

LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK



JUDUL:

Authentikasi Firebase, Google, and Phone

Disusun oleh:

M. Nur Aqil Bahri (21102144)

TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
BANYUMAS, JAWA TENGAH
2024

Pembahasan

Firestore

Autentikasi Firestore adalah layanan penting bagi pengembang aplikasi mobile yang ingin membangun aplikasi yang aman dan mudah digunakan. Layanan ini menyediakan sistem login dan registrasi pengguna yang handal, dengan berbagai metode autentikasi yang fleksibel dan fitur-fitur canggih untuk meningkatkan keamanan dan pengalaman pengguna.

a. Manfaat Utama Autentikasi Firestore:

- 1) Meningkatkan Keamanan: Firestore Authentication menggunakan teknologi enkripsi dan otentikasi mutakhir untuk melindungi data pengguna dari akses tidak sah. Hal ini termasuk enkripsi kata sandi, verifikasi dua faktor, dan deteksi penipuan.
- 2) Mempermudah Integrasi: Firestore Authentication terintegrasi dengan mulus dengan Firestore lainnya, seperti Cloud Firestore dan Realtime Database, sehingga memudahkan pengelolaan data pengguna dan membangun aplikasi yang kompleks.
- 3) Meningkatkan Skalabilitas: Firestore Authentication dapat menangani aplikasi dengan jumlah pengguna yang besar secara efisien, tanpa perlu khawatir tentang infrastruktur server.
- 4) Menyediakan Fitur Lengkap: Firestore Authentication menawarkan berbagai fitur canggih, seperti manajemen sesi, pemulihan kata sandi, reset password, dan integrasi dengan penyedia identitas pihak ketiga.
- 5) Menghemat Waktu dan Biaya: Menggunakan Firestore Authentication dapat menghemat waktu dan biaya pengembangan yang terkait dengan membangun sistem autentikasi kustom.

- 6) Mempermudah Pengelolaan Pengguna: Firebase Authentication menyediakan dashboard intuitif untuk mengelola akun pengguna, termasuk menambahkan, menghapus, dan mengubah informasi pengguna.
- 7) Mendukung Berbagai Platform: Firebase Authentication tersedia untuk berbagai platform pengembangan mobile populer, seperti Android, iOS, React Native, Flutter, dan Unity.
- 8) Memiliki Komunitas Besar: Firebase memiliki komunitas pengembang yang besar dan aktif, sehingga mudah untuk menemukan bantuan dan dukungan saat menggunakan layanan ini.

Autentikasi Firebase adalah solusi komprehensif dan fleksibel untuk kebutuhan autentikasi pengguna dalam aplikasi mobile. Layanan ini menawarkan kombinasi keamanan yang kuat, kemudahan penggunaan, skalabilitas, dan fitur-fitur canggih yang menjadikannya pilihan ideal bagi pengembang yang ingin membangun aplikasi mobile yang handal dan aman.

Dengan menggunakan Firebase Authentication, pengembang dapat fokus pada pengembangan fitur inti aplikasi mereka, tanpa perlu khawatir tentang kompleksitas sistem autentikasi. Hal ini memungkinkan mereka untuk menghadirkan aplikasi yang lebih aman, mudah digunakan, dan menarik bagi penggunanya.

Langkah-Langkah Praktikum

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



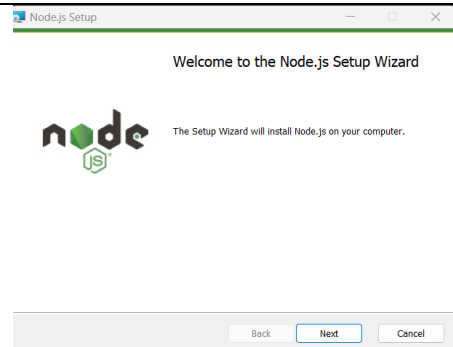
Gambar 1. Halaman Awal Situs Firebase

Membuat akun di *firebase*.



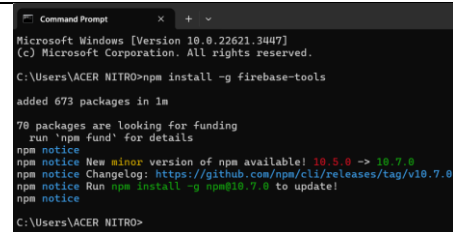
Gambar 2. Install Firebase CLI

Meng-*install firebase CLI*. Dengan mengunduh *node.JS*.



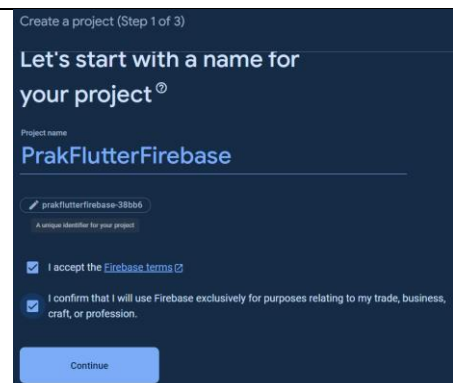
Gambar 3. Instalasi Node.JS

Install node.js






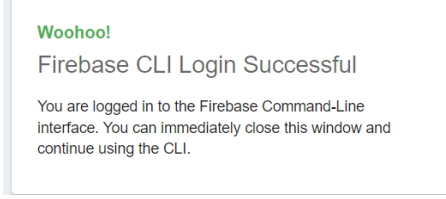
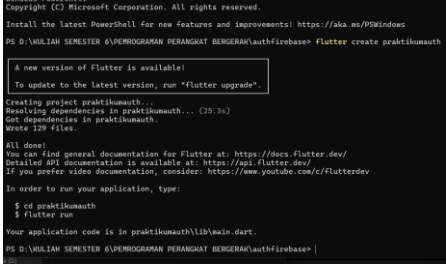
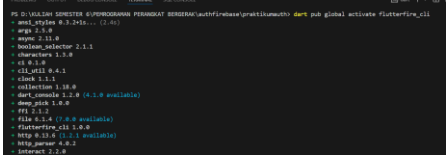
Gambar 4. Install CLI melalui npm


