

**Section** : IT Systems Development (DSI)  
**Module** : Framework cross-platform workshop  
**Teaching unit** : Mobile Development and Web  
**Level** : 3<sup>rd</sup> Year (Applied license: LMD)

## Workshop 6 : Communicating with a cloud Firebase database

---

### Objective

Create, update, read, and delete fiction products. Its UI is neat and clean with a single screen, a ListView, a bottom sheet, and a floating button

### Technical requirements

In order to start with Flutter, we need a few tools:

- A PC with a recent Windows version, or a Mac with a recent version of the macOS or Linux operating system. we can also use a Chrome OS machine, with a few tweaks.
- An Android/iOS setup.
- The Flutter SDK. It's free, light, and open source.
- Physical device/emulator/simulator
- Android Studio/IntelliJ IDEA or Visual Studio Code
- A registered Firebase account.
- A clean Flutter project with the firebase\_core plugin installed and correctly configured.
- Knowing Firebase's terms of service.

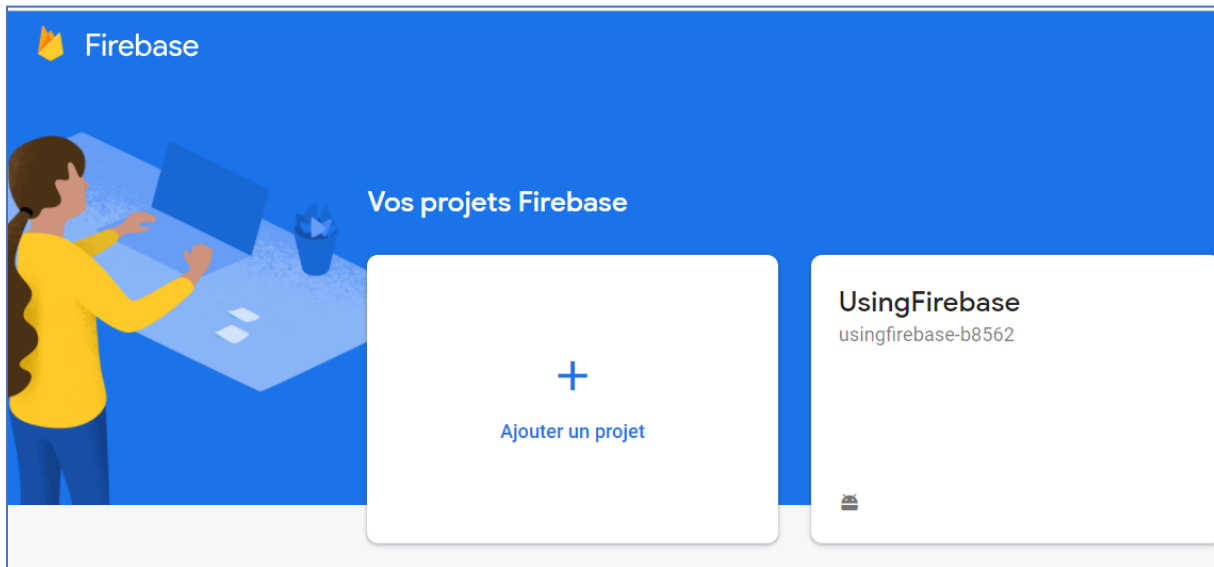
### content

We will cover the following recipe:

- Create a Firebase project.
- Adding Firebase SDK and installing the firebase\_core package
- Creating a cloud Firestore database
- Common Errors and How to Fix Them

