

# Firebase Integration

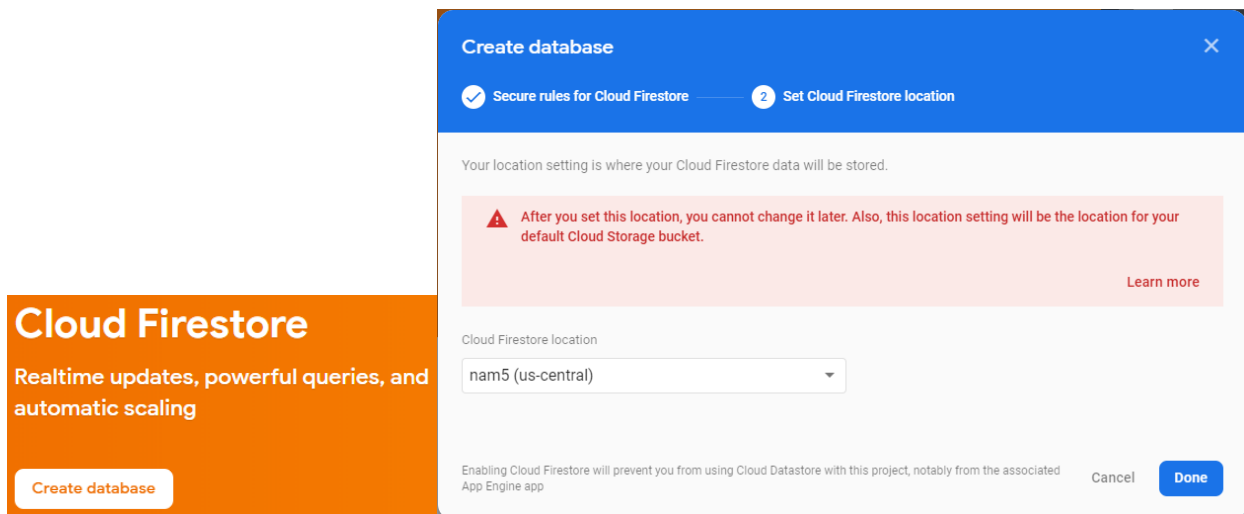
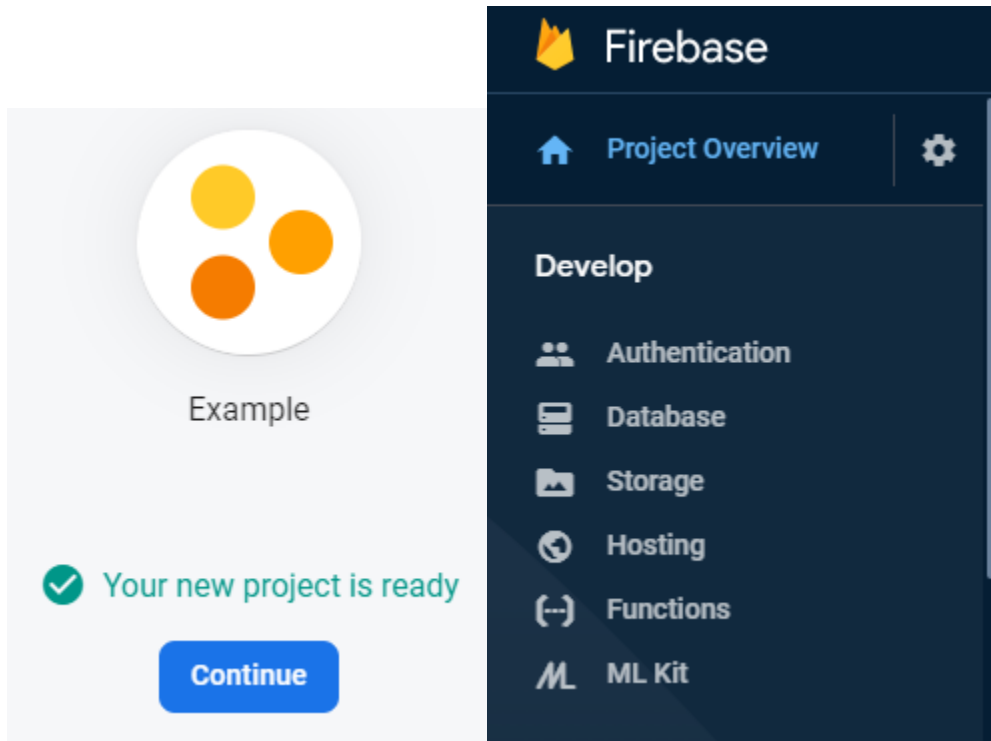
## Firebase Setting

