

# Faculté des technologies de l'information et de la communication

Département informatique appliquée



UNIVERSITÉ DES  
**MASCAREIGNES**  
SAVOIR, C'EST POUVOIR

Travaux pratique

**TP4**

Réalisé par : Nathan Carlinot RANDRIAMIHAJA

Année scolaire : 2023 - 2024

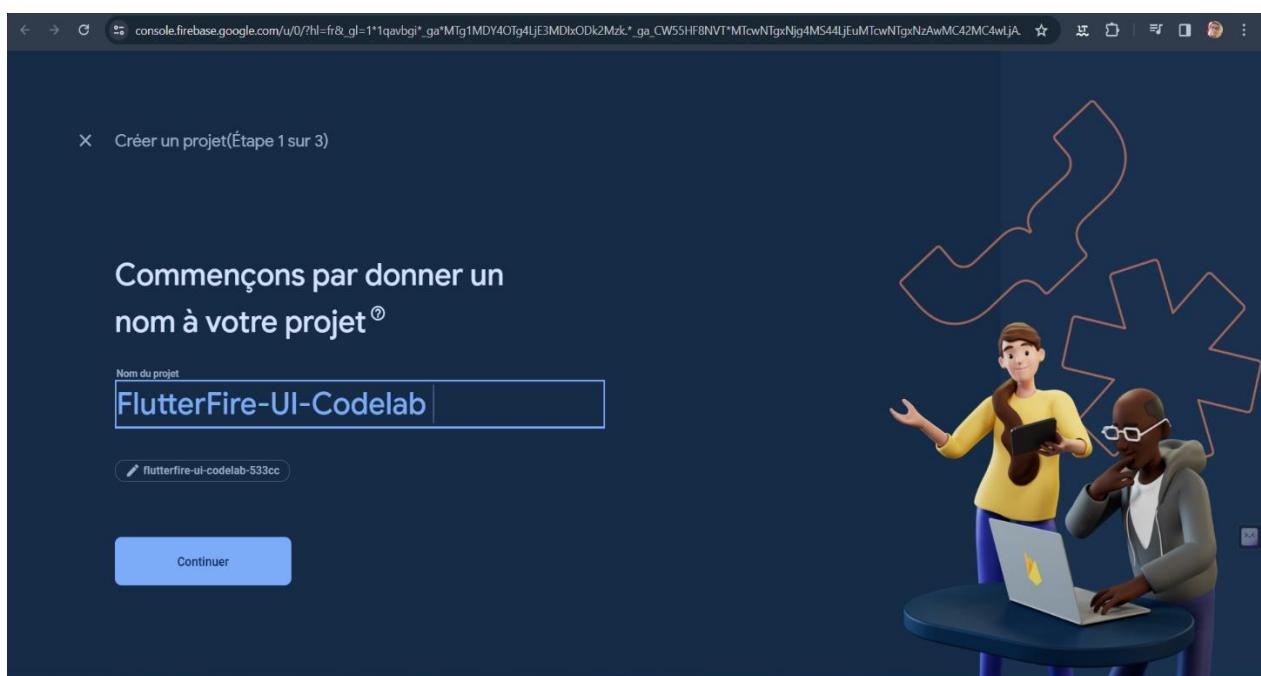
## Task 2

### A- Créer et configurer un projet Firebase

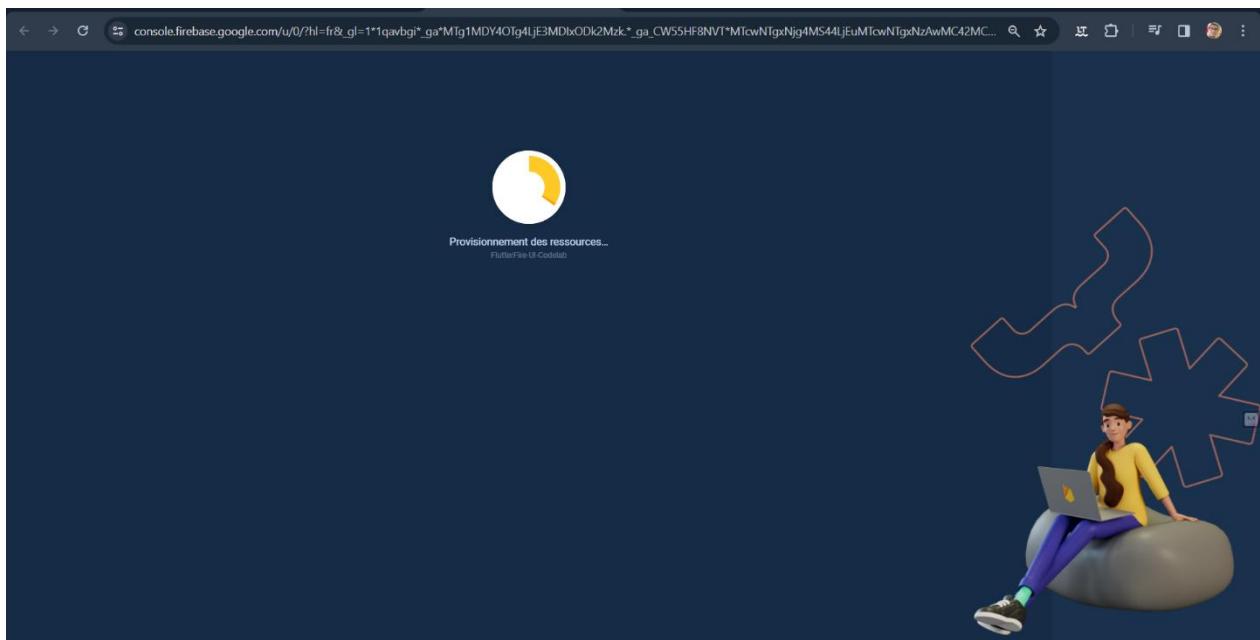
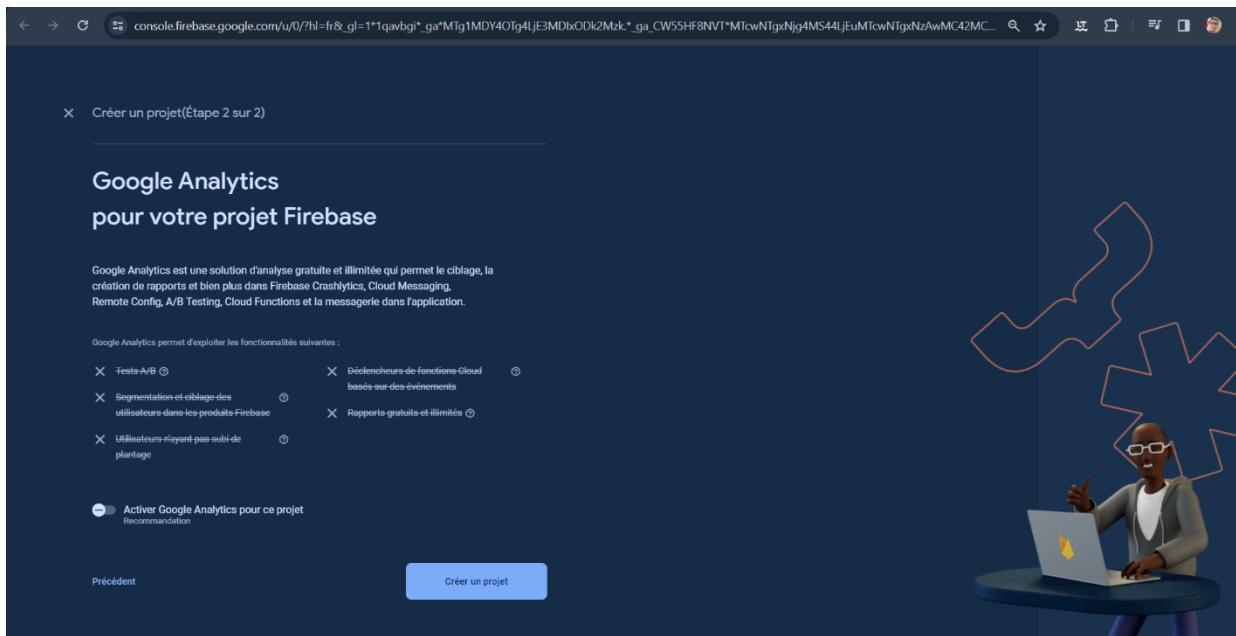
- 1- Créez un projet Firebase
- Connectez à Firebase

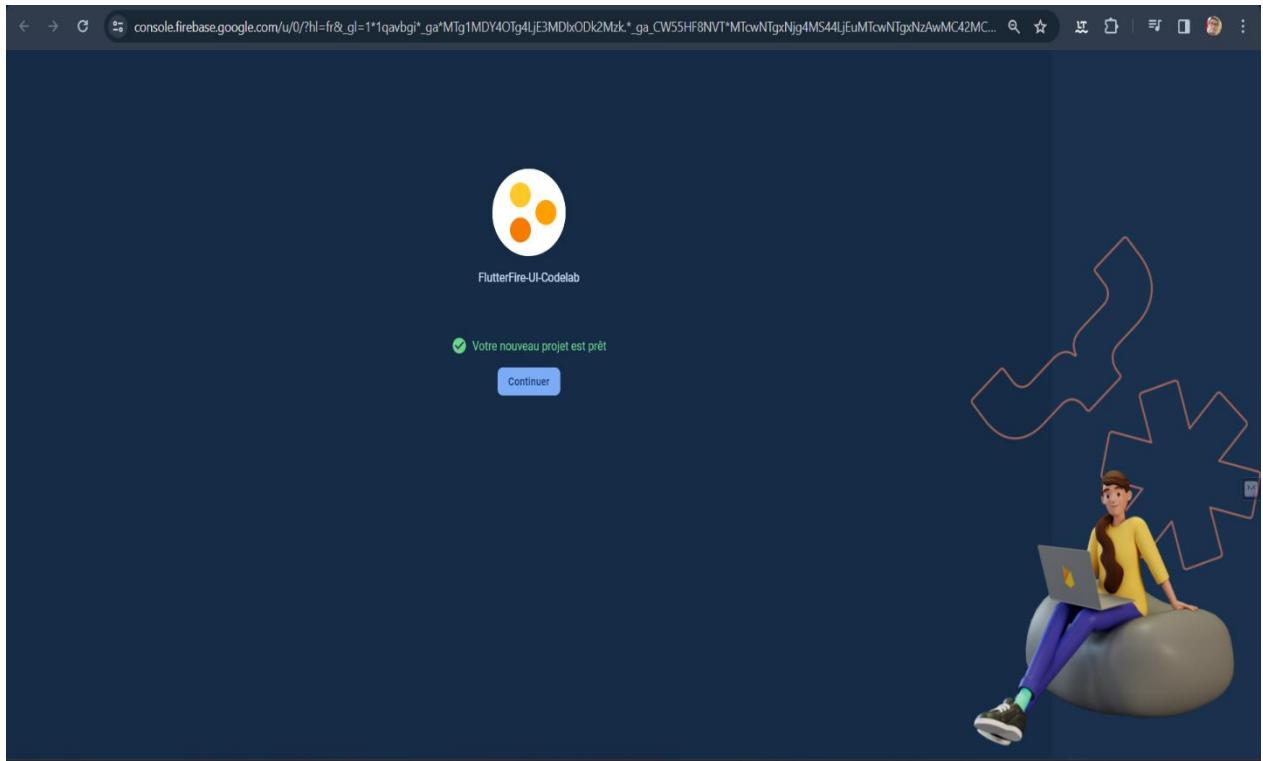


- Dans la console Firebase, cliquez sur Ajouter un projet (ou Créez un projet) et saisissez un nom pour votre projet Firebase (par exemple, "FlutterFire-UI-Codelab").



- Cliquez sur les options de création de projet. Acceptez les conditions de Firebase si vous y êtes invité. Ignorez la configuration de Google Analytics, car vous n'utiliserez pas Analytics pour cette application.



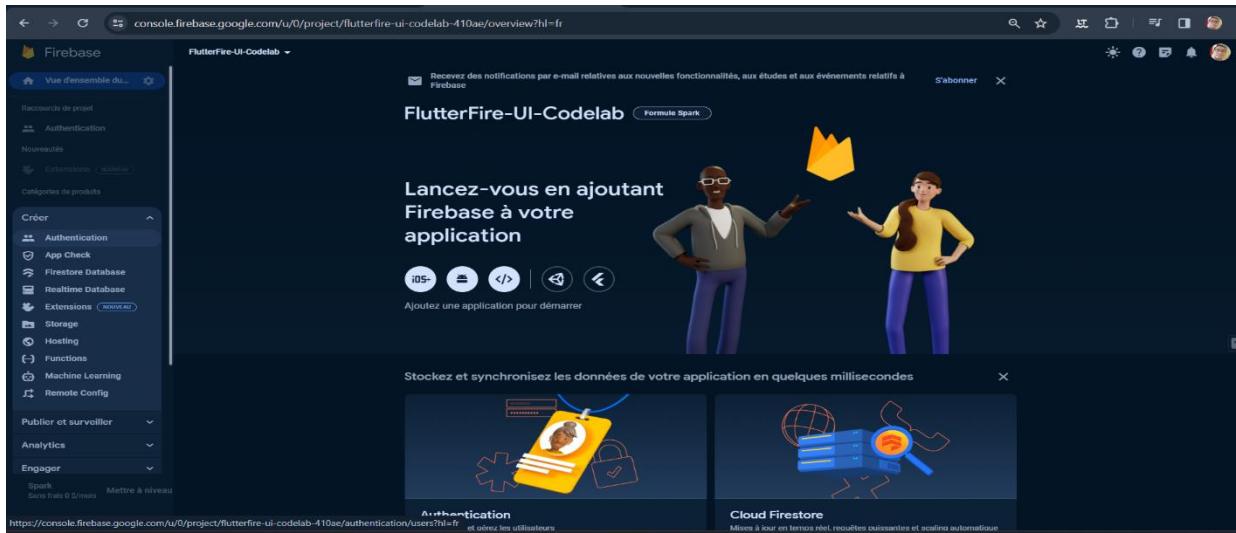


A screenshot of the Firebase console's project overview page for "FlutterFire-UI-Codelab". The left sidebar shows navigation options like "Vue d'ensemble du ...", "Extensions", "Créer", "Publier et surveiller", "Analytics", and "Engager". The main area features a large call-to-action button with the text "Lancez-vous en ajoutant Firebase à votre application" and two cartoon characters. Below this, there's a section titled "Stockez et synchronisez les données de votre application en quelques millisecondes" with cards for "Authentication" and "Cloud Firestore". A sidebar on the left says "Personnalisez votre navigation" and "Spark".

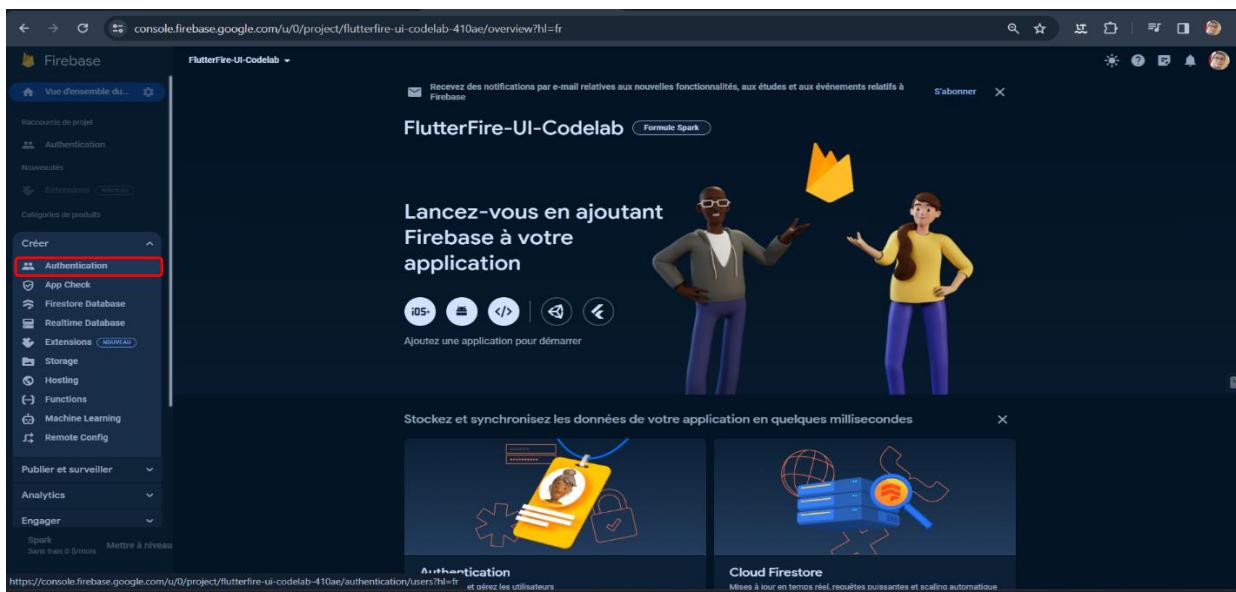
## 2- Activer la connexion par e-mail pour l'authentification Firebase

Pour permettre aux utilisateurs de se connecter à l'application Web, vous utiliserez d'abord la méthode de connexion par e-mail/mot de passe . Plus tard, vous ajouterez la méthode Google Sign-In .

- Dans la console Firebase, développez le menu Crée dans le panneau de gauche.



- Cliquez sur Authentication , puis cliquez sur le bouton Commencer , puis sur l'onglet Méthode de connexion (ou cliquez ici pour accéder directement à l'onglet Méthode de connexion ).



console.firebaseio.google.com/u/0/project/flutterfire-ui-codelab-410ae/authentication?hl=fr

**Firebase** FlutterFire-UI-Codelab

Raccourcis de projet

**Authentification**

Nouveautés

Extensions (NOUVEAU)

Catégories de produits

Créer

Publier et surveiller

Analytics

Engager

Tous les produits

Personnalisez votre navigation

Personnalisez votre navigation pour optimiser votre expérience

En savoir plus OK

Spark Sans frais 0 \$/mois Mettre à niveau

**Authentication**

Authentifiez et gérez les utilisateurs à partir de plusieurs fournisseurs sans code côté serveur

Commencer

En savoir plus

Par où commencer ? Afficher les documents

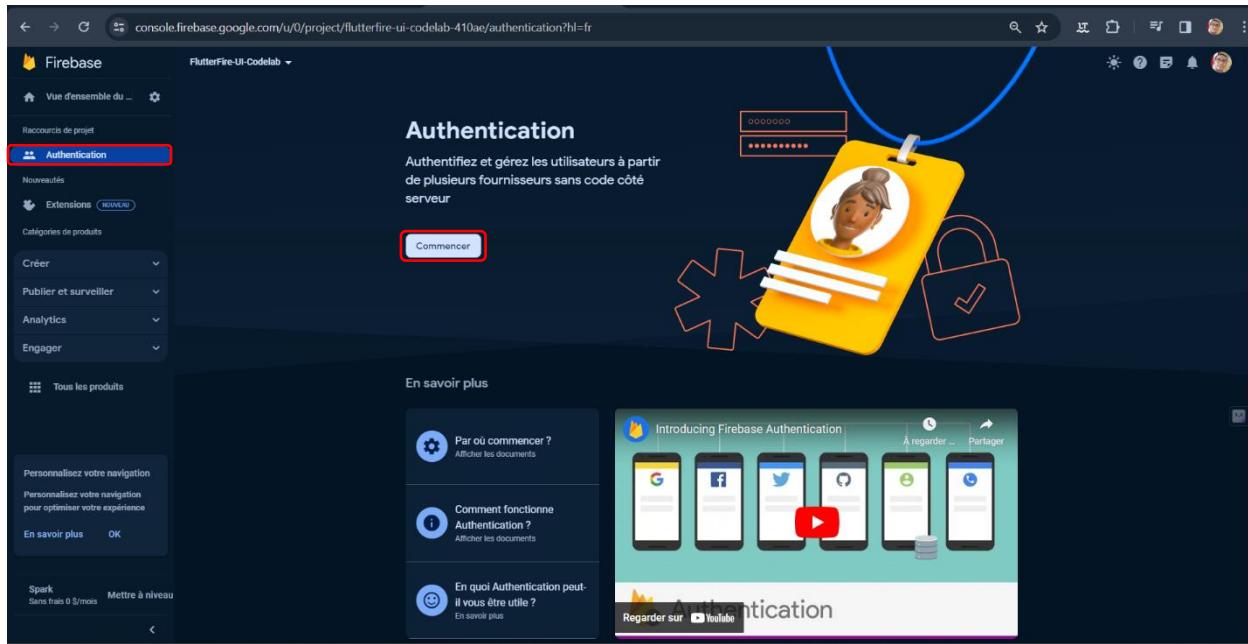
Comment fonctionne Authentication ? Afficher les documents

En quoi Authentication peut-il vous être utile ? En savoir plus

Introducing Firebase Authentication

A regarder ... Partager

Regarder sur YouTube



console.firebaseio.google.com/u/0/project/flutterfire-ui-codelab-410ae/authentication/providers?hl=fr

**Firebase** FlutterFire-UI-Codelab

Vue d'ensemble du ...

**Authentification**

Users Sign-in method Templates Usage Settings Extensions

Nouveautés

Extensions (NOUVEAU)

Catégories de produits

Créer

Publier et surveiller

Analytics

Engager

Tous les produits

Personnalisez votre navigation

Personnalisez votre navigation pour optimiser votre expérience

En savoir plus OK

Spark Sans frais 0 \$/mois Mettre à niveau

Fournisseurs de connexion

Familiarisez-vous avec Firebase Auth en ajoutant votre première méthode de connexion

Fournisseurs natifs

Adresse e-mail/Mot de passe

Téléphone

Anonyme

Autres fournisseurs

Google Facebook Play Jeux

Game Center Apple

Github Microsoft Twitter

Yahoo!

Fournisseurs personnalisés

OpenID Connect

SAML

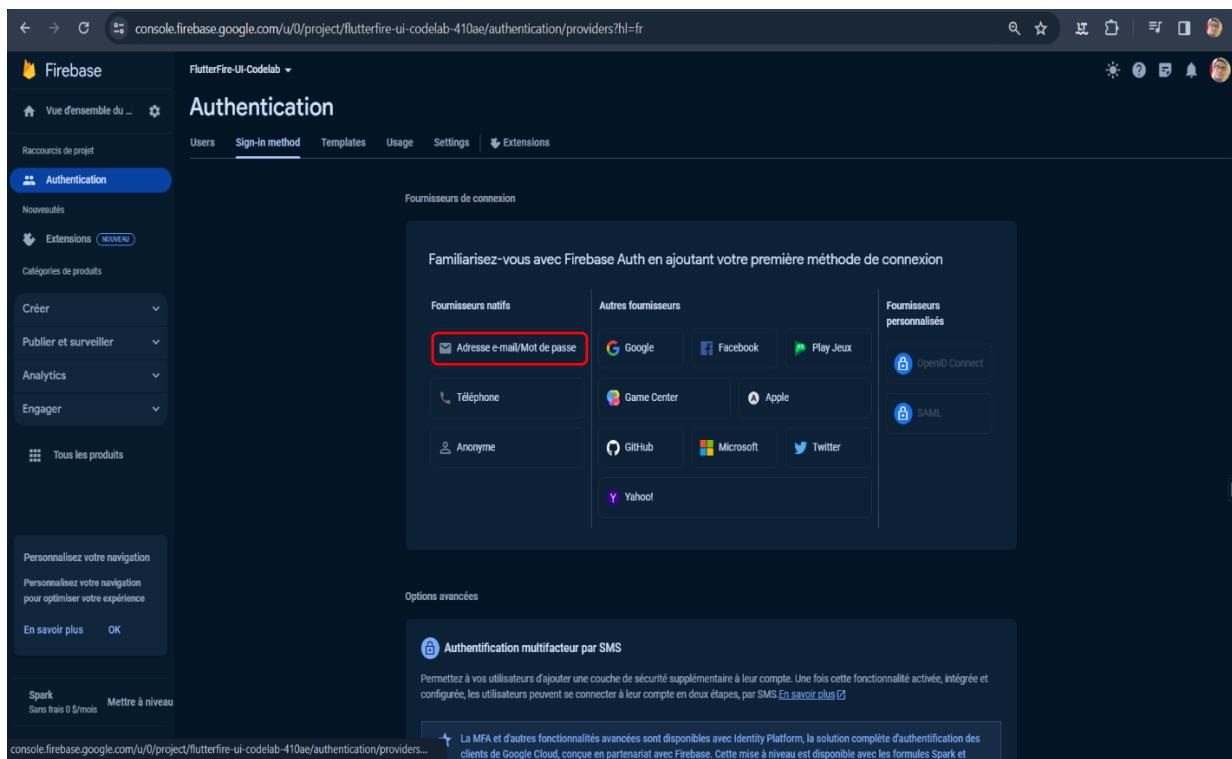
Options avancées

Authentication multifacteur par SMS

Permettez à vos utilisateurs d'ajouter une couche de sécurité supplémentaire à leur compte. Une fois cette fonctionnalité activée, intégrée et configurée, les utilisateurs peuvent se connecter à leur compte en deux étapes, par SMS En savoir plus

La MFA et d'autres fonctionnalités avancées sont disponibles avec Identity Platform, la solution complète d'authentification des clients de Google Cloud, conçue en partenariat avec Firebase. Cette mise à niveau est disponible avec les formules Spark et

console.firebaseio.google.com/u/0/project/flutterfire-ui-codelab-410ae/authentication/providers...



The screenshot shows the Firebase console's Authentication settings for a project named 'FlutterFire-UI-Codelab'. The 'Sign-in method' tab is active. A modal window is open, titled 'Fournisseurs de connexion', with a sub-section for 'Adresse e-mail/Mot de passe'. The 'Adresse e-mail/Mot de passe' input field is highlighted with a red box. Below it is a checkbox labeled 'Activer'. At the bottom right of the modal is a blue button labeled 'Enregistrer' (Save), also highlighted with a red box. The background shows other configuration options like 'Options avancées' and 'Authentification multifactor par SMS'.

## B- Configurer l'application Flutter

### 1- Obtenez le code de démarrage

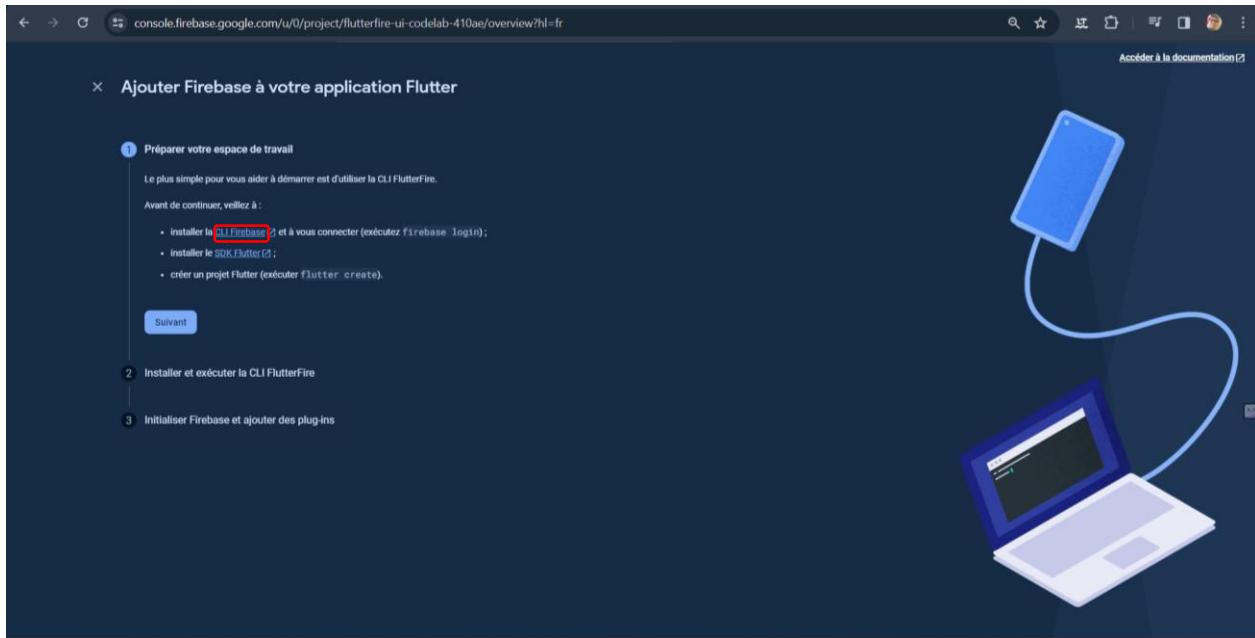
GitHub : [git clone https://github.com/flutter/codelabs.git flutter-codelabs](https://github.com/flutter/codelabs.git)

### 2- Installer la CLI Firebase

- Cliquer sur flutter

The screenshot shows the Firebase console's overview page for the 'FlutterFire-UI-Codelab' project. The main heading is 'FlutterFire-UI-Codelab' with a 'Formule Spark' badge. Below it is a large call-to-action: 'Lancez-vous en ajoutant Firebase à votre application', accompanied by two cartoon characters. Below this are two sections: 'Authentication' (with a sub-note 'Authentifiez et créez les utilisateurs') and 'Cloud Firestore' (with a sub-note 'Mises à jour en temps réel, requêtes puissantes et scalabilité automatique'). The left sidebar contains navigation links for 'Vue d'ensemble du ...', 'Raccourcis de projet', 'Authentication', 'Extensions', 'Créer', 'Publier et surveiller', 'Analytics', 'Engager', and 'Tous les produits'. A 'Personnalisez votre navigation' section is also present.

