MAD & PWA Lab Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	14
Name	Komal Milind Deolekar
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	

Project Title: MagicPin **Roll No.: 14**

AIM : To Connect Flutter UI with fireBase database

Theory :

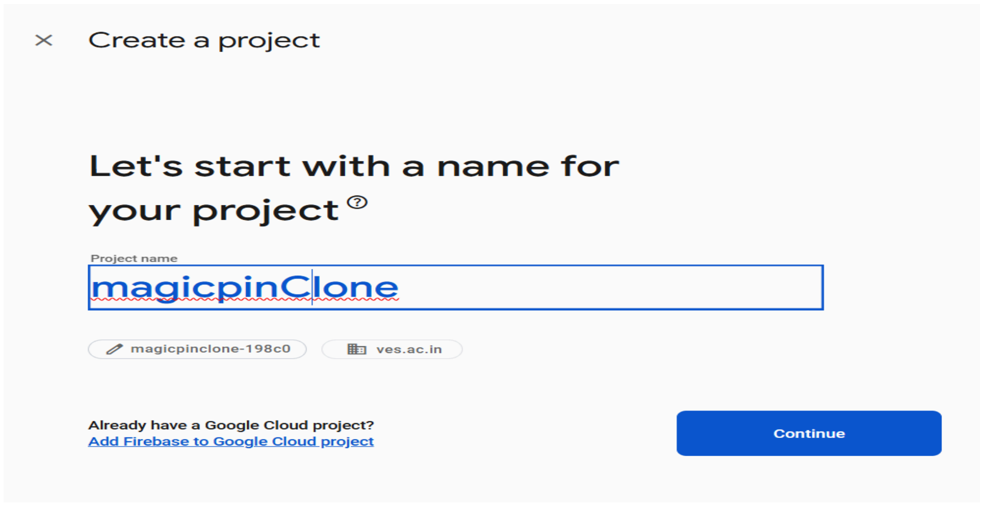
Firebase provides a powerful backend solution for Flutter applications, enabling features like authentication, real-time database, and cloud storage. By integrating Firebase, developers can easily manage user authentication and store data without setting up a separate backend. Firebase is a cloud-based platform by Google that provides backend services for mobile and web applications, eliminating the need to manage servers. It integrates seamlessly with Flutter to enhance app functionality, security, and performance.

It includes **Authentication** (email, social logins, phone sign-in), **Cloud Firestore & Realtime Database** for real-time data syncing, and **Cloud Storage** for handling images, videos, and documents. **Cloud Messaging (FCM)** enables push notifications, while **Crashlytics** and **Performance Monitoring** help in debugging and optimizing app performance. **Remote Config** allows updating app features without requiring updates, and **Analytics** provides insights into user behavior. With Firebase, Flutter apps can be more scalable, secure, and feature-rich without complex backend management.

Steps to Set Up Firebase with Android Apps

Step 1: Create a Firebase Project

1. Go to Firebase Console.
2. Click on **"Add Project"**, enter your project name, and proceed.



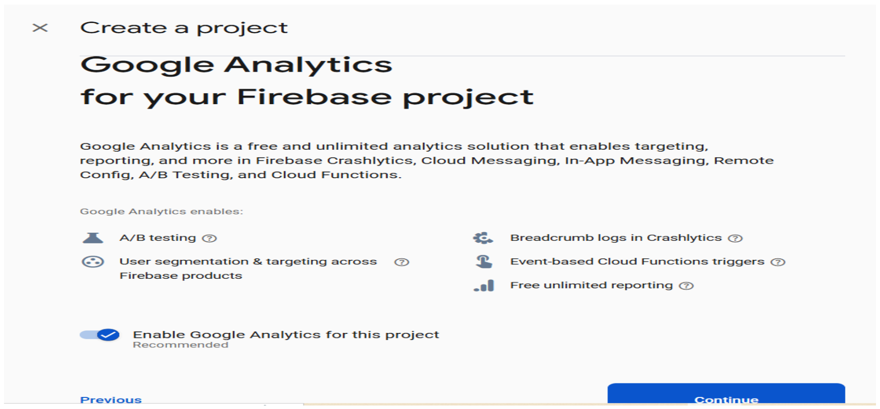
Let's start with a name for your project

Project name: magicpinClone

magicpinclone-198c0
ves.ac.in

Already have a Google Cloud project?
Add Firebase to Google Cloud project

3. Enable **Google Analytics** (optional) and complete the setup.



Create a project

Google Analytics for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

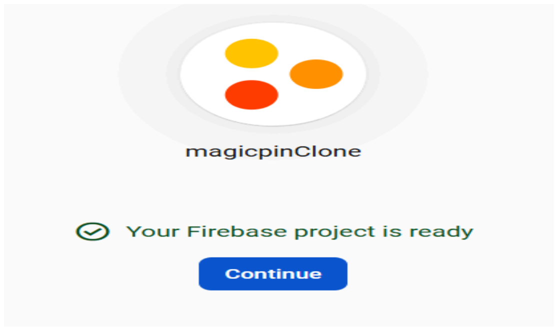
Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Breadcrumb logs in Crashlytics
- Event-based Cloud Functions triggers
- Free unlimited reporting

Enable Google Analytics for this project ✓
Recommended

If you choose to use Google Analytics, you will need to review and accept the terms and conditions prior to project creation.

After pressing Continue, your project will be created and resources will be provisioned. You will then be directed to the dashboard for the new project.



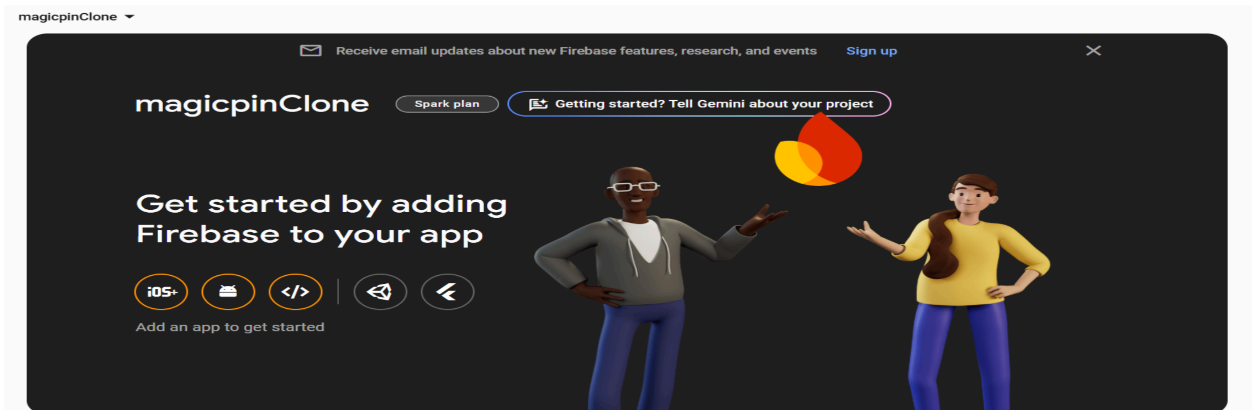
magicpinClone

Your Firebase project is ready

Step 2: Add Firebase to Your Flutter App

Adding Android support :

