Flutter
MADRID

Flutter ♥ Firebase

10 de Febrero. Auditorio – Google Campus, Madrid

Óscar Martín

Emprendedor | Desarrollador

EDUCACIÓN

Universidad Rey Juan Carlos

EXPERIENCIA LABORAL

EMPRENDIMIENTOS

Leaders ToBe

STARTUPTEC

¿Qué aprenderemos?

Repaso de conceptos básicos

Creación de proyecto en Firebase

Configuración iOS Firebase

Autenticación Firebase

Future + llamadas API

Stream + Firestore

Repaso Conceptos Básicos

Routes

```
return MaterialApp(
  title: 'Flutter Demo',
  theme: ThemeData(
    primarySwatch: Colors.blue,
  ), // ThemeData
  routes: <String, WidgetBuilder>{
    // Set routes for using the Navigator.
    '/home': (BuildContext context) => new HomePage(),
    '/home/series_home': (BuildContext context) => new SeriesHomePage(),
    '/login': (BuildContext context) => new LoginPage()      You, 8 hours ago • Cleaning code
  },
```

```
Navigator.of(context).pushReplacementNamed('/login');
```

BottomBar

```
bottomNavigationBar: BottomNavigationBar(
  onTap: onTabTapped,
  currentIndex: _currentIndex,
  items: [
    BottomNavigationBarItem(
      icon: new Icon(Icons.search),
      title: new Text('Search'),
    ), // BottomNavigationBarItem
    BottomNavigationBarItem(
      icon: new Icon(Icons.list),
      title: new Text('TO-DO'),
    ), // BottomNavigationBarItem
  ],
```

```
body: IndexedStack(
  index: _currentIndex,
  children: _tabPages,
),
```


```
final List<Widget> _tabPages = [new SeriesSearchPage(), new SeriesSavedPage()];
```

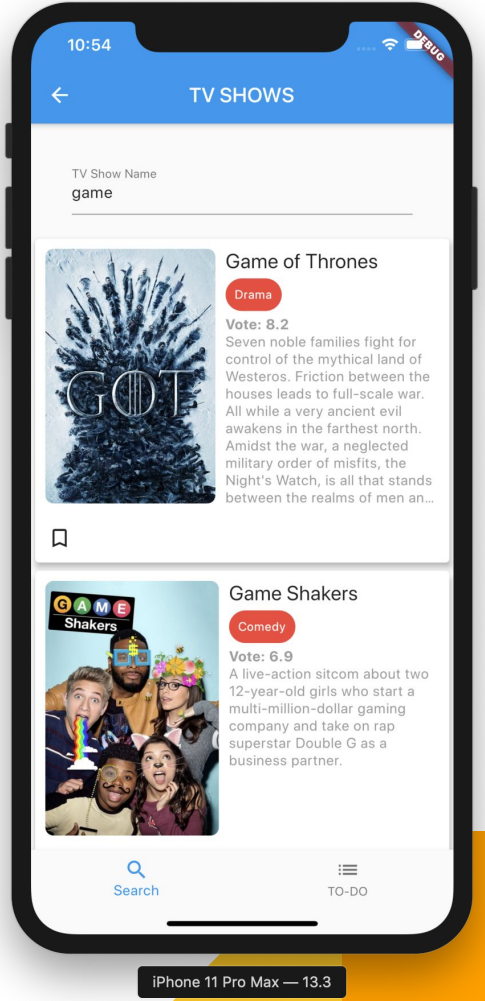
FutureBuilder

```
home: FutureBuilder<FirebaseUser>(
  future: UserController().getUserAuthenticated(),
  builder: (context, AsyncSnapshot<FirebaseUser> snapshot) {
    if (snapshot.connectionState == ConnectionState.done) {
      if (snapshot.error != null) {
        return Text(snapshot.error.toString());
      }
      return snapshot.hasData ? HomePage() : LoginPage();
    } else {
      return LoadingPage();
    }
  },
), // FutureBuilder
```

ListView

```
// Build the list
Widget buildSearchList() {
  return FutureBuilder(
    builder: (context, projectSnap) {      You, 8 hours ag
    future: SeriesAPIController().getSeries(_query),
  ); // FutureBuilder
}
```

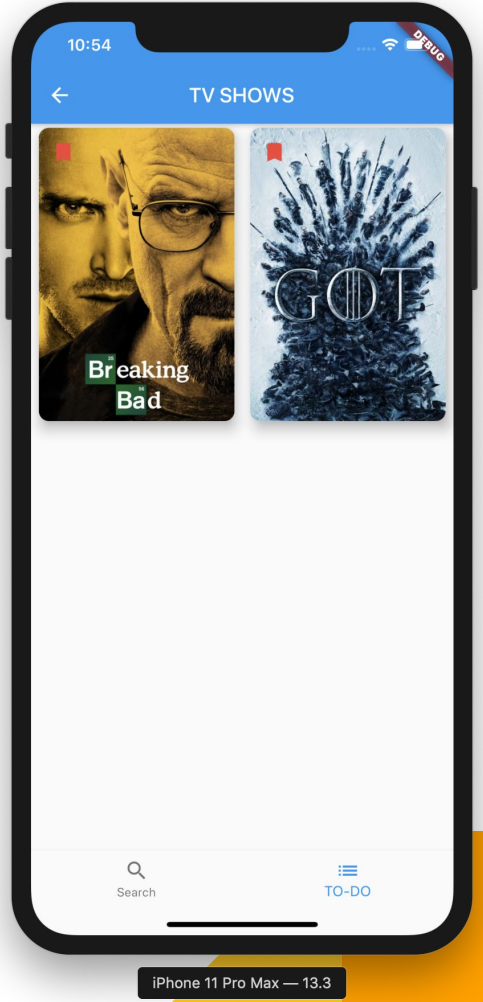
```
searchSeries = projectSnap.data;
if (searchSeries.length > 0) {
  return
  Expanded(
    child: ListView.builder(
      itemCount: searchSeries.length,
      itemBuilder: (context, index) {
        Serie serie = searchSeries[index];
        return new SerieCard(
                serie: serie,
                onToDoChanged: (Serie serie) async {
                  await SeriesController().addRemoveSerie(serie);
                });          // SerieCard
      }) // ListView.builder
  ); // Expanded
```



