**THỰC TẬP HÈ**

***FLUTTER DEVELOPER***

**Task: Đăng nhập tích hợp các mạng xã hội**

**Họ - Tên:** Đinh Nguyễn Tấn Đạt

**Mã số sinh viên:** 3118410067

**Số điện thoại:** 0708391537

**MỤC LỤC**

[**I. USER – PASSWORD:** 9](#_Toc107401173)

[**Bước 1: Cấp quyền cho Firebase:** 9](#_Toc107401174)

[**Bước 2: Thực hiện code:** 9](#_Toc107401175)

[**II. GOOGLE:** 11](#_Toc107401176)

[**Bước 1: Cấp quyền cho Firebase:** 11](#_Toc107401177)

[**Bước 2: Cài đặt thư viện:** 12](#_Toc107401178)

[**Bước 3: Thực hiện code:** 12](#_Toc107401179)

[**III. FACEBOOK:** 13](#_Toc107401180)

[**Bước 1: Tạo môi trường phát triển cho Facebook:** 13](#_Toc107401181)

[**Bước 2: Cấp quyền cho Firebase:** 14](#_Toc107401182)

[**Bước 3: Cài đặt thư viện:** 15](#_Toc107401183)

[**Bước 4: Thực hiện code:** 16](#_Toc107401184)

[**IV. APPLE:** 17](#_Toc107401185)

[**Bước 1: Tạo môi trường phát triển cho Apple:** 17](#_Toc107401186)

[**Bước 2: Cấp quyền cho Firebase:** 17](#_Toc107401187)

[**Bước 3: Cài đặt thư viện:** 18](#_Toc107401188)

[**Bước 4:Thực hiện code:** 18](#_Toc107401189)

[**V. MICROSOFT:** 19](#_Toc107401190)

[**Bước 1: Tạo môi trường phát triển cho Microsoft:** 19](#_Toc107401191)

[**Bước 2: Cấp quyền cho Firebase:** 23](#_Toc107401192)

[**Bước 3: Cài đặt thư viện:** 23](#_Toc107401193)

[**Bước 4: Thực hiện code:** 23](#_Toc107401194)

[**VI. CASBIN** 26](#_Toc107401195)

[**1: Khái niệm, cách thức hoạt động:** 26](#_Toc107401196)

[**2: Thực hiện code:** 27](#_Toc107401197)

[**VII. JAVA CAS** 28](#_Toc107401198)

***Note: Dùng firebase để thực hiện các chức năng đăng nhập. Ví dụ: User – password, Google, Facebook, Microsoft, Apple. Nên trước tiên cần cài đặt môi trường firebase.***

Cách: Sử dụng Firebase CLI:

**Bước 1: Cài đặt Firebase tool, tại cửa sổ Termial, gõ:**

npm install -g firebase-tools.

**Bước 2: Cài đặt thư viện các thư viện cho Firebase tại project**

firebase\_core:

<https://pub.dev/packages/firebase_core>

firebase\_auth:

<https://pub.dev/packages/firebase_auth>

cloud\_firestore:

<https://pub.dev/packages/cloud_firestore>

**Bước 3: Tiến hành đăng nhập firebase cho dự án, cửa sổ Terminal, gõ:**

firebase login (tiến hành đăng nhập tài khoản firebase)

**Bước 4: Cài đặt công cụ CLI. Tại Terminal của project, gõ:**

dart pub global activate flutterfire\_cli

**Bước 5: Cấu hình cho dự án hiện tại trên firebase:**

flutterfire configure

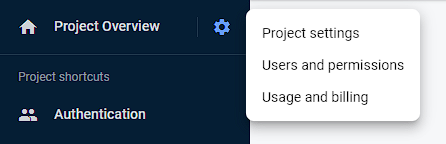
Chọn <create a new project>. Nhập tên cho Firebase project (tên này thay thế luôn cho project id trên firebase console). Nếu bị lỗi thì chứng tỏ tên vừa nhập đã được sử dụng, quay lại bước 5 để đặt lại tên khác.

Chọn platforms cho project (có thể hủy chọn platform mà mình không muốn phát triển bằng cách mũi tên lên xuống r nhấn space). Chọn xong nhấn Enter.

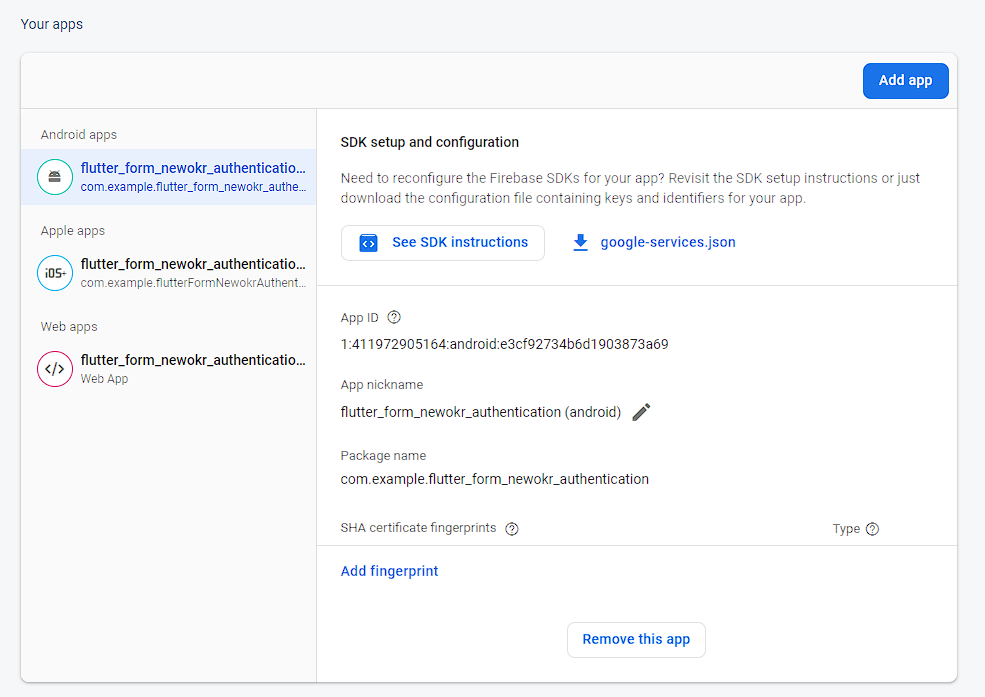
Việc cài đặt project lên firebase đã hoàn tất. Kiểm tra lại folder android (nếu có chọn platform andoid) sẽ thấy file google-service.json đã được thêm vào.

