

MODUL

PEMROGRAMAN BERBASIS PLATFORM



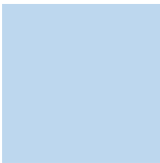
PERTEMUAN – 7

DATA 2 – FIREBASE DATABASE

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ATMA JAYA YOGYAKARTA





Daftar Gambar

| | |
|----|---|
| 2 | |
| 3 | Gambar 1. Firebase..... 4 |
| 4 | Gambar 2. Console Firebase 5 |
| 5 | Gambar 3. Buat project baru pada firebase dengan nama flutter-firebase..... 6 |
| 6 | Gambar 4. Disable Google Analytics pada project firebase..... 6 |
| 7 | Gambar 5. Link aplikasi flutter dengan firebase(Android)..... 7 |
| 8 | Gambar 6. Contoh penamaan package untuk firebase(Android) 7 |
| 9 | Gambar 7. Download file json yang telah disediakan firebase..... 8 |
| 10 | Gambar 8. Path untuk menyimpan file json sebelumnya..... 9 |
| 11 | Gambar 9. Import dependency google-service pada gradle(Android) 10 |
| 12 | Gambar 10. Apply plugin dari dependancy sebelumnya(Android) 10 |
| 13 | Gambar 11. Sesuaikan versi SDK(Android)..... 11 |
| 14 | Gambar 12. Import library firestore dan slideable pada aplikasi flutter 11 |
| 15 | Gambar 13. Modifikasi main.dart..... 12 |
| 16 | Gambar 14. Class Entity Employee..... 13 |
| 17 | Gambar 15. Buat file inputPage.dart 13 |
| 18 | Gambar 16. inputPage.dart bagian 1..... 14 |
| 19 | Gambar 17. inputPage.dart bagian 2 14 |
| 20 | Gambar 18. inputPage.dart bagian 3. Pemanggilan fungsi create yang telah 21 disediakan firebase 15 |
| 22 | Gambar 19. Main.dart bagian 1 15 |
| 23 | Gambar 20. Main.dart bagian 2 16 |
| 24 | Gambar 21. Main.dart bagian 3..... 16 |
| 25 | Gambar 22. Main.dart bagian 4 17 |
| 26 | Gambar 23. Main.dart bagian 5 17 |
| 27 | Gambar 24. Masuk ke opsi Firestore Database 18 |
| 28 | Gambar 25. Buat cloud firestore 18 |
| 29 | Gambar 26. Pilih opsi test mode 19 |
| 30 | Gambar 27. Pilih lokasi database..... 19 |
| 31 | Gambar 28. Firestore Database yang sudah jadi..... 20 |
| 32 | Gambar 29. Hasil main.dart..... 21 |
| 33 | Gambar 30. Hasil inputPage.dart..... 22 |
| 34 | Gambar 31. inputPage.dart dengan data 23 |
| 35 | Gambar 32. main.dart dengan data 24 |
| 36 | Gambar 33. Firebase database dengan data 24 |
| 37 | |



TUJUAN

Setelah menyelesaikan modul ini, praktikan diharapkan mampu :

1. Memahami apa itu Firebase .
2. Mengimplementasikan CRUD dengan firebase realtime database pada aplikasi flutter.

DASAR TEORI

A. FIREBASE

Firebase merupakan layanan Backend as a Services (BaaS) yang disediakan oleh Google berupa beragam tools dan fitur untuk membantu pengembangan sebuah aplikasi dengan lebih cepat. Selayaknya BaaS yang merupakan salah satu kategorit layanan cloud dalam pengelolaan backend sebuah aplikasi, Firebase dapat menyediakan pengelolaan database(NoSQL), hosting, authentication, API, dan berbagai fitur terkait backend sehingga developer dapat berfokus pada pembuatan front-end dan menyerahkan urusan backend pada Firebase. Berikut adalah beberapa jenis atau fitur yang tersedia di Firebase.

- Firebase Analytics
- Firebase Cloud Messaging and Notifications
- Firebase Authentication
- Firebase Cloud Firestore
- Firebase Realtime Database
- Firebase Hosting



Gambar 1. Firebase

B. FIREBASE REALTIME DATABASE

Modul ini akan berfokus pada layanan Firebase Realtime Database, yakni layanan database NoSQL yang sudah menerapkan “offline sync”. Dengan demikian, pengguna tetap dapat mengakses database tanpa koneksi internet dengan cara menyimpan sementara data pada local storage dan kemudian akan melakukan sinkronisasi ketika perangkat telah terkoneksi kembali dengan internet. Selain itu, database yang disediakan oleh Firebase bersifat realtime sehingga data yang ditampilkan pada perangkat pengguna merupakan data terbaru dan akan selalu diperbaharui apabila terjadi perubahan data pada database. Namun realtime database tidak mendukung relasi antar database.

Untuk mendukung framework flutternya, Google menyediakan fitur realtime database sendiri pada firebase dengan nama Firestore Database. Secara fungsi, tidak ada perbedaan signifikan dengan Realtime Database yang sudah dimiliki firebase sebelumnya, namun Firestore lebih dioptimalkan untuk penggunaan flutter.

*referensi :

1. <https://firebase.google.com/docs/database>



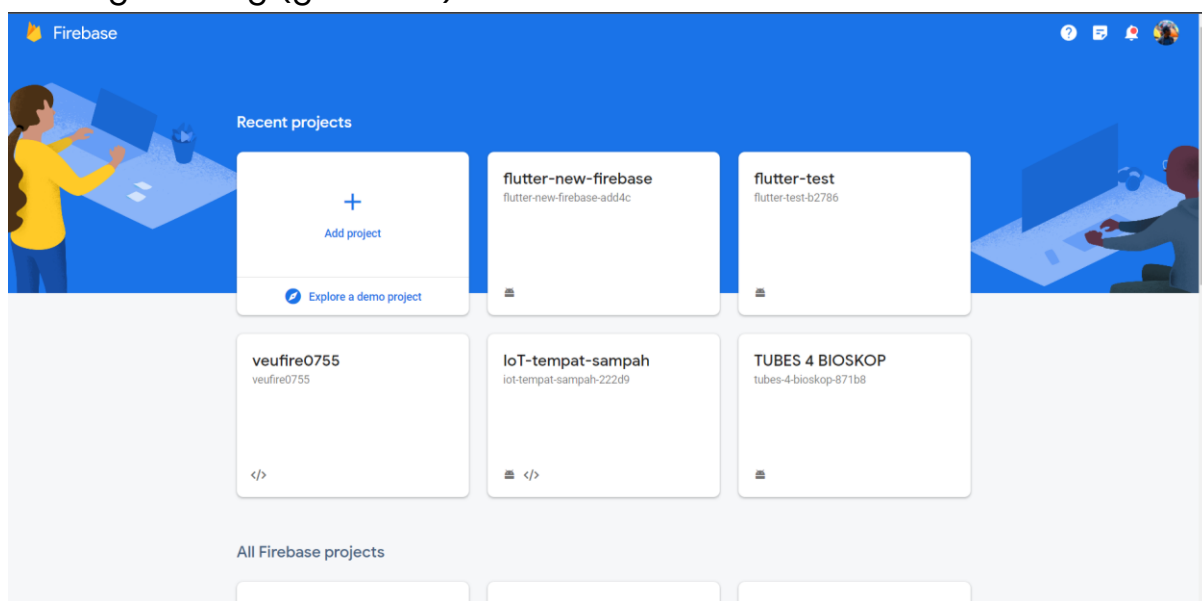
GUIDED 1 – EMPLOYEE CRUD

Poin yang dipelajari dalam Guided 1 ini, yaitu :

1. Memahami cara penggunaan firebase realtime database.
2. Mampu mengimplementasikan firebase realtime database pada aplikasi flutter.