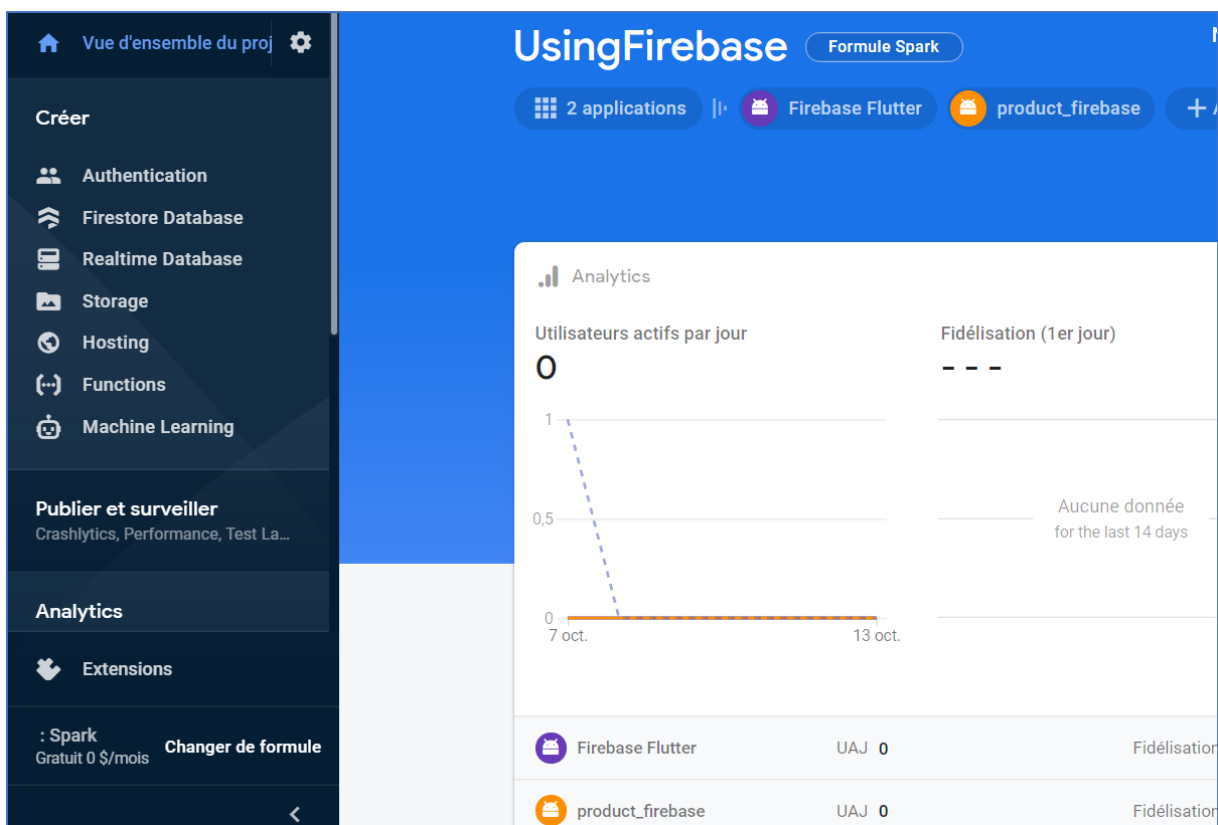
### Project configuration

1. Launch your web browser, go to <https://console.firebase.google.com/>, login, and click on the “+” button or select an existing project.



2. Give your project a name then click on the “Continue” button.
3. You can disable Google Analytics but you can reenable it later as you want. Click on the “Create project” button to initialize your project.

Here’s the project’s dashbaord:



4. Create a new Flutter project by running your IDE and give the name :  
product\_firebase
5. Add firebase\_core package to the dependencies section in your pubspec.yaml file:

```
dependencies:
  flutter:
```

```

sdk: flutter
firebase_core: ^1.7.0

```

- Run the following command in a terminal within the directory of your project Flutter:  
flutter pub get

We'll work with 2 **build.gradle** files. One lies in **<your-project>/android** and the other locates in **<your-project>/android/app**.