***\*\*\* Note: Để chạy được trên android \*\*\****

1 – Chọn Project setting tại project firebase vừa tạo:



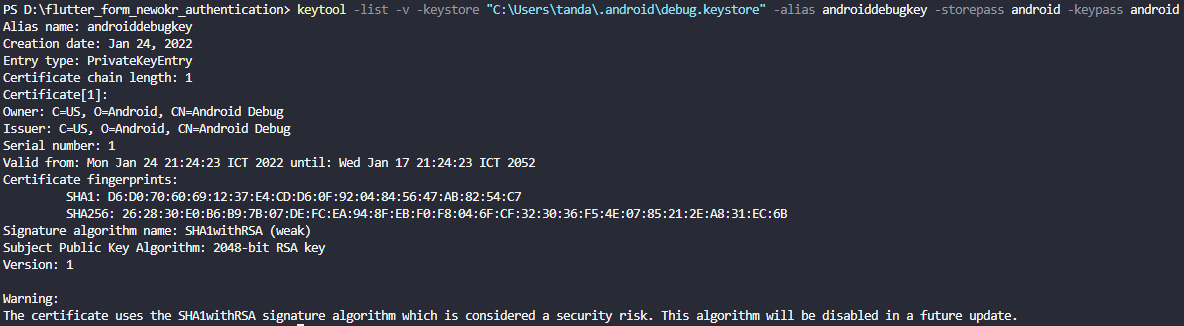
2 – Ngay phần Your apps -> Android apps. Cung cấp SHA của máy:



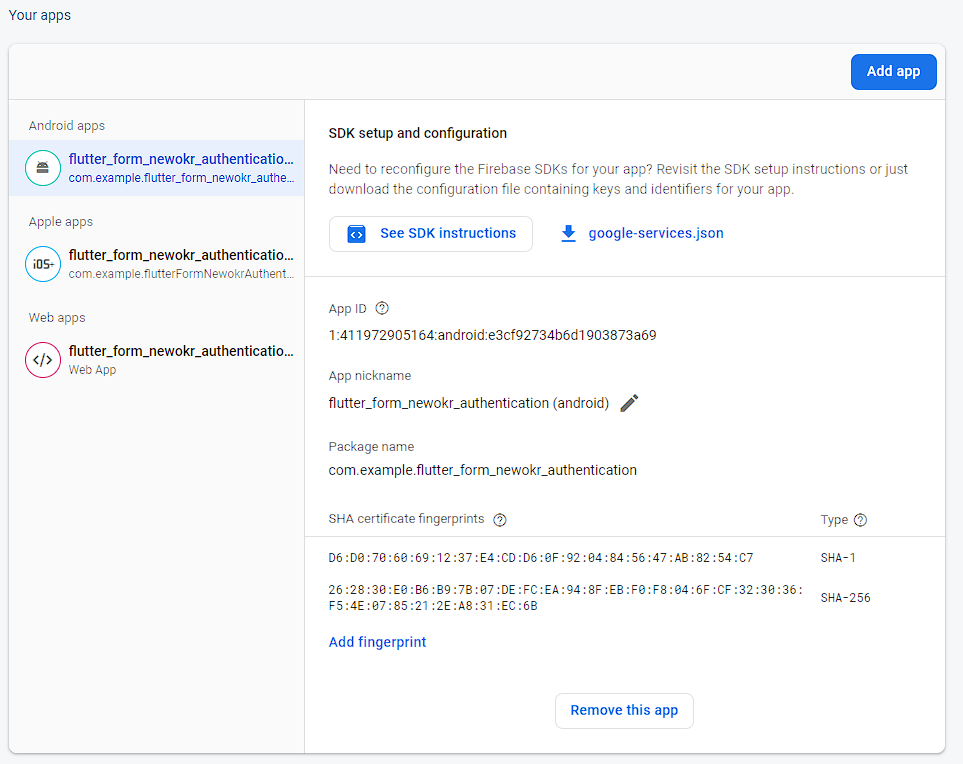
Để lấy được SHA của máy. Làm như sau:

Tại terminal của project code đang thực hiện, gõ lệnh:

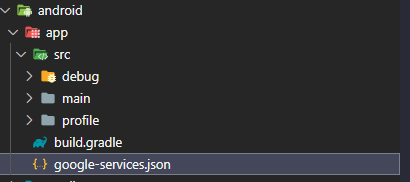
keytool -list -v -keystore "C:\Users\yourusernamehere\.android\debug.keystore" -alias androiddebugkey -storepass android -keypass android



Sau khi có key SHA rồi thì chọn “Add fingerprint”, nhập mã vào:



3 – Download lại file google-services.json thay thế file cũ trong foulder android/app.



4 – Chọn build.gradle tại foulder android/app:

defautlConfig {

minSdkVersion 21

}

***\*\*\* Note: Để login được khi sử dụng http-server \*\*\****

1. Cài gói http-server bằng lệnh npm i -g http-server
2. Chạy “flutter build web” tại file project
3. Cài đặt mkcert
4. Cài đặt scoop (Nếu chưa có) bằng lệnh tại iwr -useb get.scoop.sh | iex PowerShell. Nếu gặp vấn đề về chính sách thực thi thì chạy lệnh Set-ExecutionPolicy RemoteSigned -scope CurrentUser, sau đó chạy lại lệnh trên
5. Chạy lệnh bên dưới tại PowerShell