- Une fois cliquer sur Flutter, cette fenêtre apparait pour faire les installations, cliquons sur Cli Firebase avant tout



- Une fois le lien est cliqué, une fenetre de Firebase apparait, ensuite, cliquons sur Node.js pour télécharger sur Windows

**Configurer ou mettre à jour la CLI**

**Installer la CLI Firebase**

Vous pouvez installer la CLI Firebase à l'aide d'une méthode qui correspond à votre système d'exploitation, votre niveau d'expérience et/ou votre cas d'utilisation. Quelle que soit la façon dont vous installez la CLI, vous avez accès aux mêmes fonctionnalités et à la commande `firebase`.

**WindowsMacOSLinux**

**les fenêtres**

Vous pouvez installer Firebase CLI pour Windows à l'aide de l'une des options suivantes :

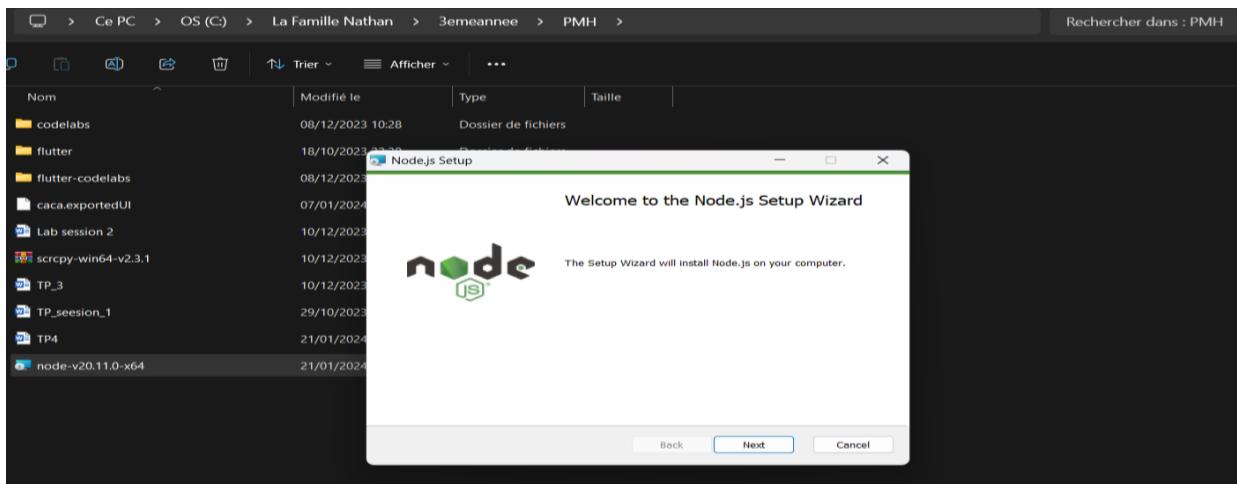
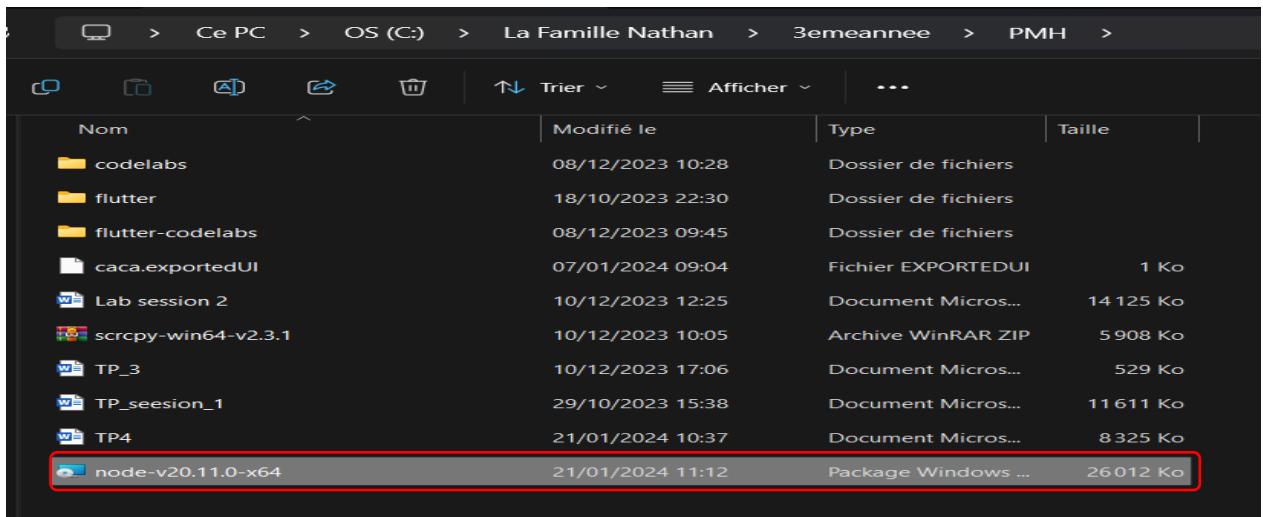
Option	Description	Recommandé pour...
binaire autonome	Téléchargez le binaire autonome pour la CLI. Ensuite, vous pouvez accéder à l'exécutable pour ouvrir un shell dans lequel vous pouvez exécuter la commande <code>firebase</code> .	Nouveaux développeurs
npm	Utilisez npm (le Node Package Manager) pour installer la CLI et activer la commande <code>firebase</code> disponible dans le monde entier.	Développeurs n'utilisant pas ou peu familiers avec Node.js

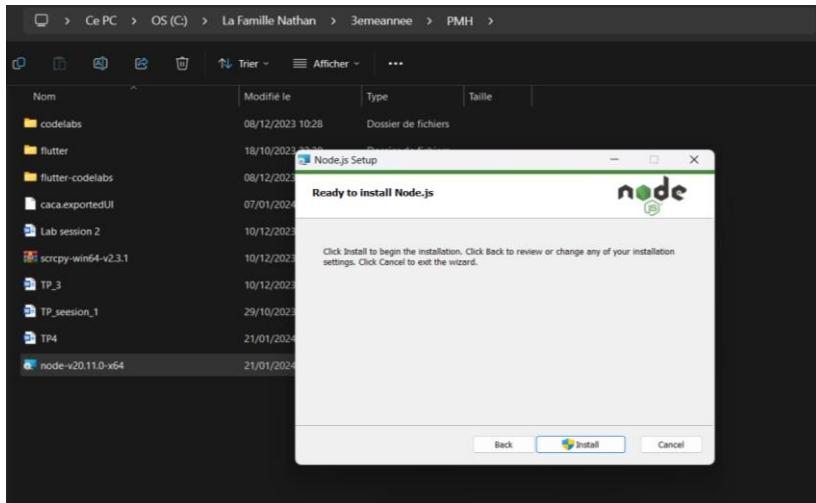
**Sur cette page**

- Configurer ou mettre à jour la CLI
- Installer la CLI Firebase
- Connectez-vous et testez la CLI Firebase
- Mise à jour vers la dernière version CLI
- Utiliser la CLI avec les systèmes CI
- Initialiser un projet Firebase
- Le fichier `firebase.json`
- Gérer les alias du projet
- Ajouter un alias de projet
- Utiliser les alias du projet
- Contrôle de source et alias de projet
- Servez et testez votre projet Firebase localement
- Déployer sur un projet Firebase
- Conflits de déploiement pour les règles de sécurité
- Quotas de déploiement
- Annuler un déploiement
- Déployer des services Firebase spécifiques
- Déployer des fonctions spécifiques

The screenshot shows the official Node.js website at [nodejs.org/en](https://nodejs.org/en). The page header includes links for LEARN, ABOUT, DOWNLOAD, GUIDES, BLOG, DOCS, and CERTIFICATION. Below the header, a banner states "Node.js® is an open-source, cross-platform JavaScript runtime environment." It features two prominent download buttons: "20.11.0 LTS Recommended For Most Users" and "21.6.0 Current Latest Features". Below these buttons are links for "Other Downloads", "Changelog", "API Docs", and "Other Downloads", "Changelog", "API Docs". A note at the bottom says "For information about supported releases, see the [release schedule](#)".

- Installer Node.js





- Installer npm sur la terminale

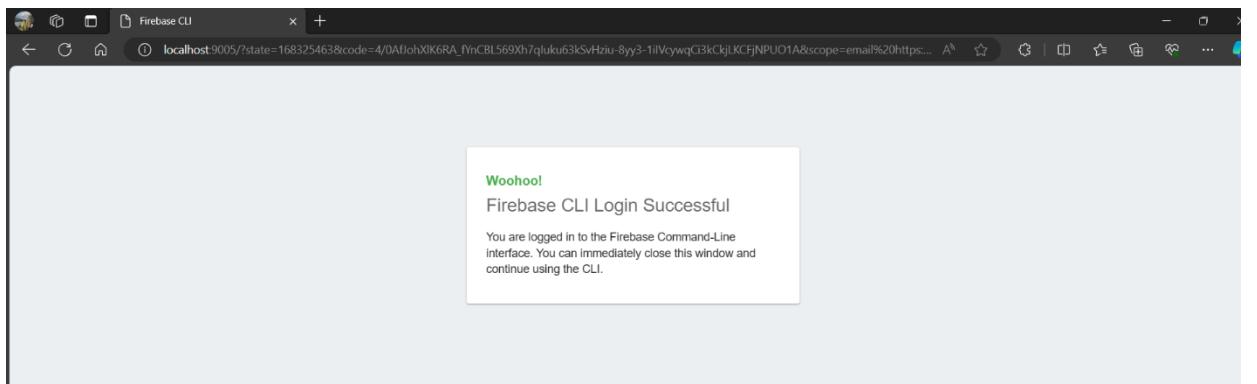
```
Invité de commandes
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\CARLINOT>npm install -g firebase-tools
added 639 packages in 60s

65 packages are looking for funding
  run 'npm fund' for details
npm notice New minor version of npm available! 10.2.4 => 10.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.3.0
npm notice Run npm install -g npm@10.3.0 to update!
npm notice

C:\Users\CARLINOT>
```

- Vérification firebase login



```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\CARLINOT>npm install -g firebase-tools
added 639 packages in 60s

65 packages are looking for funding
  run 'npm fund' for details
npm notice New minor version of npm available! 10.2.4 => 10.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.3.0
npm notice Run npm install -g npm@10.3.0 to update!
npm notice

C:\Users\CARLINOT>firebase login
Already logged in as nathancarlinot007@gmail.com

C:\Users\CARLINOT>
```

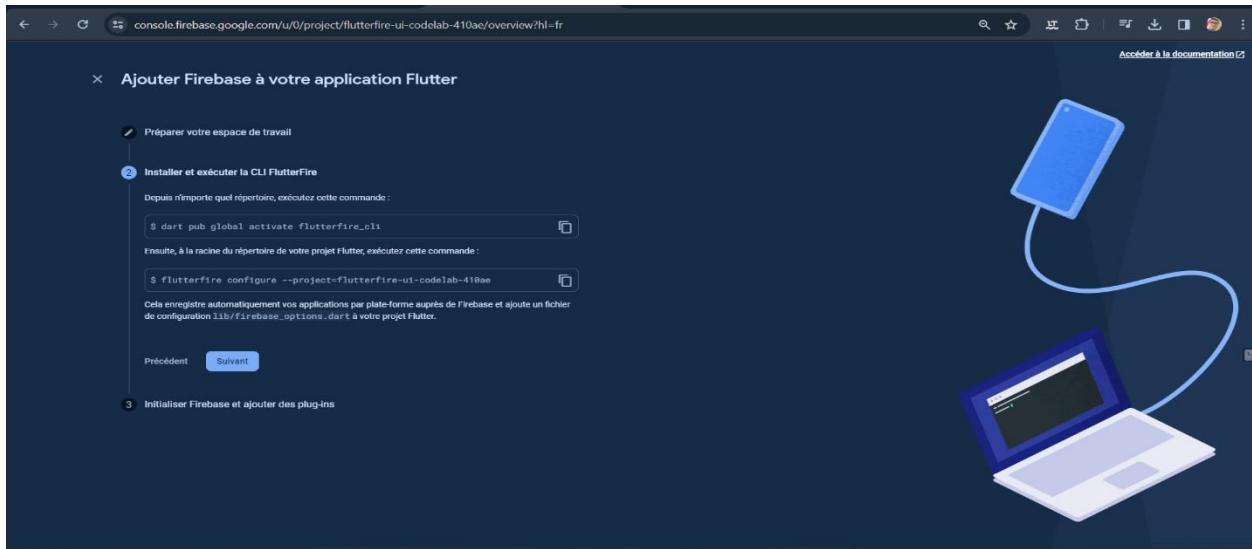
- Verification projet list

```
C:\WINDOWS\system32\cmd. x + . 
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.
C:\Users\CARLINOT>npm install -g firebase-tools
added 639 packages in 68s
65 packages are looking for funding
  run 'npm fund' for details
npm notice New minor version of npm available! 10.2.0 > 10.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.3.0
npm notice Run npm install -g npm@10.3.0 to update!
npm notice

C:\Users\CARLINOT>firebase login
Already logged in as nathancarlinot007@gmail.com
C:\Users\CARLINOT>firebase projects:list
/ Preparing the list of your Firebase projects
Project Display Name Project ID Project Number Resource Location ID
Firebase-Flutter-Codelab fir-flutter-codelab-3676e 574742714105 [Not specified]
FlutterFire-UI-Codelab flutterfire-ui-codelab-410ae 933614877584 [Not specified]

2 project(s) total.
C:\Users\CARLINOT>
```

- Ajouter Firebase à votre application Flutter



- Sur la terminale : dart pub global activate flutterfire\_cli

```
Invite de commandes - dart f x + . 
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.
C:\Users\CARLINOT>dart pub global activate flutterfire_cli
+ ansi_styles 0.3.2+ls... (1.9s)
+ args 2.1.1
+ astroid 1.1.0
+ boolean_selector 2.1.1
+ characters 1.3.0
+ ci 0.1.0
+ cli_util 0.3.5 (0.4.1 available)
+ clock 1.1.1
+ collection 1.18.0
+ dart_console 1.2.0
+ deep_pick 0.10.0 (1.0.0 available)
+ ffi 0.14.0 (0.13.0 available)
+ file 6.1.4 (7.0.0 available)
+ flutterfire_cli 0.2.7
+ http 0.13.6 (1.2.0 available)
+ http_parser 4.0.2
+ interact 2.2.0
+ intl 0.18.1 (0.19.0 available)
+ json_annotation 4.8.1
+ matcher 0.12.16+1
+ meta 1.1.6
+ path 1.9.0
+ petitparser 6.0.2
+ platform 3.1.4
+ process 4.2.4 (5.0.2 available)
+ pub_semver 2.1.4
+ pub_updater 0.2.4 (0.4.0 available)
+ pubspec 2.3.0
+ quiver 3.2.1
+ source_span 1.10.0
+ stack_trace 1.11.1
+ stream_channel 2.1.2
+ string_scanner 1.2.0
+ term_glyph 1.2.1
+ test_api 0.7.0
+ tint 2.0.1
```

```

file 6.1.4 (7.0.0 available)
+ flutterfire_cli 0.2.7
+ http 0.13.6 (1.2.0 available)
+ http_parser 4.0.2
+ interact 2.2.0
+ intl 0.18.1 (0.19.0 available)
+ json_annotation 4.8.1
+ matcher 0.12.16+1
+ meta 1.1.0
+ path 1.7.0
+ petitparser 6.0.2
+ platform 3.1.4
+ process 4.2.4 (5.0.2 available)
+ pub_semver 2.1.4
+ pub_updater 0.2.4 (0.4.0 available)
+ pubspec 2.3.0
+ quiver 3.2.1
+ source_span 1.10.0
+ stack_trace 1.11.1
+ stream_channel 2.1.2
+ string_scanner 1.2.0
+ testify_gyp 1.2.1
+ test_api 0.7.0
+ tint 2.0.1
+ typed_data 1.3.2
+ uri 1.0.0
+ win32 5.2.0
+ xml 6.5.0
+ yaml 3.1.2
Building package executables... (3.4s)
Built flutterfire_cli: flutterfire.
Installed executable flutterfire.
Activated flutterfire_cli 0.2.7.
C:\Users\CARLINOT|

```

- Ensuite, à la racine du répertoire de votre projet Flutter, exécutez cette commande :  
`flutterfire configure --project=flutterfire-ui-codelab-410ae`

```

File Edit Selection View Go Run Terminal Help < - > start
EXPLORER ... build.gradle x main.dart
START > dart_tool
> android
> .gradle
> app
build.gradle
google-services.json
> gradle
.gradleignore
build.gradle
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle
> assets
> build
> ios
> lib
app.dart
auth_gate.dart
firebase_options.dart
home.dart
main.dart
> macos
> web
  firebase
  flutter-plugins
  flutter-plugins-depend...
> OUTLINE
> TIMELINE
> DEPENDENCIES
PROBLEMS ① OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\La Famille Nathan\3emeannee\PM\flutter-codelabs.firebaseio-auth-flutterfire-ui\start> flutter upgrade
Flutter is already up to channel stable
Flutter 3.16.8 • channel stable • https://github.com/Flutter/flutter.git
Framework revision 07a909f (4 days ago) • 2024-01-16 16:22:29 -0800
Engines revision 62e2a6c6
Tools • Dart 3.2.5 • DevTools 2.28.5
No issues found! (run in 1.6s)
PS C:\La Famille Nathan\3emeannee\PM\flutter-codelabs.firebaseio-auth-flutterfire-ui\start> flutterfire configure --project=flutterfire-ui-codelab-410ae
i Found 2 Firebase projects. Selecting project flutterfire-ui-codelab-410ae.
✓ Merged configuration from your local project with the selected project to select? • web, macos, ios, android
i Firebase android app com.example.FlutterfireUiCodelab is not registered on Firebase project flutterfire-ui-codelab-410ae.
i Registered a new Firebase android app on Firebase project flutterfire-ui-codelab-410ae.
i Firebase ios app com.example.complete is not registered on Firebase project flutterfire-ui-codelab-410ae.
i Registered a new Firebase ios app on Firebase project flutterfire-ui-codelab-410ae.
i Firebase macos app com.example.complete.runnerTests is not registered on Firebase project flutterfire-ui-codelab-410ae.
i Registered a new Firebase macos app on Firebase project flutterfire-ui-codelab-410ae.
i Firebase web app complete (web) is not registered on Firebase project flutterfire-ui-codelab-410ae.
i Registered a new Firebase web app on Firebase project flutterfire-ui-codelab-410ae.
Generated FirebaseOptions file lib/firebase_options.dart already exists, do you want to override it? • yes
Firebase configuration file lib/firebase_options.dart generated successfully with the following Firebase apps:
Platform Firebase App Id
web 1:933614877584:web:73095fd2ad862ba35115
android 1:933614877584:com.google.firebase:70d46454:android:115
ios 1:933614877584:ios:02a7a1c06d5e9ff0aa5115
macos 1:933614877584:ios:b80ef1f40559ff512a35115
Learn more about using this file and next steps from the documentation:
  > https://firebase.google.com/docs/flutter/setup
PS C:\La Famille Nathan\3emeannee\PM\flutter-codelabs.firebaseio-auth-flutterfire-ui\start>

```

- Ajouter des packages Firebase à l'application Flutter
- `flutter pub add firebase_core`

```

File Edit Selection View Go Run Terminal Help < - > start
EXPLORER ... build.gradle firebase_options.dart x main.dart
START > dart_tool
> android
> assets
> build
> ios
> lib
app.dart
auth_gate.dart
firebase_options.dart
home.dart
main.dart
> macos
> web
  firebase
  flutter-plugins
  flutter-plugins-depend...
PROBLEMS ① OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\La Famille Nathan\3emeannee\PM\flutter-codelabs.firebaseio-auth-flutterfire-ui\start> flutter pub add firebase_core
"firebase_core" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
intl 0.18.1 (0.19.0 available)
js 0.6.7 (0.7.0 available)
matcher 0.12.16 (0.12.16+1 available)
material_color_utilities 0.5.0 (0.8.0 available)
meta 1.10.0 (1.11.0 available)
path 1.8.3 (1.9.0 available)
test_api 0.6.1 (0.7.0 available)
web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PM\flutter-codelabs.firebaseio-auth-flutterfire-ui\start>

```

- flutter pub add firebase\_auth

```

File Edit Selection View Go Run Terminal Help ⌘P start
EXPLORER build.gradle firebase_options.dart main.dart
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
"firebase_core" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.5.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PMM\flutter-codelabs\firebase-auth-flutterfire-ui>start> flutter pub add firebase_auth
"firebase_auth" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.5.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PMM\flutter-codelabs\firebase-auth-flutterfire-ui>start>

```

- flutter pub add firebase\_ui\_auth

```

File Edit Selection View Go Run Terminal Help ⌘P start
EXPLORER build.gradle firebase_options.dart main.dart
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
intl 0.18.1 (0.19.0 available)
js 0.6.7 (0.7.0 available)
matcher 0.12.16 (0.12.16+1 available)
material_color_utilities 0.5.0 (0.5.0 available)
meta 1.10.0 (1.11.0 available)
path 1.8.3 (1.9.0 available)
test_api 0.6.1 (0.7.0 available)
web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PMM\flutter-codelabs\firebase-auth-flutterfire-ui>start> flutter pub add firebase_auth
"firebase_auth" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.5.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PMM\flutter-codelabs\firebase-auth-flutterfire-ui>start> flutter pub add firebase_ui_auth
"firebase_ui_auth" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.5.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
PS C:\La Famille Nathan\3emeannee\PMM\flutter-codelabs\firebase-auth-flutterfire-ui>start>

```

## C- Écran de connexion

### 1- Départ

The screenshot shows the VS Code interface with the auth\_gate.dart file open in the editor. The code defines a StatelessWidget called AuthGate that returns a StreamBuilder. Inside the StreamBuilder, if there's no data, it returns a SignInScreen with providers: [EmailAuthProvider()]. If there is data, it returns a HomeScreen. The app is running on an SM-A336E device, displaying a simple sign-in screen with fields for Email and Password, and links for Register, Forgotten password, and Sign In.

```
import 'package:firebase_auth/firebase_auth.dart' hide EmailAuthProvider;
import 'package:firebase_ui_auth/firebase_ui_auth.dart';
import 'package:flutter/material.dart';
import 'home.dart';

class AuthGate extends StatelessWidget {
  const AuthGate({super.key});

  @override
  Widget build(BuildContext context) {
    return StreamBuilder<User>(
      stream: FirebaseAuth.instance.authStateChanges(),
      builder: (context, snapshot) {
        if (!snapshot.hasData) {
          return SignInScreen(
            providers: [
              EmailAuthProvider(),
            ],
          );
        }
        return const HomeScreen();
      },
    );
  }
}
```

### 2- Personnaliser l'écran de connexion

- En-têteBuilder

À l'aide de l'argument SignInScreen.headerBuilder , vous pouvez ajouter les widgets de votre choix au-dessus du formulaire de connexion.

The screenshot shows the VS Code interface with the auth\_gate.dart file open in the editor. The code is identical to the previous version, but the app is running on an SM-A336E device, displaying a sign-in screen with a large Flutter and Firebase logo at the top center, followed by the standard Email and Password fields, and the familiar Register, Forgotten password, and Sign In buttons.

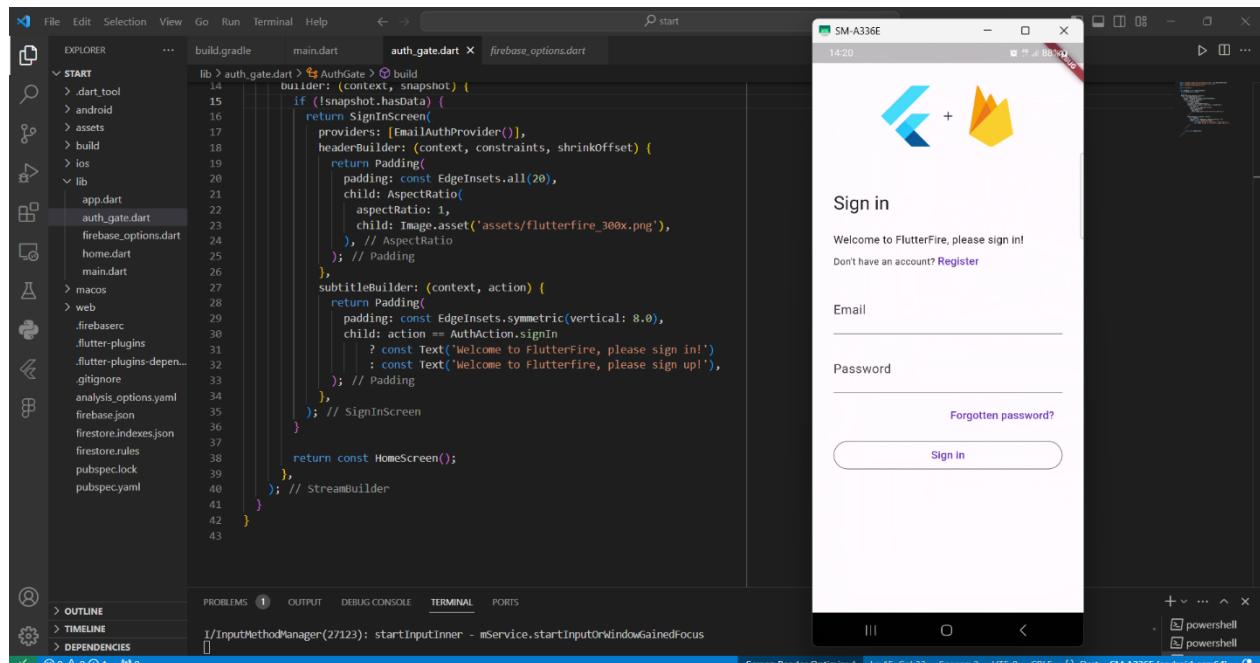
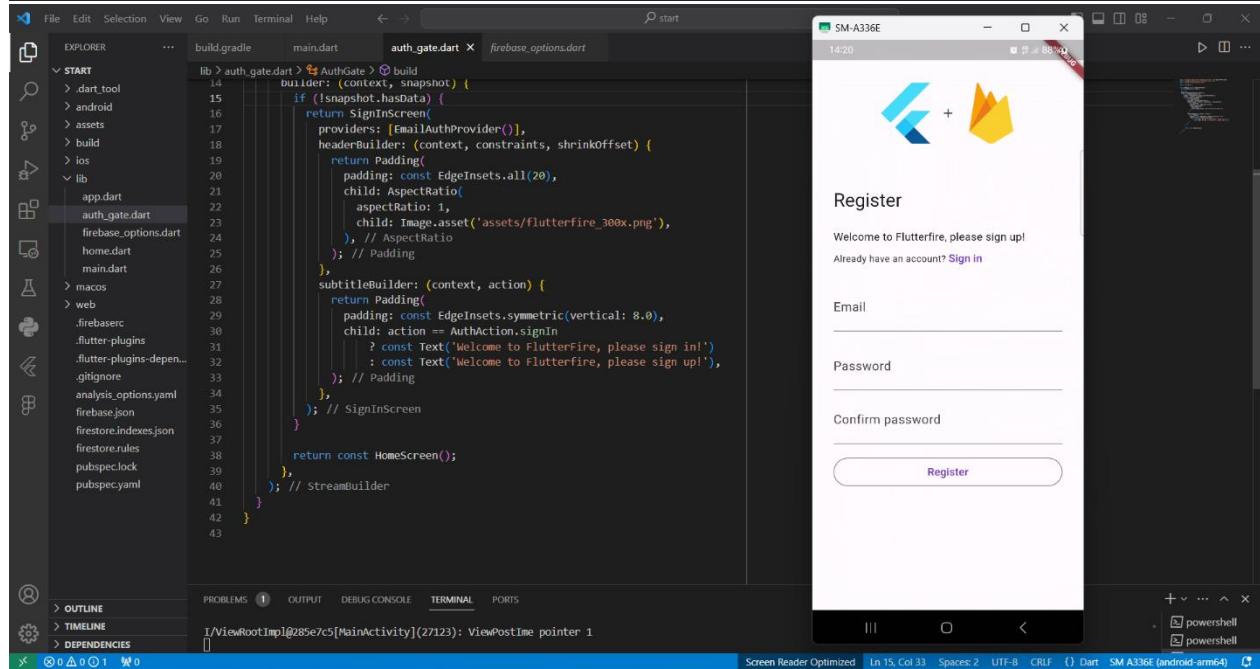
```
const AuthGate({super.key});

@override
Widget build(BuildContext context) {
  return StreamBuilder<User>(
    stream: FirebaseAuth.instance.authStateChanges(),
    builder: (context, snapshot) {
      if (!snapshot.hasData) {
        return SignInScreen(
          providers: [
            EmailAuthProvider(),
          ],
          headerBuilder: (context, constraints, shrinkOffset) {
            return Padding(
              padding: const EdgeInsets.all(20),
              child: AspectRatio(
                aspectRatio: 1,
                child: Image.asset('assets/flutterfire_300x.png'),
              ),
            );
          },
        );
      }
      return const HomeScreen();
    },
  );
}
```

- Générateur de sous-titres

L'écran de connexion expose trois paramètres supplémentaires qui vous permettent de personnaliser l'écran : subtitleBuilder , footerBuilder et sideBuilder .

