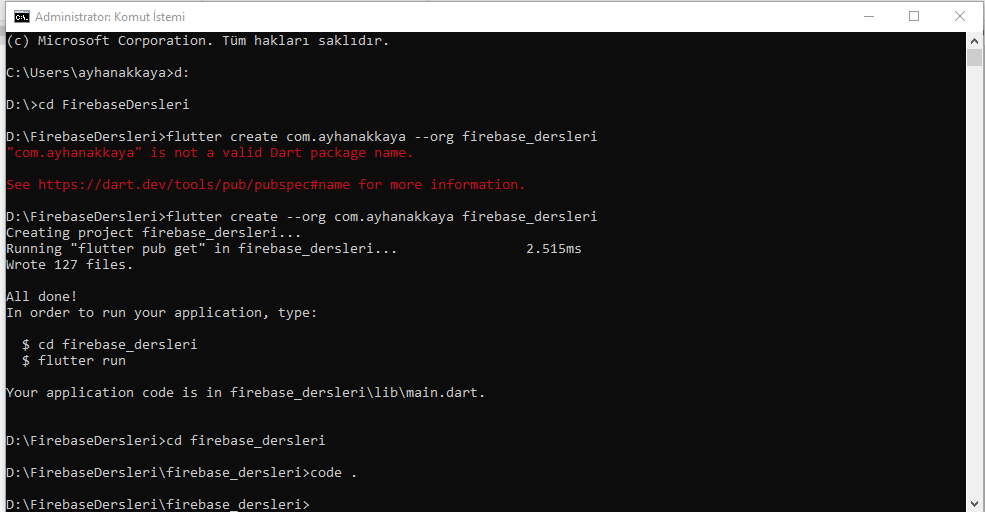
## **FIREBASE İŞLEMLERİ**

**Ders 1 : Firebase nedir? Kurulum nasıl yapılır.**

* <https://firebase.google.com/> adresine gidilir. Gmail ile giriş yapılır. Açılan sayfadan Konsola git kısmına tıklanır.
* Konsola gidildikten sonra Proje ekleme penceresinde proje oluşturulur.
* Firebase de proje oluşturmadan paket adıyla komut penceresinden proje oluşturma işlemini görelim.

**Ders 2 : Paket adıyla proje oluşturmak**

* Cmd ekranı açılır
* D:\FirebaseDersleri>flutter create --org com.ayhanakkaya firebase\_dersleri komutu yazılır
* Cd firebase\_dersleri denilerek proje içine girilir.
* Code . ile visaul studio code ekranında proje açılır.
* Android tarafında AndroidManifest.xml de paket adı görünür.



**Ders 3 : FlutterFire ve Firebase CLI Kurulumları**

* <https://firebase.flutter.dev/docs/overview/> adresine gidilir.
* Installation kısmında bulunan flutter pub add firebase\_core komutu projenin terminal ekranına yapıştırılır.
* Daha sonra Using the FlutterFire CLI kısmında bulunan aşağıdaki kod kopyalanır.

dart pub global activate flutterfire\_cli

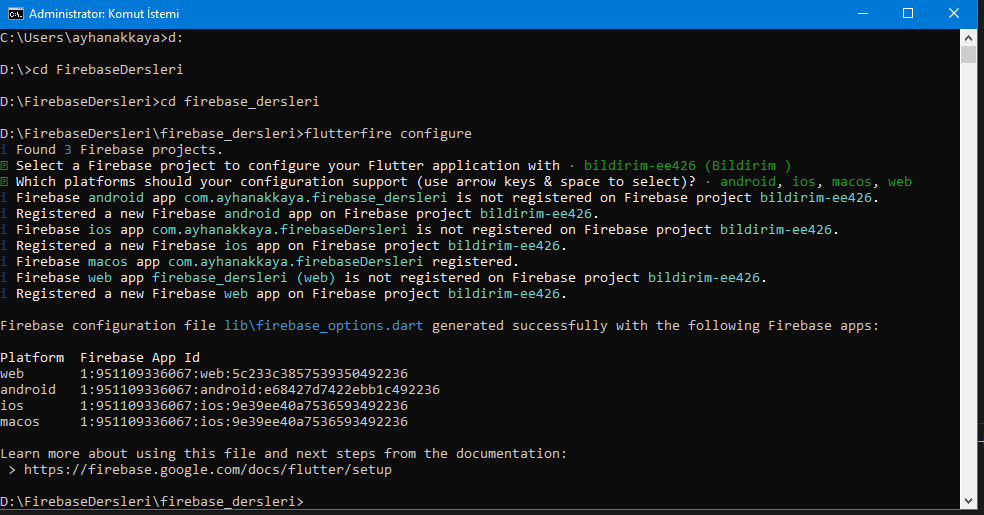
* Kopyalanan bağlantıyı komut penceresi arayüzünden pc ye kuralım.

