Aim: To Connect Flutter UI with fireBase database.

Theory:

Flutter is an open-source UI toolkit that allows developers to build cross-platform applications using a single codebase. Firebase, a cloud-based Backend-as-a-Service (BaaS) platform by Google, provides authentication, real-time databases, cloud storage, and other backend functionalities to support mobile and web applications. This experiment integrates Firebase Authentication with Flutter for user management.

Implemented in our Code

Firebase Setup: Configured Firebase project, enabled Authentication, and added Firebase dependencies.

Authentication (auth_service.dart):

- Signup: Registers users and sends email verification.
- Login: Allows access only after email verification.
- Password Reset: Sends reset email.
- Google Sign-In: Enables authentication via Google.
- Logout: Signs out users.

UI Screens:

- Signup & Login: User-friendly forms with validation.
- Reset Password: Allows password recovery.
- Auth Checker: Redirects users based on login status.

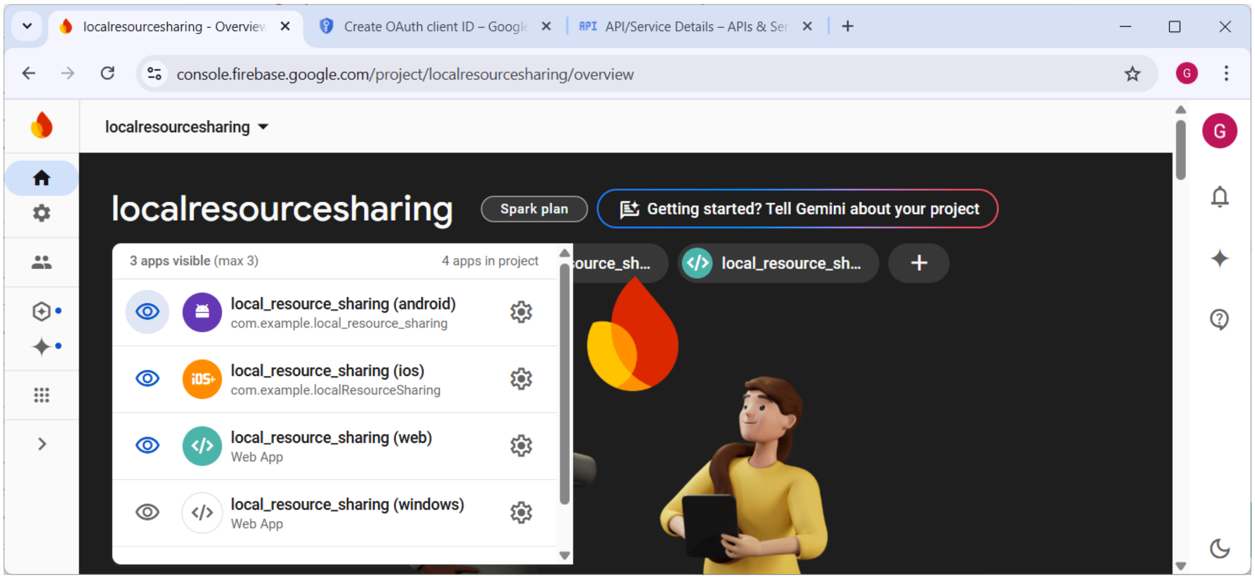
Code

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

/// **AuthService Class**
/// Handles authentication functionalities using Firebase Authentication.
class AuthService {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GoogleSignIn _googleSignIn = GoogleSignIn(
    clientId: "45384462772-ia06jsdakisp6vhuf6pa3sf1n3kbv0cr.apps.googleusercontent.com", // Add your Web Client ID here
  );

  /// **Checks if a user is logged in**
  bool isUserLoggedIn() {
    return _auth.currentUser != null && _auth.currentUser!.emailVerified;
  }

  /// **Gets the current logged-in user**
```

```
User? getCurrentUser() {
 return _auth.currentUser;
}

/// **Signs up a user and sends an email verification**
Future<bool> signUp(String email, String password) async {
 try {
  UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
   email: email,
   password: password,
  );

