



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Faculty of Engineering, Built Environment and
Information Technology

Department of Computer Science
Faculty of Engineering, Built Environment & IT
University of Pretoria

COS301 - Software Engineering

Coviduous

Installation Manual

September 18, 2021

Team Name: CAPSlock

Team Members: * - indicates team leader

Name	Surname	Student Number
Njabulo*	Skosana*	u18089102*
Rudolf	van Graan	u16040865
Clementine	Mashile	u18139508
Peter	Okumbe	u18052640
Chaoane	Malakoane	u18202374

Contents

1	Introduction	3
2	System requirements	3
2.1	Artificial intelligence and Chatbot	3
2.2	Backend	3
2.3	Frontend	4
3	Download the source code	4
4	Installation	5
4.1	Artificial intelligence	5
4.2	Chatbot	6
4.3	Backend	7
4.4	Frontend	14
4.4.1	Run on a web browser	14
4.4.2	Run on an Android emulator	15

1 Introduction

Coviduous is an app that companies and their employees can use to facilitate office space bookings during the COVID-19 pandemic. The app is divided into two distinct subsystems, related to the two different types of customers mentioned before, as well as a third **Visitor** subsystem for temporary company visitors. The **User** subsystem is how employees will interact with the app. They have the option to view available office spaces, book an office space, view their current bookings, view announcements that are made by their companies, view notifications sent to them, and manage their COVID-related health documentation. The **Admin** subsystem is how employers will interact with the app. They have the option to manage floor plans, manage employee shifts, manage company-wide announcements, send notifications to employees, and view reports based on their company's statistics.

The following sections details how to download and install the app in its current state.

2 System requirements

The current Demo 4 version of the app currently has to be run using an Android emulator or web browser, with the backend running on your computer's localhost. The system has been fully tested on Windows 10, but not on Linux, Macintosh, or earlier versions of Windows.

For all three parts of the system, you will need:

- A program to extract .zip archives. For example, [WinZip](#) or [7-Zip](#).
- Access to your computer's [command prompt](#).
- A simple text editor, such as Notepad or [Notepad++](#).

2.1 Artificial intelligence and Chatbot

To run the artificial intelligence and chatbot services, you will need:

- [Python](#) version 3.9.4 or higher
- Pip version 21.2.4 or higher (comes with Python)

2.2 Backend

To run the backend, you will need:

- [NodeJS](#) version 14.16.1 or higher
- Node Package Manager (NPM) version 7.20.6 or higher (comes with NodeJS)
- [Firebase Command Line Interface](#) (CLI) version 9.16.3 or higher
- [Java](#) version 1.8 or higher
- A [Google account](#)

2.3 Frontend

To run the frontend, you will need:

- [Flutter](#) version 2.2.3 or higher
- Dart version 2.13.4 or higher (comes with Flutter)
- [Android Studio](#) version 4.2.1 or higher

3 Download the source code

To get a copy of Coviduous, first download the repository from GitHub onto your local machine:

- In a web browser, go to <https://github.com/COS301-SE-2021/Coviduous>, click on the green "Code" button, and then "Download ZIP". [Figure 1] [Figure 2]
- Extract the .zip archive to a new folder anywhere on your computer.

