

NAME: HETAL NIKAM
CLASS: D15A
ROLL NO. :6

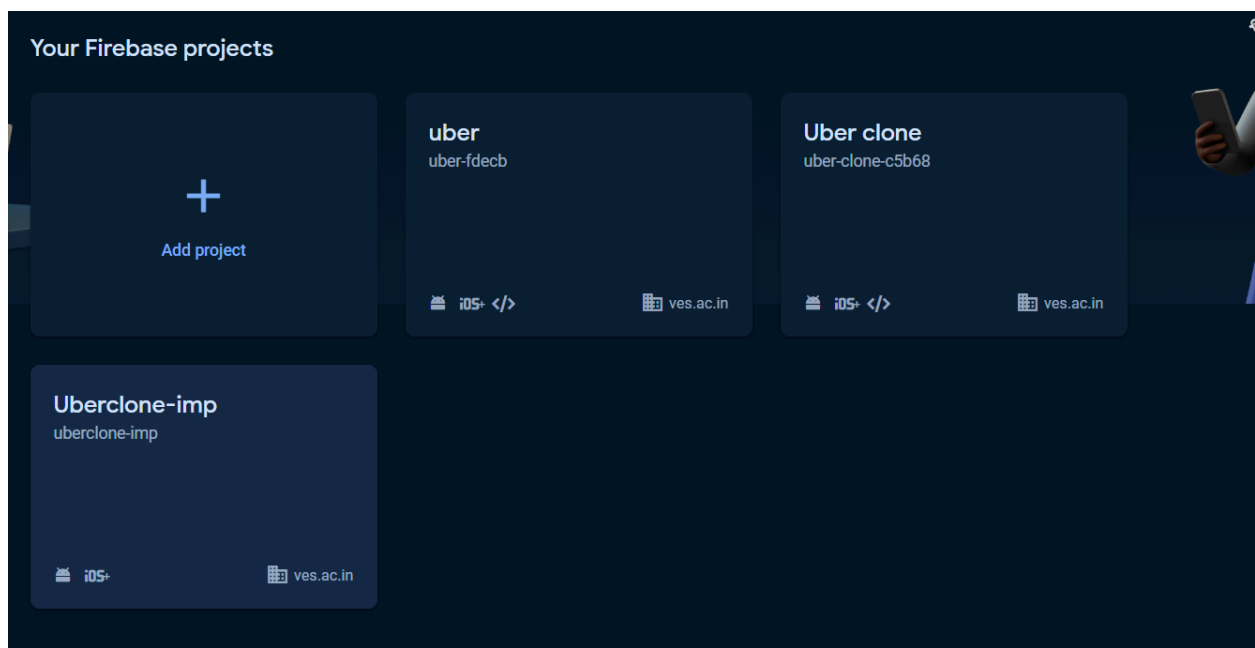
EXP 6

AIM:

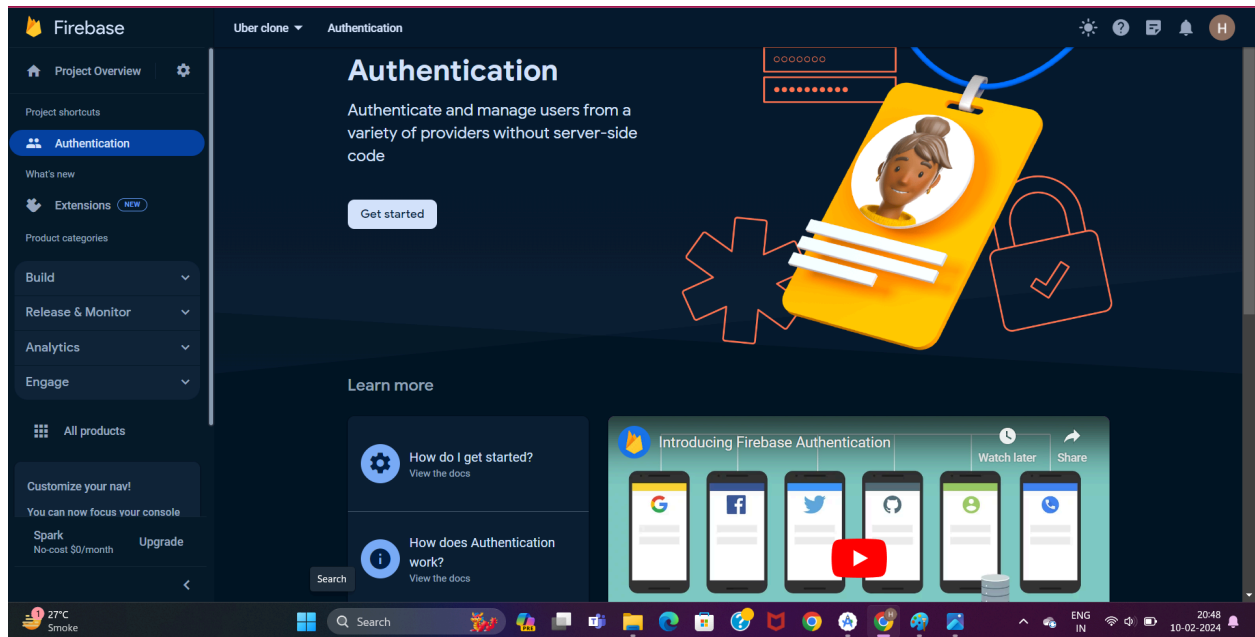
How To Set Up Firebase with Flutter for iOS and Android Apps

THEORY / STEPS:

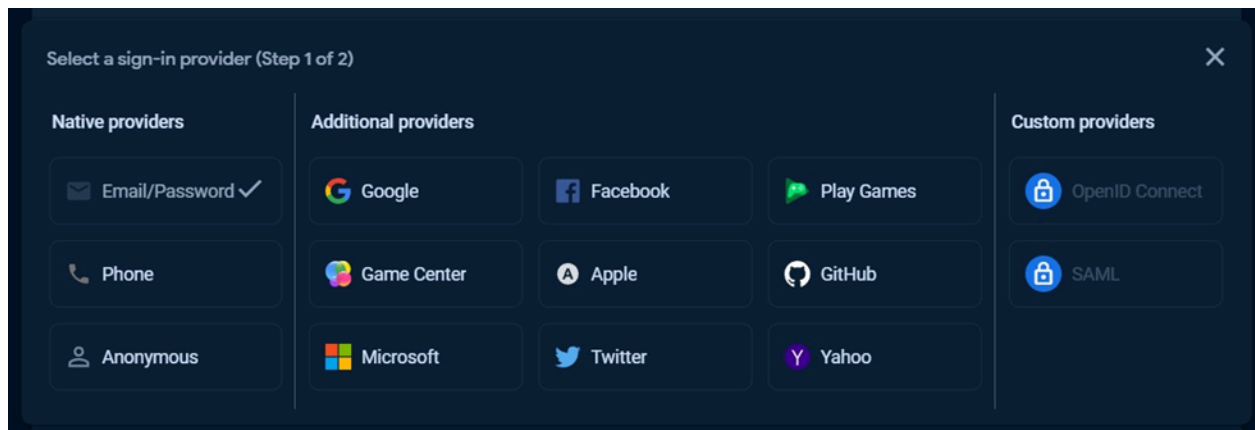
1.Create a project in firebase console.



2. From Build select Authentication to enable it



3. Here select whichever sign in method you would prefer.



4. After enabling all the E-mail ids will be stored after the authentication

10. We check the flutter initialization in main function of our application.

```
future: Firebase.initializeApp(),
```

11. We add dependencies:

```
dependencies:
  flutter:
    sdk: flutter
  fluttertoast: ^8.0.7

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  connectivity_plus:
  firebase_core:
  firebase_auth:
  firebase_database:
  font_awesome_flutter: ^9.2.0
  mapbox_gl: ^0.16.0
  flutter_map: ^6.1.0
  latlong2: ^0.9.0
  geocoding: 2.1.1
  # geocoding: ^4.0.7

  flutter_polyline_points: ^2.0.0
  # mapbox_navigation:
  url_launcher: ^6.0.9
```

These should be compatible with the flutter SDK version we are using.