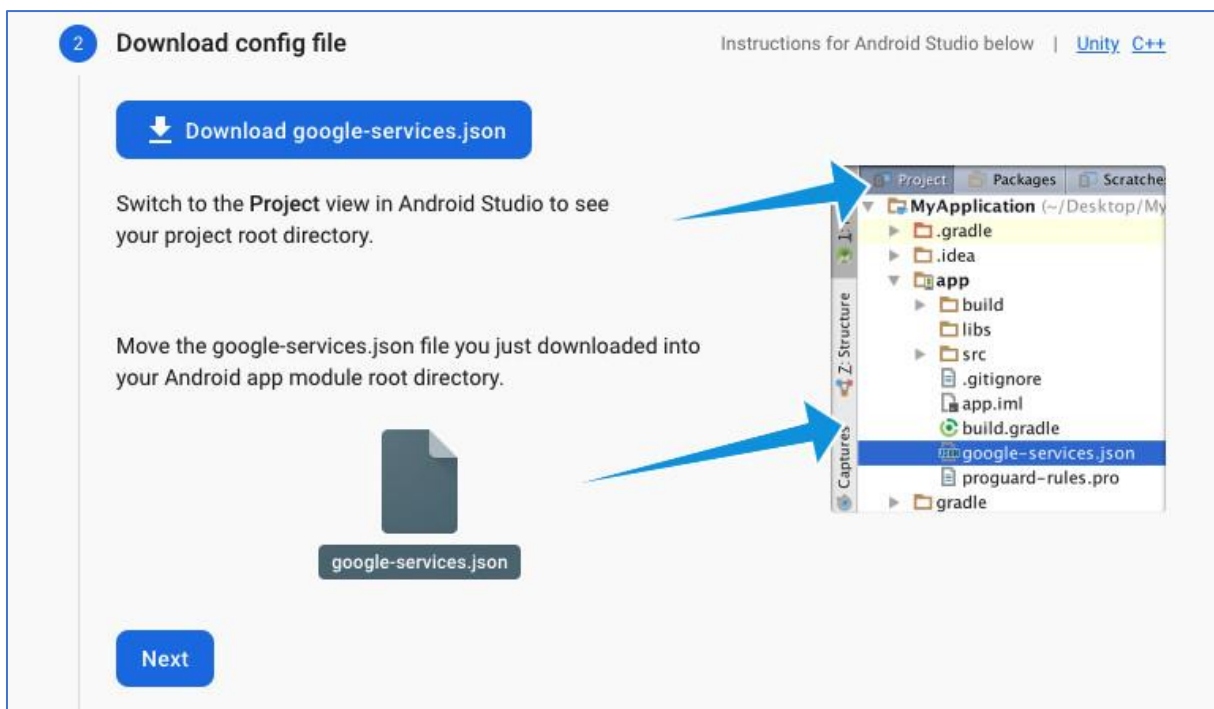
- Open **product\_firebase/android/app/build.gradle**, go to the **defaultConfig** section and see your **applicationId**:

```

defaultConfig {
    // TODO: Specify your own unique Application ID
    (https://developer.android.com/studio/build/application-id.html).
    applicationId "com.example.product_firebase"
    minSdkVersion 16
    targetSdkVersion 30
    versionCode flutterVersionCode.toInteger()
    versionName flutterVersionName
    multiDexEnabled true
}

```

- Go back to your Firebase project's dashboard, click on the "+ Add app" button and register an Android app. Enter the **applicationId** you have seen in the previous step into the "Android package name" field. Note that it cannot be changed for this Firebase Android app after it's registered with your Firebase project.
- Download the "google-services.json" file



- Move the "google-services.json" file into your **product\_firebase/android/app** directory:
- Open **product\_firebase/android/build.gradle** and add the following line to the **dependencies** section:

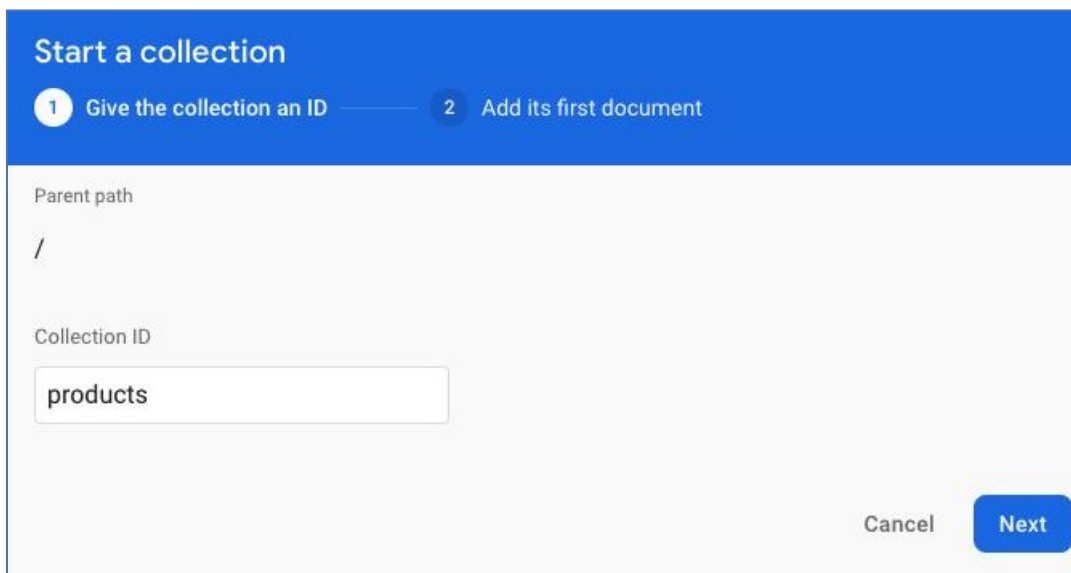
```
dependencies {  
    classpath 'com.android.tools.build:gradle:4.1.0'  
    classpath "org.jetbrains.kotlin:kotlin-gradle-  
plugin:$kotlin_version"  
    classpath 'com.google.gms:google-services:4.3.5'  
}
```