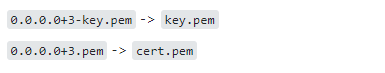




1. Vào file …/build/web chạy lệnh tại Cmd

**mkcert 0.0.0.0 localhost 127.0.0.1 ::1**

Thay đổi tên 2 file vừa khởi tạo bằng lệnh trên

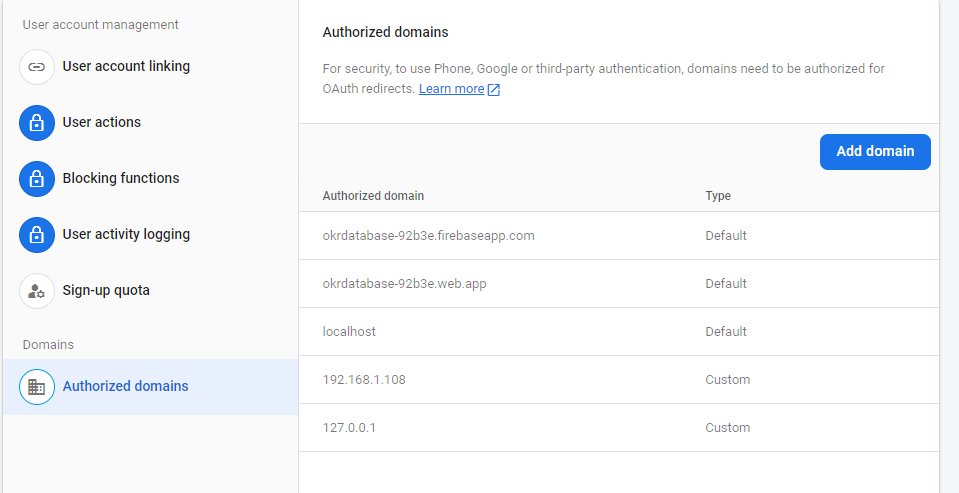


Sau đó chạy lệnh



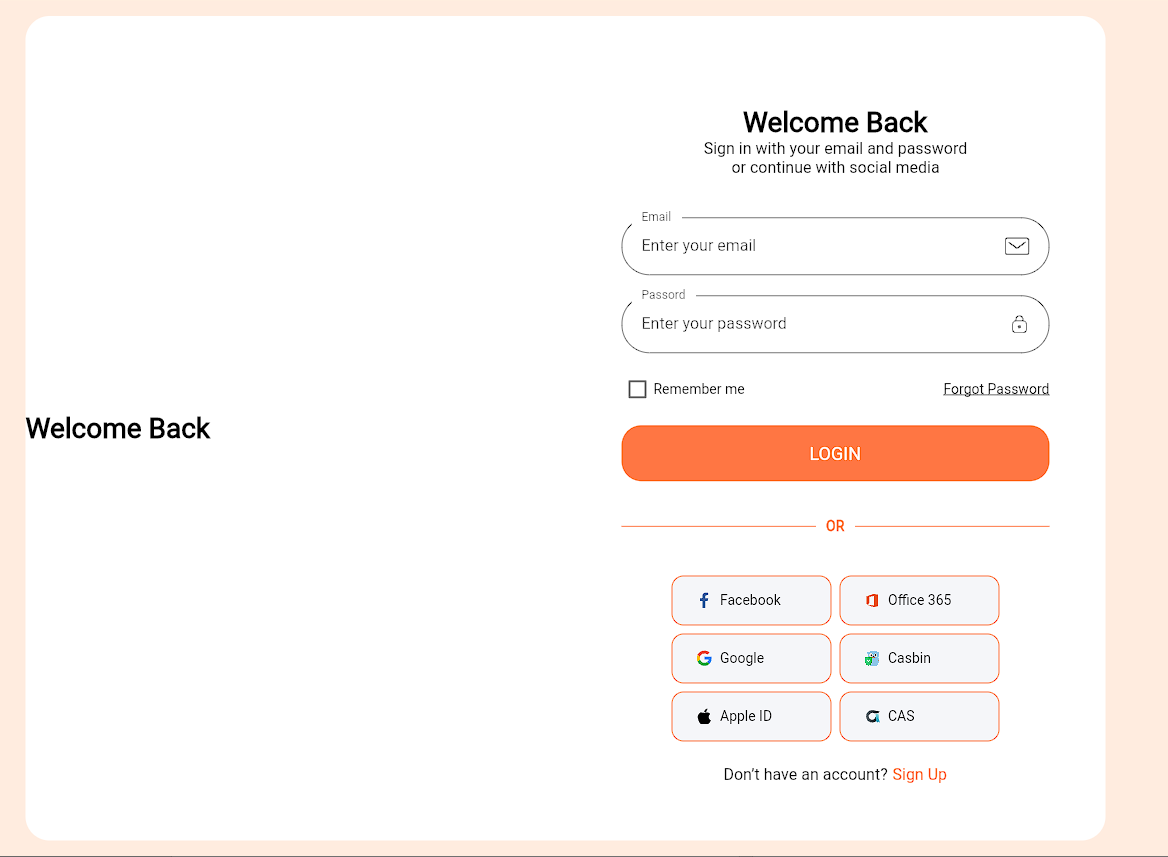
1. Add domain 127.0.0.1 vào Authentication của firebase

Firebase->Authentication->Settings->Authorized domains->Add domain-> Nhập 127.0.0.1->Add

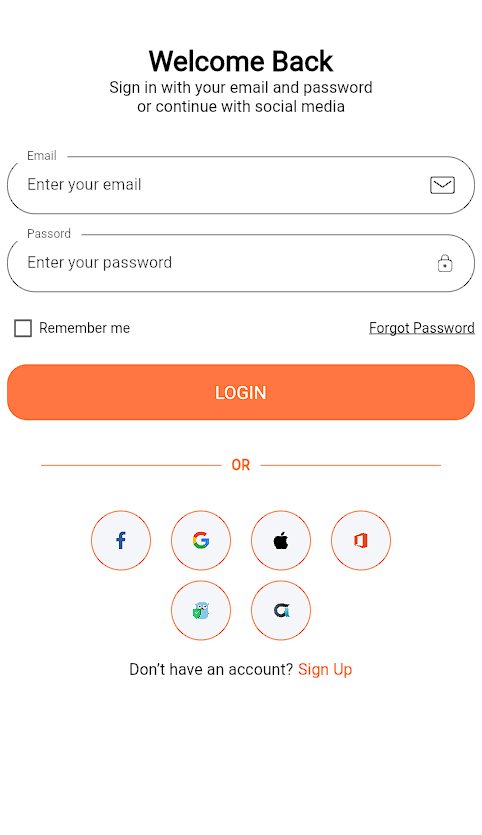


***Tạo màn hình Đăng nhập, Đăng ký:***

**Web:**



**Mobile:**



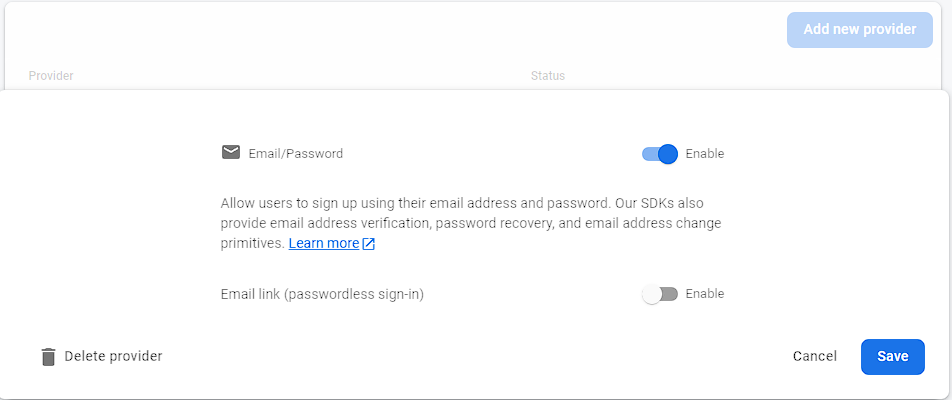
# **I. USER – PASSWORD:**

## **Bước 1: Cấp quyền cho Firebase:**

Vào firebase console (<https://console.firebase.google.com/> ), chọn project vừa tạo thành công bên trên.

Chọn “Authentication” -> Sign-in method -> Add new provider (tại Sign-in provider).

Chọn “Email/Password”. Nhấn Enable -> Save.



## **Bước 2: Thực hiện code:**

Đăng nhập bằng Email/Password:

final FirebaseAuth \_auth;

// EMAIL LOGIN

  Future<void> loginWithEmail({

    required String *email*,

    required String *password*,

    required BuildContext *context*,

  }) async {

    try {

      await \_auth.signInWithEmailAndPassword(

*email*: *email*,

*password*: *password*,

      );

    } on FirebaseAuthException catch (e) {

      showSnackBar(*context*, e.message!);

    }

  }

Đăng ký bằng Email/Password:

final FirebaseAuth \_auth;

// EMAIL SIGN UP

  Future<void> signUpWithEmail(

      {required String *email*,

      required String *password*,

      required BuildContext *context*}) async {

    try {

      await \_auth.createUserWithEmailAndPassword(

*email*: *email*,

*password*: *password*,

      );

      await sendEmailVerification(*context*);

      await \_auth.signOut();

    } on FirebaseAuthException catch (e) {

      if (e.code == 'weak-password') {

        print('The password provided is too weak.');

      } else if (e.code == 'email-already-in-use') {

        print('The account already exists for that email.');

      }

      showSnackBar(

*context*, e.message!);

    }

  }

Reset password khi quên mật khẩu:

final FirebaseAuth \_auth;

// RESET PASSWORD

  Future<void> resetPassword(

      {required String *email*, required BuildContext *context*}) async {

    try {

      await FirebaseAuth.instance.sendPasswordResetEmail(*email*: *email*);

    } on FirebaseAuthException catch (e) {

      showSnackBar(*context*, e.message!);

