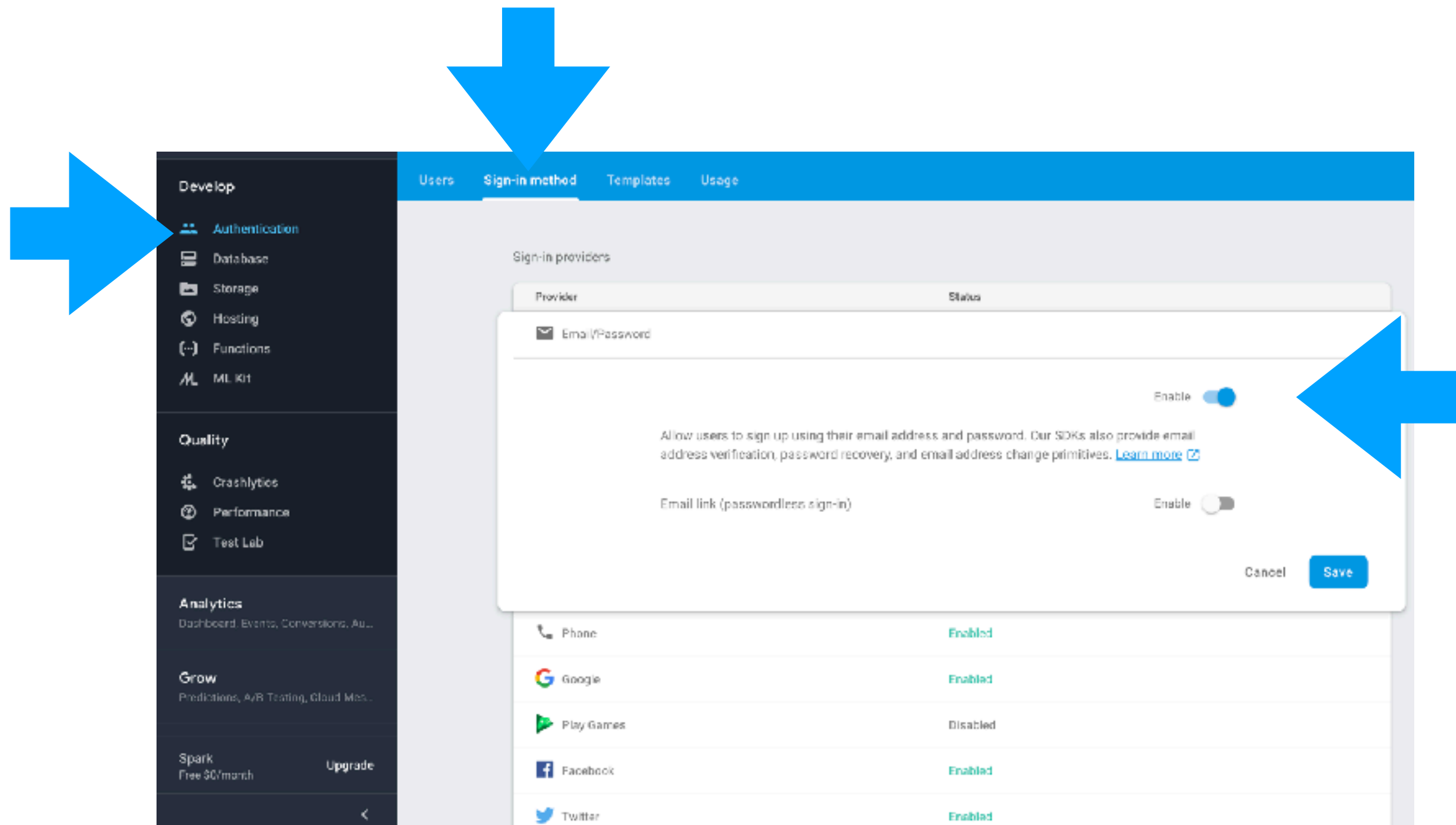
Sign-in providers

Provider	Status
✉ Email/Password	Disabled
📞 Phone	Disabled
🌐 Google	Disabled
📘 Facebook	Disabled
🐦 Twitter	Disabled
🐙 GitHub	Disabled
👤 Anonymous	Disabled

Sign in providers

Login With Email / Password

1. enable provider
2. register
3. send confirm email
4. user click link in email
5. sign in with email / password provider