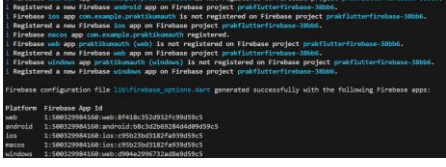
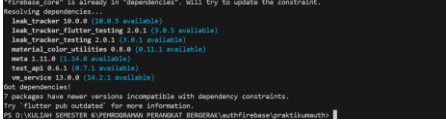
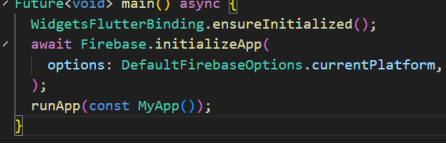
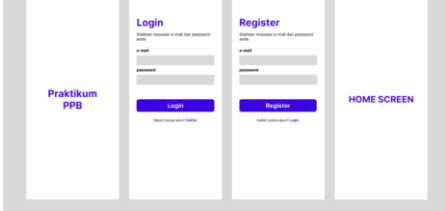
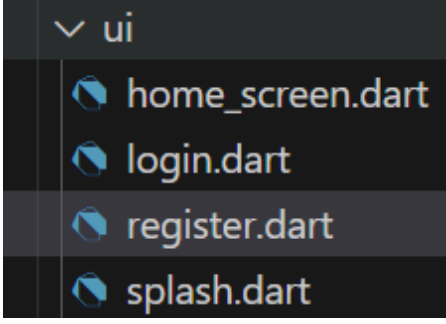
Instal *Firebase CLI* melalui *npm* dengan menjalankan perintah berikut:
npm install -g firebase-tools



Gambar 5. Tampilan membuat project *firebase*

Membuat Project baru di *firebase*.

 <p>Gambar 6. Tampilan setelah membuat project baru</p>	<p>Menambahkan <i>flutter</i> ke <i>firebase</i>.</p>
 <p>Gambar 7. Perintah <i>firebase login</i> di <i>command prompt</i></p>	<p>Langkah pertama dalam menambahkan <i>flutter</i> ke <i>firebase</i> adalah dengan masuk ke <i>firebase</i> lewat <i>command prompt</i> dengan mengetikkan perintah “<i>firebase login</i>”.</p>
 <p>Gambar 8. login ke <i>firebase CLI</i></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>.</p>
 <p>Gambar 9. Tampilan sukses terhubung dengan <i>firebase CLI</i></p>	<p>Setelah melakukan tahapan-tahapan sebelumnya, halaman sukses akan muncul seperti gambar di samping.</p>
 <p>Gambar 10. Membuat project flutter baru.</p>	<p>Membuat project flutter baru.</p>
 <p>Gambar 11. mengaktifkan paket <i>flutterfire_cli</i> secara global</p>	<p>menjalankan perintah “<i>dart pub global activate flutterfire_cli</i>” membuat tool <i>flutterfire</i> dapat diakses dari mana saja di terminal</p>

 <p>Gambar 12. konfigurasi firebase ke flutter</p>	<p>Perintah <code>flutterfire configure --project=prakflutterfirebase-38bb6</code> digunakan untuk mengkonfigurasi firebase ke dalam project flutter.</p>
 <p>Gambar 13.konfigurasi berhasil</p>	<p>Project berhasil dikonfigurasi. Dan berhasil mendapatkan firebase AppId</p>
 <p>Gambar 14. Menginstall dependensi</p>	<p>Menginstall dependensi dengan menggunakan perintah Ketik “flutter pub add firebase_core”</p>
 <p>Gambar 15. Mengubah void main</p>	<p>Mengubah void main</p>
 <p>Gambar 16. Slicing UI</p>	<p>Membuat kode untuk mengimplementasikan desain antarmuka pengguna pada gambar di samping.</p>
 <p>Gambar 17. file dalam folder ui</p>	<p>Membuat folder baru yang diberi nama “ui”, dan di dalam folder tersebut buatlah 4 file dart sesuai pada gambar di samping.</p>

```

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

