

LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK



JUDUL:

Authentikasi Firebase, Google, and Phone

Disusun oleh:

Suryani Dewi Wulandari (21102119)

TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
BANYUMAS, JAWA TENGAH
2024

Pembahasan

Firebase adalah platform pengembangan aplikasi yang disediakan oleh Google. Ini memungkinkan pengembang untuk membuat aplikasi berkualitas tinggi dengan berbagai fitur, termasuk otentikasi pengguna, penyimpanan data, analisis, pelaporan kesalahan, dan masih banyak lagi.

Salah satu fitur utama Firebase adalah **Realtime Database**, sebuah layanan database cloud yang menyimpan data dalam format JSON dan menyelesaikan secara langsung ke perangkat setiap kali data berubah. Ini memungkinkan pengembang untuk membangun aplikasi yang responsif tanpa harus menulis kode backend atau mengelola infrastruktur server.

Firebase juga menyediakan solusi **Authentication** yang kuat dan mudah digunakan, mendukung otentikasi email/password, otentikasi dengan penyedia media sosial seperti Google, Facebook, Twitter, dan lainnya, serta otentikasi dengan nomor ponsel. Selain itu, Firebase memiliki **Cloud Firestore**, sebuah database cloud yang menawarkan pengembang performa dan skala yang baik, serta model data yang lebih kaya, skalabilitas yang lebih baik, dan query yang lebih kuat dibandingkan dengan Realtime Database.

Firebase juga menyediakan layanan **Storage** untuk menyimpan file, seperti gambar, video, dan dokumen, secara aman di cloud. Ini memudahkan aplikasi untuk menyimpan dan mengelola file dengan mudah. Dengan **Cloud Functions**, pengembang dapat membuat dan mengelola backend aplikasi tanpa harus mengelola server sendiri. Mereka dapat menulis fungsi kecil yang diaktifkan oleh peristiwa di Firebase, seperti pembaruan data di database, atau pengunggahan file ke Firebase Storage.

Firebase juga memiliki fitur seperti **Analytics** untuk memberikan wawasan tentang kinerja aplikasi, **Cloud Messaging** untuk mengirim pemberitahuan push ke pengguna, **Remote Config** untuk mengonfigurasi aplikasi secara dinamis dari cloud, **Performance Monitoring** untuk melacak kinerja aplikasi, dan **Crashlytics** untuk melacak dan menganalisis crash aplikasi. Dengan fitur-fitur ini, Firebase memberikan infrastruktur yang kuat bagi pengembang

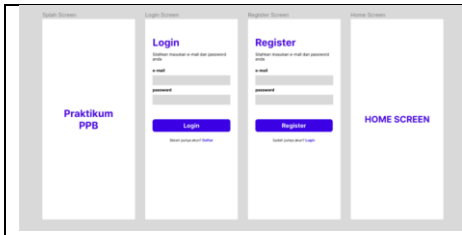
untuk membuat aplikasi yang tangguh, interaktif, dan mudah dielola.

Langkah-Langkah Praktikum

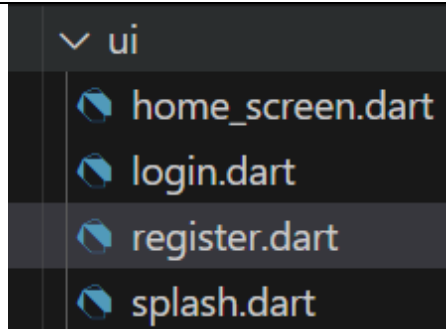
Langkah Praktikum	Pembahasan
--------------------------	-------------------