1. Click **"Add App"** → Select **Android**.



magicpinClone

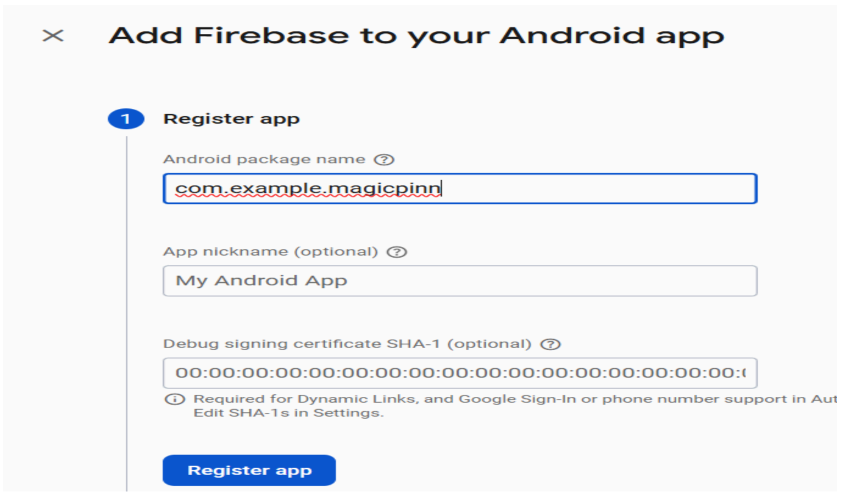
Receive email updates about new Firebase features, research, and events Sign up

magicpinClone Spark plan Getting started? Tell Gemini about your project

Get started by adding Firebase to your app

Add an app to get started

In order to add Android support to our Flutter application, select the Android logo from the dashboard. This brings us to the following screen:


Add Firebase to your Android app

1 Register app

Android package name: com.example.magicpinn

App nickname (optional): My Android App

Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Aut
Edit SHA-1s in Settings.

2. Enter the package name (found in android/app/build.gradle).

   The most important thing here is to match up the Android package name that you choose here with the one inside of our application.

   The structure consists of at least two segments. A common pattern is to use a domain name, a company name, and the application name:

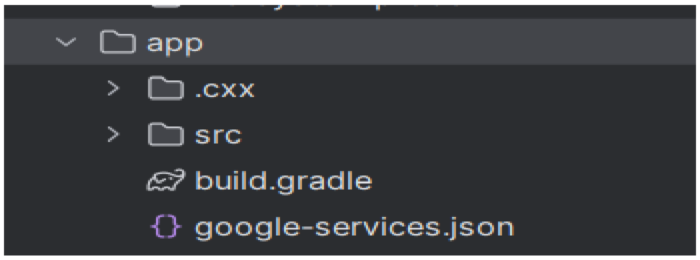
   com.example.flutterfirebaseexample

   You can copy it from applicationId in android/app/build.gradle in your code editor android/app/build.gradle

   You can skip the app nickname and debug signing keys at this stage. Select Register app to continue.

3. Download the **google-services.json** file and place it inside android/app/.

   This is important as it contains the API keys and other critical information for Firebase to use.



Adding the Firebase SDK

We'll now need to update our Gradle configuration to include the Google Services plugin.

Open android/build.gradle in your code editor and modify it to include the following:

```
dependencies {

    classpath 'com.google.gms:google-services:4.3.10'

}
```

android/buiild.gradle

```
buildscript {

    ext.kotlin_version = "1.9.22"

    repositories {

        google()

        mavenCentral()

    }

    dependencies {

        classpath 'com.google.gms:google-services:4.4.0'

        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:1.9.22"

    }
```

Finally, update the app level file at android/app/build.gradle to include the following:

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

android/app/build.gradle

```
apply plugin: 'com.android.application'
// Add this line
apply plugin: 'com.google.gms.google-services'

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:28.0.0')

}

plugins {
    id "com.android.application"
    id "kotlin-android"
    id "dev.flutter.flutter-gradle-plugin"
    id 'com.google.gms.google-services'
}

dependencies {
  // Import the Firebase BoM
   implementation platform('com.google.firebase:firebase-bom:33.9.0')
  // TODO: Add the dependencies for Firebase products you want to use
   // When using the BoM, don't specify versions in Firebase dependencies

  // Add the dependencies for any other desired Firebase products
// https://firebase.google.com/docs/android/setup#available-libraries
}
```

Or else

apply plugin: 'com.google.gms.google-services' at the end

With this update, we're essentially applying the Google Services plugin as well as looking at how other Flutter Firebase plugins can be activated such as Analytics.

Step 3: Install Firebase Dependencies

Open pubspec.yaml and add:

```
dependencies:

  firebase_core: latest_version

  firebase_auth: latest_version

  cloud_firestore: latest_version
```

Then run :

```
flutter pub get
```

Step 4: Initialize Firebase in Flutter

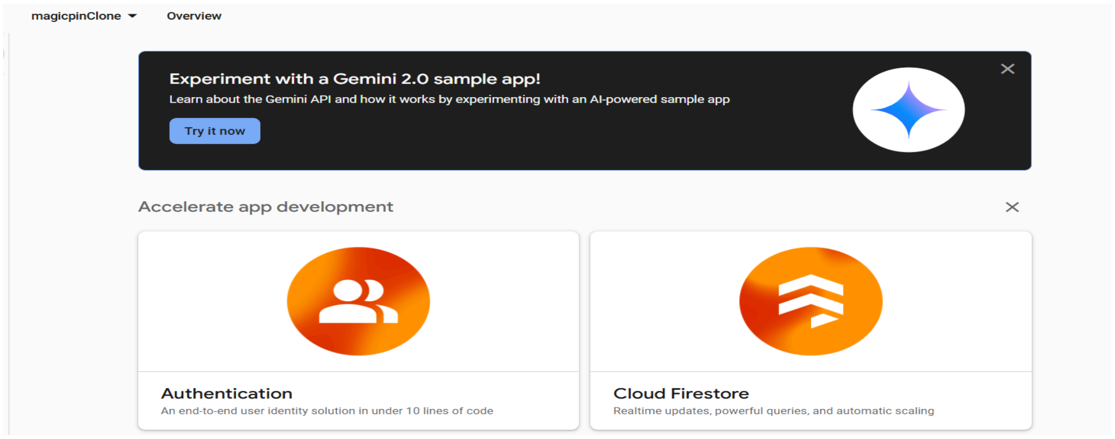
Open main.dart and update the main() function:

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

import 'package:firebase_core/firebase_core.dart';

void main() async {

  WidgetsFlutterBinding.ensureInitialized();

  await Firebase.initializeApp();

  runApp(MyApp());

}
```

Ensure that MyApp() is wrapped inside MaterialApp().

Step 5: Enable Authentication in Firebase

1. In Firebase Console, go to **Authentication** > **Sign-in method**.



magicpinClone Overview

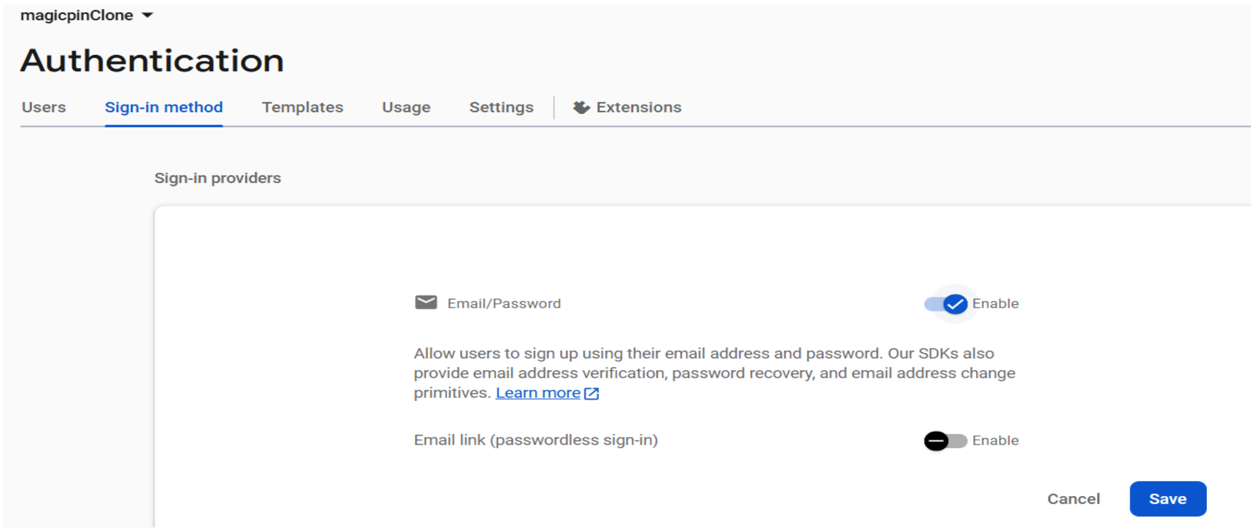
Experiment with a Gemini 2.0 sample app!
Learn about the Gemini API and how it works by experimenting with an AI-powered sample app
Try it now

Accelerate app development

Authentication
An end-to-end user identity solution in under 10 lines of code

Cloud Firestore
Realtime updates, powerful queries, and automatic scaling

2. Enable **Email/Password Authentication** (or any other method).



magicpinClone

Authentication

Users | Sign-in method | Templates | Usage | Settings | Extensions

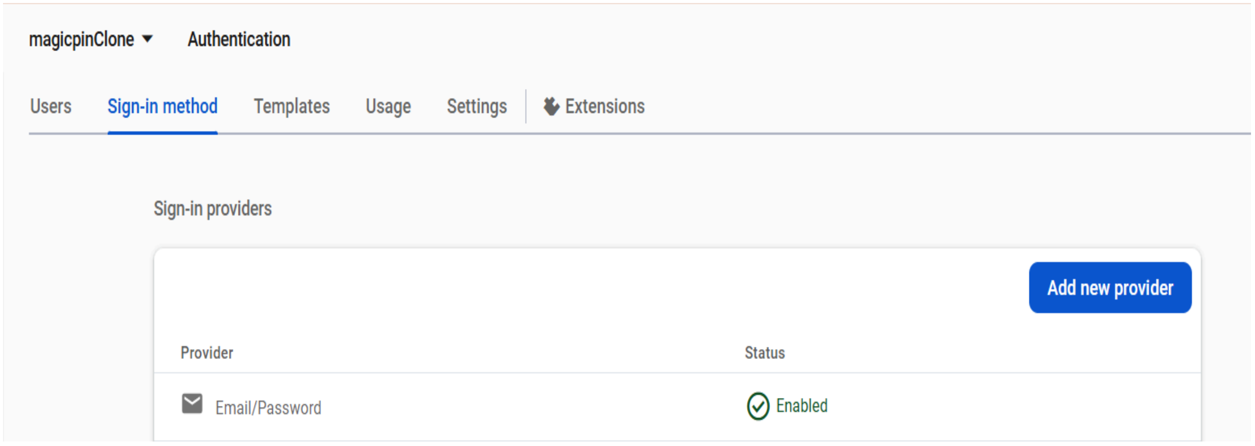
Sign-in providers

Email/Password — Enable ✓

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. Learn more

Email link (passwordless sign-in) — Enable

Cancel Save



magicpinClone Authentication

Users | Sign-in method | Templates | Usage | Settings | Extensions

Sign-in providers

Add new provider

Provider	Status
Email/Password	✓ Enabled

Step 6: Implement Firebase Authentication in Flutter

1. Register a New User (Signup)

```
import 'package:firebase_auth/firebase_auth.dart';
Future<void> signUpUser(String email, String password) async {
  try {
    UserCredential userCredential = await FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: email,
      password: password,
    );
    print("User registered: ${userCredential.user?.uid}");
  } catch (e) {
    print("Signup failed: $e");
  }
}
```

2. Login User

