



Manual for Developer

27.01.2020

Content Management System

Overview	2
Modules	2
Content Management System Interface (Front-end)	2
Introduction	2
Platform	2
Instructions for Setup	3
Firebase	4
Introduction	4
Instructions for Setup	4
1. Make a Firebase Project	4
2. Setting up Firebase Authentication	6
3. Setting up Firestore	7
4. Setting up Firebase Storage	8
5. Setting up Firebase Hosting	9
5. Setting up Firebase Functions	10
6. Installing Firebase tools	11
7. Pushing it all to Cloud	11

Overview

This document lists stepwise how to run the code for the Content Management System and to deploy the application to the hosting service.

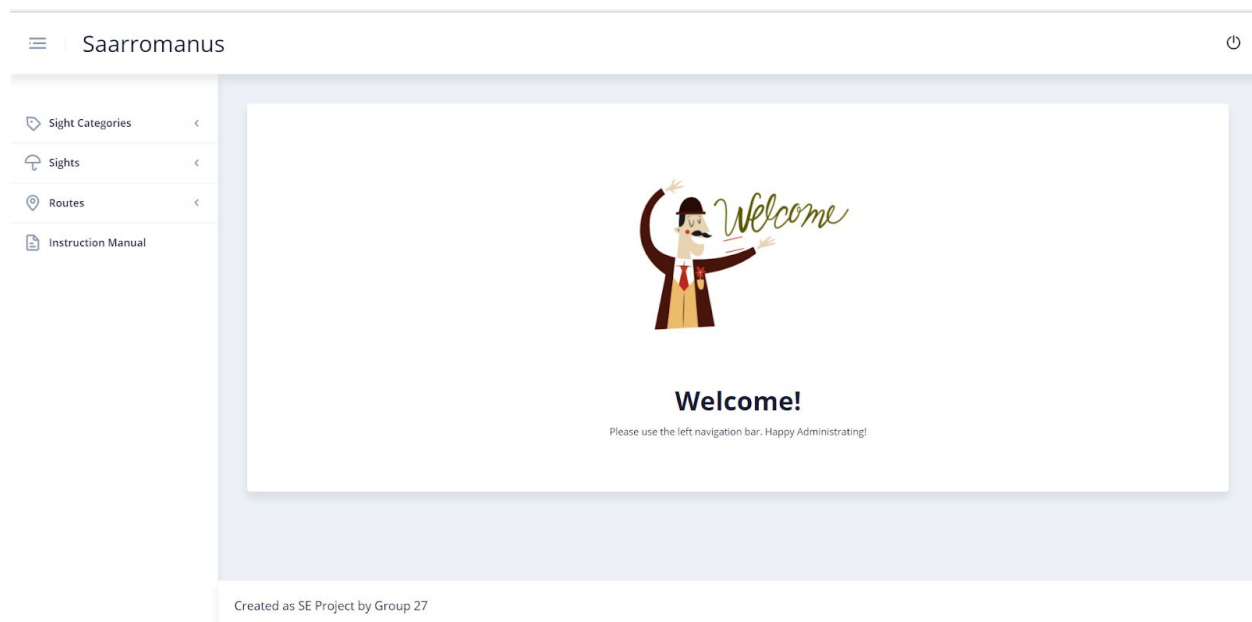
Modules

The application is divided into two parts.

1. Content Management System Interface (Front-end)

Introduction

The Content Management System (Front-end) is the interface that is visible to the administrator with the help of which one can manage (create/edit or delete) sight categories, sights and routes. Also the instruction manual can be added through the same. In short, all that is visible to the application user is managed from this portal.

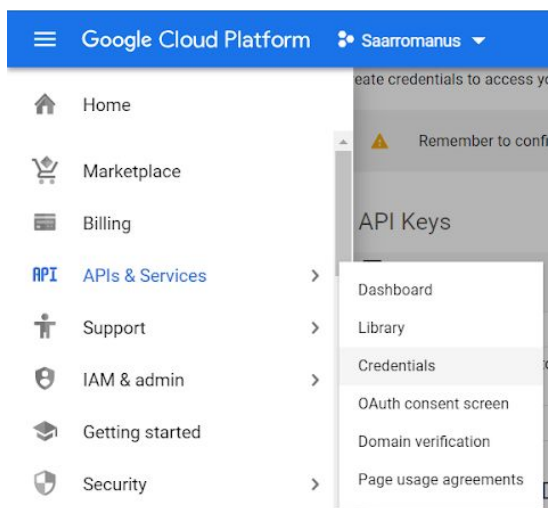


Platform

The platform used for its development is **"Angular 8"** where **"Google Firebase"** is used for backend functionality (database, hosting etc; more on it in the next step).