Le subtitleBuilder est légèrement différent dans la mesure où les arguments de rappel incluent une action de type AuthAction . AuthAction est une énumération que vous pouvez utiliser pour détecter si l'écran sur lequel se trouve l'utilisateur est l'écran « connexion » ou l'écran « s'inscrire ».

```

File Edit Selection View Go Run Terminal Help < - > start
EXPLORER ... build.gradle main.dart auth_gate.dart firebase_options.dart
START
> _dart_tool
> android
> assets
> build
> ios
lib
> app.dart
auth_gate.dart
firebase_options.dart
home.dart
main.dart
> macos
> web
.firebaseio
.flutter-plugins
.flutter-plugins-depend...
.gitignore
analysis_options.yaml
firebase.json
firestore.indexes.json
firestore.rules
pubspec.lock
pubspec.yaml
14 builder: (context, snapshot) {
15   if (!snapshot.hasData) {
16     return SignInScreen(
17       providers: [EmailAuthProvider()],
18       headerBuilder: (context, constraints, shrinkOffset) {
19         return Padding(
20           padding: const EdgeInsets.all(20),
21           child: AspectRatio(
22             aspectRatio: 1,
23             child: Image.asset('assets/flutterfire_300x.png'),
24           ), // AspectRatio
25         ); // Padding
26       },
27       subtitleBuilder: (context, action) {
28         return Padding(
29           padding: const EdgeInsets.symmetric(vertical: 8.0),
30           child: action == AuthAction.signIn
31             ? const Text('Welcome to FlutterFire, please sign in!')
32             : const Text('Welcome to Flutterfire, please sign up!'),
33         ); // Padding
34       }, // SignInScreen
35     );
36   }
37   return const HomeScreen();
38 }
39 }, // StreamBuilder
40
41 }
42
43

```

PROBLEMS OUTLINE TIMELINE DEPENDENCIES

I/InputMethodManager(27123): startingInputInner - mService.startInputOrWindowGainedFocus

Screen Reader Optimized Ln 15, Col 33 Spaces: 2 UTF-8 CRLF () Dart SM A336E (android-arm64)

File Edit Selection View Go Run Terminal Help < - > start
EXPLORER ... build.gradle main.dart auth\_gate.dart firebase\_options.dart
START
> \_dart\_tool
> android
> assets
> build
> ios
lib
> app.dart
auth\_gate.dart
firebase\_options.dart
home.dart
main.dart
> macos
> web
.firebaseio
.flutter-plugins
.flutter-plugins-depend...
.gitignore
analysis\_options.yaml
firebase.json
firestore.indexes.json
firestore.rules
pubspec.lock
pubspec.yaml
14 builder: (context, snapshot) {
15 if (!snapshot.hasData) {
16 return SignInScreen(
17 providers: [EmailAuthProvider()],
18 headerBuilder: (context, constraints, shrinkOffset) {
19 return Padding(
20 padding: const EdgeInsets.all(20),
21 child: AspectRatio(
22 aspectRatio: 1,
23 child: Image.asset('assets/flutterfire\_300x.png'),
24 ), // AspectRatio
25 ); // Padding
26 },
27 subtitleBuilder: (context, action) {
28 return Padding(
29 padding: const EdgeInsets.symmetric(vertical: 8.0),
30 child: action == AuthAction.signIn
31 ? const Text('Welcome to FlutterFire, please sign in!')
32 : const Text('Welcome to Flutterfire, please sign up!'),
33 ); // Padding
34 }, // SignInScreen
35 );
36 }
37 return const HomeScreen();
38 }
39 }, // StreamBuilder
40
41 }
42
43

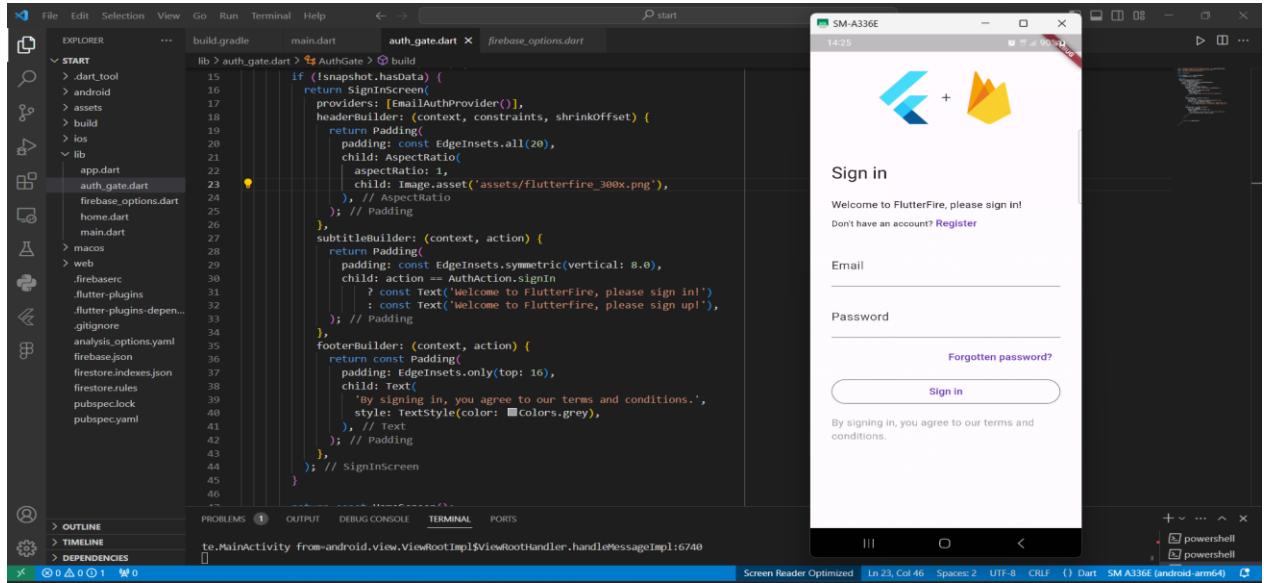
PROBLEMS OUTLINE TIMELINE DEPENDENCIES

I/ViewRootImpl@205e7c5[MainActivity](27123): ViewPostIme pointer 1

Screen Reader Optimized Ln 15, Col 33 Spaces: 2 UTF-8 CRLF () Dart SM A336E (android-arm64)

- Générateur de pied de page

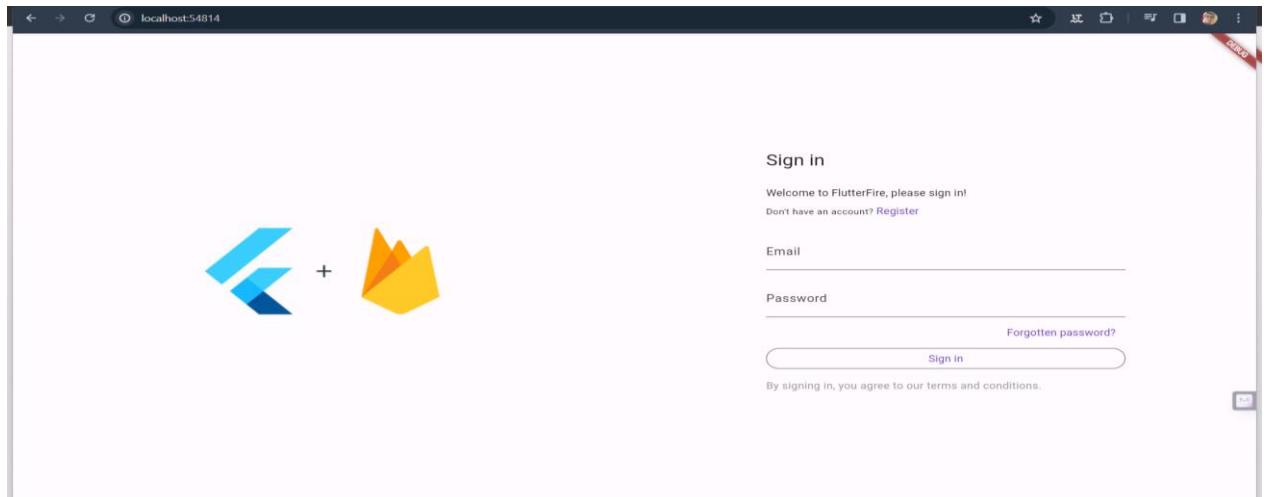
L'argument footerBuilder est le même que celui de subtitleBuilder. Il n'expose pas BoxConstraints ou shrinkOffset , car il est destiné au texte plutôt qu'aux images.



- Constructeur latéral

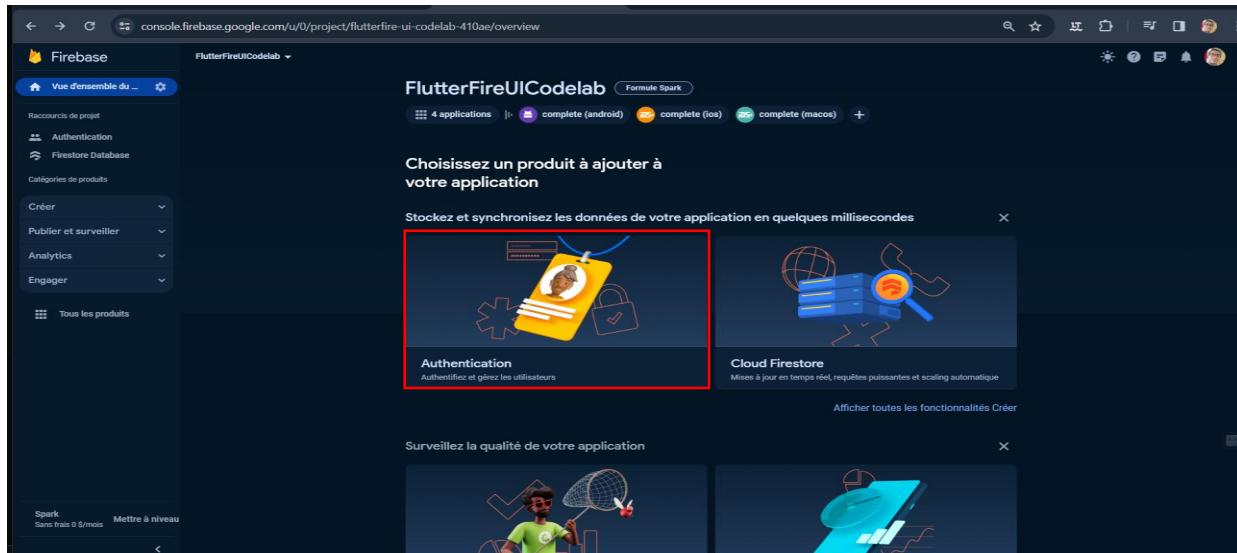
L'argument `SignInScreen.sidebuilder` accepte un rappel, et cette fois les arguments de ce rappel sont `BuildContext` et double `shrinkOffset` . Le widget renvoyé par `sideBuilder` sera affiché à gauche du formulaire de connexion, et uniquement sur des écrans larges. En fait, cela signifie que le widget ne sera affiché que sur les applications de bureau et Web.

En interne, l'interface utilisateur de FlutterFire utilise un point d'arrêt pour déterminer si le contenu de l'en-tête doit être affiché (sur des écrans grands, comme les mobiles) ou si le contenu latéral doit être affiché (sur des écrans larges, sur un ordinateur de bureau ou sur le Web). Plus précisément, si un écran fait plus de 800 pixels de large, le contenu du générateur latéral est affiché, mais pas le contenu de l'en-tête. Si l'écran fait moins de 800 pixels de large, c'est l'inverse.



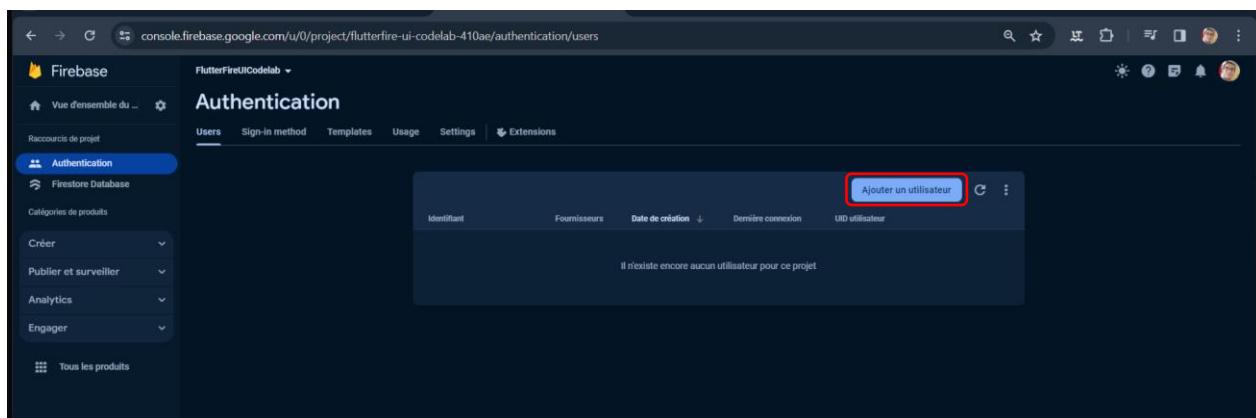
### 3- Créer un utilisateur

- Accédez au tableau "Utilisateurs" dans la console Firebase.



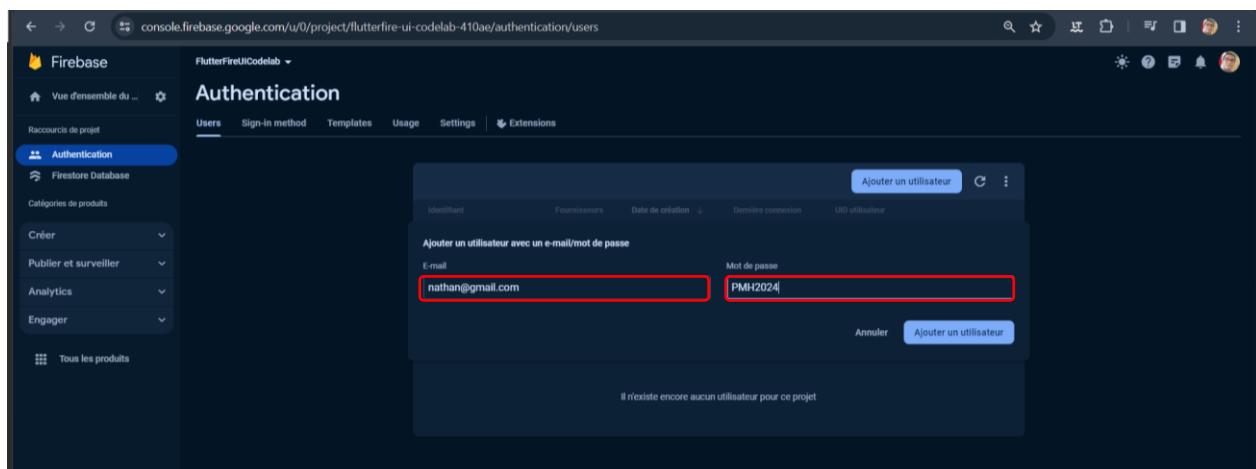
The screenshot shows the Firebase console interface. On the left, there's a sidebar with project resources like Authentication and Firestore Database. The main area is titled 'FlutterFireUICodeLab' and shows '4 applications' with status 'complete (android)', 'complete (ios)', and 'complete (macos)'. A central callout box says 'Choisissez un produit à ajouter à votre application' and lists 'Authentication' and 'Cloud Firestore'. The 'Authentication' section is highlighted with a red box. Below it, there's a section for monitoring app quality with a developer icon and a chart icon.

- Cliquez sur le bouton "Ajouter un utilisateur".



This screenshot shows the 'Authentication' section of the Firebase console. It has tabs for 'Users', 'Sign-in method', 'Templates', 'Usage', and 'Settings'. The 'Users' tab is active. A red box highlights the 'Ajouter un utilisateur' button at the top right of the user list table. The table header includes columns for 'Identifiant', 'Fournisseurs', 'Date de création', 'Dernière connexion', and 'UID utilisateur'. A message at the bottom of the table says 'Il n'existe encore aucun utilisateur pour ce projet'.

- Entrez une adresse e-mail et un mot de passe pour le nouvel utilisateur.



This screenshot shows the 'Ajouter un utilisateur avec un e-mail/mot de passe' dialog box. It has fields for 'E-mail' (containing 'nathan@gmail.com') and 'Mot de passe' (containing 'PMH2024'). A red box highlights the 'E-mail' field. At the bottom right of the dialog, there are 'Annuler' and 'Ajouter un utilisateur' buttons.

- Cliquez sur "Ajouter un utilisateur"

Ajouter un utilisateur avec un e-mail/mot de passe

E-mail: nathan@gmail.com

Mot de passe: PMH2024

Ajouter un utilisateur

Identifiant	Fournisseurs	Date de création	Dernière connexion	UID utilisateur
nathan@gmail.com		21 janv. 2024		p0PPpqdijEcne91cJEC4kz9R...

## D- Écran de profil

### 1- Ajouter un widget ProfileScreen

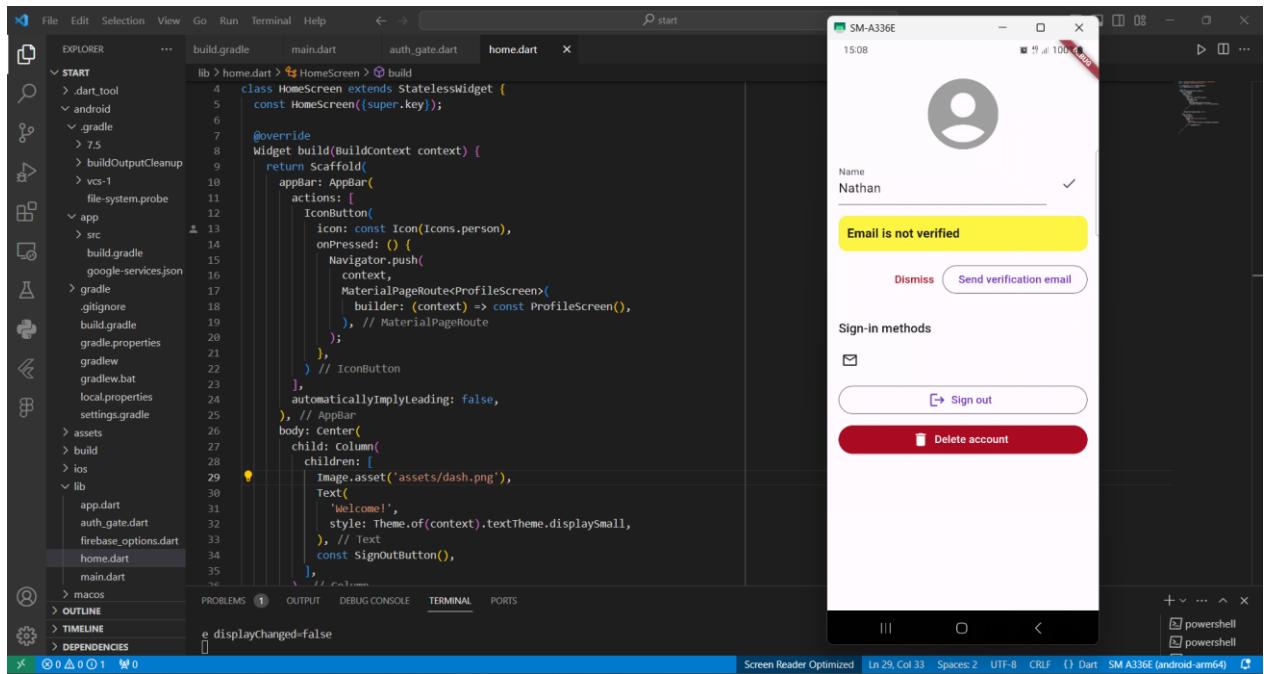
```

class HomeScreen extends StatelessWidget {
  const HomeScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        actions: [
          IconButton(
            icon: const Icon(Icons.person),
            onPressed: () {
              Navigator.push(
                context,
                MaterialPageRoute<ProfileScreen>(
                  builder: (context) => const ProfileScreen(),
                ),
              );
            },
          ),
        ],
        automaticallyImplyLeading: false,
      ),
      body: Center(
        child: Column(
          children: [
            Image.asset('assets/dash.png'),
            Text(
              'Welcome!',
              style: Theme.of(context).textTheme.displaySmall,
            ),
            const SignOutButton(),
          ],
        ),
      ),
    );
  }
}

```

Lorsque vous appuyez sur cet IconButton , votre application crée un nouvel itinéraire anonyme et y accède. Cet itinéraire affichera le widget ProfileScreen , qui est renvoyé par le rappel MaterialPageRoute.builder .



## 2- Déconnecter

Lorsque vous créez une instance de ProfileScreen , vous lui transmettez également une liste d'actions à l'argument ProfileScreen.actions . Ces actions sont du type FlutterFireUiAction . Il existe de nombreuses classes différentes qui sont des sous-types de FlutterFireUiAction et, en général, vous les utilisez pour indiquer à votre application de réagir aux différents changements d'état d'authentification. Le SignedOutAction appelle une fonction de rappel que vous lui donnez lorsque l'état d'authentification de Firebase passe à currentUser étant nul.

En ajoutant un rappel qui appelle Navigator.of(context).pop() lorsque SignedOutAction se déclenche, l'application accédera à la page précédente. Dans cet exemple d'application, il n'y a qu'un seul itinéraire permanent, qui affiche la page de connexion si aucun utilisateur n'est connecté, et la page d'accueil s'il y a un utilisateur. Étant donné que cela se produit lorsque l'utilisateur se déconnecte, l'application affichera la page de connexion.

The screenshot shows the VS Code interface with the 'home.dart' file open in the editor. The code defines a ProfileScreen with a 'SignedOutAction' and a 'SignOutButton'. The Android emulator on the right displays a user profile screen with a placeholder for 'Name', a yellow bar stating 'Email is not verified', and a 'Sign out' button highlighted with a blue border.

```

lib > home.dart > ↵ HomeScreen > ↵ build
      builder: (context) => ProfileScreen(
        actions: [
          SignedOutAction(context),
          Navigator.of(context).pop(),
        ],
      ),
    ),
  ),
),
automaticallyImplyLeading: false,
), // AppBar
body: center(
  child: column(
    children: [
      Image.asset('assets/dash.png'),
      Text(
        'Welcome!',
        style: Theme.of(context).textTheme.displaySmall,
      ),
      const SignOutButton(),
    ],
  ),
), // Column
), // Center
); // Scaffold
}

```

### 3- Ajouter des enfants à l'écran de profil

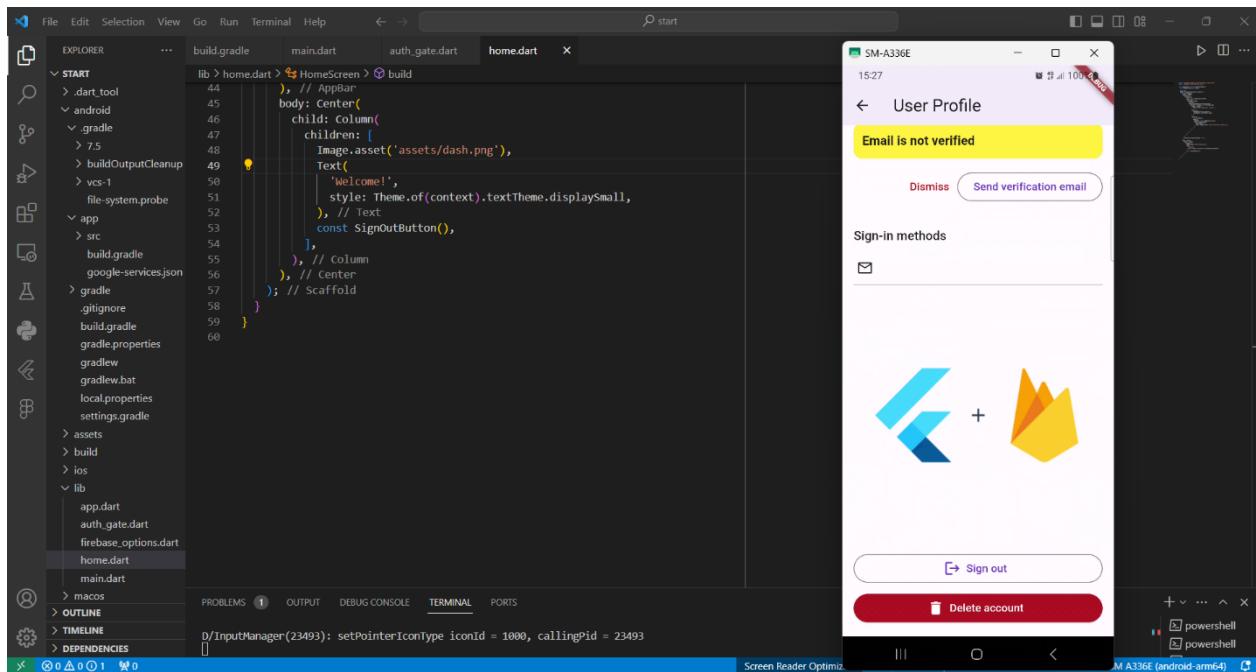
Le widget `ProfileScreen` possède également un argument facultatif nommé `enfants`. Cet argument accepte une liste de widgets, et ces widgets seront placés verticalement à l'intérieur d'un widget `Column` déjà utilisé en interne pour créer le `ProfileScreen`. Ce widget `Colonne` dans la méthode de construction `ProfileScreen` placera les enfants que vous lui transmettrez au-dessus du bouton « Déconnexion ».

The screenshot shows the VS Code interface with the 'home.dart' file open in the editor. The code now includes a 'children' list within the 'Column' of the 'ProfileScreen' constructor, containing a 'Text' widget with the string 'Welcome!' and a 'SignOutButton'. The Android emulator on the right displays a user profile screen with a placeholder for 'Name', a yellow bar stating 'Email is not verified', and two new icons at the bottom: a blue Flutter logo and a yellow Firebase logo.

```

lib > home.dart > ↵ HomeScreen > ↵ build
      builder: (context) => ProfileScreen(
        actions: [
          SignedOutAction(context),
          Navigator.of(context).pop(),
        ],
      ),
    ),
  ),
),
automaticallyImplyLeading: false,
), // AppBar
body: center(
  child: column(
    children: [
      Image.asset('assets/dash.png'),
      Text(
        'Welcome!',
        style: Theme.of(context).textTheme.displaySmall,
      ),
      const SignOutButton(),
    ],
  ),
), // Column
), // Center
); // Scaffold
}

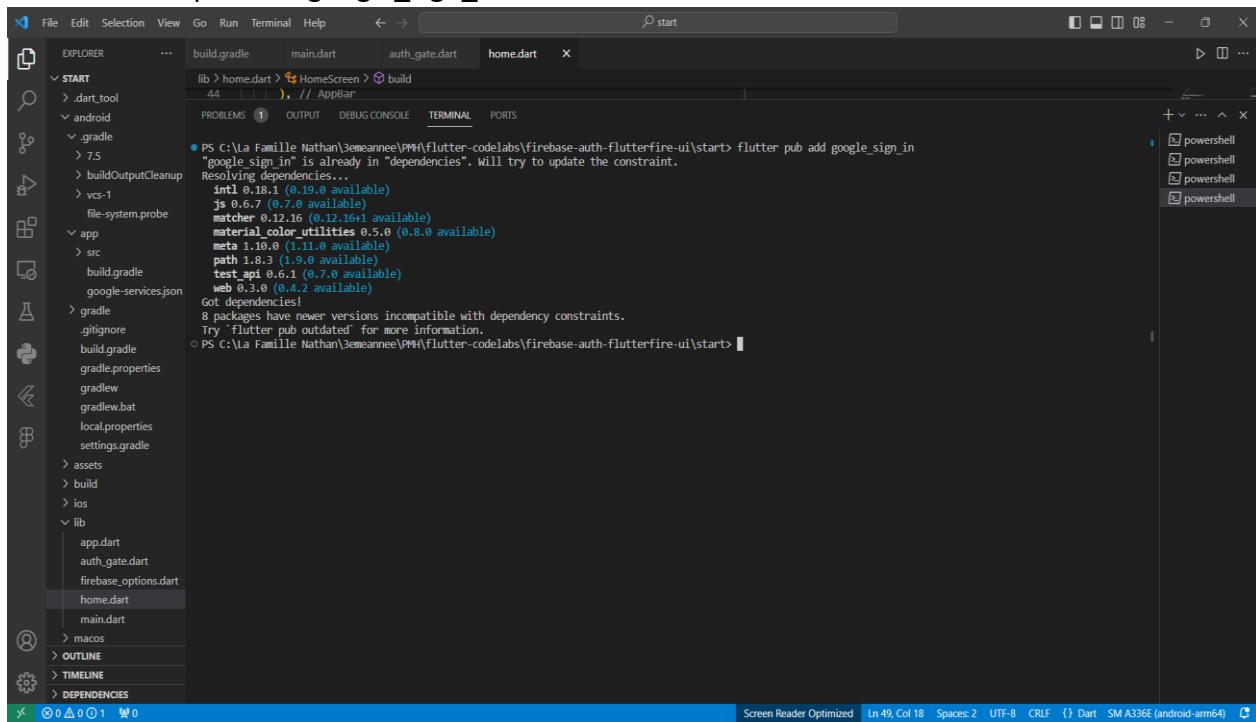
```



## E- Connexion multiplateforme Google Auth

Pour intégrer l'authentification Google, installez le plugin officiel `firebase_ui_oauth_google` et ses dépendances, qui géreront le flux d'authentification natif. Dans le terminal, accédez à la racine de votre projet Flutter et entrez la commande suivante :

- `flutter pub add google_sign_in`



- flutter pub add firebase\_ui\_oauth\_google

```

PS C:\La Famille Nathan\3emeannee\PMH\flutter-codelabs\firebase-auth-flutterfire-ui\start> flutter pub add google_sign_in
"google_sign_in" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.8.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)

Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try flutter pub outdated for more information.

PS C:\La Famille Nathan\3emeannee\PMH\flutter-codelabs\firebase-auth-flutterfire-ui\start> flutter pub add firebase_ui_oauth_google
"firebase ui oauth google" is already in "dependencies". Will try to update the constraint.
Resolving dependencies...
  intl 0.18.1 (0.19.0 available)
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.8.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
  web 0.3.0 (0.4.2 available)

Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try flutter pub outdated for more information.

PS C:\La Famille Nathan\3emeannee\PMH\flutter-codelabs\firebase-auth-flutterfire-ui\start>

```

## 1- Accédez au tableau "Utilisateurs" dans la console Firebase.

Choisissez un produit à ajouter à votre application

Stockez et synchronisez les données de votre application en quelques millisecondes

**Authentication**  
Authentifiez et gérez les utilisateurs

Cloud Firestore  
Mises à jour en temps réel, requêtes puissantes et scaling automatique

Surveillez la qualité de votre application

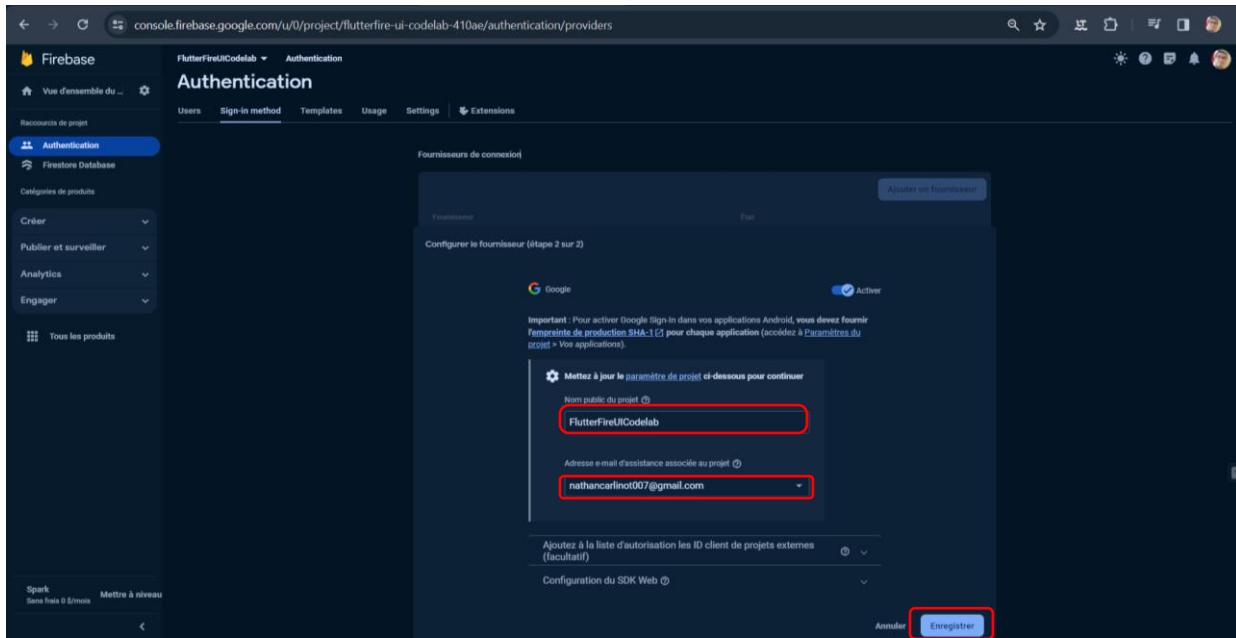
- Cliquez sur "Ajouter un nouveau fournisseur".