	Jika anda pengguna baru, maka perlu membuat akun terlebih dahulu di situs firebase									
<p>Windows</p> <p>Anda dapat menginstal Firebase CLI untuk Windows menggunakan salah satu opsi berikut:</p> <table><thead><tr><th>Opsi</th><th>Deskripsi</th><th>Disarankan untuk...</th></tr></thead><tbody><tr><td>biner mandiri</td><td>Download biner mandiri untuk CLI. Selanjutnya, Anda dapat mengasas file yang dapat dieksekusi untuk membuka shell guna menjalankan perintah Firebase.</td><td>Developer baru</td></tr><tr><td>npm</td><td>Gunakan npm (Pengelola Paket Node) untuk menginstal CLI dan mengaktifkan perintah Firebase yang tersedia secara global.</td><td>Developer yang tidak menggunakan atau belum familiar dengan Node.js</td></tr></tbody></table>	Opsi	Deskripsi	Disarankan untuk...	biner mandiri	Download biner mandiri untuk CLI. Selanjutnya, Anda dapat mengasas file yang dapat dieksekusi untuk membuka shell guna menjalankan perintah Firebase.	Developer baru	npm	Gunakan npm (Pengelola Paket Node) untuk menginstal CLI dan mengaktifkan perintah Firebase yang tersedia secara global.	Developer yang tidak menggunakan atau belum familiar dengan Node.js	Langkah selanjutnya adalah mengunduh firebase CLI dan node.js sesuai dengan sistem operasi masing-masing.
Opsi	Deskripsi	Disarankan untuk...								
biner mandiri	Download biner mandiri untuk CLI. Selanjutnya, Anda dapat mengasas file yang dapat dieksekusi untuk membuka shell guna menjalankan perintah Firebase.	Developer baru								
npm	Gunakan npm (Pengelola Paket Node) untuk menginstal CLI dan mengaktifkan perintah Firebase yang tersedia secara global.	Developer yang tidak menggunakan atau belum familiar dengan Node.js								
	setelah berhasil diunduh, selanjutnya adalah melakukan penginstallan node.js									
	Kemudia lakukan pengInstallan Firebase CLI melalui npm dengan menjalankan perintah berikut: <code>npm install -g firebase-tools</code>									
	Setelah langkah installasi selesai, kemudian buatlah project baru pada situs firebase									
	Pada project yang sudah dibuat, tambahkan flutter dengan mengklik logo flutter									
	Langkah pertama dalam menambahkan flutter ke firebase adalah dengan masuk ke firebase lewat command prompt dengan mengetikan perintah “firebase login”.									

<p>Woohoo!</p> <p>Firestore CLI Login Successful</p> <p>You are logged in to the Firebase Command-Line interface. You can immediately close this window and continue using the CLI.</p>	<p>Setelah login ke <i>firebase</i> melalui <i>command prompt</i>, jika baru pertama kali melakukan, maka akan diarahkan untuk masuk ke akun yang terhubung dengan <i>firebase CLI</i>. Setelah melakukan tahapan-tahapan sebelumnya, halaman sukses akan muncul seperti gambar di samping.</p>
	<p>Membuat project flutter baru, dengan cara membuat folder untuk menyimpan. Kemudian pada terminal berikan perintah “flutter create”</p>
	<p>menjalankan perintah “dart pub global activate flutterfire_cli” membuat tool flutterfire dapat diakses dari mana saja di terminal</p>
	<p>Perintah <code>flutterfire configure --project=prakflutterfirebase-38bb6</code> digunakan untuk mengkonfigurasi firebase ke dalam project flutter.</p>
	<p>Project berhasil dikonfigurasi. Dan berhasil mendapatkan firebase AppId</p>
	<p>Menambahkan dependensi yang dibutuhkan dengan menggunakan command “flutter pub add firebase_core”</p>
	<p>Pada file main.dart ubah void main menjadi seperti kode di samping</p>



Desain ui yang akan diterapkan pada aplikasi



Pertama, siapkan folder ui terlebih dahulu dan buat 4 file dart seperti gambar di samping

```
import 'dart:async';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';

class SplashScreen extends StatefulWidget {
  const SplashScreen({Key? key}) : super(key: key);

  @override
  State<SplashScreen> createState() => _SplashScreenState();
}

class _SplashScreenState extends State<SplashScreen> {
  @override
  void initState() {
    Timer(Duration(seconds: 3), () => Navigator.pushNamed(context, '/login'));
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              "Praktikum",
              style: TextStyle(
                color: Color(0xFF3D4DE0), // TextStyle
                fontWeight: FontWeight.bold,
                fontSize: 40,
              ), // Text
            ),
            Text(
              "PPB",
              style: TextStyle(
                color: Color(0xFF3D4DE0), // TextStyle
                fontWeight: FontWeight.bold,
                fontSize: 40,
              ), // Text
            ),
          ], // Column
        ), // Center
      ), // Scaffold
    );
  }
}
```

