

Introducción

Este instructivo tiene como objetivo guiar paso a paso en el despliegue de una base de datos y su conexión con una API en Render. A través de este proceso, se utilizará Render para subir el backend y la base de datos, asegurando una correcta configuración y conexión entre ambas, para la parte frontend realizaremos la guía paso a paso del despliegue en el proveedor Firebase.

Objetivos

- Desplegar una base de datos PostgreSQL en Render.
- Configurar un servicio web para la API y conectarlo con la base de datos desplegada.
- Realizar el despliegue automático en Render cada vez que se realicen cambios y commits en la rama principal del repositorio.
- Realizar el despliegue de la parte frontend.

Tabla de Contenido

Creación del Servicio de Base de Datos en Render

- Generación del servicio PostgreSQL
- Configuración de parámetros y selección de opciones
- Obtención de datos para la conexión con la API

Creación del Servicio Web en Render

- Configuración del despliegue desde un repositorio Git
- Definición de parámetros del servicio web
- Configuración de variables de entorno
- Despliegue del servicio web

Verificación y Gestión

- Verificación del enlace generado
- Revisión de logs y solución de errores durante la ejecución automática

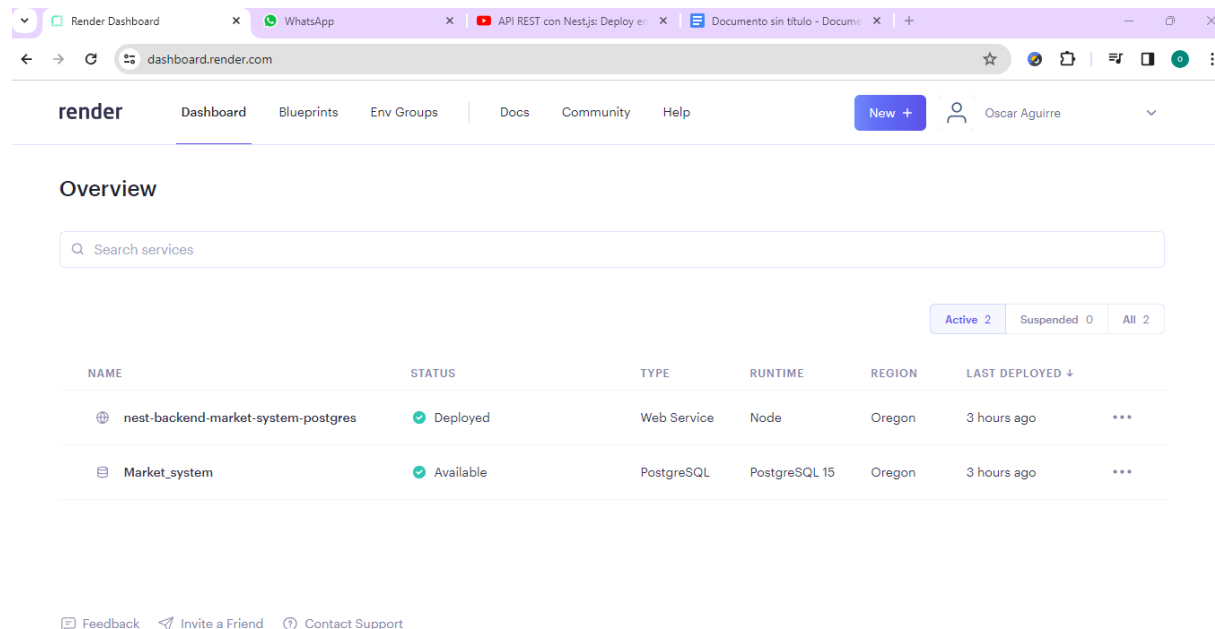
Despliegue del frontend

- Configurando Firebase.
- Configurando el proyecto para el despliegue.
- Desplegando el frontend.

1. Creación del Servicio de Base de Datos en Render

Paso 1: Generación del Servicio PostgreSQL

Para iniciar, accede a Render después de crear tu cuenta. Se recomienda vincular la cuenta de Render con GitHub para una integración más sencilla. Al ingresar, verás la interfaz de Render, excluyendo las áreas marcadas en amarillo, que corresponden a servicios previamente creados.



Paso 2: Configuración del Servicio


Haz clic en "NEW" y luego selecciona "PostgreSQL".

← → ↻ 🌐 dashboard.render.com ☆ 📁 📄 📧 📱 🌐

render Dashboard Blueprints Env Groups Docs Community Help New + 👤 Oscar Aguirre ▾

Overview

🔍 Search services



PostgreSQL

Render offers fully managed hosted PostgreSQL with internal and external connectivity, multiple storage and memory options, and automated daily backups. [Learn more.](#)

- Static Site
- Web Service
- Private Service
- Background Worker
- Cron Job
- PostgreSQL**
- Redis
- Blueprint

NAME	STATUS				LAST DEPLOYED ↓
🌐 nest-backend-market-system-postgres	✅ Deployed				
📁 Market_system	✅ Available	PostgreSQL	PostgreSQL 15	Oregon	3 hours ago ***

🗨 Feedback 📧 Invite a Friend 🛟 Contact Support

Paso 3: Opciones Adicionales

Aquí, proporciona un nombre al servicio (no a la base de datos), elige la región más cercana para reducir la latencia y selecciona la versión de PostgreSQL (como la versión 15 o la más reciente). Los campos "Database" y "User" son opcionales; Render generará valores predeterminados si no se completan. Selecciona el plan gratuito y crea la base de datos.

render Dashboard Blueprints Env Groups Docs Community Help New + 👤 Oscar Aguirre ▾

New PostgreSQL [Read the docs](#)

Name
A unique name for your PostgreSQL instance.