The screenshot shows the Firebase Authentication interface. On the left, there's a sidebar with project settings like 'Créer', 'Publier et surveiller', 'Analytics', and 'Engager'. The main area is titled 'Authentication' and has tabs for 'Users', 'Sign-in method', 'Templates', 'Usage', 'Settings', and 'Extensions'. Under 'Fournisseurs de connexion', there's a table with one row: 'Fournisseur' (Adresse e-mail/Mot de passe), 'État' (Activé), and a red box around the 'Ajouter un fournisseur' button. Below this is a section titled 'Options avancées' with a sub-section 'Authentification multifacteur par SMS' containing a note about Identity Platform.

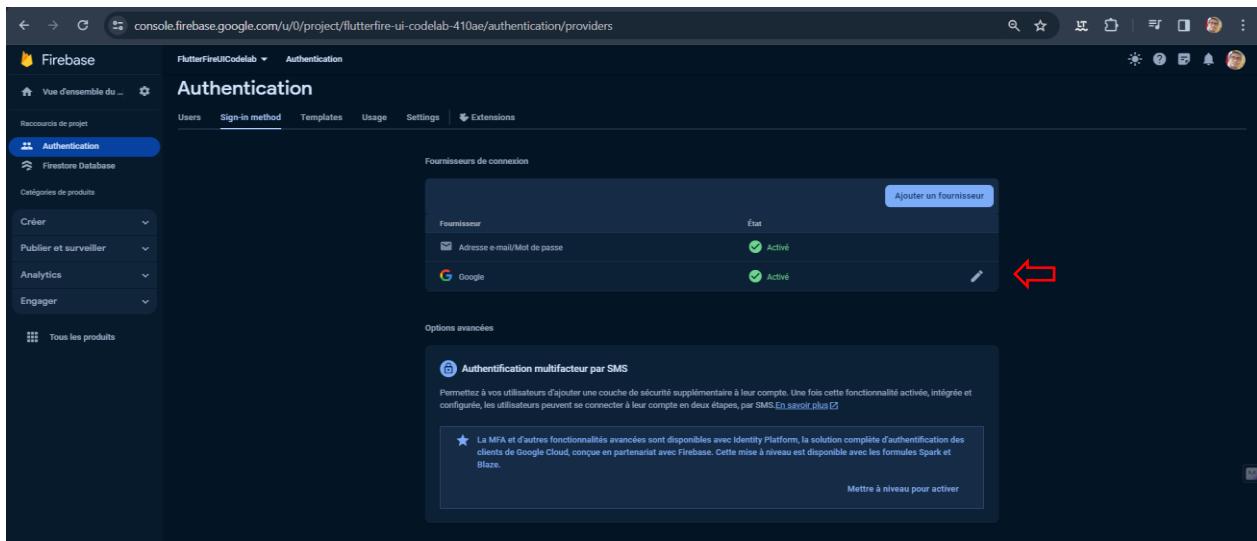
- Sélectionnez "Google".

This screenshot shows the 'Sélectionner un fournisseur de connexion (étape 1 sur 2)' dialog. It lists several providers under 'Autres fournisseurs': Google (highlighted with a red box), Facebook, Play Jeux, Game Center, Apple, GitHub, Microsoft, Twitter, and Yahoo!. There are also sections for 'Fournisseurs natifs' (Adresse e-mail/Mot de passe, Téléphone, Anonyme) and 'Fournisseurs personnalisés' (OpenID Connect, SAML).

- Basculez le commutateur intitulé « Activer » et appuyez sur « Enregistrer »

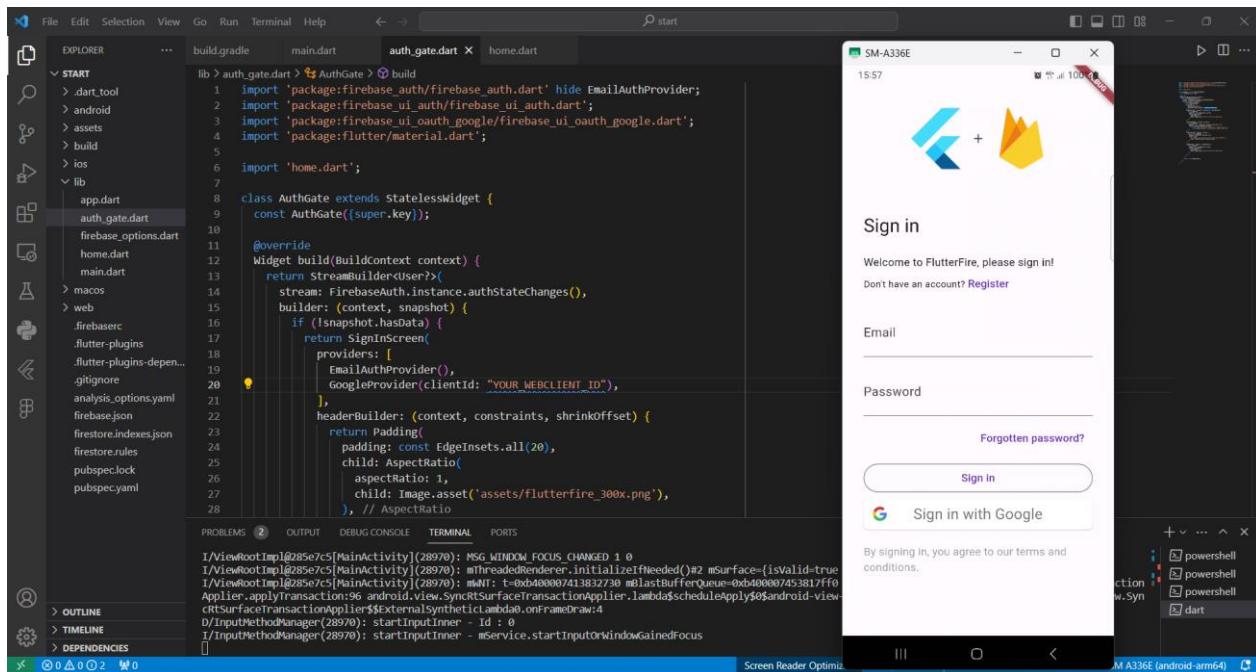


- Confirmez que le fournisseur de connexion Google a été ajouté.



## 2- Ajouter un bouton de connexion Google

Avec la connexion Google activée, ajoutez le widget nécessaire pour afficher un bouton de connexion Google stylisé sur la page de connexion.



## 3- Configurer le bouton de connexion

- Cliquez sur le fournisseur Google.

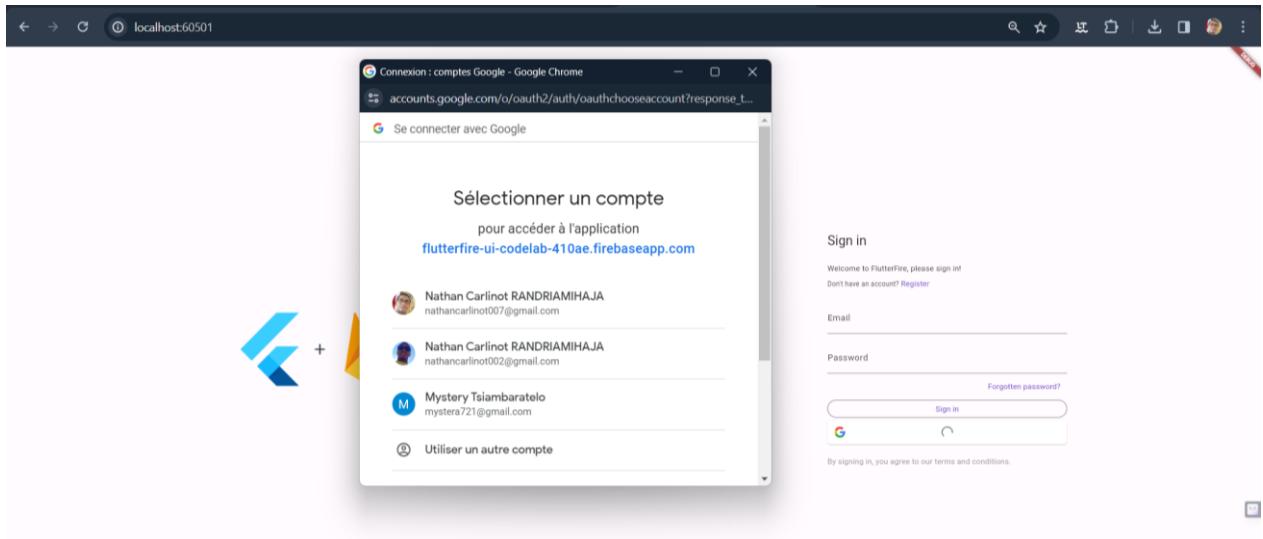
The screenshot shows the Firebase Authentication console in a web browser. The left sidebar has "Authentication" selected under "FlutterFireUICodeLab". The main "Authentication" tab is active, showing the "Sign-in method" section. Under "Fournisseurs de connexion" (Providers), there are two entries: "Adresse e-mail/Mot de passe" (Email/Password) and "Google". The "Google" entry has a red circle with a white edit icon drawn over it, indicating it is the target for configuration. Below the providers, a section titled "Authentification multifactor par SMS" (Multi-factor authentication via SMS) is visible, containing a note about Identity Platform and a "Mettre à niveau pour activer" (Update to activate) button.

- Cliquez sur le panneau d'extension "Configuration du SDK Web".

The screenshot shows the Firebase console's Authentication providers section. Under the Google provider, there is a 'Configuration du SDK Web' button highlighted with a red arrow.

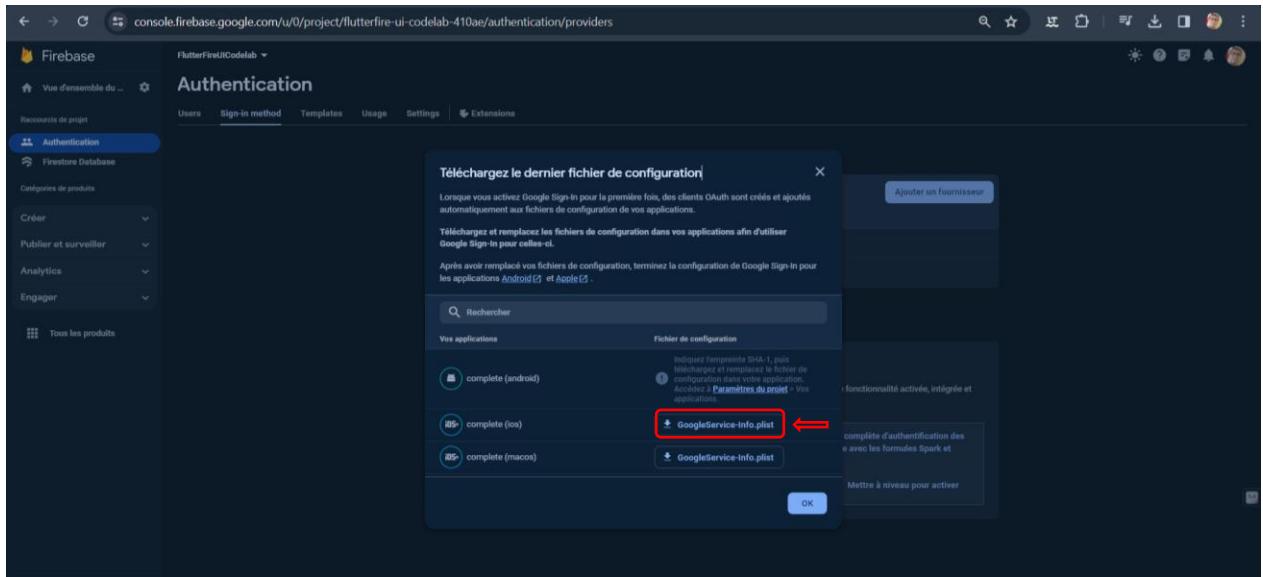
- Copiez la valeur de « ID client Web »

The screenshot shows the same configuration page as before, but now the 'ID client Web' field is highlighted with a red box. The value '933614877584-0rvcklo7561bk64jvgctpu6ghq42v.apps.googleusercontent.com' is visible in the field.



#### 4- Configurer iOS

- Cliquez sur le bouton "GoogleServices-Info.plist" pour télécharger le fichier de configuration nécessaire.



- Faites glisser et déposez le fichier téléchargé dans le répertoire appelé . /ios/Runner dans votre projet Flutter.

➔ Sélectionnez GoogleService-Info.plist dans le gestionnaire de fichiers.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<key>CLIENT_ID</key>
<string>933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0.apps.googleusercontent.com</string>
<key>REVERSED_CLIENT_ID</key>
<string>com.googleusercontent.apps.933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0</string>
<key>GCM_SENDER_ID</key>
<string>933614877584</string>
<key>PLIST_VERSION</key>
<string>1</string>
<key>BUNDLE_ID</key>
<string>com.example.complete</string>
<key>PROJECT_ID</key>
<string>flutterfire-ui-codelab-410ae</string>
<key>STORAGE_BUCKET</key>
<string>flutterfire-ui-codelab-410ae.appspot.com</string>
<key>IS_IOS_ENABLED</key>
<false></false>
<key>IS_ANALYTICS_ENABLED</key>
<false></false>
<key>IS_APPINVITE_ENABLED</key>
<true></true>
<key>IS_GCM_ENABLED</key>
<true></true>
<key>IS_SIGNIN_ENABLED</key>
<true></true>
<key>GOOGLE_APP_ID</key>
<string>1:933614877584:ios:07a61c067d5a9f0aa35115</string>
<key>CFBundleURLTypes</key>
<array>
<dict>
<key>CFBundleTypeRole</key>
<string>Editor</string>
<key>CFBundleURLSchemes</key>
<array>
<string>com.googleusercontent.apps.933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0</string>
</array>
</dict>
</array>
</plist>
```

➔ Ajoutez les attributs CFBundleURLTypes dans le fichier /ios/Runner/Info.plist.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<key>GCM_SENDER_ID</key>
<string>933614877584</string>
<key>PLIST_VERSION</key>
<string>1</string>
<key>BUNDLE_ID</key>
<string>com.example.complete</string>
<key>PROJECT_ID</key>
<string>flutterfire-ui-codelab-410ae</string>
<key>STORAGE_BUCKET</key>
<string>flutterfire-ui-codelab-410ae.appspot.com</string>
<key>IS_IOS_ENABLED</key>
<false></false>
<key>IS_ANALYTICS_ENABLED</key>
<false></false>
<key>IS_APPINVITE_ENABLED</key>
<true></true>
<key>IS_GCM_ENABLED</key>
<true></true>
<key>IS_SIGNIN_ENABLED</key>
<true></true>
<key>GOOGLE_APP_ID</key>
<string>1:933614877584:ios:07a61c067d5a9f0aa35115</string>
<key>CFBundleURLTypes</key>
<array>
<dict>
<key>CFBundleTypeRole</key>
<string>Editor</string>
<key>CFBundleURLSchemes</key>
<array>
<string>com.googleusercontent.apps.933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0</string>
</array>
</dict>
</array>
</plist>
```

Avec mon REVERSED\_CLIENT\_ID: [com.googleusercontent.apps.933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0](https://com.googleusercontent.apps.933614877584-9o267rqgj65ohs40p7ljjt6komuq4fr0)

## Task 3

A- Téléchargez la version initiale du projet depuis GitHub :

1- Clone GitHub

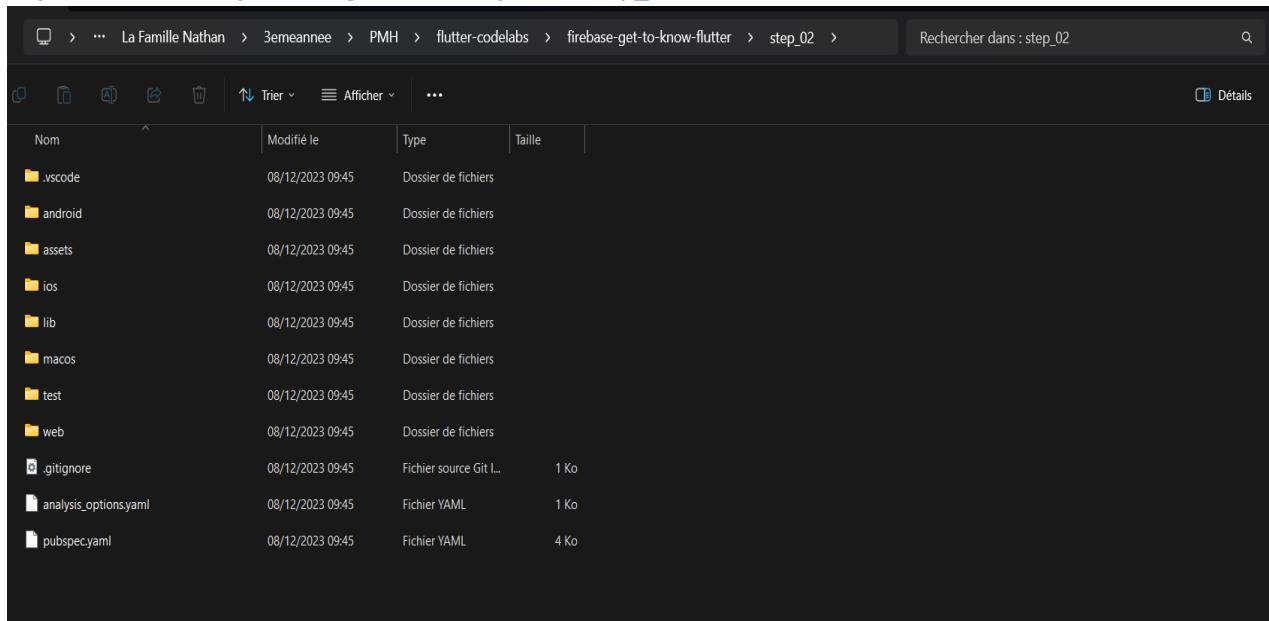
Depuis la ligne de commande, clonez le dépôt GitHub dans le répertoire flutter-codelabs :

`git clone https://github.com/flutter/codelabs.git flutter-codelabs`

2- Recherche fichier

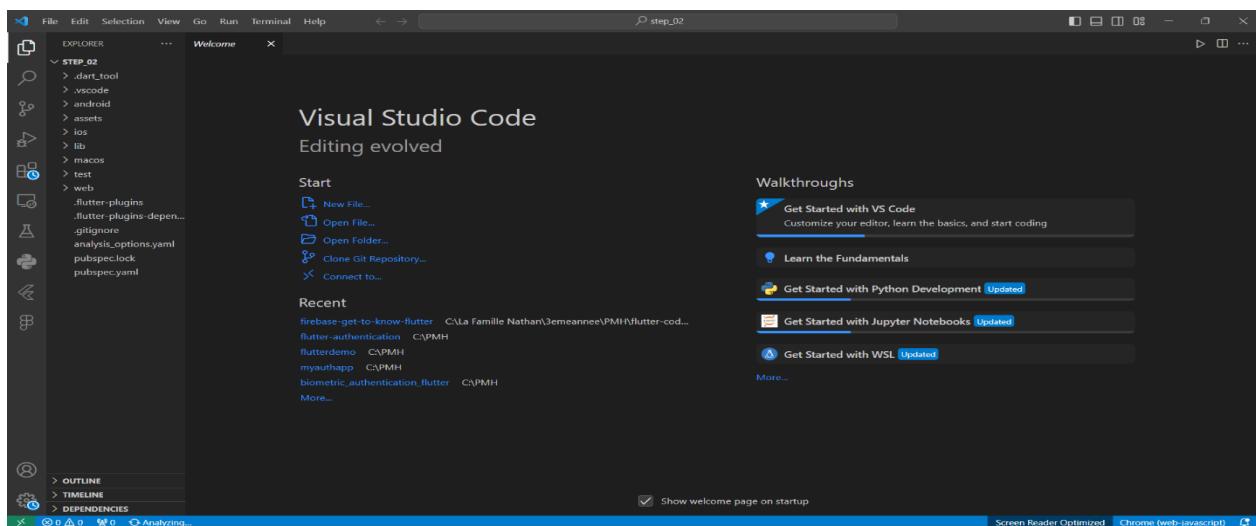
Recherchez les fichiers correspondants pour la deuxième étape :

`cd flutter-codelabs/firebase-get-to-know-flutter/step_02`



3- Ouverture repertoire

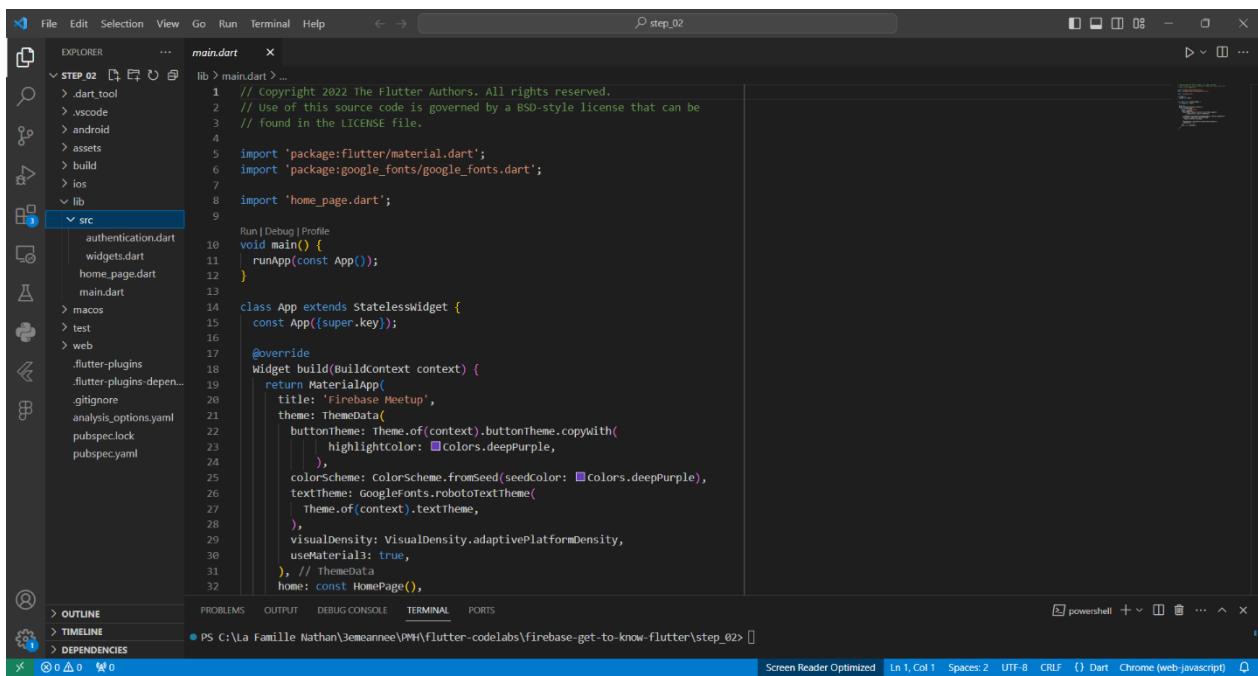
Ouvrir le répertoire `flutter-codelabs/firebase-get-to-know-flutter/step_02` dans votre IDE préféré. Ce répertoire contient le code de démarrage de l'atelier de programmation, qui consiste en une application de rencontre Flutter non encore fonctionnelle.



#### 4- Localiser les fichiers qui nécessitent du travail

Le code de cette application est réparti sur plusieurs répertoires. Cette répartition des fonctionnalités facilite le travail car elle regroupe le code par fonctionnalité.

- Recherchez les fichiers suivants :
  - ❖ lib/main.dart : Ce fichier contient le point d'entrée principal et le widget de l'application.
  - ❖ lib/home\_page.dart : Ce fichier contient le widget de la page d'accueil.
  - ❖ lib/src/widgets.dart : Ce fichier contient une poignée de widgets pour aider à standardiser le style de l'application. Ils composent l'écran de l'application de démarrage.
  - ❖ lib/src/authentication.dart : ce fichier contient une implémentation partielle de l'authentification avec un ensemble de widgets pour créer une expérience utilisateur de connexion pour l'authentification par courrier électronique Firebase. Ces widgets pour le flux d'authentification ne sont pas encore utilisés dans l'application de démarrage, mais vous les ajouterez bientôt.



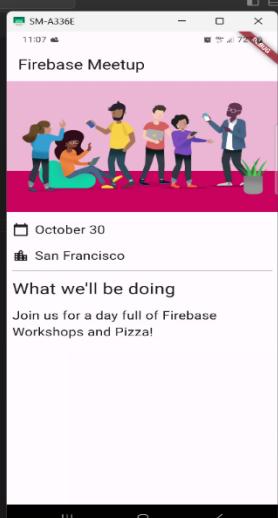
```
// Copyright 2022 The Flutter Authors. All rights reserved.  
// Use of this source code is governed by a BSD-style license that can be  
// found in the LICENSE file.  
  
import 'package:flutter/material.dart';  
import 'package:google_fonts/google_fonts.dart';  
  
import 'home_page.dart';  
  
void main() {  
  runApp(const App());  
}  
  
class App extends StatelessWidget {  
  const App({super.key});  
  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'Firebase Meetup',  
      theme: ThemeData(  
        buttonTheme: Theme.of(context).buttonTheme.copyWith(  
          highlightColor: Colors.deepPurple,  
        ),  
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),  
        textTheme: GoogleFonts.robotoTextTheme(  
          Theme.of(context).textTheme,  
        ),  
        visualDensity: VisualDensity.adaptivePlatformDensity,  
        useMaterial3: true,  
      ), // ThemeData  
      home: const HomePage(),  
    );  
  }  
}
```

#### 5- Consulter le fichier lib/main.dart

Cette application tire avantage du package google\_fonts pour établir Roboto comme police par défaut dans l'ensemble de l'interface. Vous avez la possibilité d'explorer différentes polices sur fonts.google.com et de les intégrer dans diverses parties de l'application.

L'utilisation des widgets d'assistance présents dans le fichier lib/src/widgets.dart, tels que Header, Paragraph et IconAndDetail, permet d'éliminer tout code redondant, contribuant ainsi à réduire la complexité de la mise en page décrite dans la HomePage. Cette approche favorise également une apparence et une expérience utilisateur cohérentes à travers l'ensemble de l'application.