    }

  }

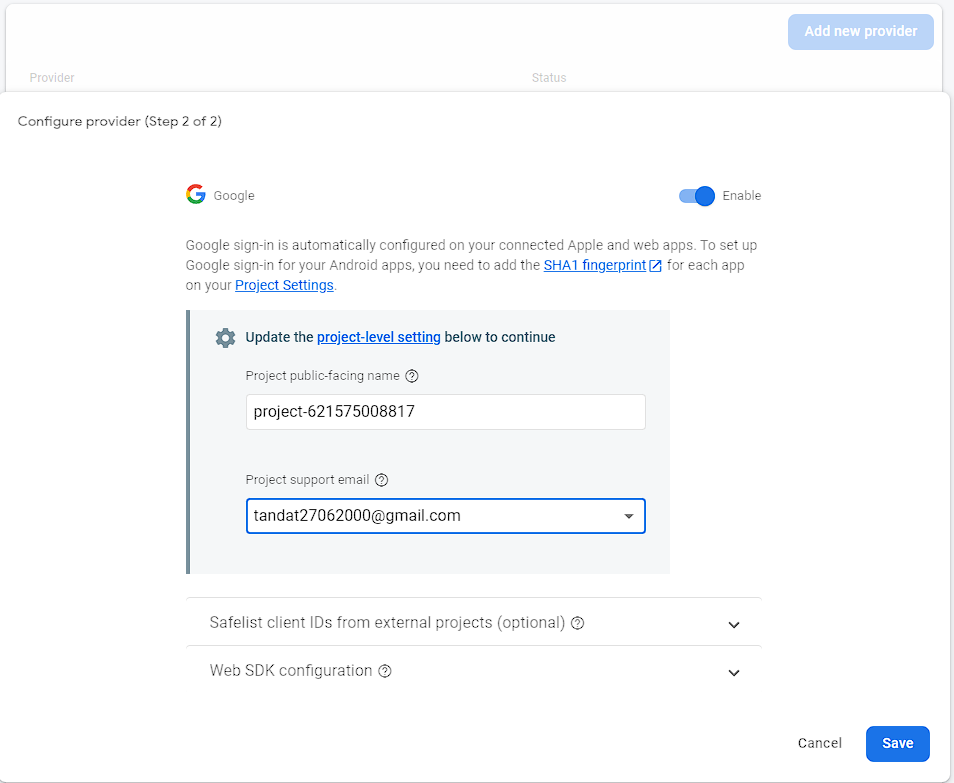
# **II. GOOGLE:**

## **Bước 1: Cấp quyền cho Firebase:**

Vào firebase console (<https://console.firebase.google.com/> ), chọn project vừa tạo thành công bên trên.

Chọn “Authentication” -> Sign-in method -> Add new provider (tại Sign-in provider).

Chọn “Google”. Nhấn Enable -> Chọn email -> Save.



## **Bước 2: Cài đặt thư viện:**

google\_sign\_in:

<https://pub.dev/packages/google_sign_in>

## **Bước 3: Thực hiện code:**

// GOOGLE SIGN IN

  Future<void> signInWithGoogle(BuildContext *context*) async {

    try {

      if (kIsWeb) {

        GoogleAuthProvider googleProvider = GoogleAuthProvider();

        googleProvider

            .addScope('https://www.googleapis.com/auth/contacts.readonly');

        googleProvider.setCustomParameters({'login\_hint': 'user@example.com'});

        UserCredential userCredential =

            await \_auth.signInWithPopup(googleProvider);

        if (userCredential.user != null) {

          if (userCredential.additionalUserInfo!.isNewUser) {

            FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

          }

        }

      } else {

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

        final GoogleSignInAuthentication? googleAuth =

          await googleUser?.authentication;

        final credential = GoogleAuthProvider.credential(

*accessToken*: googleAuth?.accessToken, *idToken*: googleAuth?.idToken,

        );

        UserCredential userCredential =

            await \_auth.signInWithCredential(credential);

        connect = "google";

        if (userCredential.user != null) {

          if (userCredential.additionalUserInfo!.isNewUser) {

            FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

          }

        }

      }

    } on FirebaseAuthException catch (e) {

      showSnackBar(*context*, e.message!);

    } on PlatformException catch (e) {

      showSnackBar(*context*, e.message!);

    } catch (e) {

      print(e);

    }

  }

# **III. FACEBOOK:**

## **Bước 1: Tạo môi trường phát triển cho Facebook:**

Truy cập: <https://developers.facebook.com/apps/>

1 – Chọn Create apps

2 – Chọn Consumers -> Next

3 – Nhập Display name (Business account chọn No Business Mananger accounts selected yet) -> Create apps

4 – Chọn Login with Facebook.

5 – Chọn Android (nếu chọn IOS thì cũng làm tương tự).

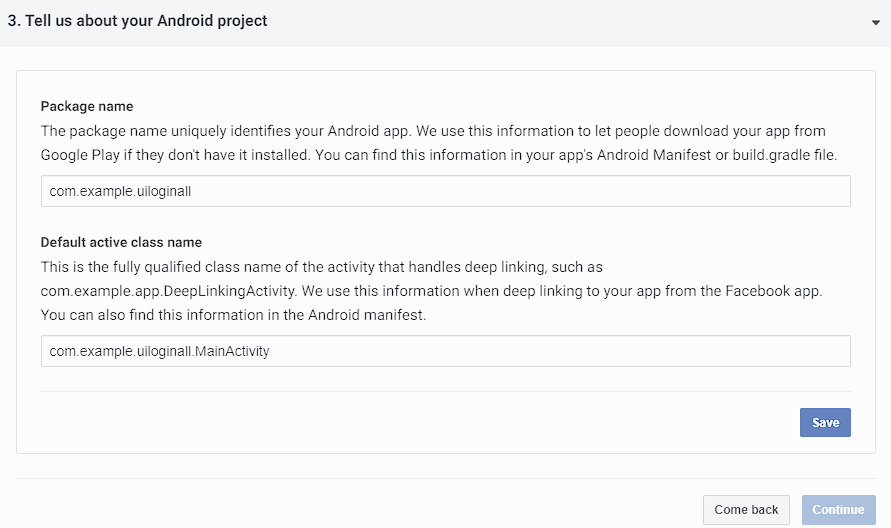
6 – Bỏ qua Step 1, 2. Step 3 nhập App ID của project đang thực hiện:

Lấy android/app/build.gradle:

Ví dụ:



Thì trong Step 3 nhập -> Save -> Continue



7 – Lấy Key Hash theo hướng dẫn tại Step 4 -> Save -> Next

8 – Enable single sign-on -> Save -> Next

9 – Tại Step 6 làm theo hướng dẫn.

10 – Xong Step 6 thì cứ Next cho tới hết. Cài đặt xong môi trường cho Facebook.

