

**Tecnológico de Estudios Superiores de Ecatepec.**

División de Ingeniería en Sistemas Computacionales.

**Materia:** Base de datos para dispositivos móviles

**Proyecto:**

**Login**

**Profesor:** Griselda Cortes Barrera

**Integrantes:**

* Ponce Flores Pedro Antonio
* Rangel Reyes Cesar Librado
* Zenteno Calderón Luis Enrique

**Grupo:** 5851

**Índice**

[**Configuración el Backend** 3](#_Toc93928501)

[**Base de datos con vistas en mongoDB del módulo asignado** 13](#_Toc93928502)

[**Visualización de datos en su Frontend** 16](#_Toc93928503)

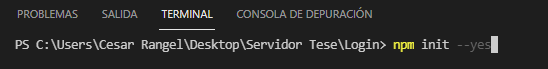
[**Enlace del Proyecto Login en GitHub** 23](#_Toc93928504)

[**Manual Técnico** 24](#_Toc93928505)

# **Configuración el Backend**

Creamos una carpeta donde vamos a alojar el servidor y poder hacer la conexión a la base de datos.

Abrimos cualquier editor de código ya sea Visual Studio Code o WebStorm, ya una vez ahí adentro abrimos lo que es la carpeta donde vamos a alojar el servidor. Después abrimos un terminal y colocamos el siguiente comando: “npm init --yes”. Con eso nos va crear lo que es el servidor.



Ya una vez que creo nuestro servidor vamos a instalar los siguientes paquetes:

* Npm i express Morgan
* Npm i nodemon –D
* Npm i underscore
* Npm i mongoose

Una vez que se hayan instalado esos paquetes nos dirigimos al archivo de package.json para poder observar y corroborar que están instalados los paquetes.



Lo que sigue una vez a dentro de ese archivo en la parte de scripts le vamos a poner el siguiente comando que es “” dev”: “nodemon src/index.js””, lo que hace es que al momento de ejecutar el comando de npm run dev va ejecutar el servidor y no tenemos que cancelar la ejecución para volver los cambios que se vallan haciendo dentro del servidor.

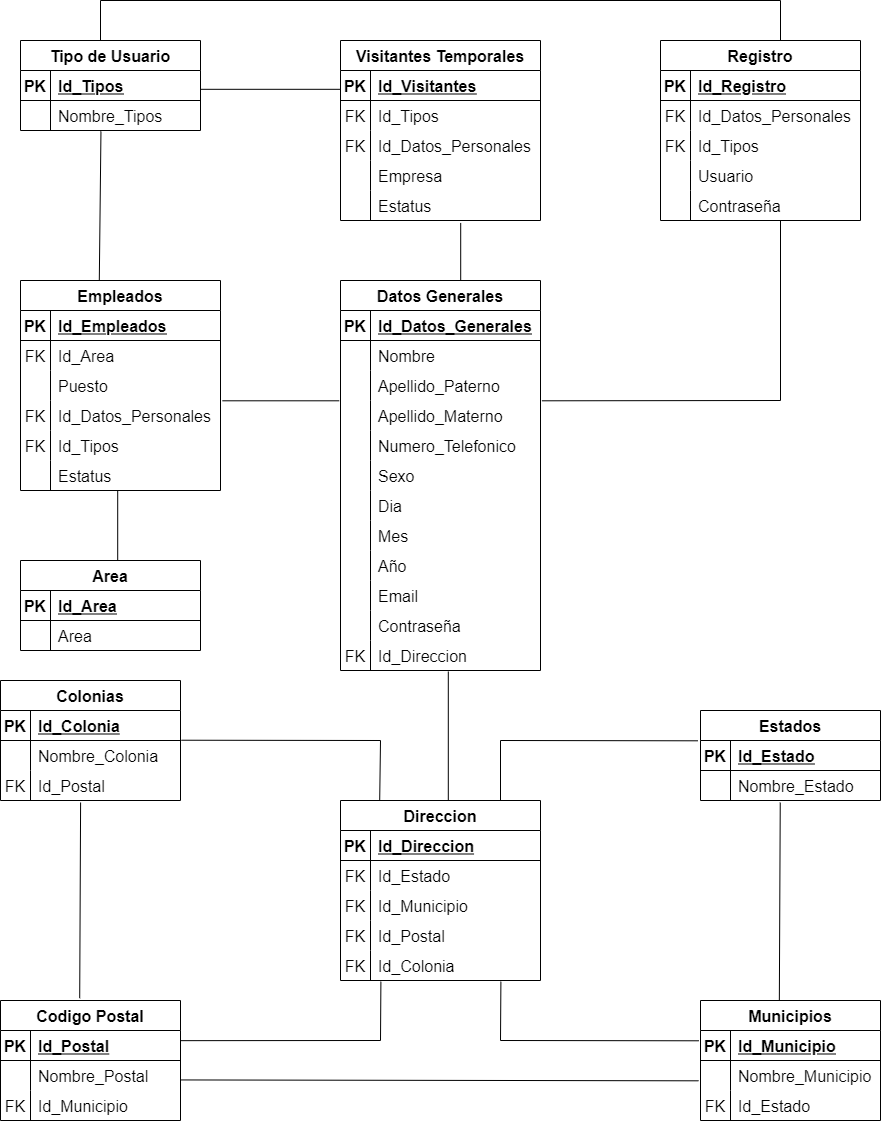
Lo que sigue es que dentro de la carpeta del LOGIN vamos a crear otra carpeta con el nombre de src donde vamos a trabajar con el servidor. Una vez que creamos esa carpeta lo que sigue es crear un archivo con el nombre index.js y otras 3 carpetas con el nombre de config, models y routes.

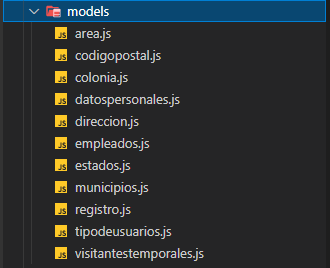
Abrimos el archivo index.js y vamos hacer la configuración y colocar las rutas que vallamos a ocupar dentro de ella, en que inicie lo que es la base de datos una vez que la conectamos.