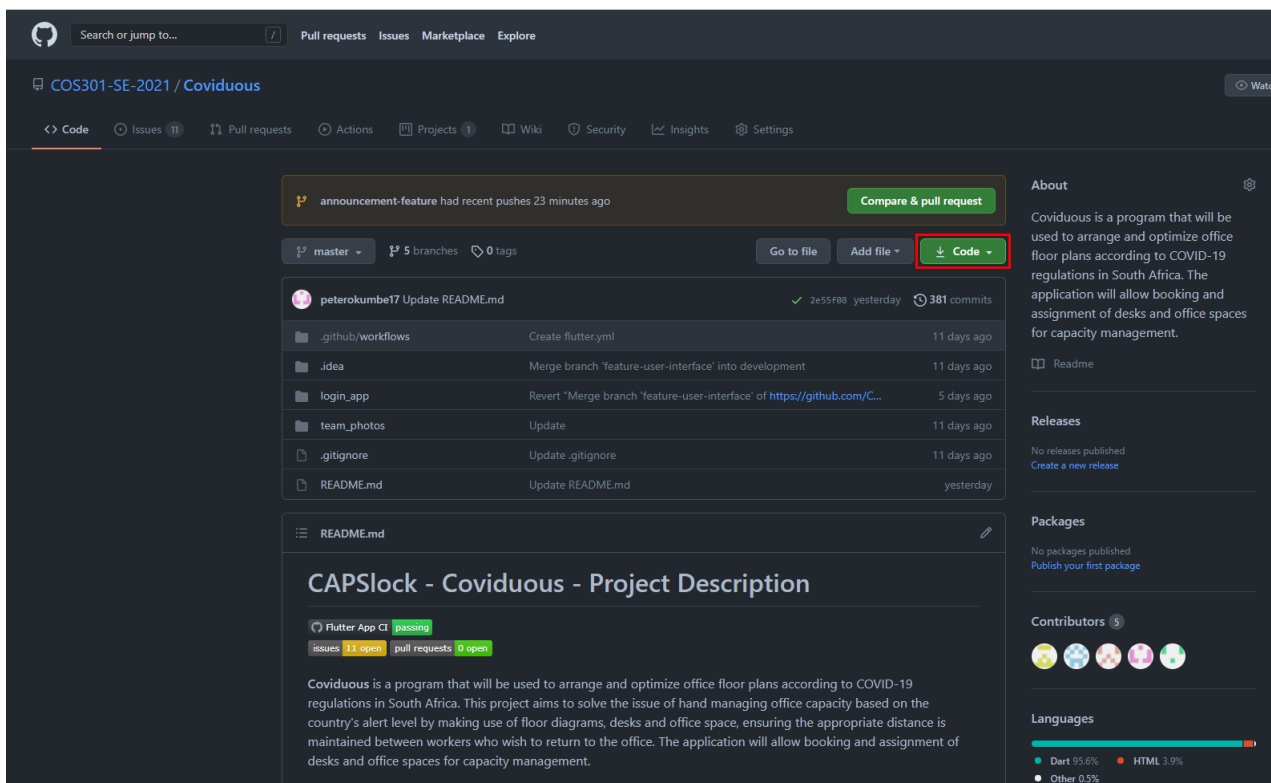


Figure 1: Download the source code

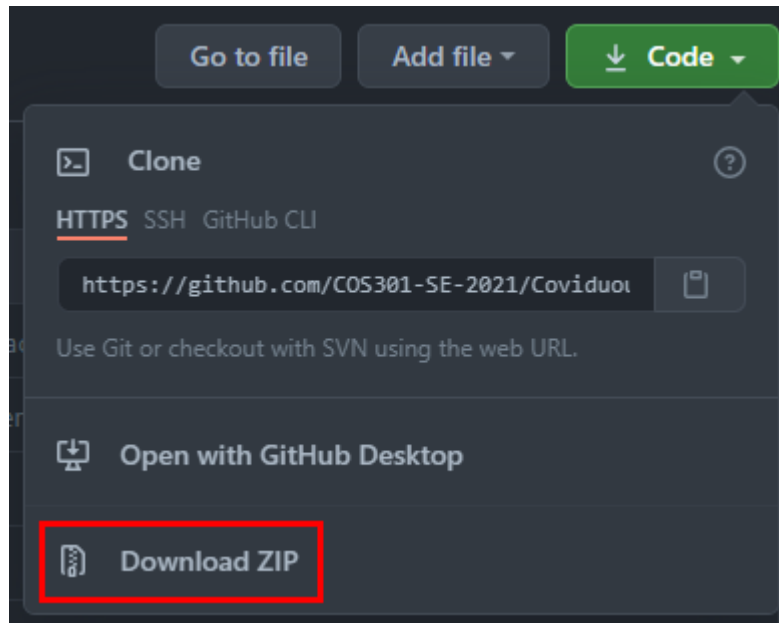


Figure 2: Download the source code

4 Installation

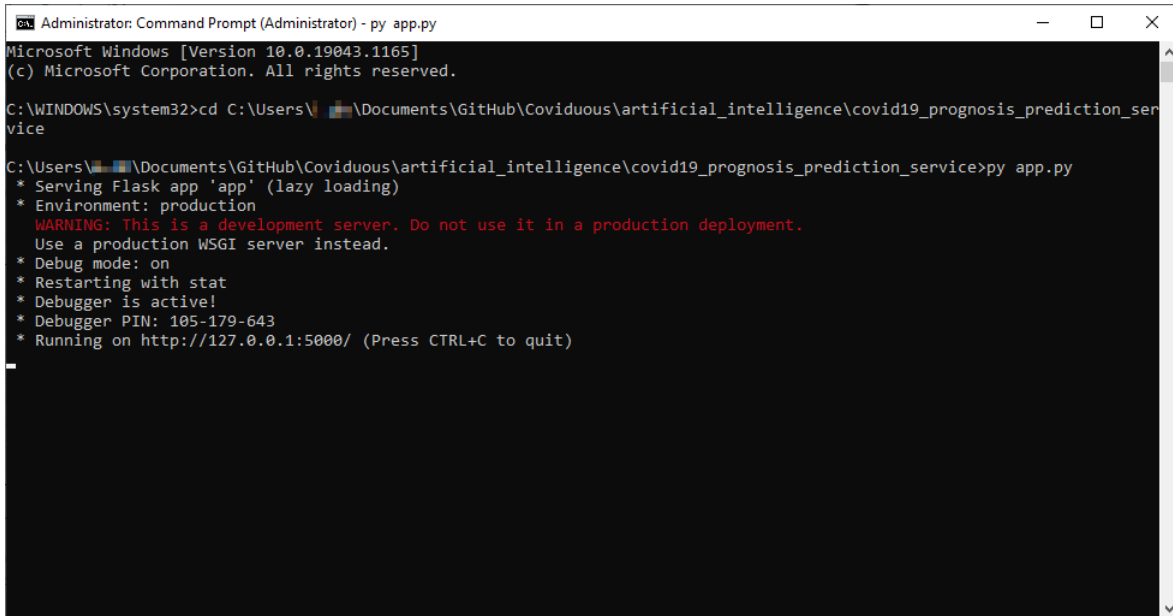
4.1 Artificial intelligence

To run the artificial intelligence service, do the following:

- Open a new command prompt, and [navigate](#) into the `artificial_intelligence/covid19_prognosis_prediction` directory of your extracted .zip archive.
- Run the following commands:
 - `py -m pip install flask`
 - `py -m pip install gunicorn`
 - `py -m pip install matplotlib`
 - `py -m pip install seaborn`
 - `py -m pip install pandas`
 - `py -m pip install numpy`
 - `py -m pip install scikit_learn`
- Then, run the service using the command: [Figure 3]

```
py app.py
```