Enable provider

Email/Password

com.google.firebase.auth.FirebaseAuth

- เป็น class เอาไว้สำหรับเรียกใช้ function การทำงานของ firebase auth
- ไม่ต้อง new instance ใช้วิธี getInstance() จาก Class

`FirebaseAuth.getInstance();`

Register

```
FirebaseAuth mAuth = FirebaseAuth.getInstance();

mAuth.createUserWithEmailAndPassword("email", "password").addOnSuccessListener(new OnSuccessListener<AuthResult>() {
    @Override
    public void onSuccess(AuthResult authResult) {

    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {

    }
});
```

com.google.firebase.auth.AuthResult

- เป็น class ที่ได้รับเมื่อ auth ทำงานสำเร็จ จะถูกส่งมาผ่าน SuccessListener
- สามารถเรียก getUser() เพื่อเข้าถึง class FirebaseUser

Confirm Email

```
void sendVerifiedEmail(FirebaseUser _user) {  
    _user.sendEmailVerification().addOnSuccessListener(new OnSuccessListener<Void>() {  
        @Override  
        public void onSuccess(Void aVoid) {  
            }  
    }).addOnFailureListener(new OnFailureListener() {  
        @Override  
        public void onFailure(@NonNull Exception e) {  
            }  
    });  
}
```

Current User

- Firebase Auth จะทำการเก็บ token และ check ให้อ่า user ด้ทำการ login แล้วหรือยัง
- ในโปรแกรมสามารถดูได้ว่a user ขณะนั้นคือใครได้ ดังนี้

```
FirebaseAuth mAuth = FirebaseAuth.getInstance();  
FirebaseUser mUser = mAuth.getCurrentUser();
```

- ถ้าได้ค่ามาแสดงว่a user ด้ทำการ login ไว้แล้ว
- สามารถหาค่า uid ได้จาก class FirebaseUser

Logout

```
FirebaseAuth mAuth = FirebaseAuth.getInstance();  
mAuth.signOut();
```