Abrimos la carpeta de config y creamos un archivo con el nombre db.js, lo que va ser esta carpeta va guardar el archivo donde nos vamos a conectar en la base de datos de Mongo DB. Vamos usar las siguientes líneas de comandos para podernos conectarnos a la base de datos.



Lo que sigue es abrir la carpeta de models y crear los archivos con los nombres las tablas que vamos a usar en la base de datos, en la siguiente imagen podemos ver cuáles son esas tablas y sus atributos que se van hacer usar.

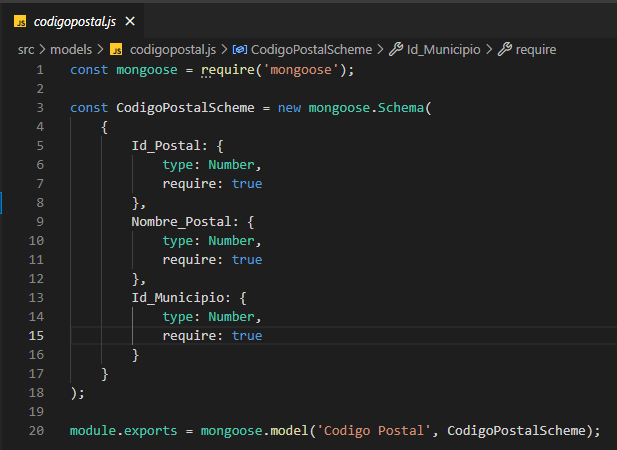


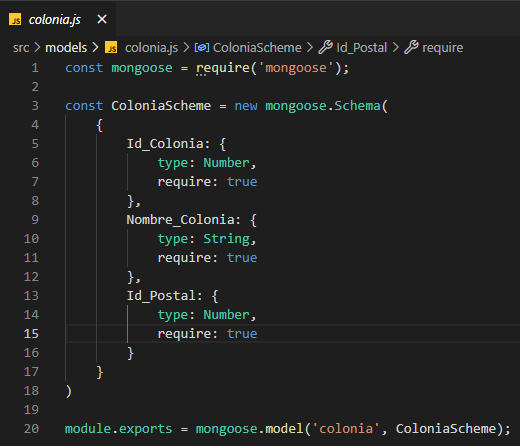


Dentro de ella abrimos el archivo que le pusimos el nombre de área y vamos especificar los datos que vamos a ocupar en la base de datos.

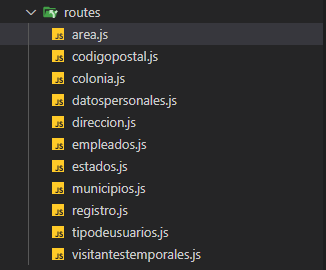


Estas líneas de código los podemos usar en los demás archivos nada más lo que tenemos que hacer es cambiar atributos y los nombre que va tener dentro de la base de datos.

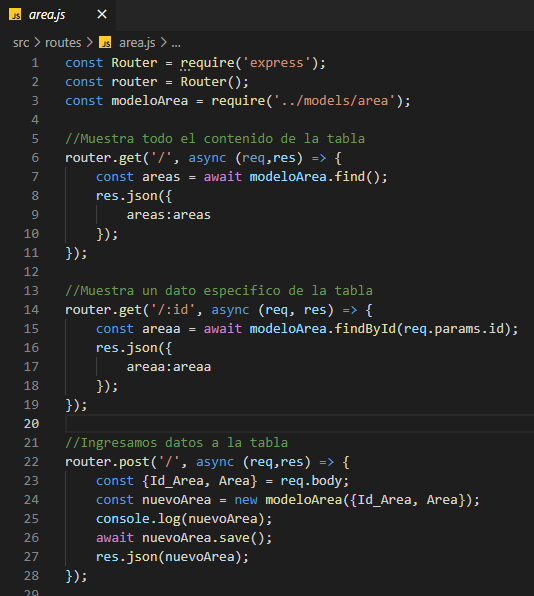


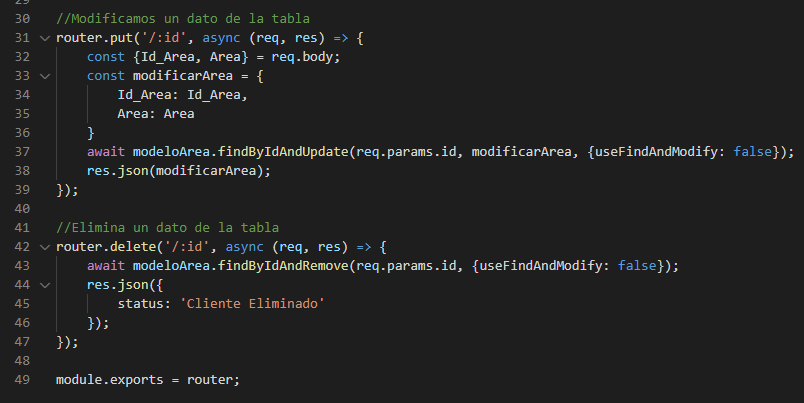


Lo que sigue es abrir la carpeta de routes y crear los mismos archivos que tiene models y config. En esta carpeta lo que hace es declarar las rutas que se van a usar al momento de navegar en internet y también que función va ser al momento de trabajar con ellas.