## Affichage

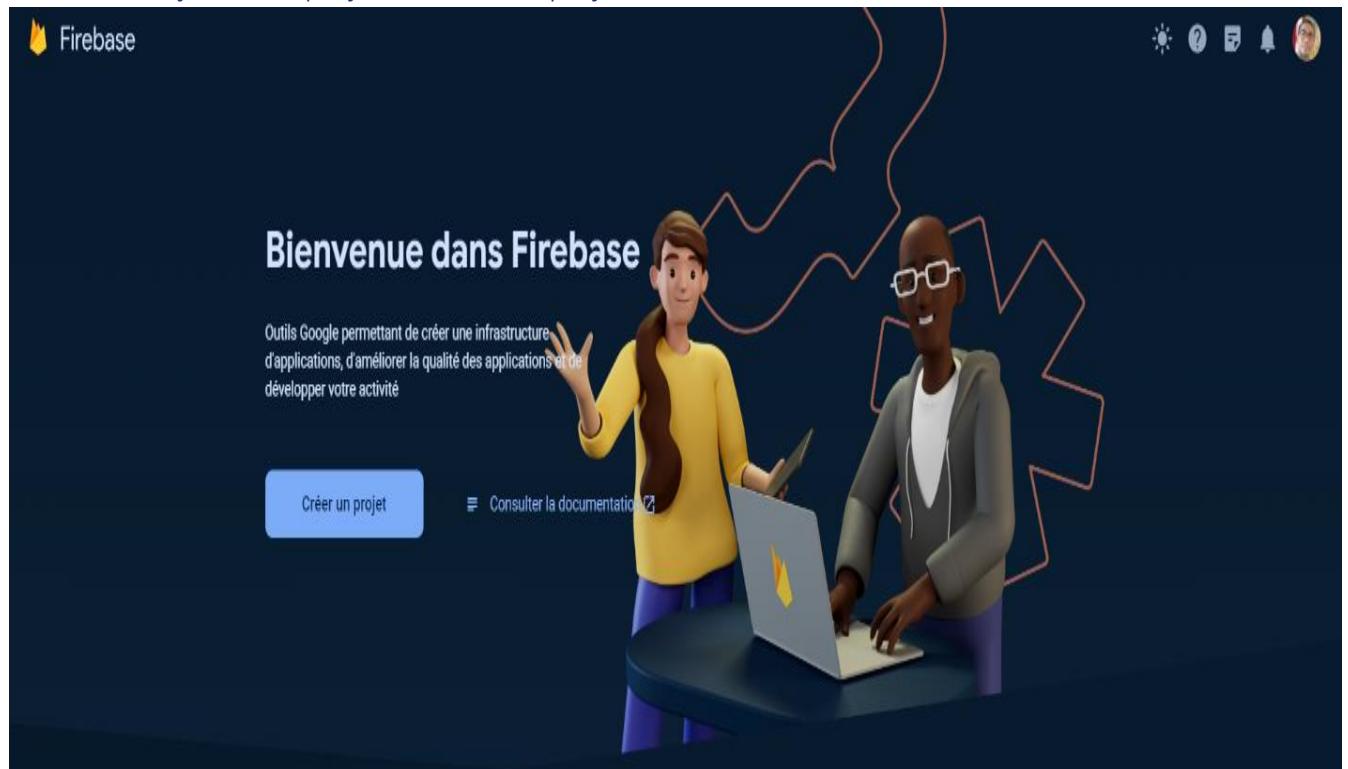


The screenshot shows a Flutter development environment in VS Code. The code editor displays the file `authentication.dart` under the `STEP_02` folder. The code implements a `AuthFunc` widget that checks if the user is logged in and provides a sign-in or sign-out button. The `build` method returns a `Row` with two `StyleButton`s. The `onPressed` function for the sign-in button pushes a route, while the sign-out button calls a `signout()` function.

```
lib > src > authentication.dart > AuthFunc > AuthFunc
1 // Copyright 2022 The Flutter Authors. All rights reserved.
2 // Use of this source code is governed by a BSD-style license that can be
3 // found in the LICENSE file.
4
5 import 'package:flutter/material.dart';
6 import 'package:go_router/go_router.dart';
7
8 import 'widgets.dart';
9
10 class AuthFunc extends StatelessWidget {
11   const AuthFunc({
12     super.key,
13     required this.loggedin,
14     required this.signout,
15   });
16
17 final bool loggedin;
18 final void Function() signout;
19
20 @override
21 Widget build(BuildContext context) {
22   return Row(
23     children: [
24       Padding(
25         padding: const EdgeInsets.only(left: 24, bottom: 8),
26         child: StyleButton(
27           onPressed: () {
28             !loggedin ? context.push('/sign-in') : signout();
29           },
30           child: !loggedin ? const Text('RSVP') : const Text('Logout'),
31         ),
32       ),
33       Visibility(
34         visible: loggedin,
35         child: Padding(
36           padding: const EdgeInsets.only(left: 24, bottom: 8),
37           child: StyleButton(
38             onPressed: () {
39               signout();
40             },
41             child: const Text('Logout'),
42           ),
43         ),
44       ),
45     ],
46   );
47 }
48
49 
```

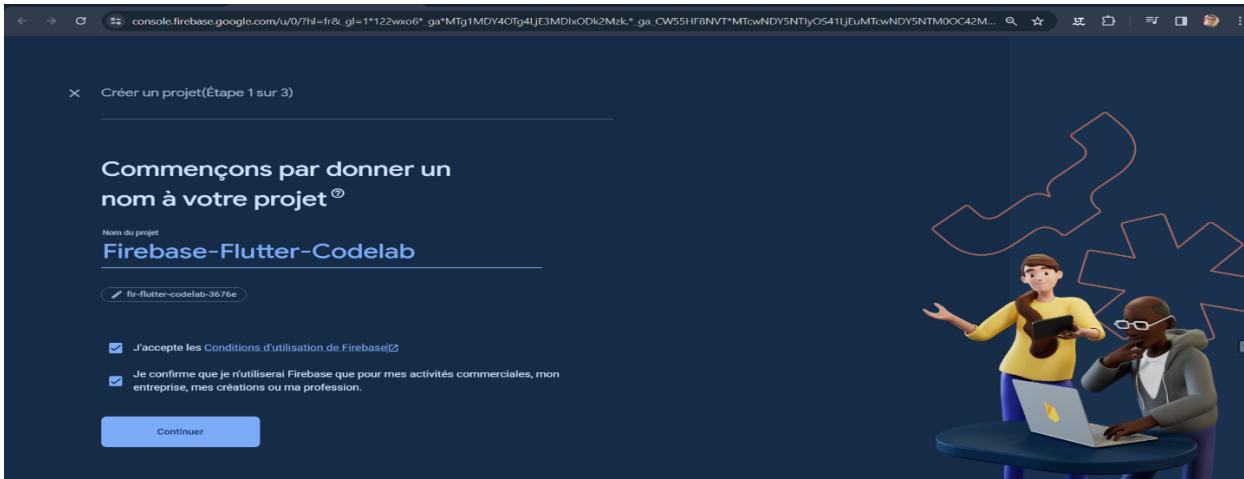
## B- Créer et configurer un projet Firebase

- 1- Créer un projet Firebase
- 2- Ajouter un projet ou Créeer un projet .



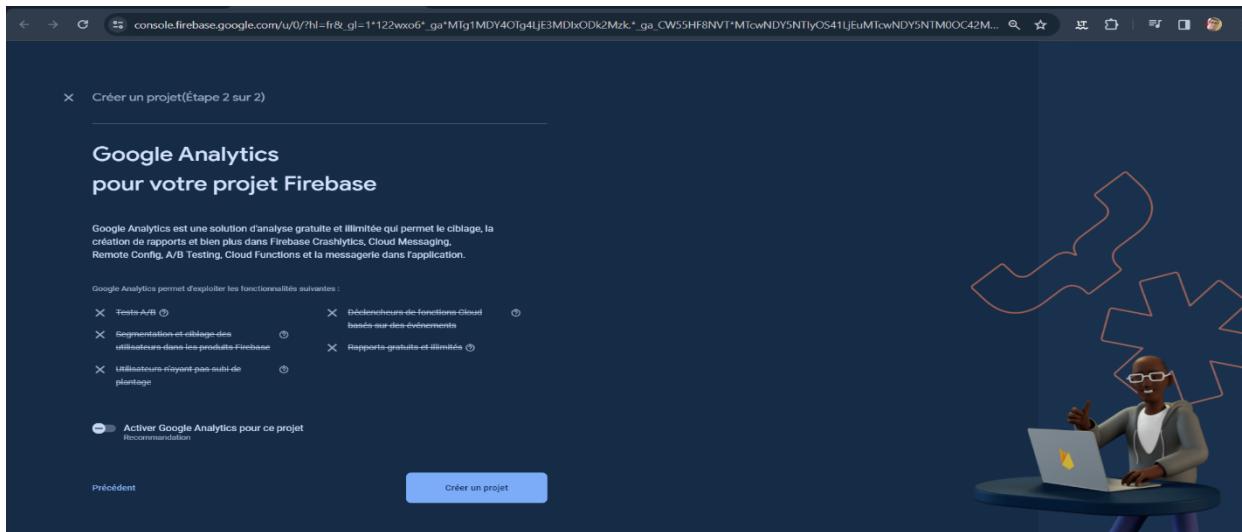
### 3- Nom du projet

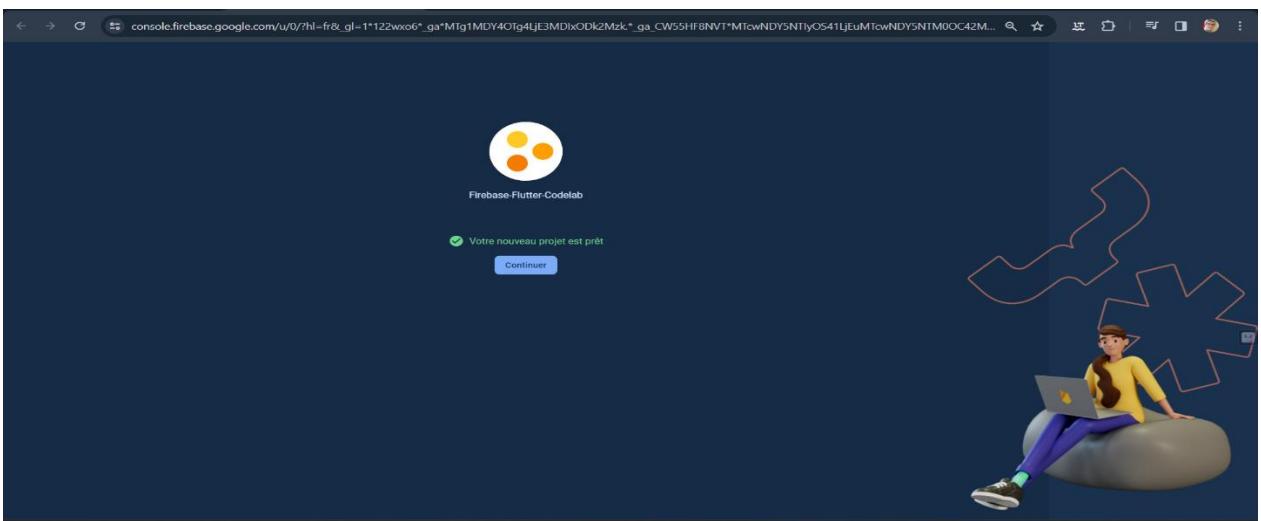
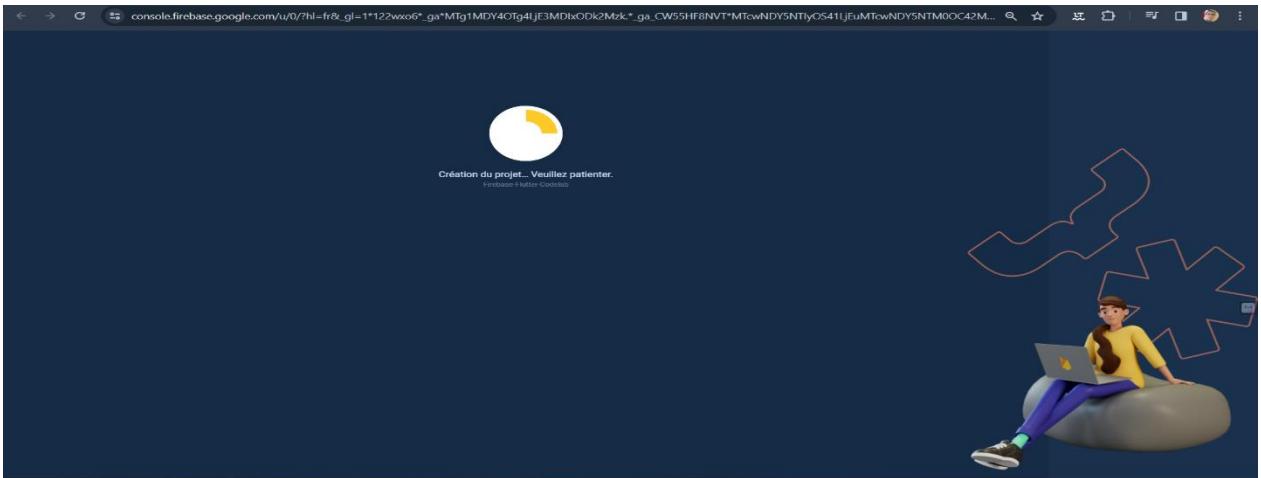
Dans le champ Nom du projet , saisissez Firebase-Flutter-Codelab , puis cliquez sur Continuer .



### 4- Google Analytics

Cliquez sur les options de création de projet. Si vous y êtes invité, acceptez les conditions de Firebase, mais ignorez la configuration de Google Analytics, car vous ne l'utiliserez pas pour cette application.





## C- Activer l'authentification de connexion par e-mail

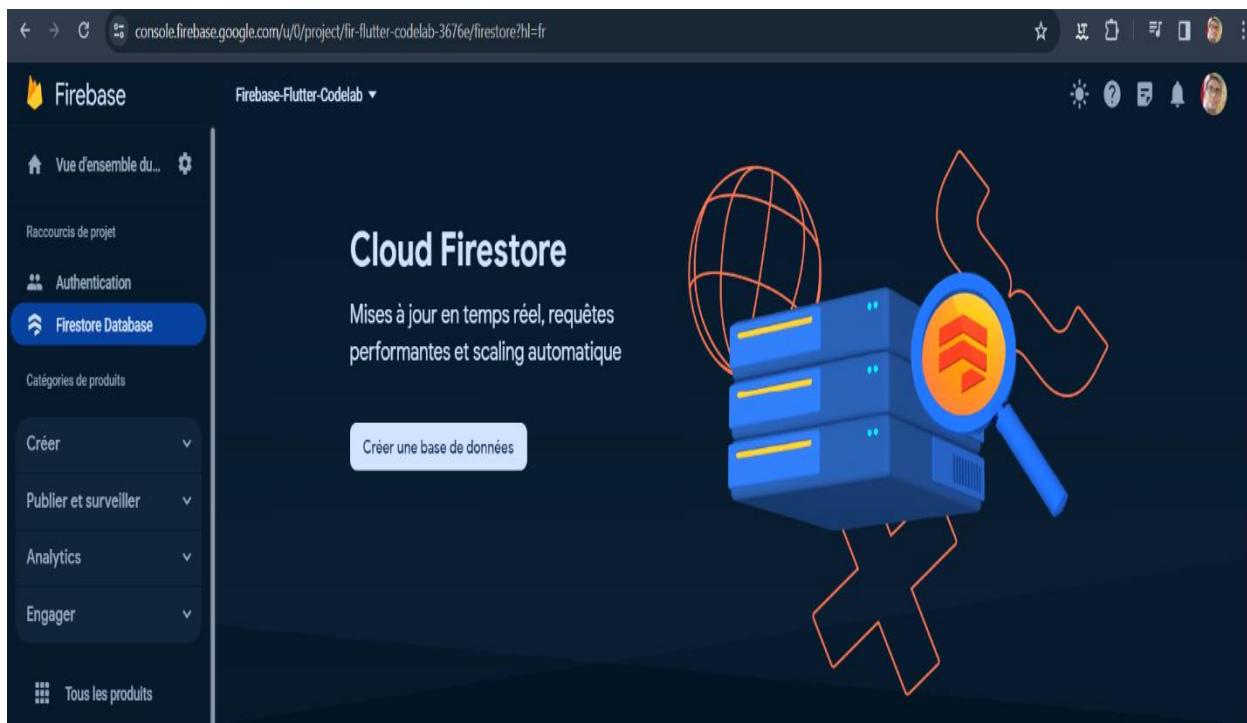
- 1- Dans le volet de présentation du projet de la console Firebase, développez le menu **Créer**.
- 2- Cliquez sur **Authentification > Commencer > Méthode de connexion > E-mail/Mot de passe > Activer > Enregistrer**

The screenshot shows the Firebase Authentication interface. On the left, there's a sidebar with project settings like 'Créer', 'Publier et surveiller', 'Analytics', and 'Engager'. The main area is titled 'Authentication' and has tabs for 'Users', 'Sign-in method', 'Templates', 'Usage', 'Settings', and 'Extensions'. A central box is titled 'Fournisseurs de connexion' (Providers) and contains three columns: 'Fournisseurs natifs' (Native providers) with 'Adresse e-mail/Mot de passe', 'Téléphone', and 'Anonyme'; 'Autres fournisseurs' (Other providers) with 'Google', 'Facebook', 'Play Jeux', 'Game Center', 'Apple', 'GitHub', 'Microsoft', 'Twitter', and 'Yahoo!'; and 'Fournisseurs personnalisés' (Custom providers) with 'OpenID Connect' and 'SAML'.

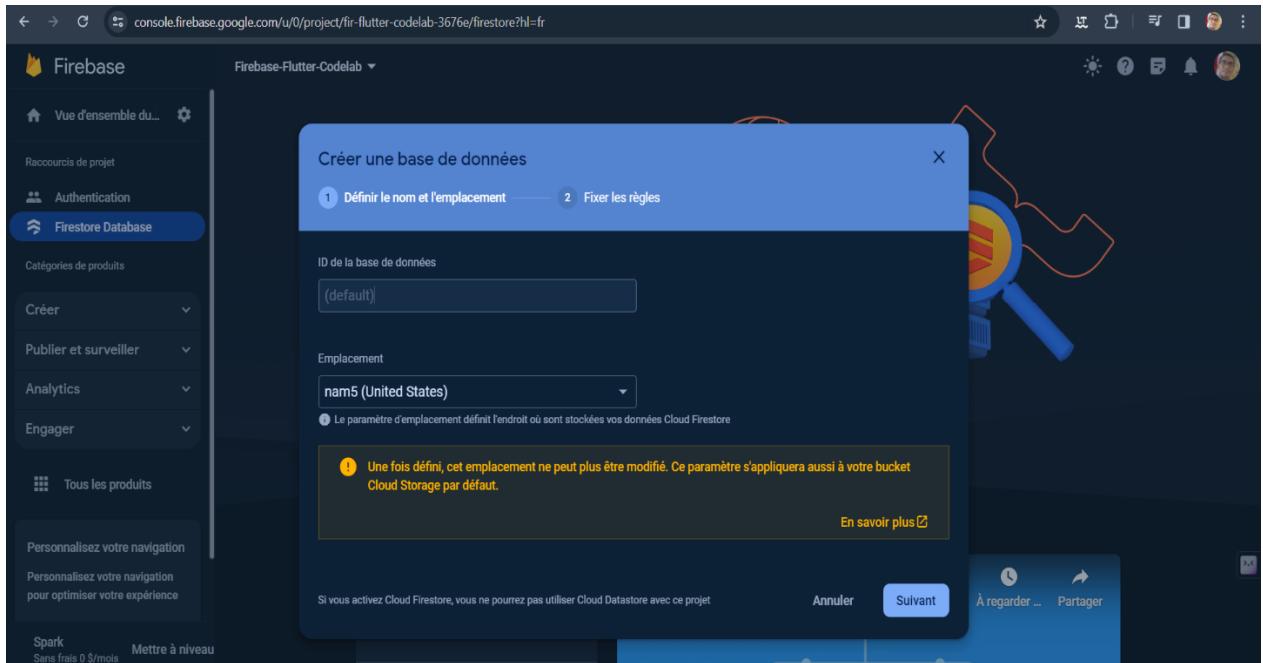
This screenshot shows the 'Email/Password' provider activation step. It features a large input field for 'Adresse e-mail/Mot de passe' with a checked 'Activer' (Enable) button. Below it is a descriptive text about email and password sign-in. At the bottom, there are buttons for 'Annuler' (Cancel), 'Enregistrer' (Register), and 'OK'.

## D- Activer Firestore

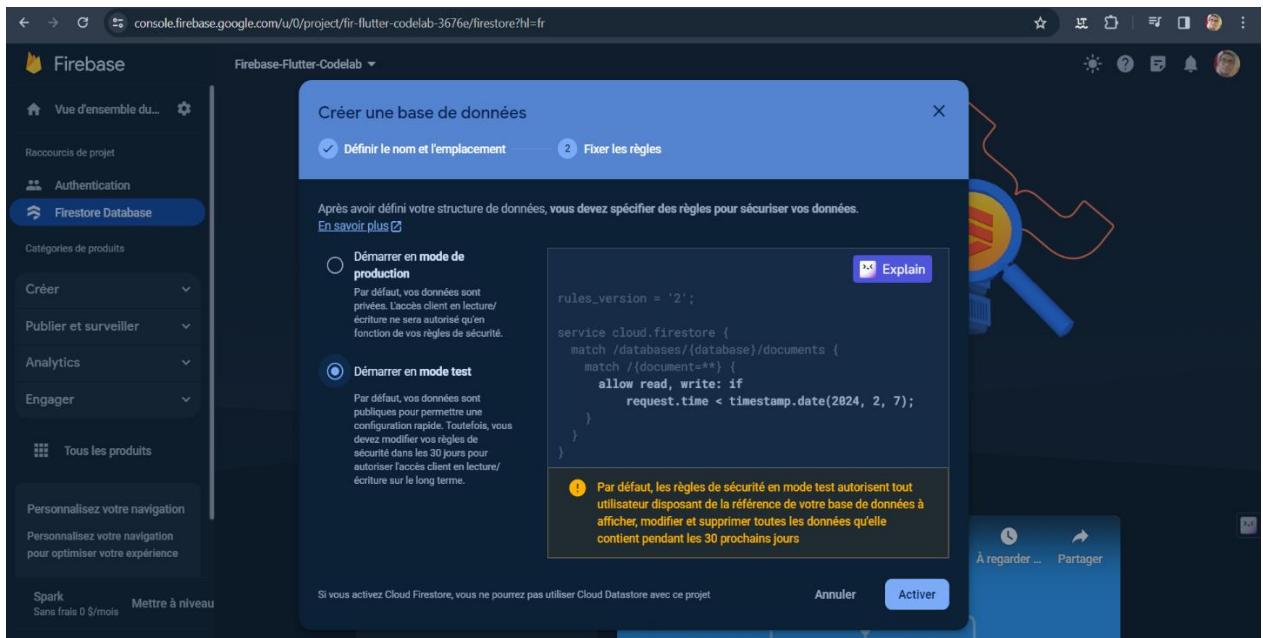
1- Dans le menu Créer , cliquez sur Base de données Firestore > Créer une base de données



2- Cliquez sur Suivant , puis sélectionnez l'emplacement de votre base de données. Vous pouvez utiliser la valeur par défaut. Vous ne pourrez pas modifier l'emplacement ultérieurement.



3- Sélectionnez Démarrer en mode test , puis lisez la clause de non-responsabilité concernant les règles de sécurité. Le mode test garantit que vous pouvez écrire librement dans la base de données pendant le développement.



#### 4- Cliquez sur Activer .

The screenshot shows the Firebase Cloud Firestore console. On the left, there's a sidebar with navigation links like 'Vue d'ensemble du projet', 'Authentication', 'Firestore Database' (which is selected), 'Catégories de produits', 'Créer', 'Publier et surveiller', 'Analytics', 'Engager', and 'Tous les produits'. Below these are sections for 'Personnalisez votre navigation' and 'Spark'. The main area is titled 'Cloud Firestore' and has tabs for 'Données', 'Règles', 'Index', 'Utilisation', and 'Extensions'. A banner at the top says 'Protégez vos ressources Cloud Firestore des utilisations abusives telles que la fraude à la facturation et le hameçonnage' and 'Configurer App Check'. At the bottom, it says 'Votre base de données est prête à l'emploi. Ajoutez simplement des données.' There's also a 'Vue Panneau' button.

#### E- Configurer Firebase

##### 1- Configurer les dépendances

Vous devez ajouter les bibliothèques *FlutterFire* pour les deux produits Firebase que vous utilisez dans cette application: Authentication et Firestore.

À partir de la ligne de commande, ajoutez les profondeurs suivantes:

```
$ flutter pub add firebase_core
```

The terminal window shows the command being run and its output. It lists the packages being resolved and updated, including firebase\_core, firebase\_core\_platform\_interface, firebase\_core\_web, go\_router, js, build, matcher, material\_color\_utilities, meta, path, test\_api, and web. It also notes that 4 dependencies were changed and 8 packages have newer versions incompatible with dependency constraints.

```
PS C:\La Famille Nathan\3emeannee\PWV\flutter-codelabs\firebase-get-to-know-flutter\step_02> flutter pub add firebase_core
Resolving dependencies...
+ firebase_core 2.24.2
+ firebase_core_platform_interface 5.0.0
+ firebase_core_web 2.10.0
+ go_router 12.1.3 (13.0.1 available)
+ js 0.6.7 (0.7.0 available)
+ build 0.12.16 (0.12.16+1 available)
+ matcher 0.12.16 (0.12.16+1 available)
+ material_color_utilities 0.5.0 (0.8.0 available)
+ meta 1.10.0 (1.11.0 available)
+ path 1.8.3 (1.9.0 available)
+ test_api 0.6.1 (0.7.0 available)
+ web 0.3.0 (0.4.0 available)
Changed 4 dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
```

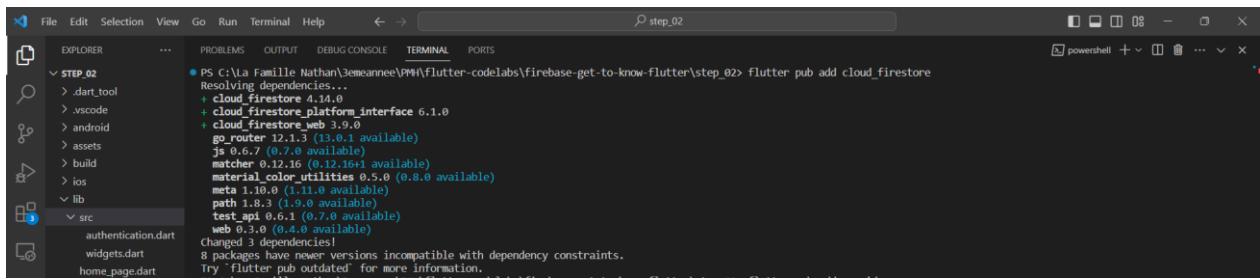
Le package `firebase_core` est le code commun requis pour tous les plugins Firebase Flutter.

```
$ flutter pub add firebase_auth
```

```
PS C:\a Famille Nathan\Semaine10\Flutter\flutter-code-labs\firebase-get-to-know-flutter\step_02> flutter pub add firebase_auth  
Resolving dependencies...  
+ flutterfire_internals 1.3.16  
+ firebase_auth 4.16.0  
+ firebase_auth_platform_interface 7.0.9  
+ firebase_auth_web 5.8.13  
+ go_router 12.1.3 (13.0.1 available)  
+ js 0.6.7 (0.7.0 available)  
+ matcher 0.12.16 (0.12.16+1 available)  
+ material_color_utilities 0.5.0 (0.8.0 available)  
+ meta 1.1.8 (1.1.9 available)  
+ path 1.8.3 (1.9.0 available)  
+ test_api 0.6.1 (0.7.0 available)  
+ web 0.1.8 (0.2.0 available)  
Changed 4 dependencies!  
8 packages have newer versions incompatible with dependency constraints.  
Try 'flutter pub outdated' for more information.
```

Le package `firebase_auth` permet l'intégration avec l'authentification.

```
$ flutter pub add cloud_firestore
```



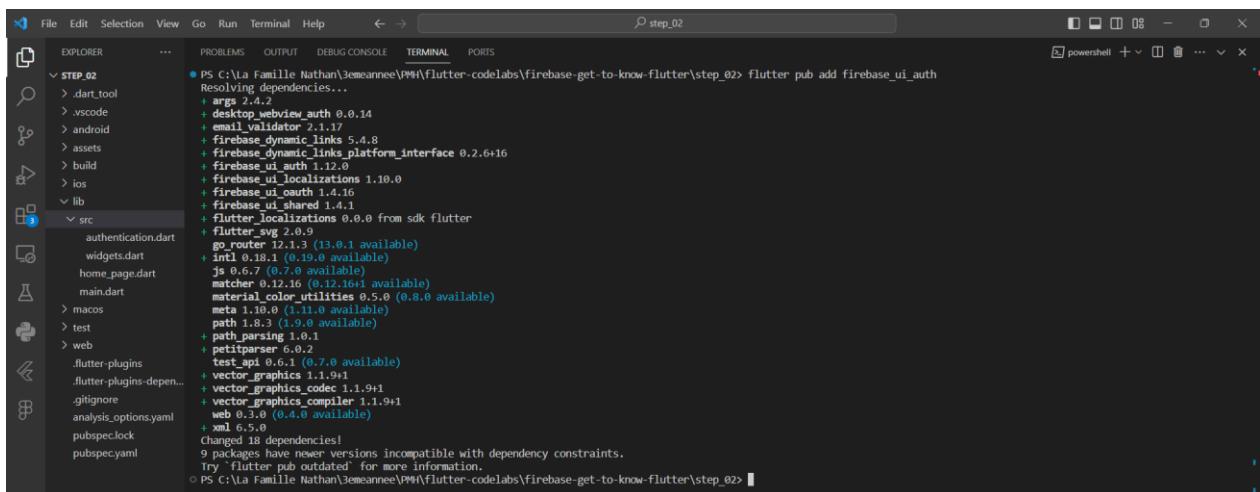
Le package `cloud_firestore` permet d'accéder au stockage de données Firestore.

```
$ flutter pub add provider
```

```
PS C:\La Faculté Notion\3eme année\PMP\Flutter-codelabs\firebase-get-to-know-flutter\step_02> flutter pub add provider
Resolving dependencies...
  • geolocator 12.1.3 (13.0.1 available)
    + js 0.6.7 (0.7.0 available)
    + matcher 0.12.16 (0.12.16+1 available)
    + material_color_utilities 0.5.0 (0.8.0 available)
    + meta 1.10.0 (1.11.0 available)
    + nested 1.0.0
    + path_provider_windows 0.9.0 available)
    + provider 6.1.1
    + test_api 0.0.1 (0.0.7 available)
    + web 0.1.0 (0.1.0 available)
Changed 2 dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
```