Instructions for Setup

1. Go to the directory containing the CRM code. In our case, it is "Backend" and run command prompt in the directory.
2. Assuming "**Node**" and "**NPM (Node Packet Manager)**" is already installed on the system and the relevant environment variables are setup, simply run "**npm install**" This will install the Angular-cli and all other dependencies.
3. The map components need Google Maps "*Places*" and "*JavaScript API*" to function. These can be enabled from [Google Cloud Platform](#) under APIs & Services > Library after selecting or creating a project. The resultant API would be then available in APIs and Services > Credentials.



4. This API needs to be added to the following files for the maps component to work;
 - \Backend\src\app\pages\sights\sights.module.ts
 - \Backend\src\app\pages\historic-routes\historic-routes.module.ts

```
@NgModule({
  imports: [
    ThemeModule,
    AgmCoreModule.forRoot({
      apiKey: 'NEW_API_KEY_GOES_HERE',
      libraries: ['places', 'geometry', 'drawing'],
    }),
  ],
})
```

5. Finally run **"ng serve"** in the command prompt and the server would be live on **"<http://localhost:4200/>"**
6. The application can also be tested by running **"ng test."** Additionally to generate a coverage report, **"ng test --code-coverage"** can be used. This will generate a folder named "coverage" in the root folder.

2. Firebase

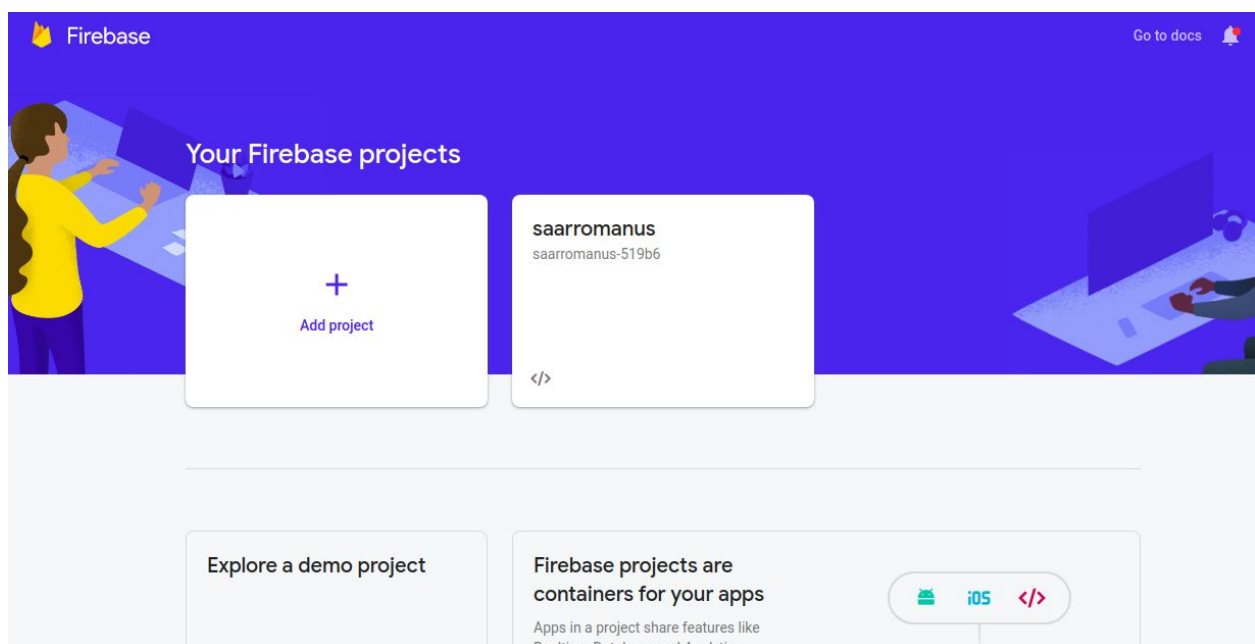
Introduction

The project uses Firebase for real-time database, storage, hosting, authentication and serverless functions. The serverless functions are used for REST API for mobile application integration. All the configurations are setup within the repository of the code. The CMS application is tightly coupled with the real-time database and storage. The hosting and the serverless functions can be used from other vendors or solution with little to none changes in the configuration. As firebase is also connected with Google Cloud Platform, it is assumed that you do have functioning Google Cloud Platform.