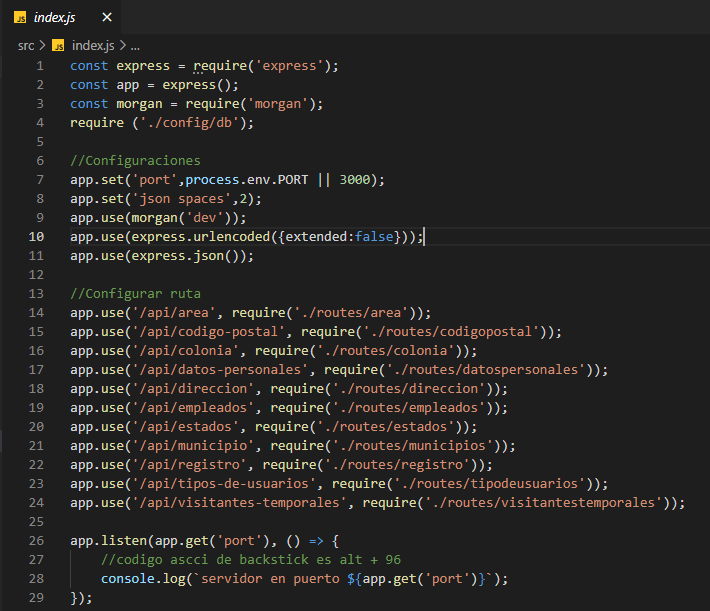
Abrimos el archivo de área y vamos a usar el siguiente código.



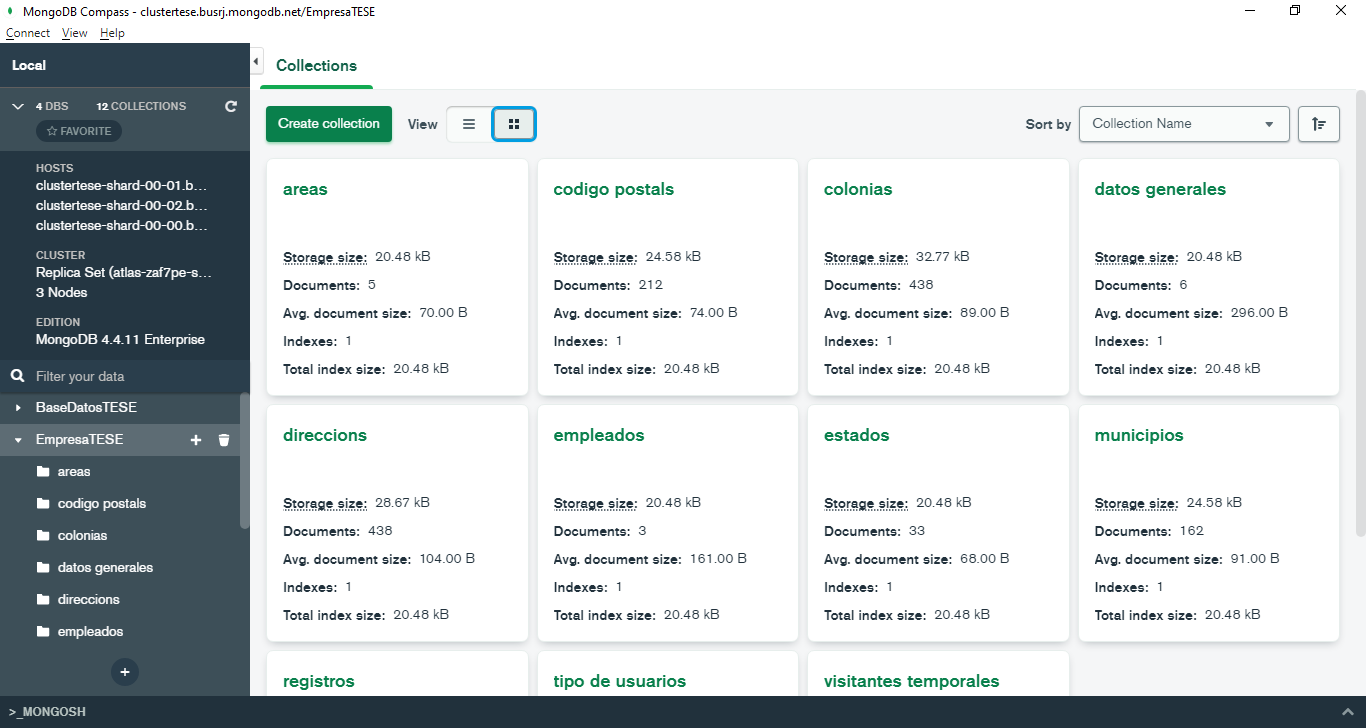


Estas líneas de código las podemos usar en los otros archivos nada más hay que cambiar la ruta de archivo models y también los parámetros de lo que contienen la tabla.

Lo que sigue abrimos el archivo index.js y vamos a usar el siguiente código dentro de ella, ahí vamos a especificar las rutas que se van a trabajar, en iniciar la base de datos y el puerto con el que vamos a trabajar al momento de navegar en internet.