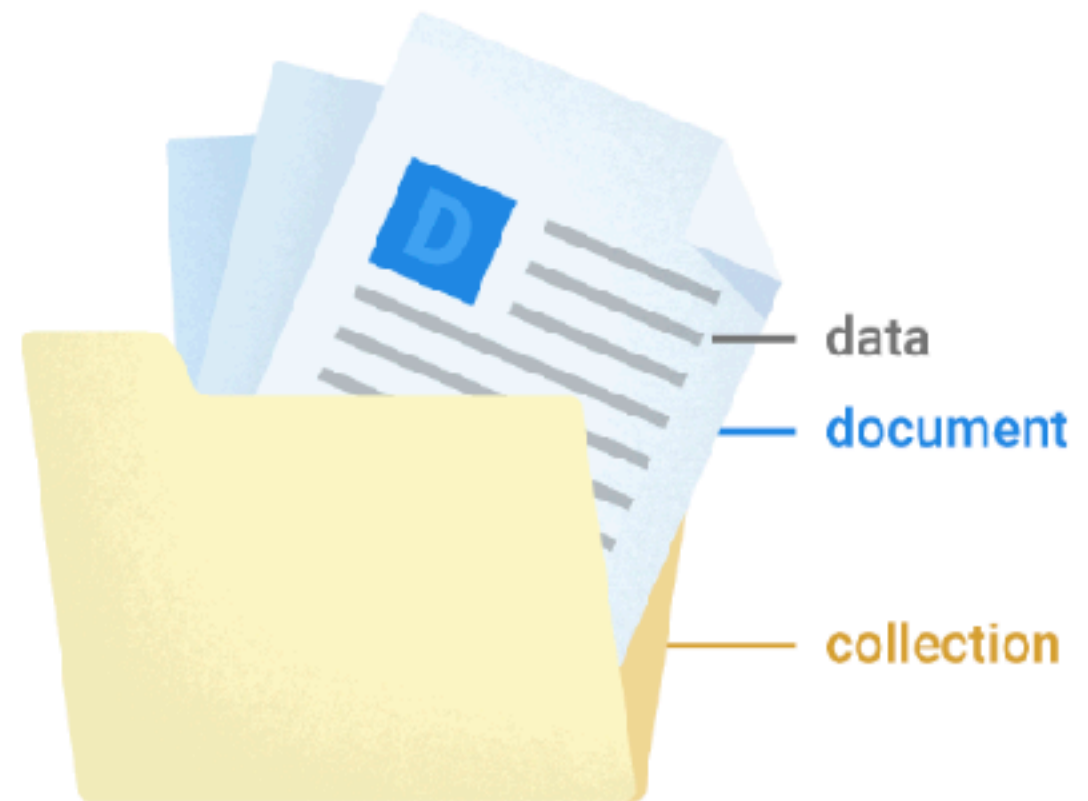
Todo

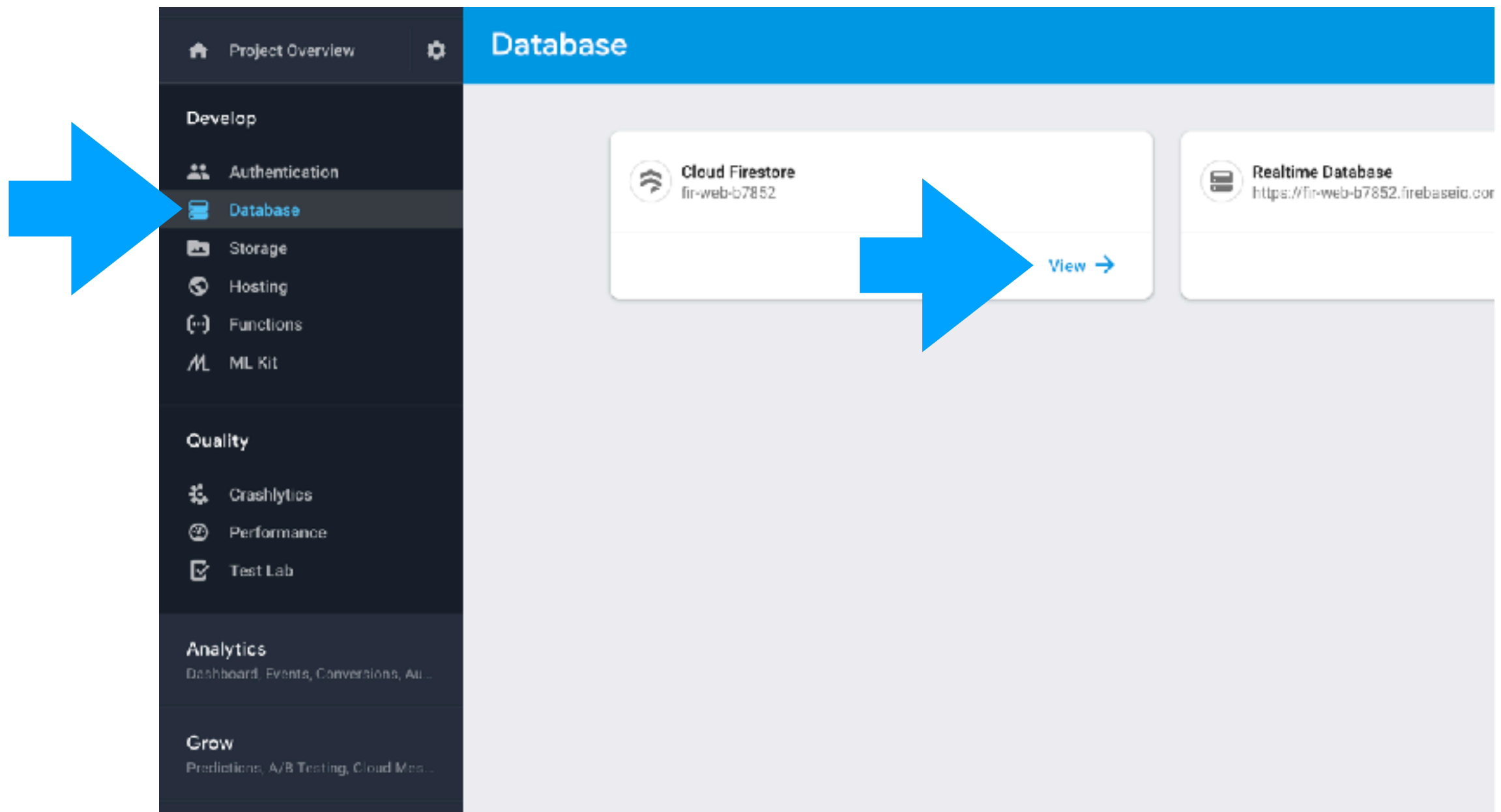
- Sign in
- Register
- Sign out
- Current user
- Add new record to db
- Delete record
- Update record
- List all record