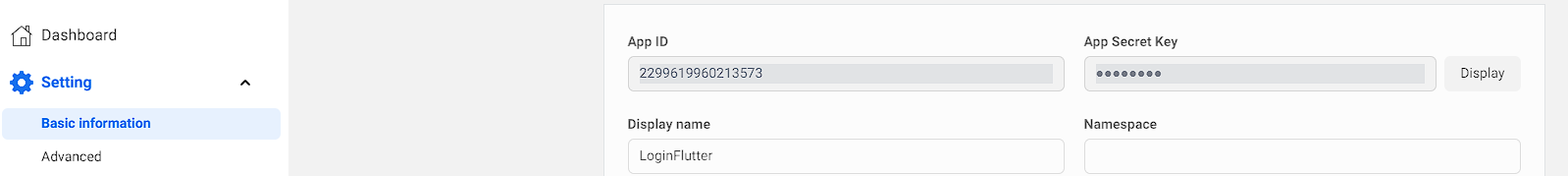
## **Bước 2: Cấp quyền cho Firebase:**

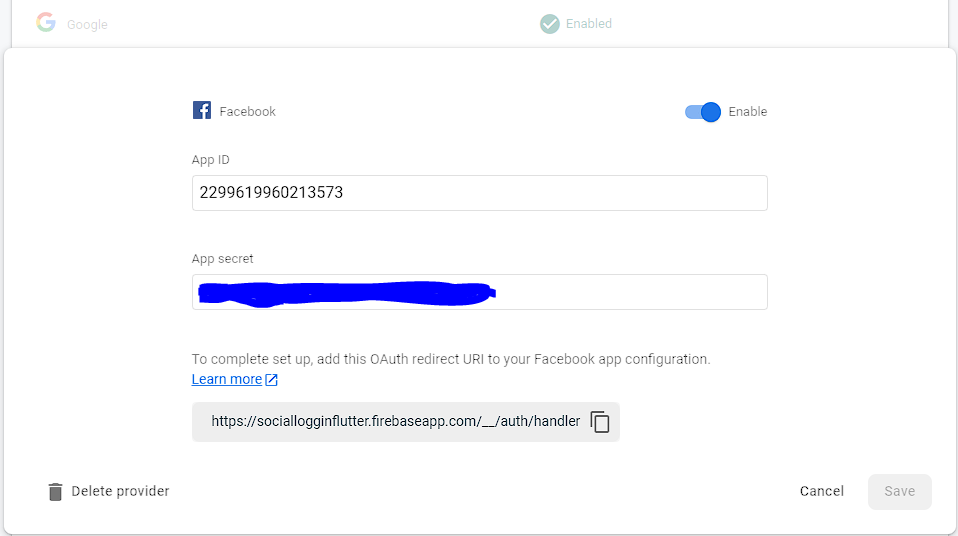
Vào firebase console (<https://console.firebase.google.com/> ), chọn project vừa tạo thành công bên trên.

Chọn “Authentication” -> Sign-in method -> Add new provider (tại Sign-in provider).

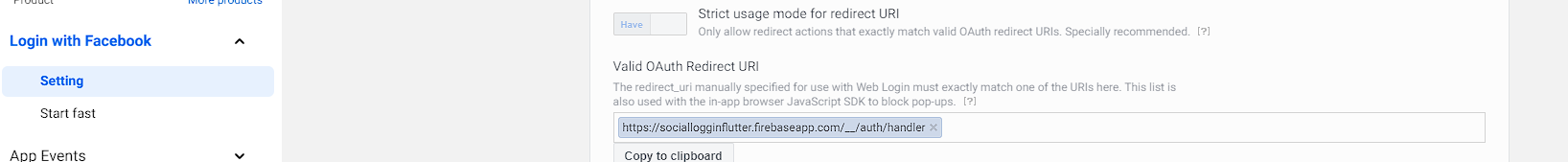
Chọn “Facebook”. Nhấn Enable -> Nhập App ID và App secret (Lấy ở <https://developers.facebook.com/apps/>) -> Copy đường link của firebase -> Save.

Lấy App ID và App secret:





Nhập đường link vừa Copy của firebase -> Save changes:



Hoàn thành xong liên kết giữa Firebase và Facebook.

## **Bước 3: Cài đặt thư viện:**

flutter\_facebook\_auth:

<https://pub.dev/packages/flutter_facebook_auth>

## **Bước 4: Thực hiện code:**

final FirebaseAuth \_auth;

// FACEBOOK SIGN IN

  Future<void> signInWithFacebook(BuildContext *context*) async {

    try {

      if (kIsWeb) {

        FacebookAuthProvider facebookProvider = FacebookAuthProvider();

        facebookProvider.addScope('email');

        facebookProvider.addScope('public\_profile');

        facebookProvider.setCustomParameters({

          'display': 'popup',

        });

        UserCredential userCredential =

            await FirebaseAuth.instance.signInWithPopup(facebookProvider);

        if (userCredential.user != null) {

          if (userCredential.additionalUserInfo!.isNewUser) {

            FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

          }

        }

      } else {

        final LoginResult loginResult =

            await FacebookAuth.instance.login(*permissions*: [

          'public\_profile',

          'email',

        ]);

        if (loginResult.status == *LoginStatus*.success) {

          final OAuthCredential credential =

              FacebookAuthProvider.credential(loginResult.accessToken!.token);

          UserCredential userCredential =

              await \_auth.signInWithCredential(credential);

          connect = "facebook";

          if (userCredential.user != null) {

            if (userCredential.additionalUserInfo!.isNewUser) {

              FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

            }

          }

        }

      }

    } on FirebaseAuthException catch (e) {

      showSnackBar(*context*, e.message!); // Displaying the error message

    } on PlatformException catch (e) {

      showSnackBar(*context*, e.message!);

    }

  }

## **IV. APPLE:**

## **Bước 1: Tạo môi trường phát triển cho Apple:**

Truy cập: <https://developer.apple.com/account/resources>

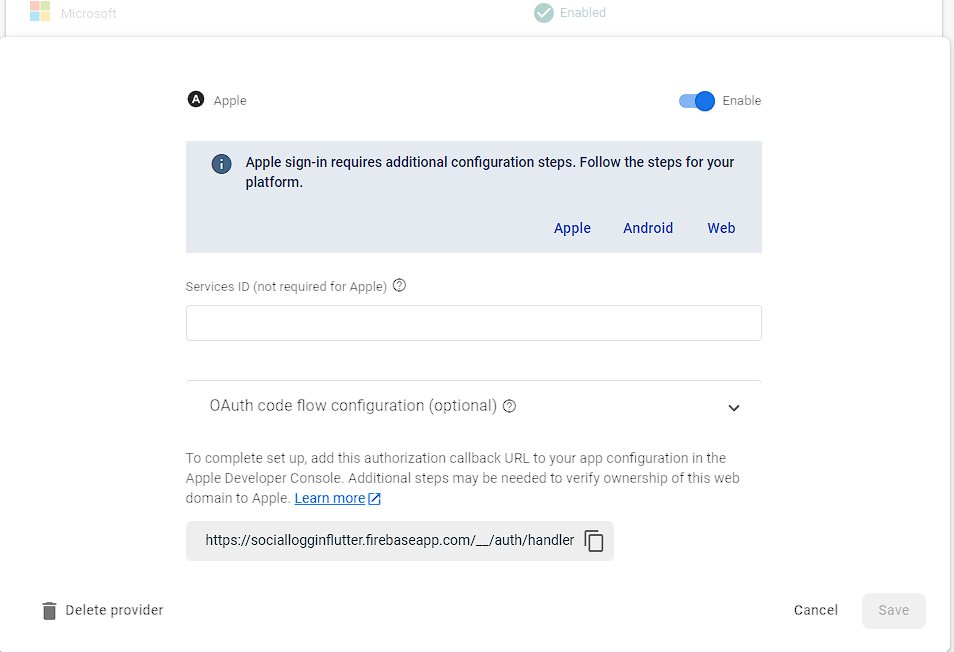
Phải có tài khoản Apple Developer

## **Bước 2: Cấp quyền cho Firebase:**

Vào firebase console (<https://console.firebase.google.com/> ), chọn project vừa tạo thành công bên trên.