```

Gambar 18. kode untuk splash.dart

Tambahkan kode di samping pada file splash.dart

```

lib > ui > login.dart > ...
DOKULAH SEMESTER 6/PEMROGRAMAN PERANGKAT
BERGERAKAuthlibbasepraktikumauthlib
3 class LoginScreen extends StatefulWidget {
4   const LoginScreen({key? key}) : super(key: key);
5   @override
6   State<LoginScreen> createState() => _LoginScreenState();
7 }
8
9 class _LoginScreenState extends State<LoginScreen> {
10   final emailEdc = TextEditingController();
11   final passEdc = TextEditingController();
12   bool passInvisible = false;
13   @override
14   Widget build(BuildContext context) {
15     return Scaffold(
16       body: Container(
17         margin: EdgeInsets.symmetric(horizontal: 30, vertical: 70),
18         child: ListView(
19           children: [
20             Text(
21               "Login",
22               style: TextStyle(
23                 fontSize: 40,
24                 fontWeight: FontWeight.bold,
25                 color: Color(0xFF3D4DE0), // TextStyle
26               ), // Text
27             ),
28             SizedBox(
29               height: 15,
30             ), // SizedBox
31             Text(
32               "Silahkan masukan e-mail dan password anda",
33               style: TextStyle(
34                 fontSize: 16,
35               ), // TextStyle
36             ), // Text
37             SizedBox(
38               height: 25,
39             ), // SizedBox
40             Text(
41               "e-mail",
42               style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
43             ), // Text
44             TextFormField(
45               controller: emailEdc,
46             ), // TextFormField
47             SizedBox(
48               height: 10,
49             ), // SizedBox
50             Text(
51               "password",

```

Tambahkan kode di samping pada bagian file login.dart

Gambar 19. kode untuk login.dart

Tambahkan kode di samping pada bagian file register.dart

```

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

```

Gambar 20. kode untuk register.dart

```

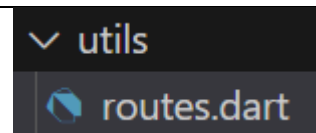
import 'package:flutter/material.dart';

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(0xff3D4DE0), // TextStyle
          ), // Text // Center
        ), // Scaffold
      ),
    );
  }
}

```

Gambar 21. kode untuk home_screen.dart

Tambahkan kode di samping pada bagian home_screen.dart



Gambar 22. file routes pada folder utils

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”.

```

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';

```

Gambar 23. kode untuk routes.dart

Tambahkan kode di samping pada bagian file routes.dart

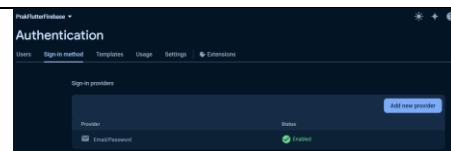
```

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
  }
}

```

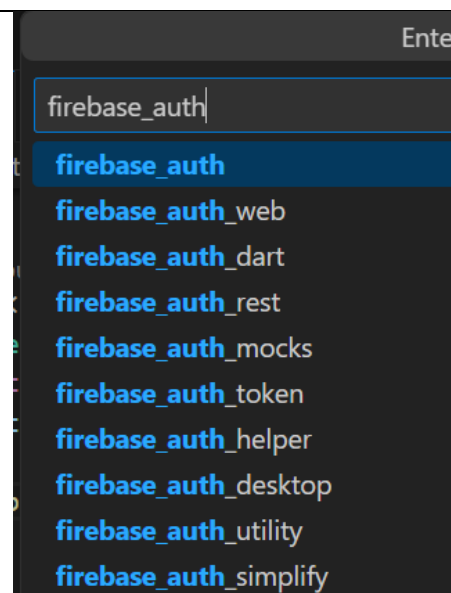
Gambar 24. kode untuk main.dart

Tambahkan kode di samping pada bagian file main.dart



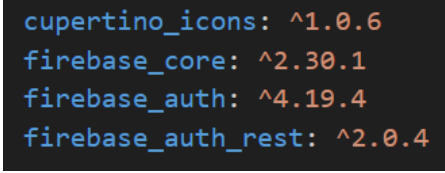
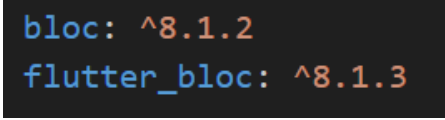
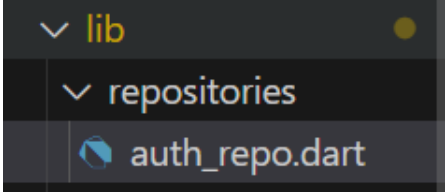
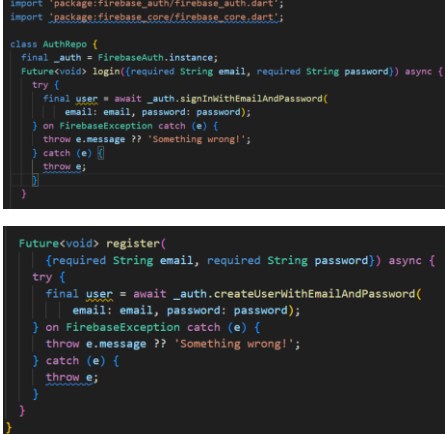
Gambar 25. menambahkan autentikasi email di firebase

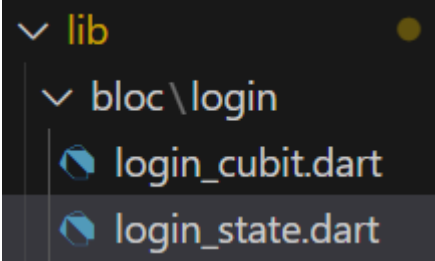
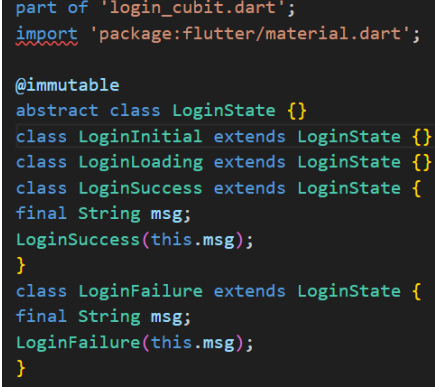
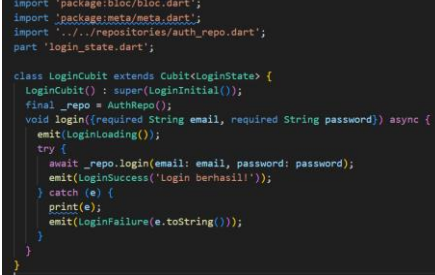
Menambahkan autentikasi email dan password pada dashboard firebase di project kita.