If we get an error with any of these we can try running:

Flutter pub upgrade

Flutter pub get

CODE:

LOGIN PAGE.DART

```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:uber_practice/Authentication/signup_screen.dart';
import 'package:uber_practice/methods/common_methods.dart';
import 'package:uber_practice/pages/homepage.dart';
import 'package:firebase_core/firebase_core.dart';

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

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  TextEditingController emailTextEditingController = TextEditingController();
  TextEditingController passwordTextEditingController =
    TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;
  bool isLoading = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
            colors: [Colors.blue[900]!, Colors.blue[400]!],
          ),
        ),
        child: Center(
          child: SingleChildScrollView(
            child: Container(
              margin: EdgeInsets.symmetric(horizontal: 20),
              padding: EdgeInsets.all(20),
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  SizedBox(height: 30),
                  TextFormField(
                    controller: emailTextEditingController,
                    keyboardType: TextInputType.emailAddress,
```

```

        decoration: InputDecoration(
          labelText: "Email",
          prefixIcon: Icon(Icons.email, color: Colors.black87),
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
            borderSide: BorderSide.none,
          ),
          filled: true,
          fillColor: Colors.white.withOpacity(0.5),
          hintStyle: TextStyle(color: Colors.black87),
          labelStyle: TextStyle(color: Colors.black87),
        ),
        style: TextStyle(
          fontSize: 18,
          color: Colors.black87,
        ),
      ),
    SizedBox(height: 20),
    TextFormField(
      controller: passwordTextEditingController,
      obscureText: true,
      decoration: InputDecoration(
        labelText: "Password",
        prefixIcon: Icon(Icons.lock, color: Colors.black87),
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(10),
          borderSide: BorderSide.none,
        ),
        filled: true,
        fillColor: Colors.white.withOpacity(0.5),
        hintStyle: TextStyle(color: Colors.black87),
        labelStyle: TextStyle(color: Colors.black87),
      ),
      style: TextStyle(
        fontSize: 18,
        color: Colors.black87,
      ),
    ),
    SizedBox(height: 30),
    SizedBox(
      width: double.infinity,
      child: ElevatedButton(
        onPressed: isLoading ? null : _handleLogin,
        style: ElevatedButton.styleFrom(
          primary: Colors.white,
          padding: EdgeInsets.symmetric(vertical: 15),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10),
          ),
        ),
      ),
    ),
  ),
),

```



```

try {
  if (emailTextEditingController.text.isEmpty ||
      passwordTextEditingController.text.isEmpty) {
    CommonMethods.showToast("Please fill in all fields.");
    setState(() {
      isLoading = false;
    });
    return;
  }

  UserCredential userCredential = await _auth.signInWithEmailAndPassword(
    email: emailTextEditingController.text,
    password: passwordTextEditingController.text,
  );

  if (userCredential != null) {
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => HomePage()),
    );
  } else {
    CommonMethods.showToast("Login failed. Please check your
credentials.");
  }
} catch (e) {
  print("Error during login: $e");
  String errorMessage = "An error occurred during login.";
  if (e is FirebaseAuthException) {
    switch (e.code) {
      case 'user-not-found':
        errorMessage = "No user found with this email.";
        break;
      case 'wrong-password':
        errorMessage = "Incorrect password.";
        break;
      case 'invalid-email':
        errorMessage = "Invalid email format.";
        break;
      default:
        errorMessage = "Login failed. Please try again later.";
    }
  }
  CommonMethods.showToast(errorMessage);
} finally {
  setState(() {
    isLoading = false;
  });
}
}