GridView

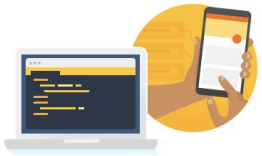
```
Widget build(BuildContext context) {
  if(userId == null) return Container();
  return StreamBuilder(
    builder: (context, projectSnap) {…
    stream: SeriesAPIController().savedSeriesFor(userId),
  ); // StreamBuilder
}
```

```
if (projectSnap.data.length > 0) {
  return GridView.builder(
      gridDelegate: new SliverGridDelegateWithFixedCrossAxisCount(
          crossAxisSpacing: 0,
          childAspectRatio: 0.7,
          mainAxisSpacing: 5,
          crossAxisCount: 2), // SliverGridDelegateWithFixedCrossAxisCount
      itemCount: projectSnap.data.length,
      itemBuilder: (context, index) {
        Serie serie = projectSnap.data[index];
        return new SavedSerieCard(
            serie: serie,
            onToDoChanged: (Serie serie) {
              setState(() {
                SeriesController().addRemoveSerie(serie);
              });
            }); // SavedSerieCard
      }); // GridView.builder
} else {
```



Creando Proyecto Firebase

A comprehensive app development platform



Compila mejores apps

Cloud Firestore
Almacena y sincroniza los datos de tu app a escala global

ML Kit BETA
Aprendizaje automático para desarrolladores de apps para dispositivos móviles

Cloud Functions
Ejecuta código de back-end para dispositivos móviles sin administrar servidores

Authentication
Autentica usuarios de forma simple y segura

Hosting
Entrega recursos de aplicaciones web con velocidad y seguridad

Cloud Storage
Almacena y envía archivos a la escala de Google

Realtime Database
Almacena y sincroniza datos de app en milisegundos

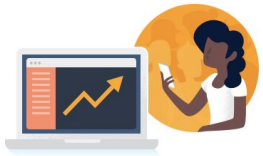
Mejora la calidad de las apps

Crashlytics
Prioriza y soluciona problemas con informes de fallas potentes y en tiempo real

Performance Monitoring
Obtén estadísticas sobre el rendimiento de tu app

Test Lab
Prueba la app en dispositivos alojados en Google

App Distribution BETA
Distribute pre-release versions of your app to your trusted testers



Haz crecer tu negocio

In-App Messaging BETA
Usa mensajes contextuales para interactuar con los usuarios activos de la app

Google Analytics
Obtén datos de analítica ilimitados sobre tu app sin cargo

Predictions
Smart user segmentation based on predicted behavior

A/B Testing BETA
Optimiza la experiencia que ofrece tu app a través de experimentos

Cloud Messaging
Envía notificaciones y mensajes orientados

Remote Config
Modifica tu app sin implementar una versión nueva

Dynamic Links
Usa vínculos directos con atribución para impulsar el crecimiento

Crear un proyecto(paso 1 de 3)

Comencemos con el nombre de tu proyecto

Ingresa el nombre de tu proyecto

Add Firebase to one of your existing Google Cloud Platform projects Learn more

TODO APP

Continuar

Crear un proyecto(paso 2 de 2)

para tu proyecto de Firebase

Google Analytics es una solución de análisis ilimitada y gratuita que permite usar la orientación, los informes y otras funciones en Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions y Cloud Functions.

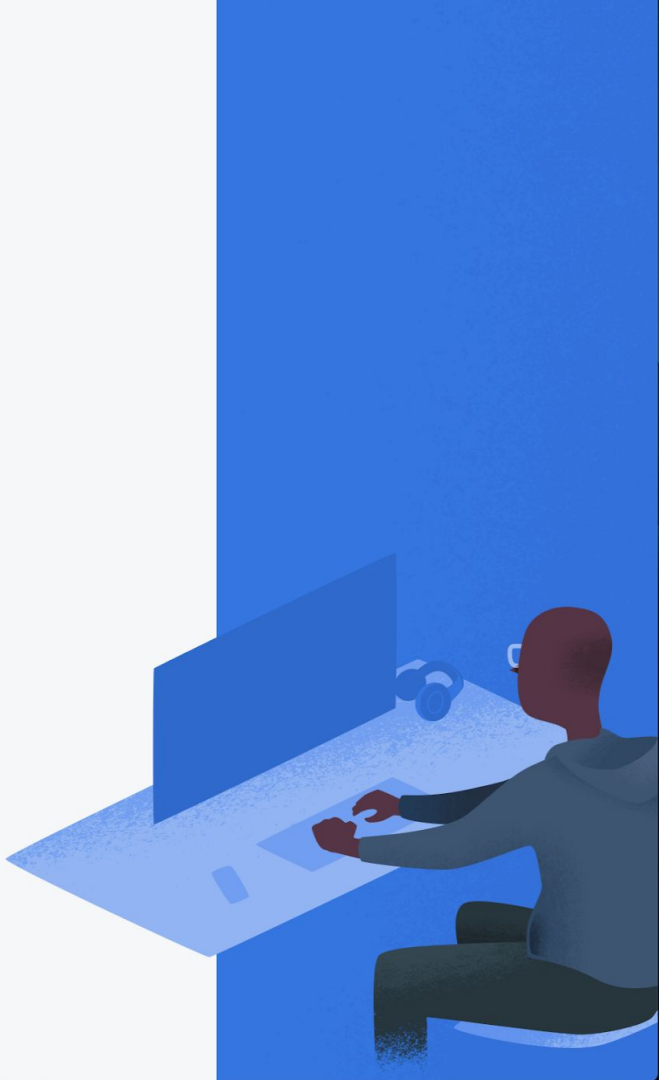
Google Analytics habilita las siguientes funciones:

- ~~A/B Testing~~
- ~~Segmentación de usuarios y orientación a ellos en los productos de Firebase~~
- ~~Predicción del comportamiento de los usuarios~~
- ~~Usuarios que no experimentaron fallas~~
- ~~Activadores de Cloud Functions basados en eventos~~
- ~~Informes ilimitados y gratuitos~~

