MPL Practical 06

Aim: To Connect Flutter UI with fireBase database.

Theory:

Firebase is a cloud-based backend service that provides features such as authentication, real-time database, cloud storage, and hosting for mobile applications. In Flutter, Firebase is integrated using the firebase_auth and cloud_firestore packages, enabling seamless communication between the app and Firebase services.

Implementation in Our Project
In this experiment, we integrated Firebase into our Flutter application, MedLink, to enable user authentication. This includes email/password authentication, Google Sign-In, password reset functionality, and email verification.

Key Features Implemented

1. User Registration with Email & Password
    - Users can sign up using their email and password.
    - After registration, an email verification is sent to the user.
    - Only verified users are allowed to log in.

2. User Login
    - Users can log in with their registered email and password.
    - The system checks if the email is verified before granting access.

3. Google Sign-In
    - Users can sign in with their Google accounts using Firebase Authentication.

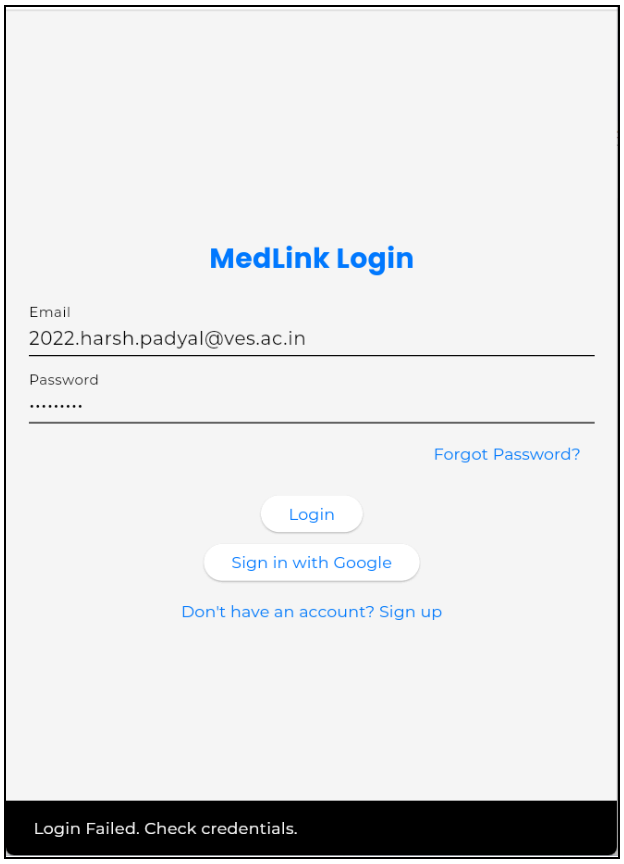
4. Password Reset
    - If a user forgets their password, they can request a password reset email.

5. Logout Functionality
    - A logout button allows users to sign out from Firebase authentication.

---

utils/auth_service.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:google_sign_in/google_sign_in.dart';

class AuthService {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GoogleSignIn _googleSignIn = GoogleSignIn(
    clientId: "641097014683-pr9jdi0s76of2nb3suugi6ualiij0lq1.apps.googleusercontent.com",
  );

  // Sign Up with Email & Password
  Future<User?> signUp(String email, String password) async {
    try {
      print("Attempting to sign up user: $email");
      UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
        email: email,
        password: password,
      );

      // Send email verification
      await userCredential.user?.sendEmailVerification();
      print("Verification email sent to: ${userCredential.user?.email}");

      return userCredential.user;
    } catch (e) {
      print("Sign Up Error: $e");
      return null;
    }
  }

  // Login with Email & Password
  Future<User?> signIn(String email, String password) async {
    try {
      print("Attempting to sign in user: $email");
      UserCredential userCredential = await _auth.signInWithEmailAndPassword(
        email: email,
        password: password,
      );
      if (userCredential.user?.emailVerified == true) {
        print("Login Successful for: ${userCredential.user?.email}");
        return userCredential.user;
      } else {
        print("Login failed: Email not verified");
        await _auth.signOut();
        return null;
      }
    } catch (e) {
      print("Login Error: $e");
      return null;
    }
  }