```



```
}
```

SIGNUP SCREEN.DART

```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:uber_practice/methods/common_methods.dart';
import 'package:uber_practice/pages/homepage.dart';
import '../widgets/loading.dart';
import 'login_screen.dart';

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

  @override
  State<SignUpScreen> createState() => _SignUpScreenState();
}

class _SignUpScreenState extends State<SignUpScreen> {
  TextEditingController userNameTextEditingController =
    TextEditingController();
  TextEditingController userPhoneTextEditingController =
    TextEditingController();
  TextEditingController emailTextEditingController = TextEditingController();
  TextEditingController passwordTextEditingController =
    TextEditingController();

  CommonMethods cMethods = CommonMethods();

  checkIfNetworkIsAvailable() {
    cMethods.checkConnectivity(context);
    signUpFormValidation();
  }

  signUpFormValidation() {
    if (userNameTextEditingController.text.trim().length < 4) {
      CommonMethods.showToast("Your name must be at least 4 characters");
    } else if (userPhoneTextEditingController.text.trim().length < 10) {
      CommonMethods.showToast("Your number must be at least 10 characters");
    } else if (!emailTextEditingController.text.trim().contains("@")) {
      CommonMethods.showToast("Please enter a valid email address");
    } else if (passwordTextEditingController.text.trim().length < 6) {
      CommonMethods.showToast("Your password must be at least 6 characters");
    } else {
      registerNewUser();
    }
  }
}
```

```

registerNewUser() async {
  showDialog(
    context: context,
    barrierDismissible: false,
    builder: (BuildContext context) => LoadingDialog(messageText:
"Registering your account..."),
  );

  try {
    UserCredential userCredential = await
FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: emailTextEditingController.text.trim(),
      password: passwordTextEditingController.text.trim(),
    );

    if (userCredential != null) {
      DatabaseReference usersRef =
FirebaseDatabase.instance.ref().child("users").child(userCredential.user!.uid)
;

      Map userDataMap = {
        "name": userNameTextEditingController.text.trim(),
        "email": emailTextEditingController.text.trim(),
        "phone": userPhoneTextEditingController.text.trim(),
        "id": userCredential.user!.uid,
        "password":passwordTextEditingController.text.trim(),
        "blockStatus": "no",
      };
      await usersRef.set(userDataMap);

      // Navigate to the home page upon successful registration
      Navigator.pushReplacement(context, MaterialPageRoute(builder: (c) =>
HomePage())));
    }
  } catch (e) {
    print("Error during registration: $e");
    CommonMethods.showToast("Registration failed. Please try again later.");
  } finally {
    Navigator.pop(context); // Close loading dialog
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.transparent,
    body: Container(
      decoration: BoxDecoration(
        gradient: LinearGradient(
          colors: [Colors.purple, Colors.blue],

```

```

        begin: Alignment.topCenter,
        end: Alignment.bottomCenter,
        stops: [0.0, 1.0],
        tileMode: TileMode.clamp,
    ),
),
child: Center(
  child: SingleChildScrollView(
    child: Padding(
      padding: const EdgeInsets.all(20),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            "Create an Account",
            style: TextStyle(
              fontSize: 32,
              fontWeight: FontWeight.bold,
              color: Colors.white,
            ),
          ),
          SizedBox(height: 40),
          TextField(
            controller: userNameTextEditingController,
            keyboardType: TextInputType.text,
            style: TextStyle(color: Colors.white), // Set input text
color to white
            decoration: InputDecoration(
              labelText: "User Name",
              labelStyle: TextStyle(color: Colors.white),
            ),
          ),
          SizedBox(height: 20),
          TextField(
            controller: userPhoneTextEditingController,
            keyboardType: TextInputType.number,
            style: TextStyle(color: Colors.white), // Set input text
color to white
            decoration: InputDecoration(
              labelText: "Phone Number",
              labelStyle: TextStyle(color: Colors.white),
            ),
          ),
          SizedBox(height: 20),
          TextField(
            controller: emailTextEditingController,
            keyboardType: TextInputType.emailAddress,