Chọn “Authentication” -> Sign-in method -> Add new provider (tại Sign-in provider).

Chọn “Apple”. Nhấn Enable -> Save



## **Bước 3: Cài đặt thư viện:**

sign\_in\_with\_apple:

<https://pub.dev/packages/sign_in_with_apple>

## **Bước 4:Thực hiện code:**

//APPLE SIGN IN

  Future<void> signInWithAppleID(BuildContext *context*) async {

    if (kIsWeb) {

      final provider = OAuthProvider("apple.com")

        ..addScope('email')

        ..addScope('name');

      UserCredential userCredential =

          await FirebaseAuth.instance.signInWithPopup(provider);

      if (userCredential.user != null) {

        if (userCredential.additionalUserInfo!.isNewUser) {

          FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

        }

      }

    } else {

      final rawNonce = generateNonce();

      final nonce = sha256ofString(rawNonce);

      final appleCredential = await SignInWithApple.getAppleIDCredential(

*scopes*: [

*AppleIDAuthorizationScopes*.email,

*AppleIDAuthorizationScopes*.fullName,

        ],

*nonce*: nonce,

      );

      final oauthCredential = OAuthProvider("apple.com").credential(

*idToken*: appleCredential.identityToken,

*rawNonce*: rawNonce,

      );

      UserCredential userCredential =

          await FirebaseAuth.instance.signInWithCredential(oauthCredential);

      if (userCredential.user != null) {

        if (userCredential.additionalUserInfo!.isNewUser) {

          FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));}}}}

# **V. MICROSOFT:**

## **Bước 1: Tạo môi trường phát triển cho Microsoft:**

1 – Truy cập <https://azure.microsoft.com/en-us/services/active-directory/external-identities/b2c/#overview> để đăng ký tài khoản free.

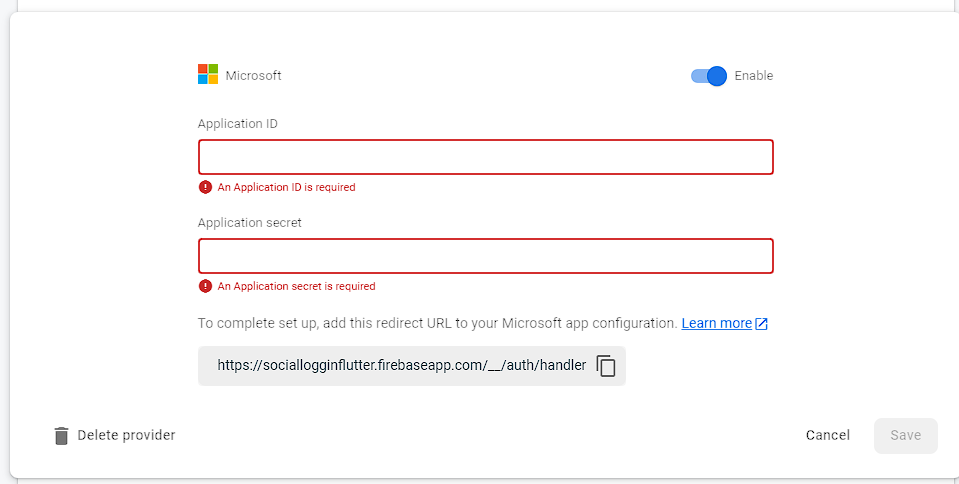
2 – Truy cập <https://portal.azure.com/> để thực hiện cài đặt môi trường cho Microsoft.

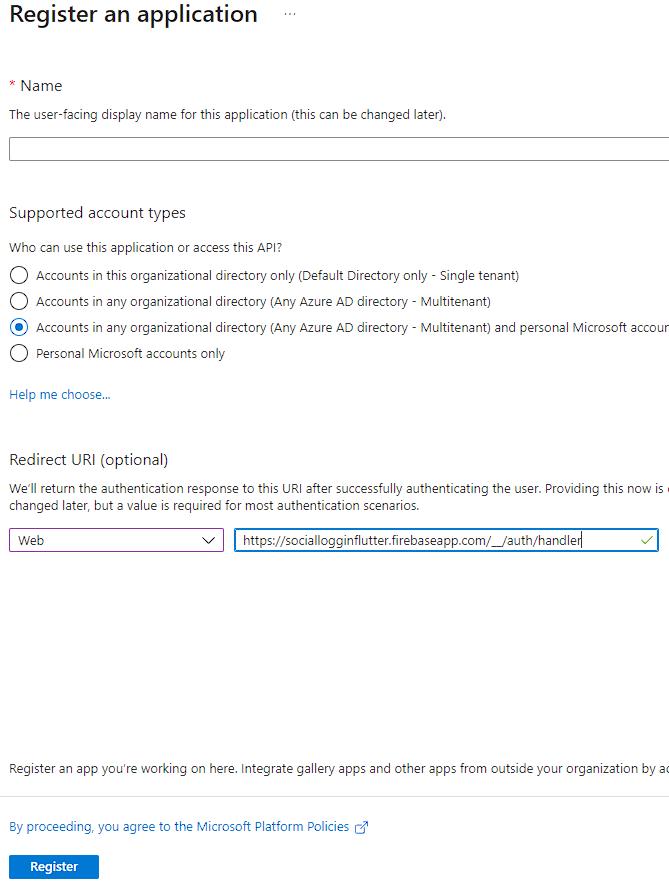
3 – Chọn Azure Acitve Directory -> App registrations -> New registrations:

4 – Chọn Accounts in any ….. and personal Microsoft accounts ….

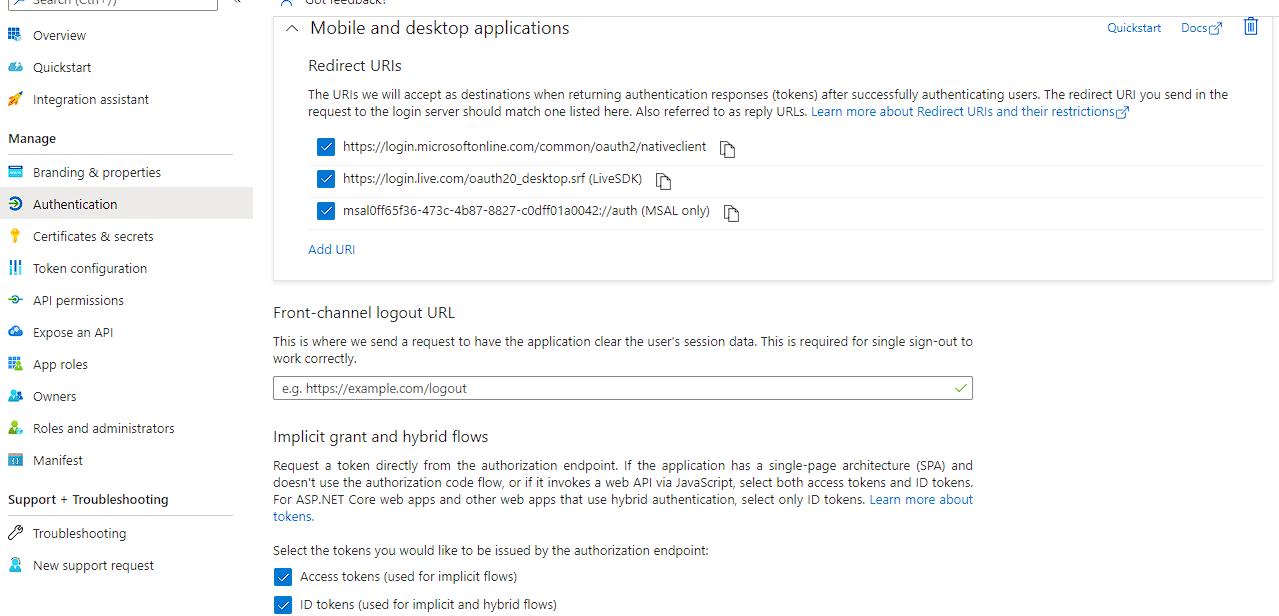
5 – Vào firebase console kích hoạt Sign-in method, chọn Microsoft. Lấy link firebase.