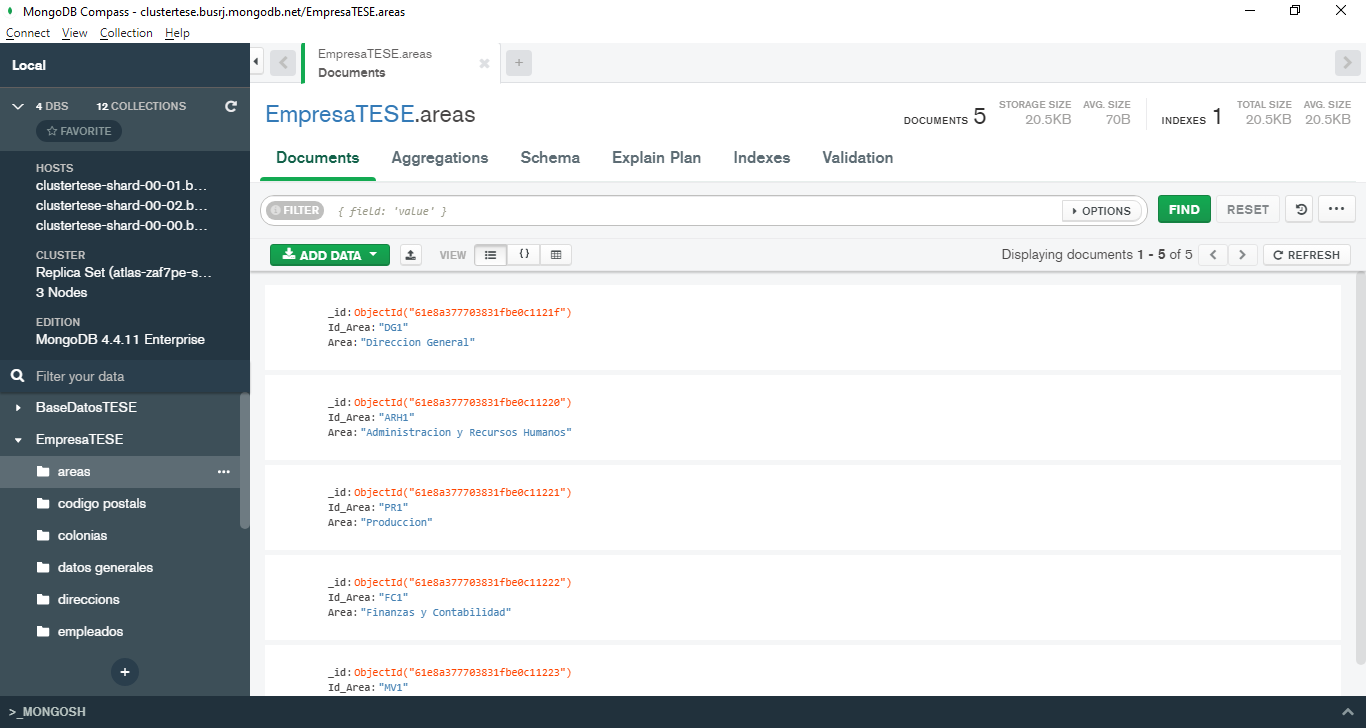


Lo que sigue abrimos lo que es Mongo DB para ver que nos creó la tabla y sus apartados que se van a ocupar.



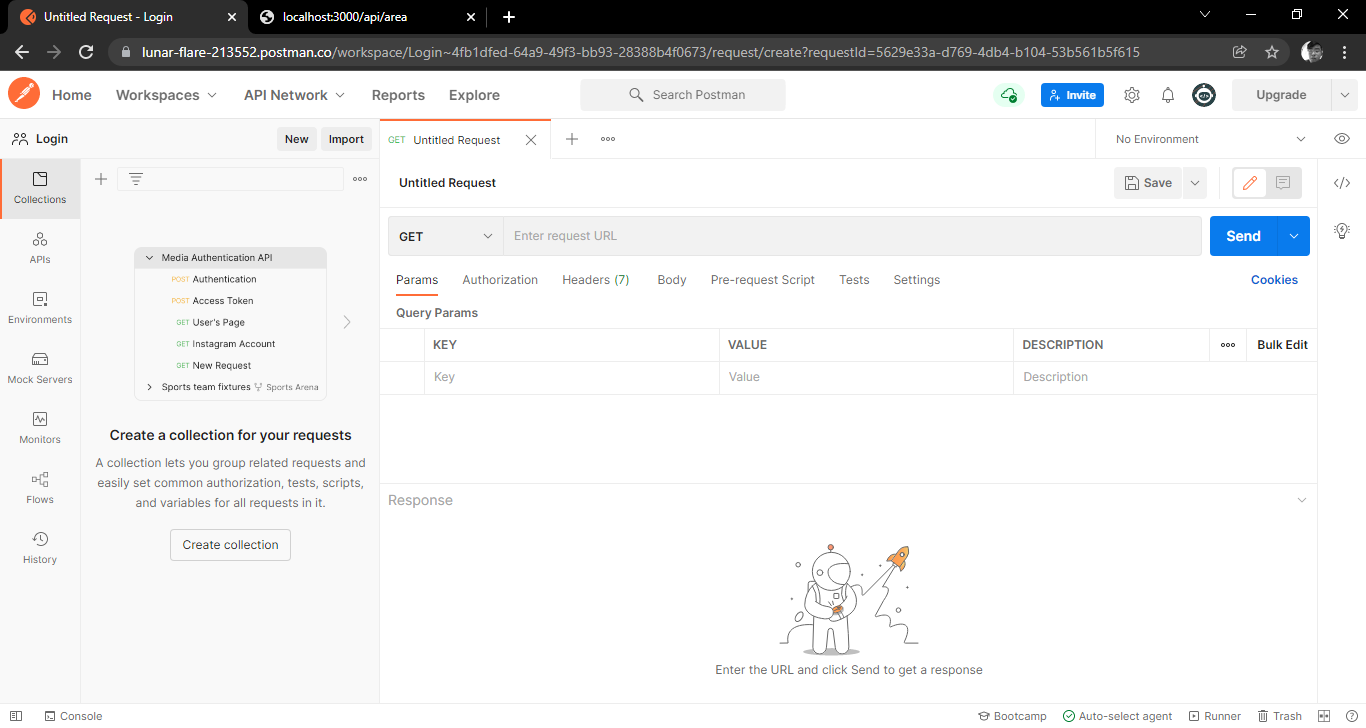
En la imagen anterior podemos ver que nos creó las tablas y nombre que especificamos que era EmpresaTESE.

Lo que sigue es abrir el apartado de área darle clic en ADD DATA y subir un archivo que tenemos creado. Hay que hacerlo con los demás apartados.



Lo que sigue abrimos el navegador y vamos a entrar la siguiente enlace <https://www.postman.com/> , creamos una cuenta y dentro de esa dirección.