Database Optional
The PostgreSQL 'dbname'

User Optional

Region
The **region** where your PostgreSQL instance runs. Services must be in the same region to communicate privately and you currently have services running in Oregon.

PostgreSQL Version

Datadog API Key Optional
The API key to use for sending metrics to Datadog. Setting this will enable Datadog monitoring.

Instance Type	RAM	CPU	Storage	Price
<input checked="" type="radio"/> Free	256 MB	0.1 CPU	1 GB	\$0 / month
<input type="radio"/> Starter	256 MB	0.1 CPU	1 GB	\$7 / month
<input type="radio"/> Standard	1 GB	1 CPU	16 GB	\$20 / month
<input type="radio"/> Pro	4 GB	2 CPU	96 GB	\$95 / month
<input type="radio"/> Pro Plus	8 GB	4 CPU	256 GB	\$185 / month

Need a [custom instance type](#)? We support up to 512 GB RAM, 64 CPUs, and 5 TB storage.

Free databases will expire in 90 days and will be deleted if not upgraded. No automatic backups are created for free databases. [Learn more about free instance type limits.](#)

Access more features like [Point-in-Time Recovery](#) by upgrading to a team plan.

Create a team

Create Database

Paso 4: Obtención de Datos

Espera a que se complete la creación de la base de datos. Una vez finalizado, desplázate hacia abajo para obtener información crucial para la conexión con la API: anota el puerto, la base de datos, el nombre de usuario, la contraseña y la URL externa.

General

Name	Market_system	Edit
Created	4 hours ago	
Status	<div>Available</div>	
PostgreSQL Version	15	
Region	Oregon (US West)	
Read Replica	<div>Add Read Replica</div>	
Storage	4.79% used out of 1.0 GiB <div></div>	
Datadog API Key	<div>Add Datadog API Key</div>	

Connections

Hostname ⓘ dpg-clkd8dh8mmjc73dvljcg-a

Port 5432

Database market_system

Username market_system_user

Password ⓘ

Internal Database URL ⓘ

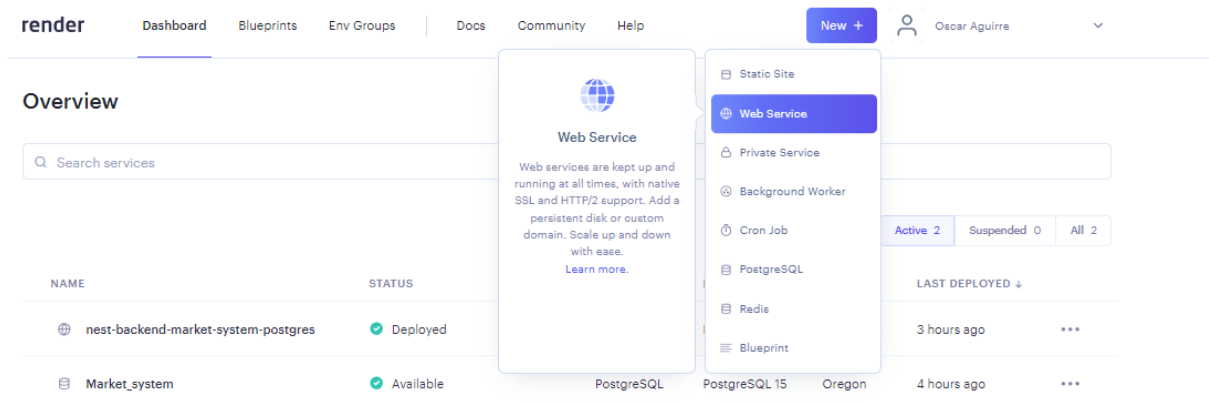
External Database URL ⓘ

PSQL Command ⓘ

2. Creación del Servicio Web en Render

Paso 1: Configuración del Despliegue desde un Repositorio Git

Selecciona "NEW" y luego "Web Service". Elige la opción para obtener el código desde un repositorio Git. Selecciona el repositorio correspondiente o configura la cuenta de GitHub si no está asociada.



Create a new Web Service

Connect a Git repository, or use an existing image.

How would you like to deploy your web service?

☒ Build and deploy from a Git repository
Connect a GitHub or GitLab repository.


☐ Deploy an existing image from a registry **ADVANCED**
Pull a public image from any registry or a private image from Docker Hub, GitHub, or GitLab.

Next


Create a new Web Service

Connect your Git repository or use an existing public repository URL.


Connect a repository

 Smendoza120 / TPS_FDS-2671339-H51 • an hour ago


Connect

 Oscar980817 / Market_Sytem_Backend • 3 hours ago


Connect

 Oscar980817 / frontend-market-system • a day ago


Connect

 Oscar980817 / TPS_FDS-2671339-H51 • 2 days ago


Connect



 Oscar980817 / Market_Sytem_Back • a month ago



Connect


 Oscar980817 / pho_conecc_oscar • 2 months ago


Connect

 GitHub


 @Oscar980817  • 16 repos



 @Smendoza120  • 1 repo



 GitLab


 + Connect account


en caso de que el repositorio no esté asociado o la cuenta de github no aparezca seleccionaremos configure account, donde tendremos que ingresar el usuario y contraseña de github para que este lo reconozca

 GitHub

 @Oscar980817  • 16 repos

 @Smendoza120  • 1 repo

 GitLab

 + Connect account

Paso 2: Configuración del Servicio Web

Rellena los detalles siguientes:

- Nombre del servicio web: Un nombre descriptivo para identificar el servicio en Render.
- Región para desplegar la aplicación o API: Selecciona la región más cercana al público objetivo para minimizar la latencia.
- Rama principal del repositorio: Por lo general, es "main" o "master", dependiendo de la estructura del repositorio.
- Directorio raíz: Puede dejarse por defecto, a menos que la estructura del repositorio requiera especificar una ruta específica.