Le package `firebase_ui_auth` fournit un ensemble de widgets et d'utilitaires pour augmenter la vitesse du développeur avec les flux d'authentification.

```
$ flutter pub add firebase ui auth
```



## F- Configurer macOS

### 1- macOS / Runner / Debugprofile.Entlements

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>com.apple.security.app-sandbox</key>
    <true/>
    <key>com.apple.security.cs.allow-jit</key>
    <true/>
    <key>com.apple.security.network.server</key>
    <true/>
    <key>com.apple.security.network.client</key>
    <true/>
</dict>
</plist>
```

### 2- macOS / Runner / Release.Entlements

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>com.apple.security.app-sandbox</key>
    <true/>
    <key>com.apple.security.network.client</key>
    <true/>
</dict>
</plist>
```

## G- Ajouter la fonctionnalité RSVP

### ❖ Ajouter une logique commerciale avec le package Provider

Utilisez-le package provider pour rendre un objet d'état d'application centralisé disponible dans toute l'arbre de widgets de flutter de l'application:

### 1- Créez un nouveau fichier nommé app\_state.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_ui_auth/firebase_ui_auth.dart';
import 'package:flutter/material.dart';
import 'package:firebase_options.dart';

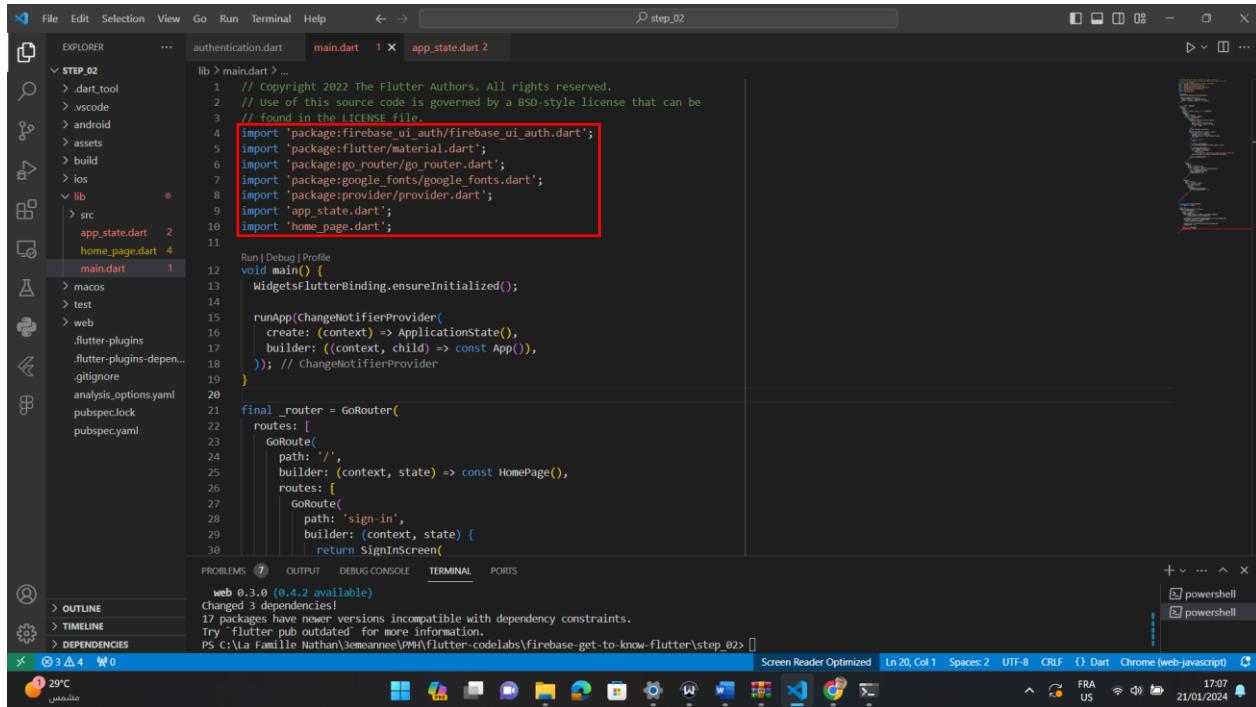
class ApplicationState extends ChangeNotifier {
    ApplicationState() {
        init();
    }

    bool _loggedIn = false;
    bool get loggedIn => _loggedIn;

    Future<void> init() async {
        await Firebase.initializeApp(
            options: DefaultFirebaseOptions.currentPlatform);
        FirebaseAuth.instance.userChanges().listen((user) {
            if (user != null) {
                _loggedIn = true;
            } else {
                _loggedIn = false;
            }
            notifyListeners();
        });
    }
}
```

## ❖ Intégrer le flux d'authentification

### 1- Modifiez les importations en haut du fichier lib/main.dart :



The screenshot shows the VS Code interface with the main.dart file open in the editor. The imports for firebase, flutter, go\_router, provider, and google\_fonts are highlighted with a red box. The code itself is mostly identical to the previous screenshot.

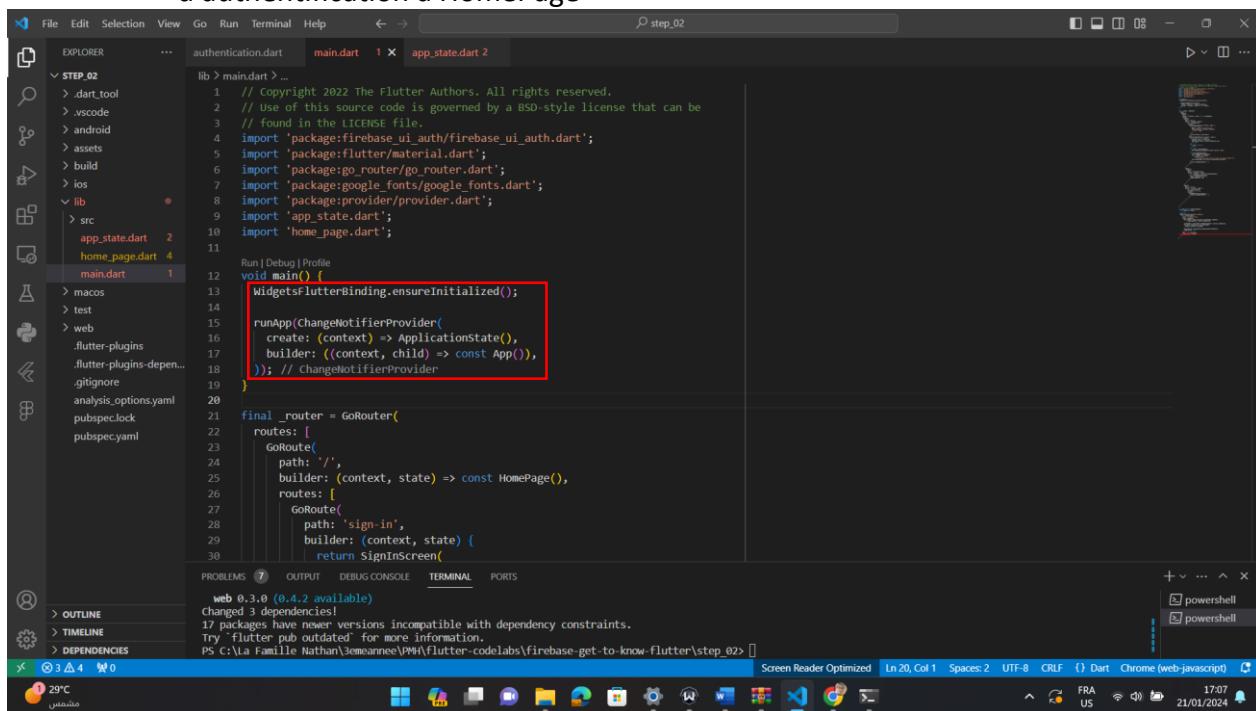
```
// Copyright 2022 The Flutter Authors. All rights reserved.
// Use of this source code is governed by a BSD-style license that can be
// found in the LICENSE file.
import 'package:firebase_ui_auth/firebase_ui_auth.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:provider/provider.dart';
import 'app_state.dart';
import 'home_page.dart';

void main() {
  WidgetsFlutterBinding.ensureInitialized();

  runApp(ChangeNotifierProvider(
    create: (context) => Applicationstate(),
    builder: ((context, child) => const App()),
  )); // ChangeNotifierProvider

  final _router = GoRouter(
    routes: [
      GoRoute(
        path: '/',
        builder: (context, state) => const HomePage(),
        routes: [
          GoRoute(
            path: 'sign-in',
            builder: (context, state) {
              return SignInScreen();
            }
          )
        ]
      )
    ]
  );
}
```

### 2- Connectez l'état de l'application à linitialisation de lapplication, puis ajoutez le flux dauthentification à HomePage



The screenshot shows the VS Code interface with the main.dart file open. The line where the application state is initialized is highlighted with a red box. The rest of the code is identical to the previous screenshots.

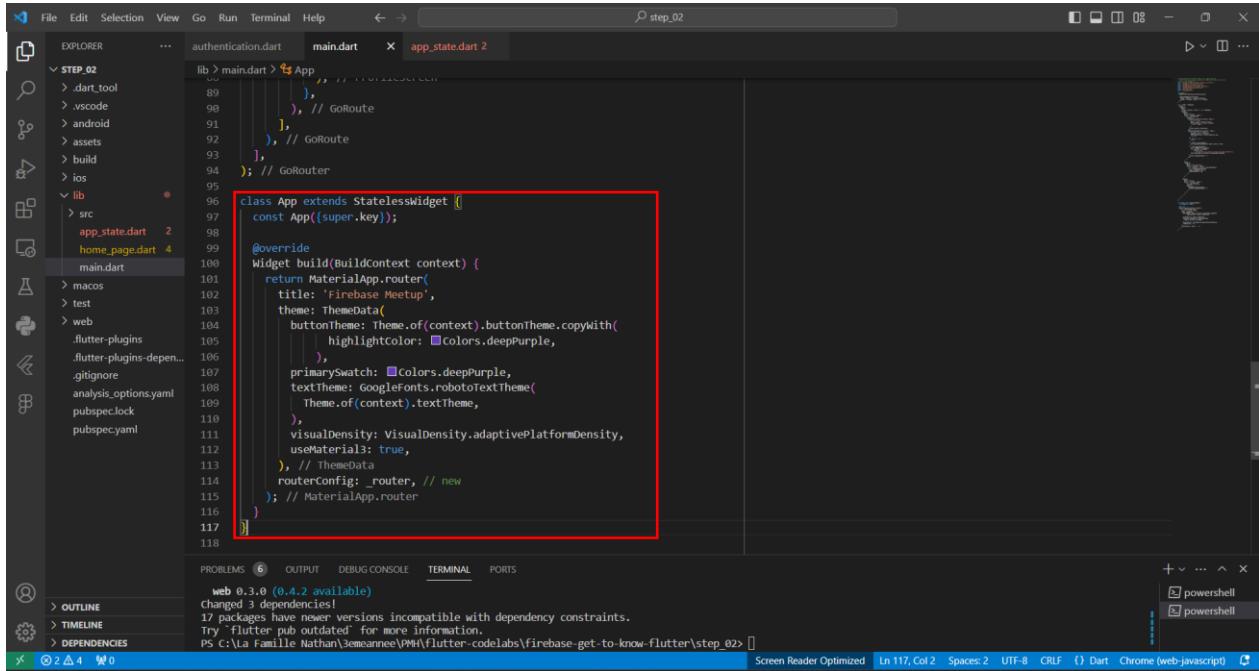
```
// Copyright 2022 The Flutter Authors. All rights reserved.
// Use of this source code is governed by a BSD-style license that can be
// found in the LICENSE file.
import 'package:firebase_ui_auth/firebase_ui_auth.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:provider/provider.dart';
import 'app_state.dart';
import 'home_page.dart';

void main() {
  WidgetsFlutterBinding.ensureInitialized();

  runApp(ChangeNotifierProvider(
    create: (context) => Applicationstate(),
    builder: ((context, child) => const App()),
  )); // changeNotifierProvider

  final _router = GoRouter(
    routes: [
      GoRoute(
        path: '/',
        builder: (context, state) => const HomePage(),
        routes: [
          GoRoute(
            path: 'sign-in',
            builder: (context, state) {
              return SignInScreen();
            }
          )
        ]
      )
    ]
  );
}
```

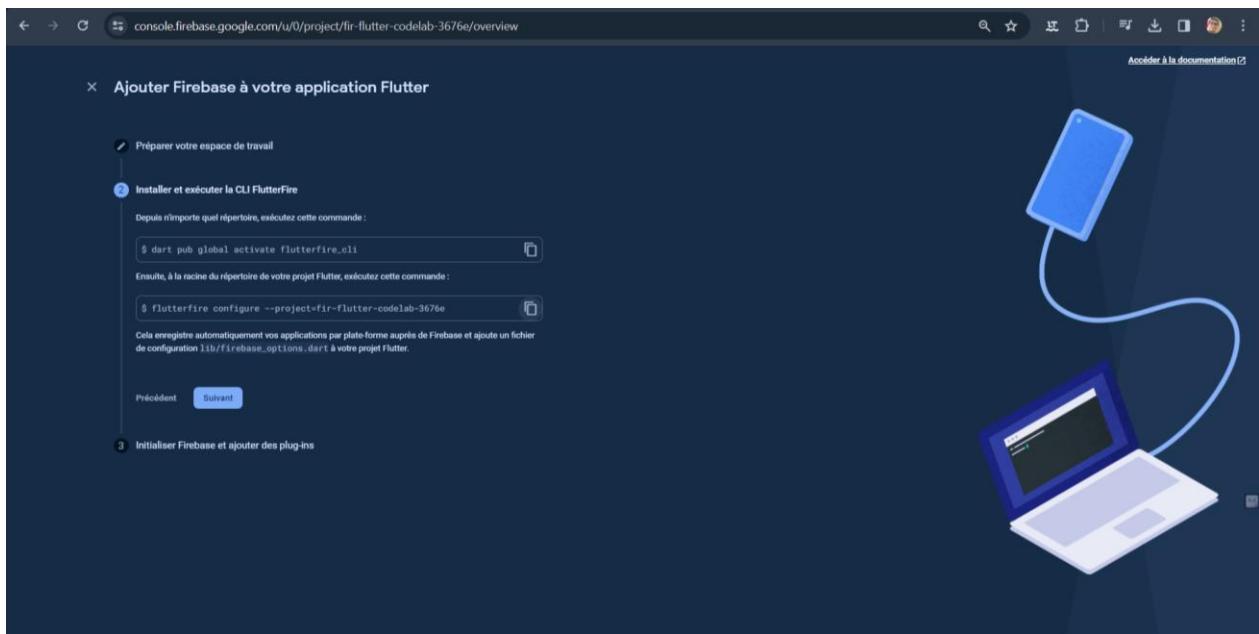
### 3- Mettez à jour votre application pour gérer la navigation vers les différents écrans fournis par FirebaseUI, en créant une configuration GoRouter



```
class App extends StatelessWidget {
  const App({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp.router(
      title: 'Firebase Meetup',
      theme: ThemeData(
        buttonTheme: Theme.of(context).buttonTheme.copyWith(
          highlightColor: Colors.deepPurple,
        ),
        primarySwatch: Colors.deepPurple,
        textTheme: GoogleFonts.robotoTextTheme(
          Theme.of(context).textTheme,
        ),
        visualDensity: VisualDensity.adaptivePlatformDensity,
        useMaterial3: true,
      ), // ThemeData
      routerConfig: _router, // new
    ); // MaterialApp
  }
}
```

### 4- Ajouter Firebase à votre application Flutter



The screenshot shows the 'Ajouter Firebase à votre application Flutter' (Add Firebase to your app) wizard in the Firebase console. It is on step 2: 'Installer et exécuter la CLI FlutterFire'. The steps are:

- Préparer votre espace de travail
- Installer et exécuter la CLI FlutterFire
- Initialiser Firebase et ajouter des plug-ins

For step 2, it provides instructions to run the command \$ dart pub global activate flutterfire\_cli and then \$ flutterfire configure --project=fir-flutter-codelab-3676e. A note says: 'Cela enregistre automatiquement vos applications par plate-forme auprès de Firebase et ajoute un fichier de configuration lib/firebase\_options.dart à votre projet Flutter.'

A blue smartphone icon is connected by a wire to a blue laptop icon, symbolizing connectivity.

## 5- Exécuter la commande

The screenshot shows the VS Code interface with the terminal tab active. The command `flutterfire configure --project=fir-flutter-codelab-3676e` has been run, and the output shows the configuration file `lib/firebase\_options.dart` was generated successfully for web, android, ios, and macos platforms. It also lists the Firebase App Ids for each platform.

```
Waiting for connection from debug service on chrome... 25,1s
Failed to compile application.
PS C:\La Famille Nathan\3emeannee\PMP\flutter-codelabs\firebase-get-to-know-flutter\step_02> flutterfire configure --project=fir-flutter-codelab-3676e
i Found 2 Firebase projects. Selecting project fir-flutter-codelab-3676e.
i Which platforms should your configuration support? [use arrow keys & space to select]? .android, .ios, .macos, .web
i Firebase android app com.example.gtk_flutter is not registered on Firebase project fir-flutter-codelab-3676e.
i Registered a new Firebase android app on Firebase project fir-flutter-codelab-3676e.
i Firebase ios app com.example.gtkFlutter is not registered on Firebase project fir-flutter-codelab-3676e.
i Registered a new Firebase ios app on Firebase project fir-flutter-codelab-3676e.
i Firebase macos app com.example.gtkFlutter_RunnerTests is not registered on Firebase project fir-flutter-codelab-3676e.
i Registered a new Firebase macos app on Firebase project fir-flutter-codelab-3676e.
i Firebase web app gtk_flutter (.web) is not registered on Firebase project fir-flutter-codelab-3676e.
i Registered a new Firebase web app on Firebase project fir-flutter-codelab-3676e.
i Registered a new Firebase web app on Firebase project fir-flutter-codelab-3676e.

Platform Firebase App Id
web 1:574742f14105:web:fd42902773bc89697428
android 1:574742f14105:android:1079a86f28573ce9697428
ios 1:574742f14105:ios:42f272b70251cb697428
macos 1:574742f14105:ios:51c1817065a1c2c697428

Learn more about using this file and next steps from the documentation:
> https://firebase.google.com/docs/flutter/setup
PS C:\La Famille Nathan\3emeannee\PMP\flutter-codelabs\firebase-get-to-know-flutter\step_02>
```

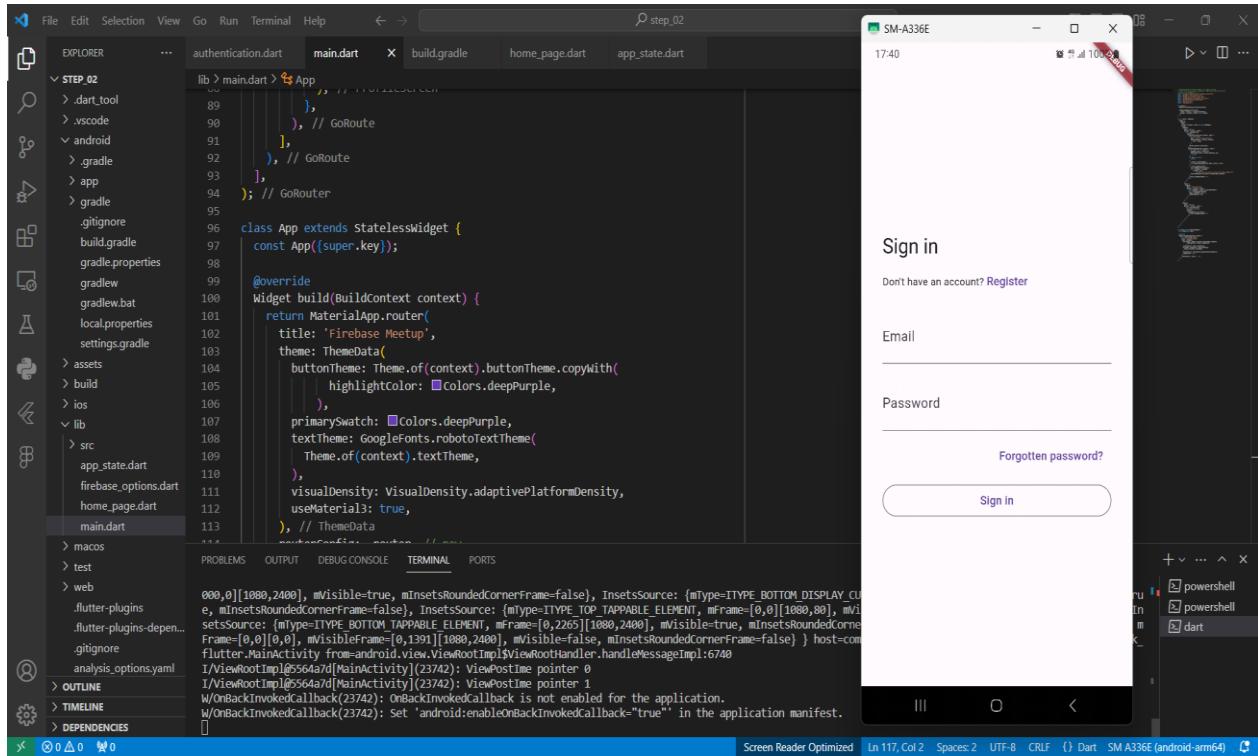
## 6- Tester le flux d'authentification

The screenshot shows the VS Code interface with the main.dart file open in the editor. The code defines a GoRouter and a StatelessWidget named App. A preview window on the right shows a mobile application titled "Firebase Meetup" with a group of diverse people. Below the title, it says "October 30" and "San Francisco". A "RSVP" button is visible. The text "What we'll be doing" and "Join us for a day full of Firebase Workshops and Pizza!" are also displayed. The bottom status bar indicates the device is "SM-A336E".

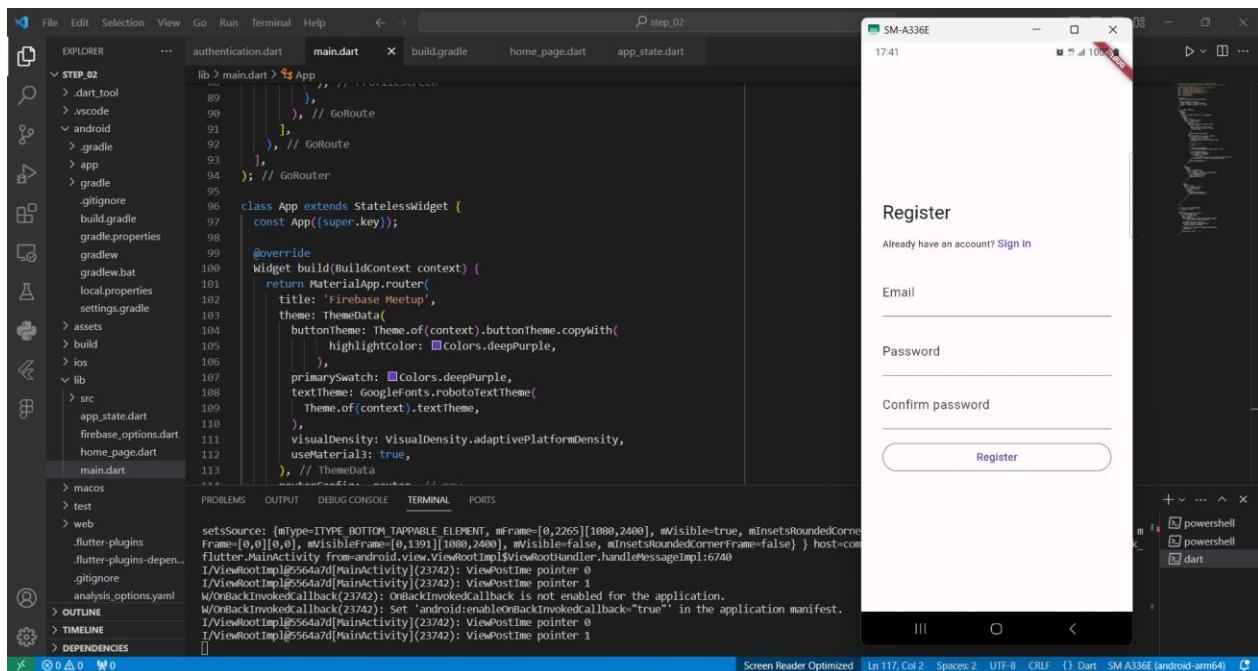
```
lib/main.dart
89     },
90   ],
91 ],
92 ],
93 ],
94 );
95
96 class App extends StatelessWidget {
97   const App({super.key});
98
99   @override
100   Widget build(BuildContext context) {
101     return MaterialApp.router(
102       title: 'Firebase Meetup',
103       theme: ThemeData(
104         buttonTheme: Theme.of(context).buttonTheme.copyWith(
105           highlightColor: Colors.deepPurple,
106         ),
107         primarySwatch: Colors.deepPurple,
108         textTheme: GoogleFonts.robotoTextTheme(
109           Theme.of(context).textTheme,
110         ),
111         visualDensity: VisualDensity.adaptivePlatformDensity,
112         useMaterial3: true,
113       ),
114     );
115   }
116 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

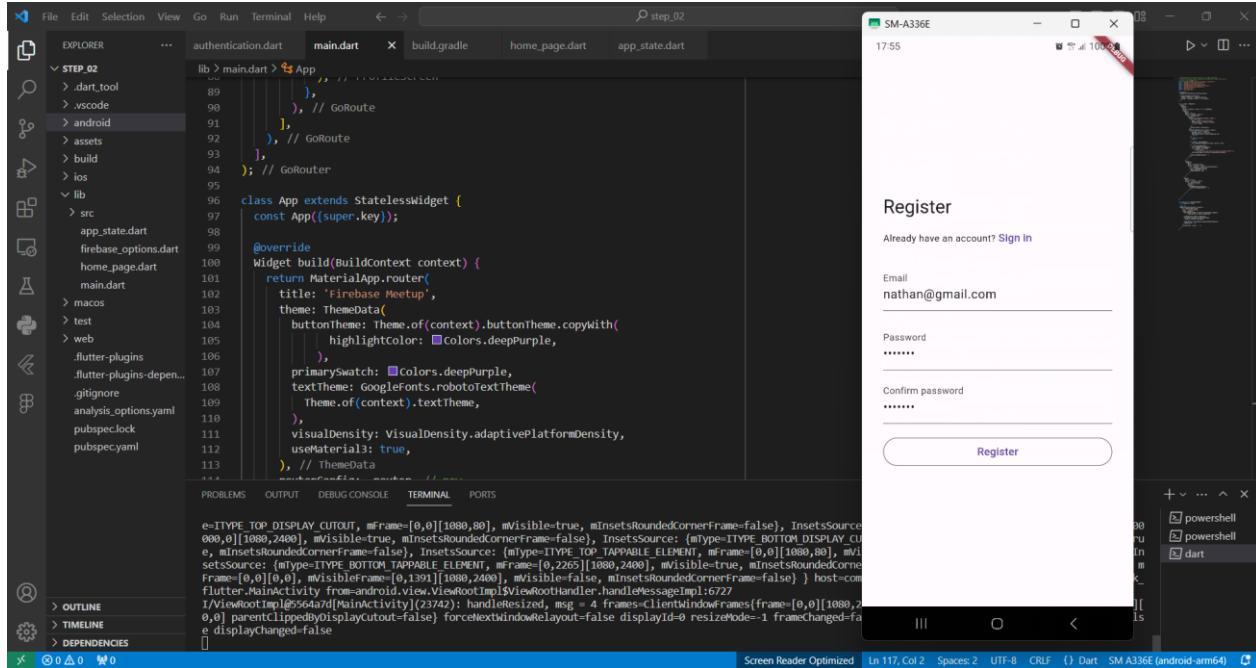
- a. Dans l'application, appuyez sur le bouton RSVP pour lancer le SignInScreen .



- b. Entrez une adresse email. Si vous êtes déjà inscrit, le système vous demande de saisir un mot de passe. Sinon, le système vous invite à remplir le formulaire d'inscription.



- c. Saisissez un mot de passe de moins de six caractères pour vérifier le flux de gestion des erreurs. Si vous êtes inscrit, vous voyez à la place le mot de passe.



Une fois, je clique sur Register, mes données sont envoyé dans la base de donnée de Firebase

Méthode	Fournisseur	Date de création	Dernière connexion	UID utilisateur
nathan@gmail.com		21 janv. 2024		yf485nWl.mTgnul.b32BnWJic...

- d. Saisissez des mots de passe incorrects pour vérifier le flux de gestion des erreurs.

The screenshot shows the VS Code interface with the main.dart file open in the editor. The code defines an App widget that uses a GoRouter to handle routes. The router has two routes: one for the home page and another for the authentication screen. The authentication screen is a StatelessWidget that displays a sign-in form with fields for Email and Password, and links for Register and Forgotten password. Below the form, an error message indicates that the supplied auth credential is incorrect, malformed or has expired. The Android emulator window titled 'SM-A336E' shows the sign-in screen with the same UI elements and error message.