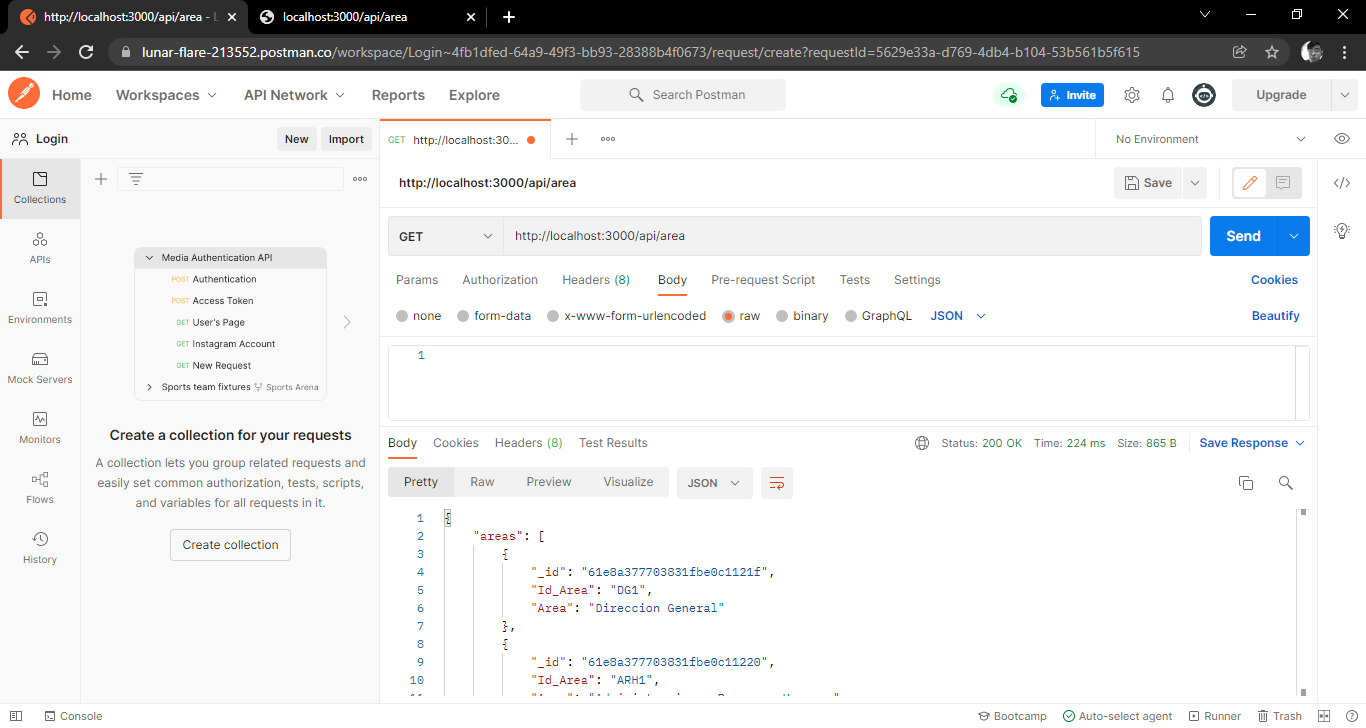
En la barra de esa página nos dirigimos a lo que dice workspaces y le damos al botón de créate workspace.



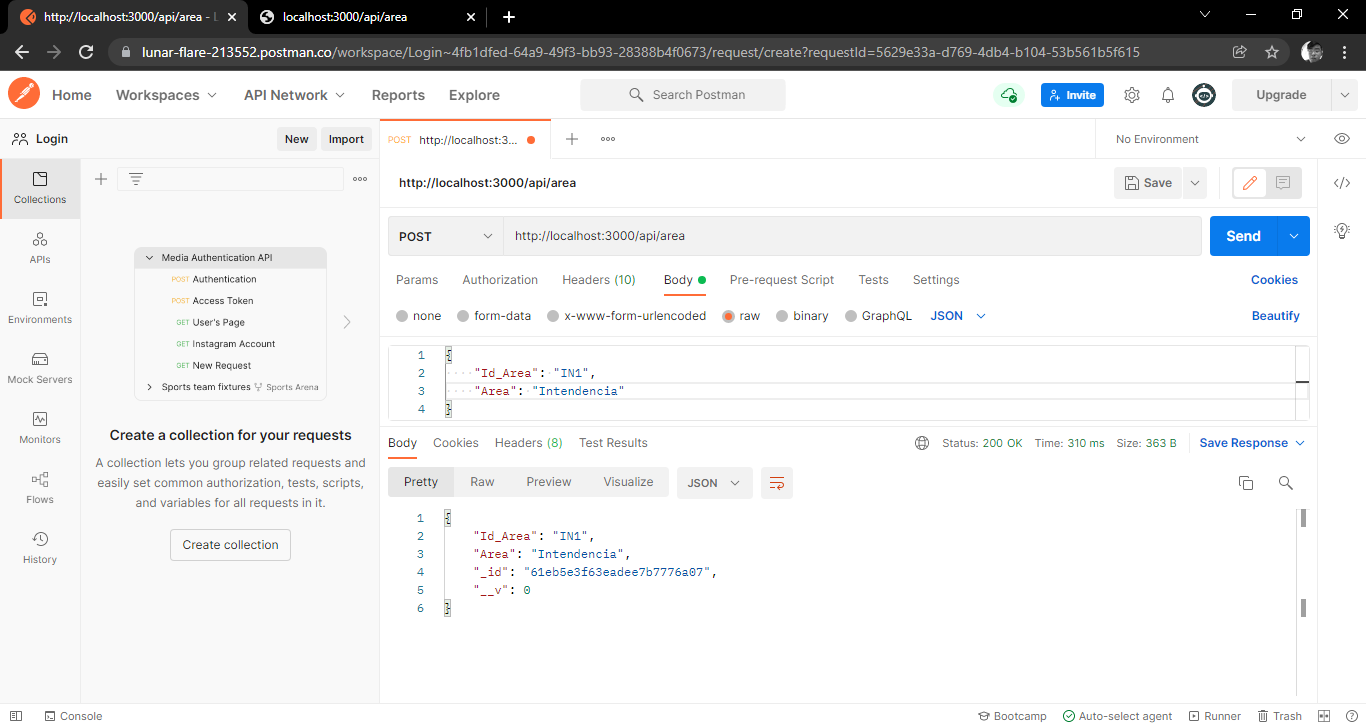
Nos vamos en el apartado de Headers, en el apartado de KEY colocamos content-type y en el apartado de VALUE colocamos application/json.