- Tipo de lenguaje de programación o contenedor Docker: Selecciona el lenguaje de programación utilizado para la aplicación o especifica si se trata de un contenedor Docker configurado previamente para Render.

You are deploying a web service for [Oscar980817/frontend-market-system](#).

You seem to be using Node, so we've autofilled some fields accordingly. Make sure the values look right to you!

Name A unique name for your web service.	<input type="text" value="example-service-name"/> ❗ Required
Region The region where your web service runs. Services must be in the same region to communicate privately and you currently have services running in Oregon .	<input type="text" value="Oregon (US West)"/>
Branch The repository branch used for your web service.	<input type="text" value="main"/>
Root Directory <small>Optional</small> Defaults to repository root. When you specify a root directory that is different from your repository root, Render runs all your commands in the specified directory and ignores changes outside the directory.	<input type="text" value="e.g. src"/>
Runtime The runtime for your web service.	<input type="text" value="Node"/>

Paso 3: Comandos para Build y Start

En el campo "Build Command", introduce los comandos necesarios para instalar dependencias y realizar el build de la aplicación. Por ejemplo, en NestJS (un framework de Node), se pueden ejecutar comandos para instalar dependencias y construir el código para producción.

El campo "Start Command" debe ejecutar la carpeta dist, que contiene el código minificado listo para producción, obtenido generalmente desde el package.json.

```
{
  "name": "backend",
  "version": "0.0.1",
  "description": "",
  "author": "",
  "private": true,
  "license": "UNLICENSED",
  "scripts": {
    "build": "nest build",
    "format": "prettier --write \"src/**/*.ts\" \"test/**/*.ts\"",
    "start": "nest start",
    "start:dev": "nest start --watch",
    "start:debug": "nest start --debug --watch",
    "start:prod": "node dist/main",
    "lint": "eslint \"{src,apps,libs,test}/**/*.ts\" --fix",
    "test": "jest",
    "test:watch": "jest --watch",
    "test:cov": "jest --coverage",
    "test:debug": "node --inspect-brk -r tsconfig-paths/register -r ts-node/register node_modules/.bin/jest --runInBand",
    "test:e2e": "jest --config ./test/jest-e2e.json"
  }
}
```


Build Command

This command runs in the root directory of your repository when a new version of your code is pushed, or when you deploy manually. It is typically a script that installs libraries, runs migrations, or compiles resources needed by your app.

```
$ yarn install; yarn build
```

Start Command

This command runs in the root directory of your app and is responsible for starting its processes. It is typically used to start a webserver for your app. It can access environment variables defined by you in Render.

```
$ npm run start:prod
```

Please enter your payment information to select an instance type with higher limits.

Instance Type	RAM	CPU	Price
<input checked="" type="radio"/> Free	512 MB	0.1 CPU	\$0 / month
<input type="radio"/> Starter	512 MB	0.5 CPU	\$7 / month
<input type="radio"/> Standard	2 GB	1 CPU	\$25 / month

Paso 4: Configuración de Variables de Entorno

Si la API requiere variables de entorno, accede a "Advanced" antes de crear el servicio web. Aquí, puedes añadir las variables necesarias para el funcionamiento de la aplicación, como datos de conexión a la base de datos PostgreSQL.

Advanced ^

Use environment variables to store API keys and other configuration values and secrets. You can access them in your code like regular environment variables, for example with `os.getenv()` in Python or `process.env` in Node.

Generate













Required

Add Environment Variable

You can store secret files (like `.env` or `secrets.yml` files and private keys) in Render. These files can be accessed during builds and in your code just like regular files.

Environment Variables

Use environment variables to store API keys and other configuration values and secrets. You can access them in your code like regular environment variables, for example with `os.getenv()` in Python or `process.env` in Node.

Key	Value	
API_CONFIG_AUTH_DURATION_JWT	
API_CONFIG_AUTH_SECRET_JWT	
API_EMAIL	
API_EMAIL_PASSWORD	
POSTGRES_DATABASE	
POSTGRES_HOST	
POSTGRES_PASSWORD	
POSTGRES_PORT	
POSTGRES_SSL	
POSTGRES_USERNAME	

Paso 5: Creación del Servicio Web

Finalmente, haz clic en "Create Web Service" para establecer el despliegue automático con cada commit en la rama principal del repositorio.

ADD INCLUDED PATH

Ignored Paths

Changes that match these paths will not trigger a new build.