Firestore

Firestore

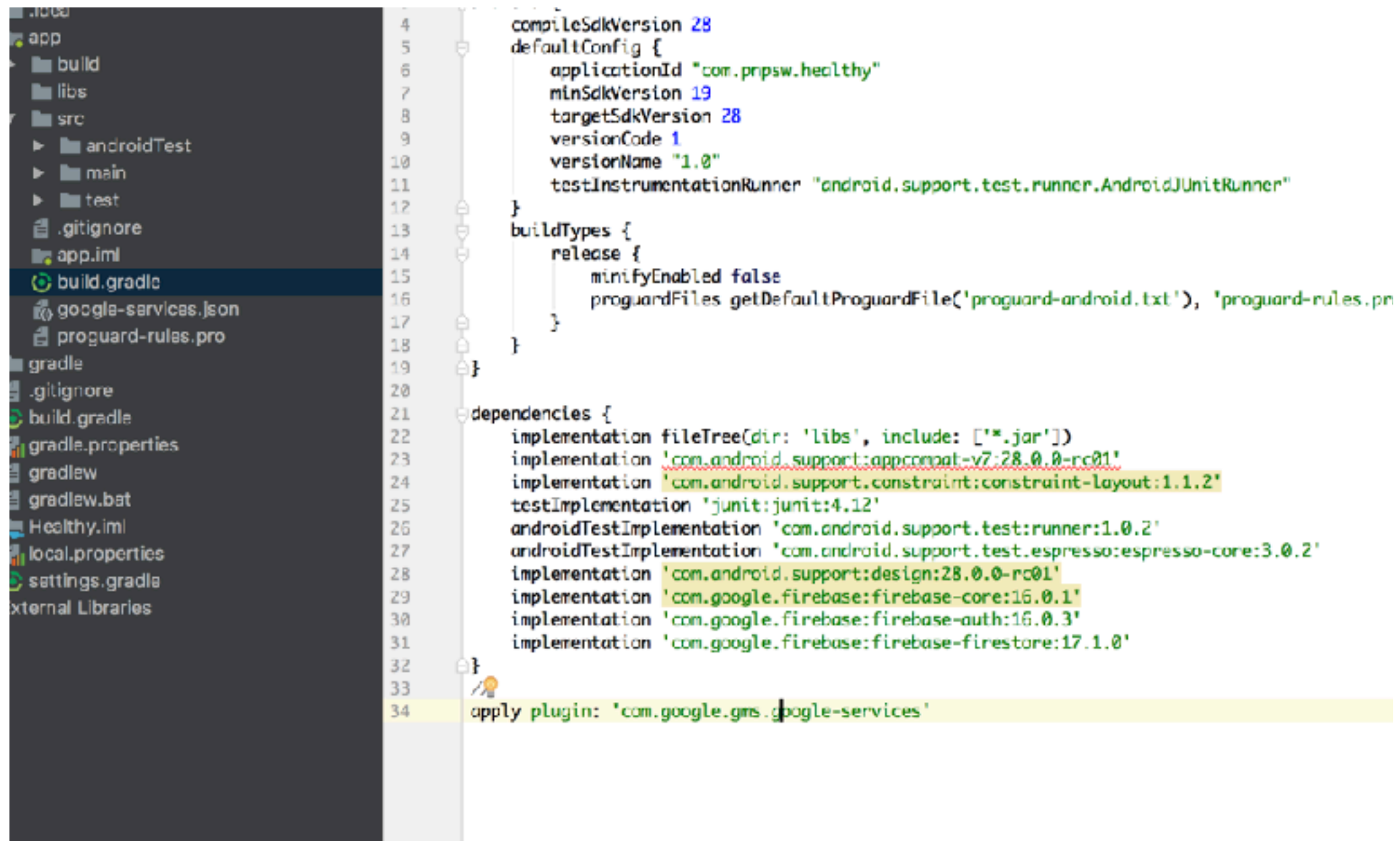
- คือ no sql db ที่ทำงานบน cloud ของ google
- เหมาะสำหรับ software พวกที่ไม่ต้องการความเป็น transaction
- เหมาะสำหรับงานที่ต้องการความ realtime
- เหมาะสำหรับงานที่ต้องการเก็บข้อมูลเยอะ และหลากหลาย