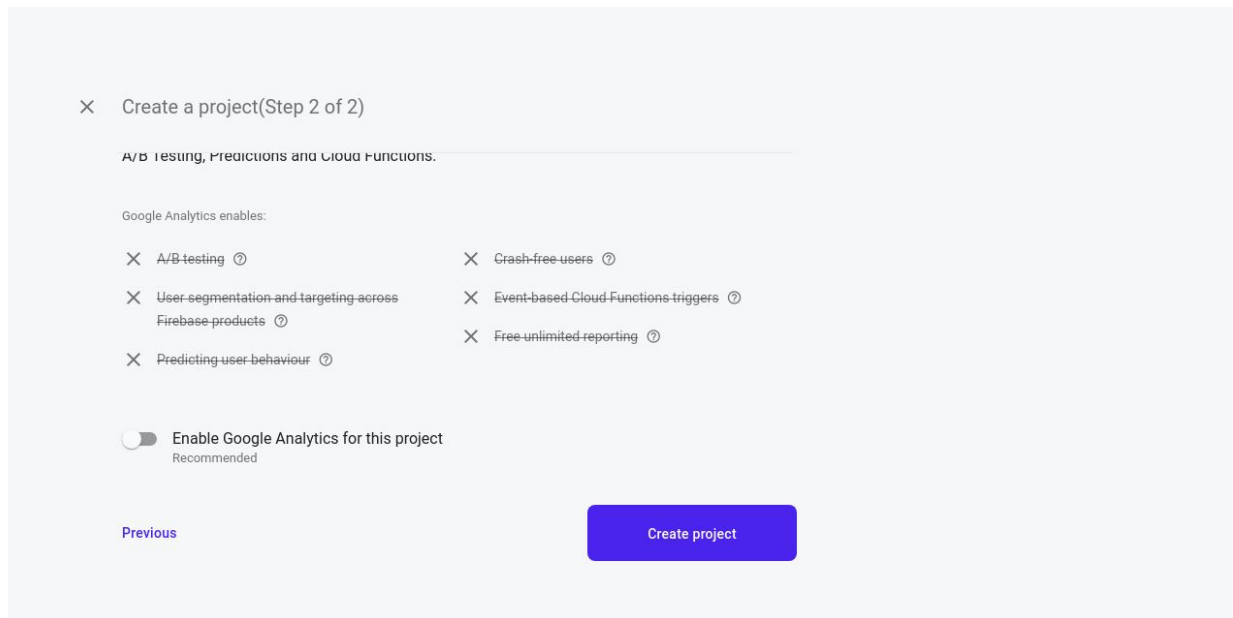
Instructions for Setup

1. Make a Firebase Project

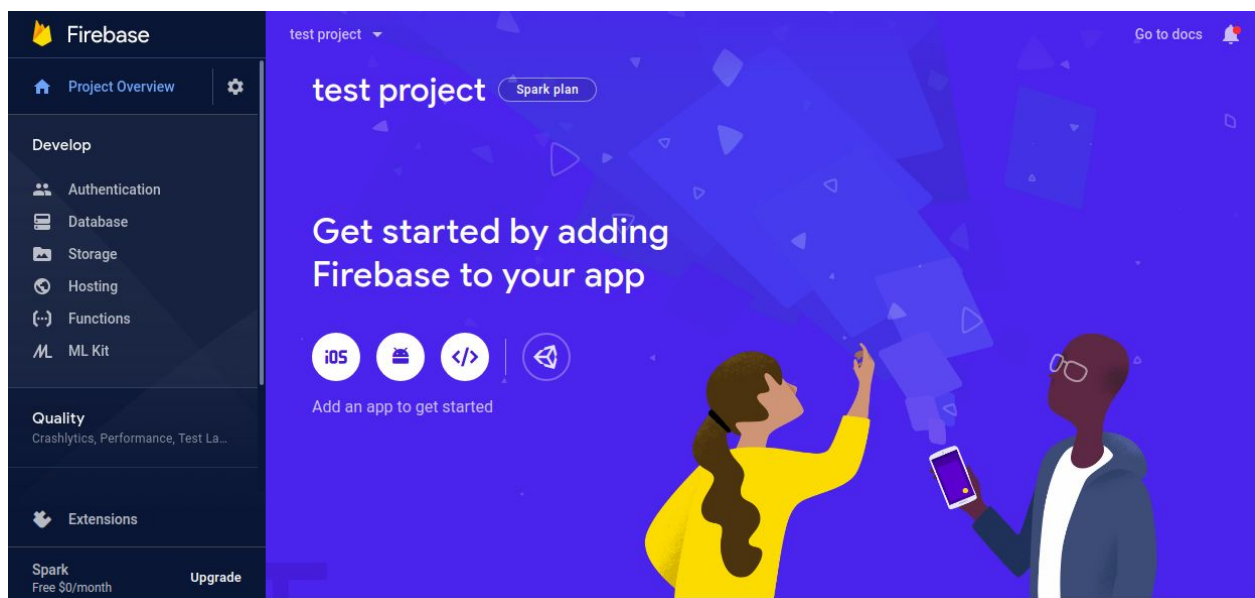
After login with a Google Developer Account, go to the [firebase console](#). Here, you can add new projects. It provides a simple step by step wizard for creating a new project.