En esa página hay una barra donde podemos colocar la dirección de nuestro servidor y la parte de izquierda hay una caja de opciones una con get para poder hacer la consulta, otra con post para ingresar datos, otra put para modificar datos y delete para eliminar damos.

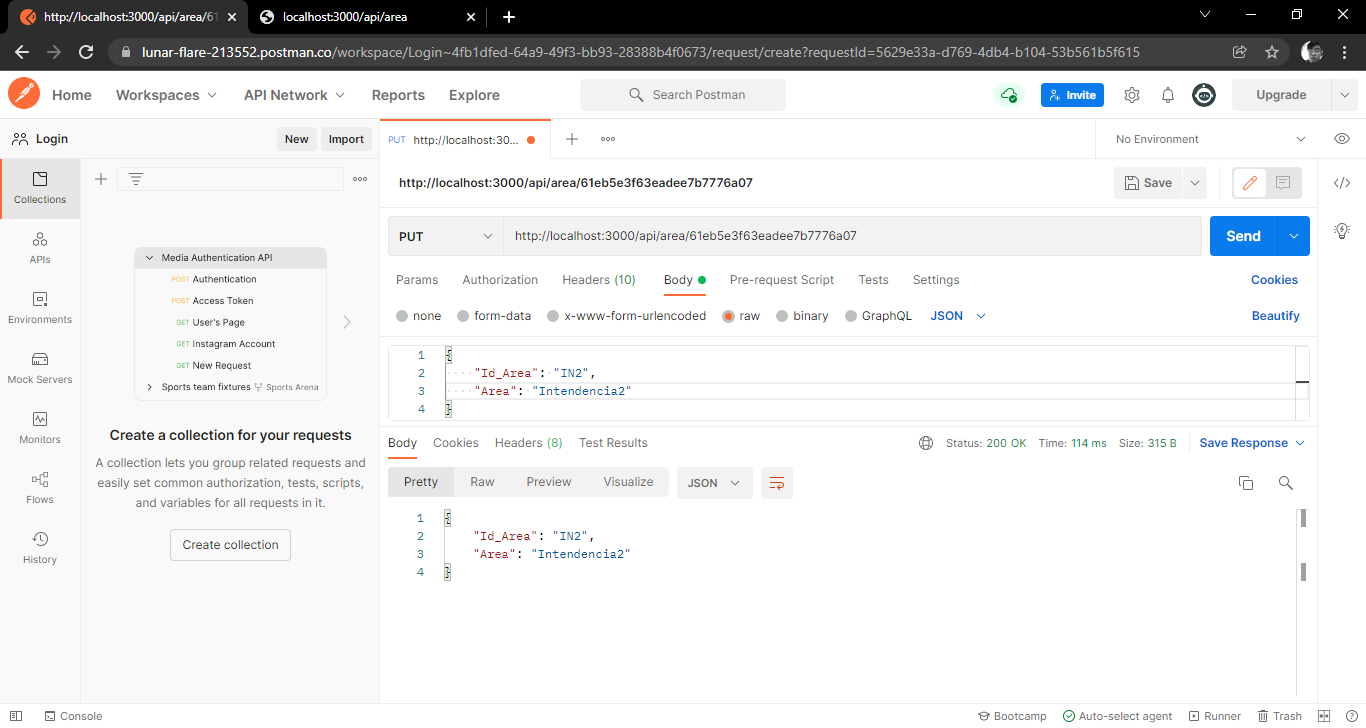
Vamos a usar el de get para poder consultar.

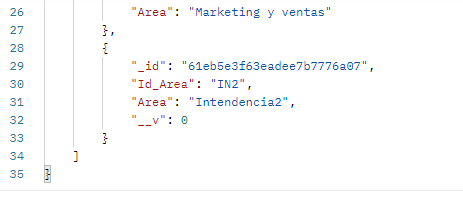


podemos ver que nos hacer la consulta de la parte de área, lo que sigue es usar post para ingresar un nuevo dato.