  // Google Sign-In
  Future<User?> signInWithGoogle() async {
    try {
      final GoogleSignInAccount? googleUser = await _googleSignIn.signIn();
      if (googleUser == null) return null; // User canceled

      final GoogleSignInAuthentication googleAuth = await googleUser.authentication;
      final AuthCredential credential = GoogleAuthProvider.credential(
        accessToken: googleAuth.accessToken,
        idToken: googleAuth.idToken,
      );

      UserCredential userCredential = await _auth.signInWithCredential(credential);
      return userCredential.user;
    } catch (e) {
      print("Google Sign-In Error: $e");
      return null;
    }
  }

  // Sign Out (Google & Email)
  Future<void> signOut() async {
    await _auth.signOut();
    await _googleSignIn.signOut();
  }

  // Get current user
  User? getCurrentUser() {
    return _auth.currentUser;
  }

  // Reset Password
  Future<bool> resetPassword(String email) async {
    try {
      await _auth.sendPasswordResetEmail(email: email);
      return true; // Email sent successfully
    } catch (e) {
      print("Password Reset Error: $e");
      return false; // Failed to send email
    }
  }
}
```



MedLink Login

Email
2022.harsh.padyal@ves.ac.in

Password
·········

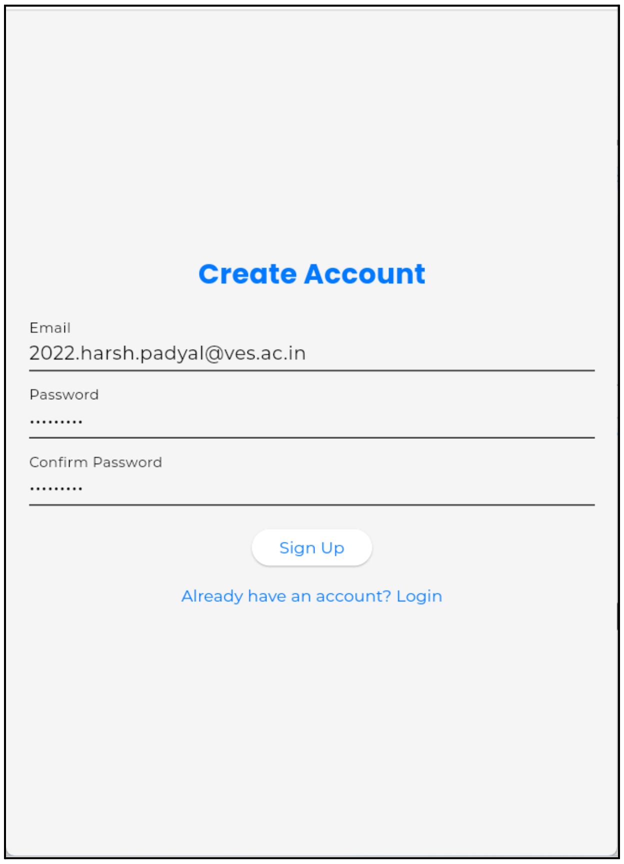
Forgot Password?

Login

Sign in with Google

Don't have an account? Sign up

Login Failed. Check credentials.



Create Account

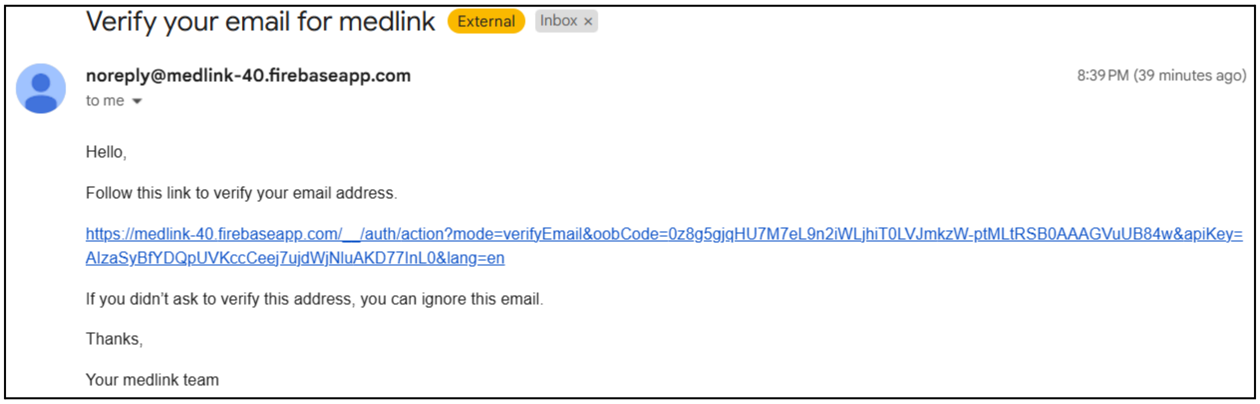
Email
2022.harsh.padyal@ves.ac.in

Password
·········

Confirm Password
·········

Sign Up

Already have an account? Login



Verify your email for medlink External Inbox ×