You can add Google Analytics for the project as well but it will require you to make Google Analytics accounts which is beyond the scope of this document. So for now, we can disable Google Analytics. You can always add it later for no additional cost.

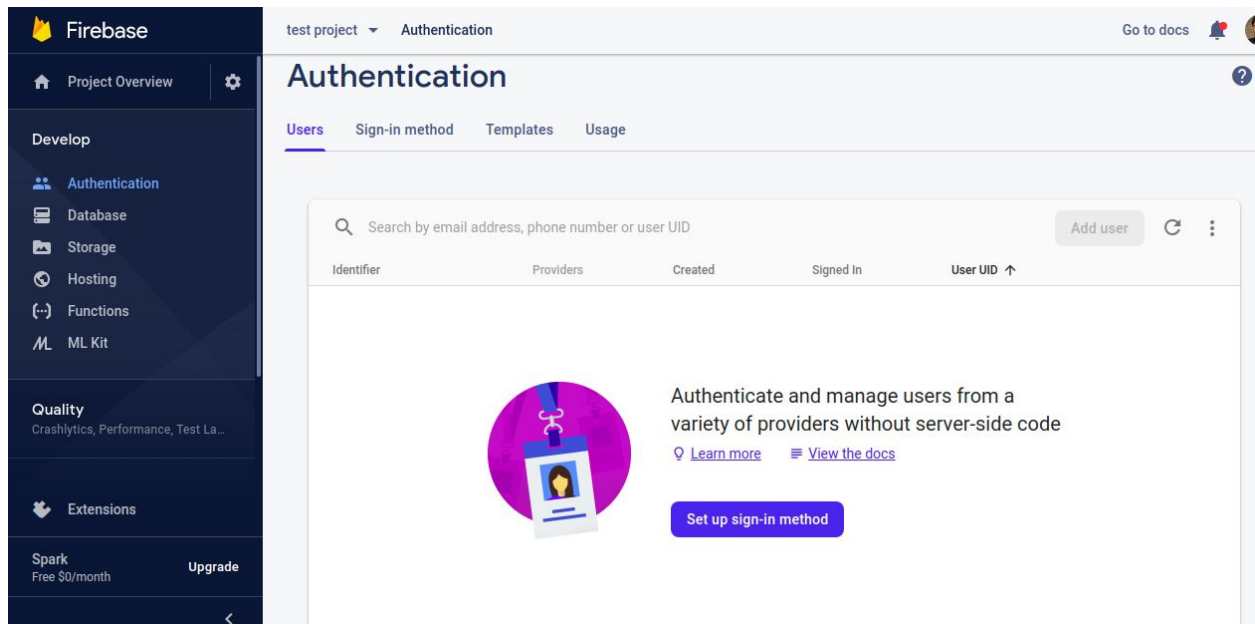


Once the project is created, you should see a similar console with the details you had entered for your project.