  // Send email verification
  await userCredential.user!.sendEmailVerification();
  return true;
 } catch (e) {
  print("Sign Up Error: $e");
  return false;
 }
}

/// **Logs in a user only if email is verified**
Future<User?> signIn(String email, String password) async {
 try {
  UserCredential userCredential = await _auth.signInWithEmailAndPassword(
   email: email,
   password: password,
  );

  if (userCredential.user!.emailVerified) {
   return userCredential.user;
  } else {
   await userCredential.user!.sendEmailVerification();
   print("Please verify your email first.");
   return null;
  }
 } catch (e) {
  print("Login Error: $e");
  return null;
 }
}
```

```
/// **Logs out the user (Google & Email)**
Future<void> signOut() async {
  await _auth.signOut();
  await _googleSignIn.signOut();
}

/// **Reset Password via Email**
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
  }
}

/// **Google Sign-In Authentication**
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);
    if (userCredential.user!.emailVerified) {
      return userCredential.user;
    } else {
      print("Please verify your email first.");
      return null;
    }
  } catch (e) {
    print("Google Sign-In Error: $e");
    return null;
  }
```

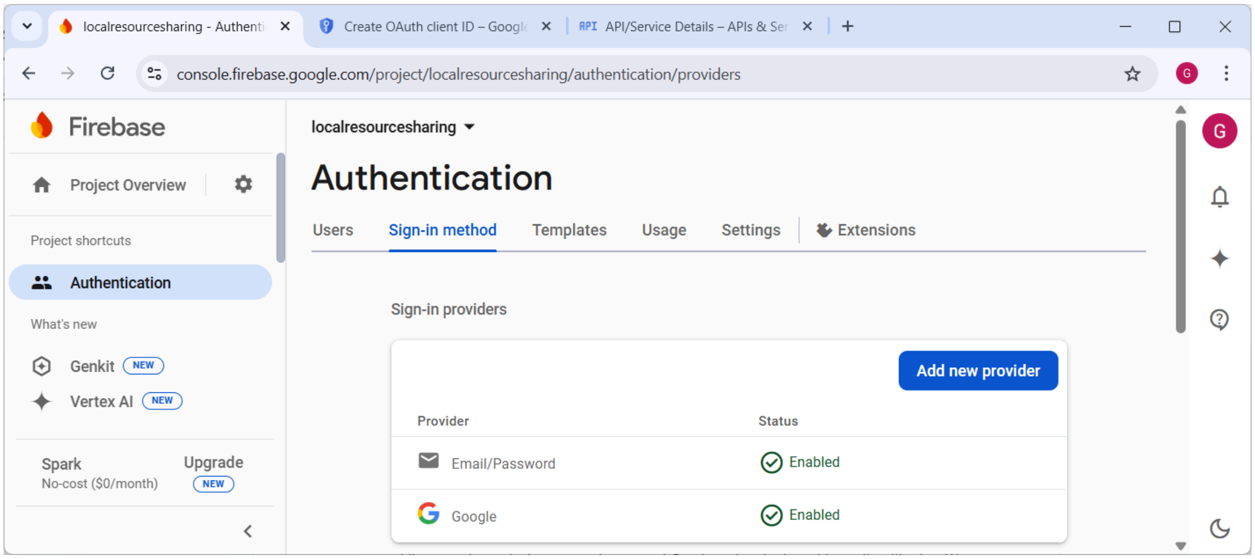
```
 }
}
```


localresourcesharing - Overview
Create OAuth client ID – Google
API/Service Details – APIs & Ser
console.firebase.google.com/project/localresourcesharing/overview
localresourcesharing
Spark plan
Getting started? Tell Gemini about your project