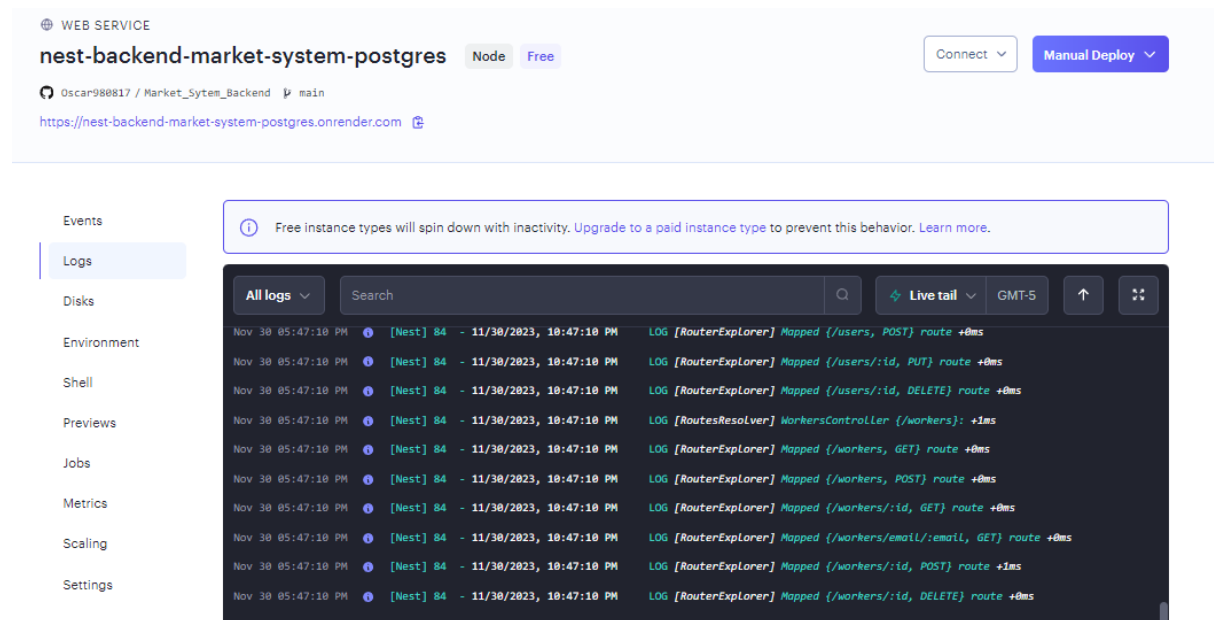
+ Add Ignored Path

Create Web Service

3. Verificación y Gestión

Paso 1: Verificación del Enlace Generado

Una vez creado el servicio web, verifica el enlace generado para acceder a tu API o aplicación desplegada. Este enlace te permitirá realizar pruebas y compartir tu proyecto.



Paso 2: Revisión de Logs y Solución de Errores

En caso de errores durante la ejecución, accede a los registros (logs) proporcionados por Render. Estos registros te darán información detallada sobre cualquier problema que pueda surgir durante la ejecución de tu aplicación. Utiliza esta información para solucionar cualquier inconveniente.

Paso 3: Gestión Automática de Deploys

Recuerda que al configurar el servicio web en Render, se habilita la funcionalidad de deploys automáticos. Cada vez que hagas un push a la rama principal de tu repositorio, Render ejecutará automáticamente el despliegue con los nuevos cambios en tu aplicación.

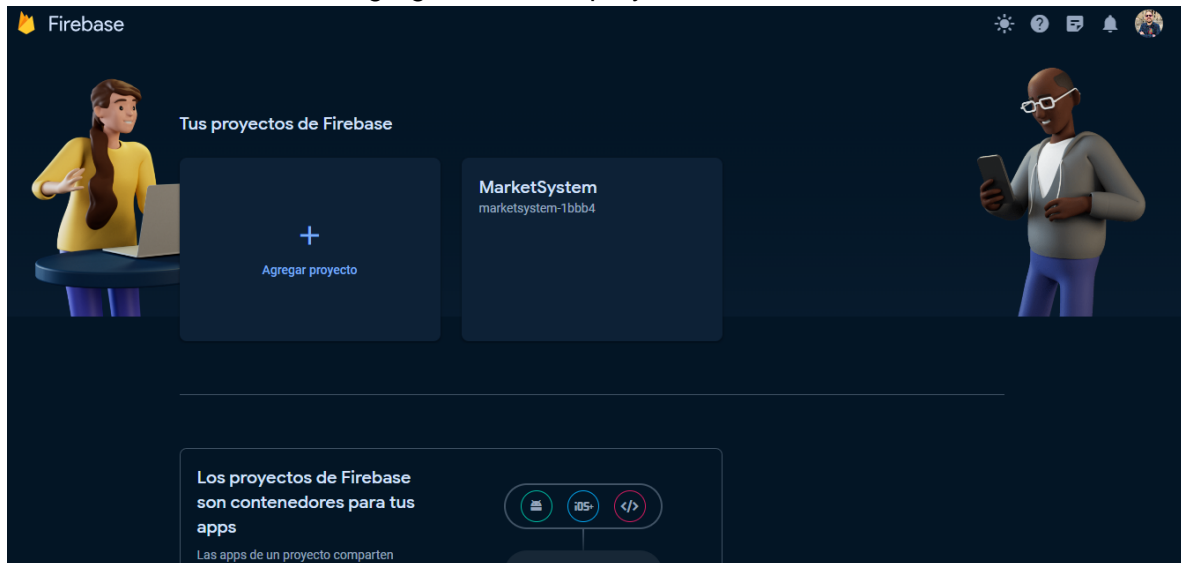
Paso 4: Mantenimiento y Actualizaciones

Realiza un seguimiento periódico de tu servicio desplegado en Render. Asegúrate de mantener tus dependencias actualizadas y realiza pruebas regulares para garantizar un funcionamiento óptimo de tu aplicación o API.