Habilitar Google Analytics para este proyecto
Opción recomendada

Anterior

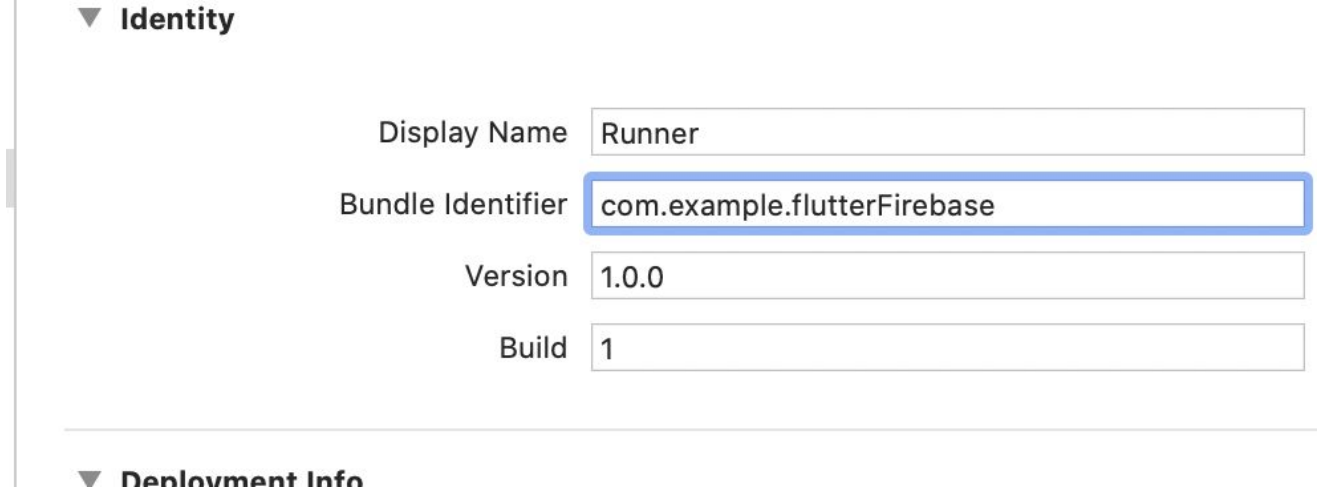
Crear proyecto

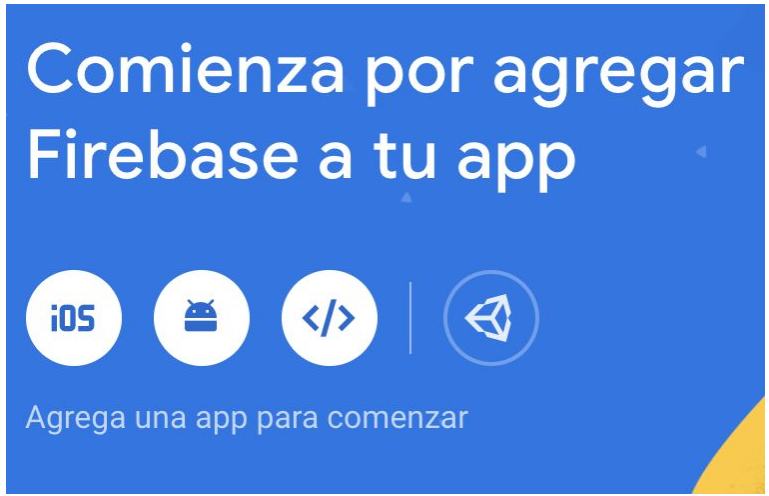


Dependencias en Flutter

```
cloud_firestore:
firebase_auth:      You, a
```

Configuración iOS Firebase

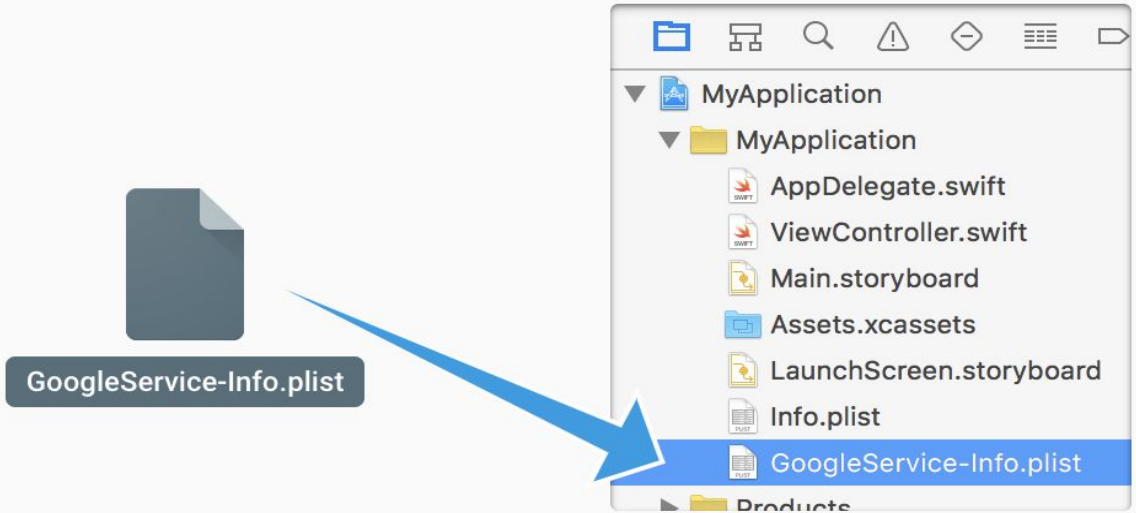

Identity
Display Name
Runner
Bundle Identifier
com.example.flutterFirebase
Version
1.0.0
Build
1
Deployment Info