ource_sh...
local_resource_sh...

localresourcesharing

3 apps visible (max 3) 4 apps in project

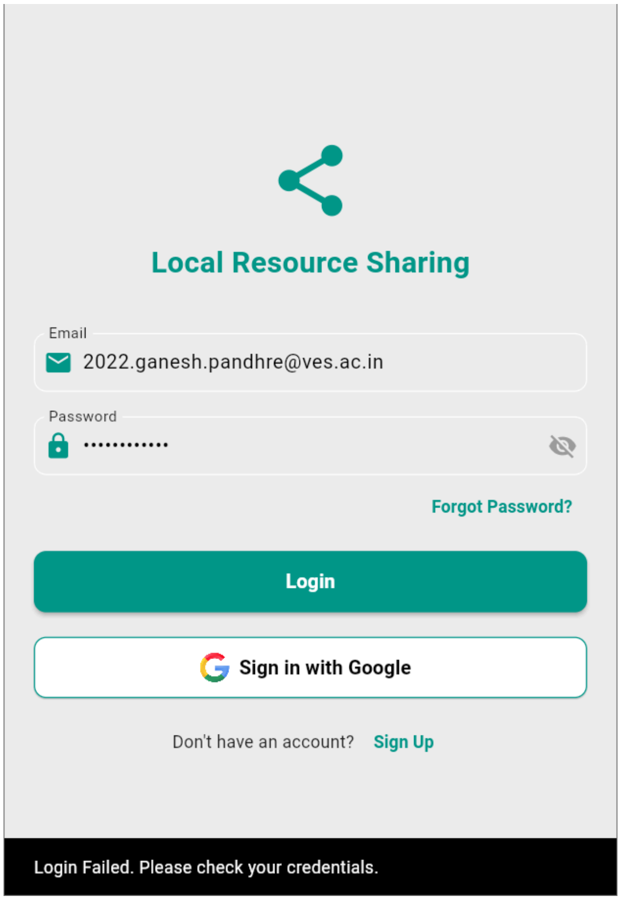
- local_resource_sharing (android)
  com.example.local_resource_sharing
- local_resource_sharing (ios)
  com.example.localResourceSharing
- local_resource_sharing (web)
  Web App
- local_resource_sharing (windows)
  Web App



Authentication

Sign-in providers

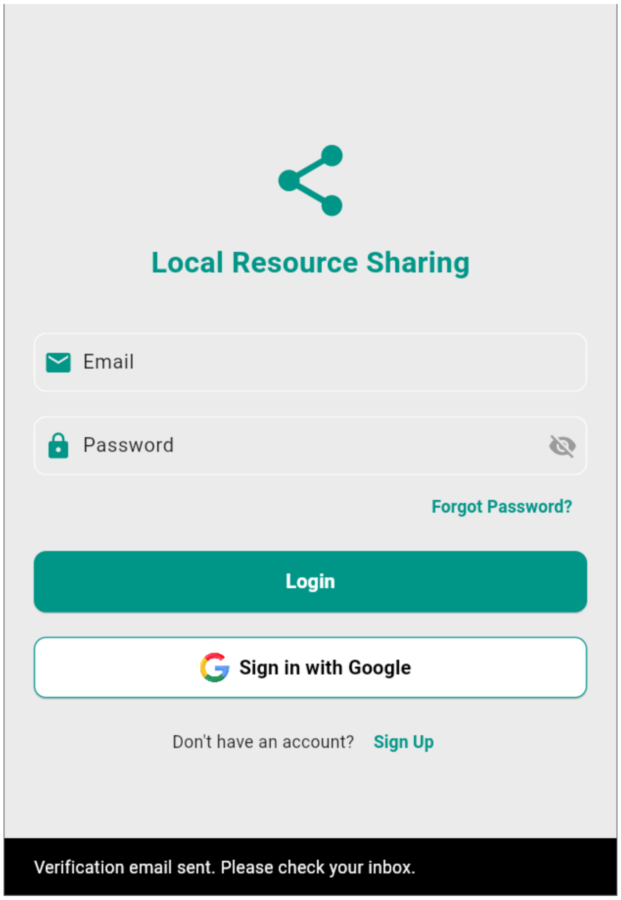
Provider	Status
Email/Password	✓ Enabled
Google	✓ Enabled



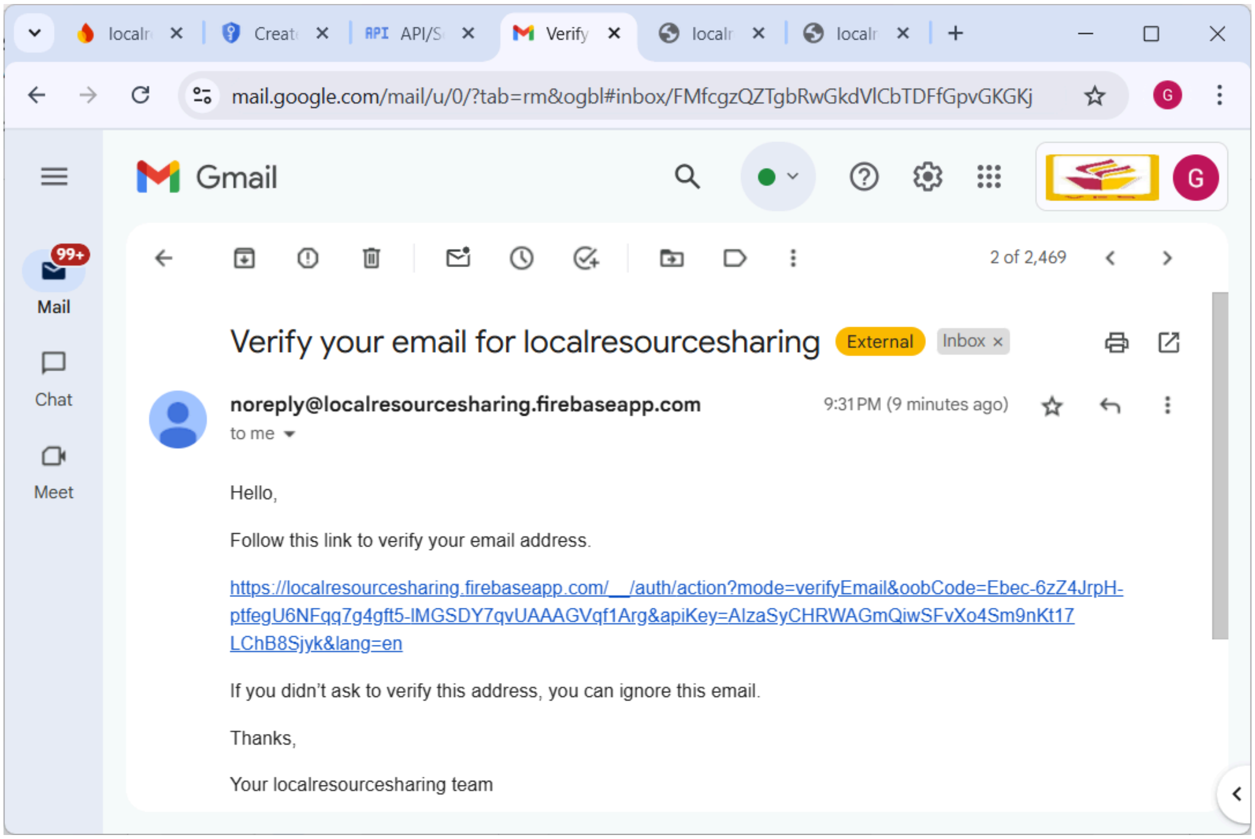
Email
2022.ganesh.pandhre@ves.ac.in

Password
••••••••••••

Login Failed. Please check your credentials.



Verification email sent. Please check your inbox.



mail.google.com/mail/u/0/?tab=rm&ogbl#inbox/FMfcgzQZTgbRwGkdVlCbTDFfGpvGKGKj