Quay lại Redirect URI chọn Web, nhập đường dẫn firebase vào -> Register



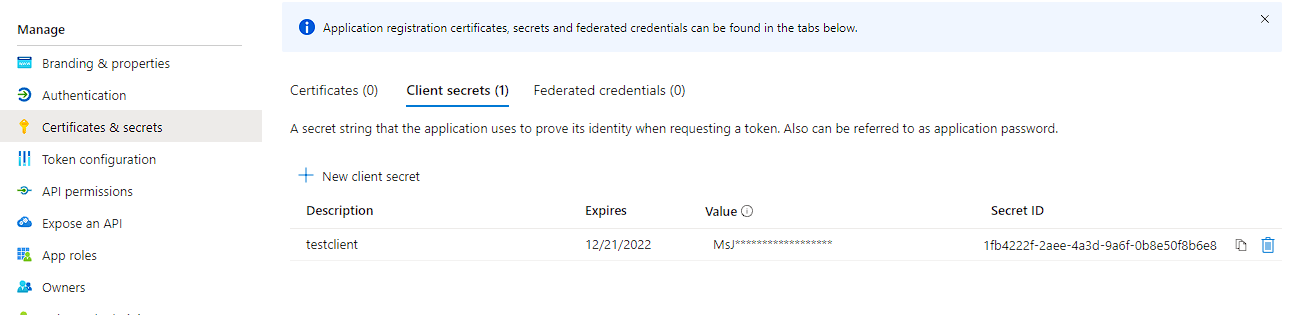


6 – Chọn App registration vừa tạo, vào Authentication, chọn “Add a platform” -> “Mobile and desktop applications”. Chọn access tokens với id tokens -> Save

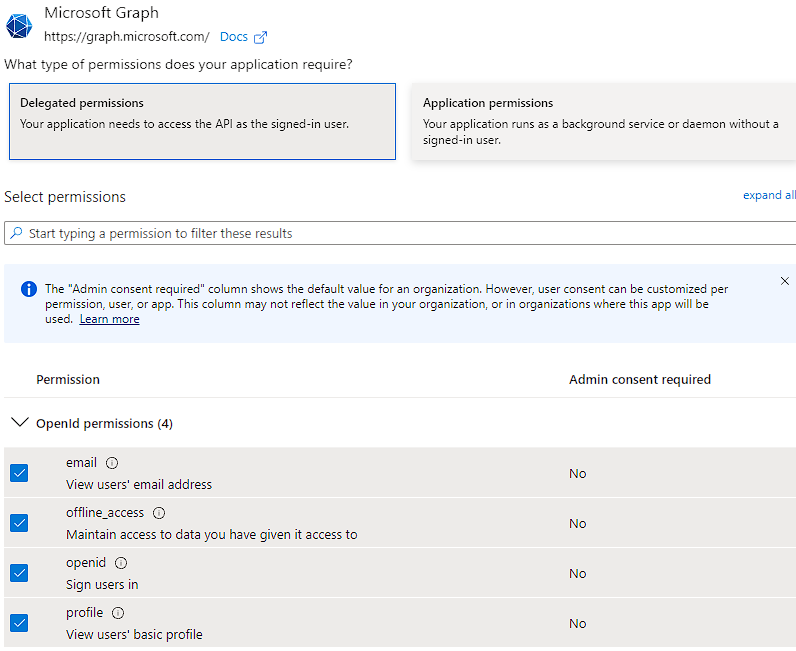


7 – Vào Certificates & secrets. Tạo Client secrets.

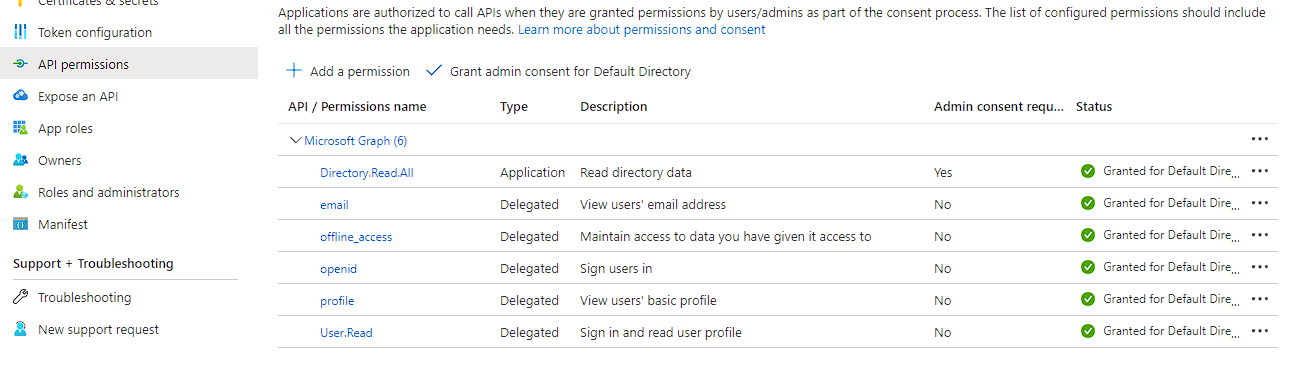
Note: Lưu lại Value.



8 – Vào API permissions -> “Add a permission” -> Microsoft Graph -> Delegated permissions. Chọn các permissions -> Add permisstions.

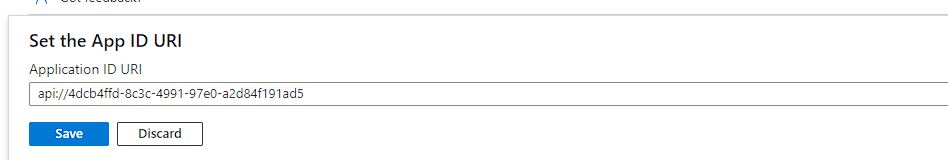


Chọn Grant admin consent for Default Directory -> Yes.



9 – Vào Expose an API. Dòng đầu có “Application ID URI”. Chọn Set -> Save.