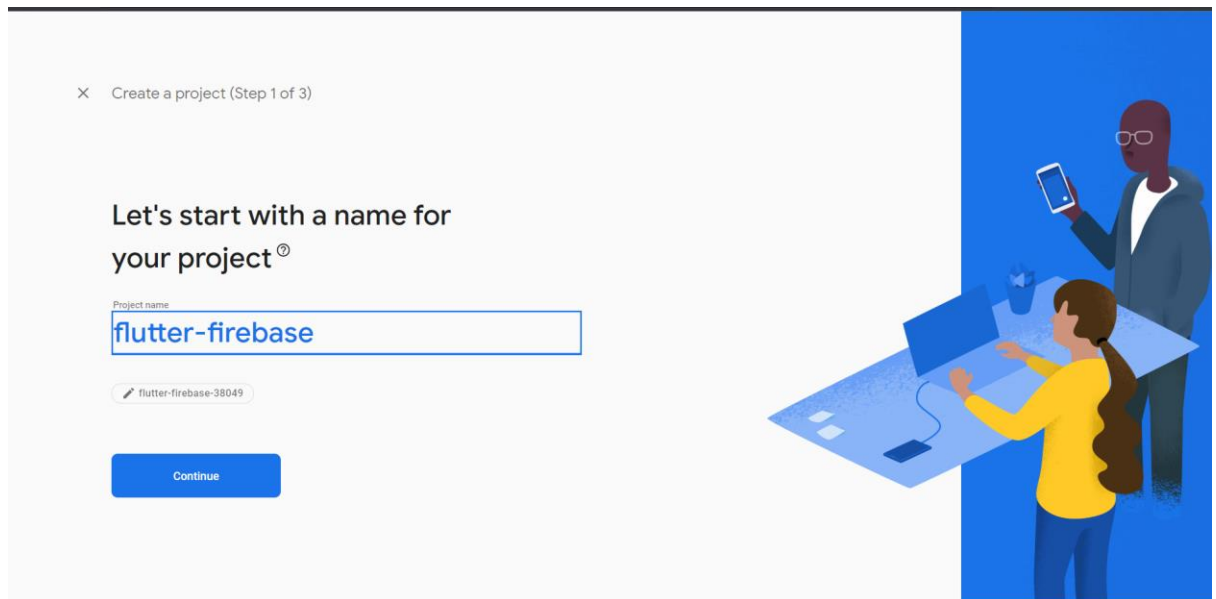
Pada guided 1 ini, kita akan mencoba membuat aplikasi flutter sederhana dengan fitur CRUD dengan Firebase Realtime Database:

1. Buat project flutter dengan nama gd6_x_yyyy, x merupakan nama kelas, dan yyyy merupakan 4 digit npm terakhir praktikan terlebih dahulu.
2. Kemudian buka console firebase pada laman : <https://console.firebase.google.com/> dan masuk dengan akun gmail masing-masing (gambar 1).



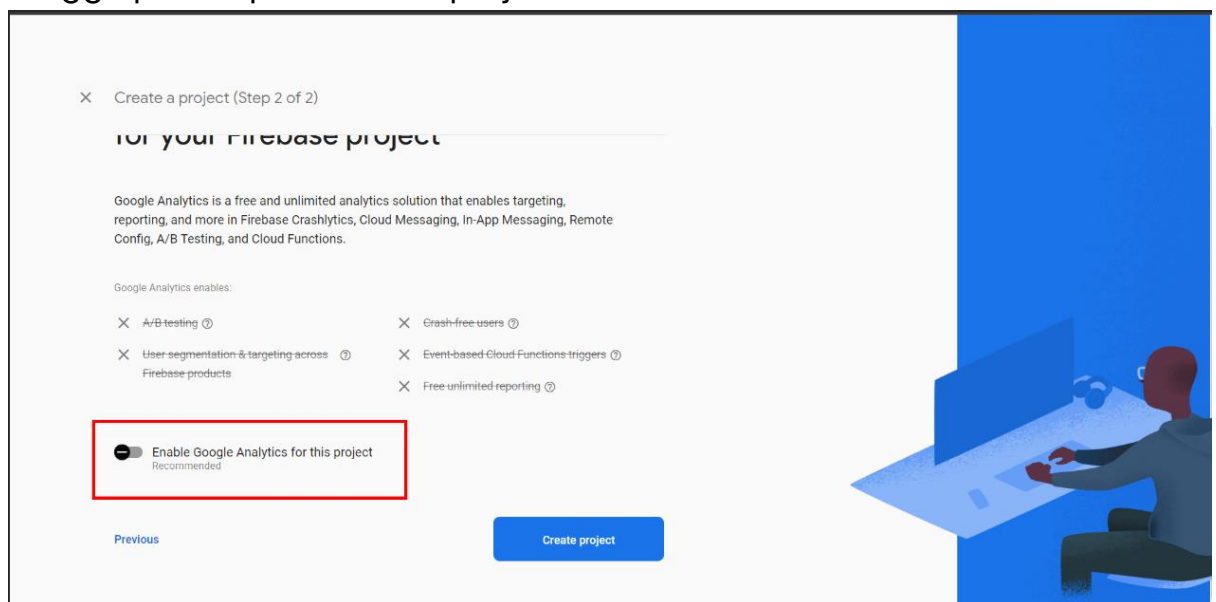
Gambar 2. Console Firebase

3. Buat project baru dengan nama flutter-firebase



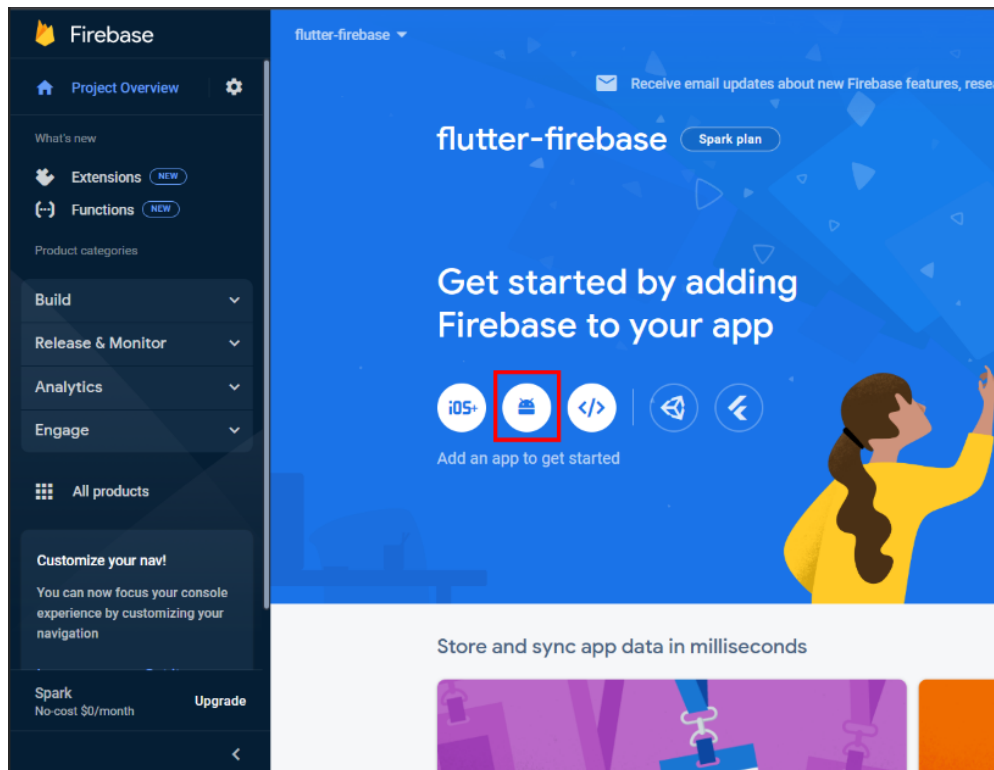
Gambar 3. Buat project baru pada firebase dengan nama flutter-firebase

4. Disable Google Analytics pada project kemudian create project dan tunggu proses pembuatan project.



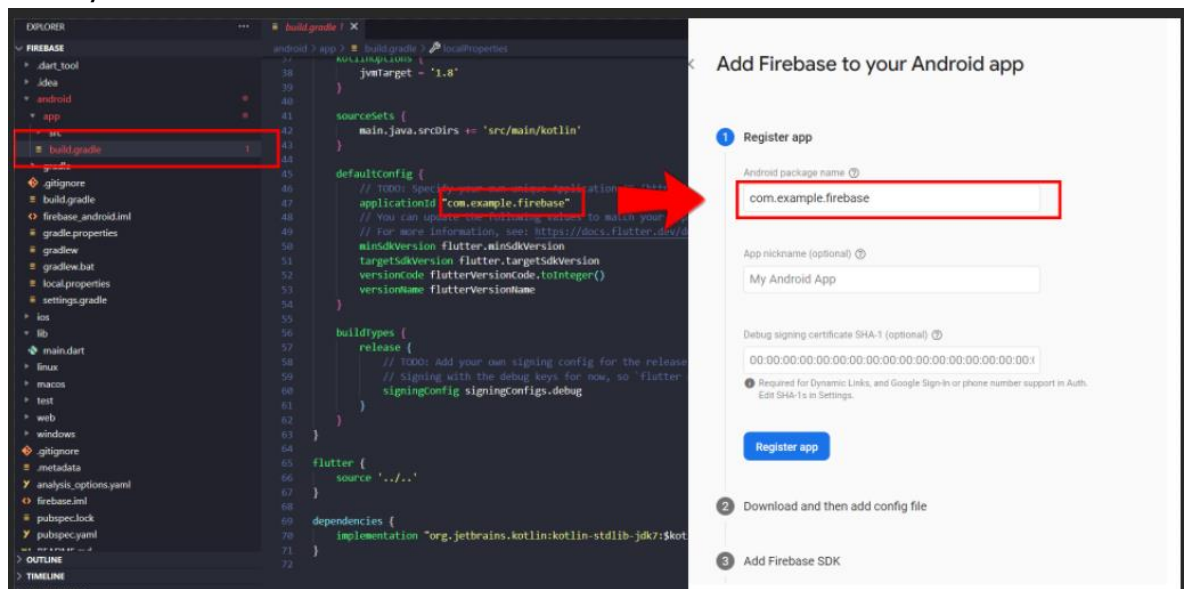
Gambar 4. Disable Google Analytics pada project firebase

5. Link project firebase dengan aplikasi flutter yang sudah dibuat. Modul ini akan menggunakan android sebagai device, maka pilih logo android pada halaman home project.