C:\Users\ayhanakkaya>dart pub global activate flutterfire\_cli

* Kopyalama işlemi sırasında aşağıdaki hata mesajı çıkacaktır.

Warning: Pub installs executables into C:\Users\ayhanakkaya\AppData\Local\Pub\Cache\bin, which is not on your path.You can fix that by adding that directory to your system's "Path" environment variable.A web search for "configure windows path" will show you how.

* Hata mesajında belirtilen sarı alanı path e eklemek gerekiyor.
* FirebaseCLI sitesine gidilir. npm kısmında bir bağlantıyı kurmak gerekiyor. npm kurmak için node.js kurulu olması gerekmektedir. Sitesine gidilip node.js kurulur.
* npm install -g firebase-tools kodu cmd ye yazılır. Çalıştığını anlamak için cmd ye firebase yazalım. Komutlar vs. görünüyor ise sorunsuz yüklenmiştir. firebase login yazarsak kendi konsol ekranımıza ulaşırız.
* <https://console.firebase.google.com/> adresinden Add Project denilerek firebase\_dersleri adında bir proje oluşturalım.
* <https://firebase.flutter.dev/docs/cli> adresinden devam edelim. VS Code üzerinde açılmış olan projemizde terminal açılır ve flutterfire configure yazılır.



* Artık firebase projemize entegre edilmiş oldu.lib klasörü altında firebase ile ligli ayarların olduğu dosyalar eklenmiş oldu.sonraki projelerde artık projeye sadece flutterfire configure işlemini yapmak yeterli olacaktır.
* <https://firebase.flutter.dev/docs/cli> adresinde main.dart üzerinde aşağıdaki kodları ekleyelim.

*import* 'firebase\_options.dart';

await Firebase.initializeApp(

options: DefaultFirebaseOptions.currentPlatform,

);

Main.dart son hali

import 'package:firebase\_core/firebase\_core.dart';

import 'package:flutter/material.dart';

// Import the generated file

import 'firebase\_options.dart';

void main() async {

  WidgetsFlutterBinding.ensureInitialized();

  await Firebase.initializeApp(

    options: DefaultFirebaseOptions.currentPlatform,

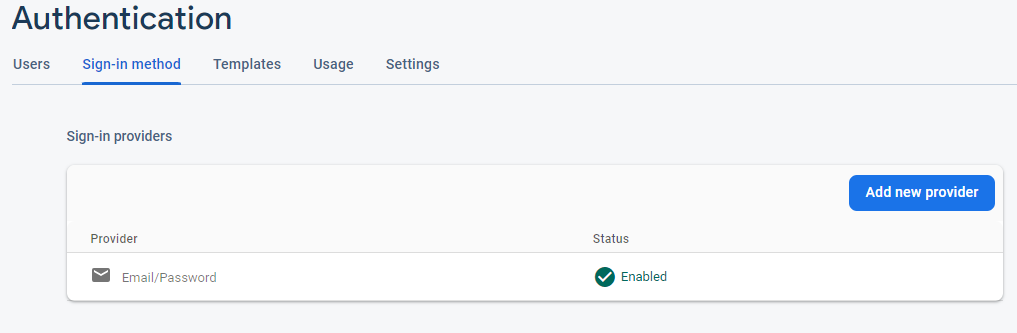
  );

  runApp(const MyApp());

}

**Ders 4 : Email ve Şifre ile Kullanıcı Oluşturma**

* [**https://console.firebase.google.com/project/fir-dersleri-759c2/overview**](https://console.firebase.google.com/project/fir-dersleri-759c2/overview)projemizin olduğu firebaseadresine gidilir.
* Build🡪Autentication🡪Get Started🡪Sign-in Methods🡪Email/Passwords seçilip enable diyerek save yapılır.