Verify your email for localresourcesharing

noreply@localresourcesharing.firebaseapp.com 9:31 PM (9 minutes ago)

to me

Hello,

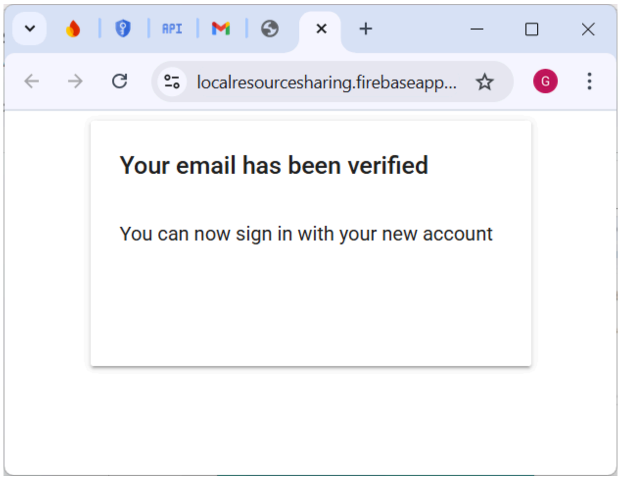
Follow this link to verify your email address.

https://localresourcesharing.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=Ebec-6zZ4JrpH-ptfegU6NFqq7g4gft5-lMGSDY7qvUAAAGVqf1Arg&apiKey=AIzaSyCHRWAGmQiwSFvXo4Sm9nKt17LChB8Sjyk&lang=en

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

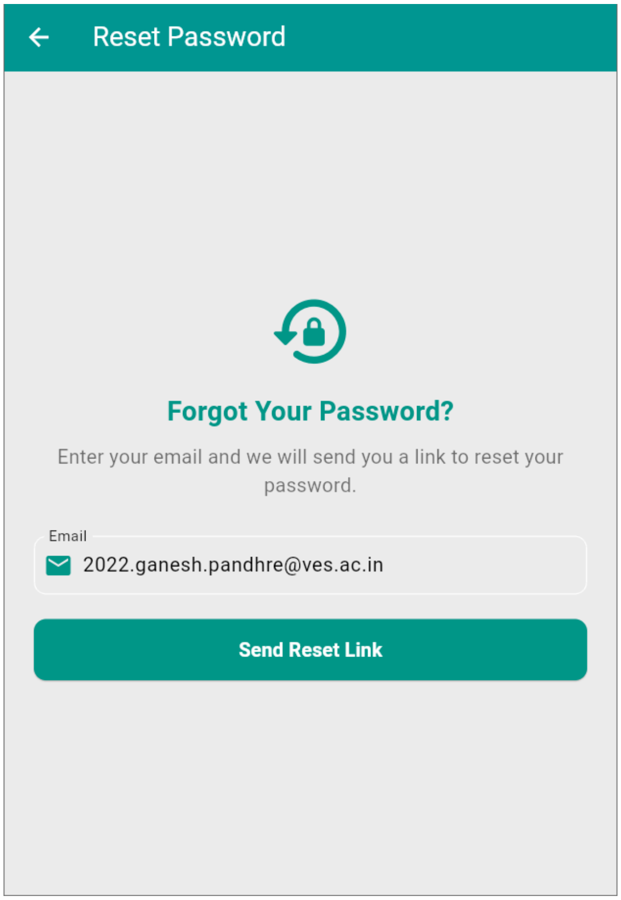
Thanks,

Your localresourcesharing team



Your email has been verified

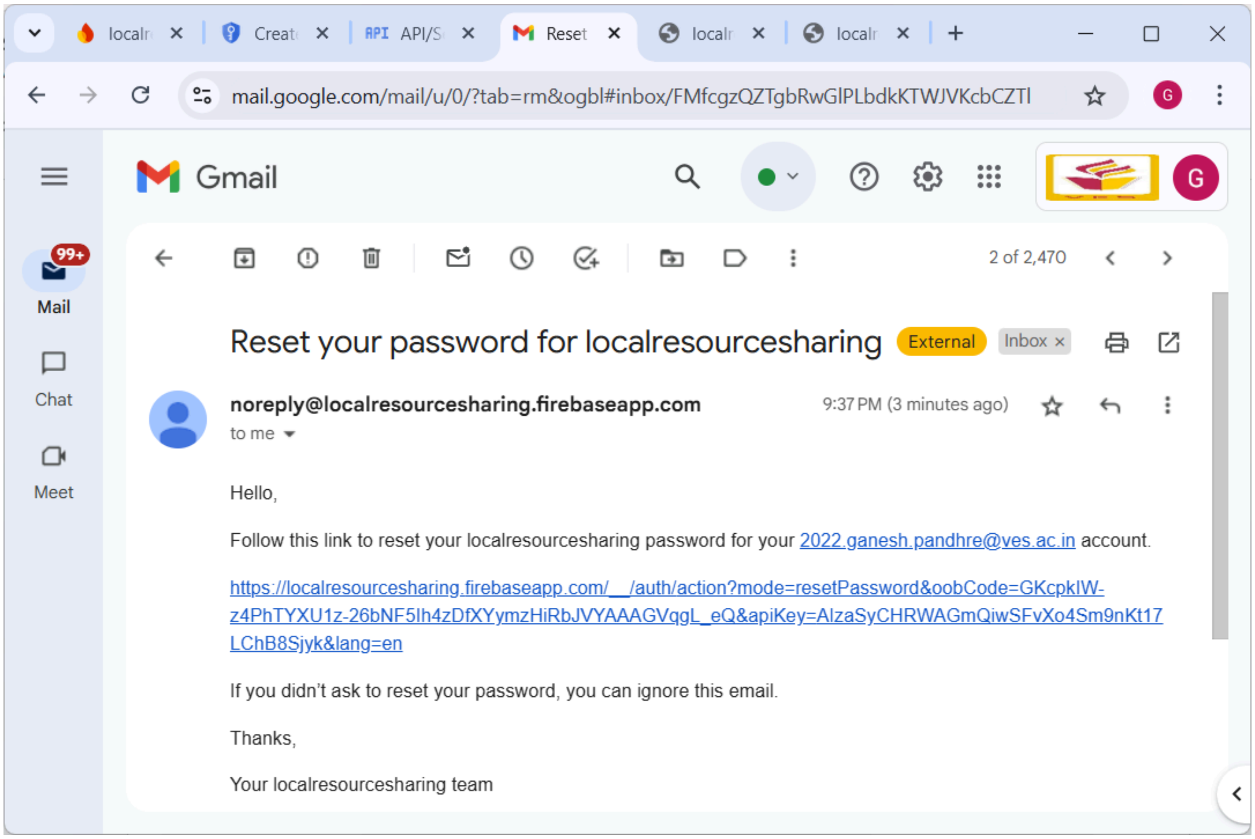
You can now sign in with your new account



Forgot Your Password?

Enter your email and we will send you a link to reset your password.