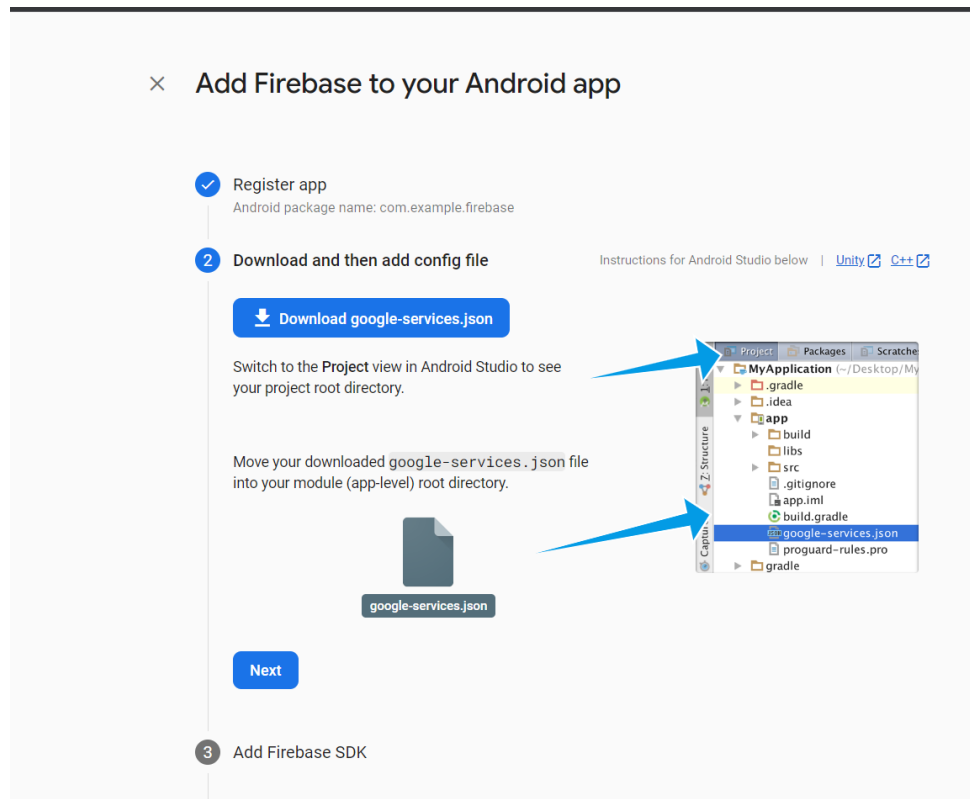
Gambar 5. Link aplikasi flutter dengan firebase(Android)

6. Isikan Android Package Name pada firebase dengan applicationId project flutter yang terdapat pada folder android\app\build.gradle.
*Untuk device selain android dapat disesuaikan yakni pada folder lainnya sesuai OS dari device.



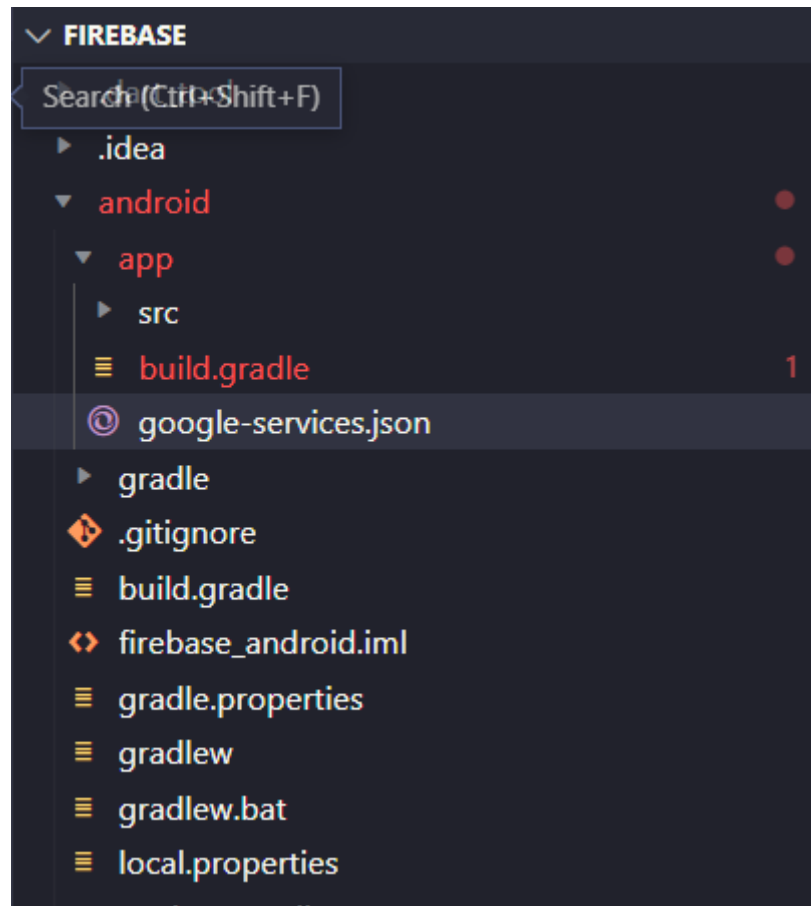
Gambar 6. Contoh penamaan package untuk firebase(Android)

7. Download file json yang telah disediakan. File ini yang nantinya akan mengurus backend dari aplikasi flutter.



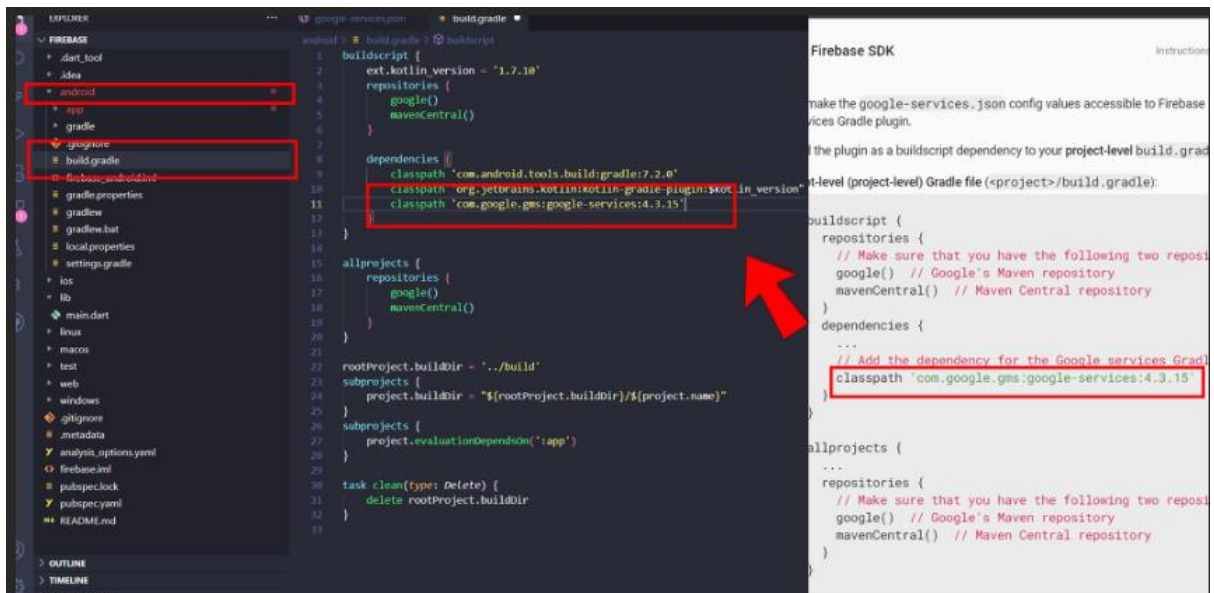
Gambar 7. Download file json yang telah disediakan firebase

8. Copy file json tersebut kedalam project flutter pada folder android/app. *Untuk device selain android dapat disesuaikan yakni pada folder lainnya sesuai OS dari device.