12. Open `product_firebase/android/app/build.gradle` and add the following line right below the **apply plugin: 'com.android.application'** line:

```
apply plugin: 'com.android.application'  
apply plugin: 'com.google.gms.google-services'  
apply plugin: 'kotlin-android'  
apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"  
  
android {...
```

## Create a firebase database

13. Go to your Firebase project's dashboard, select "Firestore Database" then click on the "Create database" button
14. Select "Start in test mode" then click on the "Next" button (you can change to "production mode" later if you want).
15. Select a location then click the "Enable" button. The closer the location is, the lower the latency will be and the faster your app will function.
16. Click "+ Start Collection" to create a new collection.
17. Enter "products" and click "Next".



18. Add the first documents to our "products" collection. It has 2 fields: "name" (string) and "price" (number).

**Start a collection**

✓ Give the collection an ID — 2 Add its first document

Document parent path  
/products

Document ID  
PNSLP2h60KGliMF6smpn

| Field | Type   | Value |
|-------|--------|-------|
| name  | string |       |
| price | number |       |

Cancel Save

19. Fill in the fields then click “Save”. Now we get a Firestore database ready to use.

**Cloud Firestore**

Données Règles Index Utilisation

Prototypiez et testez vos applications de bout en bout avec la suite d'émulateurs locaux, désormais compatible avec Firebase A

🏠 > products > AKW8gRdN1NA...

| usingfirebase-b8562        | products               | AKW8gRdN1NAggkgU2J9I       |
|----------------------------|------------------------|----------------------------|
| + Commencer une collection | + Ajouter un document  | + Commencer une collection |
| products >                 | AK4wt2vcY3RfQtLZ0XY    | + Ajouter un champ         |
|                            | AKW8gRdN1NAggkgU2J91 > | name: "PC"                 |
|                            |                        | price: 2                   |

## Installing cloud\_firestore

20. You can add **cloud\_firestore** to your project by executing the command below:

```
flutter pub add cloud_firestore
```

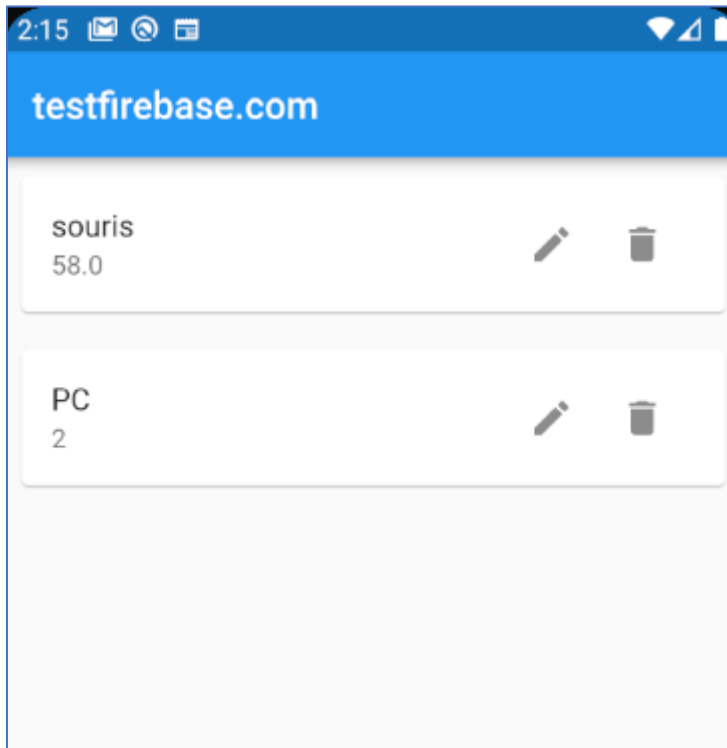
21. Or you can also follow the traditional method: Add the following to the **dependencies** block in your **pubspec.yaml**:

```
cloud_firestore: ^2.5.3
```