```
Future<void> loginUser(String email, String password) async {
  try {
    UserCredential userCredential = await FirebaseAuth.instance.signInWithEmailAndPassword(
      email: email,
      password: password,
    );
    print("User logged in: ${userCredential.user?.uid}");
  } catch (e) {
    print("Login failed: $e");
  }
}
```

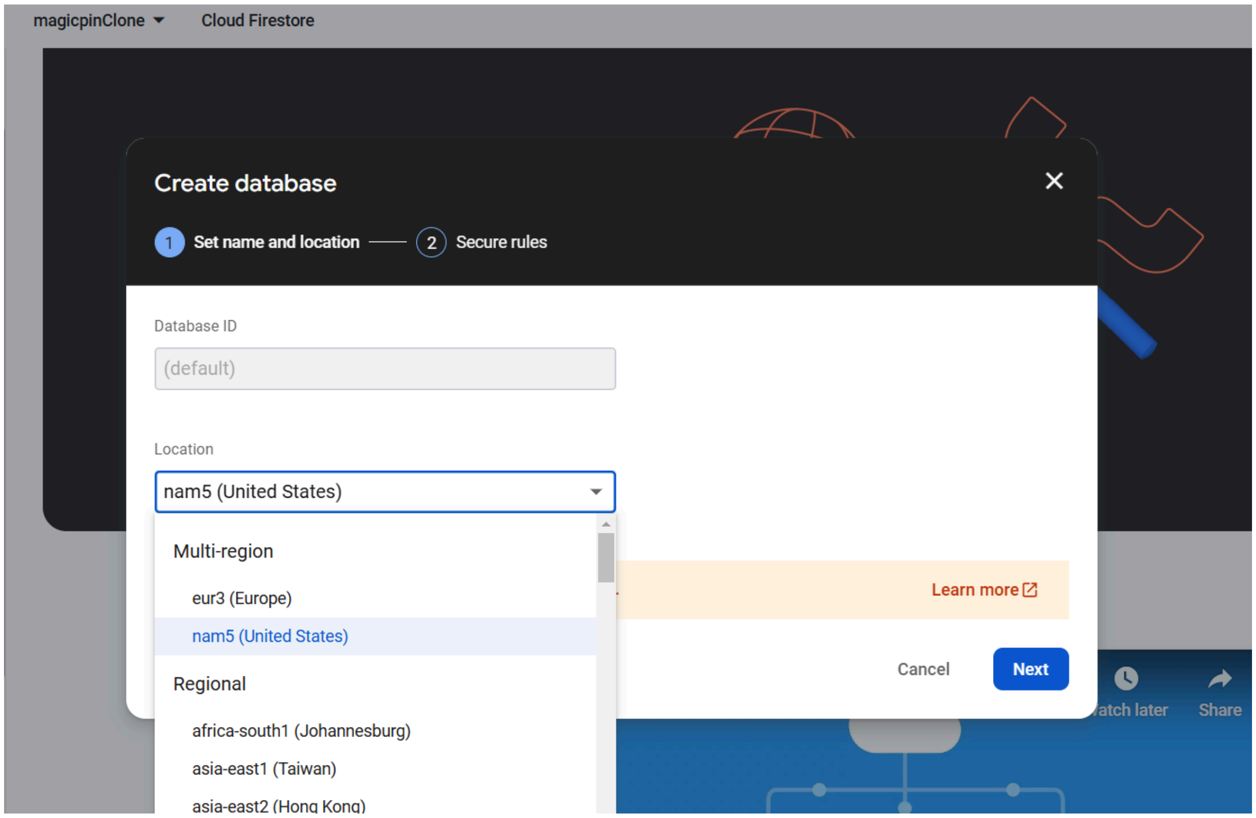
3. Logout User

```
Future<void> logoutUser() async {
  await FirebaseAuth.instance.signOut();
  print("User logged out");
}
```

Project Title: MagicPin **Roll No.: 14**

Step 7: Enable Firestore Database in Firebase

1. In Firebase Console, go to **Firestore Database**.
2. Click **Create Database**, select **Start in Test Mode**, and enable Firestore.



Database ID

(default)

Location

nam5 (United States)

Multi-region

eur3 (Europe)

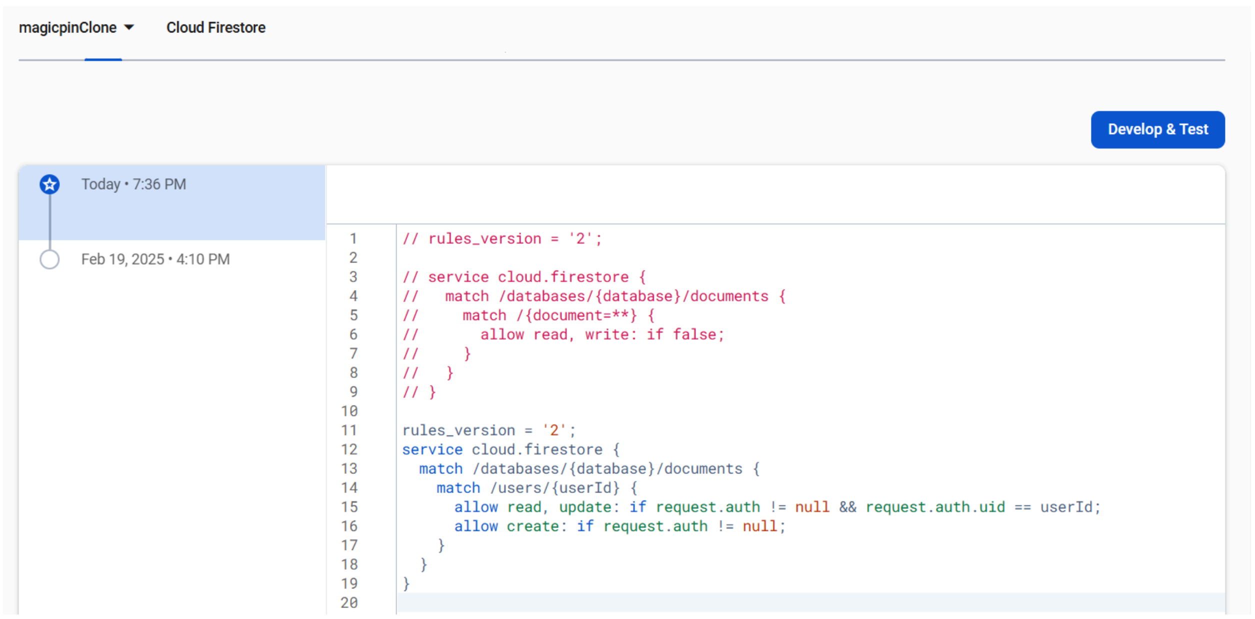
nam5 (United States)

Regional

africa-south1 (Johannesburg)

asia-east1 (Taiwan)

asia-east2 (Hong Kong)



