Modul Praktikum

Pemrograman Mobile

10

AUTHENTICATION

Tujuan Pembelajaran

A. Mahasiswa memahami dan mengenal layanan authentication

B. Mahasiswa dapat mengimplementasikan layanan authentication ke dalam sebuah aplikasi android

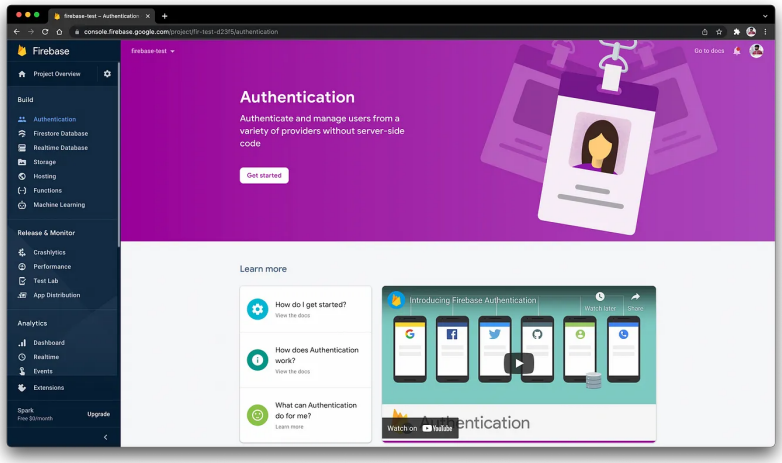
Dasar Teori

Authentication

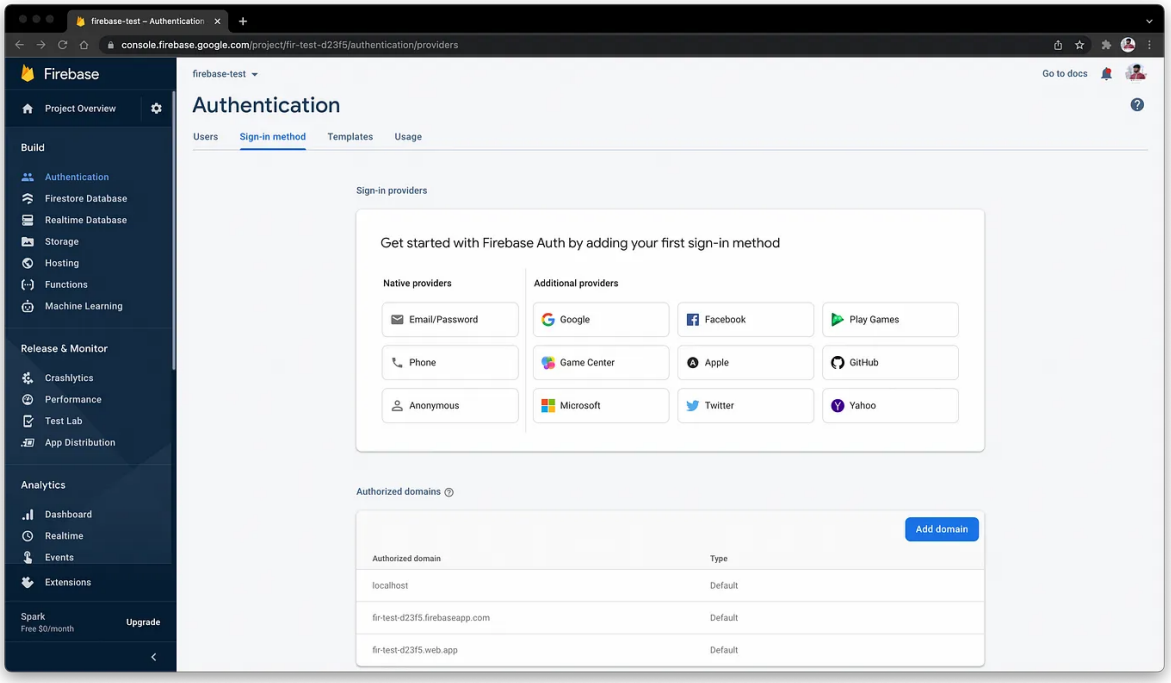
Firebase Authentication adalah salah satu layanan back-end, fitur Android dan iOS, SDK yang mudah digunakan, dan tampilan interfaces yang siap pakai untuk mengautentikasi pengguna ke aplikasi yang kamu buat. Firebase Authentication mendukung autentikasi menggunakan nomor telepon, sandi, penyedia identitas gabungan populer seperti seperti Google, Facebook, dan sebagainya.