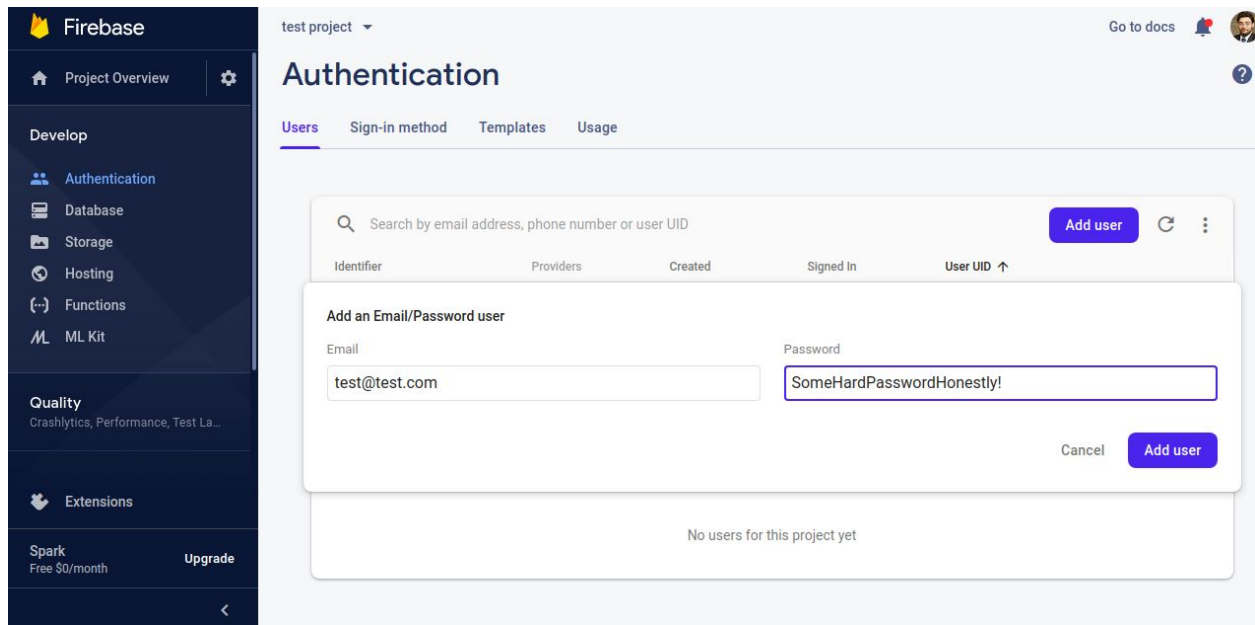
2. Setting up Firebase Authentication

From the project console screen, go to 'Authentication' in the sidebar menu. It should show you something like this:



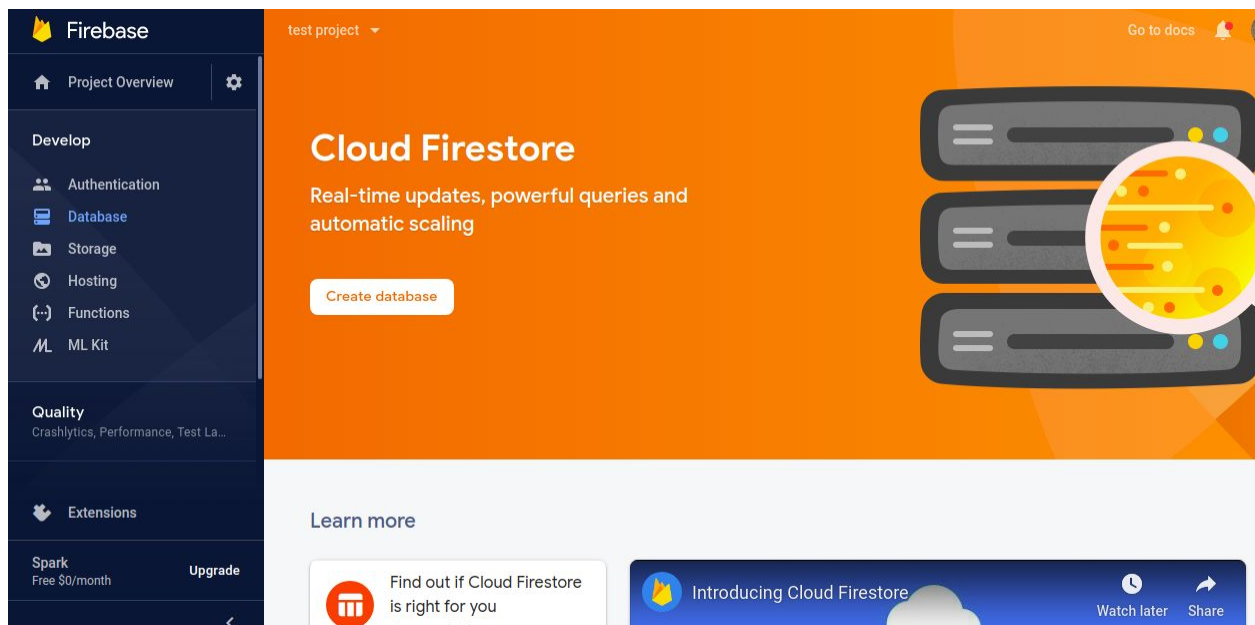
Click on 'Set up sign-in method' and choose 'Email/Password' option. Click 'Enable' for starting to use Email/Password for login to the CMS.

Once enabled, you can add users from the firebase console. The email/password entered here are used for the CMS only.