Comienza por agregar Firebase a tu app
iOS
Agrega una app para comenzar

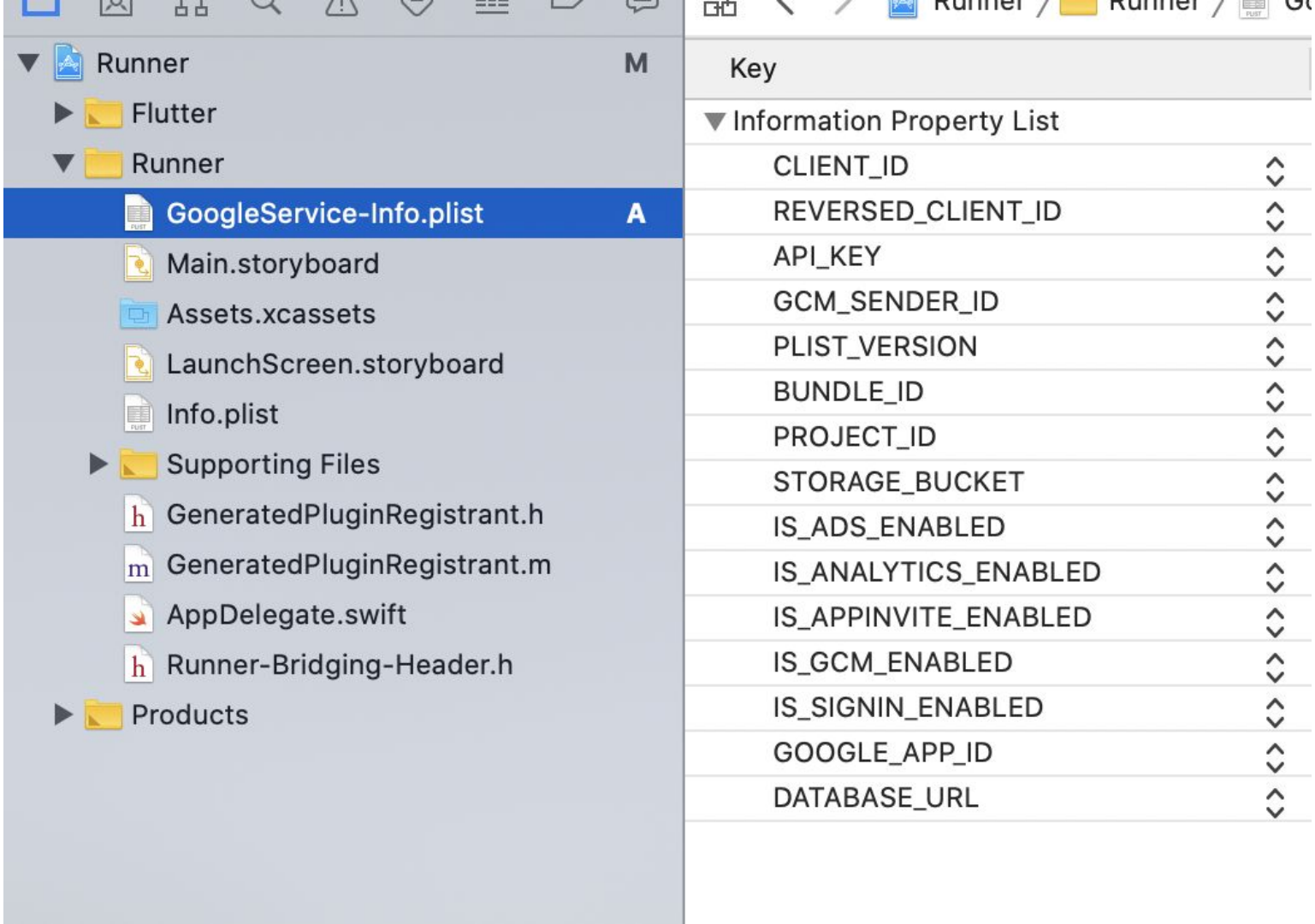
2 **Descargar archivo de configuración** Instrucciones para Xcode a continuación | Unity C++

Descargar GoogleService-Info.plist

Coloca el archivo GoogleService-Info.plist que acabas de descargar en el directorio raíz de tu proyecto de Xcode y agrégalo a todos los destinos.



Anterior Siguiente


Runner
M
Flutter
Runner
GoogleService-Info.plist
A
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Supporting Files
GeneratedPluginRegistrant.h
GeneratedPluginRegistrant.m
AppDelegate.swift
Runner-Bridging-Header.h
Products
Runner
Runner
Key
Information Property List
CLIENT_ID
REVERSED_CLIENT_ID
API_KEY
GCM_SENDER_ID
PLIST_VERSION
BUNDLE_ID
PROJECT_ID
STORAGE_BUCKET
IS_ADS_ENABLED
IS_ANALYTICS_ENABLED
IS_APPINVITE_ENABLED
IS_GCM_ENABLED
IS_SIGNIN_ENABLED
GOOGLE_APP_ID
DATABASE_URL