Tambahkan kode di samping pada file splash.dart

```
body: Center(
  child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    crossAxisAlignment: CrossAxisAlignment.center,
    children: [
      Text(
        "Praktikum",
        style: TextStyle(
          color: Color(0xFF3D4DE0), // TextStyle
          fontWeight: FontWeight.bold,
          fontSize: 40,
        ), // Text
      ),
      Text(
        "PPB",
        style: TextStyle(
          color: Color(0xFF3D4DE0), // TextStyle
          fontWeight: FontWeight.bold,
          fontSize: 40,
        ), // Text
      ),
    ], // Column
  ), // Center
), // Scaffold
);
```

```
lib > ui > login.dart > ...
DXYULIAH SEMESTER 6 PEMROGRAMAN PERANGKAT ...
BERGURUKANAuthlibbasepraktikumauthlib
class LoginScreen extends StatefulWidget {
  const LoginScreen({Key? key}) : super(key: key);
  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final emailEdc = TextEditingController();
  final passEdc = TextEditingController();
  bool passInvisible = false;
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        margin: EdgeInsets.symmetric(horizontal: 30, vertical: 70),
        child: ListView(
          children: [
            Text(
              "login",
              style: TextStyle(
                color: Color(0xFF3D4DE0), // TextStyle
                fontWeight: FontWeight.bold,
                fontSize: 40,
              ), // Text
            ),
          ],
        ),
      ),
    );
  }
}
```

Tambahkan kode di samping pada bagian file login.dart

```

47       SizedBox(
48         height: 15,
49       ), // SizedBox
50       Text(
51         "Silahkan masukan e-mail dan password anda",
52         style: TextStyle(
53           fontSize: 16,
54         ), // TextStyle
55       ), // Text
56       SizedBox(
57         height: 25,
58       ), // SizedBox
59       Text(
60         "e-mail",
61         style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
62       ), // Text
63       TextFormField(
64         controller: emailEdc,
65       ), // TextFormField
66       SizedBox(
67         height: 10,
68       ), // SizedBox
69       Text(
70         "password",

```

```

71       ), // Text
72       TextFormField(
73         controller: passEdc,
74         decoration: InputDecoration(
75           suffixIcon: IconButton(
76             icon: Icon(
77               passInvisible ? Icons.visibility : Icons.visibility_off,
78             ),
79             onPressed: () {
80               setState(() {
81                 passInvisible = !passInvisible;
82               });
83             },
84           ), // IconButton
85         ), // InputDecoration
86         obscureText: !passInvisible,
87       ), // TextFormField
88       SizedBox(
89         height: 50,
90       ), // SizedBox
91       ElevatedButton(
92         onPressed: () {
93           Navigator.pushNamed(context, "/home");
94         },

```

```

95       ), // ElevatedButton
96       Row(
97         mainAxisAlignment: MainAxisAlignment.center,
98         children: [
99           Text("Belum punya akun ?"),
100           TextButton(
101             onPressed: () {
102               Navigator.pushNamed(context, "/register");
103             },
104             child: Text(
105               "Daftar",
106               style: TextStyle(
107                 fontWeight: FontWeight.bold,
108                 color: Color(0xff3D4DE0), // TextStyle
109               ), // Text // TextButton
110             ),
111           ), // Row
112         ], // ListView
113       ), // Container
114     ); // Scaffold
115   }

```

```

import 'package:flutter/material.dart';

class RegisterScreen extends StatefulWidget {
  const RegisterScreen({Key? key}) : super(key: key);
  @override
  State<RegisterScreen> createState() => _RegisterScreenState();
}

class _RegisterScreenState extends State<RegisterScreen> {
  final emailEdc = TextEditingController();
  final passEdc = TextEditingController();
  bool passInvisible = false;
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        margin: EdgeInsets.symmetric(horizontal: 30, vertical: 70),
        child: ListView(
          children: [
            Text(
              "Register",
              style: TextStyle(
                fontSize: 40,
                fontWeight: FontWeight.bold,
                color: Color(0xff3D4DE0), // TextStyle

```

Tambahkan kode di samping pada bagian file register.dart

```

        SizedBox(
          height: 15,
        ), // SizedBox
        Text(
          "silahkan masukan e-mail dan password anda",
          style: TextStyle(
            fontSize: 16,
          ), // TextStyle
        ), // Text
        SizedBox(
          height: 25,
        ), // SizedBox
        Text(
          "e-mail",
          style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
        ), // Text
        TextFormField(
          controller: emailEdc,
        ), // TextFormField
        SizedBox(
          height: 10,
        ), // SizedBox
        Text(
          "password",