เข้า console

เลือก database / Cloud Firestore



เพิ่ม firestore เข้าไปที่ build.gradle

implementation 'com.google.firebase:firebase-firestore:17.1.0'

Add / Set Object

```
Firestore mdb = Firestore.getInstance();
mdb.collection("collection").document("document").set("data").addOnSuccessListener(new OnSuccessListener<Void>() {
    @Override
    public void onSuccess(Void aVoid) {

    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {

    }
});
```

Add list data

```
Firestore mdb = Firestore.getInstance();
mdb.collection("collection").add("list data").addOnSuccessListener(new OnSuccessListener<DocumentReference>() {
    @Override
    public void onSuccess(DocumentReference documentReference) {

    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {

    }
});
```

Delete Object

```
mdb.collection("collection").document("document").delete();
```

Get Document Data

```
DocumentReference docRef = mDB.collection("collection").document("document");
docRef.get().addOnSuccessListener(new OnSuccessListener<DocumentSnapshot>() {
    @Override
    public void onSuccess(DocumentSnapshot documentSnapshot) {
        documentSnapshot.getData(); // map
        documentSnapshot.toObject(Beans.class); // object
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
    }
});
```


Get all object from collection

```
mDB.collection("collection").addSnapshotListener(new EventListener<QuerySnapshot>() {  
    @Override  
    public void onEvent(  
@javax.annotation.Nullable QuerySnapshot documentSnapshots,  
@javax.annotation.Nullable FirebaseFirestoreException e) {  
        for (QueryDocumentSnapshot doc : documentSnapshots) {  
            if (doc.get("field_name") != null) {  
                // your logic here  
            }  
        }  
    }  
});
```

Workshop (10 คะแนน)

- ส่งงานผ่าน github ของตัวเองที่แจ้งไว้
- ส่งงานได้จนถึงวันที่ 16 Sep 2018
- แจ้งผลการตรวจภายในวันที่ 23 Sep 2018
- วิธีการให้คะแนน
 - รับผ่านและทำครบได้ 10 คะแนน
 - รับผ่านแต่ทำไม่ครบ 0 คะแนน
 - รับไม่ผ่าน 0 คะแนน

- แก้ไขหน้า register ให้เหลือแค่ field ดังนี้
 - Email, Password, Re-Password
- แก้ไขหน้า register ให้มี logic ดังนี้
 - Password ต้องมีความกว้างขั้นต่ำ 6 ตัวอักษร
 - password ทั้งสองต้องมีค่าเท่ากัน
 - เมื่อ กดปุ่ม register new account ให้ทำการสร้าง user UU google firebase auth
 - เมื่อสร้างเสร็จ ให้ส่ง email confirm ไปที่ user
 - เมื่อส่งเสร็จให้ไปที่หน้า login
- ในกรณี error ให้แสดง error ด้วย Toast

- แก้ไขหน้า login ให้มี logic ดังนี้
 - เมื่อกดปุ่ม Login ให้ใช้ firebase auth email/password provider ให้การ sign in
 - ต้องทำการตรวจสอบก่อนว่า user คนนั้นได้ทำการ confirm email เรียบร้อยแล้ว
 - เมื่อ login สำเร็จให้ไปที่หน้า menu
 - ในกรณีที่เคย login อยู่แล้ว (มี current user) ให้ไปที่หน้า menu

- แก้ไขหน้า menu ดังนี้
 - เพิ่ม menu ชื่อว่า sign out เมื่อกดแล้วให้ทำการ sign out ออกจาก firebase auth
 - เมื่อ sign out สำเร็จ ให้ไปที่หน้า login

- แก้ไขหน้า Weight Form ดังนี้
 - เมื่อกดปุ่ม บันทึก ให้เก็บข้อมูลลงใน firebase firestore ดังนี้
 - Collection = “myfitness”
 - Document = “uid”
 - Collection = “weight”
 - Document = “date”
 - Data = Weight Object
 - เมื่อบันทึกสำเร็จให้ไปที่หน้า weight
 - ในกรณีมีปัญหาให้แจ้ง error เป็น Toast

- แก้ไขหน้า Weight ดังนี้
- ให้ดึงข้อมูลจาก firebase firestore ดังนี้
 - Collection = “myfitness”
 - Document = “uid”
 - Collection = “weight”