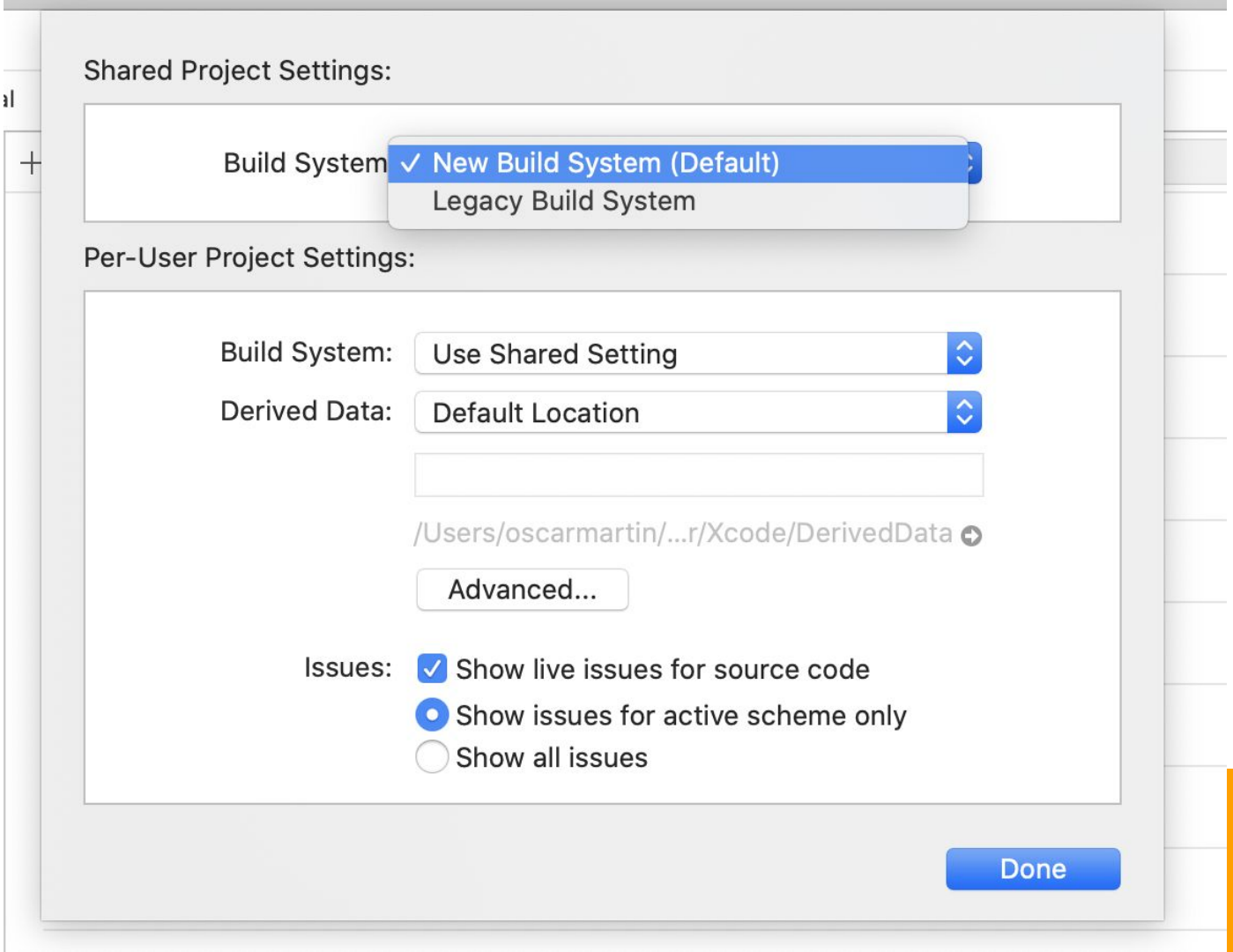
Configuración iOS Firebase: extras

Gitignore

```
#Ocultar info plist con firebase config
**/ios/Runner/GoogleService-Info.plist
/lib/config/constants.dart
      You, a few seconds ago • Uncommitted changes
```

Instalar POD

```
→  ios git:(master) ✗ pod init
→  ios git:(master) ✗ pod install
Analyzing dependencies
Downloading dependencies
Installing BoringSSL-GRPC (0.0.3)
```


Shared Project Settings:
Build System
✓ New Build System (Default)
Legacy Build System
Per-User Project Settings:
Build System:
Use Shared Setting
Derived Data:
Default Location
/Users/oscarmartin/...r/Xcode/DerivedData
Advanced...
Issues:
Show live issues for source code
Show issues for active scheme only
Show all issues
Done

Autenticación Firebase

Almacena y sincroniza datos de app en milisegundos

Authentication

Autentica y administra usuarios

Proveedores de inicio de sesión

Proveedor	Estado
Correo electrónico/contraseña	Habilitada
Teléfono	Inhabilitado
Google	Inhabilitado
Play Juegos	Inhabilitado
Game Center Beta	Inhabilitado
Facebook	Inhabilitado
Twitter	Inhabilitado

Usuarios | Método de inicio de sesión | Plantillas | Uso

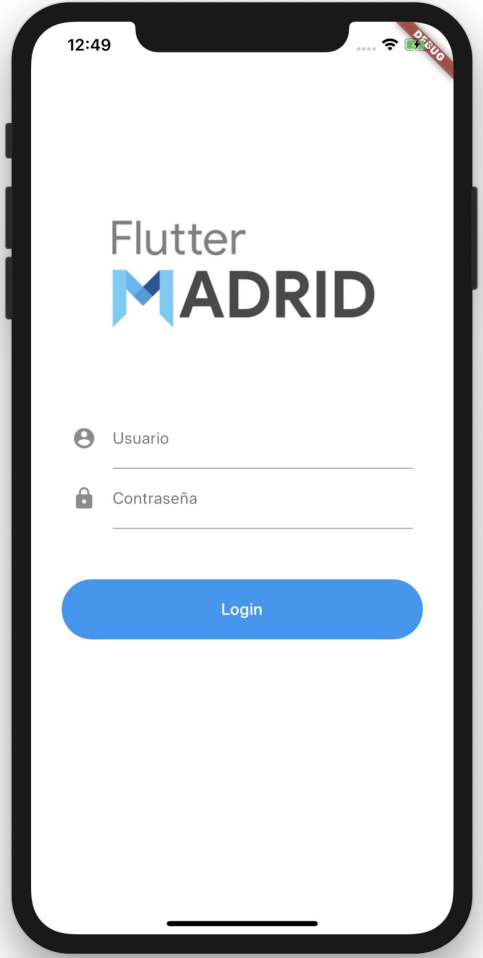
Buscar por dirección de correo electrónico, número de teléfono o UID de usuario

Añadir usuario

Identificador	Proveedores	Fecha de creación	Inicio de sesión	UID de usuario
martinm.oscar@gmail.com	✉	2 feb. 2020	2 feb. 2020	SoGDHUENRFO0apksxv6h705CieF2

Filas por página: 50 1-1 de 1

Página de Login


12:49
DEBUG
Flutter
MADRID
Usuario
Contraseña
Login
iPhone 11 Pro Max — 13.3

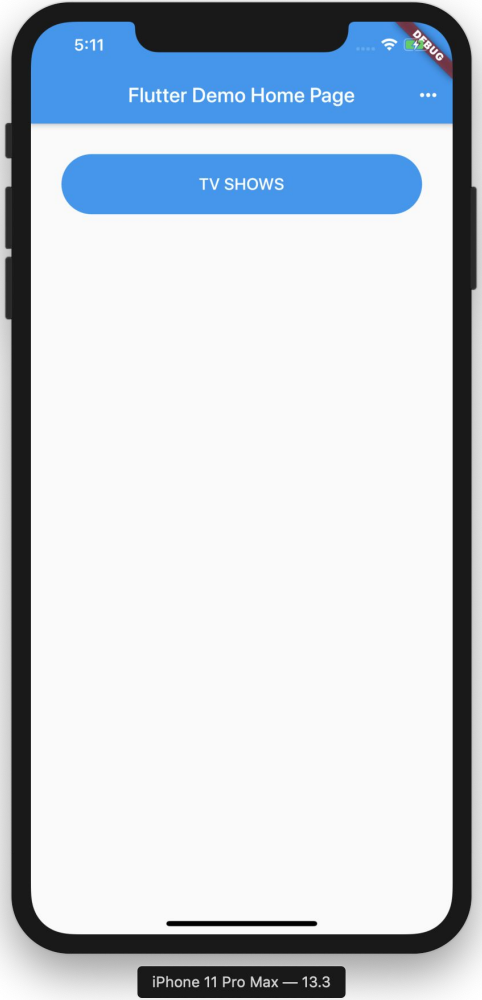
```
void _signInWithEmailAndPassword() async {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  FirebaseUser user;
  try {
    user = (await _auth.signInWithEmailAndPassword(
      email: _emailController.text,
      password: _passwordController.text,
    ))
        .user;
  } catch (e) {}

  if (user != null) {
    Navigator.of(context).pushReplacementNamed('/home');
  } else {
    showAlertDialog(context);
  }
}
```