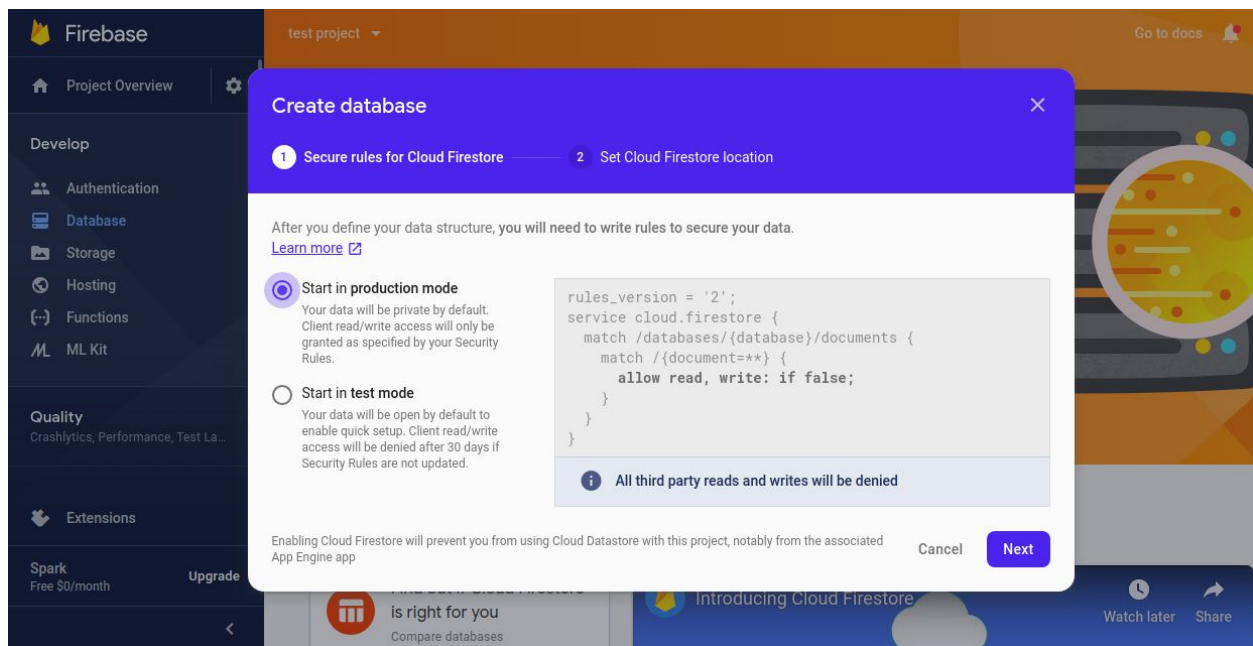


3. Setting up Firestore

From the project console screen, go to 'Database' in the sidebar menu. It should show you something like this:



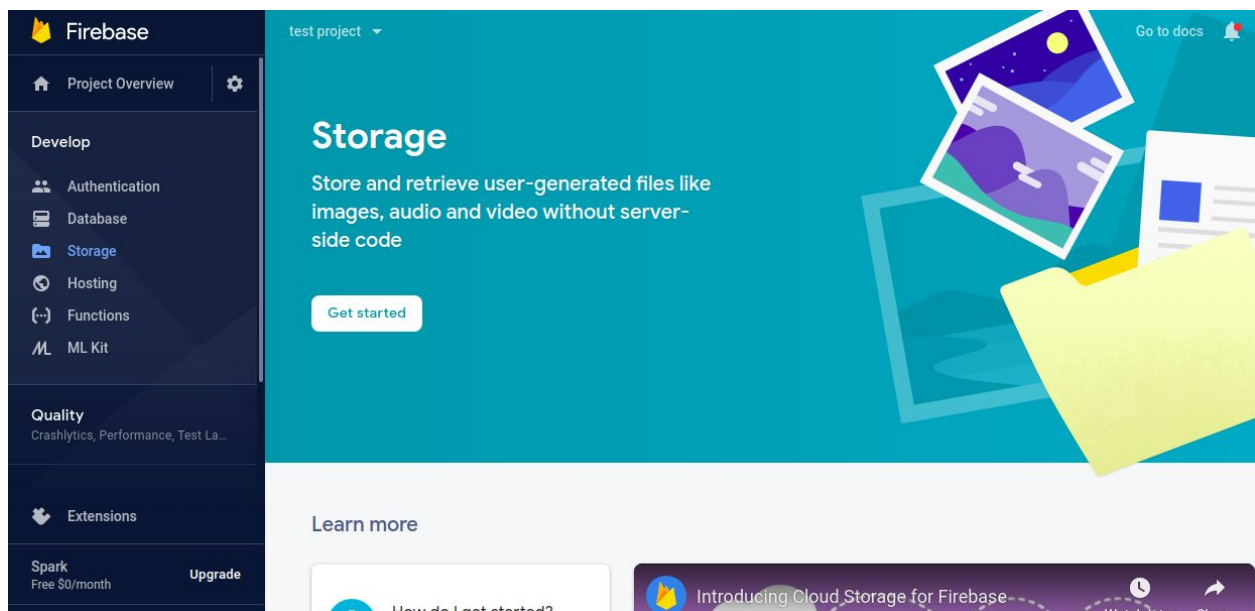
Click on 'Create database' here. We are going to make a Cloud Firestore. Firebase has two options: Real-time database and Cloud Firestore. You have to choose **Cloud Firestore**.



Start in test mode for now, the actual rules are set up within the repository. The wizard will ask for a location for the server. For this project, it is recommended that european location is used for better latency.