You can stop the service by pressing `Ctrl+C` while the command prompt is open.



```
Administrator: Command Prompt (Administrator) - py app.py
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Users\...\Documents\GitHub\Coviduous\artificial_intelligence\covid19_prognosis_prediction_service

C:\Users\...\Documents\GitHub\Coviduous\artificial_intelligence\covid19_prognosis_prediction_service>py app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 105-179-643
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Figure 3: Start up the AI service

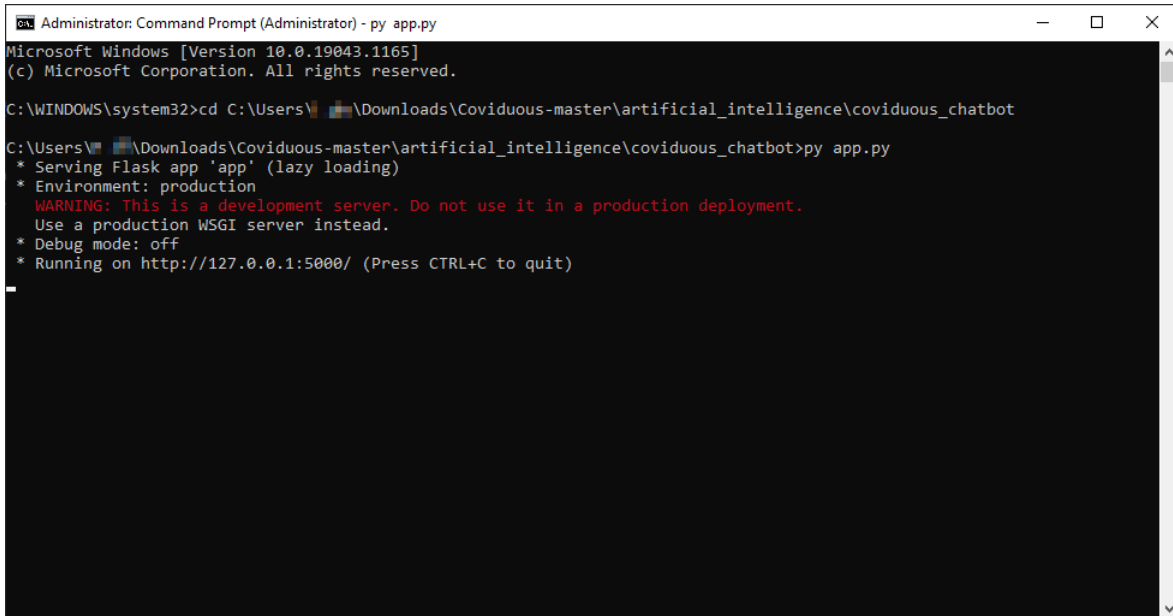
4.2 Chatbot

To run the chatbot service, do the following:

- Open a new command prompt, and [navigate](#) into the `artificial_intelligence/coviduous_chatbot` directory of your extracted .zip archive.
- Run the following commands:
 - `py -m pip install flask`
 - `py -m pip install collection`
 - `py -m pip install nltk`
 - `py -m pip install six`
- Then, run the service using the command: [Figure 4]

```
py app.py
```

You can stop the service by pressing Ctrl+C while the command prompt is open.



```
Administrator: Command Prompt (Administrator) - py app.py
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Users\...\Downloads\Coviduous-master\artificial_intelligence\coviduous_chatbot

C:\Users\...\Downloads\Coviduous-master\artificial_intelligence\coviduous_chatbot>py app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Figure 4: Start up the chatbot service

4.3 Backend

As the backend requires a private key generated by Firebase to interact with it, you will need to set up a new Firebase project. We cannot provide the private key that we have generated for our system as that is a huge security risk.

- Log into your Google account and go to <https://firebase.google.com>. Select "Go to console" at the top right corner of the page. [Figure 5]
- Select "Add project", enter a name when prompted, disable Google Analytics, and select "Create project". [Figure 6] [Figure 7]
- After your project has been created, you will be taken to its overview page. Under "Get started by adding Firebase to your app", select the Android option. [Figure 8]
- Enter an Android package name when prompted and (optionally) an app nickname. [Figure 9]
- Download the google-services.json file and go to where you extracted the .zip archive on your computer. Place the file in frontend/android/app, overwriting the google-services.json that is already there. [Figure 10]
- Ignore the instructions on how to add the Firebase SDK, as that has already been done for Coviduous. Select "Continue to the console". [Figure 11]
- Back on the project overview page, select the "Authentication" option in the "Build" tab on the left side of the screen. [Figure 12]
- Select "Get started", and on the next page, under "Sign-in providers", edit the "Email/Password" option and enable it. You do not need to enable "Email link (passwordless sign-in)". Click "Save". [Figure 13]

- Then, on the left side of your screen, next to "Project Overview", select the settings icon (the gear) and then "Project settings". Navigate to the "Service accounts" tab and select "Generate new private key". This will download a .json file to your computer. Rename the file to "permissions.json" and place it in the backend/functions folder of your extracted .zip archive. [Figure 14]
- In backend/functions, create a new file named ".env", without anything before the dot. Using a text editor, add lines in the file as shown in Figure 15. The FirebaseClientAPIKey can be found in your Firebase project's settings, under the "General" tab. Copy the key and paste it after the equals sign. The FirebaseClientProjectID is your project's ID (also found in the "General" tab), and the FirebaseClientAuthDomain is your project's ID with ".firebaseapp.com" appended to the end. For example, "test.firebaseio.com".
- Finally, in a new command prompt, [navigate](#) to the backend/functions directory and run the command:

```
npm install
```