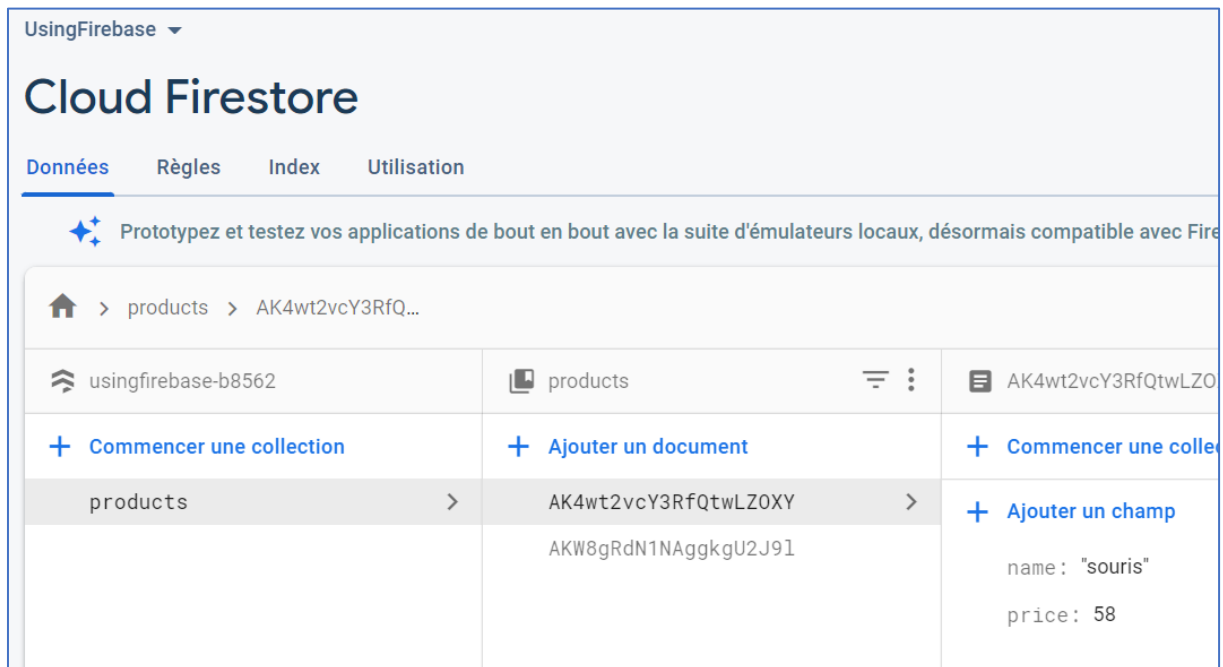
22. Run:

```
flutter pub get
```

23. write now the main code for crud operation



And add an example of product. Switch it in the firebase DB



## Common Errors and How to Fix Them

1- In some cases, you need to upgrade fire\_core

2- Error about multiDex

This error happens when your app and the libraries it references exceed 65,536 methods, the build error just means your app has reached the limit of the Android build architecture, so to fix this go to android/app/build.gradle directory and add the multiDexEnabled true and implementation 'com.android.support:multidex:2.0.1' in thus manner

```
defaultConfig {
    // TODO: Specify your own unique Application ID
    (https://developer.android.com/studio/build/application-id.html).
    .....
    multiDexEnabled true
}
```

```
dependencies {
    .....
    implementation 'com.android.support:multidex:2.0.1'
}
```

## Application

Predict crud operations on classes and students tables with Firebase