Email: 2022.ganesh.pandhre@ves.ac.in



Reset your password for localresourcesharing

noreply@localresourcesharing.firebaseapp.com 9:37 PM (3 minutes ago)

to me

Hello,

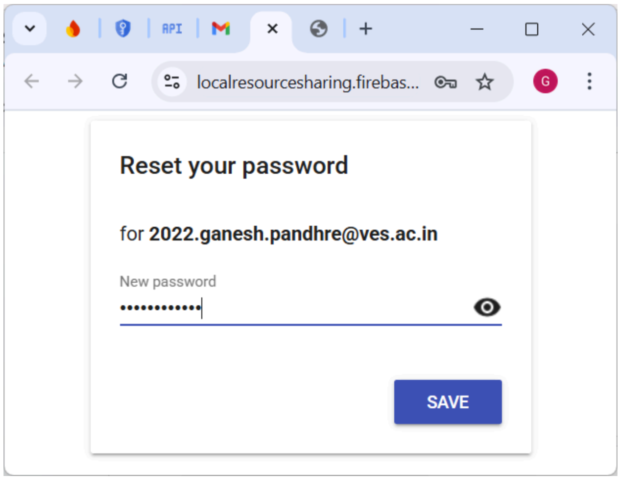
Follow this link to reset your localresourcesharing password for your 2022.ganesh.pandhre@ves.ac.in account.

https://localresourcesharing.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=GKcpklW-z4PhTYXU1z-26bNF5lh4zDfXYymzHiRbJVYAAAGVqgL_eQ&apiKey=AIzaSyCHRWAGmQiwSFvXo4Sm9nKt17LChB8Sjyk&lang=en

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

Thanks,

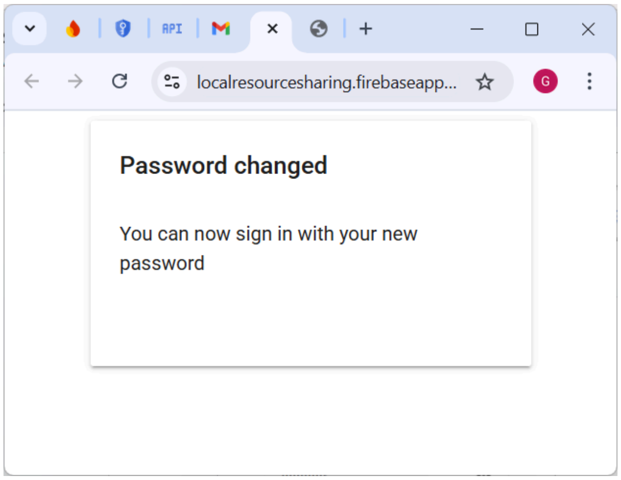
Your localresourcesharing team



Reset your password

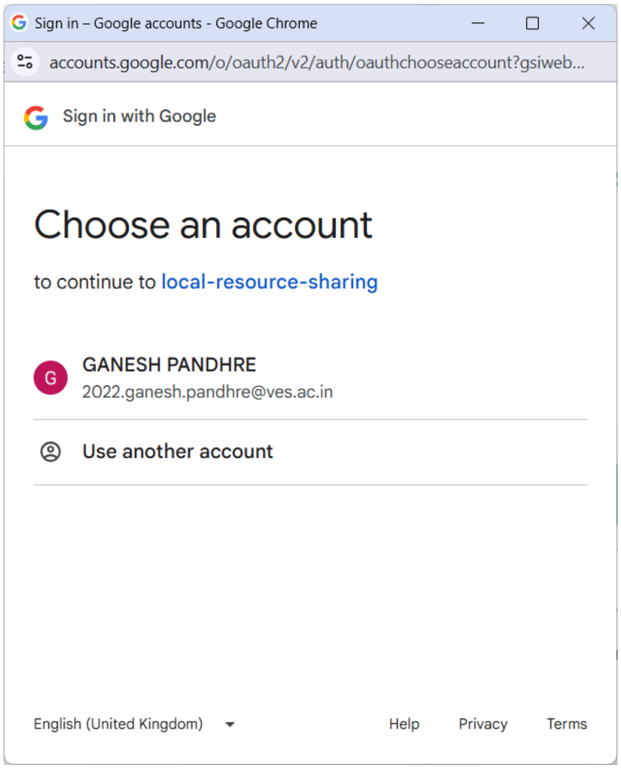
for **2022.ganesh.pandhre@ves.ac.in**