```
// rules_version = '2';

// service cloud.firestore {
//   match /databases/{database}/documents {
//     match /{document=**} {
//       allow read, write: if false;
//     }
//   }
// }

rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /users/{userId} {
      allow read, update: if request.auth != null && request.auth.uid == userId;
      allow create: if request.auth != null;
    }
  }
}

```

Step 8: Perform Database Operations in Flutter

1. Add Data to Firestore

```
import 'package:cloud_firestore/cloud_firestore.dart';
void addUser() {
  FirebaseFirestore.instance.collection('users').add({
    'name': 'John Doe',
    'email': 'johndoe@example.com',
  });
}
```

2. Read Data from Firestore

```
void fetchUsers() {
  FirebaseFirestore.instance.collection('users').get().then((snapshot) {
    for (var doc in snapshot.docs) {
      print(doc.data());
    }
  });
}
```

3. Update Data in Firestore

```
void updateUser(String docID) {
  FirebaseFirestore.instance.collection('users').doc(docID).update({
    'name': 'Jane Doe',
  });
}
```

4. Delete Data from Firestore

```
void deleteUser(String docID) {
  FirebaseFirestore.instance.collection('users').doc(docID).delete();
}
```

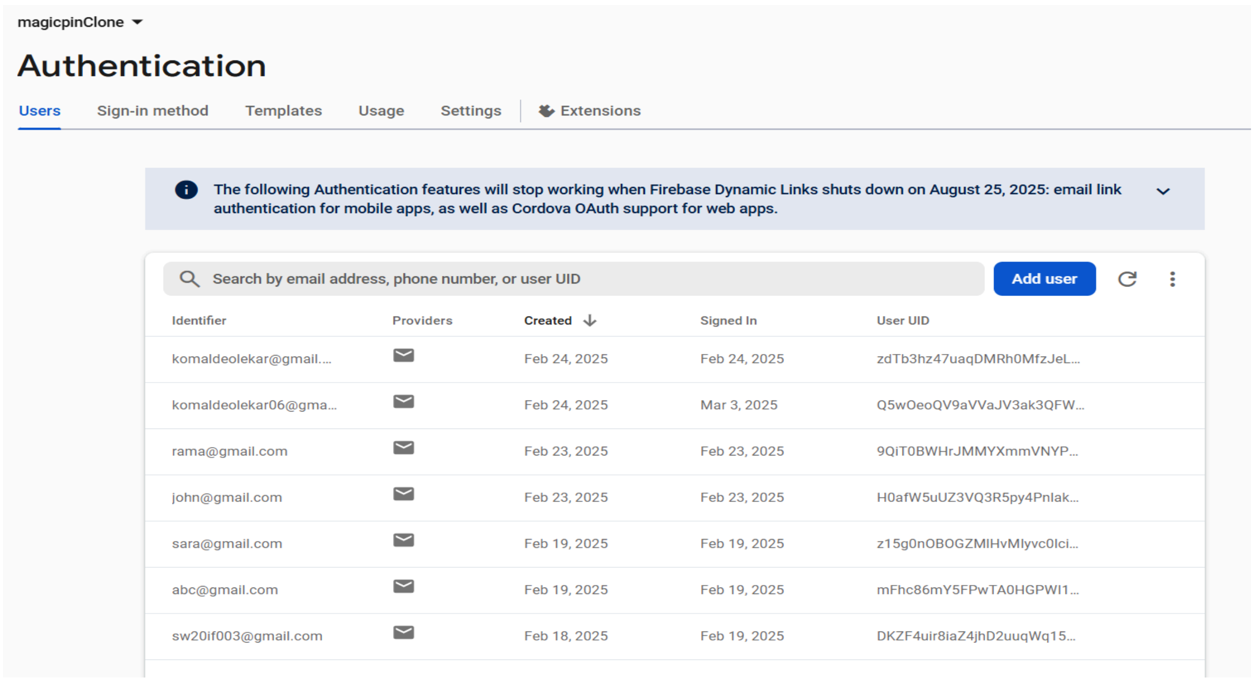
Step 9: Test Firebase Connection

Now update the codes and run the app using:

```
flutter run
```

Project Title: MagicPin **Roll No.: 14**

1. Perform actions like adding, reading, updating, and deleting data.
2. Check Firebase Console to verify the changes.



magicpinClone

Authentication

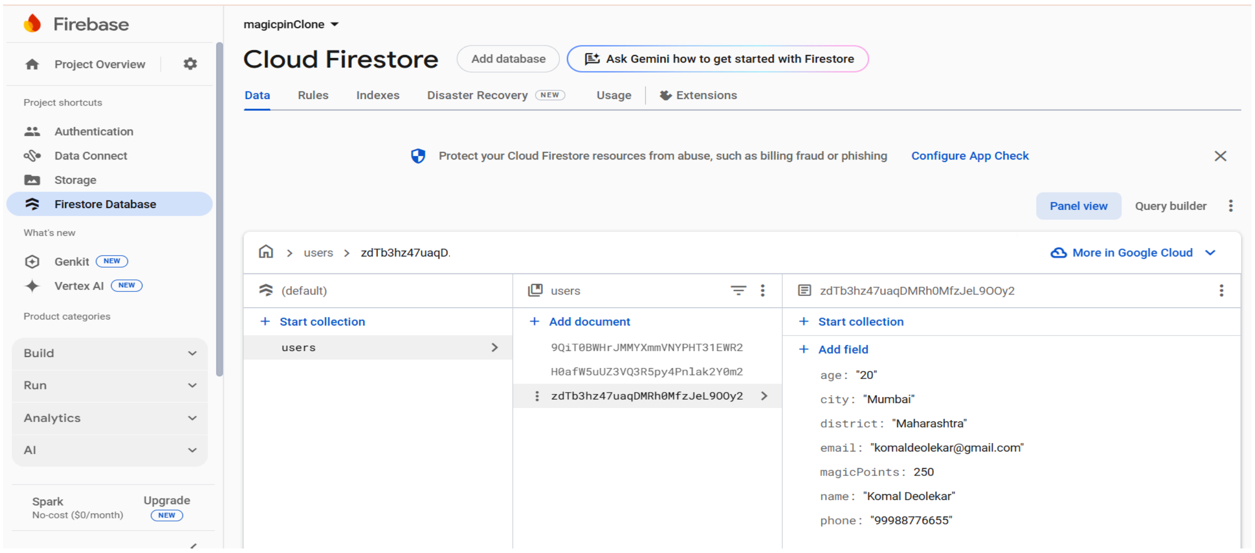
Users | Sign-in method | Templates | Usage | Settings | Extensions

The following Authentication features will stop working when Firebase Dynamic Links shuts down on August 25, 2025: email link authentication for mobile apps, as well as Cordova OAuth support for web apps.

Search by email address, phone number, or user UID

Add user

Identifier	Providers	Created	Signed In	User UID
komaldeolekar@gmail....	✉	Feb 24, 2025	Feb 24, 2025	zdTb3hz47uaqDMRh0MfzJeL...
komaldeolekar06@gma...	✉	Feb 24, 2025	Mar 3, 2025	Q5wOeoQV9aVVaJV3ak3QFW...
rama@gmail.com	✉	Feb 23, 2025	Feb 23, 2025	9QiTOBWHrJMMYXmmVNYP...
john@gmail.com	✉	Feb 23, 2025	Feb 23, 2025	H0afW5uUZ3VQ3R5py4Pnlak...
sara@gmail.com	✉	Feb 19, 2025	Feb 19, 2025	z15g0nOBOGZMIHvMlyvc0lci...
abc@gmail.com	✉	Feb 19, 2025	Feb 19, 2025	mFhc86mY5FPwTA0HGPWI1...
sw20if003@gmail.com	✉	Feb 18, 2025	Feb 19, 2025	DKZF4uir8iaZ4jhD2uuqWq15...



magicpinClone

Cloud Firestore | Add database | Ask Gemini how to get started with Firestore

Data | Rules | Indexes | Disaster Recovery NEW | Usage | Extensions

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing Configure App Check

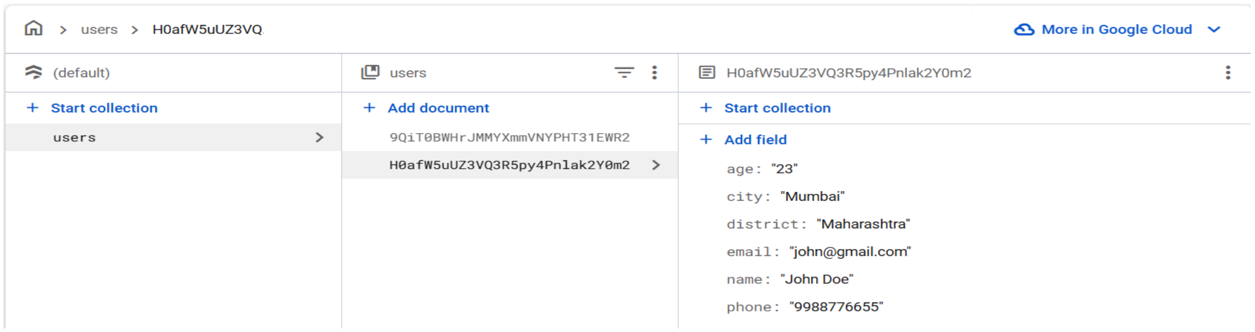
Panel view | Query builder

users > zdTb3hz47uaqD. More in Google Cloud

(default)
+ Start collection
users

users
+ Add document
9QiT0BWHrJMMYXmmVNYPHT31EWR2
H0afW5uUZ3VQ3R5py4Pnlak2Y0m2
zdTb3hz47uaqDMRh0MfzJeL9OOy2

zdTb3hz47uaqDMRh0MfzJeL9OOy2
+ Start collection
+ Add field
age: "20"
city: "Mumbai"
district: "Maharashtra"
email: "komaldeolekar@gmail.com"
magicPoints: 250
name: "Komal Deolekar"
phone: "99988776655"



users > H0afW5uUZ3VQ More in Google Cloud

(default)
+ Start collection
users

users
+ Add document
9QiT0BWHrJMMYXmmVNYPHT31EWR2
H0afW5uUZ3VQ3R5py4Pnlak2Y0m2

H0afW5uUZ3VQ3R5py4Pnlak2Y0m2
+ Start collection
+ Add field
age: "23"
city: "Mumbai"
district: "Maharashtra"
email: "john@gmail.com"
name: "John Doe"
phone: "9988776655"

Codes :

sign _up_page.dart

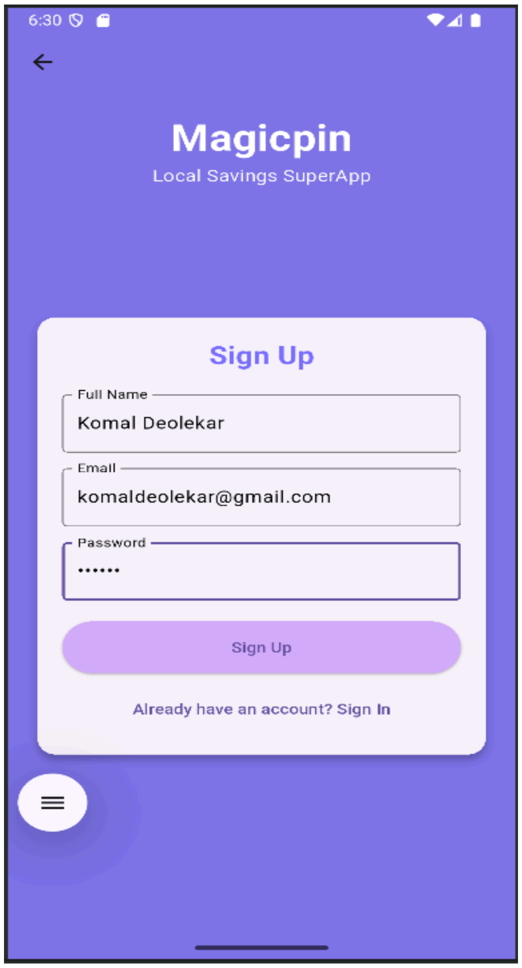
```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'main_screen.dart';
import 'sign_in_page.dart';

class SignUpPage extends StatefulWidget {
 @override
 _SignUpPageState createState() =>
_SignUpPageState();
}

class _SignUpPageState extends State<SignUpPage>
{
 final TextEditingController nameController =
TextEditingController();
 final TextEditingController emailController =
TextEditingController();
 final TextEditingController passwordController =
TextEditingController();
 final FirebaseAuth _auth = FirebaseAuth.instance;
 final FirebaseFirestore _firestore =
FirebaseFirestore.instance;
 bool isLoading = false;


 void signUp() async {
  if (nameController.text.trim().isEmpty ||
    emailController.text.trim().isEmpty ||
    passwordController.text.trim().isEmpty) {
   Fluttertoast.showToast(msg: "All fields are
required!");
   return; // Stop execution if any field is empty
  }

  setState(() => isLoading = true);

  try {
   UserCredential userCredential = await
_auth.createUserWithEmailAndPassword(
    email: emailController.text.trim(),
    password: passwordController.text.trim(),
   );