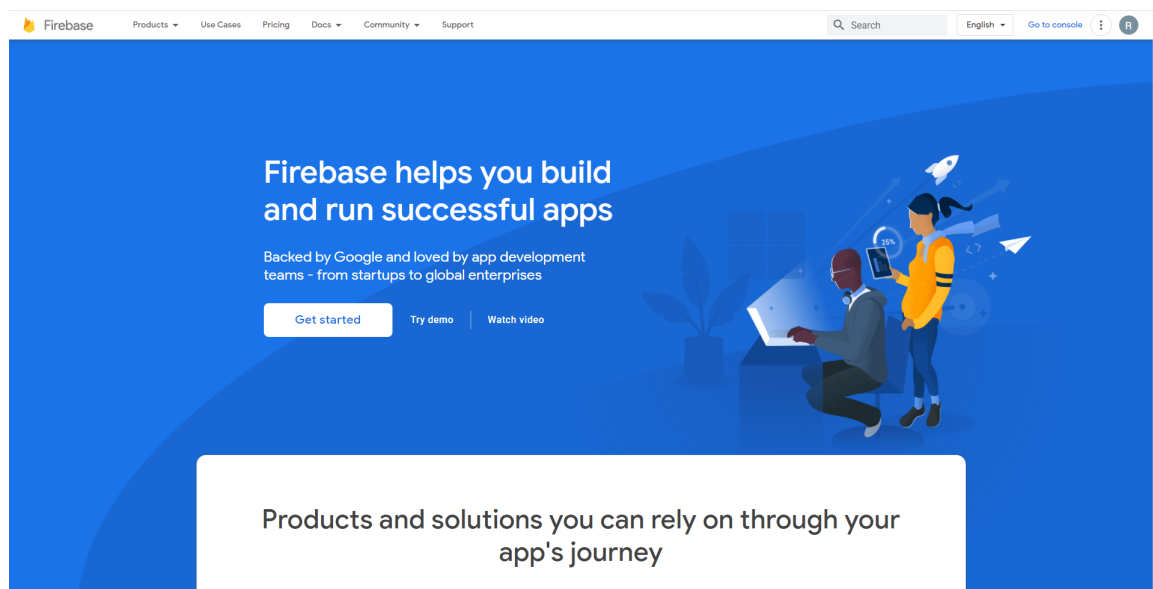


Figure 5: **Set up Firebase**

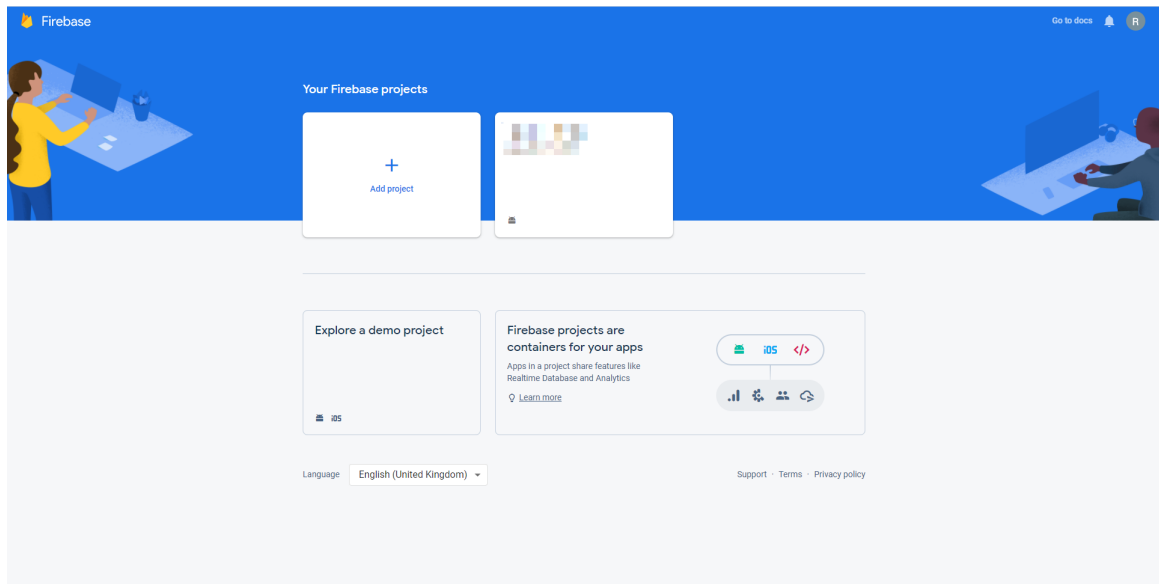


Figure 6: Set up Firebase

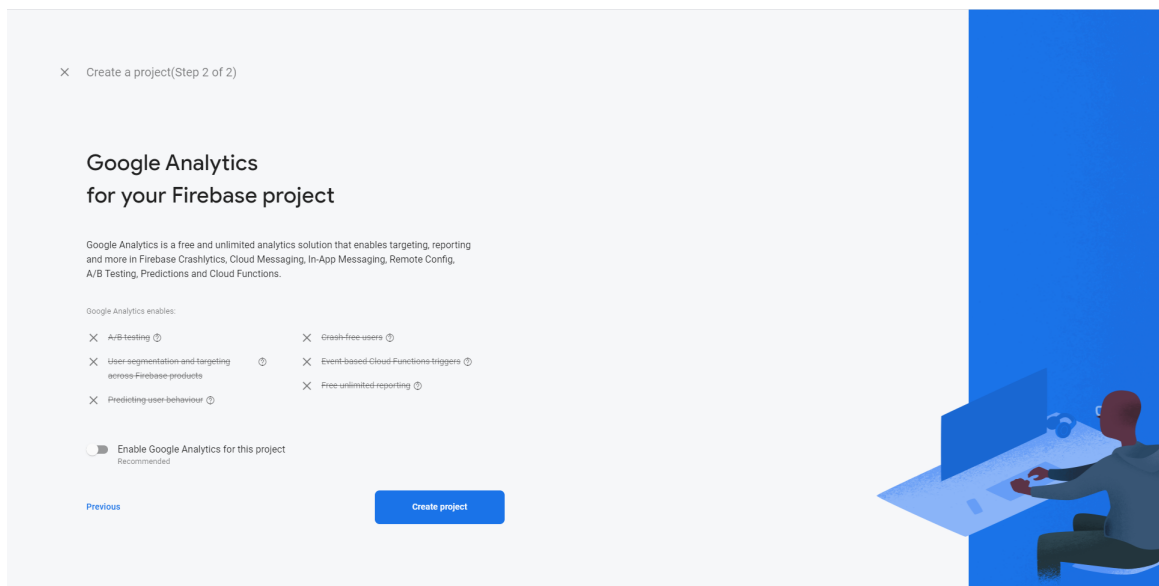


Figure 7: Set up Firebase

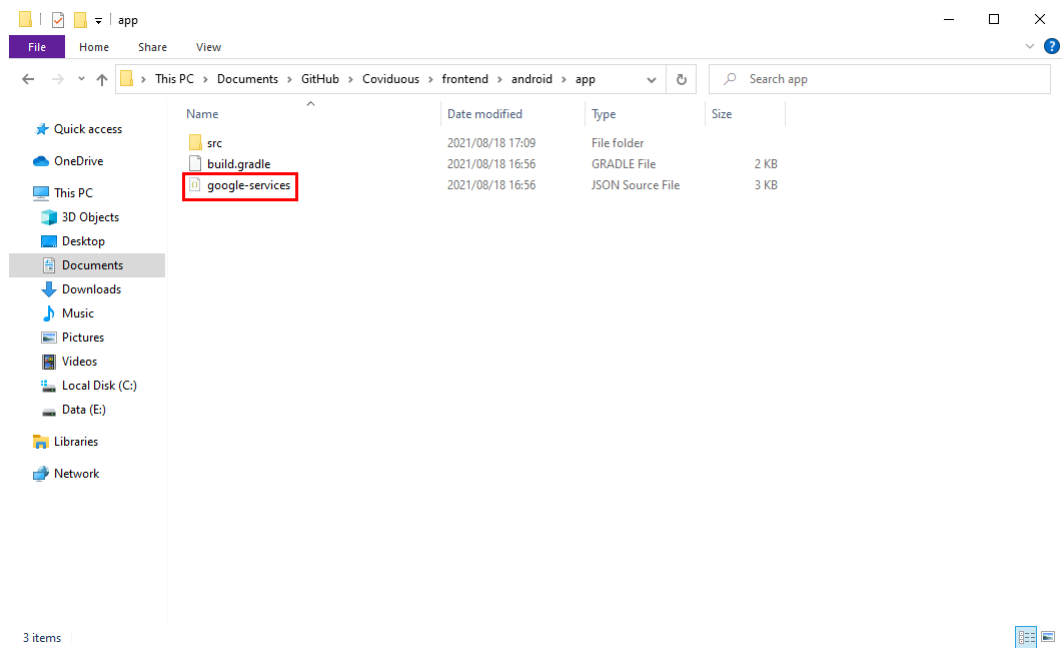


Figure 10: Generate Firebase config files

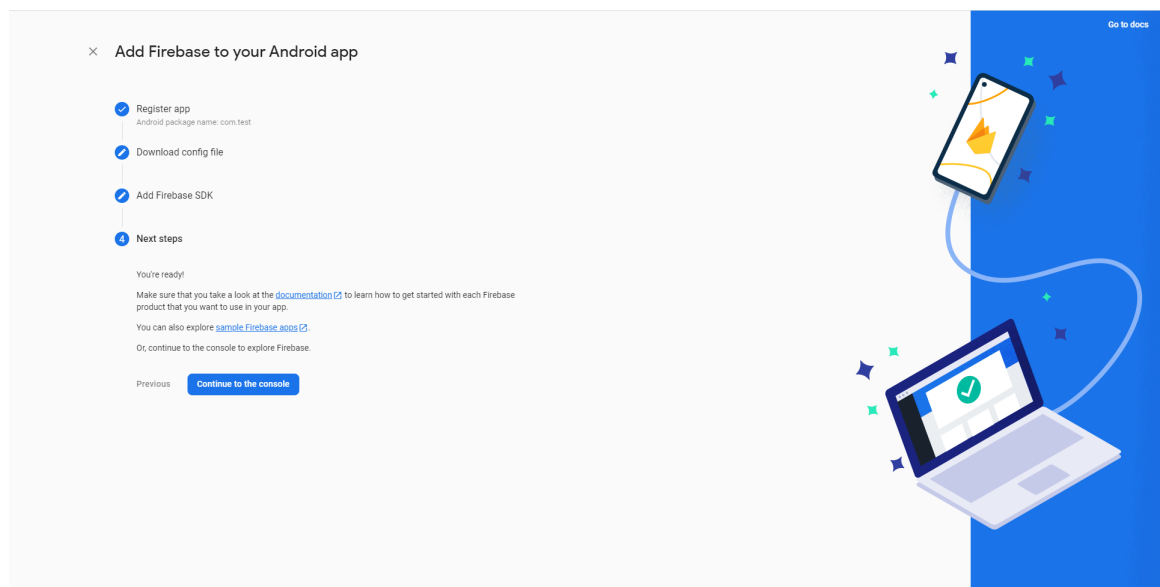


Figure 11: Generate Firebase config files

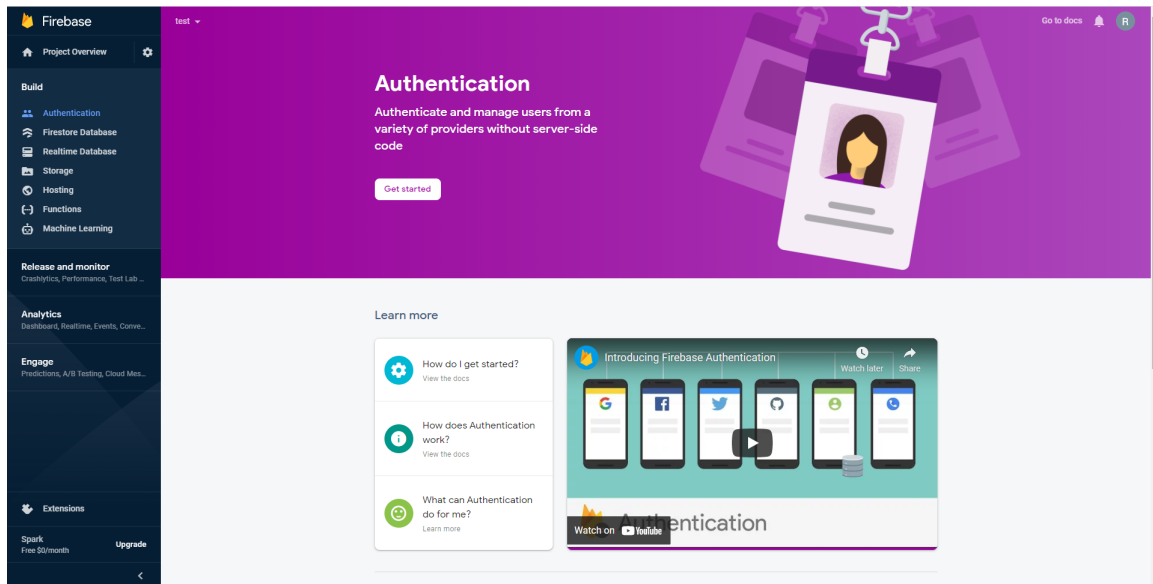


Figure 12: Set up Firebase Authentication