```










```

        style: TextStyle(color: Colors.white), // Set input text
color to white
        decoration: InputDecoration(
          labelText: "Email",
          labelStyle: TextStyle(color: Colors.white),
        ),
      ),
      SizedBox(height: 20),
      TextField(
        controller: passwordTextEditingController,
        obscureText: true,
        keyboardType: TextInputType.text,
        style: TextStyle(color: Colors.white), // Set input text
color to white
        decoration: InputDecoration(
          labelText: "Password",
          labelStyle: TextStyle(color: Colors.white),
        ),
      ),
      SizedBox(height: 40),
      SizedBox(
        width: double.infinity,
        child: ElevatedButton(
          onPressed: () {
            checkIfNetworkIsAvailable();
          },
          style: ElevatedButton.styleFrom(
            primary: Colors.white,
            padding: EdgeInsets.all(15),
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(10),
            ),
          ),
          child: Text(
            "Sign Up",
            style: TextStyle(fontSize: 18, color: Colors.purple),
          ),
        ),
      ),
      SizedBox(height: 20),
      Center(
        child: TextButton(
          onPressed: () {
            Navigator.push(context, MaterialPageRoute(builder: (c)
=> LoginScreen()));
          },
          child: Text(
            "Already have an account? Sign in",
            style: TextStyle(

```


```
        color: Colors.white,    ),  
),  
),  
),  
),  
),  
),  
),  
),  
),  
);  
}  
}
```

OUTPUT:
Authentication:

<div> <div>  Search by email address, phone number, or user UID </div> <div> Add user   </div> </div>				
Identifier	Providers	Created 	Signed In	User UID
d@gmail.com		Feb 21, 2024	Feb 21, 2024	tJkxqyFhcwZ27LcsfDxQG0jN...
v@gmail.com		Feb 21, 2024	Feb 21, 2024	gHvzCKINdXMbvPbXFWy0xu4...
hetal@gmail.com		Feb 20, 2024	Feb 21, 2024	n6y8K3E8fBcWMmui0b5qqM...
s@gmail.com		Feb 19, 2024	Feb 21, 2024	rf2dB0zT0EhRswvEVXKv7Cst...
a@gmail.com		Feb 18, 2024	Feb 18, 2024	CqbXIIAoilhKUiIubsL2lxWqccD3

The users and the pickup and destination points are stored in the data

The users are stored with their uid


 <https://uber-fdecb-default-rtdb.firebaseio.com>

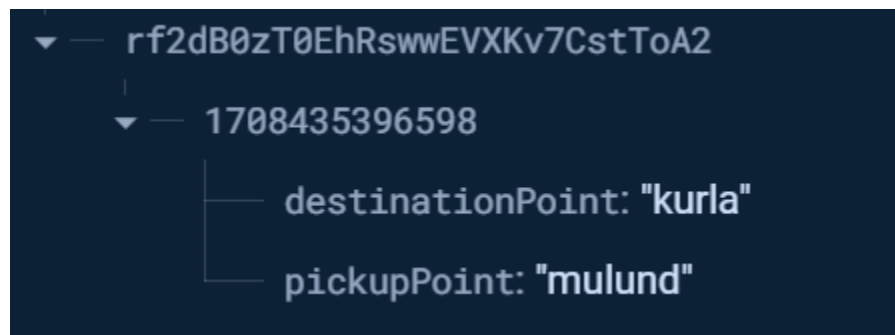
<https://uber-fdecb-default-rtdb.firebaseio.com/>

- ▶ user_locations
- ▶ users

The users with their id

▼ rf2dB0zT0EhRswwEVXKv7CstToA2

- blockStatus: "no"
- email: "s@gmail.com"
- id: "rf2dB0zT0EhRswwEVXKv7CstToA2"
- name: "sanika" 
- password: "1234567"
- phone: "1234567890"



CONCLUSION:

Connected firebase with twitter clone project and handled related errors.