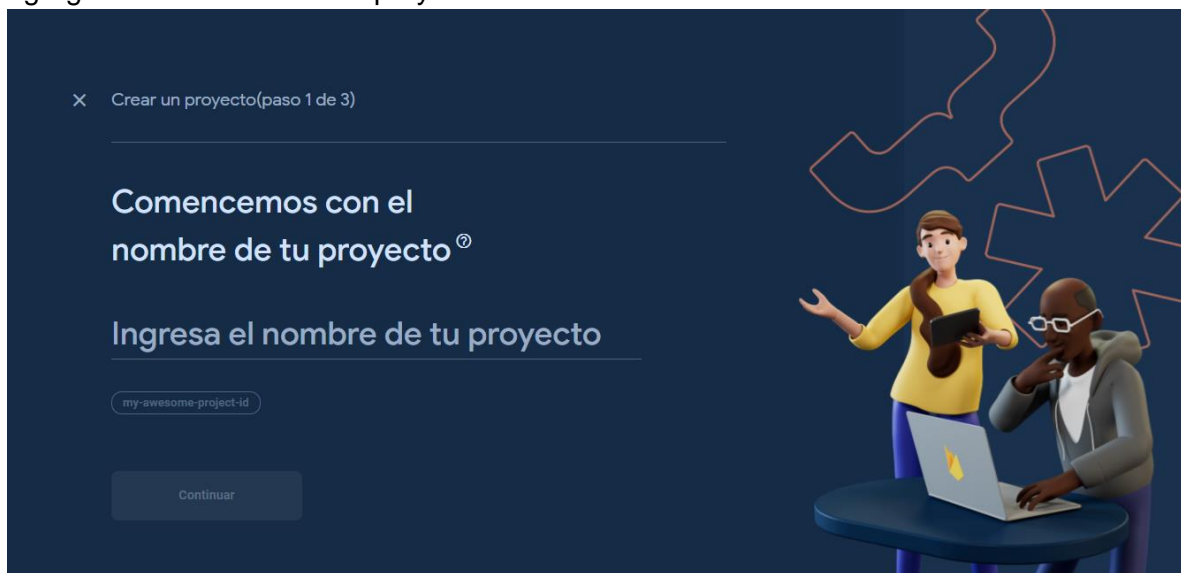
4. Despliegue del frontend.

Paso 1: Configurando Firebase

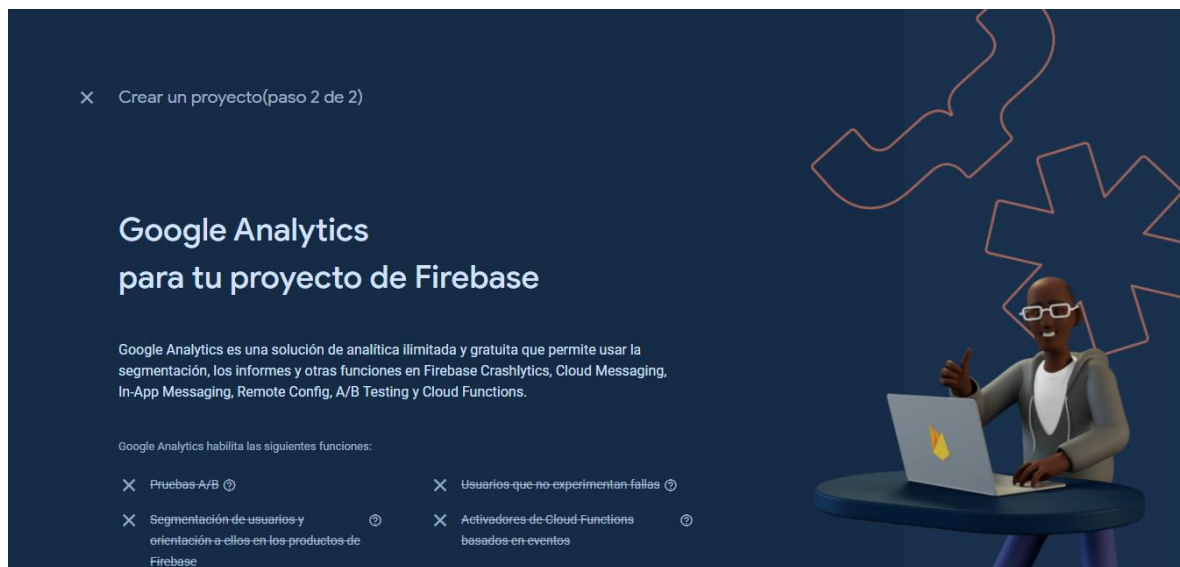
Primero debemos ingresar a la página web de Firebase, creamos una cuenta, una vez creada la cuenta vamos a agregar un nuevo proyecto.