4. Setting up Firebase Storage

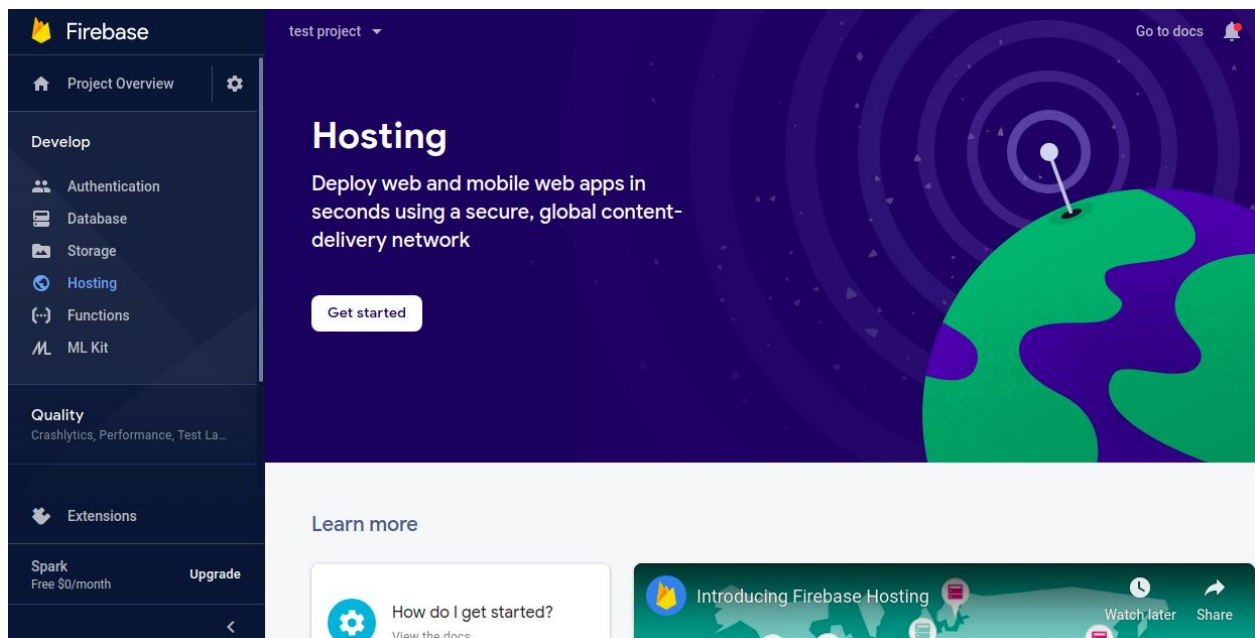
From the project console screen, go to 'Storage' in the sidebar menu. It should show you something like this:



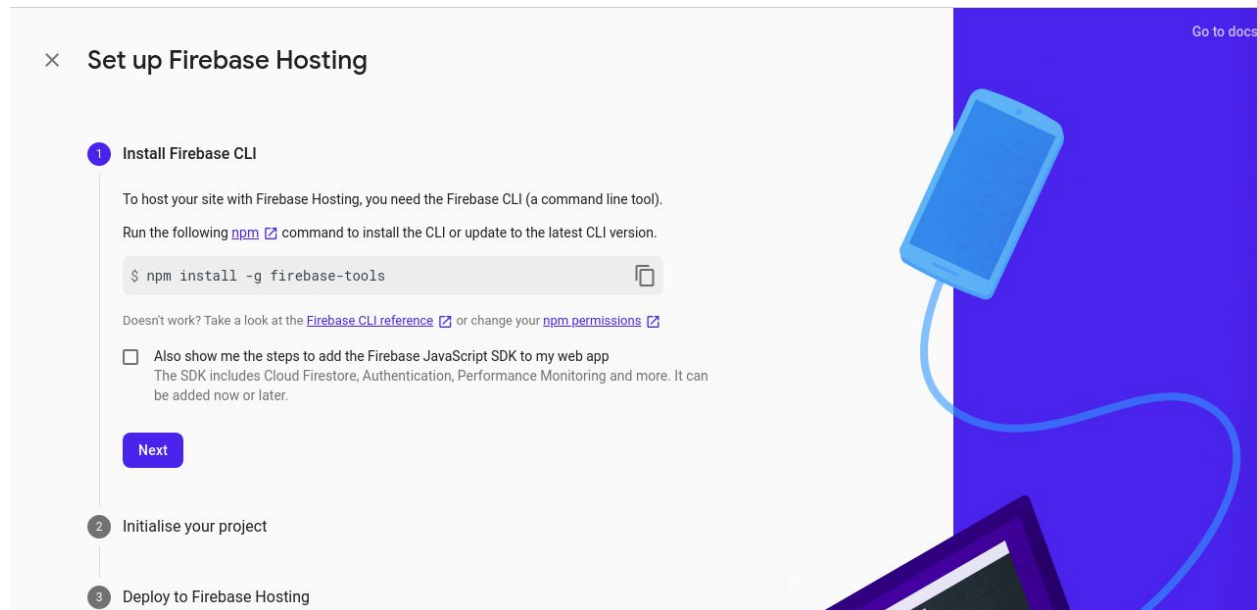
Click on 'Get started' here. Follow the wizards here as again the rules are set up within the repository.