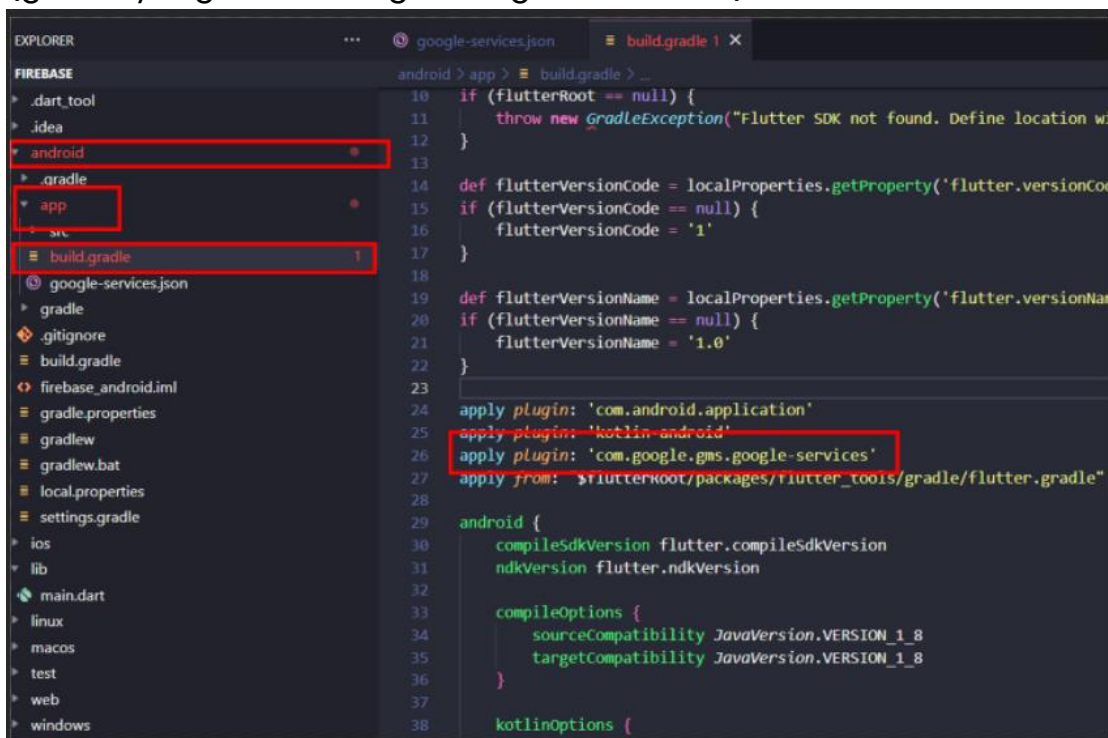
Gambar 8. Path untuk menyimpan file json sebelumnya

9. Salin dependencies yang google-service pada aplikasi flutter yakni pada android->build.gradle(berbeda dengan gradle pada langkah sebelumnya) agar aplikasi dapat mengakses json yang sudah diimport sebelumnya. *Untuk device selain android dapat disesuaikan yakni pada folder lainnya sesuai OS dari device.



Gambar 9. Import dependency google-service pada gradle(Android)

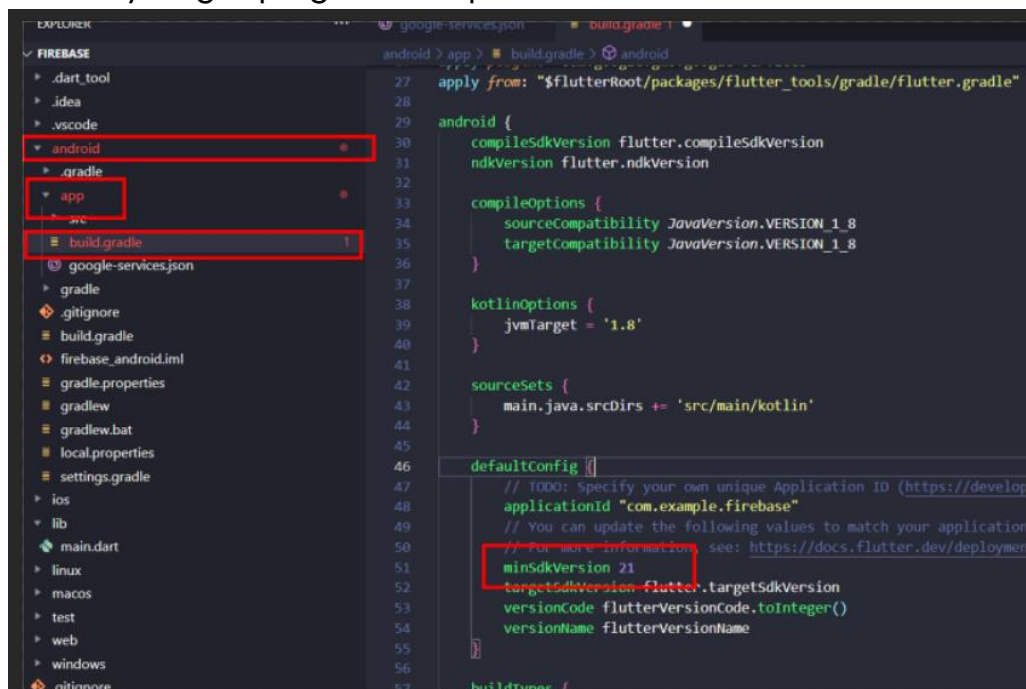
10. Apply plugin google-service dengan pada android\app\build.gradle (gradle yang sama dengan langkah nomor 6)



Gambar 10. Apply plugin dari dependency sebelumnya(Android)

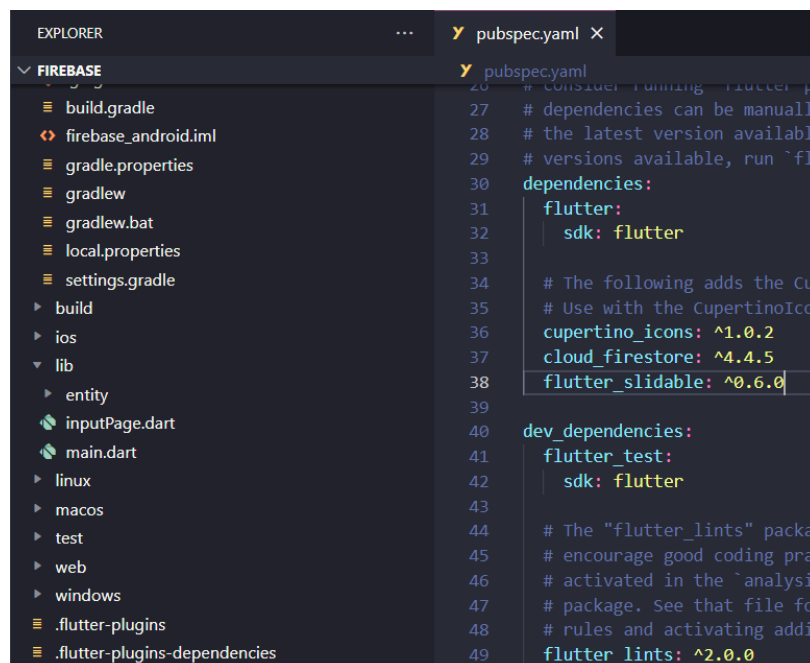
11. Untuk memastikan firebase dapat berjalan dengan baik set minimum Software Development Kit(SDK pada gradle menjadi 21. Kemudian lanjutkan register app pada firebase hingga selesai. Jangan lupa

untuk save semua file pada flutter yang sudah dimodifikasi sebelumnya agar plugin tersimpan.



Gambar 11. Sesuaikan versi SDK(Android)