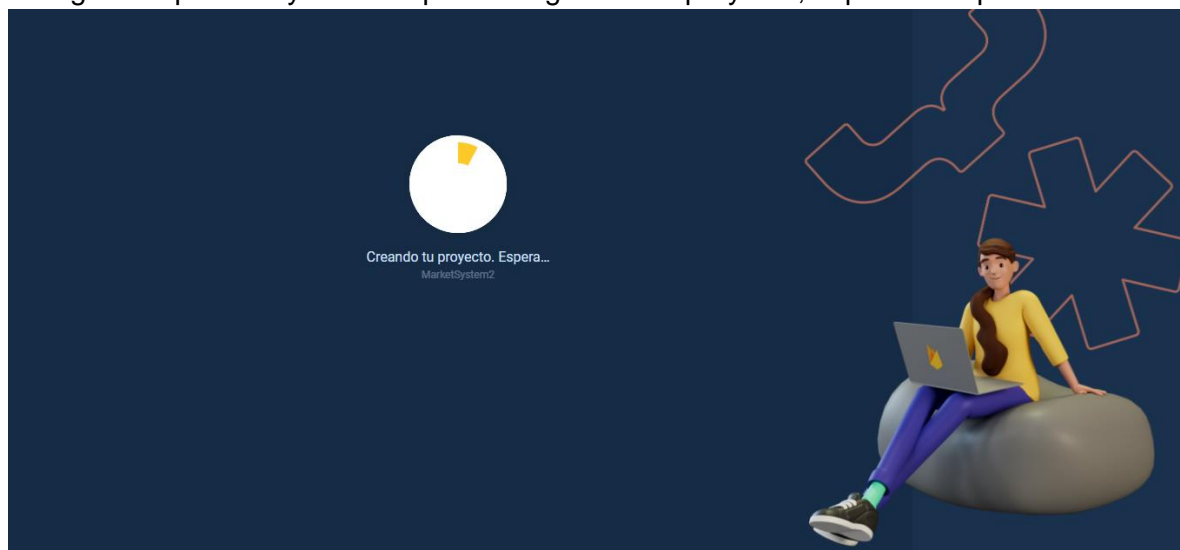
Una vez le demos clic a agregar el proyecto, nos redireccionara a otra pantalla donde agregaremos el nombre del proyecto.



Continuamos con la configuración del sitio, por lo que ahora nos mostrara una opción para incorporar Google Analytics, para esta configuración no la necesitamos, por lo que la omitiremos.



La siguiente pestaña ya nos empezara a generar el proyecto, esperamos que finalice.



Una vez finalizada la creación de nuestro proyecto, nos dirigimos al panel izquierdo en la sección configuración y damos clic en el ítem Hoisting.

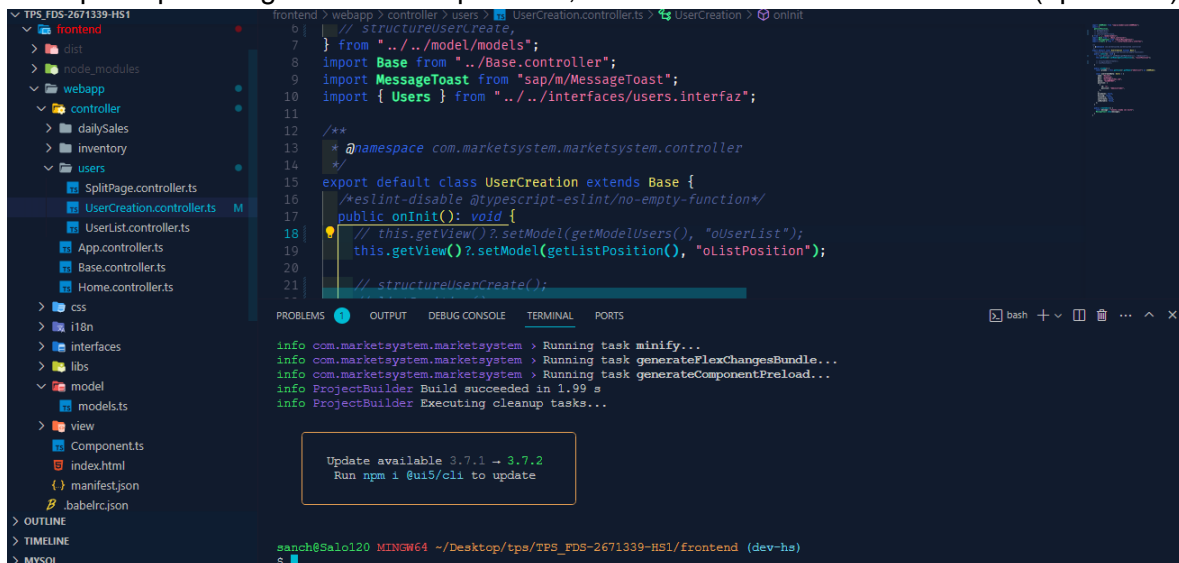


Ahora vamos a configurar el Hoisting, por lo que una vez sigamos el paso anterior, nos mostrara un botón para empezar con la configuración,



Paso 2: Configurando el proyecto para el despliegue.

Antes de continuar en la pagina para subir nuestro aplicativo, configuraremos nuestro proyecto para poder subirlo sin dificultades, por lo que tenemos que realizar el comando de Build para que nos genere una carpeta dist, en nuestro caso el comando seria (npm build)



The screenshot shows a VS Code editor with a file explorer on the left, a TypeScript file named `UserCreation.controller.ts` in the center, and a terminal at the bottom. The file explorer shows a project structure with folders like `frontend`, `webapp`, `controller`, `users`, `css`, `lib`, `interfaces`, `libs`, `model`, `models`, `view`, `Components`, `index.html`, `manifest.json`, and `.babelrc.json`. The `UserCreation.controller.ts` file contains the following code:

```
import { Base } from "../Base.controller";
import { MessageToast } from "sap/m/MessageToast";
import { Users } from "../interfaces/users.interfaz";

export default class UserCreation extends Base {
  public onInit(): void {
    // this.getView().setModel(getModelUsers(), "oUserList");
    // this.getView().setModel(getListPosition(), "oListPosition");
    // structureUserCreate();
  }
}
```

The terminal shows the output of the `npm build` command:

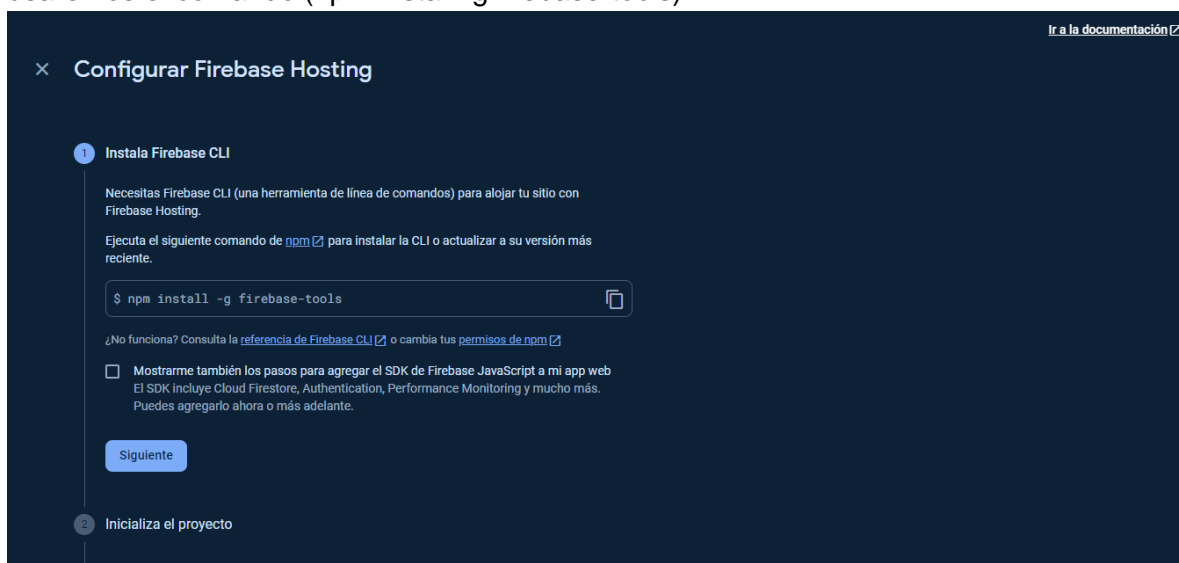
```
info com.marketsystem.marketsystem > Running task minify...
info com.marketsystem.marketsystem > Running task generateFlexChangesBundle...
info com.marketsystem.marketsystem > Running task generateComponentPreload...
info ProjectBuilder Build succeeded in 1.99 s
info ProjectBuilder Executing cleanup tasks...
```

A notification box in the terminal indicates an update available from 3.7.1 to 3.7.2, with the command `npm i @ui5/cli` to update.

Paso 3: Desplegando el frondend.

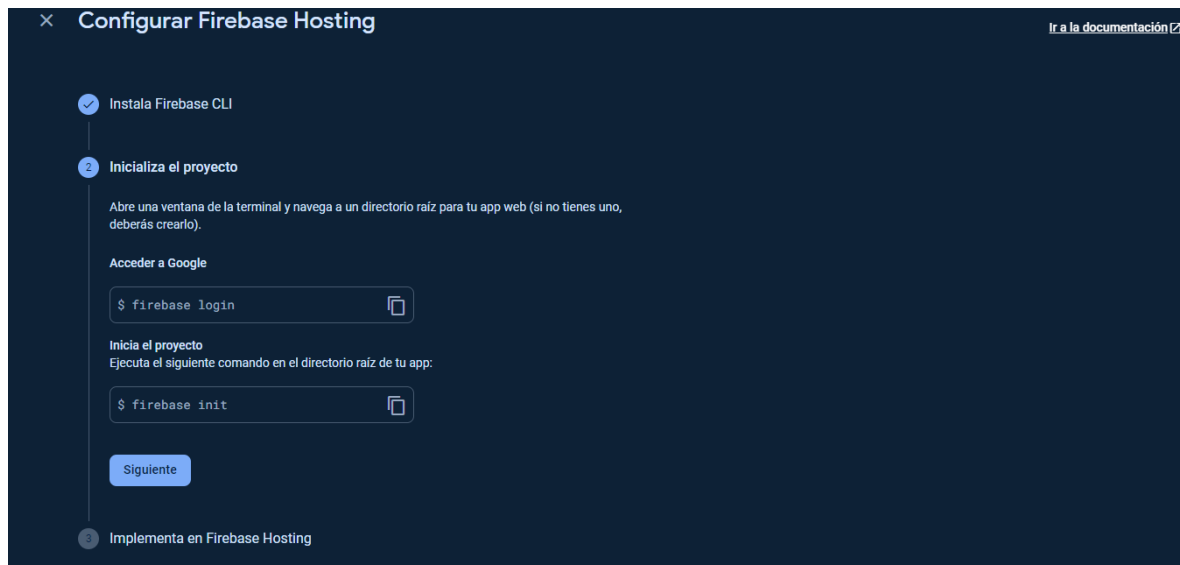
Una vez que creamos nuestro proyecto y el código lo optimizamos para poder realizar el despliegue de este, ahora seguiremos con los pasos que nos muestra nuestro gestor de nube, por lo que volveremos a la pagina web y seguimos el paso a paso.

Tenemos que instalar las dependencias de Firebase dentro de nuestro proyecto, por lo que usaremos el comando (npm install -g firebase-tools)



The screenshot shows the "Configurar Firebase Hosting" page. It includes a section for "Instala Firebase CLI" with instructions on how to install the CLI using `npm install -g firebase-tools`. There is a checkbox for "Mostrarme también los pasos para agregar el SDK de Firebase JavaScript a mi app web" and a "Siguiente" button. The page also has a "Ir a la documentación" link in the top right corner.

