

Lab

- Setting up Firebase Auth
- Setting up Google Sign In
- Setting up Image Picker
- Running the Chat App

Setting Up Firebase Auth

- Enable Email sign-in
- Enable Google auth
 - Needs [SHA-1 release fingerprint](#) for Android apps
 - Terminal: `keytool -list -v -alias androiddebugkey -keystore ~/.android/debug.keystore` (for Mac)
 - Paste SHA fingerprint in “Project Settings > Your apps” section
 - Replace “google-services.json” and “GoogleService-Info.plist” files in “android/app”, “ios/Runner”, and “macos/Runner” folders

Authentication

Users Sign-in method Templates Us

Sign-in providers

Get started with Firebase /
method

Native providers

✉ Email/Password

☎ Phone

👤 Anonymous

Additional providers

🌐 Google

📘 Facebook

🎮 Play Games

🎮 Game Center

Custom providers

🔒 OpenID Connect

🔒 SAML

Add new provider

Provider

Status

✉ 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

🗑 Delete provider

Cancel

Save

🌐 Google

🟢 Enabled

Configure provider (Step 2 of 2)



 Enable

Important: To enable Google sign-in for your Android apps, **you must provide the [SHA-1 release fingerprint](#)** for each app (go to [Project Settings](#) > *Your apps* section).

 Update the [project-level setting](#) below to continue

Public-facing name for project 

project-31678082436

Support email for project 

Not configured 

 Please select an email address

Safelist client IDs from external projects (optional) 



Web SDK configuration 



SHA-1 release fingerprint (Windows)

- Download Java x64 installer:
<https://www.oracle.com/tw/java/technologies/downloads/#jdk17-windows>
- Add `C:\Program Files\Java\jdk{version}\bin` to System Path
In cmd: `java -version` for checking Java successfully installed.
- In cmd:
`keytool -list -v -alias androiddebugkey -keystore "C:\Users\{your user name}\.android\debug.keystore"`
Default Password: `android`
- It should print your fingerprint like this: (This is just an example)
Certificate fingerprint: SHA1:
`DA:39:A3:EE:5E:6B:4B:0D:32:55:BF:EF:95:60:18:90:AF:D8:07:09`
- Go back to Firebase **Project Setting**.

SHA-1 release fingerprint (Mac)

- Download Java 17 x64 DMG Installer:
<https://www.oracle.com/tw/java/technologies/downloads/#jdk17-mac>
- In terminal:
- `mkdir ~/.android`
- `keytool -genkey -v -keystore ~/.android/debug.keystore -storepass android -alias androiddebugkey -keypass android -keyalg RSA -keysize 2048 -validity 10000`
- Enter your name and skip other questions
- `keytool -list -v -alias androiddebugkey -keystore ~/.android/debug.keystore`
Default Password: `android`
- It should print your fingerprint like this: (This is just an example)
Certificate fingerprint: SHA1:
`DA:39:A3:EE:5E:6B:4B:0D:32:55:BF:EF:95:60:18:90:AF:D8:07:09`
- Go back to Firebase **Project Setting**.

[Add app](#)


Android apps

 **flutter_app (android)**
chat.app

Apple apps

 **flutter_app (ios)**
chat.app

Web apps

 **flutter_app (web)**
Web App **flutter_app (windows)**
Web App

SDK setup and configuration

Need to reconfigure the Firebase SDKs for your app? Revisit the SDK setup instructions or just download the configuration file containing keys and identifiers for your app.


[See SDK instructions](#)[↓ google-services.json](#)

2. Download this

App ID 

1:31678082436:android:19e90c3a9e2e54771c41d1


App nickname

flutter_app (android) 

Package name

chat.app

1. Paste your SHA1

SHA certificate fingerprints Type 

fingerprint

[Add fingerprint](#)[Remove this app](#)

Your apps

Add app

Android apps



flutter_app (android)
chat.app

Apple apps



flutter_app (ios)
chat.app

Web apps



flutter_app (web)
Web App



flutter_app (windows)
Web App

SDK setup and configuration

Need to reconfigure the Firebase SDKs for your app? Revisit the SDK setup instructions or just download the configuration file containing keys and identifiers for your app.