Praktikum

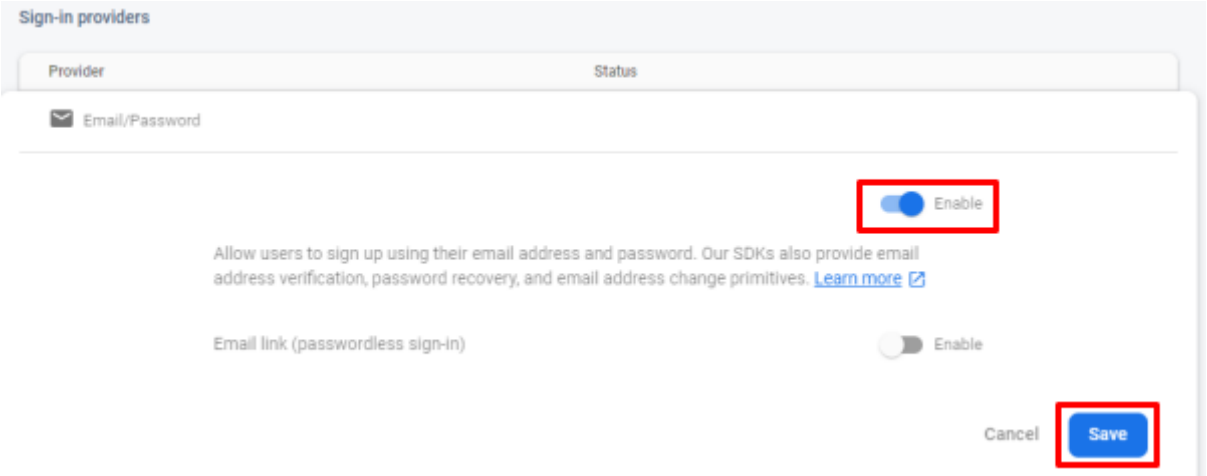
Langkah pertama yang harus dilakukan adalah "**Buat Projek Mobile di Firebase**". Pada modul kali ini tidak akan dijelaskan kembali untuk langkah pertama ini, silahkan cek modul sebelumnya. Setelah mengkonfigurasi projek mobile di firebase, pilih menu "**Authentication**" yang ada di *sidebar*, kemudian pilih "**get started**" untuk memulai.



Setelah memulai akan muncul list provider autentikasi yang bisa dipakai di aplikasi.



Pada praktik kali ini, pilih Email/Password dan Google sebagai metode yang digunakan autentikasi.



Untuk **mengaktifkan metode email/password** agar bisa digunakan, jangan lupa untuk **"enable"** kemudian **"save".**

Setelah langkah-langkah diatas telah diikuti, selanjutnya adalah menuliskan code pada program. *Package* yang digunakan pada praktikum adalah firebase_auth dan firebase_core.

- **Menambahkan form login dan register**

  Silahkan download source code DISINI. Lalu ekstrak dan pindahkan file tersebut kedalam folder **lib project Firebase kalian yang dikerjakan pada praktikum sebelumnya**.

- **Membuat Class Auth**

  Silahkan buat file baru bernama **AUTH.dart** pada folder **lib** lalu buatlah kode seperti berikut :

```
import 'package:firebase_auth/firebase_auth.dart';

class Auth {
  final FirebaseAuth _auth = FirebaseAuth.instance;

  Future<void> regis(String email, String password) async {
    final user = await _auth.createUserWithEmailAndPassword(
      email: email,
      password: password,
    );
  }

  Future<void> login(String email, String password) async {
    final user = await _auth.signInWithEmailAndPassword(
      email: email,
      password: password,
    );
  }
}
```

- **Register**

  Tambahkan method **handleSubmit()** pada class **Regis** (**DILUAR WIDGET BUILD**) untuk digunakan pada **onPressed** milik button **Submit** sebagai berikut :

```
handleSubmit()async{
   if (!_formKey.currentState!.validate()) return;
   final email = _ctrlEmail.value.text;
   final password = _ctrlPassword.value.text;

   setState(() => _loading = true);
   await Auth().regis(email, password);
   setState(() => _loading = false);
 }
```

  Lalu panggil method tersebut pada properti **onPressed** tombol submit :

```
ElevatedButton(
            onPressed: () => handleSubmit(),
            child: _loading ?
            const SizedBox(
              width: 20,
              height: 20,
              child: CircularProgressIndicator(
                color: Colors.white,
                strokeWidth: 2,
              ),
            )
            : Text("Submit"),
),
```

  *** Gunakan password minimal 5 karakter (huruf / angka) pada saat mencoba register akun.**

- **Login**

  Tambahkan method **handleSubmit()** pada class **Login** (**DILUAR WIDGET BUILD**) untuk digunakan pada **onPressed** milik button **Submit** sebagai berikut :

```
handleSubmit()async{
   if (!_formKey.currentState!.validate()) return;
   final email = _ctrlEmail.value.text;
   final password = _ctrlPassword.value.text;

   setState(() => _loading = true);
   await Auth().login(email, password);
   setState(() => _loading = false);
 }
```

  Lalu panggil method tersebut pada properti **onPressed** tombol submit :

```
ElevatedButton(
            onPressed: () => handleSubmit(),
            child: _loading ?
            const SizedBox(
              width: 20,
              height: 20,
              child: CircularProgressIndicator(
                color: Colors.white,
                strokeWidth: 2,
              ),
            )
            : Text("Submit"),
),
```

Kemudian ubah properti **home** pada class **MyApp** di file main.dart menjadi seperti berikut :

```
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: StreamBuilder<User?>(
        stream: FirebaseAuth.instance.authStateChanges(),
        builder: (context,snapshot){
          if (snapshot.hasData){
            return MyHomePage();
          }else{
            return Login();
          }
        },
      ),
    );
  }
}
```

- **Logout**

  Tambahkan tombol Logout di bagian **actions** pada **AppBar** yang ada di File **MyHomePage.dart** sebagai berikut :

```
actions: [
        TextButton(
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => LihatData(),
              ),
            );
          },
          child: Text(
            "Lihat Data",
            style: TextStyle(
              color: Colors.black,
            ),
          ),
        ),
        TextButton(
        child: Text(
          "Logout",
          style: TextStyle(
            color: Colors.red,
            fontWeight: FontWeight.bold,
          ),
        ),
        onPressed: () {
          FirebaseAuth.instance.signOut();
        },
      ),
      ],
```

NOTE :

- Step by Step diatas merupakan **dasar** dari materi Firebase Authentication. Silahkan anda **kombinasikan** dengan materi-materi pada modul sebelumnya.
- Berikut adalah link referensi bila anda ingin menambahkan error handling**.**