1. ล็อกอิน Firebase ด้วย Google Account แล้วสร้าง Firebase Project ใหม่ ชื่อ Example
2. กด Continue -> Continue -> Create Project

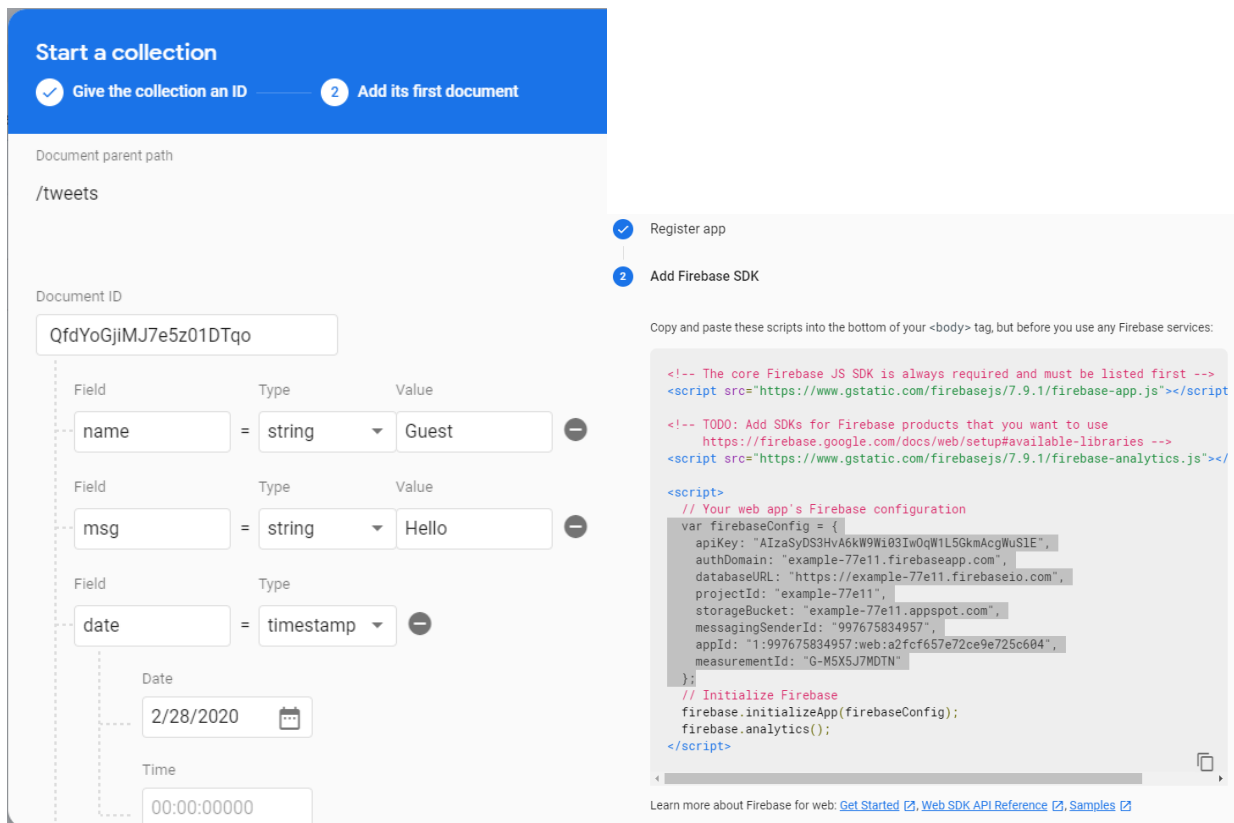
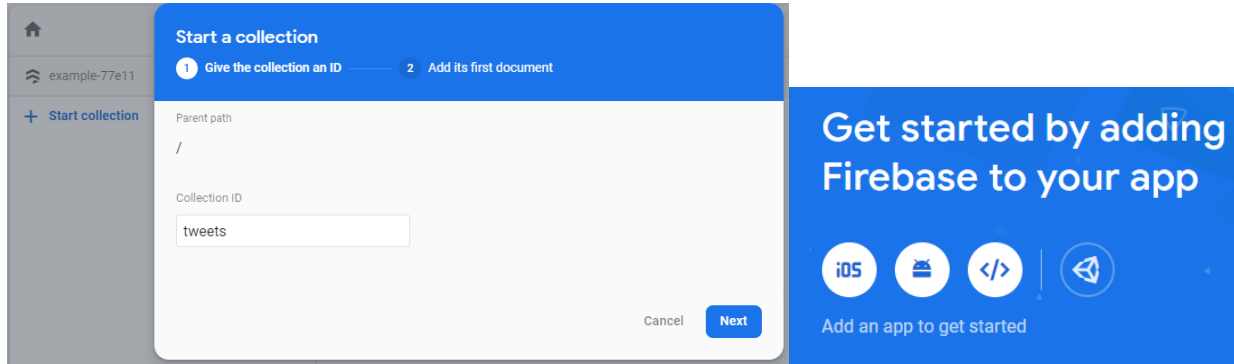
The image displays three sequential screenshots from the Firebase console during the project creation process:

- Top Left:** A dashboard titled "Your Firebase projects" with a large blue "+ Add project" button.
- Top Right:** A dialog box titled "Create a project (Step 1 of 3)". It prompts the user to name the project, with "Example" entered in the "Project name" field. Below, a unique ID "example-77e11" is shown. A blue "Continue" button is at the bottom.
- Bottom Left:** A dialog box titled "Create a project (Step 2 of 3)". It describes Google Analytics integration. Under "Google Analytics enables:", several features are listed with toggle switches: "A/B testing", "User segmentation & targeting across Firebase products", "Predicting user behavior", "Crash-free users", "Event-based Cloud Functions triggers", and "Free unlimited reporting". The "Enable Google Analytics for this project" option is checked and marked as "Recommended". "Previous" and "Continue" buttons are at the bottom.
- Bottom Right:** A dialog box titled "Create a project (Step 3 of 3)". It is titled "Configure Google Analytics" and asks to "Choose or create a Google Analytics account". A dropdown menu shows "Default Account for Firebase". Below, there is an option to "Automatically create a new property in this account". A note at the bottom explains that a new Google Analytics property will be created and linked to the project. "Previous" and "Create project" buttons are at the bottom.

3. หลังจากสร้าง Project เสร็จ เลือก Database -> Create Database
4. เลือก Location เป็น nam5(us-central) แล้วกด Done



- สร้าง Collection ชื่อ Tweets แล้วเพิ่ม Document ตามรูปด้านล่าง (เลือก auto-id)
- กดที่สัญลักษณ์ </> เพื่อ Copy Config เอาไว้ใช้ในแอปต่อไป
- แก้ไข Rule จาก false เป็น true



## Connect to Firebase

1. สร้างไฟล์ environment.ts แล้ว copy โค้ดจาก Firebase

```
export const environment = {
  firebaseConfig : {
    apiKey: "XXX",
    authDomain: "XXX",
    databaseURL: "XXX",
    projectId: "XXX",
    storageBucket: "XXX",
    messagingSenderId: "XXX",
    appId: "XXX",
    measurementId: "XXX"
  }
}
```