noreply@medlink-40.firebaseapp.com
to me

8:39 PM (39 minutes ago)

Hello,

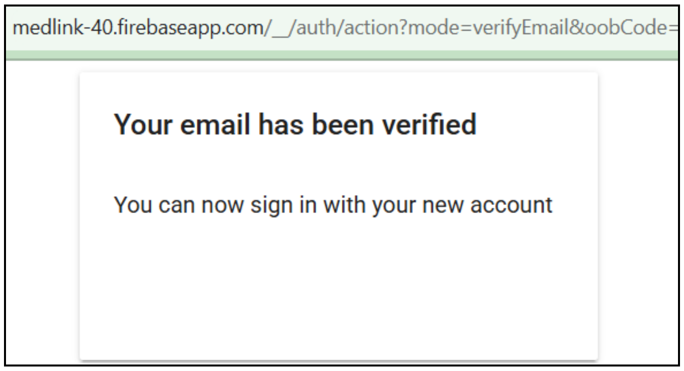
Follow this link to verify your email address.

https://medlink-40.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=0z8g5gjqHU7M7eL9n2iWLjhiT0LVJmkzW-ptMLtRSB0AAAGVuUB84w&apiKey=AIzaSyBfYDQpUVKccCeej7ujdWjNluAKD77InL0&lang=en

If you didn't ask to verify this address, you can ignore this email.

Thanks,

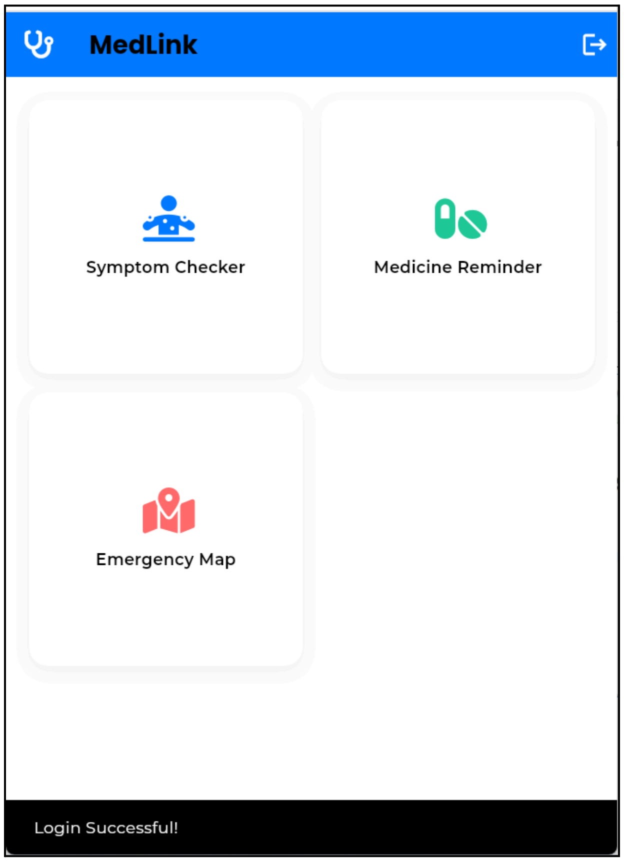
Your medlink team

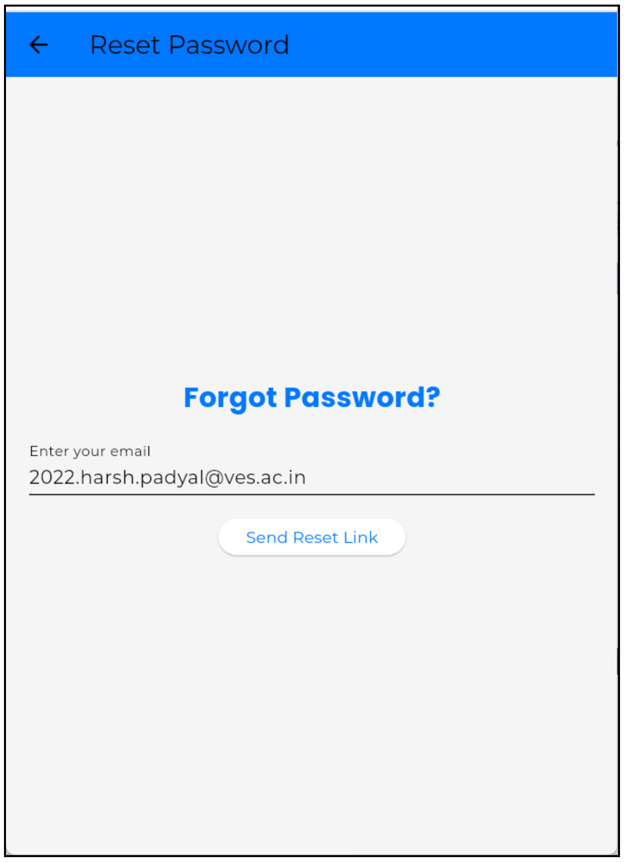


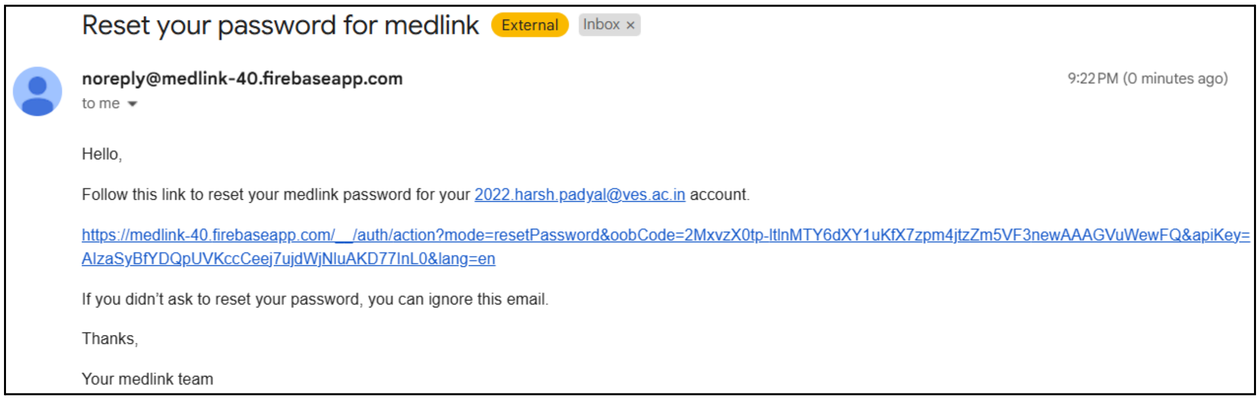
medlink-40.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=

Your email has been verified

You can now sign in with your new account







Reset your password for medlink External Inbox ×

noreply@medlink-40.firebaseapp.com 9:22 PM (0 minutes ago)
to me

Hello,