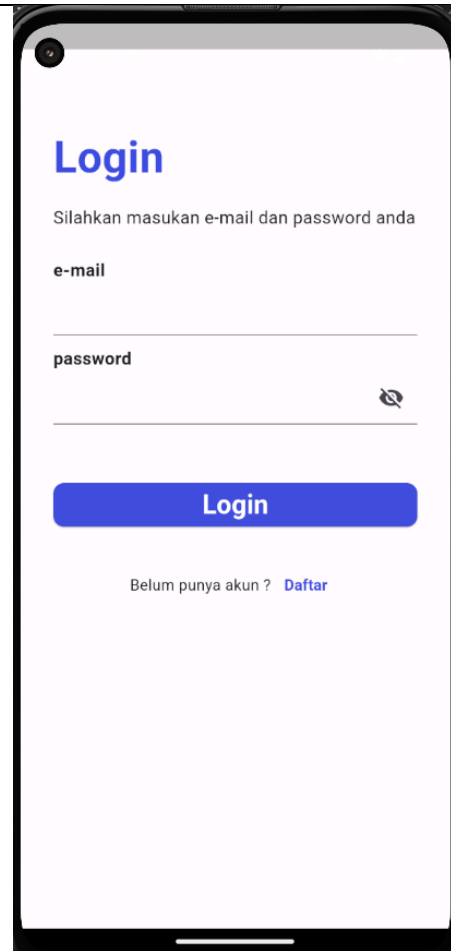


Menginstal dependensi firebase_auth

<i>Gambar 26. dependensi firebase_auth</i>	
--	--

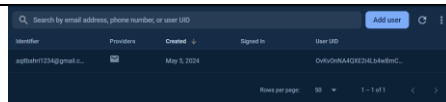
 <pre> cupertino_icons: ^1.0.6 firebase_core: ^2.30.1 firebase_auth: ^4.19.4 firebase_auth_rest: ^2.0.4 </pre> <p><i>Gambar 27. cek firebase_auth</i></p>	<p>Cek ketersediaan dependensi firebase_auth pada pubspec.yaml</p>
 <pre> bloc: ^8.1.2 flutter_bloc: ^8.1.3 </pre> <p><i>Gambar 28. menambahkan dependensi lain</i></p>	<p>Untuk memproses logika bisnis pada aplikasi kita akan menggunakan flutter bloc,</p> <p>sehingga perlu menambahkan 2 dependensi yaitu :</p> <p>bloc: ^8.1.2</p> <p>flutter_bloc: ^8.1.3</p> <p>jangan lupa untuk menjalankan perintah “flutter pub upgrade”</p>
 <p><i>Gambar 29. menambahkan file baru pada folder baru repositories</i></p>	<p>Buat folder “repositories” pada folder “lib”, kemudian buat file “auth_repo.dart”.</p> <p>folder repositories ini akan kita gunakan untuk berkomunikasi dengan firebase/API dari back end.</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><i>Gambar 30. kode untuk file auth_repo.dart</i></p>	<p>Kita akan membuat 2 buah Future<void> dalam class AuthRepo untuk login dan register pada file auth_repo.dart.</p>

 <p><i>Gambar 31. struktur untuk mengatur bloc</i></p>	<p>Kemudian kita akan mengatur bloc untuk login. Buat folder dengan struktur seperti gambar di samping.</p>
 <p><i>Gambar 32. kode untuk file login_state.dart</i></p>	<p>Tambahkan kode di samping pada file login_state.dart</p>
 <p><i>Gambar 33. kode untuk file login_cubit.dart</i></p>	<p>Tambahkan kode di samping pada file login_cubit.dart</p>



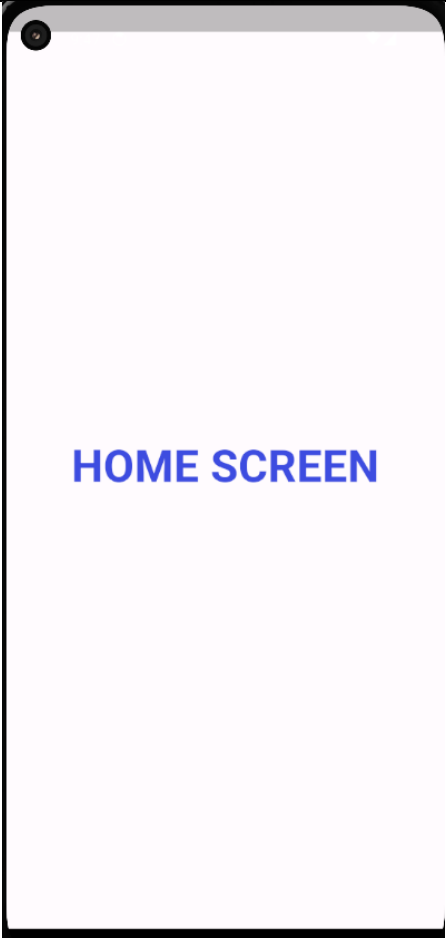
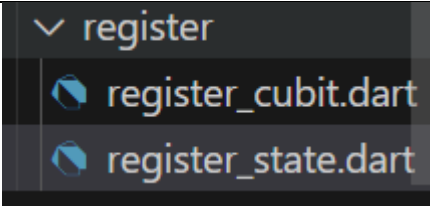
Gambar 34. Run program