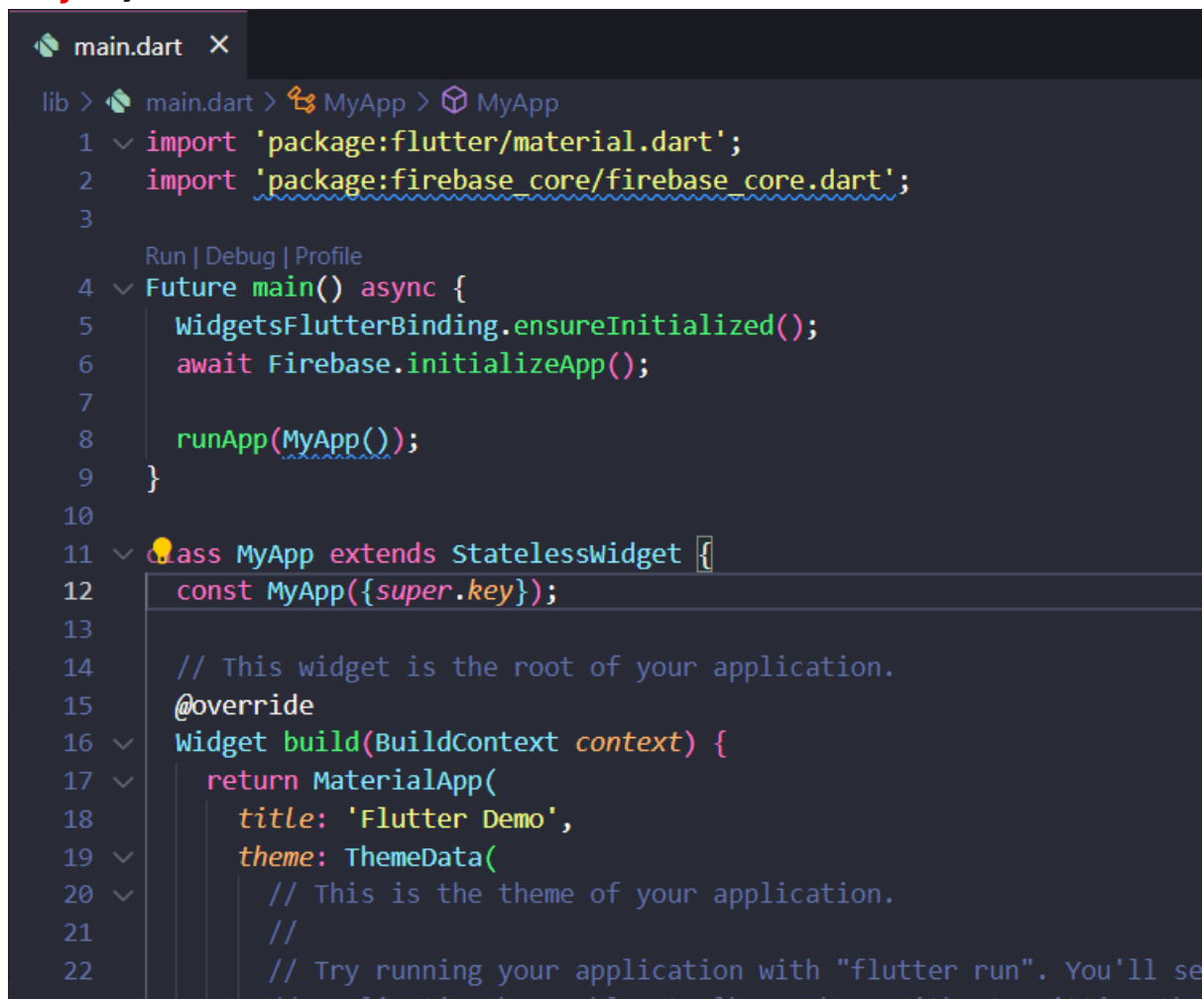
12. Tambahkan library cloud_firestore pada aplikasi flutter agar aplikasi dapat menggunakan firebase. Tambahkan juga library slideable untuk UI



Gambar 12. Import library firestore dan slideable pada aplikasi flutter



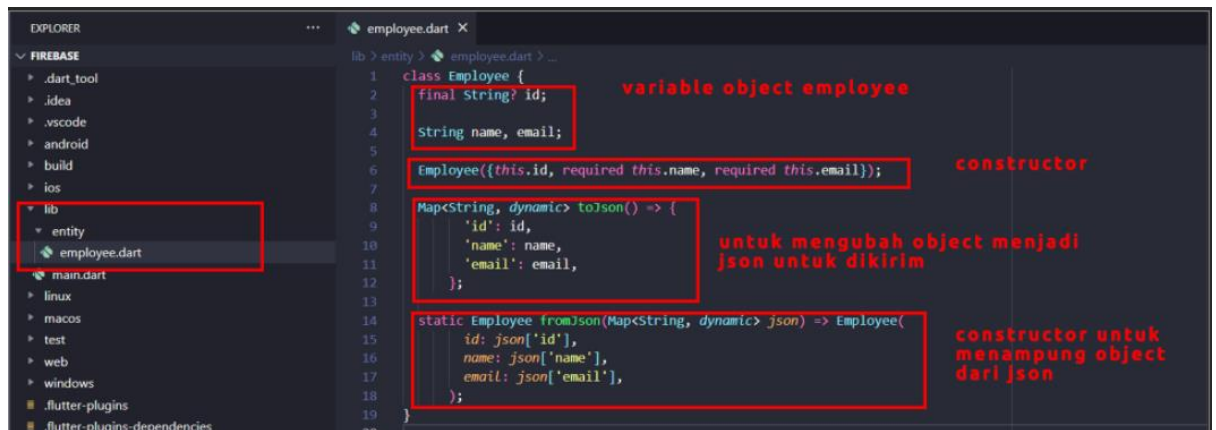
13. Lakukan test untuk memastikan firebase sudah terintegrasi dengan baik dengan cara memodifikasi main.dart sebagai berikut(**Run Project**)



```
main.dart X
lib > main.dart > MyApp > MyApp
1 import 'package:flutter/material.dart';
2 import 'package:firebase_core/firebase_core.dart';
3
4 Run | Debug | Profile
5 Future main() async {
6   WidgetsFlutterBinding.ensureInitialized();
7   await Firebase.initializeApp();
8   runApp(MyApp());
9 }
10
11 class MyApp extends StatelessWidget {
12   const MyApp({super.key});
13
14   // This widget is the root of your application.
15   @override
16   Widget build(BuildContext context) {
17     return MaterialApp(
18       title: 'Flutter Demo',
19       theme: ThemeData(
20         // This is the theme of your application.
21         //
22         // Try running your application with "flutter run". You'll see
23         // the application with the theme from this file.
```

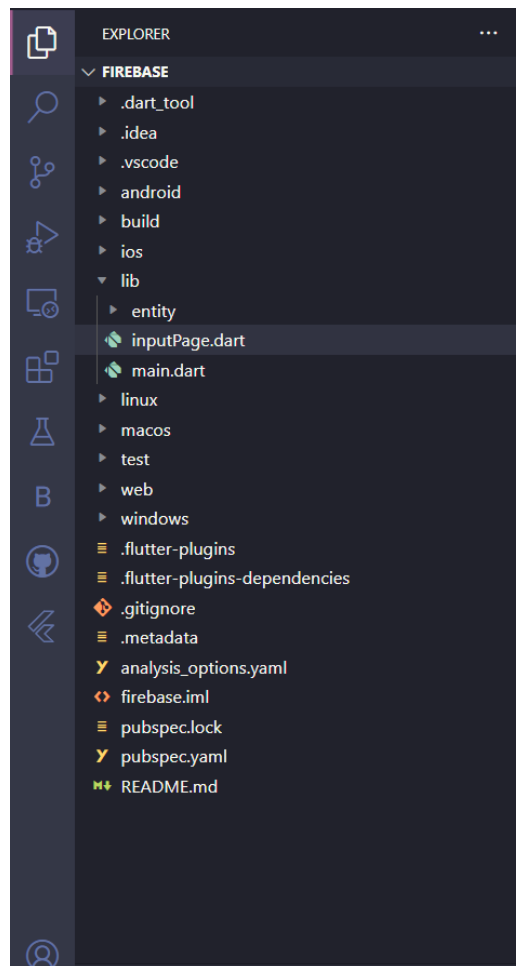
Gambar 13. Modifikasi main.dart

14. Apabila aplikasi dapat tampil seperti default flutter project dan tidak menunjukkan black screen, maka firebase siap untuk digunakan.
15. Buat class Employee sebagai berikut



Gambar 14. Class Entity Employee

16. Buat `inputPage.dart` sebagai halaman create dan update object sebagai berikut



Gambar 15. Buat file `inputPage.dart`



```
inputPage.dart X
lib > inputPage.dart > _InputPageState
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import 'package:flutter/material.dart';
3 import 'package:firebase/entity/employee.dart';
4
5 class InputPage extends StatefulWidget {
6   const InputPage(
7     {super.key,
8     required this.title,
9     required this.id,
10    required this.name,
11    required this.email});
12
13   final String? title, name, email, id;
14
15   @override
16   State<InputPage> createState() => _InputPageState();
17 }
18
19 class _InputPageState extends State<InputPage> {
20   TextEditingController controllerName = TextEditingController();
21   TextEditingController controllerEmail = TextEditingController();
22
23   @override
24   Widget build(BuildContext context) {
25     if (widget.id != null) {
26       controllerName.text = widget.name!;
27       controllerEmail.text = widget.email!;
28     }
29     return Scaffold(
30       appBar: AppBar(
31         title: Text("INPUT EMPLOYEE"),
32       ), // AppBar
```

Gambar 16. inputPage.dart bagian 1

```
inputPage.dart X
lib > inputPage.dart > _InputPageState > build
33 body: ListView(
34   padding: EdgeInsets.all(16),
35   children: <Widget>[
36     TextField(
37       controller: controllerName,
38       decoration: const InputDecoration(
39         border: UnderlineInputBorder(),
40         labelText: 'Name',
41       ), // InputDecoration
42     ), // TextField
43     SizedBox(height: 24),
44     TextField(
45       controller: controllerEmail,
46       decoration: const InputDecoration(
47         border: UnderlineInputBorder(),
48         labelText: 'Email',
49       ), // InputDecoration
50     ), // TextField
51     SizedBox(height: 48),
52     ElevatedButton(
53       child: Text('Save'),
54       onPressed: () async {
55         if (widget.id == null) {
56           createEmployee(
57             name: controllerName.text, email: controllerEmail.text);
58         } else {
59           final docEmployee = FirebaseFirestore.instance
60             .collection('employee')
61             .doc(widget.id);
62           docEmployee.update({
63             'name': controllerName.text,
64             'email': controllerEmail.text,
65           });
66         }
67         Navigator.pop(context);
68       },
69     ), // ElevatedButton
```