[See SDK instructions](#)

[GoogleService-Info.plist](#)

Download this

App ID

1:31678082436:ios:b79b1e1ae1ac86331c41d1

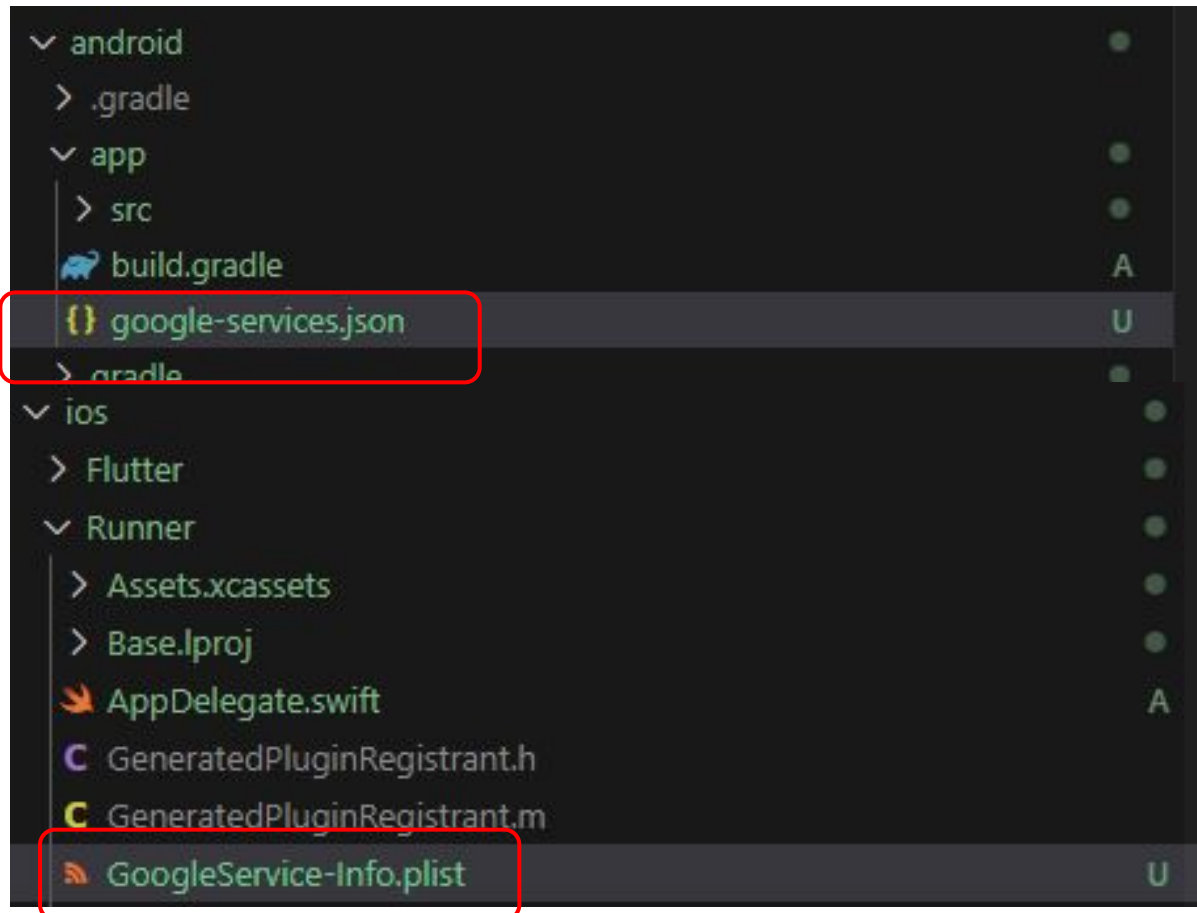
Encoded App ID

app-1-31678082436-ios-b79b1e1ae1ac86331c41d1

App nickname

flutter_app (ios)

If you don't have android or ios folder, you can run:
flutter create --platforms=ios .
flutter create --platforms=android .
flutter create --platforms=web .
(If error occurs, change your directory name by replacing - to _)



Lab

- Setting up Firebase Auth
- **Setting up Google Sign In**
- Setting up Image Picker
- Running the Chat App

google_sign_in Package

- Run `flutterfire configure`
- Android (See following page)
 - Filled out all required fields (if any) in [OAuth consent screen](#)
- iOS (See following page)
 - Follow [the instructions](#)
 - Add to “ios/Runner/Info.plist”:
 - `<key>GIDClientID</key>`
`<string>...</string>`
 - `<key>CFBundleURLTypes</key>`
`<array>...</array>`

google_sign_in install

- `$ flutter pub add google_sign_in`
(run in your project)

- [OAuth consent screen](#)

google sign in install

Select a resource

NEW PROJECT



DATALAB.CS.NTHU.EDU.TW ▼

- If you can't find your chat project, click all

🔍 Search projects and folders

RECENT

STARRED

ALL

Name

ID

✓ ☆ ⋮ [chat2](#) ?

chat2-cfa19

☆ ⋮ [SStodo](#) ?

youngss


☆ ⋮ [chat](#) ?


chat-5c9ca


📁 [datalab.cs.nthu.edu.tw](#) ?