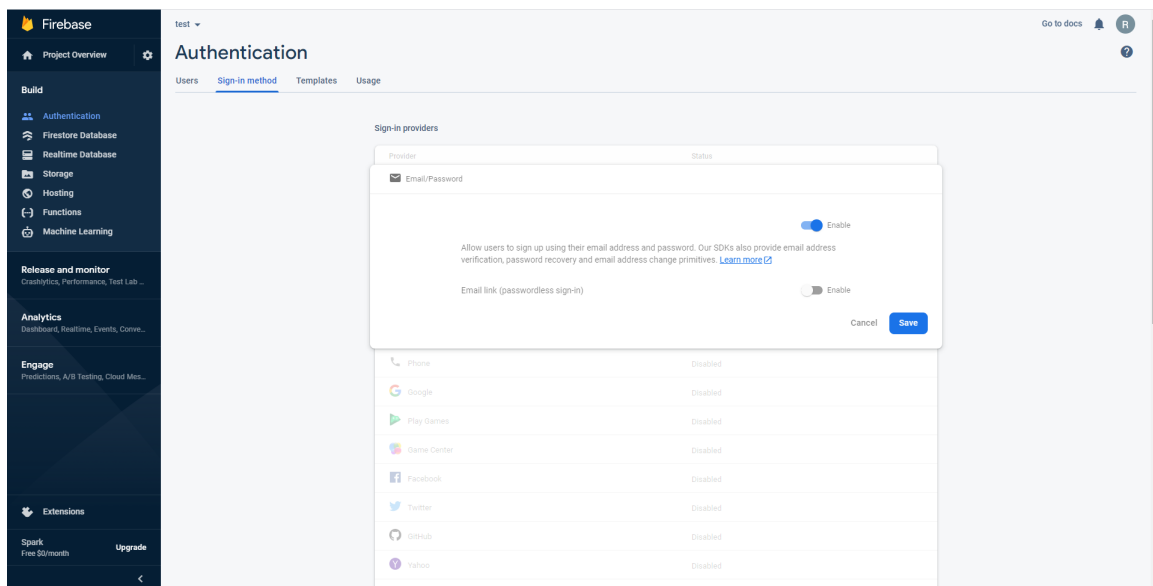


Figure 13: Set up Firebase Authentication

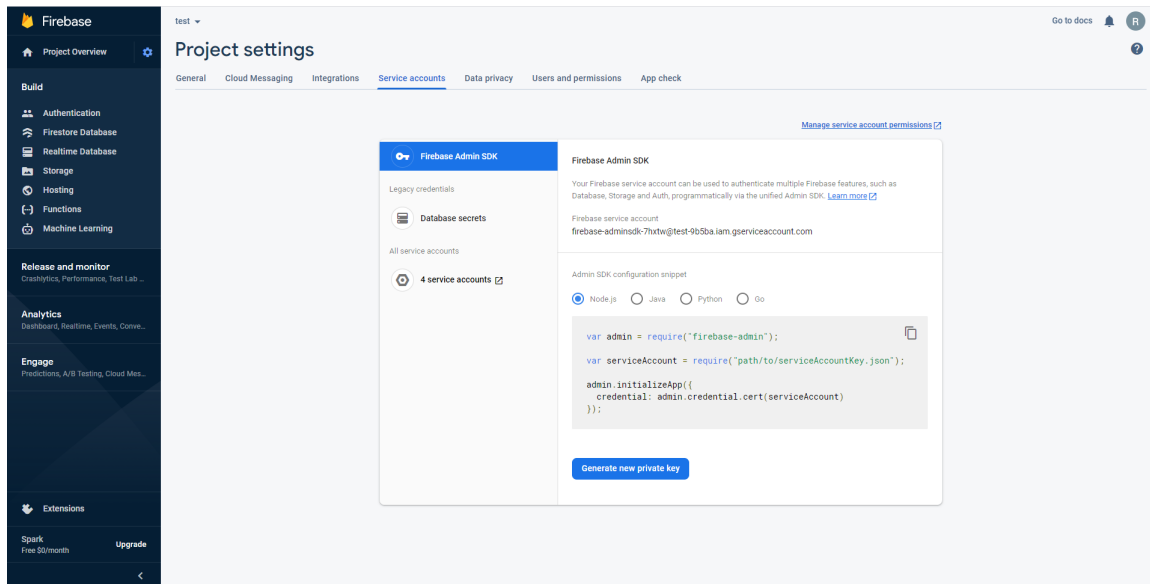


Figure 14: Download your app's private key

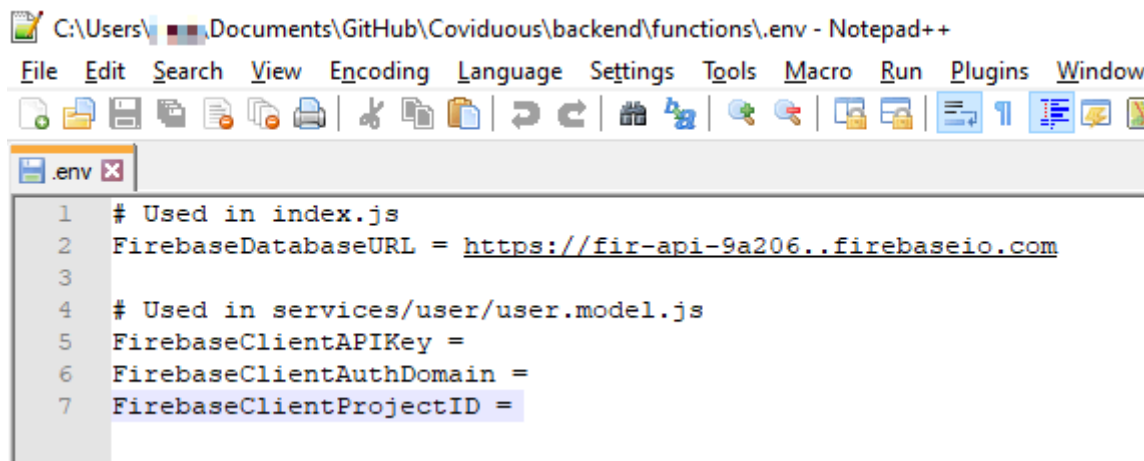


Figure 15: Create .env file

Now that Firebase is set up for your project, you have to set up the emulators on your local machine. Using your computer's command prompt, navigate into the backend folder of your extracted .zip archive. Run the following commands:

- `firebase --version`
 - This is to check if you have Firebase CLI installed. If it is not, please follow the link under the "System requirements" section in this manual.
- `firebase init`
- `firebase init emulators`
 - Install the emulators for Functions and Firestore. The rest are unnecessary. Navigate the options using your arrow keys, select an option using the spacebar, and confirm your selection using the enter key.