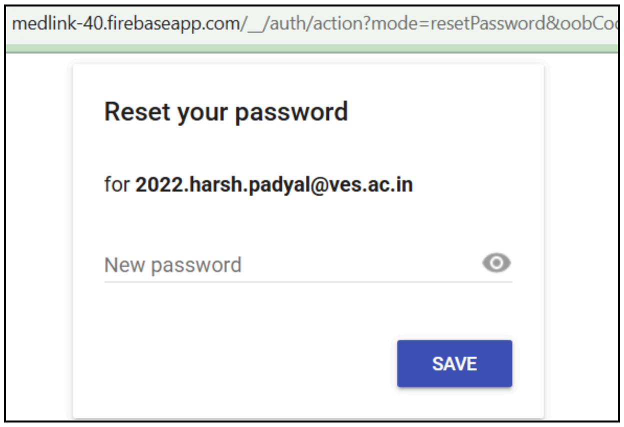
Follow this link to reset your medlink password for your 2022.harsh.padyal@ves.ac.in account.

https://medlink-40.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=2MxvzX0tp-ItlnMTY6dXY1uKfX7zpm4jtzZm5VF3newAAAGVuWewFQ&apiKey=AIzaSyBfYDQpUVKccCeej7ujdWjNluAKD77InL0&lang=en

If you didn't ask to reset your password, you can ignore this email.

Thanks,

Your medlink team

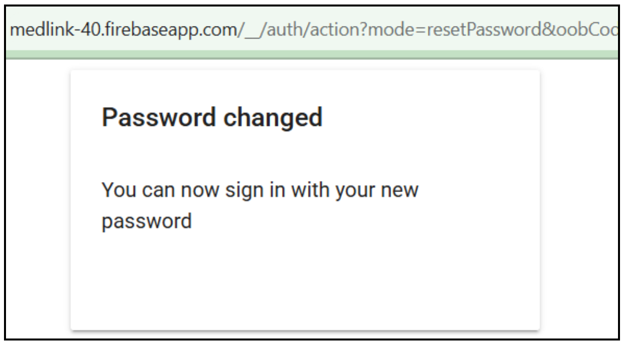

medlink-40.firebaseapp.com/__/auth/action?mode=resetPassword&oobCo

Reset your password

for **2022.harsh.padyal@ves.ac.in**

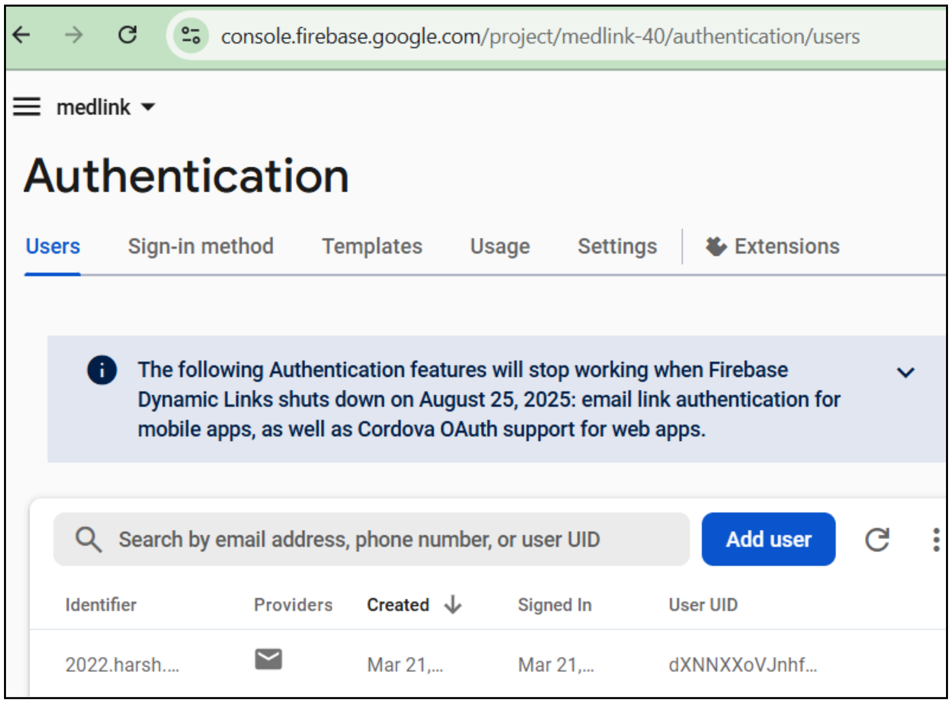
New password

SAVE


medlink-40.firebaseapp.com/__/auth/action?mode=resetPassword&oobCod

Password changed

You can now sign in with your new password


console.firebase.google.com/project/medlink-40/authentication/users