2. Install Firebase Module ใน app.module.ts

- a. Import AngularFireModule, AngularFireDatabaseModule, AngularFirestore และ environment
- b. เพิ่ม AngularFireModule, AngularFireDatabaseModule ใน @NgModule imports
- c. Initialize App ของ AngularFireModule ด้วย firebaseConfig ใน environment
- d. เพิ่ม AngularFirestore ใน @NgModule providers

```
import { AngularFireModule } from '@angular/fire';
import { AngularFireDatabaseModule } from '@angular/fire/database';
import { AngularFirestore } from '@angular/fire/firestore';
import { environment } from './environment';
...
@NgModule({
  imports: [
    ...
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireDatabaseModule
  ],
  ...
  providers: [..., AngularFirestore]
})
```

3. สร้าง Service ชื่อ Firebase เพื่อทำหน้าที่เชื่อมต่อกับ Firebase Database
  - a. Import AngularFirestore
  - b. providedIn: 'root'
  - c. Inject AngularFirestore ใน constructor
  - d. สร้างฟังก์ชัน getTweets โดยดึง Tweets Collection มาจาก Firebase

```
import { Injectable } from '@angular/core';
import { AngularFirestore } from '@angular/fire/firestore';

@Injectable({
  providedIn: 'root'
})
export class FirebaseService {

  constructor(
    private firestore: AngularFirestore
  ) { }

  getTweets() {
    return this.firestore.collection('tweets').snapshotChanges();
  }
}
```

4. แก้ไข Tweet Class ให้ข้อมูลเหมือนกับที่กำหนดไว้ใน Firebase Database
  - a. เปลี่ยน id จาก number เป็น string
5. แก้ไข Timeline Component ให้ดึงข้อมูลจาก Firebase Database
  - a. Import FirebaseService
  - b. Inject FirebaseService ใน constructor
  - c. แก้ไขฟังก์ชัน ngOnInit โดยไป Subscribe ฟังก์ชัน getTweets ของ FirebaseService เนื่องจากข้อมูลที่ได้มามีส่วน Header จึงต้องทำการ map ข้อมูลเข้ามา การใช้ as Tweet เป็นการบังคับให้ map ข้อมูลให้อยู่ในรูปแบบที่กำหนดในคลาส Tweet
  - d. เปลี่ยนค่า tweet.date ให้เป็น tweet.date.seconds\*100 ใน Display Tweet Component

```
import { FirebaseService } from '../firebase.service';
...
export class TimelineComponent implements OnInit {
  constructor(
    private fServ : FirebaseService
```

```

) { }

ngOnInit() {
  this.fServ.getTweets().subscribe(val => {
    this.tweets = val.map( e => {
      return {
        id: e.payload.doc.id,
        ...e.payload.doc.data()
      } as Tweet
    })
  });
}
}

```

```

<div class="tweet">
  <b>{{tweet.name}}</b> · {{tweet.date.seconds*1000 | timeAgo}}
  <p>{{tweet.msg}}</p>
  ...
</div>

```

## Practice

1. แก้ไข Add Tweet Component ให้เพิ่มข้อมูลไปยัง Firebase
  - a. กำหนดฟังก์ชัน addTweet ใน FirebaseService ดังนี้

```

import * as firebase from 'firebase/app';
...
addTweet(val1 : string, val2 : string) {
  let tweet = {
    name: val1,
    msg: val2,
    date: firebase.firestore.Timestamp.now(),
  };
  return this.firestore.collection('tweets').add(tweet);
}

```

2. แก้ไข Delete Component ให้ลบข้อมูลจาก Firebase
  - a. กำหนดฟังก์ชัน deleteTweet ใน FirebaseService ดังนี้

```

deleteTweet(id : string) {
  return this.firestore.collection('tweets').doc(id).delete();
}

```