Damos clic en el botón continuar, una vez realizamos la instalación de dependencias de Firebase, ahora iniciamos sesión, para ello usamos el comando (firebase login), como ya tenemos una cuenta Firebase y actualmente nos encontramos logueados, realizara una autenticación SSO para comprobar nuestra cuenta.



Cuando nos logueamos ahora tenemos que inicializar Firebase, para ello usamos el comando (firebase init), y realizamos la siguiente configuración

- Marcamos la opción de Hoisting,
- Siguiente paso, nos preguntara si subiremos un proyecto existente o crearemos uno nuevo, por lo que seleccionamos que subiremos uno existente.
- Nos mostrara el proyecto, por lo que damos enter.
- Nos preguntara que carpeta tomara, por lo que le decimos que dist.
- Nos preguntara si queremos configurarla como una single app, damos que no.
- Nos pedirá que, si queremos configurar compiladores y despliegues automáticos, damos que no.
- Nos preguntara si realizaremos la compilación del proyecto, como se realizo anteriormente, marcamos la opción no.


```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: node

#####  #####  #####  #####  #####  #####  #####
##      ## ##      ## ##      ##      ## ##      ##
#####  ##  #####  #####  #####  #####  #####
##      ## ##      ## ##      ##      ##      ##
##      #####  #####  #####  #####  #####

You're about to initialize a Firebase project in this directory:

D:\TrasteandoSAP\simple-openui5

? Are you ready to proceed? Yes
? Which Firebase CLI features do you want to set up for this folder? Press Space to select features, then Enter to co
pace> to select, <a> to toggle all, <i> to invert selection)
  ( ) Database: Configure Firebase Realtime Database and deploy rules
  ( ) Firestore: Deploy rules and create indexes for Firestore
  ( ) Functions: Configure and deploy Cloud Functions
> ( ) Hosting: Configure and deploy Firebase Hosting sites
  ( ) Storage: Deploy Cloud Storage security rules
  ( ) Emulators: Set up local emulators for Firebase features
  ( ) Remote Config: Get, deploy, and rollback configurations for Remote Config

? Please select an option: Use an existing project
? Select a default Firebase project for this directory: simple-openui5 (simple-openui5)
i Using project simple-openui5 (simple-openui5)

=== Hosting Setup

Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? dist
? Configure as a single-page app (rewrite all urls to /index.html)? No
? Set up automatic builds and deploys with GitHub? No
+ Wrote dist/404.html
? File dist/index.html already exists. Overwrite? No
i Skipping write of dist/index.html

i Writing configuration info to firebase.json...
i Writing project information to .firebaserc...

+ Firebase initialization complete!
PS D:\TrasteandoSAP\simple-openui5>
```

Por último agregamos el ultimo comando el cual es (firebase deploy), aquí empezaremos a subir el proyecto a nuestro proveedor de nube, cuando finalice nos darán una URL del hoisting, también puedes revisar este link desde la página Firebase.

Ir a la documentación

×

Configurar Firebase Hosting

✓

Instala Firebase CLI

✓

Inicializa el proyecto

3

Implementa en Firebase Hosting

Cuando tengas todo listo, implementa tu app web

Ubica los archivos estáticos (p. ej., HTML, CSS y JS) en el directorio de implementación de la app (el directorio predeterminado es "public"). Luego, ejecuta este comando desde el directorio raíz de tu app:

\$ firebase deploy

Después de la implementación, consulta tu app en marketsystem2-160b4.web.app.

¿Necesitas ayuda? Consulta [los documentos de Hosting](#).

Ir a la consola

```
+ Firebase initialization complete!
PS D:\TrasteandoSAP\simple-openui5> firebase deploy

=== Deploying to 'simple-openui5'...

i deploying hosting
i hosting[simple-openui5]: beginning deploy...
i hosting[simple-openui5]: found 42 files in dist
+ hosting[simple-openui5]: file upload complete
i hosting[simple-openui5]: finalizing version...
+ hosting[simple-openui5]: version finalized
i hosting[simple-openui5]: releasing new version...
+ hosting[simple-openui5]: release complete

+ Deploy complete!

Project Console: https://console.firebase.google.com/project/simple-openui5/overview
Hosting URL: https://simple-openui5.web.app
PS D:\TrasteandoSAP\simple-openui5>
```

🔥

Firestore

MarketSystem

Hosting > Administrar sitio

🌞

?

📄

🔔

👤

Descripción gener...

Accesos directos a proyectos

🔗

Hosting

Authentication

Storage

Categorías de producto

Compilación

Lanzamiento y supervisión

Analytics

Participación

Todos los productos

Spark

Sin costo \$0 por mes

Actualizar

Versión actual

★

sanchezharold13@gmail.com

29/11/23, 21:33 UTC

1d4c3f

Versiones anteriores

★

sanchezharold13@gmail.com

29/11/23, 21:33 UTC

1d4c3f

🔗

sanchezharold13@gmail.com

29/11/23, 21:28 UTC

da20a7

Configuración de almacenamiento de actualizaciones

Ver detalles →

Dominios

marketsystem-1bbb4.web.app

Predeterminado

marketsystem-1bbb4.firebaseio.com

Predeterminado

Agregar un dominio personalizado

Ver los 2 dominios →

Canales de vista previa

Beta

Este es el link del aplicativo desplegado: <https://marketsystem-1bbb4.web.app/>