Jalankan program. inputan email dan password akan disimpan pada emailEdc dan passEdc kemudian dengan menggunakan bloc inputan tersebut dikirim ke firebase melalui repositori yang sudah kita buat, apabila berhasil maka akan masuk ke home screen, apabila gagal akan muncul pesan gagal.



Gambar 35. menambahkan user

Mengatur untuk bagian register di firebase

 <p><i>Gambar 36. berhasil login</i></p>	<p>Data yang baru ditambahkan, kita cek untuk mengisi pada bagian login.</p>
 <p><i>Gambar 37. folder register</i></p>	<p>Login sudah berhasil kita buat, maka selanjutnya kita akan membuat bagian register.</p> <p>Karena kita sudah membuat Future<void> register. Selanjutnya kita atur untuk blocnya. Buat 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><i>Gambar 38. kode file register_state.dart</i></p>	<p>Tambahkan kode di samping pada bagian file register_state.dart</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><i>Gambar 39. kode file register_cubit.dart</i></p>	<p>Tambahkan kode di samping pada bagian file register_cubit.dart</p>
	<p>Menggunakan bloc pada register.dart, sehingga kode program menjadi seperti berikut</p>

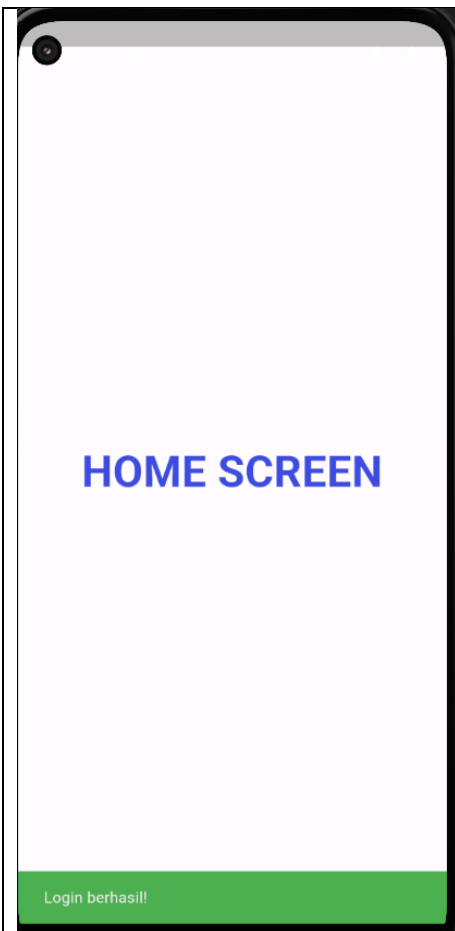
Gambar 40. test register

Mencoba register

Identifier	Providers	Created	Signed In	User UID
anjay@gmail.com		May 5, 2024	May 5, 2024	4f1f8a5170f19a0b9b2e0a8...
anjaytest1234@gmail.c...		May 5, 2024		0a8f0c8a5d3e2d8d3a8b8c...