* <https://firebase.flutter.dev/docs/cli> adresine gidilip Authentication kısmına geçilir.
* Projemizin terminal kısmına flutter pub add firebase\_auth kodu yazılır.
* Main.dart içerisinde ;

FirebaseAuth auth = FirebaseAuth.instance;

* Kodu aşağıdaki kısma yazılıp import edilir.

class \_MyHomePageState extends State<MyHomePage> {

  FirebaseAuth auth = FirebaseAuth.instance;

* Kod aşağıdaki gibi düzenlenebilir.

class \_MyHomePageState extends State<MyHomePage> {

  late FirebaseAuth auth;

  @override

  void initState() {

    super.initState();

    auth = FirebaseAuth.instance;

  }

* Aşağıdaki alanda bir kullanıcı adı ve şifre oluşturalım.

class \_MyHomePageState extends State<MyHomePage> {

  late FirebaseAuth auth;

  final String \_email = 'ayhanakkayameb@gmail.com';

  final String \_password = 'Akkaya.6344';

  @override

  void initState() {

    super.initState();

    auth = FirebaseAuth.instance;

  }

* Body kısmına aşağıdaki kodları düzenleyelim ve metodun adını ekleyelim.

Center(

          child: Column(

            mainAxisAlignment: MainAxisAlignment.center,

            children: [

              ElevatedButton(

                style: ElevatedButton.styleFrom(primary: Colors.red),

                child: const Text('Email/Şifre Giriş'),

                onPressed: () {

                  createUserEmailAndPassword();

                },

              ),

* createUserEmailAndPassword(); isimli metot içerisinde işlemlerimizi yapalım.

void createUserEmailAndPassword() async {

    try {

      var \_userCredential = await auth.createUserWithEmailAndPassword(

          email: \_email, password: \_password);

      debugPrint(\_userCredential.toString());

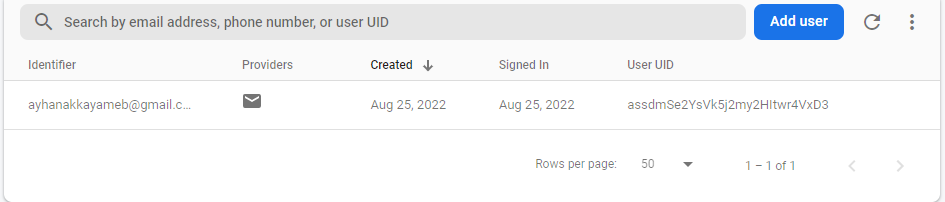
    } catch (e) {

      debugPrint(e.toString());

    }

  }

* Projeyi çalıştırdığımız zaman butona basalım ve firebase console ekranına gelelim.



* Görüldüğü gibi kullanıcımız o proje için oluşturulmuş oldu.
* Şimdi de var olan kullanıcılar için giriş işlemini oluşturalım. İkinci bir butonu alta ekleyelim.

ElevatedButton(

                style: ElevatedButton.styleFrom(primary: Colors.blue),

                child: const Text('Email/Şifre Giriş'),

                onPressed: () {

                  logincreateUserEmailAndPassword();

                },

              ),

* logincreateUserEmailAndPassword(); metodunu yazalım.

 void logincreateUserEmailAndPassword() async {

    try {

      var \_userCredential = await auth.signInWithEmailAndPassword(

          email: \_email, password: \_password);

      debugPrint(\_userCredential.toString());

    } catch (e) {

      debugPrint(e.toString());

    }

  }

* Görüldüğü gibi var olan kullanıcının sisteme girişi gerçekleştirilmiş oldu.