5. Setting up Firebase Hosting

From the project console screen, go to 'Hosting' in the sidebar menu. It should show you something like this:



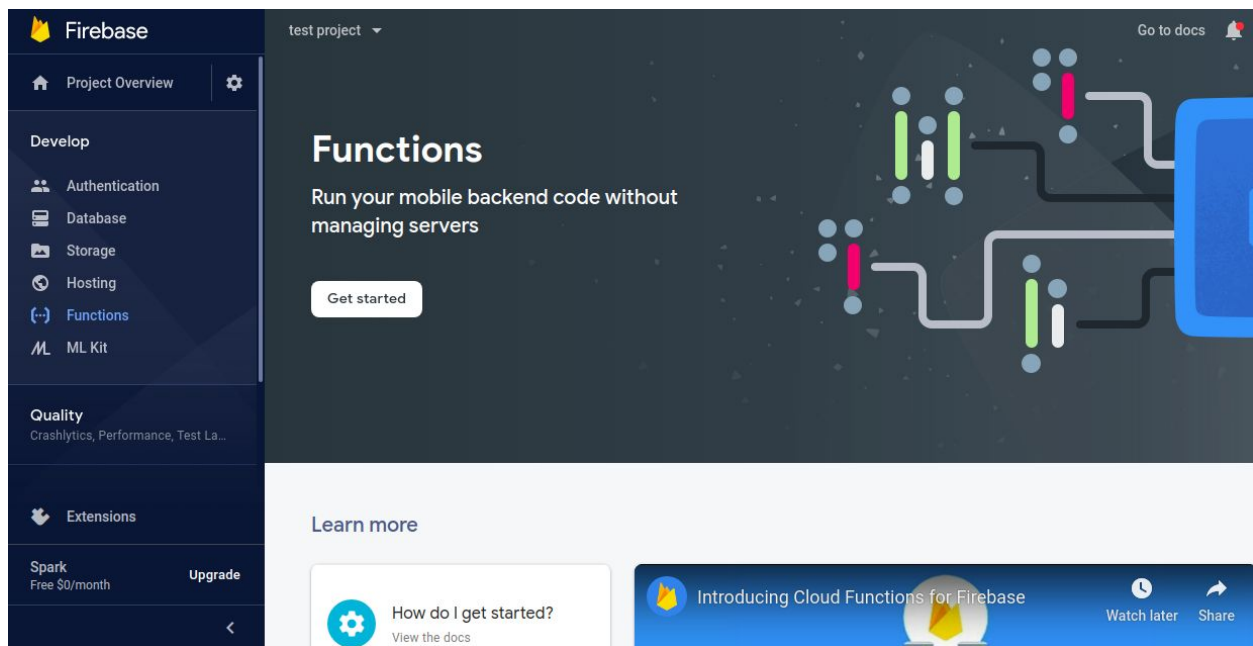
Click on 'Get started'. We assume you have knowledge of npm and are comfortable with a command line interface and have the code repository setup at your pc as mentioned in Module 1.



Just click next for these steps as these are done via the console which will be explained later.

5. Setting up Firebase Functions

From the project console screen, go to 'Functions' in the sidebar menu. It should show you something like this:



Click on 'Get started' and follow the wizard by just clicking on next. The functions are set up using the command line interface.

6. Installing Firebase tools

Open up a terminal and use the following command:

`npm install -g firebase-tools`

You should be able to use firebase from your console now.

7. Pushing it all to Cloud

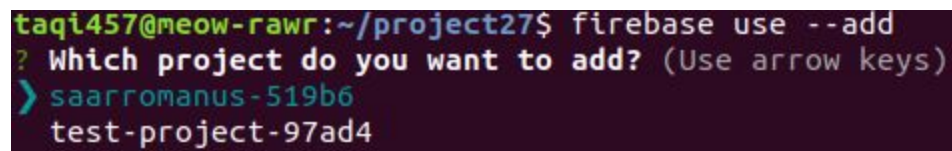
Now that all of this is setup, we can push our configuration to this new project. Open up a terminal and go to the location of the project. First, use the following command:

`firebase login`

This will open up a google authentication on your default browser. Log in there and you are set-up for the project. Now use following command:

`firebase use --add`

It should look something similar to this and list your projects here.



```
taqi457@meow-rawr:~/project27$ firebase use --add
? Which project do you want to add? (Use arrow keys)
> saarromanus-519b6
   test-project-97ad4
```

We are using **test-project-97ad4**, you have to use the project id of the project you created in step 1. Use any alias for the project and you are done. Now to deploy it all to cloud just use the following command:

“firebase deploy”

And you are done. Congratulations!