medlink

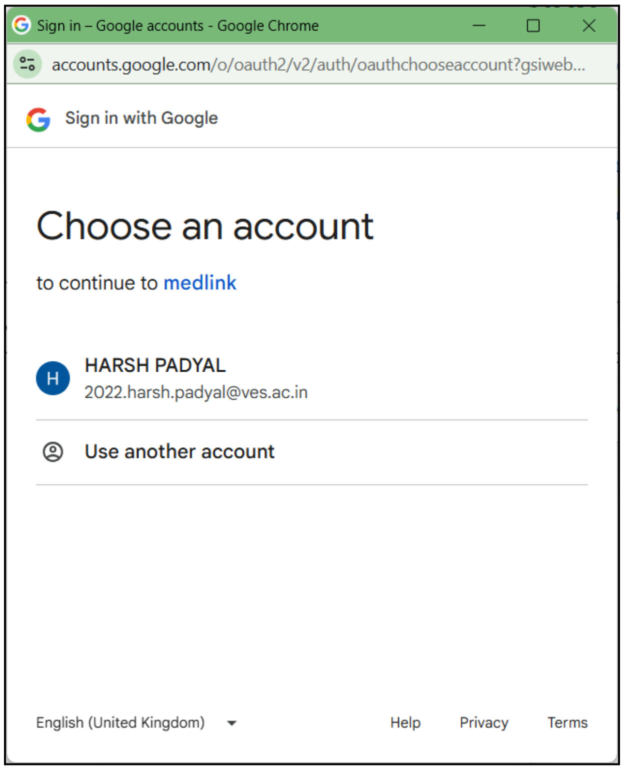
Authentication

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
2022.harsh....	✉	Mar 21,...	Mar 21,...	dXNNXXoVJnhf...


Sign in – Google accounts - Google Chrome

accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?gsiweb...

Sign in with Google

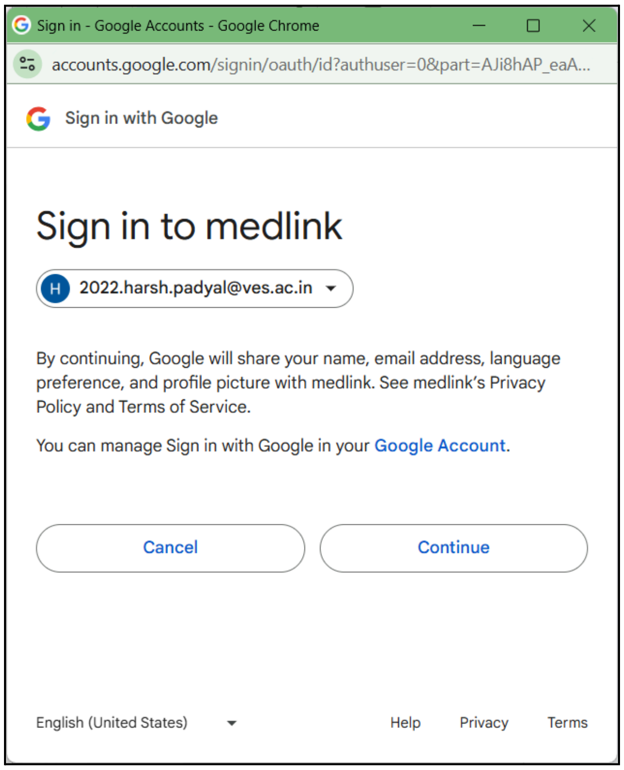
Choose an account

to continue to medlink

HARSH PADYAL
2022.harsh.padyal@ves.ac.in

Use another account

English (United Kingdom) Help Privacy Terms


Sign in - Google Accounts - Google Chrome

accounts.google.com/signin/oauth/id?authuser=0&part=AJi8hAP_eaA...

Sign in with Google

Sign in to medlink

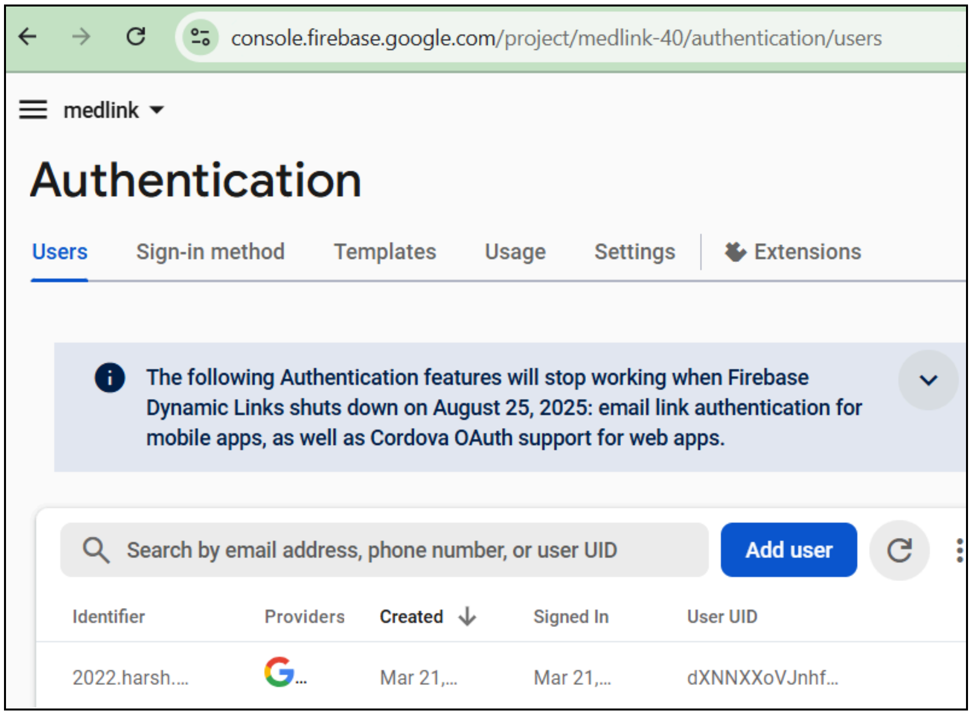
2022.harsh.padyal@ves.ac.in

By continuing, Google will share your name, email address, language preference, and profile picture with medlink. See medlink's Privacy Policy and Terms of Service.

You can manage Sign in with Google in your Google Account.

Cancel Continue

English (United States) Help Privacy Terms


console.firebase.google.com/project/medlink-40/authentication/users
medlink
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
2022.harsh....	G...	Mar 21,...	Mar 21,...	dXNNXXoVJnhf...

Conclusion

We successfully integrated Firebase Authentication with our Flutter app, implementing email/password signup, Google Sign-In, email verification, and password reset features. During development, we encountered issues such as email verification not updating immediately and Google Sign-In configuration errors, which we resolved by manually refreshing user authentication state and correctly setting up Firebase credentials.