Persistencia de login

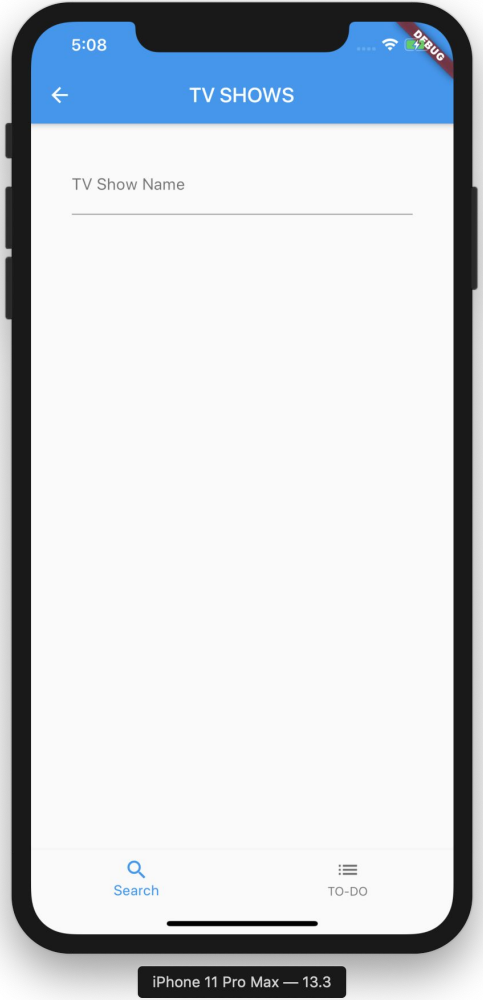
```
Future<FirebaseUser> getUserAuthenticated() {
  return FirebaseAuth.instance.currentUser();
}
```

```dart
Future<bool> signOut() async {      You, 3 minutes ago • I
  try {
    await FirebaseAuth.instance.signOut();
  } catch (e) {
    // Error doing logout
    return false;
  }
  return true;
}
```

Buscador de Series




```
onPressed: () {
  Navigator.of(context).pushNamed('/home/series_home');
},
child: Text(
```




```
), // AppBar
body: _children[_currentIndex],        You, a few seconds ag
bottomNavigationBar: BottomNavigationBar(
  onTap: onTabTapped,
  currentIndex: _currentIndex,
  items: [
    BottomNavigationBarItem(
      icon: new Icon(Icons.search),
      title: new Text('Search'),
    ), // BottomNavigationBarItem
    BottomNavigationBarItem(
      icon: new Icon(Icons.list),
      title: new Text('TO-DO'),
    ), // BottomNavigationBarItem
  ],
), // BottomNavigationBar
```

THE MOVIE DB API

API Overview

Our API is available for everyone to use. A TMDb user account is required to request an API key. Professional users are approved on a per application basis.

As always, you must attribute TMDb as the source of your data. Please be sure to read the API FAQ.

API Documentation

To view all the methods available, you should head over to developers.themoviedb.org. Everything outlined on this page is simply a high level overview to help you understand what is available.

Finding Data

There are 3 ways to search for and find movies, TV shows and people on TMDb. They're outlined below.

`/search` - Text based search is the most common way. You provide a query string and we provide the closest match. Searching by text takes into account all original, translated, alternative names and titles.

`/discover` - Sometimes it useful to search for movies and TV shows based on filters or definable values like ratings, certifications or release dates. The discover method make this easy. For some example queries, and to get an idea about the things you can do with discover, take a look here.

Settings

- Edit Profile
- Account Settings
- Notification Settings
- Blocked Users
- Import List
- Sharing Settings
- Social Settings
- Connected Apps
- API
- Delete Account

API Details Directory Sessions Stats

TMDb offers a powerful API service that is free to use as long as you properly attribute us as the sou
the data and/or images you use. You can find the logos for attribution here.

Documentation

Our primary documentation is located at developers.themoviedb.org.

Support

If you have questions or comments about the information covered here, please create a post on ou
support forums.