Gambar 17. inputPage.dart bagian 2



```
69     } // ElevatedButton
70   ], // <Widget>[]
71   )); // ListView // Scaffold
72 }
73
74 Future createEmployee({required String name, required String email}) async {
75   final docEmployee = FirebaseFirestore.instance.collection('employee').doc();
76
77   final employee = Employee(id: docEmployee.id, name: name, email: email);
78   final json = employee.toJson();
79
80   await docEmployee.set(json);
81 }
82 }
83
```

Gambar 18. inputPage.dart bagian 3. Pemanggilan fungsi create yang telah disediakan firebase

17. Modifikasi main.dart sebagai berikut

```
main.dart x
lib > main.dart > HomePage
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import 'package:flutter/material.dart';
3 import 'package:flutter_slidable/flutter_slidable.dart';
4 import 'package:firebase/entity/employee.dart';
5 import 'package:firebase/inputPage.dart';
6 import 'package:firebase_core/firebase_core.dart';
7
8 Run | Debug | Profile
9 Future main() async {
10   WidgetsFlutterBinding.ensureInitialized();
11   await Firebase.initializeApp();
12   runApp(const MyApp());
13 }
14
15 class MyApp extends StatelessWidget {
16   const MyApp({super.key});
17
18   // This widget is the root of your application.
19   @override
20   Widget build(BuildContext context) {
21     return MaterialApp(
22       title: 'Firebase',
23       theme: ThemeData(
24         primarySwatch: Colors.blue,
25       ), // ThemeData
26       home: const HomePage(
27         title: 'Firebase',
28       ), // HomePage
29     ); // MaterialApp
30   }
31 }
```

Gambar 19. Main.dart bagian 1


```

main.dart x
lib > main.dart > MyApp > build
32 class HomePage extends StatefulWidget {
33   const HomePage({super.key, required this.title});
34
35   final String title;
36
37   @override
38   State<HomePage> createState() => _HomePageState();
39 }
40
41 class _HomePageState extends State<HomePage> {
42   @override
43   void initState() {
44     // refresh();
45     super.initState();
46   }
47
48   @override
49   Widget build(BuildContext context) {
50     return Scaffold(
51       appBar: AppBar(
52         title: Text("EMPLOYEE"),
53         actions: [
54           IconButton(
55             icon: Icon(Icons.add),
56             onPressed: () async {
57               Navigator.push(
58                 context,
59                 MaterialPageRoute(
60                   builder: (context) => const InputPage(
61                     title: 'INPUT EMPLOYEE',
62                     id: null,
63                     name: null,
64                     email: null)), // InputPage // MaterialPageRoute
65               );
66               // ).then(()) => refresh();
67             }, // IconButton

```

Gambar 20. Main.dart bagian 2

```

main.dart x
lib > main.dart > _HomePageState
51 appBar: AppBar(
52   title: Text("EMPLOYEE"),
53   actions: [
54     IconButton(
55       icon: Icon(Icons.add),
56       onPressed: () async {
57         Navigator.push(
58           context,
59           MaterialPageRoute(
60             builder: (context) => const InputPage(
61               title: 'INPUT EMPLOYEE',
62               id: null,
63               name: null,
64               email: null)), // InputPage // MaterialPageRoute
65         );
66         // ).then(()) => refresh();
67       }, // IconButton
68     ), // IconButton
69   ], // AppBar
70   body: StreamBuilder(
71     stream: getEmployee(),
72     builder: (context, snapshot) {
73       if (snapshot.hasError) {
74         Center(child: Text('Something went wrong'));
75       }
76       if (snapshot.hasData) {
77         final employees = snapshot.data!;
78         return ListView(
79           children: employees.map(buildEmployee).toList(),
80           // ListView
81         );
82       } else {
83         return Center(child: Text('NO DATA'));
84       }
85     )); // StreamBuilder // Scaffold
86 }
87

```

Gambar 21. Main.dart bagian 3



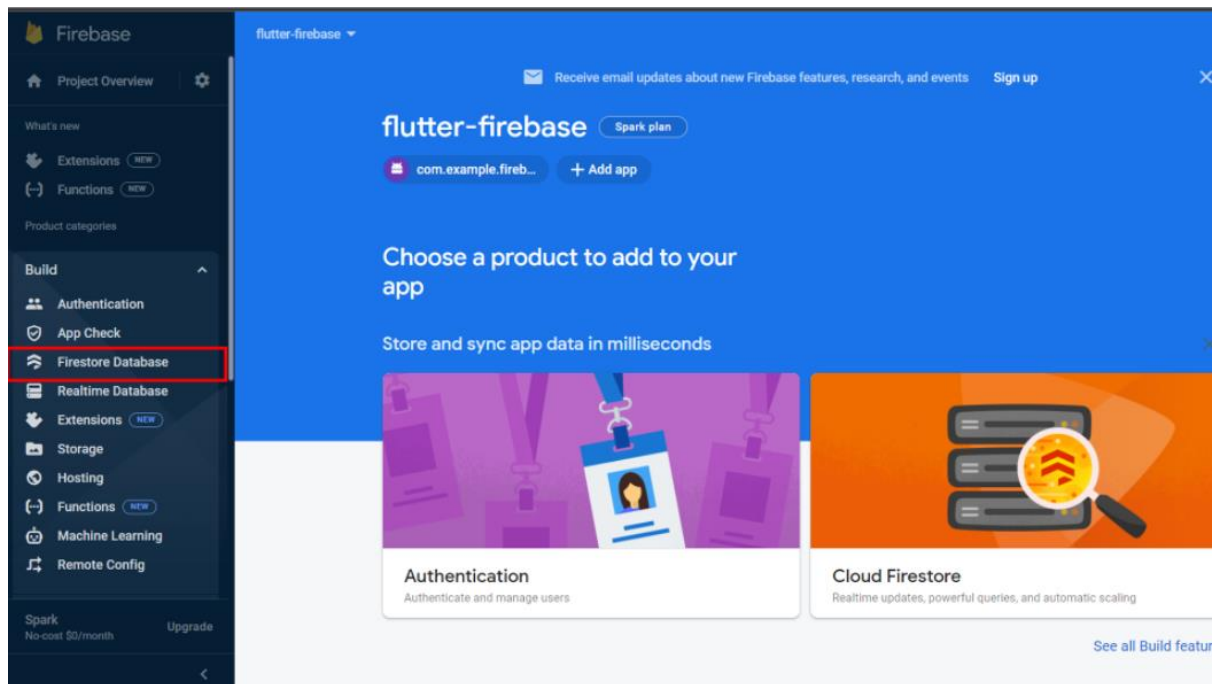
```
main.dart x
lib > main.dart > _HomePageState > buildEmployee
87
88   Widget buildEmployee(Employee employee) => Slidable(
89     child: ListTile(
90       title: Text(employee.name),
91       subtitle: Text(employee.email),
92     ), // ListTile
93     actionPane: SlidableDrawerActionPane(),
94     secondaryActions: [
95       IconSlideAction(
96         caption: 'Update',
97         color: Colors.blue,
98         icon: Icons.update,
99         onTap: () async {
100           Navigator.push(
101             context,
102             MaterialPageRoute(
103               builder: (context) => InputPage(
104                 title: 'INPUT EMPLOYEE',
105                 id: employee.id,
106                 name: employee.name,
107                 email: employee.email,
108               ), // InputPage // MaterialPageRoute
109             );
110           // .then(() => refresh());
111         },
112       ), // IconSlideAction
113       IconSlideAction(
114         caption: 'Delete',
115         color: Colors.red,
116         icon: Icons.delete,
117         onTap: () async {
118           final docEmployee = FirebaseFirestore.instance
119             .collection('employee')
120             .doc(employee.id);
121           docEmployee.delete();
122         },
123       ) // IconSlideAction
```

Gambar 22. Main.dart bagian 4

```
124     ],
125   ); // Slidable
126
127   Stream<List<Employee>> getEmployee() => FirebaseFirestore.instance
128     .collection('employee')
129     .snapshots()
130     .map((snapshot) =>
131       snapshot.docs.map((doc) => Employee.fromJson(doc.data())).toList());
132   }
133
```