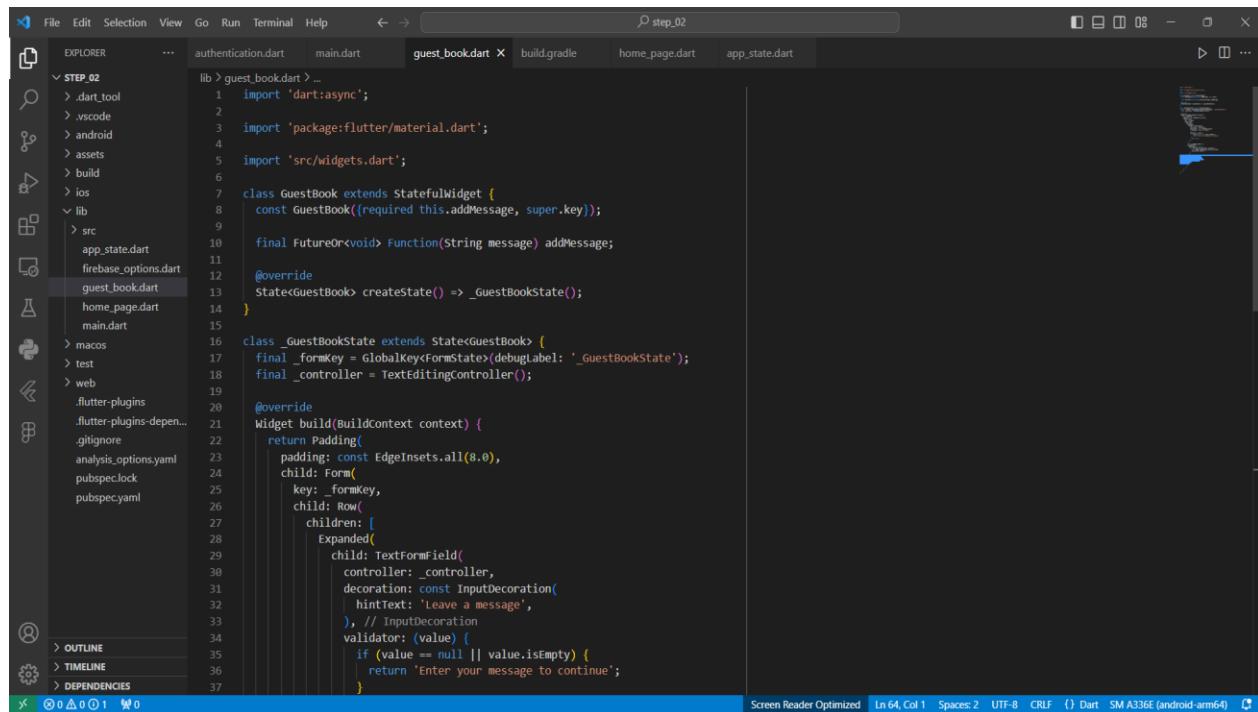
- a- Entrez le mot de passe correct. Vous voyez l'expérience de connexion, qui offre à l'utilisateur la possibilité de se déconnecter.

The screenshot shows the VS Code interface with the main.dart file open in the editor. The code is identical to the previous screenshot, defining the same App widget with its routes. The Android emulator window titled 'SM-A336E' shows the home screen of the 'Firebase Meetup' application. The screen features a banner with a group of people, a date ('October 30'), a location ('San Francisco'), and two buttons ('Logout' and 'Profile'). Below the banner, there is a section titled 'What we'll be doing' with the text 'Join us for a day full of Firebase Workshops and Pizza!'. The bottom status bar of the emulator shows the text 'Screen Reader Optimized'.

## H- Écrire des messages sur Firestore

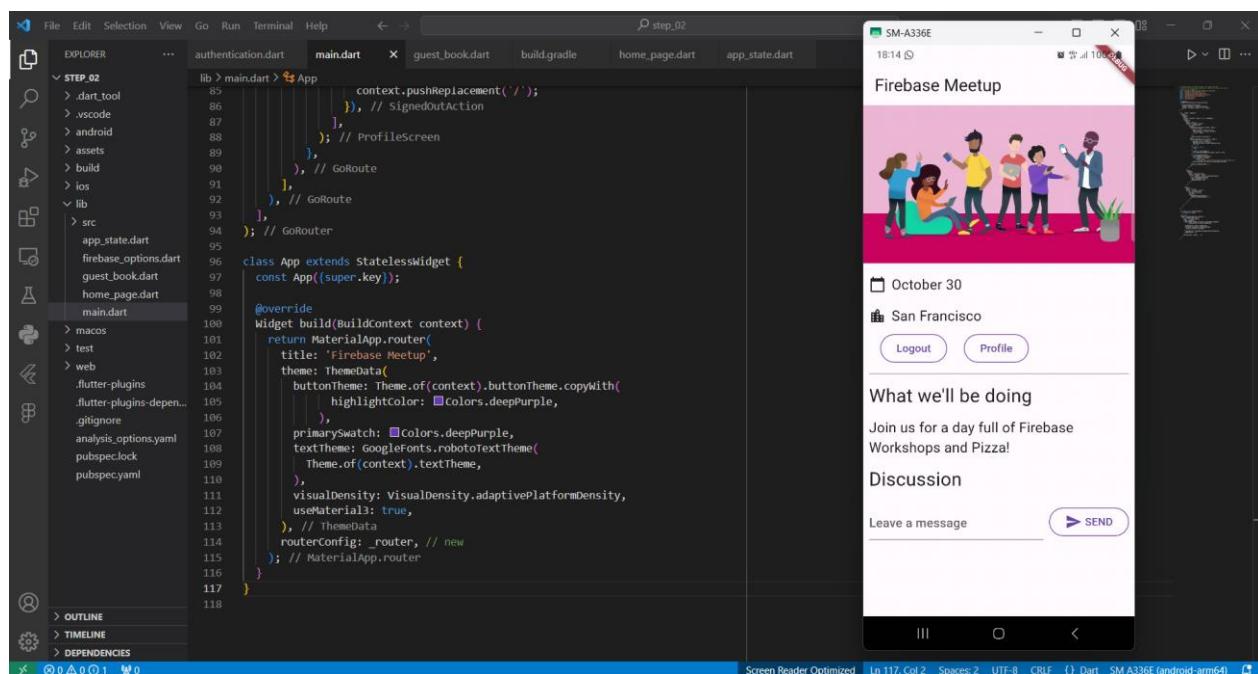
### 1- Ajouter des messages à Firestore

Créez un nouveau fichier nommé guest\_book.dart , ajoutez un widget avec état GuestBook pour construire les éléments d'interface utilisateur d'un champ de message et d'un bouton d'envoi

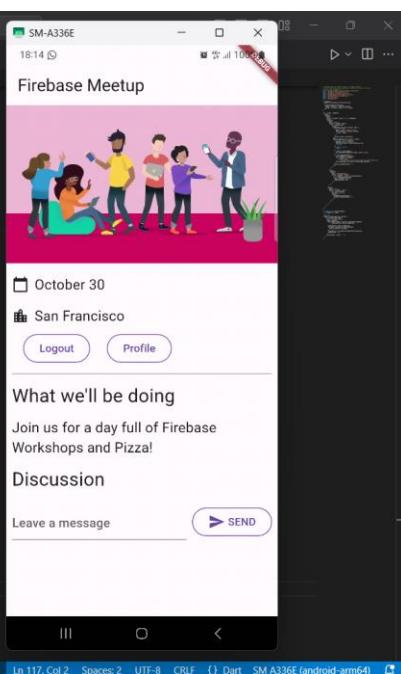


```
lib > guest_book.dart > ...
1 import 'dart:async';
2
3 import 'package:flutter/material.dart';
4
5 import 'src/widgets.dart';
6
7 class GuestBook extends StatefulWidget {
8   const GuestBook({required this.addMessage, super.key});
9
10   final FutureOr<void> Function(String message) addMessage;
11
12   @override
13   State<GuestBook> createState() => _GuestBookState();
14 }
15
16 class _GuestBookState extends State<GuestBook> {
17   final GlobalKey<FormState> _formKey = GlobalKey<FormState>(debugLabel: '_GuestBookState');
18   final TextEditingController _controller = TextEditingController();
19
20   @override
21   Widget build(BuildContext context) {
22     return Padding(
23       padding: const EdgeInsets.all(8.0),
24       child: Form(
25         key: _formKey,
26         child: Row(
27           children: [
28             Expanded(
29               child: TextFormField(
30                 controller: _controller,
31                 decoration: const InputDecoration(
32                   hintText: 'Leave a message',
33                 ), // InputDecoration
34                 validator: (value) {
35                   if (value == null || value.isEmpty) {
36                     return 'Enter your message to continue';
37                   }
38                 }
39               )
40             )
41           ],
42         ),
43       ),
44     );
45   }
46 }
```

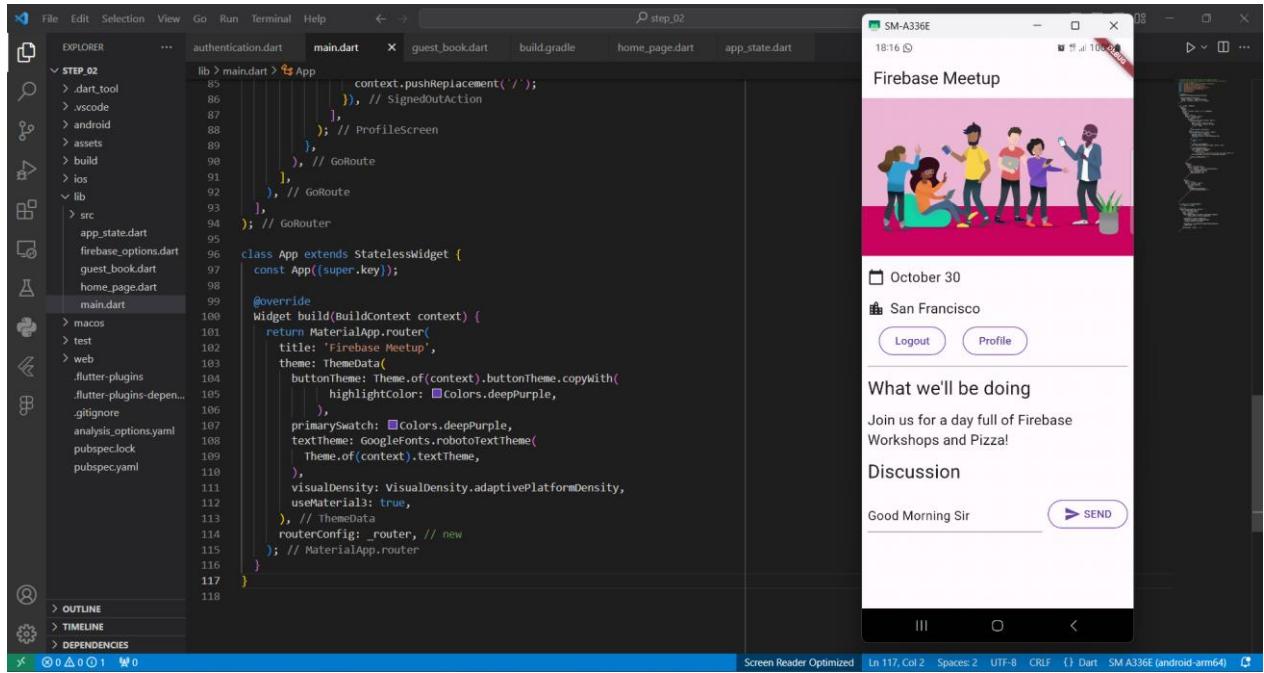
### 2- Aperçu de l'application



```
lib > main.dart > App
85   context.pushReplacement('/'); // SignedOutAction
86   }, // ProfileScreen
87   ), // GoRoute
88   ), // GoRoute
89   ), // GoRouter
90
91 class App extends StatelessWidget {
92   const App({super.key});
93
94   @override
95   Widget build(BuildContext context) {
96     return MaterialApp.router(
97       title: 'Firebase Meetup',
98       theme: ThemeData(
99         buttonTheme: Theme.of(context).buttonTheme.copyWith(
100           highlightColor: Colors.deepPurple,
101         ),
102         primarySwatch: Colors.deepPurple,
103         textTheme: GoogleFonts.robotoTextTheme(
104           Theme.of(context).textTheme,
105         ),
106         visualDensity: VisualDensity.adaptivePlatformDensity,
107         useMaterial3: true,
108       ), // ThemeData
109       routerConfig: _router, // new
110     ); // MaterialApp.router
111
112   }
113
114   _router = MaterialPageRoute(
115     builder: (context) => const FirebaseMeetup(),
116   );
117 }
```



### 3- Tester l'envoi de messages



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure for "STEP\_02". Files listed include authentication.dart, lib/main.dart, guest\_book.dart, build.gradle, home\_page.dart, app\_state.dart, main.dart, android, assets, build, ios, lib/src, firebase\_options.dart, guest\_book.dart, home\_page.dart, main.dart, .gitignore, analysis\_options.yaml, pubspec.lock, and pubspec.yaml.
- Code Editor (Top Right):** Displays the content of the main.dart file. The code defines a MaterialApp with a router configuration that includes routes for SignedOutAction, ProfileScreen, GoRoute, and a final route. It also defines an App StatelessWidget that returns the MaterialApp.
- Output (Bottom Right):** Shows the output of the command "flutter run -d SM-A336E". The log indicates the app is running on the device with the message "Running on SM-A336E".
- Flutter Application Preview (Center):** The app is titled "Firebase Meetup" and features a pink header with the title. Below the header is a cartoon illustration of five people at a meet-up. The main content area displays the text "October 30" and "San Francisco" with "Logout" and "Profile" buttons. A section titled "What we'll be doing" encourages users to join for workshops and pizza. A "Discussion" section contains the message "Good Morning Sir" with a "SEND" button.

Cette action écrit le message dans votre base de données Firestore. Cependant, vous ne voyez pas le message dans votre application Flutter actuelle, car vous devez toujours implémenter la récupération des données, ce que vous ferez à l'étape suivante. Cependant, dans le tableau de bord de base de données de la console Firebase, vous pouvez voir votre message ajouté dans la collection de guestbook . Si vous envoyez plus de messages, vous ajoutez plus de documents à votre collection de guestbook .

The screenshot shows the Firebase Cloud Firestore interface. On the left, there's a sidebar with project navigation (Vue d'ensemble du projet, Authentication, Firestore Database), categories (Créer, Publier et surveiller, Analytics, Engager), and a search bar. The main area is titled "Cloud Firestore" and has tabs for Données, Règles, Index, Utilisation, and Extensions. A banner at the top right says "Protégez vos ressources Cloud Firestore des utilisations abusives telles que la fraude à la facturation et le homecoming". Below the banner, there's a "Vue Panneau" tab selected. The main content area shows a document structure under "guestbook > YM0ynaqNv2I...". The document contains fields: name ("nathan"), text ("Good Morning Sir"), timestamp (1705846685659), and userId ("yE48SwWLmTgeutb328xWJePDGxZ").

## I- Lire les messages

### 1- Synchroniser les messages

- a- Créez un nouveau fichier `guest_book_message.dart`, ajoutez la classe suivante pour exposer une vue structurée des données que vous stockez dans Firestore.

The screenshot shows the VS Code editor with the file `guest_book_message.dart` open. The code defines a class `GuestBookMessage` with properties `name` and `message`. The code is highlighted with a red rectangle.

```

class GuestBookMessage {
  GuestBookMessage({required this.name, required this.message});

  final String name;
  final String message;
}

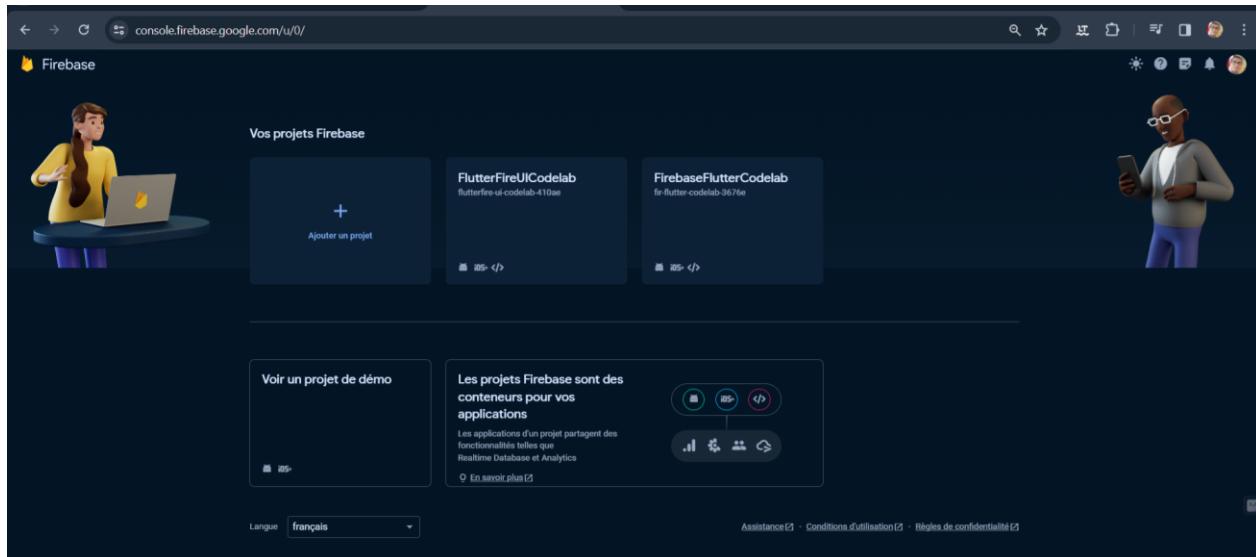
```

## Task 4: Local development for your Flutter apps using the Firebase Emulator Suite

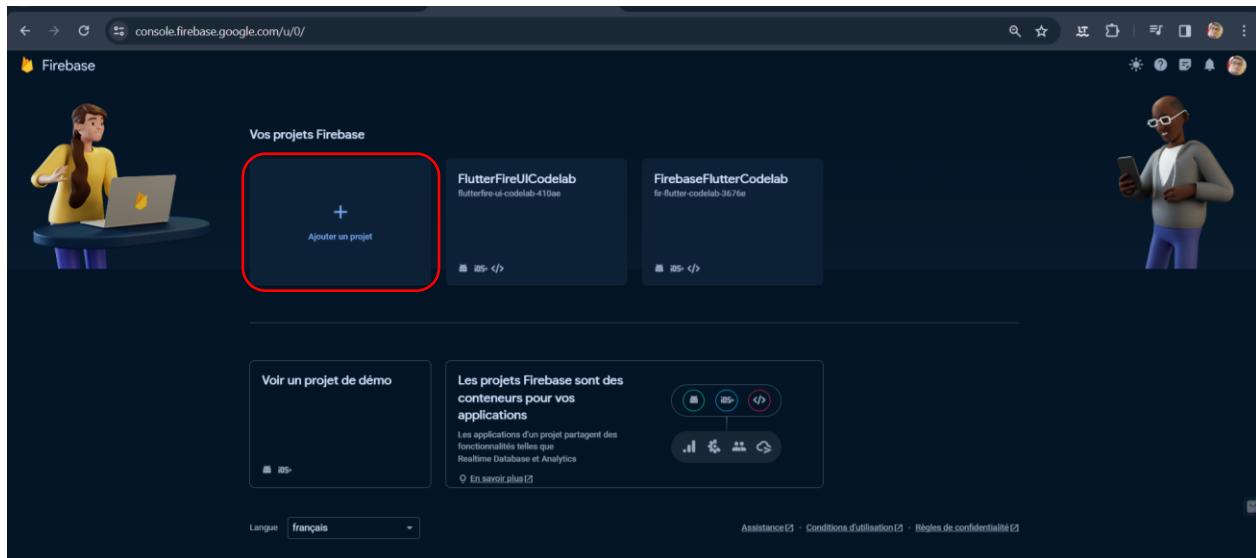
### A- Créer et configurer un projet Firebase

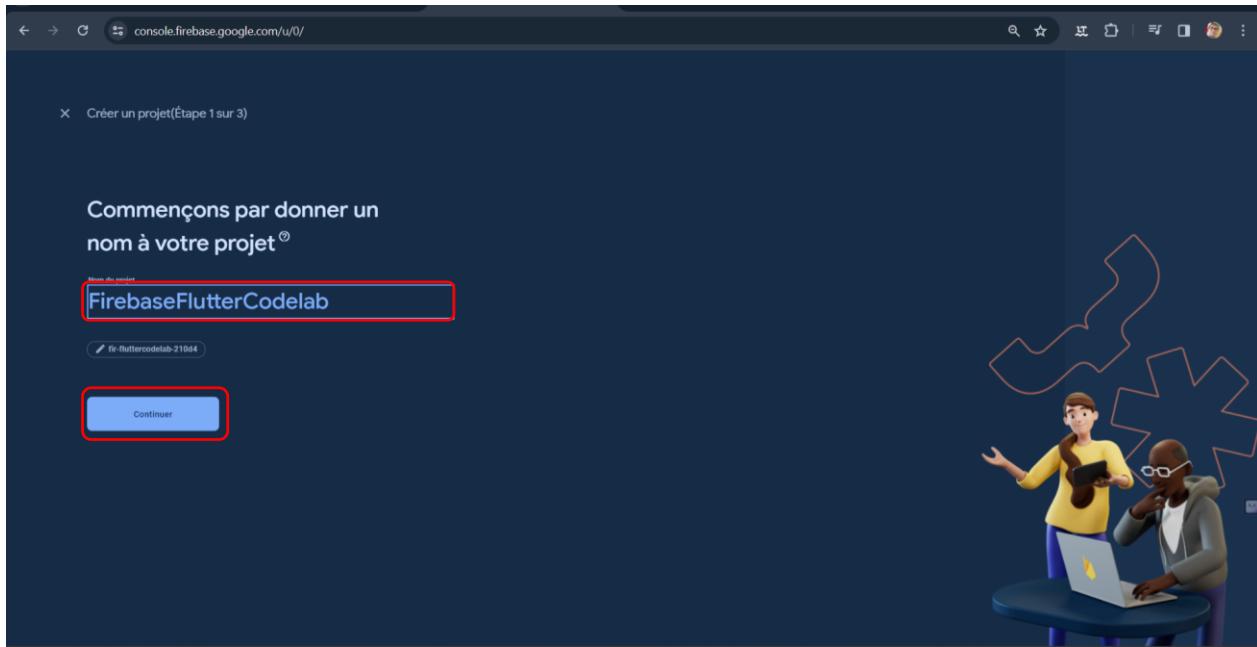
#### 1- Créer un projet Firebase

- Connectez-vous à la console Firebase.

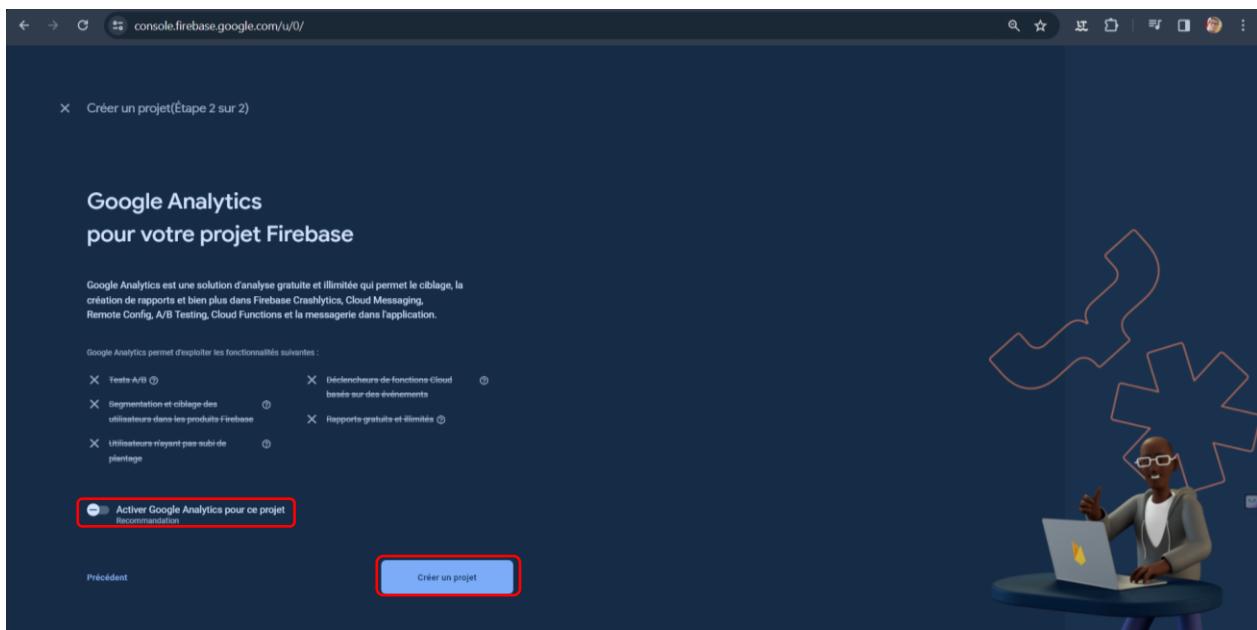


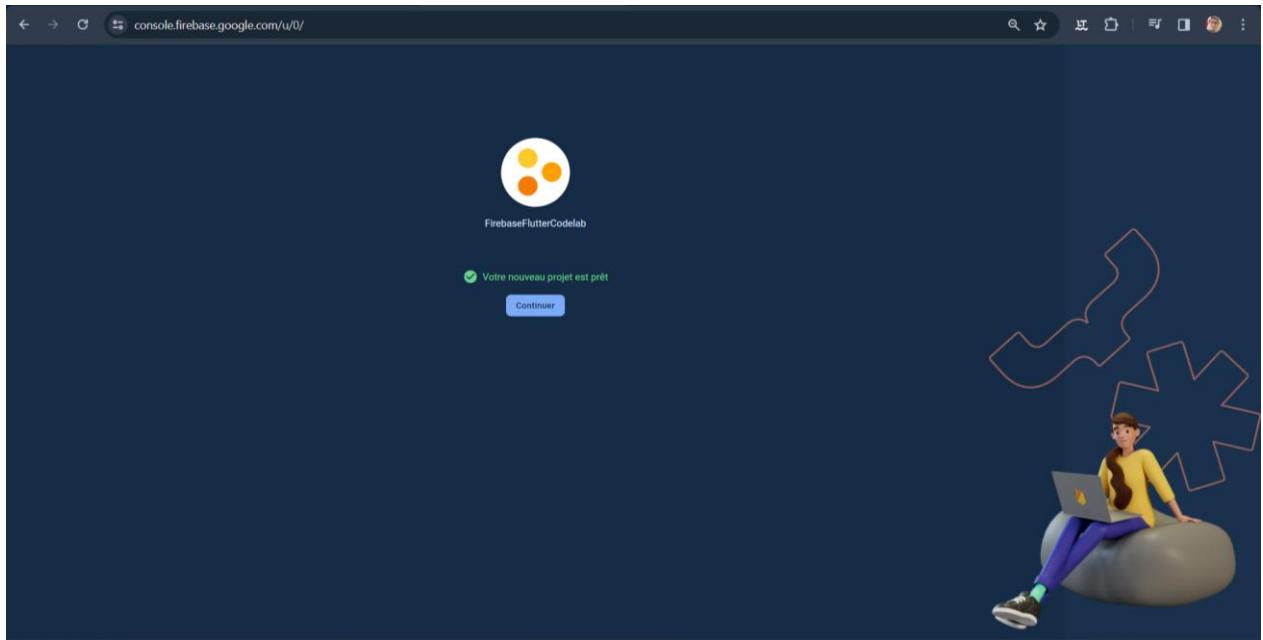
- Dans la console Firebase, cliquez sur Ajouter un projet (ou Créez un projet) et saisissez un nom pour votre projet Firebase





- Cliquez sur les options de création de projet. Acceptez les conditions de Firebase si vous y êtes invité. Ignorez la configuration de Google Analytics, car vous n'utiliserez pas Analytics pour cette application.





## 2- Activer Cloud Firestore

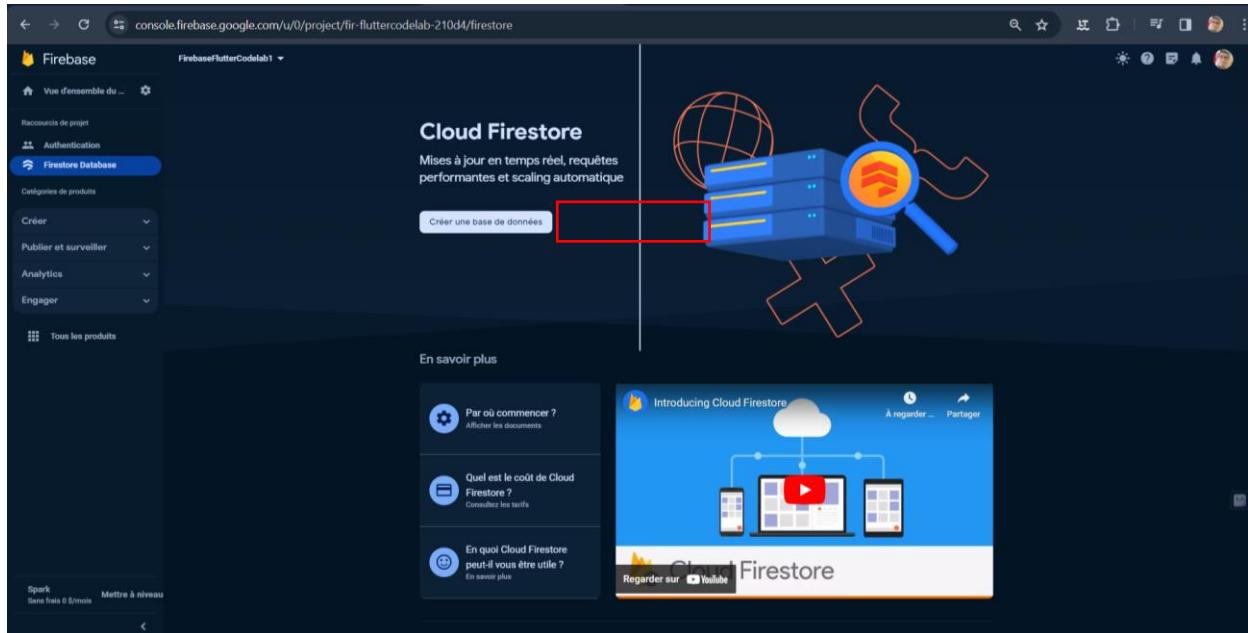
L'application Flutter utilise Cloud Firestore pour enregistrer les entrées de journal.