1059381537114


- [OAuth consent screen](#)


API APIs & Services 

 Enabled APIs & services


 Library

 Credentials

 **OAuth consent screen**


 Page usage agreements

OAuth consent screen

project-31678082436  EDIT APP

Verification Status

Verification not required

Your consent screen is being shown, but your app has not been reviewed so your users may not see all of your information, and you will not be able to request certain OAuth scopes. [Learn more](#) 

Edit app registration

 OAuth consent screen —  Scopes —  Optional info —  **Summary**

Only needs to fill in email in step 1.

OAuth consent screen

[EDIT](#)

User type

External

see next page

```
<key>GIDClientID</key>  
<string>[YOUR IOS CLIENT ID]</string>
```

```
<key>CFBundleURLTypes</key>
```

```
<array>  
  <dict>  
    <key>CFBundleTypeRole</key>  
    <string>Editor</string>  
    <key>CFBundleURLSchemes</key>  
    <array>  
      <string>com.googleusercontent.apps.861823949799-vc35cprkp249096uujjn0wnmcvjppkn</string>  
    </array>  
  </dict>  
</array>
```

IOS app

```
GoogleService-Info.plist U x Info.plist A
ios > Runner > GoogleService-Info.plist
1 <?xml version="1.0" encoding="UTF-8"
2 <!DOCTYPE plist PUBLIC "-//Apple//D
3 <plist version="1.0">
4 <dict>
5   <key>CLIENT_ID</key>
6   <string>31678082436-ajey...
7   <key>REVERSED_CLIENT_ID</key>
8   <string>com.googleusercontent.a
9   <key>ANDROID_CLIENT_ID</key>
10  <string>31678082436-hcrsf0031tg
```

```
GoogleService-Info.plist U Info.plist M x
ios > Runner > Info.plist
You, 1 second ago | 1 author (You)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.a
3 <plist version="1.0">
4 <dict>
5   <key>GIDClientID</key>
6   <!-- Copied from GoogleService-Info.plist key CLIENT_ID -->
7   <string>[YOUR IOS CLIENT ID]</string>
8
9   <key>CFBundleURLTypes</key>
10  <array>
11    <dict>
12      <key>CFBundleTypeRole</key>
13      <string>Editor</string>
14      <key>CFBundleURLSchemes</key>
15      <array>
16        <!-- iOS: Replace this value: -->
17        <!-- Copied from GoogleService-Info.plist key REVERSED_CLIENT_ID -->
18        <string>com.googleusercontent.apps.861823949799-vo
19      </array>
20    </dict>
21  </array>
```

- the instructions
step 4 and step 6

Web

In web/index.html add:

```
<meta name="google-signin-client_id" content="YOUR CLIENT ID">
```

Get client id in OAuth page:

The image shows a composite of three screenshots illustrating the process of adding a Google Sign-in client ID to a Flutter web application.

Left Screenshot (Google Cloud Console): The 'APIs & Services' page is shown with the 'Credentials' tab selected. Under 'API Keys', three keys are listed: 'iOS key (auto created by Google Service)', 'Android key (auto created by Google Service)', and 'Browser key (auto created by Google Service)'. All three keys have a green checkmark icon.

Right Screenshot (VS Code): The 'index.html' file is open, showing the HTML structure. The following meta tags are present in the head section:

```
<meta charset="UTF-8">
<meta content="IE=Edge" http-equiv="X-UA-Compatible">
<meta name="description" content="A new Flutter project.">
<meta name="google-signin-client_id" content="824417574920-9apd...">
```

The last line, which includes the client ID, is highlighted with a red box.

Bottom Screenshot (OAuth 2.0 Client IDs Table): This table lists the client IDs created for the application:

Name	Creation date	Type	Client ID
Android client for chat.app (auto created by Google Service)	May 22, 2024	Android	824417574920-msgf...
iOS client for chat.app (auto created by Google Service)	May 22, 2024	iOS	824417574920-rm85...
Web client (auto created by Google Service)	May 22, 2024	Web application	824417574920-9apd...

The 'Web client' row is highlighted with a red box, indicating the client ID to be used in the HTML meta tag.

Web

If encounter problem, check error message.
You might see a link to google people API.
Just open and enable it.

```
Google Sign-in failed with error: ClientException: {  
  "error": {  
    "code": 403,  
    "message": "People API has not been used in project 824417574920 before or it is disabled. Enable it by visiting https://console.de  
velopers.google.com/apis/api/people.googleapis.com/overview?project=824417574920 then retry. If you enabled this API recently, wait a f  
ew minutes for the action to propagate to our systems and retry.",  
    "status": "PERMISSION_DENIED",  
    "details": [ ]  
  }  
}
```

<https://console.developers.google.com/apis/api/people.googleapis.com/overview?project=31678082436>

Web

If you can't link two sign-in method, click Web client
Modify URLs to your localhost port:

API

APIs & Services

Enabled APIs & services

Library

Credentials

OAuth consent screen

Page usage agreements

←

Client ID for Web application

The name of your OAuth 2.0 client. This name is only used to console and will not be shown to end users.