```

```

        ), // Text
        TextFormField(
          controller: passEdc,
          decoration: InputDecoration(
            suffixIcon: IconButton(
              icon: Icon(
                passInvisible ? Icons.visibility_off : Icons.visibility,
                size: 24,
              ),
              onPressed: () {
                setState(() {
                  passInvisible = !passInvisible; // Toggle isPasswordVisible ketika i
                });
              },
            ), // IconButton
          ), // InputDecoration
          obscureText:
            !passInvisible, // Atur obscureText berdasarkan isPasswordV
        ), // TextFormField
        SizedBox(
          height: 50,
        ), // SizedBox
        ElevatedButton(
          onPressed: () {},

```

```

          height: 25,
        ), // SizedBox
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text("Sudah punya akun?"),
            TextButton(
              onPressed: () {
                Navigator.pushNamed(context, '/login');
              },
              child: Text(
                "Login",
                style: TextStyle(
                  fontWeight: FontWeight.bold,
                  color: Color(0xff3D4DE8), // TextStyle
                ), // Text // TextButton
              ),
            ), // Row
          ], // ListView
        ), // Container
      ); // Scaffold
    }
  }
}

```

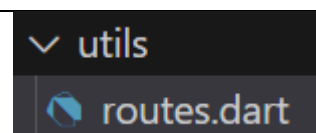
```

import 'package:flutter/material.dart';

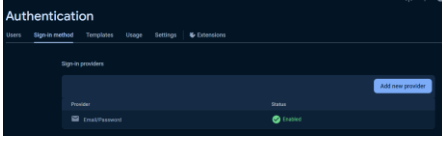
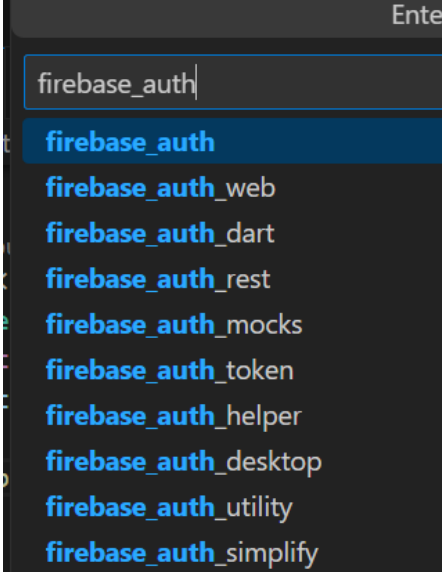
class HomeScreen extends StatelessWidget {
  const HomeScreen({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: Text(
          "HOME SCREEN",
          style: TextStyle(
            fontSize: 40,
            fontWeight: FontWeight.bold,
            color: Color(0xff3D4DE8), // TextStyle
          ), // Text // Center
        ), // Scaffold
      );
    }
  }
}

```

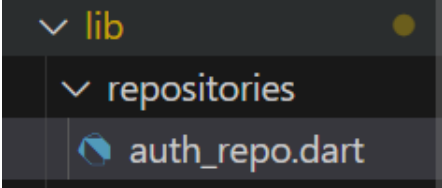
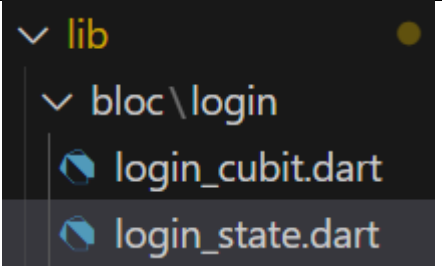
Tambahkan kode di samping pada bagian home_screen.dart