Activer Cloud Firestore :

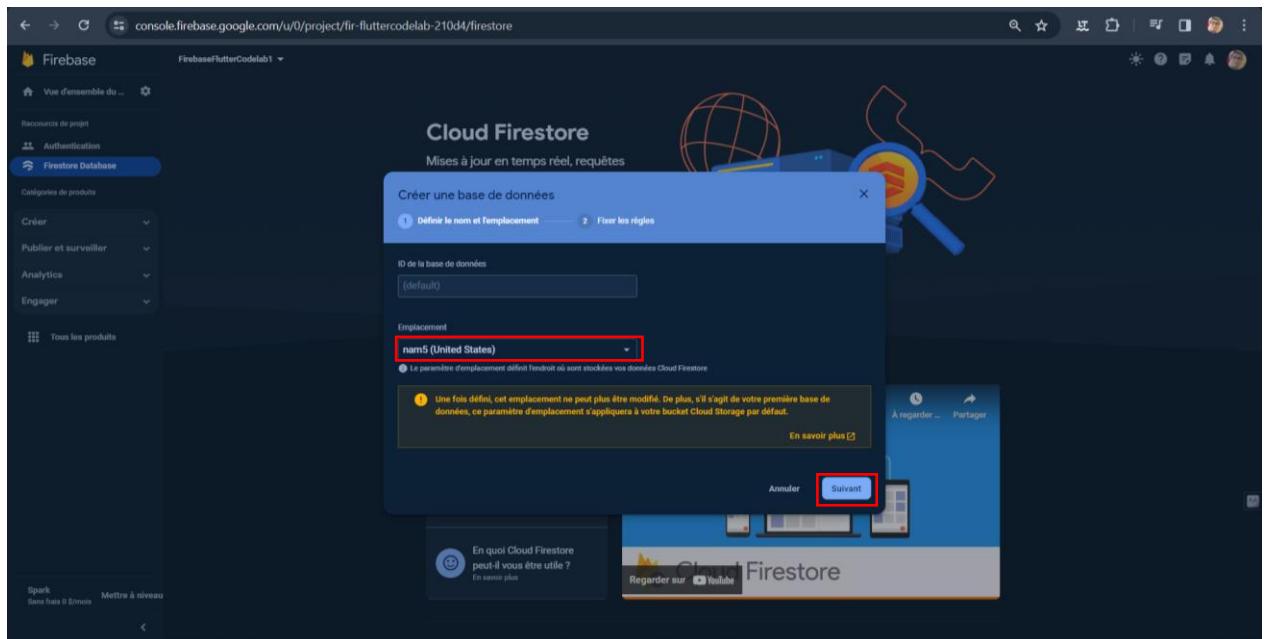
- Dans la section Build de la console Firebase, cliquez sur Cloud Firestore

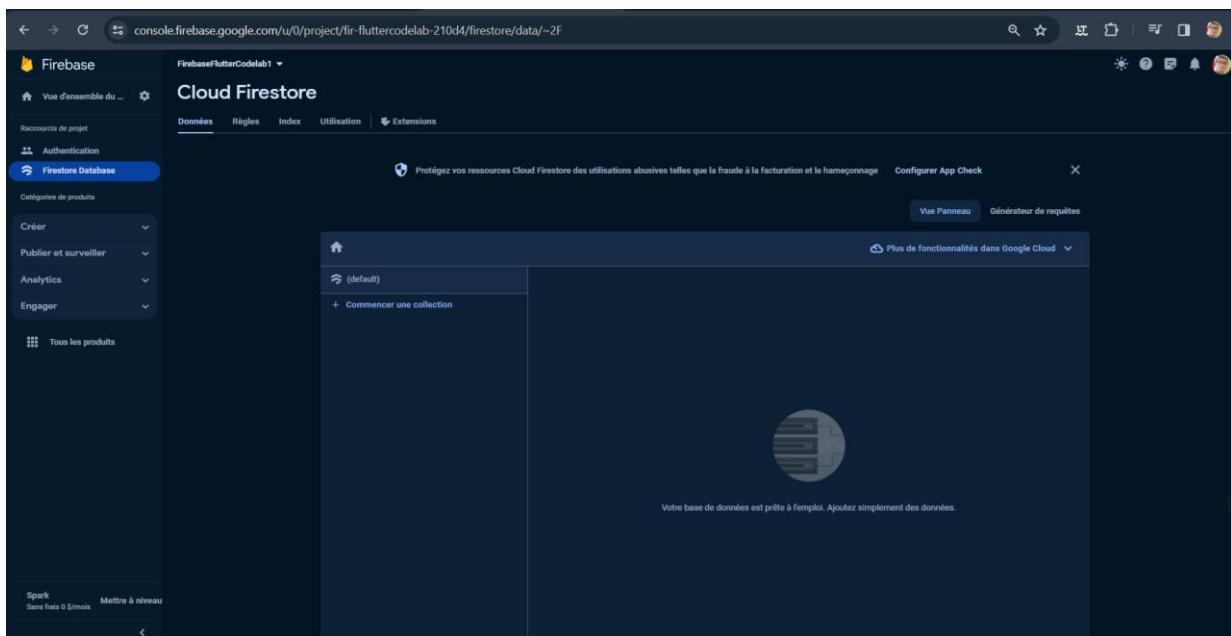
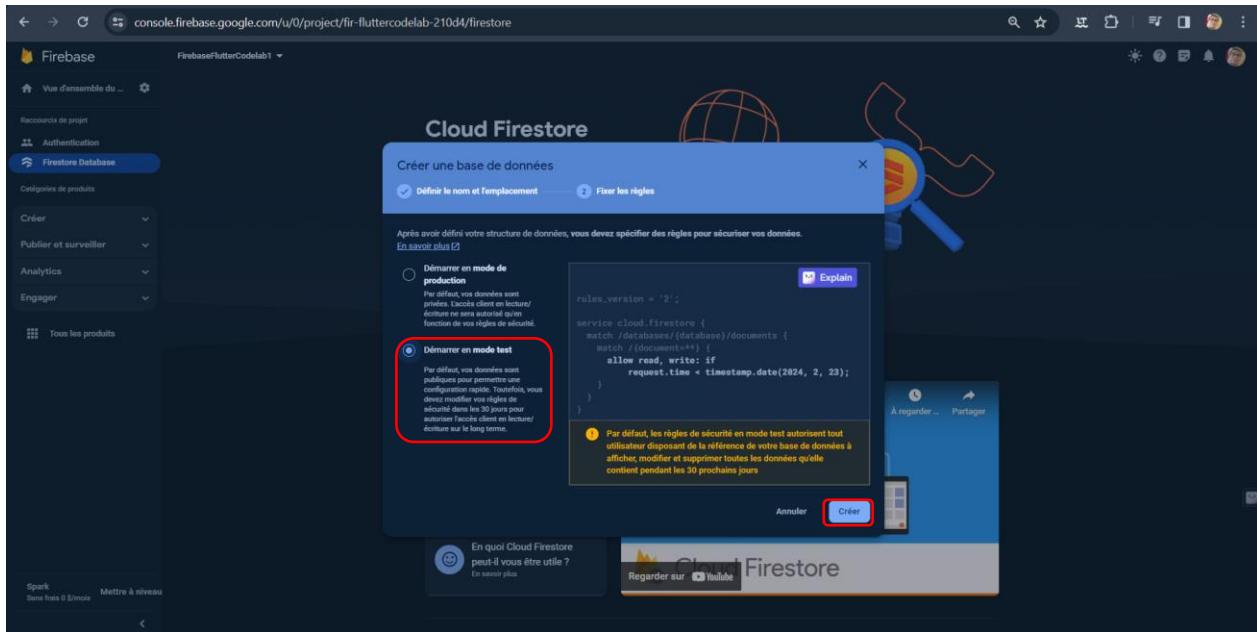
A screenshot of the Firebase console at 'console.firebaseio.google.com/u/0/project/fir-fluttercodelab-210d4/overview'. The left sidebar shows 'Raccourcis de projet' with 'Authentication' and 'Firestore Database' selected. A central banner says 'Lancez-vous en ajoutant Firebase à votre application' with two cartoon characters. Below it, a callout says 'Stockez et synchronisez les données de votre application en quelques millisecondes'. Two cards are shown: 'Authentication' (Authentifiez et gérez les utilisateurs) and 'Cloud Firestore' (Mises à jour en temps réel, requêtes puissantes et scaling automatique). At the bottom, there are buttons for 'Afficher toutes les fonctionnalités' and 'Créer'.

- Cliquez sur Créer une base de données .



- Sélectionnez l'option Démarrer en mode test . Lisez l'avertissement sur les règles de sécurité. Le mode test garantit que vous pouvez écrire librement dans la base de données pendant le développement. Cliquez sur Suivant





## B- Configurer l'application Flutter

1- Obtenez le code de démarrage

[git clone https://github.com/flutter/codelabs.git flutter-codelabs](https://github.com/flutter/codelabs.git)

2- Installer la CLI Firebase

- Connectez-vous à Firebase à l'aide de votre compte Google en exécutant la commande suivante : [firebase login](#)

```
C:\WINDOWS\system32\cmd. x + 
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\CARLINOT>firebase login
Already logged in as nathancarlinot007@gmail.com

C:\Users\CARLINOT>
```

- Testez que la CLI est correctement installée et a accès à votre compte en répertoriant vos projets Firebase. Exécutez la commande suivante : `firebase projects:list`

```
C:\WINDOWS\system32\cmd. x + 
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\CARLINOT>firebase login
Already logged in as nathancarlinot007@gmail.com

C:\Users\CARLINOT>firebase projects:list
/ Preparing the list of your Firebase projects



| Project Display Name    | Project ID                   | Project Number | Resource Location ID |
|-------------------------|------------------------------|----------------|----------------------|
| FirebaseFlutterCodelab  | fir-flutter-codelab-3676e    | 574742714105   | [Not specified]      |
| FirebaseFlutterCodelab1 | fir-fluttercodelab-210d4     | 183196702565   | [Not specified]      |
| FlutterFireUICodeLab    | flutterfire-ui-codelab-410ae | 933614877584   | [Not specified]      |

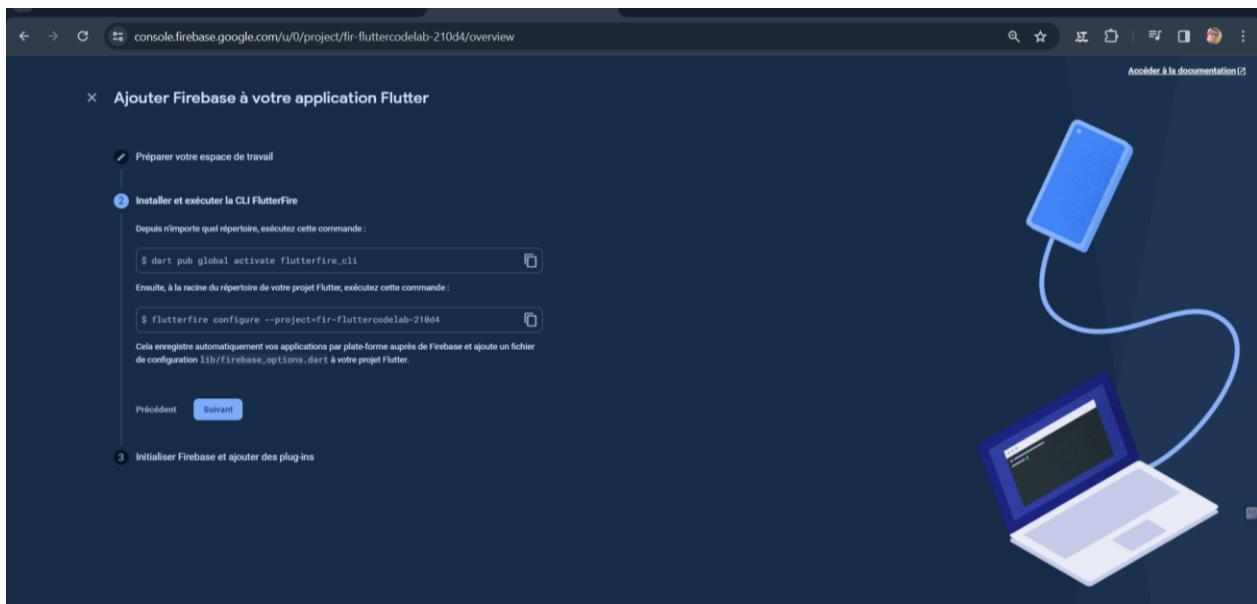


3 project(s) total.

C:\Users\CARLINOT>
```

### 3- Installez la CLI FlutterFire

La CLI FlutterFire est construite sur la CLI Firebase et facilite l'intégration d'un projet Firebase avec votre application Flutter.



Tout d'abord, installez la CLI :

- `dart pub global activate flutterfire_cli`

```

C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [version 10.0.22621.2861]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\CARLINOT>firebase login
Already logged in as nathancarlinot007@gmail.com

C:\Users\CARLINOT>firebase projects:list
/ Preparing the list of your Firebase projects



| Project Display Name    | Project ID                   | Project Number | Resource Location ID |
|-------------------------|------------------------------|----------------|----------------------|
| FirebaseFlutterCodelab  | fir-flutter-codelab-3676e    | 574742714105   | [Not specified]      |
| FirebaseFlutterCodelab1 | fir-fluttercodelab-210d4     | 183196702565   | [Not specified]      |
| FlutterFireUICodelab    | flutterfire-ui-codelab-410ae | 933614877584   | [Not specified]      |



3 project(s) total.

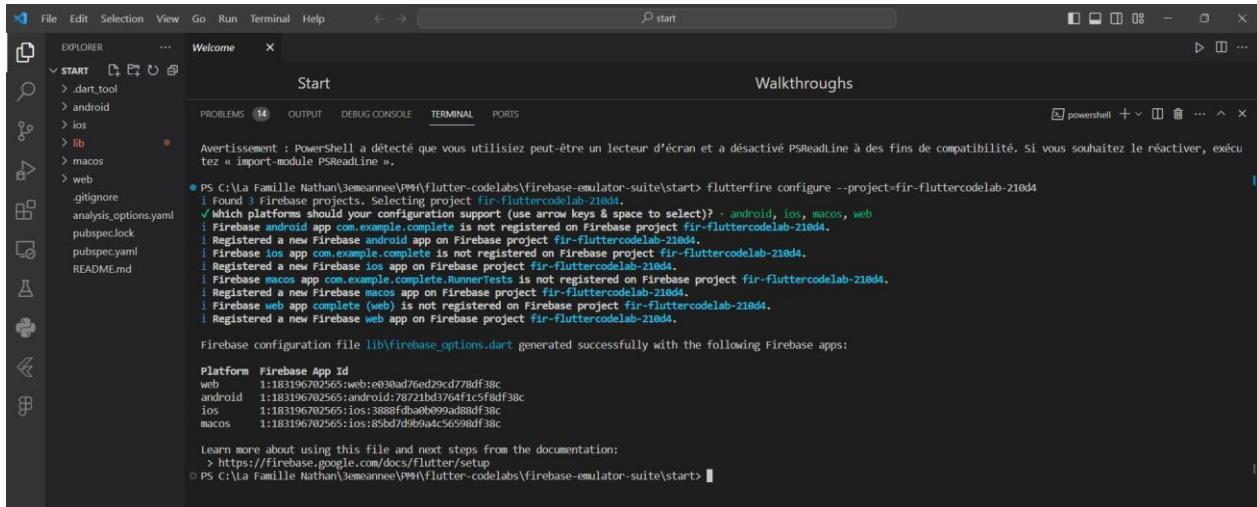
C:\Users\CARLINOT>dart pub global activate flutterfire_cli
'dart' n'est pas reconnu en tant que commande interne
ou externe, un programme exécutable ou un fichier de commandes.

C:\Users\CARLINOT>dart pub global activate flutterfire_cli
Package flutterfire_cli is currently active at version 0.2.7.
The package flutterfire_cli is already activated at newest available version.
To recompile executables, first run 'dart pub global deactivate flutterfire_cli'.
Installed executable flutterfire.
Activated flutterfire_cli 0.2.7.

C:\Users\CARLINOT>

```

- `flutterfire configure --project=fir-fluttercodelab-210d4`



- `android/app/build.gradle`

Sur le defaultConfig, on change le « applicationId » par « com.example.FirebaseFlutterCodelab »

```

build.gradle
...
    defaultConfig {
        // TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html).
        applicationId "com.example.FirebaseFlutterCodelab"
        ...
    }
}

```

#### 4- Ajouter des packages Firebase à l'application Flutter

La dernière étape de configuration consiste à ajouter les packages Firebase pertinents à votre projet Flutter. Dans le terminal, assurez-vous que vous êtes à la racine du projet Flutter à flutter-codelabs/firebase-emulator-suite/start . Ensuite, exécutez les trois commandes suivantes :

- flutter pub add firebase\_core

```

flutter pub add firebase_core
Resolving dependencies...
+ firebase_core 2.24.2
+ firebase_core_platform_interface 5.0.0
+ firebase_core_web 2.10.0
  ...
  ...
Changelog 5 dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.

```

- flutter pub add firebase\_auth

```

flutter pub add firebase_auth
Resolving dependencies...
+ firebase_auth 4.16.0
+ firebase_auth_platform_interface 7.0.9
+ firebase_auth_web 2.10.0
  ...
  ...
Changelog 6 dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.

```

- flutter pub add cloud\_firestore

The screenshot shows the VS Code interface with the following details:

- File Explorer (left):** Shows the project structure with files like build.gradle, firebase\_options.dart, app\_state.dart, entry.dart, and main.dart.
- Terminal (bottom):** Displays the command: `PS C:\La Famille Nathan\Semaine04\Flutter-codelabs\firebase-emulator-suite\start> flutter pub add cloud_firestore`. It also shows dependency resolution output:
  - Resolving dependencies...
  - + cloud\_firestore 4.14.0
  - + cloud\_firestore\_platform\_interface 6.1.0
  - + flutterfire\_core 2.10.0
  - go router 0.1.3 (0.0.1 available)
  - js 0.6.7 (0.7.0 available)
  - matcher 0.12.16 (0.17.1+1 available)
  - material\_color\_utilities 0.5.0 (0.6.0 available)
  - meta 1.10.0 (1.11.0 available)
  - path 1.8.3 (1.9.0 available)
  - test\_api 0.6.1 (0.7.0 available)
  - web 0.3.0 (0.4.2 available)
- Problems (center):** Shows 16 errors.
- Output (center):** Shows the output of the dependency resolution process.
- Debug Console (center):** Not currently active.
- Terminal (center):** Active, showing the command and its output.
- Ports (center):** Not currently active.
- PowerShell (right):** Available as a terminal option.

## C- Activation des émulateurs Firebase

## 1- Démarrez les émulateurs

Cette sortie vous indique quels émulateurs sont en cours d'exécution et où vous pouvez aller pour voir les émulateurs. Tout d'abord, consultez l'interface utilisateur de l'émulateur sur localhost:4000 .

The screenshot shows the Firebase Emulator Suite interface with the following details:

- Authentication emulator:** Status On (green), Port number 9099, Go to emulator button.
- Firebase emulator:** Status On (green), Port number 8080, Go to emulator button.
- Realtime Database emulator:** Status Off (grey), Port number N/A, Go to emulator button.
- Functions emulator:** Status Off (grey), Port number N/A, View logs button.
- Storage emulator:** Status Off (grey), Port number N/A, Storage button.
- Hosting emulator:** Status Off (grey), Port number N/A.

A banner at the top states: "Remember this is a local environment. Visit the production version of flutter-firebase-codelab-d6b79 in the console." with buttons for "View project" and "Dismiss".

## 2- L'émulateur Firebase Auth

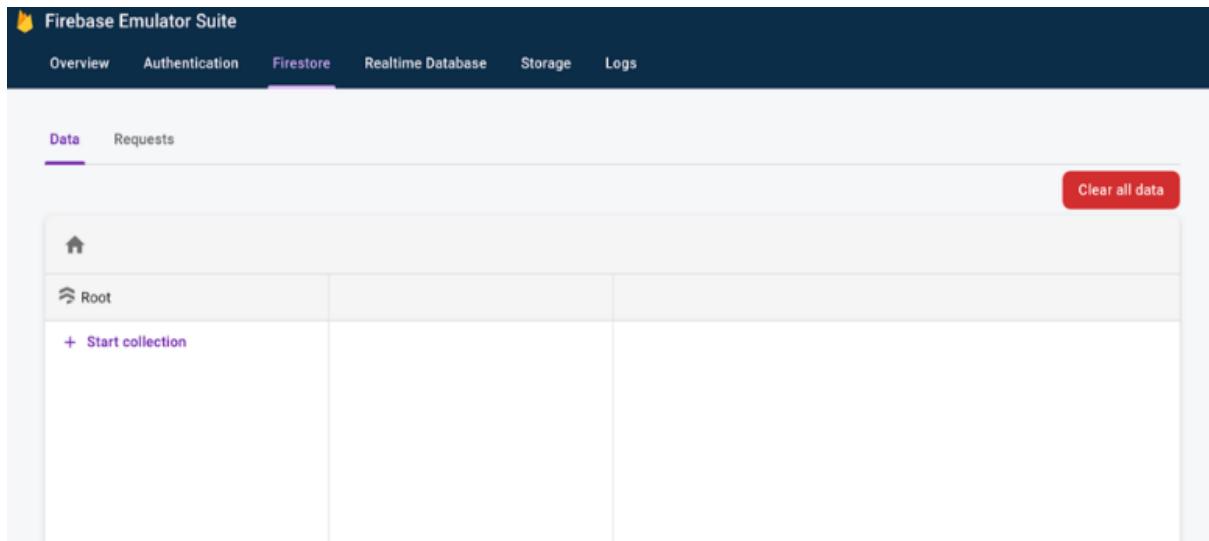
Le premier émulateur que vous utiliserez est l'émulateur d'authentification. Commencez avec l'émulateur Auth en cliquant sur « Aller à l'émulateur » sur la carte d'authentification dans l'interface utilisateur, et vous verrez une page qui ressemble à ceci :

The screenshot shows the Authentication emulator interface with the following details:

- Search bar:** "Search by user UID, email address, phone number, or display name".
- Add user button:** A purple button with a person icon and the text "Add user".
- User table:** A table with columns: Identifier, Provider, Created, Signed In, and User UID. It currently displays the message "No users for this project yet".

### 3- Lire et écrire des données sur l'émulateur Firestore

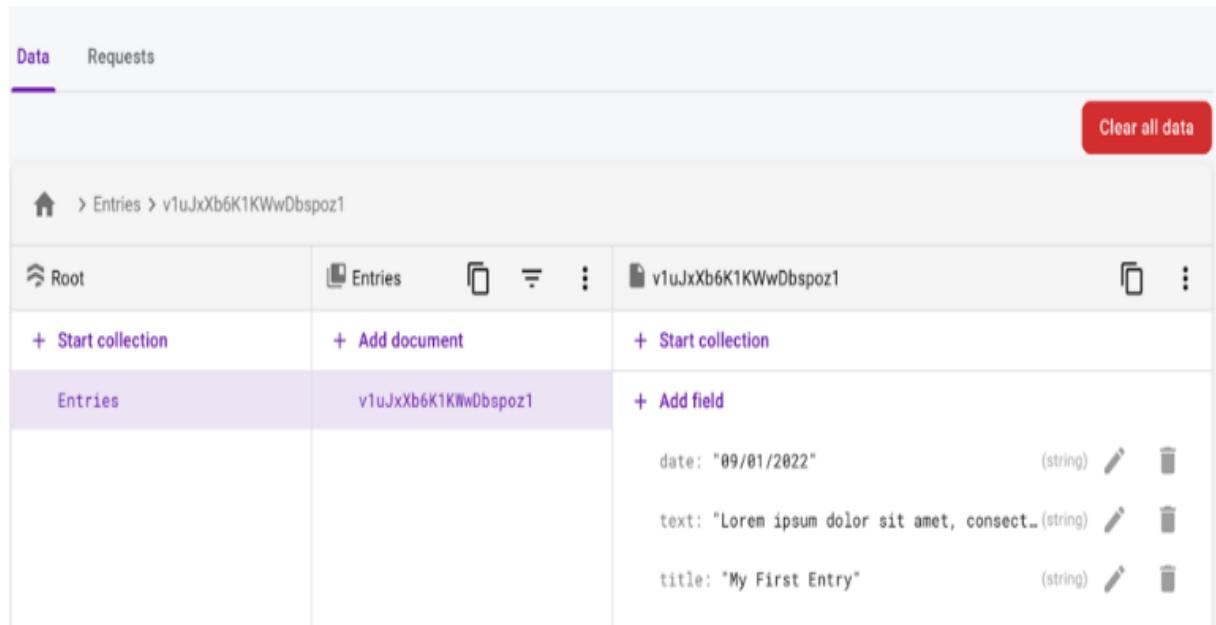
Tout d'abord, consultez l'émulateur Firestore. Sur la page d'accueil de l'interface utilisateur de l'émulateur ( localhost:4000 ), cliquez sur "Aller à l'émulateur" sur la carte Firestore.



The screenshot shows the Firebase Emulator Suite interface. At the top, there's a navigation bar with tabs: Overview, Authentication, Firestore (which is highlighted in purple), Realtime Database, Storage, and Logs. Below the navigation bar, there are two tabs: Data (which is selected and highlighted in blue) and Requests. In the main content area, there's a 'Clear all data' button. Under the 'Data' tab, the 'Root' collection is listed, and there's a '+ Start collection' button below it. The interface has a clean, modern design with a light gray background and white cards for data entries.

### 4- L'onglet requêtes dans l'émulateur Firestore

Dans l'interface utilisateur, accédez à l'émulateur Firestore et regardez l'onglet "Données". Vous devriez voir qu'il y a maintenant une collection à la racine de votre base de données appelée "Entries". Cela devrait avoir un document contenant les mêmes informations que vous avez saisies dans le formulaire.



The screenshot shows the same interface as the previous one, but with the 'Requests' tab selected. The 'Root' collection is still visible, but the 'Entries' collection is now the active item. A specific document, 'v1uJxXb6K1KWwDbspoz1', is selected. The document details are shown in a table:

Entries	v1uJxXb6K1KWwDbspoz1
+ Start collection	+ Add document
Entries	v1uJxXb6K1KWwDbspoz1
	+ Add field
	date: "09/01/2022" (string) <span style="color: #007bff;">Edit</span> <span style="color: #dc3545;">Delete</span>
	text: "Lorem ipsum dolor sit amet, consectetur..." (string) <span style="color: #007bff;">Edit</span> <span style="color: #dc3545;">Delete</span>
	title: "My First Entry" (string) <span style="color: #007bff;">Edit</span> <span style="color: #dc3545;">Delete</span>

## 5- Requêtes d'émulateur Firestore

Time	Method	Path
11:25:42	✓ LIST	/Entries/*
11:26:19	✓ LIST	/Entries/*
11:27:10	✓ CREATE	/Entries/v1uJxXb6K1KwDbspoz1
11:27:10	✓ LIST	/Entries/*
11:27:10	✓ LIST	/Entries/*

Vous pouvez cliquer sur l'un de ces éléments de liste et voir de nombreuses informations utiles. Cliquez sur l'élément de liste CREATE qui correspond à votre demande pour créer une nouvelle écriture de journal. Vous verrez un nouveau tableau qui ressemble à ceci :

The screenshot shows the Firestore Emulator tool interface. On the left, a list of requests is shown, with the fifth item selected: "11:27:10 ✓ CREATE /Entries/v1uJxXb6K1KwDbspoz1". To the right, a detailed view of this request is displayed.

**Detailed information:**

- request.auth
- request.method
- "create" (string)

**request.path:** /databases/(default)/documents/Entries/(path)

**request.resource:** ...name...: /databases/(default)/docs/(path)

**Code Snippet (request body):**

```
1 rules_version = '2';
2 service cloud.firestore {
3     match /databases/{database}/documents {
4         match /(document=*) {
5             allow read, write: if
6                 request.time < timestamp.date(2022, 8, 9);
7         }
8     }
9 }
```

Resultat :

Welcome back, Dash! DEBUG

Title  

---

Date (DD/MM/YYYY):  

---

Text  

---

**Submit**

Welcome back, Dash! DEBUG

**My Second Entry**

Date: 01/02/1990

'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.';

## D- Exporter et importer des données dans l'émulateur

### 1- Exporter les données de l'émulateur

.emulators\_data est un argument qui indique à Firebase où exporter les données. Si le répertoire n'existe pas, il est créé. Vous pouvez utiliser n'importe quel nom pour ce répertoire.

*firebase emulators:export ./emulators\_data*

### 2- Importer des données d'émulateur

exécutez la commande emulators:start que vous avez déjà vue, mais avec un indicateur lui indiquant quelles données importer :

*firebase emulators:start --import ./emulators\_data*

### 3- Exporter automatiquement les données lors de la fermeture des émulateurs

Lorsque vous démarrez vos émulateurs, exécutez la commande emulators:start avec deux indicateurs supplémentaires.

*firebase emulators:start --import ./emulators\_data --export-on-exit*

Vos données seront désormais enregistrées et rechargées chaque fois que vous travaillerez avec les émulateurs de ce projet. Vous pouvez également spécifier un répertoire différent comme argument de –export-on-exit flag , mais il s'agira par défaut du répertoire transmis à –import .