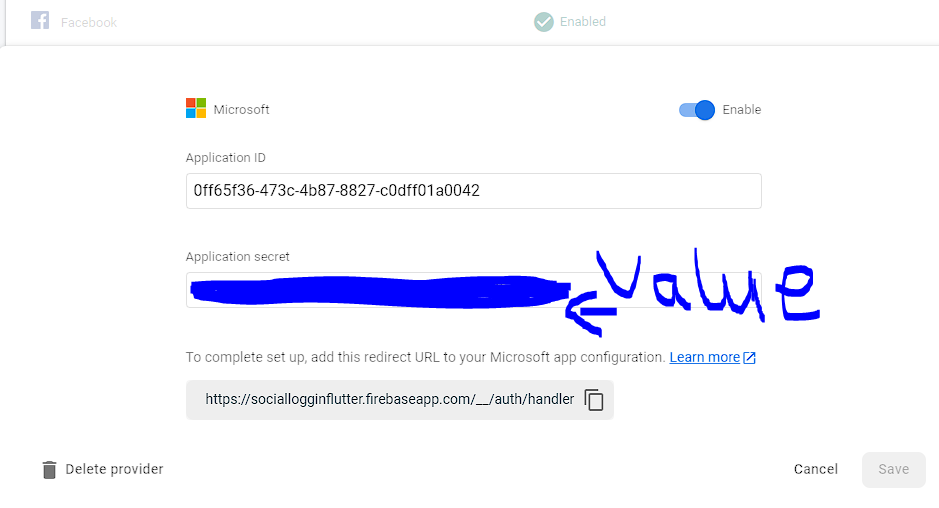
Cài đặt xong môi trường cho Microsoft

## **Bước 2: Cấp quyền cho Firebase:**

Vào firebase console (<https://console.firebase.google.com/> ), chọn project vừa tạo thành công bên trên.

Chọn “Authentication” -> Sign-in method -> Add new provider (tại Sign-in provider).

Chọn “Microsoft”. Nhấn Enable -> Nhập App ID và App secret (Lấy ở <https://portal.azure.com/> ) -> Save



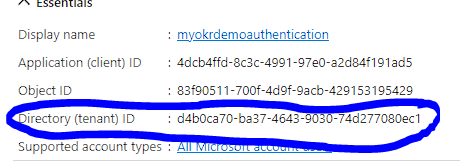
## **Bước 3: Cài đặt thư viện:**

firebase\_auth\_oauth:

<https://pub.dev/packages/firebase_auth_oauth>

## **Bước 4: Thực hiện code:**

(Thay thế code trong ‘tenant’:’…’ bằng tenant ID của bạn)



final FirebaseAuth \_auth;

//MICROSOFT SIGN IN

  Future<void> signInWithMicrosoft(BuildContext *context*) async {

    try {

      if (kIsWeb) {

        final provider = OAuthProvider("microsoft.com");

        provider.addScope('email');

        provider.addScope('openid');

        provider.addScope('offline\_access');

        provider.addScope('profile');

        provider.setCustomParameters(

          {

            'tenant': 'd4b0ca70-ba37-4643-9030-74d277080ec1',

            'prompt': 'login',

          },

        );

        UserCredential userCredential =

            await FirebaseAuth.instance.signInWithPopup(provider);

        if (userCredential.user != null) {

          if (userCredential.additionalUserInfo!.isNewUser) {

            FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

          }

        }

      } else {

        OAuthCredential microsoftUser =

            await FirebaseAuthOAuth().signInOAuth("microsoft.com", [

          "email openid profile offline\_access"

        ], {

          'tenant': 'd4b0ca70-ba37-4643-9030-74d277080ec1',

          'prompt': 'login',

        });

        UserCredential userCredential =

            await \_auth.signInWithCredential(microsoftUser);

        if (userCredential.user != null) {

          if (userCredential.additionalUserInfo!.isNewUser) {

            FirestoreService(*uid*: userCredential.user!.uid).addUser(UserData(

*email*: userCredential.user!.email ?? "",

*uid*: userCredential.user!.uid,

*name*: userCredential.user!.displayName ?? ""));

          }

        }

      }

    } on FirebaseAuthException catch (e) {

      showSnackBar(*context*, e.message!);

    } on PlatformException catch (e) {

      showSnackBar(*context*, e.message!);

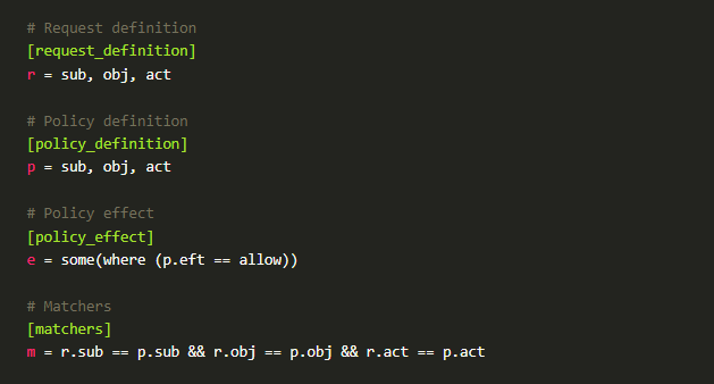
    }

  }

# **VI. CASBIN**

## **1: Khái niệm, cách thức hoạt động:**

Trong Casbin, mô hình kiểm tra quyền được cấu hình bằng một tập tin có dạng PERM (Policy, Effect, Request, Matchers).



**Request:** Là phần định nghĩa các thông số được gửi đến trong request yêu cầu phân quyền. Như hình minh hoạ, ta có thể thấy phần request có 3 thông số là sub (subject – đối tượng truy cập), obj (object – tài nguyên truy cập), act (action – hoạt động truy cập).

**Policy:** Định nghĩa các thông số có trong policy - các “dữ liệu đầu vào”, mô tả các người dùng, vai trò, quyền hạn… trong Casbin. Ví dụ một policy như “p, admin, data2, read” có nghĩa là: admin (sub) có quyền read (act) trên tài nguyên data2 (obj). Tập hợp tất cả policy sẽ giúp Casbin hiểu đối tượng nào được truy cập tài nguyên nào với hoạt động gì. Các dữ liệu này có thể được lưu trữ dưới dạng tập tin CSV, PostgresDB, SQL Server, hay MySQL database

Một policy thường có 4 phần: sub, obj, act, eft (effect – kết quả khi khớp policy, thường là allow hay deny). Trong trường hợp không có eft thì mặc định là allow

**Matcher:** Định nghĩa “quy luật khớp” (matching rule) giữa request và các policy. Như hình minh hoạ, ta có thể thấy “khớp” được định nghĩa là các thông số sub, obj và act trong request và policy đều giống nhau. Trong thực tế, ta có thể định nghĩa các quy luật phức tạp hơn bằng việc sử dụng Function trong Casbin như keyMatch cho URL, regexMatch, ipMatch cho dãy IP

Khi một policy được khớp, eft tương ứng sẽ trả về và xử lý tiếp với policy effect để đưa ra kết quả cuối cùng.

**Policy effect**: Định nghĩa quy luật để đưa ra kết quả cuối cùng sau khi tổng hợp tất cả eft của các policy khớp. Như hình minh họa, some (where(p.eft == allow)) nghĩa là chỉ cần một policy được match thì kết quả trả về sẽ là true.

Khi Casbin khởi tạo, danh sách các policy sẽ được nạp vào bộ nhớ. Khi nhận được request phân quyền, Casbin sẽ dựa vào tập tin cấu hình để kiểm tra policy theo matcher và policy effect để đưa ra kết quả cuối cùng. Hiện tại Casbin có sẵn rất nhiều tập tin cấu hình theo các mô hình phân quyền thông dụng mà bạn có thể tìm thấy tại đây.

## **2: Thực hiện code:**

# **VII. JAVA CAS**