New password
••••••••••••



Password changed

You can now sign in with your new password

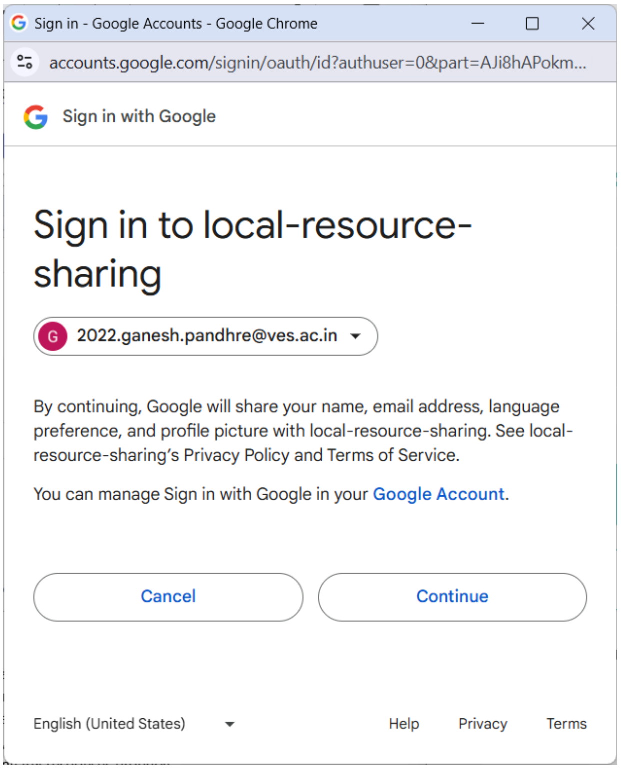

Sign in – Google accounts - Google Chrome
accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?gsiweb...
Sign in with Google
English (United Kingdom)
Help
Privacy
Terms

Choose an account

to continue to local-resource-sharing

GANESH PANDHRE
2022.ganesh.pandhre@ves.ac.in

Use another account

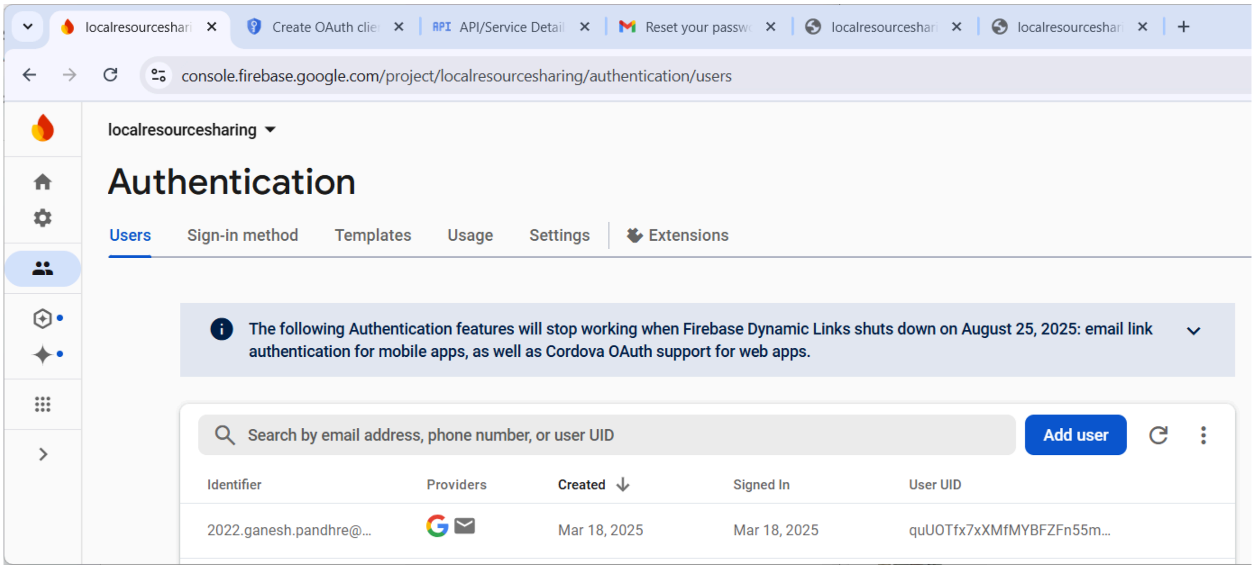

Sign in - Google Accounts - Google Chrome
accounts.google.com/signin/oauth/id?authuser=0&part=AJi8hAPokm...
Sign in with Google
Cancel
Continue
English (United States)
Help
Privacy
Terms

Sign in to local-resource-sharing

2022.ganesh.pandhre@ves.ac.in

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

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



Authentication

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

Identifier	Providers	Created	Signed In	User UID
2022.ganesh.pandhre@...	G ✉	Mar 18, 2025	Mar 18, 2025	quUOTfx7xXMfMYBFZFn55m...

Conclusion

In this experiment, we successfully implemented Firebase Authentication in the Local Resource Sharing app, enabling user signup, login, password reset, and Google authentication with email verification. We encountered issues like Firebase email verification delays, Google Sign-In Client ID errors, and incorrect return types in authentication functions, which we resolved by correctly configuring Firebase settings, updating authentication logic, and handling API responses properly.