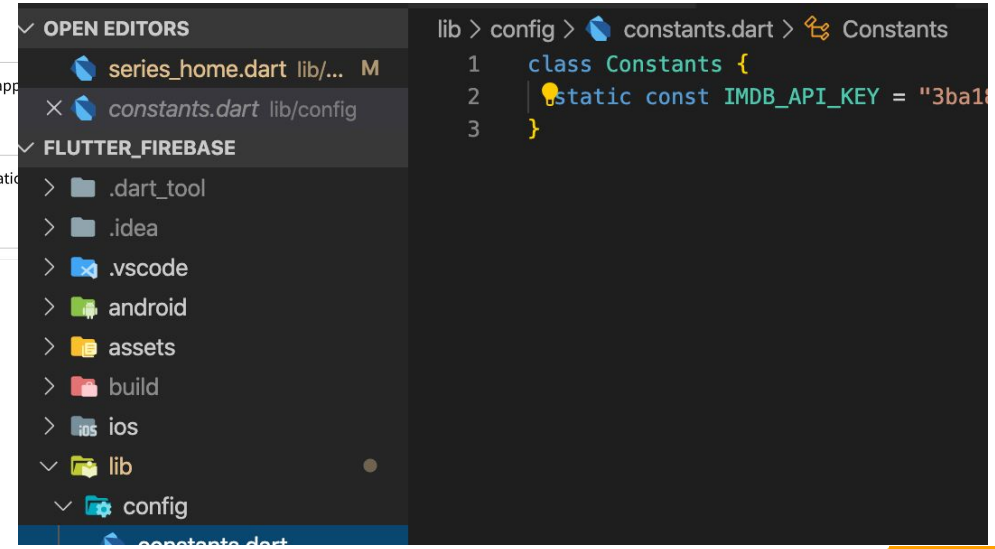
API Details

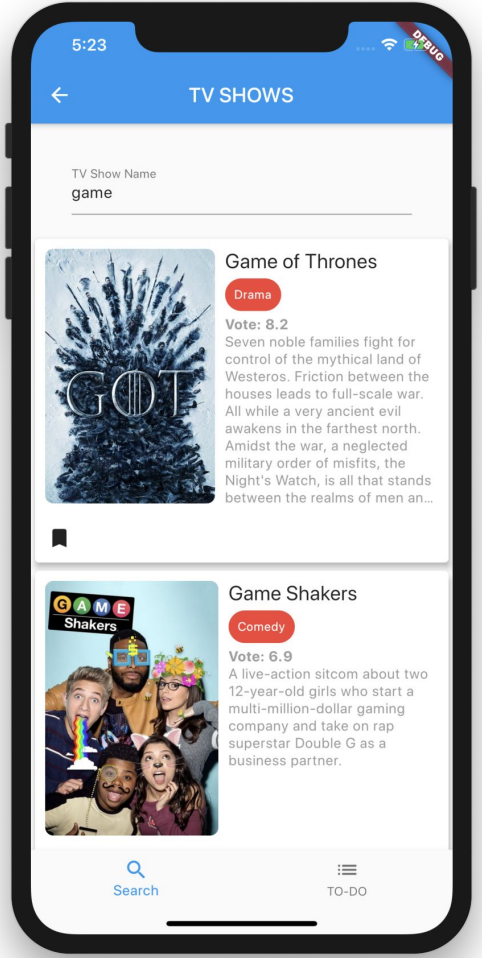
If you'd like to edit the details of your app

App Directory

Once you have completed your applicatio

API Key (v3 auth)






```dart
Future<List<Serie>> getSeries(String name) async {
  if (name == "") return new List<Serie>();
  String fixSearchUrl = "https://api.themoviedb.org/3/search/tv?";
  String apiKeyUrl = "api_key=" + Constants.IMDB_API_KEY;
  String parametersSearchUrl = "&language=en-US&query=" + name + "&page=1";
  String fullUrl = fixSearchUrl + apiKeyUrl + parametersSearchUrl;

  var response = await http.get(fullUrl);
  var genres = await getGenres();

  //Parsing results
  Map<String, dynamic> parsedJson = json.decode(response.body);
  List results = parsedJson["results"];
  var series = new List<Serie>();
  var partialImageUrl = "https://image.tmdb.org/t/p/w500";
  for (Map result in results) {
    String serieName = result["original_name"];
    String posterPath = result["poster_path"];
    String backPath = result["backdrop_path"];
    int id = result["id"];
    double voteAverage = result["vote_average"].toDouble();
    String overview = result["overview"];

    // Genre
    List<int> genreIds = result["genre_ids"].cast<int>();
    String genreName = "";
    if (genreIds.length > 0) {
      int genreId = genreIds[0];

      for (Map item in genres) {
        if (item["id"] == genreId) {
          genreName = item["name"];
        }
      }
    }

    var serie = new Serie();
```

Guardar en BBDD

Local(offline) vs Cloud Firestore(online and offline)

BBDD local

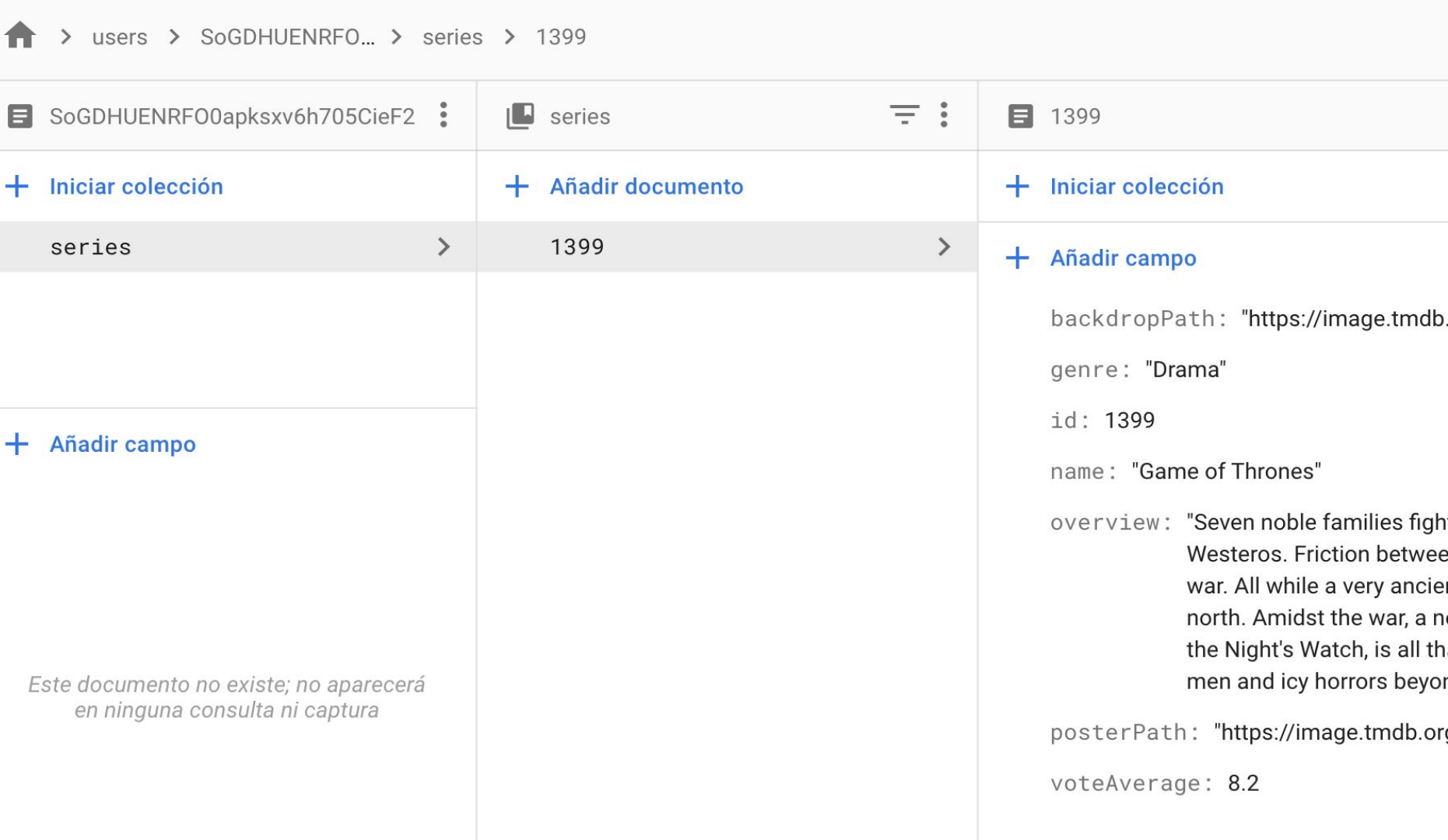
```
Serie(
    {this.id,
    this.name,
    this.posterPath,
    this.backdropPath,
    this.isInToDo,
    this.voteAverage,
    this.genre,
    this.overview});

factory Serie.fromMap(Map<String, dynamic> json) => new Serie(
    id: json["id"],
    name: json["name"],
    isInToDo: true,
    posterPath: json["posterPath"],
    backdropPath: json["backdropPath"],
    voteAverage: json["voteAverage"],
    genre: json["genre"],
    overview: json["overview"]);

Map<String, dynamic> toMap() => {
      "id": id,
      "name": name,
      "posterPath": posterPath,
      "backdropPath": backdropPath,
      "voteAverage": voteAverage,
      "genre": genre,
      "overview": overview
    };
}
```