The domains of the URIs you add below will be your [OAuth consent screen](#) as [authorized domain](#)

Authorized JavaScript origins

For use with requests from a browser

URIs 1 *

http://localhost

URIs 2 *

http://localhost:61293

URIs 3 *

https://chat2-cfa19.firebaseio.com

+ ADD URI

OAuth 2.0 Client IDs

☐

Name

☐

[Android client for chat.app \(auto created by Google Service\)](#)

☐

[iOS client for chat.app \(auto created by Google Service\)](#)

☐

[Web client \(auto created by Google Service\)](#)

19

Lab

- Setting up Firebase Auth
- Setting up Google Sign In
- **Setting up Image Picker**
- Running the Chat App

Storage

Set up Cloud Storage

- 1 Secure rules for Cloud Storage
- 2 Set Cloud Storage location

After you define your data structure, you will need to write rules to secure your data.

[Learn more](#)

☐ Start in production mode

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

☒ Start in test mode

Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';

service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if
        request.time < timestamp.date(2024, 6, 21);
    }
  }
}
```

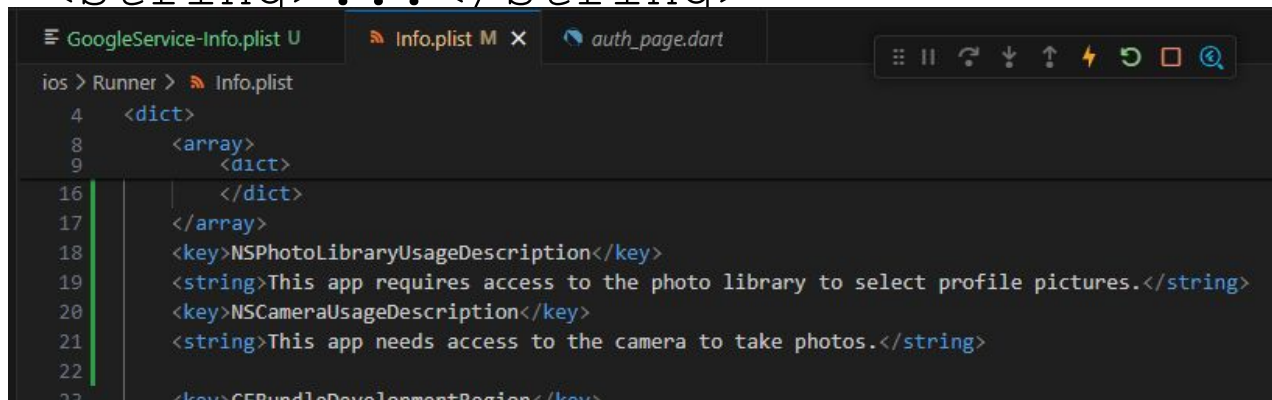
The default security rules for test mode allow anyone with your storage bucket reference to view, edit and delete all data in your storage bucket for the next 30 days

Cancel

Next

Image Picker

- \$ flutter pub add image_picker
- iOS: Add to “ios/Runner/Info.plist”:
 - `<key>NSPhotoLibraryUsageDescription</key>`
`<string>...</string>`
 - `<key>NSCameraUsageDescription</key>`
`<string>...</string>`



```
ios > Runner > Info.plist
4  <dict>
8      <array>
9          <dict>
16             </dict>
17         </array>
18         <key>NSPhotoLibraryUsageDescription</key>
19         <string>This app requires access to the photo library to select profile pictures.</string>
20         <key>NSCameraUsageDescription</key>
21         <string>This app needs access to the camera to take photos.</string>
22     </dict>
23     <key>CFBundleDevelopmentRegion</key>
```

- Follow the [installation guide](#) for more details

(Optional) MacOS

- `google_sign_in` package requires higher platform target than default
 - Set `platform :osx, '10.15'` in “macos/Podfile”
- `image_picker` package:
 - Add to “macos/Runner/*.entitlements”:
 - `<key>com.apple.security.files.user-selected.read-only</key>`
`<true/>`

Running Chat App

- Sign up using your email address
- Send some chat messages
- Log out, then log in with Google using same email address
 - Account linking will be triggered
- Check:
 - Image picker runs correctly, and selected file stored in Cloud Storage
 - User and Message docs created in Firestore
 - After account linking, your User doc should record two log-in methods