- Now you can start up the emulators by navigating into the functions directory and running the command:

```
firebase emulators:start
```

You can stop the emulators by pressing Ctrl+C while the command prompt is open.

4.4 Frontend

Using Flutter, you can run an emulated version of the frontend app on either a web browser or an Android emulator (provided by Android Studio).

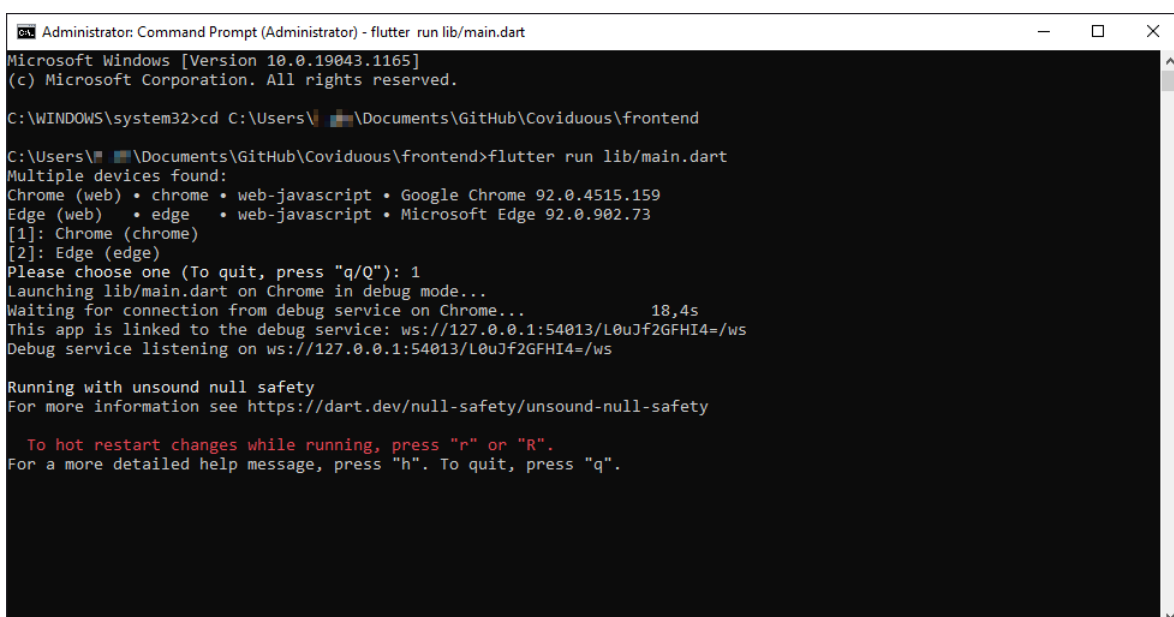
4.4.1 Run on a web browser

To run the app on a web browser: [Figure 16] [Figure 17]

- Open a new command prompt and [navigate](#) into the frontend directory of your extracted .zip archive.
- Run the command:

```
flutter run lib/main.dart
```

- When prompted, select which browser you would like to use. After a few seconds, the app should open in a new browser window. Depending on whether the backend and the AI service are also running, the app may have limited functionality.
- To stop the app, press the Q key while the command prompt is open. Give it some time to shut down properly (usually a minute or so). If it takes much longer than a minute, force the app to stop with Ctrl+C.



```
Administrator: Command Prompt (Administrator) - flutter run lib/main.dart
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Users\...\Documents\GitHub\Coviduuous\frontend

C:\Users\...\Documents\GitHub\Coviduuous\frontend>flutter run lib/main.dart
Multiple devices found:
Chrome (web) • chrome • web-javascript • Google Chrome 92.0.4515.159
Edge (web)   • edge   • web-javascript • Microsoft Edge 92.0.902.73
[1]: Chrome (chrome)
[2]: Edge (edge)
Please choose one (To quit, press "q/Q"): 1
Launching lib/main.dart on Chrome in debug mode...
Waiting for connection from debug service on Chrome... 18,4s
This app is linked to the debug service: ws://127.0.0.1:54013/L0uJf2GFHI4=/ws
Debug service listening on ws://127.0.0.1:54013/L0uJf2GFHI4=/ws

Running with unsound null safety
For more information see https://dart.dev/null-safety/unsound-null-safety

To hot restart changes while running, press "r" or "R".
For a more detailed help message, press "h". To quit, press "q".
```

Figure 16: Run the app in the command prompt

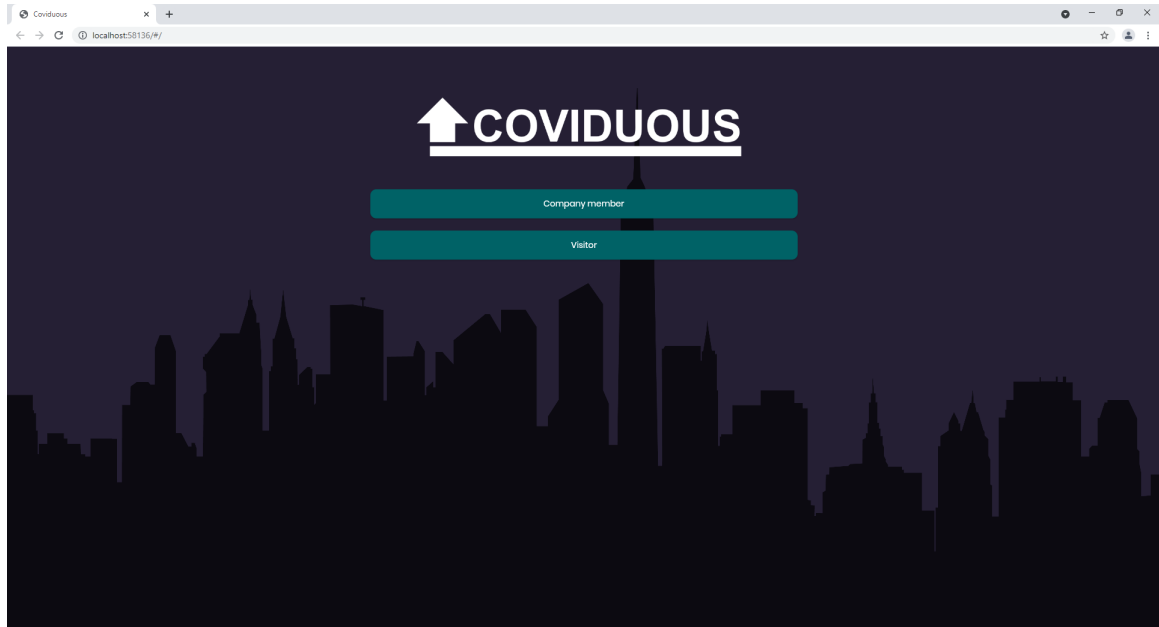


Figure 17: The app running on Google Chrome

4.4.2 Run on an Android emulator

To run the app on an Android emulator:

- Open Android Studio and navigate to the [AVD manager](#) and create a new AVD if there is not one already. [Figure 18]
- Start your emulator and wait for it to display a phone home screen. [Figure 19]
- Run the command:

```
flutter run lib/main.dart
```

[Figure 20]

- Flutter will automatically try to use the emulator when it is running. After a few minutes, the app should appear on your emulator screen. [Figure 21]
- To stop the app, press the Q key while the command prompt is open. Give it some time to shut down properly (usually a minute or so). If it takes much longer than a minute, force the app to stop with Ctrl+C.

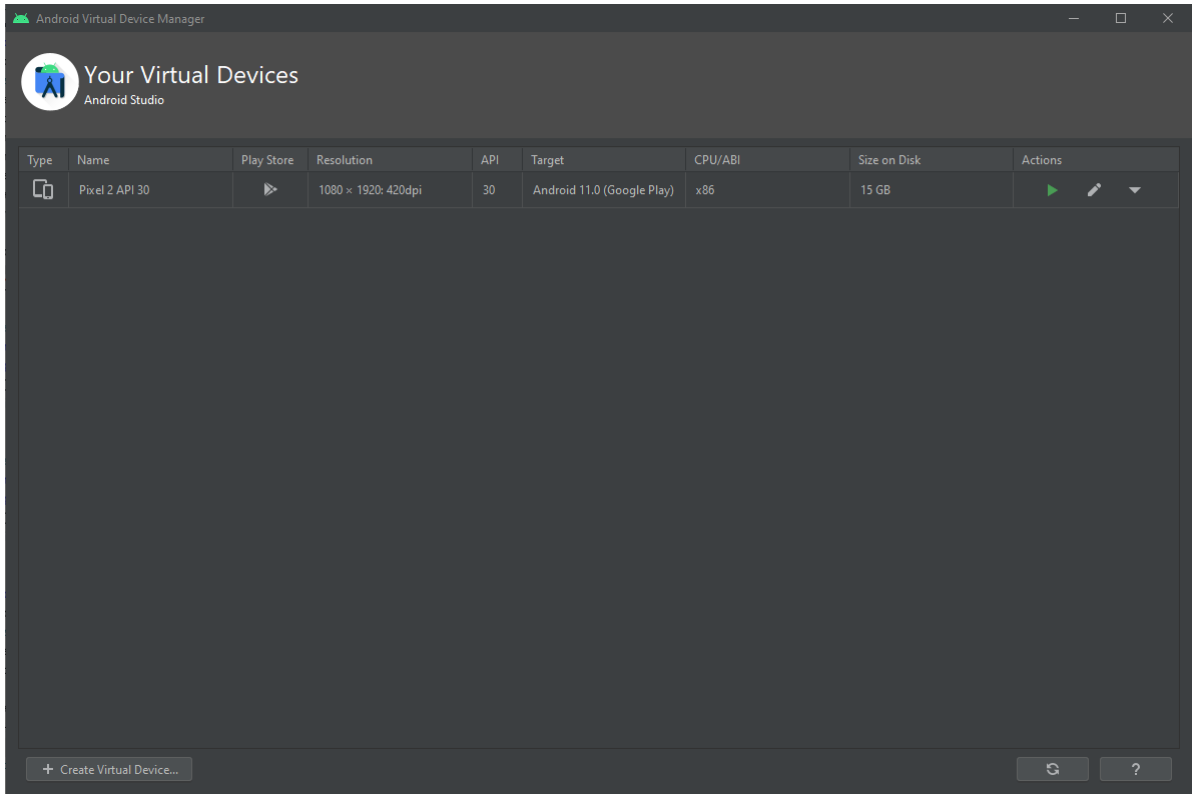


Figure 18: Open the AVD manager in Android Studio



Figure 19: Run the Android emulator

```
Administrator: Command Prompt (Administrator) - flutter run lib/main.dart
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Users\...\Documents\GitHub\Coviduous\frontend

C:\Users\...\Documents\GitHub\Coviduous\frontend>flutter run lib/main.dart
Using hardware rendering with device sdk gphone x86. If you notice graphics artifacts, consider enabling software
rendering with "--enable-software-rendering".
Launching lib/main.dart on sdk gphone x86 in debug mode...
Running Gradle task 'assembleDebug'...
Running Gradle task 'assembleDebug'... Done                    94,6s
✓ Built build/app/outputs/flutter-apk/app-debug.apk.
Syncing files to device sdk gphone x86...                    143ms

Flutter run key commands.
r Hot reload.
R Hot restart.
h Repeat this help message.
d Detach (terminate "flutter run" but leave application running).
c Clear the screen
q Quit (terminate the application on the device).

Running with unsound null safety
For more information see https://dart.dev/null-safety/unsound-null-safety

An Observatory debugger and profiler on sdk gphone x86 is available at: http://127.0.0.1:50245/PwT2xSXsmV4=/
W/DynamiteModule( 5470): Local module descriptor class for providerinstaller not found.
W/System ( 5470): Ignoring header X-Firebase-Locale because its value was null.
I/DynamiteModule( 5470): Considering local module providerinstaller:0 and remote module providerinstaller:0
W/ProviderInstaller( 5470): Failed to load providerinstaller module: No acceptable module found. Local version is 0 and
```

Figure 20: Run the app in the command prompt



Figure 21: The app running on the emulator