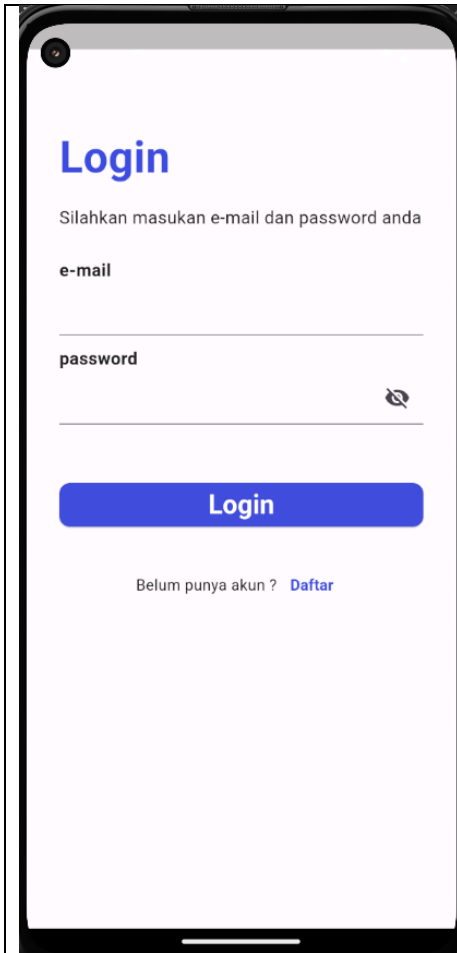


Selanjutnya kita perlu mendaftarkan state yang sudah kita buat dalam route dengan cara buat folder “utils” di dalam folder “lib”, dan buat file “routes.dart” di dalam folder “utils”.

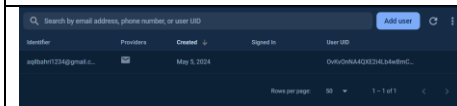
<pre> MaterialPageRoute _pageRoute({required Widget body, required RouteSettings settings}) => MaterialPageRoute(builder: (_) => body, settings: settings); Route? generateRoute(RouteSettings settings) { Route? _route; final _args = settings.arguments; switch (settings.name) { case rLogin: _route = _pageRoute(body: LoginScreen(), settings: settings); break; case rRegister: _route = _pageRoute(body: RegisterScreen(), settings: settings); break; case rHome: _route = _pageRoute(body: HomeScreen(), settings: settings); break; } return _route; } final NAV_KEY = GlobalKey<NavigatorState>(); const String rLogin = '/login'; const String rRegister = '/register'; const String rHome = '/home'; </pre>	<p>Tambahkan kode di samping pada bagian file routes.dart</p>
<pre> class MyApp extends StatelessWidget { const MyApp({Key? key}) : super(key: key); @override Widget build(BuildContext context) { return MaterialApp(title: "Praktikum 6", debugShowCheckedModeBanner: false, navigatorKey: NAV_KEY, generateRoute: generateRoute, home: SplashScreen(),); // MaterialApp } } </pre>	<p>Tambahkan kode di samping pada bagian file main.dart</p>
	<p>Menambahkan autentikasi email dan password pada dashboard firebase di project kita.</p>
	<p>Menginstal dependensi firebase_auth</p>
<pre> cupertino_icons: ^1.0.6 firebase_core: ^2.30.1 firebase_auth: ^4.19.4 firebase_auth_rest: ^2.0.4 </pre>	<p>Cek ketersediaan dependensi firebase_auth pada pubspec.yaml</p>
<pre> bloc: ^8.1.2 flutter_bloc: ^8.1.3 </pre>	<p>Untuk memproses logika bisnis pada aplikasi kita akan menggunakan flutter bloc,</p>

	sehingga perlu menambahkan 2 dependensi yaitu : bloc: ^8.1.2 flutter_bloc: ^8.1.3 jangan lupa untuk menjalankan perintah "flutter pub upgrade"
--	---