Gambar 41. data akun yang masuk

User yang baru didaftarkan berhasil ditambahkan



Gambar 42. bukti login berhasil

Mencoba login dengan akun yang baru saja di tambahkan

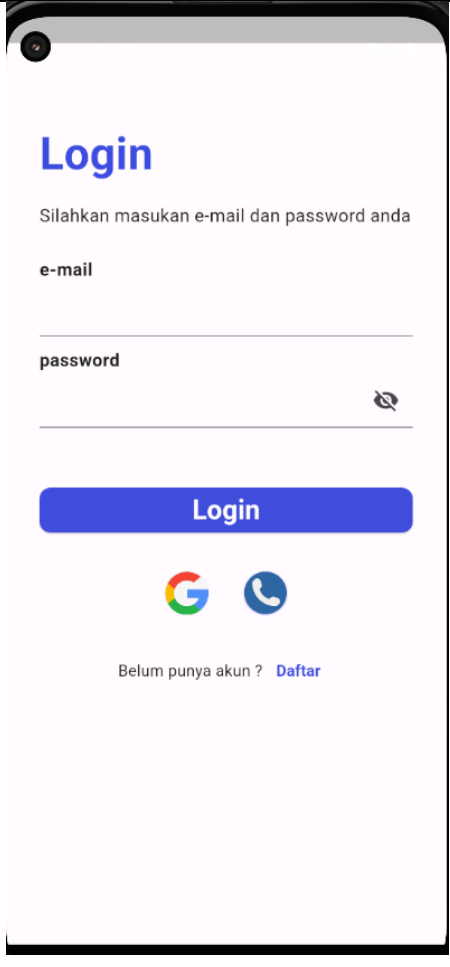
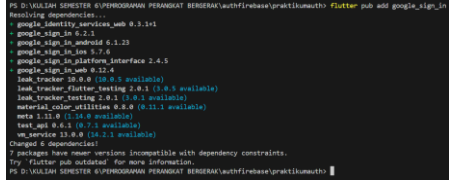
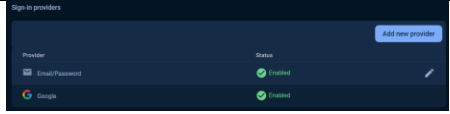
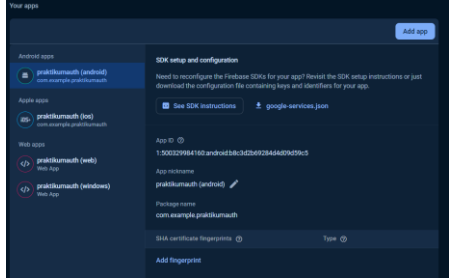
```

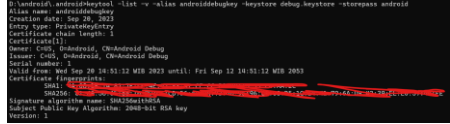
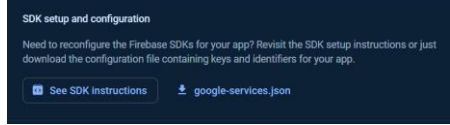
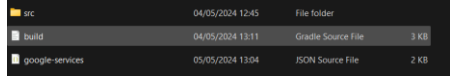
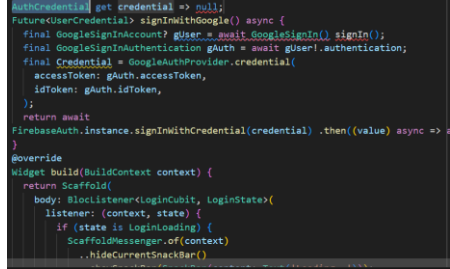
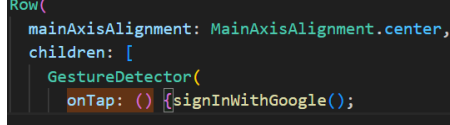
children: [
  GestureDetector(
    onTap: () {},
    child: const CircleAvatar(
      radius: 20.0,
      backgroundImage: NetworkImage(
        'https://img1.pngdownloader.id/20190228/obj/kissimg-google-logo-google-account-g-suite-google'
      ), // CircleAvatar
    ), // GestureDetector
  ), // GestureDetector
  const SizedBox(
    width: 30.0,
  ), // SizedBox
  GestureDetector(
    onTap: () {},
    child: const CircleAvatar(
      radius: 20.0,
      backgroundImage: NetworkImage(
        'https://emerging.com/thum/business/33615-blue-icon-symbol-telephone-computer-logo.png'
      ), // CircleAvatar
    ), // GestureDetector
  ), // GestureDetector
]

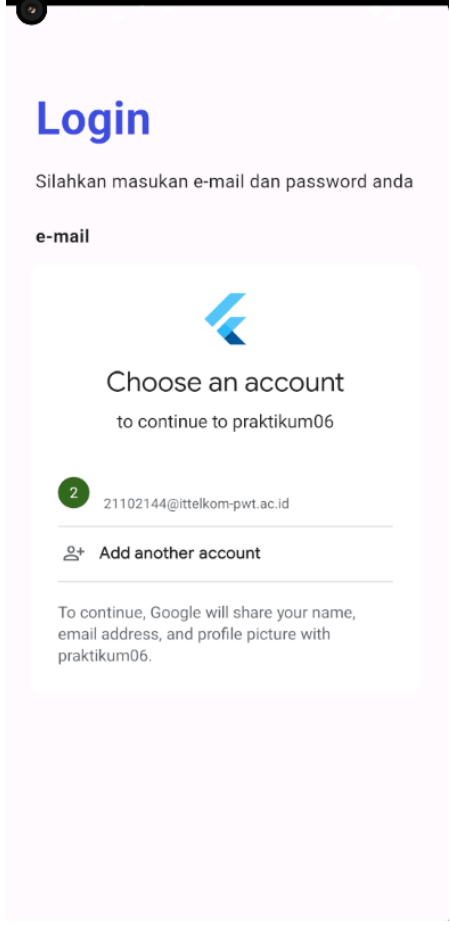
```