Gambar 23. Main.dart bagian 5

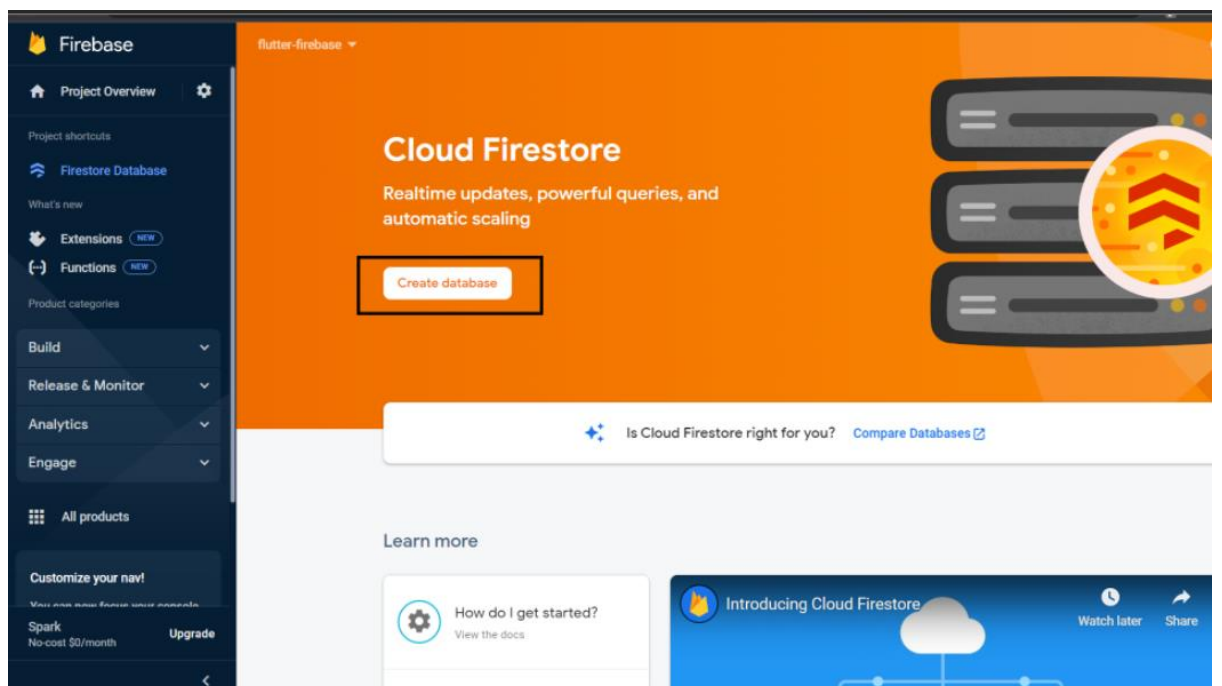
18. Buka kembali console firebase dan masuk ke project firebase yang telah dibuat sebelumnya lalu buat realtime database baru.



189

190

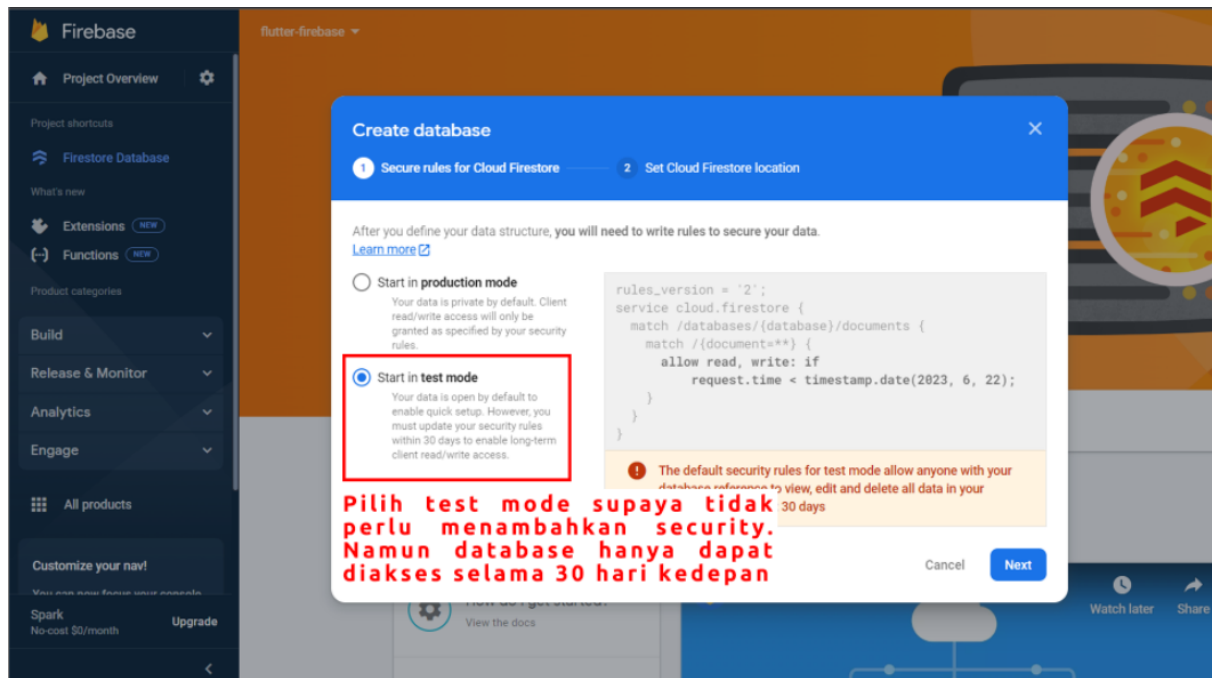
Gambar 24. Masuk ke opsi Firestore Database