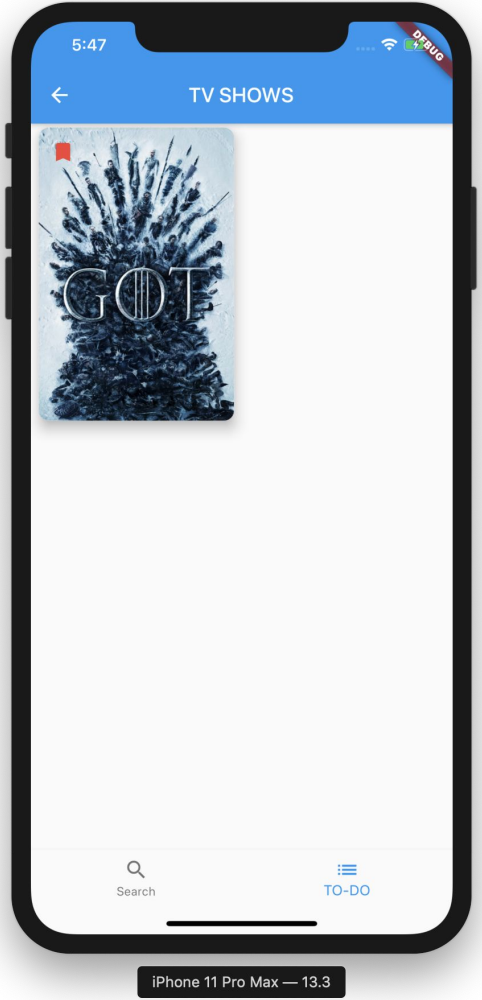
```
Future<Database> get database async {
  if (_database != null) return _database;
  _database = await getDatabaseInstance();
  return _database;
}

Future<Database> getDatabaseInstance() async {
  Directory directory = await getApplicationDocumentsDirectory();
  String path = join(directory.path, "series.db");
  return await openDatabase(path,
      version: 1, onUpgrade: _onUpgrade, onCreate: _onCreate);
}
    You, 43 minutes ago • Adding series home, search and saved page
Future _onCreate(Database db, int) async {
  await db.execute("CREATE TABLE Serie ("
      "id integer primary key,"
      "name TEXT,"
      "posterPath TEXT,"
      "backdropPath TEXT,"
      "voteAverage real,"
      "genre TEXT,"
      "overview TEXT"
      ")");
}
```

```
addSerie(Serie serie) async {
  final db = await database;
  var raw = await db.insert(
    serieTableName,
    serie.toMap(),
    conflictAlgorithm: ConflictAlgorithm.replace,
  );
  return raw;
}

Future<List<Serie>> getAllSeries() async {
  final db = await database;
  var response = await db.query(serieTableName);
  List<Serie> list = response.map((c) => Serie.fromMap(c)).toList();
  return list;
}

removeSerie(int id) async {
  final db = await database;
  return db.delete(serieTableName, where: "id = ?", whereArgs: [id]);
}
```

Firestore

```
saveSerie(Serie serie) async {
  String userUid = await UserController().getUserUID();
  var series = Firestore.instance.collection("users/" + userUid + "/series");
  series.document(serie.id.toString()).setData(serie.toMap());
}

removeSerie(String id) async {
  String userUid = await UserController().getUserUID();
  var series = Firestore.instance.collection("users/" + userUid + "/series");
  series.document(id).delete();
}
```


users
SoGDHUENRFO...
series
1399
SoGDHUENRFO0apksxv6h705CieF2
Iniciar colección
series
Añadir campo
Este documento no existe; no aparecerá en ninguna consulta ni captura
series
Añadir documento
1399
1399
Iniciar colección
Añadir campo
backdropPath: "https://image.tmdb.
genre: "Drama"
id: 1399
name: "Game of Thrones"
overview: "Seven noble families figh
Westeros. Friction betwee
war. All while a very ancie
north. Amidst the war, a n
the Night's Watch, is all th
men and icy horrors beyo
posterPath: "https://image.tmdb.or
voteAverage: 8.2


5:47
DEBUG
TV SHOWS
GOT
Search
TO-DO
iPhone 11 Pro Max — 13.3

Reglas Firestore

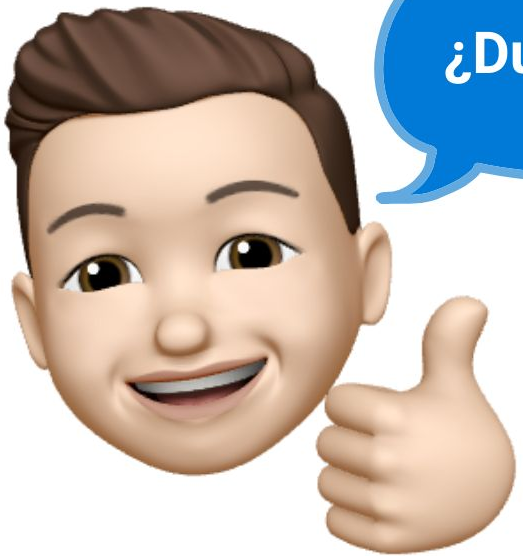
```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /users/{user}/series/{document} {
      allow create,read, update, delete:  if request.auth.uid == user;
    }
  }
}
```

Demo Time

https://demo.flutter.madrid/

dummy@gmail.com - dummy123

Ver proyecto en
https://github.com/omartinma/flutter_firebase



CONTÁCTAME

martinm.oscar@gmail.com
@oscar__martin
@omartinma