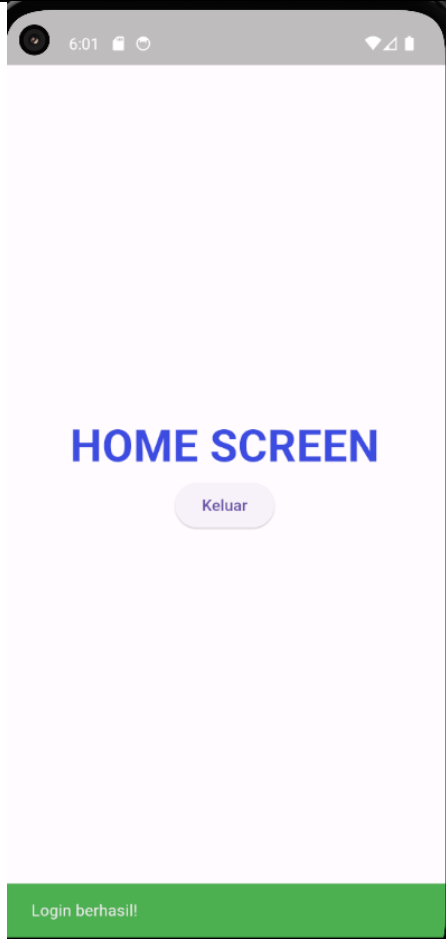

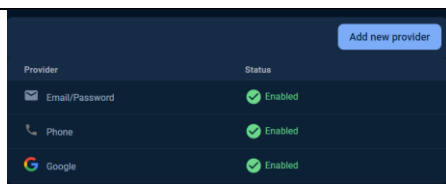

Gambar 43. kode tambahan untuk login.dart

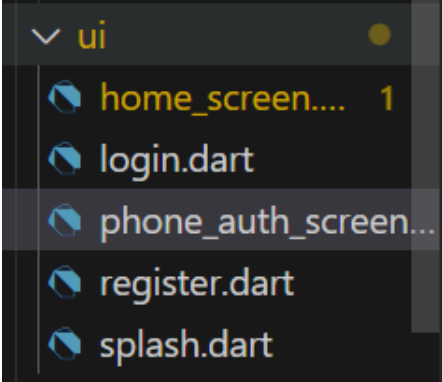
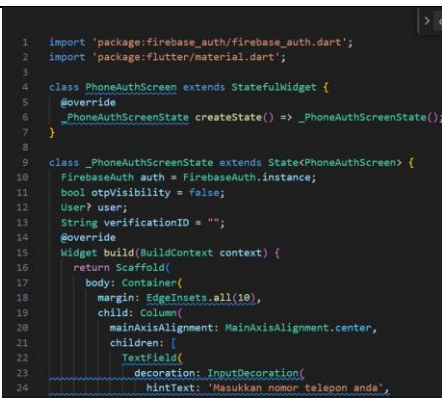
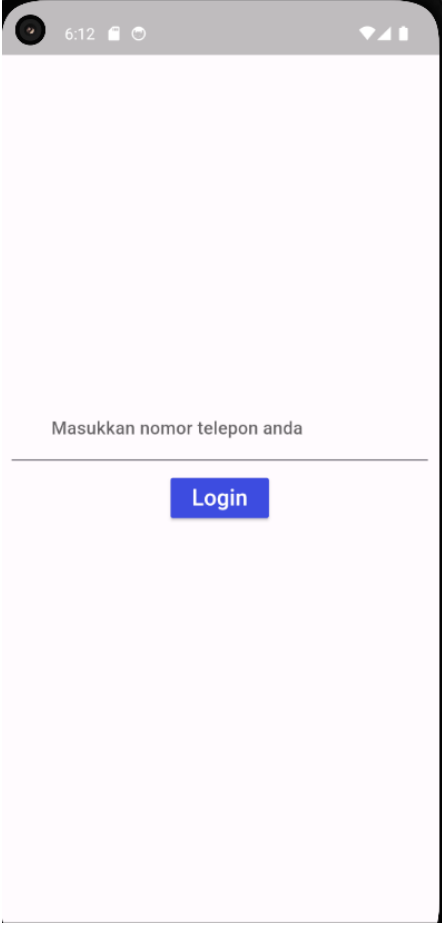
Menambahkan autentikasi baru menggunakan Google dan Phone, dengan menambahkan beberapa fungsi. Yang pertama tambahkan icon google dan Phone pada kode login.dart

	<p>Icon berhasil ditambahkan</p>
 <p><i>Gambar 44. menambahkan depedensi</i></p>	<p>Tambahkan dependency google_sign_in dengan menjalankan command flutter pub add google_sign_in pada terminal.</p>
 <p><i>Gambar 45. menambah metode masuk</i></p>	<p>Untuk mengaktifkan otentikasi menggunakan akun Google perlu menambahkan sign_in method Google pada web firebase</p>
 <p><i>Gambar 46. menambahkan fingerprint pada project</i></p>	<p>Untuk menggunakan otentikasi menggunakan akun Google pada Android perlu menambahkan fingerprint pada Project Settings web firebase. Dengan cara pergi ke project settings dan pilih fingerprints</p>