**Ders 5 : Email ve Şifre Oturum Açma ,Kapama,Kullanıcı Silme ve AuthState**

* AuthState ile projedeki oturum akışlarına anlık olarak erişebiliriz. Şimdi projemiz içersinde initState metoduna aşağıdaki kodları console ekranından ekleyelim.

auth.authStateChanges().listen((User? user) {

      if (user == null) {

        debugPrint('User oturumu kapalı');

      } else {

        debugPrint(

            'User oturum açık ${user.email} ve email durumu ${user.emailVerified}');

      }

    });

* Çıkış işlemleri için projeye bir ElevatedButton ekleyelim.

ElevatedButton(

              onPressed: () {

                signOutUser();

              },

              style: ElevatedButton.styleFrom(primary: Colors.yellow),

              child: const Text('Oturumu Kapat'),

            ),

* signOutUser(); metodunu yazalım.

void signOutUser() async {

    await auth.signOut();

  }

* User oluşturulur iken email i onaylanmamış ise onaylamak için aşağıdaki metoda \_myUser kısmı eklenir.

void createUserEmailAndPassword() async {

    try {

      var \_userCredential = await auth.createUserWithEmailAndPassword(

          email: \_email, password: \_password);

      var \_myUser = \_userCredential.user;

      if (!\_myUser!.emailVerified) {

        await \_myUser.sendEmailVerification();

      } else {

        debugPrint('kullanıcın maili onaylanmış, ilgili sayfaya gidebilir.');

      }

      debugPrint(\_userCredential.toString());

    } catch (e) {

      debugPrint(e.toString());

    }

* Şimdi de kullanıcı silme işlemini yapalım.
* Butonumuzu ekleyelim.

ElevatedButton(

                onPressed: () {

                  deleteUser();

                },

                style: ElevatedButton.styleFrom(primary: Colors.purple),

                child: const Text('Kullanıcıyı sil'),

              ),

* deleteUser(); metodunu ekleyelim.

void deleteUser() async {

    if (auth.currentUser != null) {

      await auth.currentUser!.delete();

    } else {

      debugPrint('Kullanıcı oturum açmadığı için silinemez');

    }

  }

* Console gidince kullanıcının silindiğini görebilirsiniz.

**Ders 6 : Email ve Şifre Güncelleme ve Reauthenticate**

* Bir tane günceleme butonu ekleyelim.

ElevatedButton(

                onPressed: () {

                  changePassword();

                },

                style: ElevatedButton.styleFrom(primary: Colors.brown),

                child: const Text('Parola Değiştir.'),

              ),

* changePassword(); metodunun içeriğini dolduralım.

void changePassword() async {

    try {

      await auth.currentUser!.updatePassword('password');

      await auth.signOut();

    } on FirebaseAuthException catch (e) {

      if (e.code == 'requires-recent-login') {

        debugPrint('reauthenticate olunacak');

        var credential =

            EmailAuthProvider.credential(email: \_email, password: \_password);

        await auth.currentUser!.reauthenticateWithCredential(credential);

        await auth.currentUser!.updatePassword('password');

        await auth.signOut();

        debugPrint('şifre güncellendi');

      }

    } catch (e) {

      debugPrint(e.toString());

    }

  }

* Email i de güncelleme işlemini yapalım. Öncelikle email güncelle butonunu ekleyelim.

ElevatedButton(

                onPressed: () {

                  changeEmail();

                },

                style: ElevatedButton.styleFrom(primary: Colors.pink),

                child: const Text('Email Değiştir'),

              ),

* changeEmail(); metodunu yazalım.

void changeEmail() async {

    try {

      await auth.currentUser!.updateEmail('ayhanakkaya@gmail.com');

      await auth.signOut();

    } on FirebaseAuthException catch (e) {

      if (e.code == 'requires-recent-login') {

        debugPrint('reauthenticate olunacak');

        var credential =

            EmailAuthProvider.credential(email: \_email, password: \_password);

        await auth.currentUser!.reauthenticateWithCredential(credential);

        await auth.currentUser!.updateEmail('emrealtunbilek@gmail.com');

        await auth.signOut();

        debugPrint('email güncellendi');