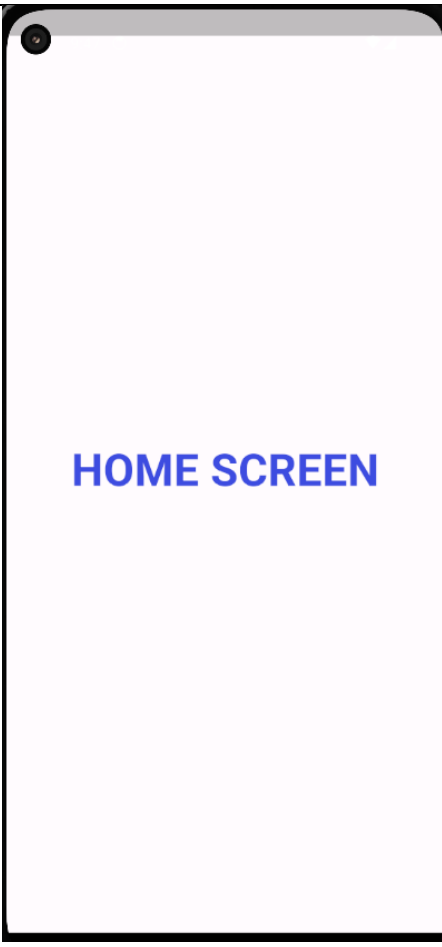
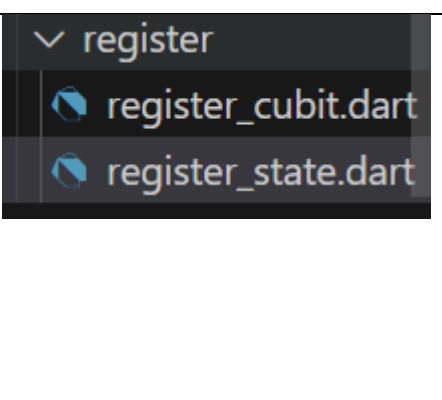
 <p>Gambar 1. menambahkan file baru pada folder baru repositories</p>	<p>Buatlah sebuah direktori bernama "repositories" di dalam folder "lib", dan selanjutnya tambahkan file bernama "auth_repo.dart" di dalamnya. Folder "repositories" ini akan digunakan untuk menghubungkan aplikasi dengan Firebase atau API dari backend.</p>
<pre>import 'package:firebase_auth/firebase_auth.dart'; import 'package:firebase_core/firebase_core.dart'; class AuthRepo { final _auth = FirebaseAuth.instance; Future<void> login((required String email, required String password)) async { try { final user = await _auth.signInWithEmailAndPassword(email: email, password: password); } on FirebaseAuthException catch (e) { throw e.message ?? 'Something wrong!'; } catch (e) { throw e; } } Future<void> register((required String email, required String password)) async { try { final user = await _auth.createUserWithEmailAndPassword(email: email, password: password); } on FirebaseAuthException catch (e) { throw e.message ?? 'Something wrong!'; } catch (e) { throw e; } } }</pre>	<p>Kita akan membuat 2 buah Future<void> dalam class AuthRepo untuk login dan register pada file auth_repo.dart.</p>
	<p>Untuk mengatur Bloc untuk login, pertama kita perlu membuat folder dengan struktur seperti yang ditunjukkan pada gambar di samping. Setelah itu, kita akan membuat file login_cubit.dart di dalam folder bloc.</p>
<pre>part of 'login_cubit.dart'; import 'package:flutter/material.dart'; @immutable abstract class LoginState {} class LoginInitial extends LoginState {} class LoginLoading extends LoginState {} class LoginSuccess extends LoginState { final String msg; LoginSuccess(this.msg); } class LoginFailure extends LoginState { final String msg; LoginFailure(this.msg); }</pre>	<p>Tambahkan kode di samping pada file login_state.dart</p>
<pre>import 'package:bloc/bloc.dart'; import 'package:meta/meta.dart'; import '../repositories/auth_repo.dart'; part 'login_state.dart'; class LoginCubit extends Cubit<LoginState> { LoginCubit() : super(LoginInitial()); final _repo = AuthRepo(); void login((required String email, required String password)) async { emit(LoginLoading()); try { await _repo.login(email: email, password: password); emit(LoginSuccess('Login berhasil!')); } catch (e) { print(e); emit(LoginFailure(e.toString())); } } }</pre>	<p>Tambahkan kode di samping pada file login_cubit.dart</p>

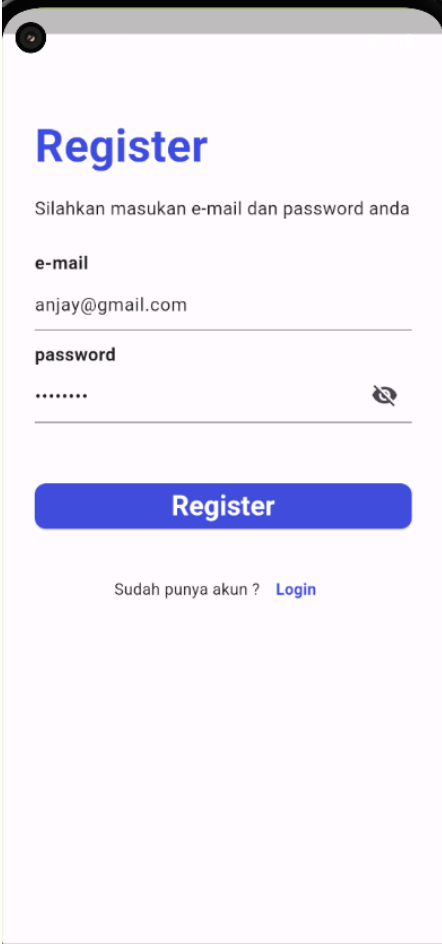
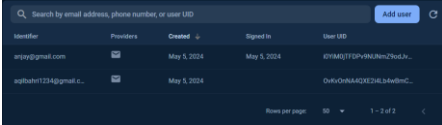