   // Save user data in Firestore
   await
_firestore.collection("users").doc(userCredential.user
!.uid).set({
    "name": nameController.text.trim(),
    "email": emailController.text.trim(),
    "magicPoints": 250,
   });

   Fluttertoast.showToast(msg: "Sign Up
Successful!");

   // Navigate to MainScreen only if sign-up is
successful
   Navigator.pushReplacement(
    context,
    // MaterialPageRoute(builder: (context) =>
MainScreen()),
     MaterialPageRoute(builder: (context) =>
SignInPage()),

   );
  } catch (e) {
   Fluttertoast.showToast(msg: "Sign Up Failed:
${e.toString()}");
  } finally {
   setState(() => isLoading = false);
  }
 }


 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: Color(0xFF7D73E8),
   appBar: AppBar(
    centerTitle: true,
    backgroundColor: Color(0xFF7D73E8),
    elevation: 0, // Remove shadow
   ),
   body: Center(
    child: Padding(
     padding: const EdgeInsets.all(20.0),
        child: Column(
          children: [
            Text(
              "Magicpin",
              style: TextStyle(fontSize: 35, fontWeight:
FontWeight.bold, color: Colors.white),
            ),
            Text(
              'Local Savings SuperApp',
              style: TextStyle(
                fontSize: 16,
                color: Colors.white,
              ),
            ),
            SizedBox(height: 120),
            Card(
              elevation: 5,
              shape:
RoundedRectangleBorder(borderRadius:
BorderRadius.circular(15)),
              child: Padding(
                padding: const EdgeInsets.all(20.0),
                child: Column(
                  mainAxisSize: MainAxisSize.min,
                  children: [
                    Text("Sign Up", style:
TextStyle(fontSize: 24, fontWeight:
FontWeight.bold,color: Color(0xFF796EFF) )),
                    SizedBox(height: 20),
                    TextField(
                      controller: nameController,
                      style: TextStyle(color: Colors.black),
                      decoration: InputDecoration(
                        labelText: "Full Name",
                        border: OutlineInputBorder(),
                        labelStyle: TextStyle(color:
Colors.black),
                      ),
                    ),
                    SizedBox(height: 15),
                    TextField(
                      controller: emailController,
                      style: TextStyle(color: Colors.black),
                      decoration: InputDecoration(
                        labelText: "Email",
                        border: OutlineInputBorder(),
                        labelStyle: TextStyle(color:
Colors.black),
                      ),
                    ),
                    SizedBox(height: 15),
                    TextField(
                      controller: passwordController,
                      style: TextStyle(color: Colors.black),
                      decoration: InputDecoration(
                        labelText: "Password",
                        border: OutlineInputBorder(),
                        labelStyle: TextStyle(color:
Colors.black),
                      ),
                      obscureText: true,
                    ),
                    SizedBox(height: 20),
                    isLoading
                        ? CircularProgressIndicator()
                        : ElevatedButton(
                      onPressed: signUp,
                      style:
ElevatedButton.styleFrom(minimumSize:
Size(double.infinity, 50),backgroundColor:
Color.fromRGBO(212, 171, 250, 1)),
                      child: Text("Sign Up"),
                    ),
                    SizedBox(height: 10),
                    TextButton(
                      onPressed: () =>
Navigator.pushNamed(context, '/login'),
                      child: Text("Already have an account?
Sign In"),
                    ),
                  ],
                ),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

Sign_in_page.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'main_screen.dart'; // Redirect to home after
login
import 'phone_number_page.dart';

class SignInPage extends StatefulWidget {
  @override
  _SignInPageState createState() =>
_SignInPageState();
}

class _SignInPageState extends State<SignInPage> {
  final TextEditingController emailController =
TextEditingController();
  final TextEditingController passwordController =
TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;
  bool isLoading = false;

  void signIn() async {
    setState(() => isLoading = true);

    try {
      await _auth.signInWithEmailAndPassword(
        email: emailController.text.trim(),
        password: passwordController.text.trim(),
      );

      Fluttertoast.showToast(msg: "Login
Successful!");
      // Navigator.pushReplacement(context,
MaterialPageRoute(builder: (context) =>
MainScreen()));
      Navigator.pushReplacement(context,
MaterialPageRoute(builder: (context) =>
PhoneNumberPage()));
    } catch (e) {
      Fluttertoast.showToast(msg: "Login Failed:
${e.toString()}");
    } finally {
      setState(() => isLoading = false);
    }
  }

  void resetPassword() async {
    if (emailController.text.isNotEmpty) {
      try {
        await _auth.sendPasswordResetEmail(email:
emailController.text.trim());
        Fluttertoast.showToast(msg: "Password reset
email sent!");
      } catch (e) {
        Fluttertoast.showToast(msg: "Error:
${e.toString()}");
      }
    } else {
      Fluttertoast.showToast(msg: "Enter your email to
reset password");
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Color(0xFF7D73E8),
      body: Center(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            // mainAxisSize: MainAxisSize.min,
            children: [
              SizedBox(height: 100), // Adds space at the
top
              Text(
                "Magicpin",
                style: TextStyle(fontSize: 35, fontWeight:
FontWeight.bold, color: Colors.white),
              ),
              Text(
                'Local Savings SuperApp',
                style: TextStyle(
                  fontSize: 16,
                  color: Colors.white,
                ),
              ),
              SizedBox(height: 120),
              Card(
                elevation: 5,
                shape:
RoundedRectangleBorder(borderRadius:
BorderRadius.circular(15)),
                child: Padding(
              padding: const EdgeInsets.all(20.0),
              child: Column(
                mainAxisSize: MainAxisSize.min,
                children: [
                  Text("Sign In", style:
TextStyle(fontSize: 24, fontWeight: FontWeight.bold
, color: Color(0xFF796EFF))),
                  SizedBox(height: 20),
                  TextField(
                    controller: emailController,
                    style: TextStyle(color: Colors.black),
// Input text in black
                    decoration: InputDecoration(
                      labelText: "Email",
                      border: OutlineInputBorder(),
                    ),
                  ),
                  SizedBox(height: 15),
                  TextField(
                    controller: passwordController,
                    style: TextStyle(color: Colors.black),
// Input text in black
                    decoration: InputDecoration(
                      labelText: "Password",
                      border: OutlineInputBorder(),
                    ),
                    obscureText: true,
                  ),
                  SizedBox(height: 20),
                  isLoading
                      ? CircularProgressIndicator()
                      : ElevatedButton(
                    onPressed: signIn,
                    style:
ElevatedButton.styleFrom(minimumSize:
Size(double.infinity, 50),
                      backgroundColor:
Color.fromRGBO(212, 171, 250, 1), ),
                    child: Text("Sign In"),