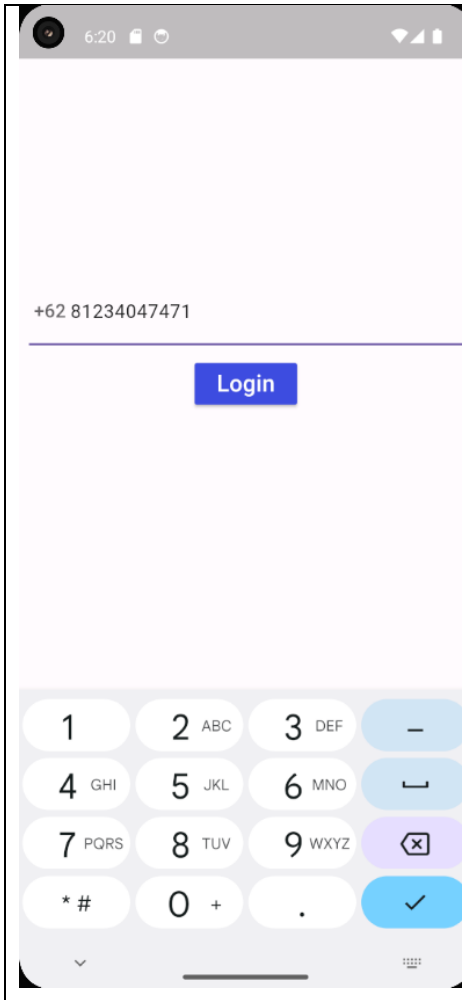
 <p><i>Gambar 47. informasi SHA</i></p>	<p>Untuk mendapatkan SHA1 dan SHA256 menggunakan command berikut pada Terminal: <code>keytool -list -v \</code> <code>-alias androiddebugkey -</code> <code>keystore %USERPROFILE%\android\debug.keystore</code> kemudian tambahkan kode SHA1 ke firebase</p>
 <p><i>Gambar 48. google service</i></p>	<p>Unduh Google service.json</p>
 <p><i>Gambar 49. file Google Service.json yang baru</i></p>	<p>Ganti file google service yang lama dengan yang baru pada direktori app</p>
 <p><i>Gambar 50. kode tambahan pada login.dart</i></p>	<p>Masih pada login.dart tambahkan fungsi berikut untuk sign in menggunakan akun Google</p>
 <p><i>Gambar 51. kode tambahan pada login.dart</i></p>	<p>Tambahkan fungsi signInWithGoogle pada logo Google</p>

 <p><i>Gambar 52. signin Google</i></p>	<p>Disini login menggunakan Google sudah bisa diakses,</p>
<pre>Widget build(BuildContext context) { return Scaffold(body: Center(child: Column(mainAxisAlignment: MainAxisAlignment.center, children: [Text("HOME_SCREEN", style: TextStyle(fontSize: 48, fontWeight: FontWeight.bold, color: Color(0xff3D4DE8),), // TextStyle), // Text ElevatedButton(onPressed: () { // Tambahkan kode untuk tombol keluar di sini }, child: const Text('Keluar'),), // ElevatedButton],), // Column), // Center); // Scaffold }</pre>	<p>Menambahkan tombol keluar pada home_screen</p>

	<p>Tampilan home_screen terbaru</p>
 <pre> ElevatedButton(onPressed: () async { GoogleSignIn().signOut(); FirebaseAuth.instance .signOut() .then((value) => Navigator.pushAndRemoveUntil(context, MaterialPageRoute(builder: (context) => const LoginScreen(),), // MaterialPageRoute (route) => false)); }, child: const Text('Keluar'),), // ElevatedButton </pre>	<p>Menambahkan fungsi keluar pada tombol keluar</p>
	<p>Menambahkan metode login baru yaitu dengan menggunakan phone</p>
 <pre> GestureDetector(onTap: () { Navigator.push(context, MaterialPageRoute(builder: (context) => PhoneAuthScreen(), // MaterialPageRoute),); },) </pre>	<p>Tambahkan fungsi route ke phone_auth_screen</p>

	<p>Pada folder ui, tambahkan file phone_auth_screen.dart</p>
	<p>Tambahkan kode di samping pada file phone_auth_screen.dart</p>
	<p>Tampilan login menggunakan phone</p>

<pre>class _PhoneAuthScreenState extends State<PhoneAuthScreen> { FirebaseAuth auth = FirebaseAuth.instance; bool otpVisibility = false; User? user; String verificationID = ""; TextEditingController _phoneController = TextEditingController(); TextEditingController _otpController = TextEditingController(); @override void dispose() { _phoneController.dispose(); _otpController.dispose(); super.dispose(); } }</pre>	<p>Menambahkan TextEditingController dan Dispose controller tersebut agar tidak terjadi memory leak</p>
<pre>void loginWithPhone() async { auth.verifyPhoneNumber(phoneNumber: "+62" + _phoneController.text, timeout: const Duration(seconds: 60), verificationCompleted: (PhoneAuthCredential credential) async { await auth.signInWithCredential(credential).then((value) { print("You are logged in successfully"); }); },); }</pre>	<p>Pasang phone controller pada text field</p>
<pre>void loginWithPhone() async { auth.verifyPhoneNumber(phoneNumber: "+62" + _phoneController.text, timeout: const Duration(seconds: 60), verificationCompleted: (PhoneAuthCredential credential) async { await auth.signInWithCredential(credential).then((value) { print("You are logged in successfully"); }); }, verificationFailed: (FirebaseAuthException? e) { print(e?.message); }, codeSent: (String verificationId, int? resendToken) { otpVisibility = true; verificationID = verificationId; setState(() {}); }, codeAutoRetrievalTimeout: (String verificationId) {},); }</pre>	<p>Buat fungsi untuk otentikasi menggunakan nomor handphone.</p>
<pre>void verifyOTP() async { PhoneAuthCredential credential = PhoneAuthProvider.credential(verificationId: verificationID, smsCode: _otpController.text); await auth.signInWithCredential(credential).then((value) { setState(() { user = FirebaseAuth.instance.currentUser; }); }).whenComplete(() { if (user != null) { ScaffoldMessenger.of(context) ..hideCurrentSnackBar() ..showSnackBar(const SnackBar(content: Text('Berhasil Login'))); Navigator.pushReplacement(context, MaterialPageRoute(builder: (context) => HomeScreen(),), // MaterialPageRoute); } }); }</pre>	<p>Buat fungsi OTP</p>



Memasukan nomor telepon