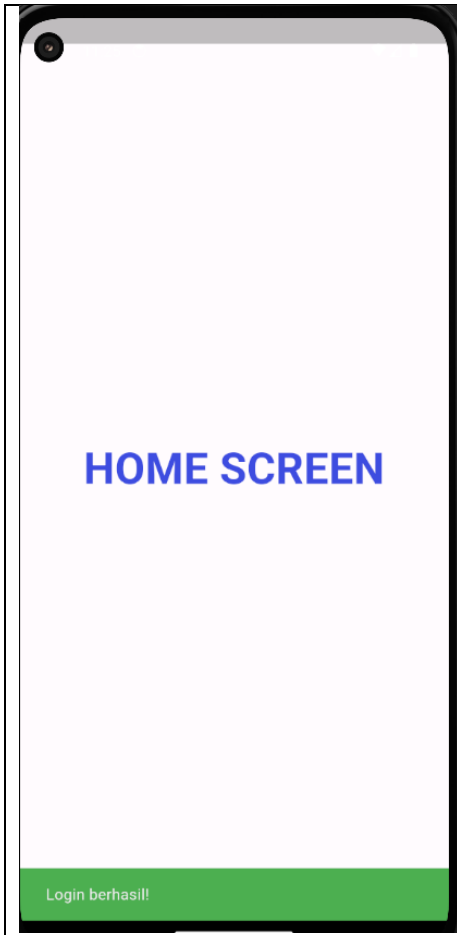
Untuk menjalankan program tersebut, Anda perlu memastikan bahwa inputan email dan password yang dimasukkan oleh pengguna disimpan dalam variabel emailEdc dan passEdc. Kemudian, Anda harus menggunakan Bloc untuk mengirim inputan tersebut ke Firebase melalui repositori yang telah dibuat sebelumnya. Jika proses login berhasil, pengguna akan diarahkan ke home screen. Namun, jika terjadi kegagalan, pesan kesalahan akan ditampilkan kepada pengguna. Pastikan juga untuk menangani semua kemungkinan skenario, seperti inputan tidak valid atau koneksi internet yang terputus.



Mengatur untuk bagian register di firebase

	<p>Data yang baru ditambahkan akan dicek untuk diisi pada bagian login.</p>
	<p>Setelah berhasil membuat fitur login, langkah selanjutnya adalah membuat fitur register. Karena kita sudah memiliki <code>Future<void> register</code>, selanjutnya kita akan mengatur blok untuk fitur tersebut. Buatlah folder "register" di dalam folder "bloc".</p>
<pre>part of 'register_cubit.dart'; import 'package:flutter/material.dart'; @immutable abstract class RegisterState {} class RegisterInitial extends RegisterState {} class RegisterLoading extends RegisterState {} class RegisterSuccess extends RegisterState { final String msg; RegisterSuccess(this.msg); } class RegisterFailure extends RegisterState { final String msg; RegisterFailure(this.msg); }</pre>	<p>Tambahkan kode di samping pada bagian file <code>register_state.dart</code></p>

<pre>import 'package:bloc/bloc.dart'; import 'package:meta/meta.dart'; import '../repositories/auth_repo.dart'; part 'register_state.dart'; class RegisterCubit extends Cubit<RegisterState> { RegisterCubit() : super(RegisterInitial()); final _repo = AuthRepo(); Future<void> register({required String email, required String password}) async { emit(RegisterLoading()); try { await _repo.register(email: email, password: password); emit(RegisterSuccess('Berhasil!')); } catch (e) { print(e); emit(RegisterFailure(e.toString())); } } }</pre>	<p>Tambahkan kode di samping pada bagian file register_cubit.dart</p>
	<p>Mencoba register</p>
	<p>User yang baru didaftarkan berhasil ditambahkan</p>



Mencoba login dengan akun yang baru saja di tambahkan