      }

    } catch (e) {

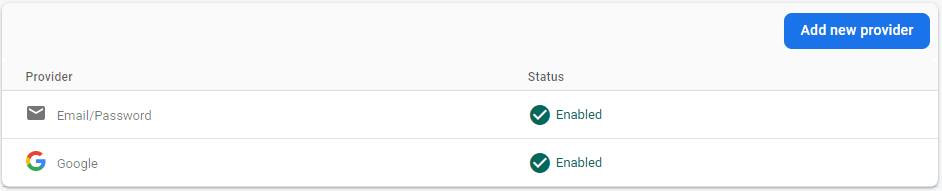
      debugPrint(e.toString());

    }

  }

**Ders 7 : Gmail ile Oturum Açma**

* [**https://console.firebase.google.com/project/fir-dersleri-759c2/overview**](https://console.firebase.google.com/project/fir-dersleri-759c2/overview)projemizin olduğu firebaseadresine gidilir.
* Build🡪Autentication🡪Get Started🡪Sign-in Methods🡪Add New Providers 🡪 Google seçilir. Enable dedikten sonra Project support mail kısmına kendi mail adresimizi yazalım ve Save yapalım



Projenin olduğu editörde Terminal i açalım ve alttaki kodları yazalım.

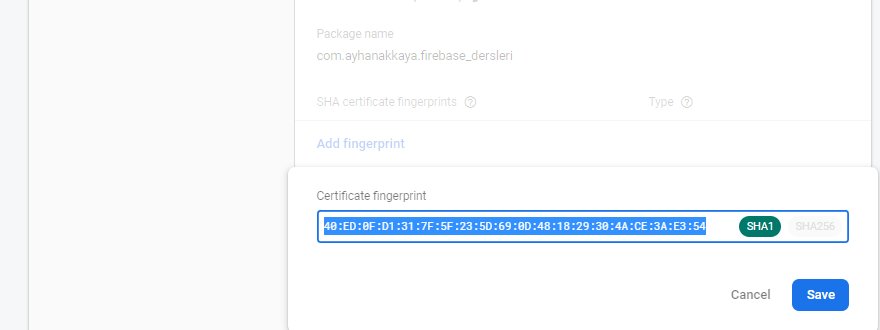


Kodlar yazıldıktan sonra terminalde şu kod kopyalanır.SHA1 ve SHA256 kısmının sayı kısımları kopyalanır.

40:ED:0F:D1:31:7F:5F:23:5D:69:0D:48:18:29:30:4A:CE:3A:E3:54 (SHA1)

C6:0D:56:11:11:71:2A:15:40:FD:CD:16:9E:06:3E:18:A2:A2:05:D3:26:8E:2C:82:1A:29:96:4A:EC:2D:A8:B8 (SHA256)

Kopyalanan bağlantı firebase proje settings içerisinde ilgili yere Add fingerprints diyerek yapıştırılır.





Daha sonra aynı ekranda bulunan Google-services.json dosyası indirilip projede android🡪app altına yapıştırılır.

Daha sonra app🡪build.gradle içine aşağıdaki kod eklenir

apply plugin: 'com.google.gms.google-services'

Daha sonra android🡪build.gradle altına aşağıdaki kod yazılır.

classpath 'com.google.gms:google-services:4.3.8'

Herhangi bir paketi vs code üzerinde indirebilmek için ctrl+shift+P ye basılır.Add Dart Dependency 🡪 google signin ile paketleri projeye ekleyebiliriz.indiğine pubspec.yaml da bakılabilir.

ElevatedButton(

                onPressed: () {

                  googleIleGiris();

                },

                style: ElevatedButton.styleFrom(primary: Colors.green),

                child: const Text('gmail ile giriş '),

              ),

void googleIleGiris() async {

    final GoogleSignInAccount? googleUser = await GoogleSignIn().signIn();

    // Obtain the auth details from the request

    final GoogleSignInAuthentication? googleAuth =

        await googleUser?.authentication;

    // Create a new credential

    final credential = GoogleAuthProvider.credential(

      accessToken: googleAuth?.accessToken,

      idToken: googleAuth?.idToken,

    );

    // Once signed in, return the UserCredential

    await FirebaseAuth.instance.signInWithCredential(credential);

  }

**Ders 8 : Telefon ile Oturum Açma**

**Bu kısım sonra işlenecektir.**