                  ),
                  SizedBox(height: 10),
                  TextButton(onPressed: resetPassword,
child: Text("Forgot Password?")),
                  TextButton(
                    onPressed: () =>
Navigator.pushNamed(context, '/signup'),
                    child: Text("Don't have an account?
Sign Up"),
                  ),
                ],
              ),
            ),
          ),
        ],
      ),
    ),
  ),
);
  }
}
```

Project Title: MagicPin **Roll No.: 14**

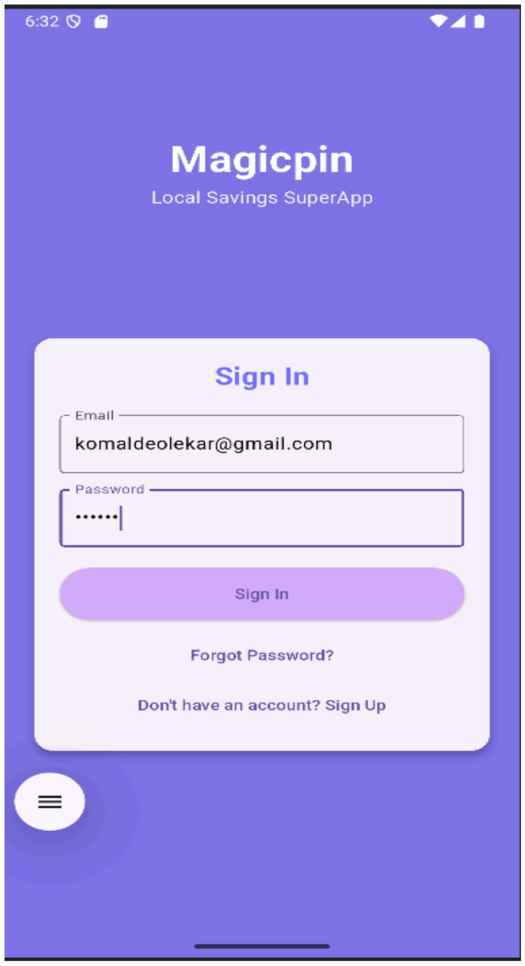
ScreenShots :



Komal Deolekar

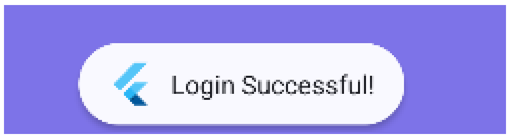
komaldeolekar@gmail.com

••••••



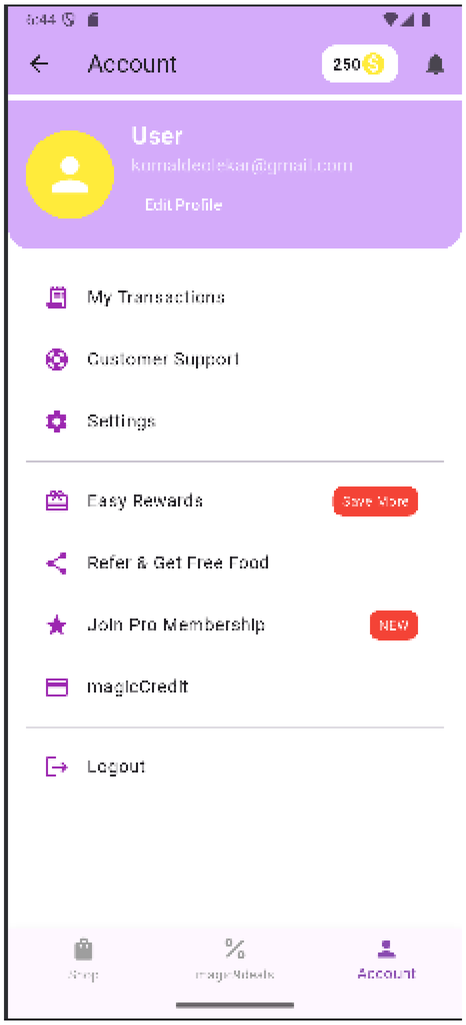
komaldeolekar@gmail.com

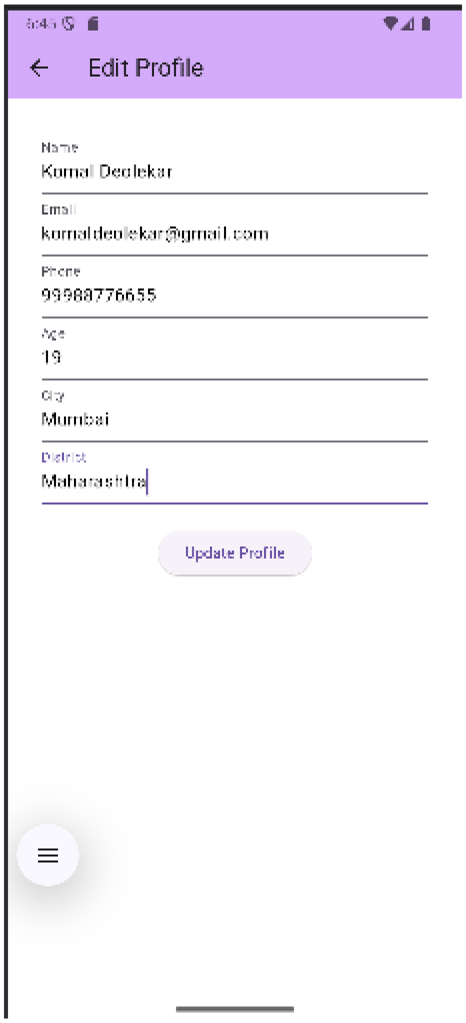
••••••



Login Successful!

Project Title: MagicPin **Roll No.: 14**





Name
Komal Deolekar

Email
komaldeolekar@gmail.com

Phone
99988776655

Age
19

City
Mumbai

District
Maharashtra