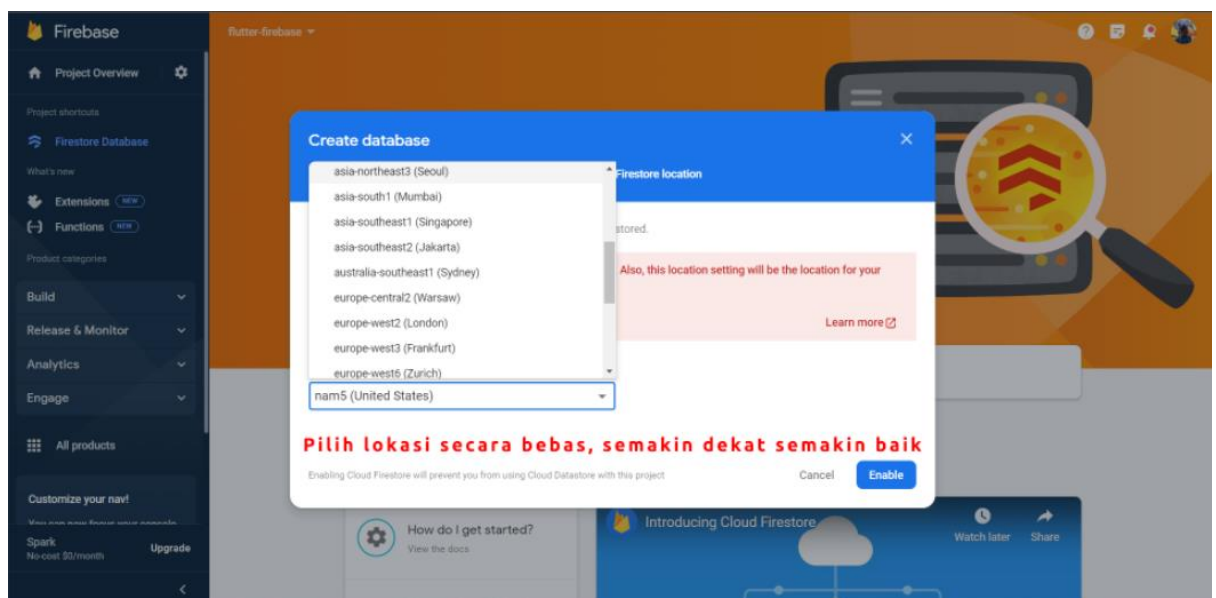
191

192

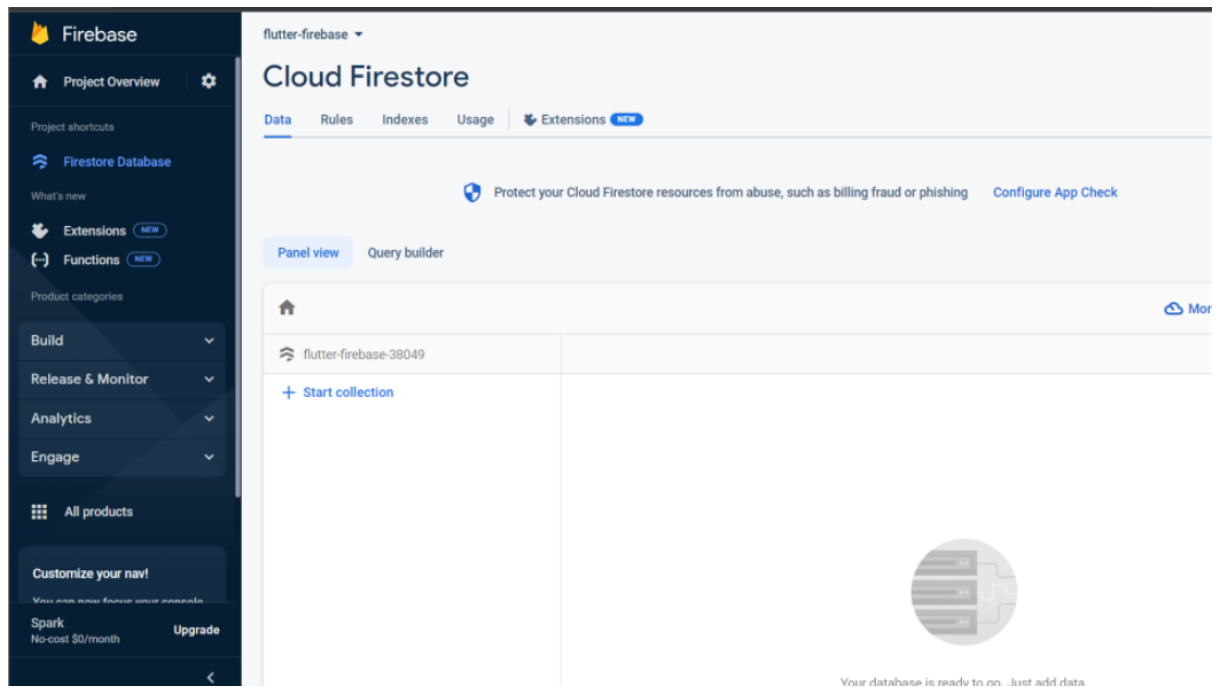
Gambar 25. Buat cloud firestore



Gambar 26. Pilih opsi test mode

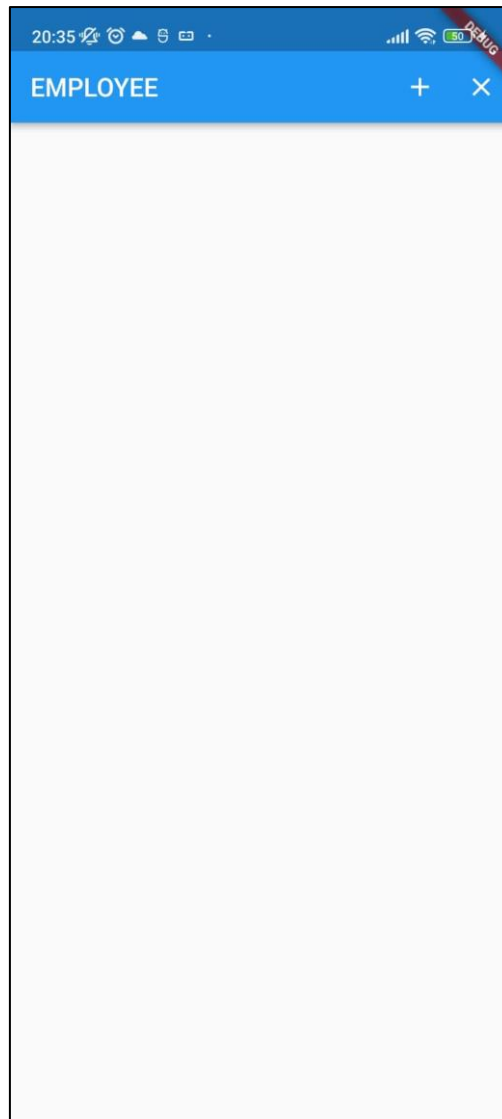


Gambar 27. Pilih lokasi database



Gambar 28. Firestore Database yang sudah jadi

19. Run aplikasi flutter.



Gambar 29. Hasil main.dart



20:35 50% Debug

← INPUT EMPLOYEE

Name

Email

Save

202

203

Gambar 30. Hasil inputPage.dart



8:33

← INPUT EMPLOYEE

Name
Paulus Pandu Windito

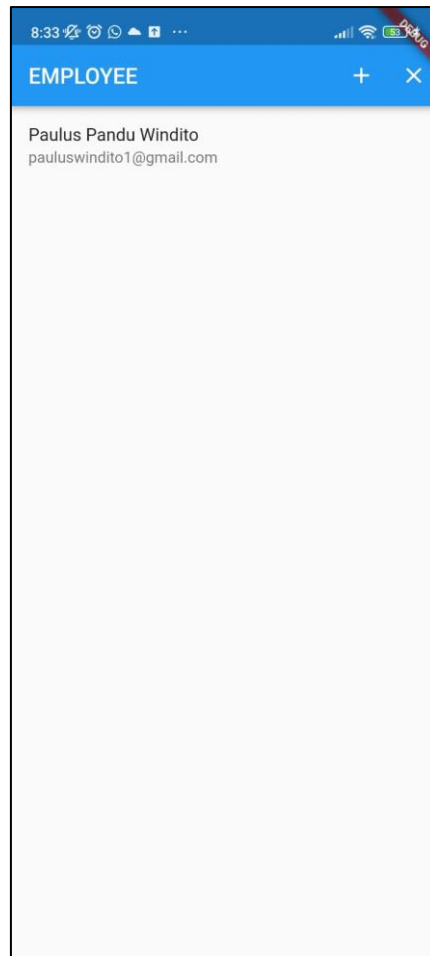
Email
pauluswindito1@gmail.com

Save

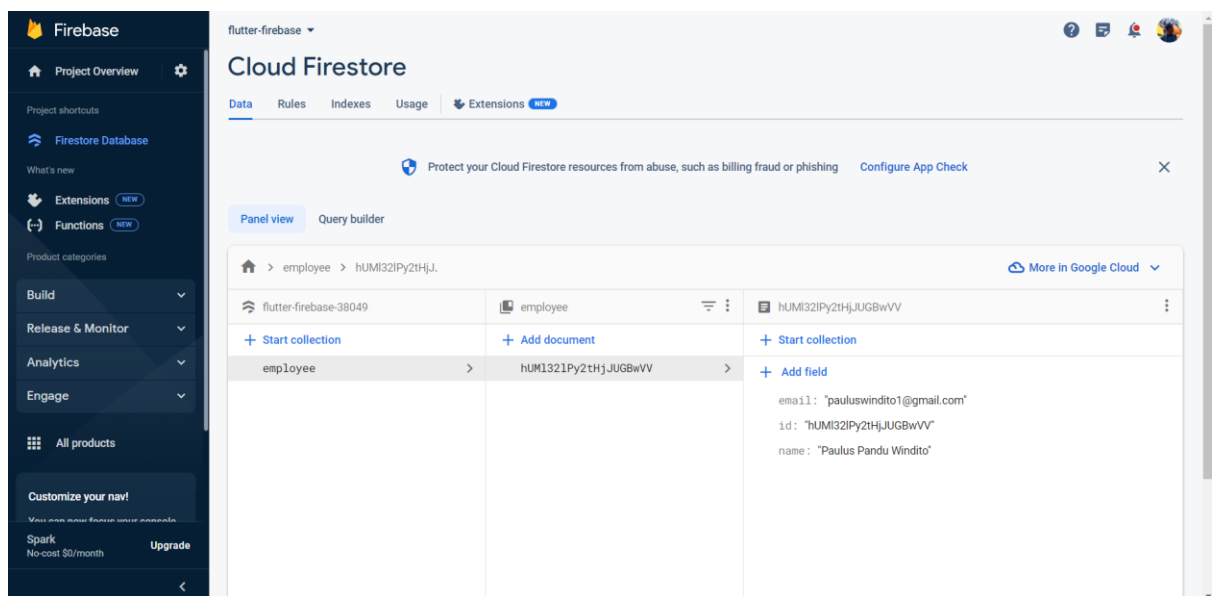
204

205

Gambar 31. inputPage.dart dengan data



Gambar 32. main.dart dengan data



Gambar 33. Firebase database dengan data



20.Data yang tertampil pada aplikasi flutter merupakan data realtime, sehingga apabila kita mengganti data langsung pada firestore, maka secara otomatis data yang tertampil pada aplikasi akan berubah.

21. Lakukan percobaan pada fungsi update dan delete dan pastikan fungsi sudah berjalan dengan baik.



240 **ATURAN Pengerjaan Guided :**

- 241 - **Guided dikerjakan selama waktu perkuliahan berlangsung.**
- 242 - **Penamaan projek guided harus sesuai dengan yang sudah**
- 243 **dicontohkan.**
- 244 - **Guided wajib diupload ke github dengan status private dengan**
- 245 **penamaan: `gd6_x_yyyy`**
- 246 ○ **x à kelas**
- 247 ○ **yyyyà 4 DIGIT TERAKHIR NPM**
- 248 - **Setelah diupload melalui github, jangan lupa untuk**
- 249 **mengumpulkan keseluruhan link file github melalui situs kuliah.**
- 250 - **Cara upload ke github, silahkan melihat pada modul **"UPLOAD****
- 251 **GITHUB"**.