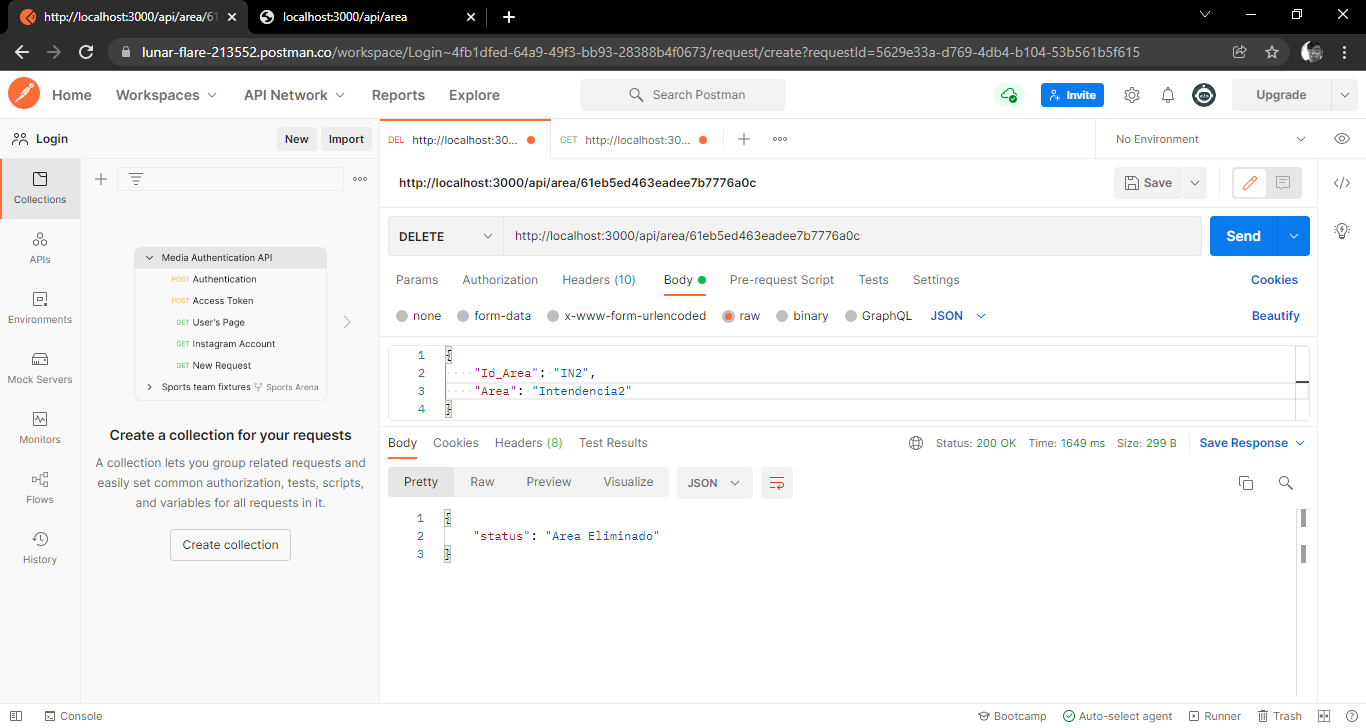


Podemos observar que nos ingresó un nuevo dato, lo que sigue es usar put para modificar el dato, pero tenemos que usar la id que le creo mongo y colocar en nuestra ruta ya que no especificamos la id no va hacer modificaciones.





Podemos observar que nos modificar el dato y por último vamos a eliminar ese dato, vamos a ocupar la misma id para localizarlo, ya que también si no especificamos la id no va eliminar un dato.



En la imagen anterior podemos ver que elimino el dato.

# **Base de datos con vistas en mongoDB del módulo asignado**

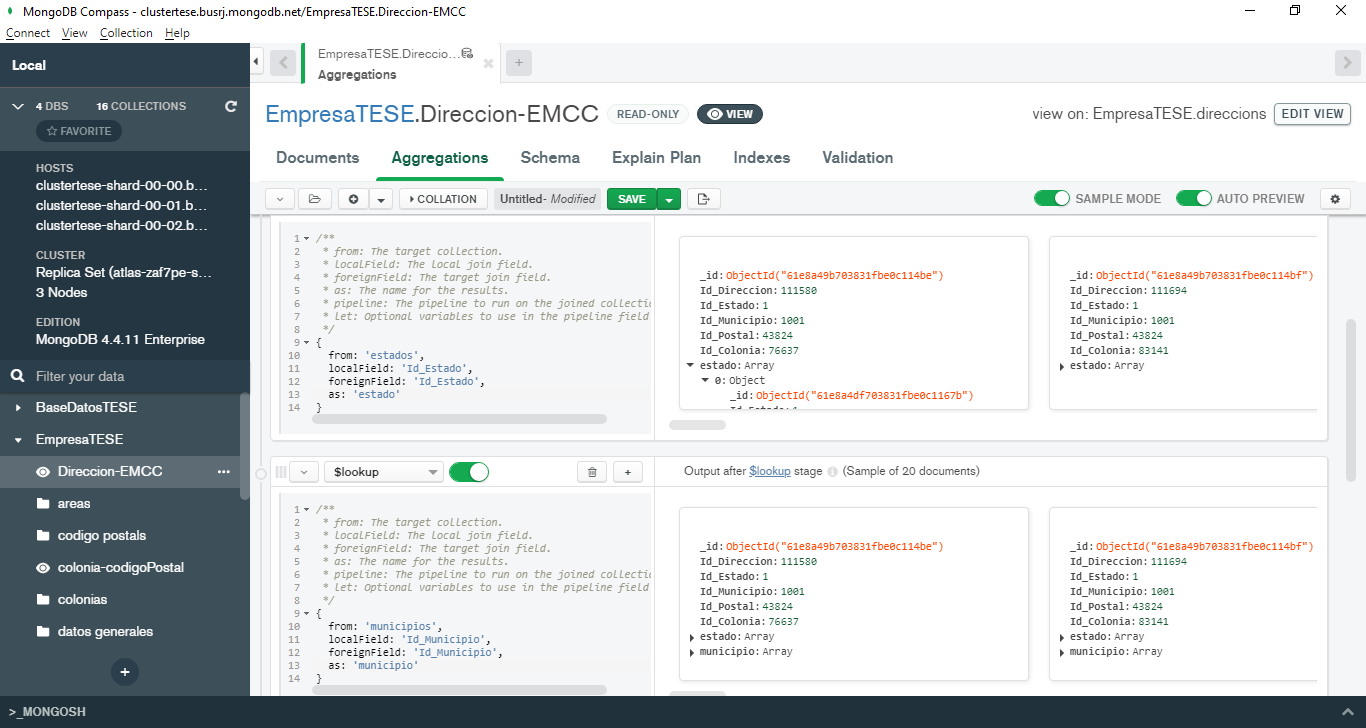
Ya una vez que creamos el backend de nuestro servidor, lo que sigue es crear las vistas de las tablas con las que vamos a trabajar. Las tablas que trabaja nuestro modulo son:

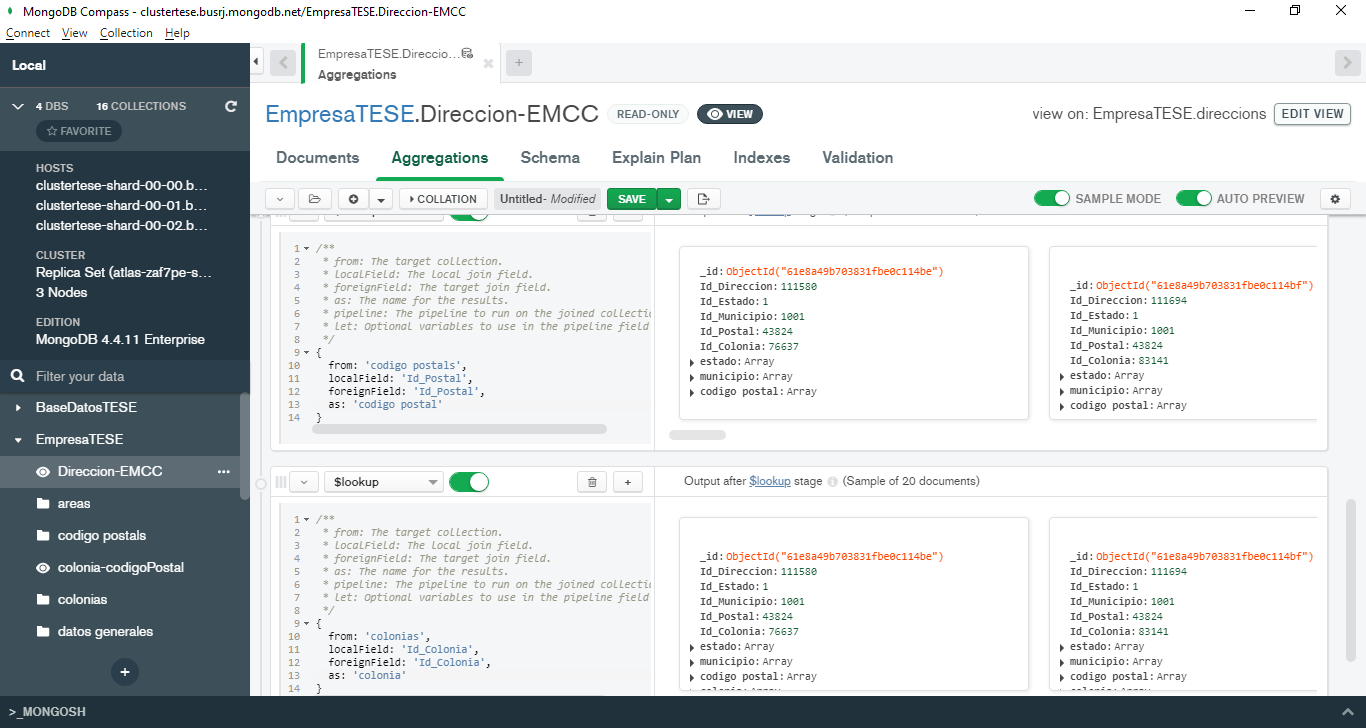
* Registro
* Datos Generales
* Dirección
* Estados
* Municipios
* Código Postal
* Colonias

Abrimos lo que es direcciones y vamos hacer la relación correspondiente que es:

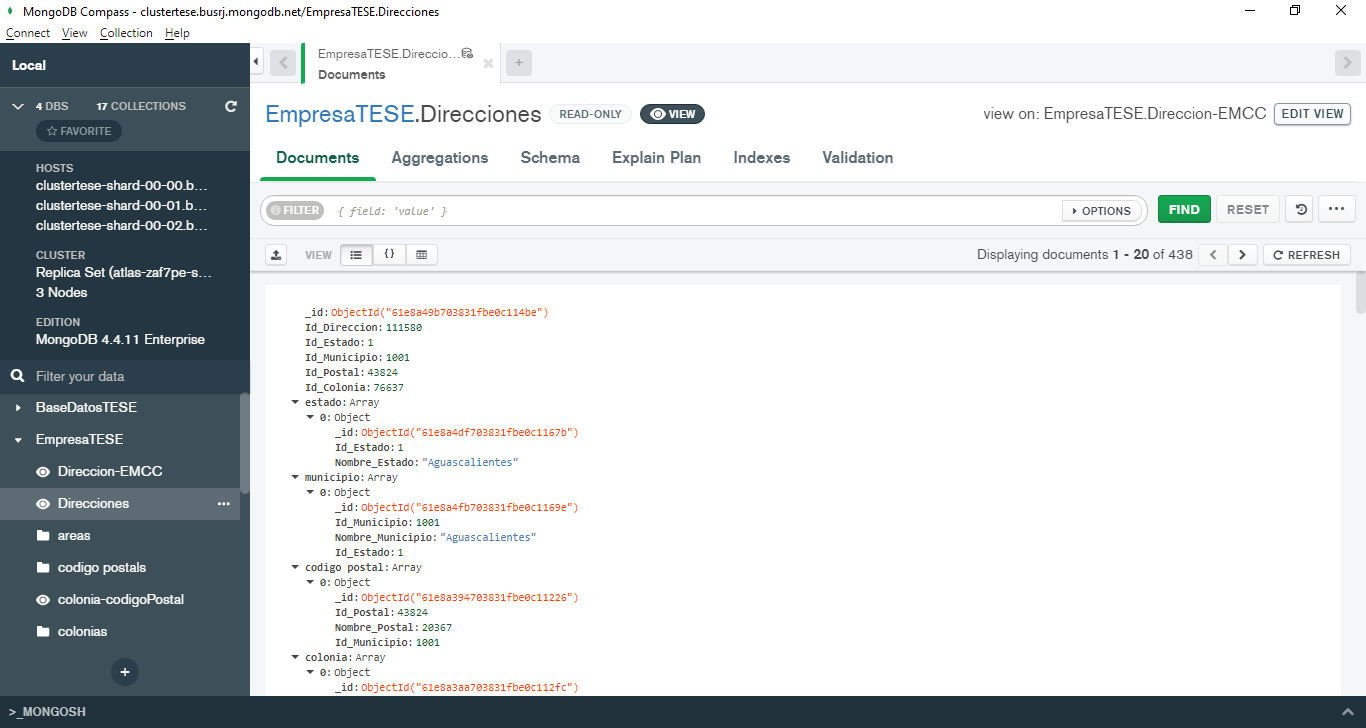
* Dirección – Estado
* Dirección – Municipio
* Dirección – Código Postal
* Dirección – Colonia

Nos dirigimos al apartado de Aggegations y el parte de abajo vamos hacer las relaciones con la tabla. Buscamos la opción Lookup para hacer las relaciones.

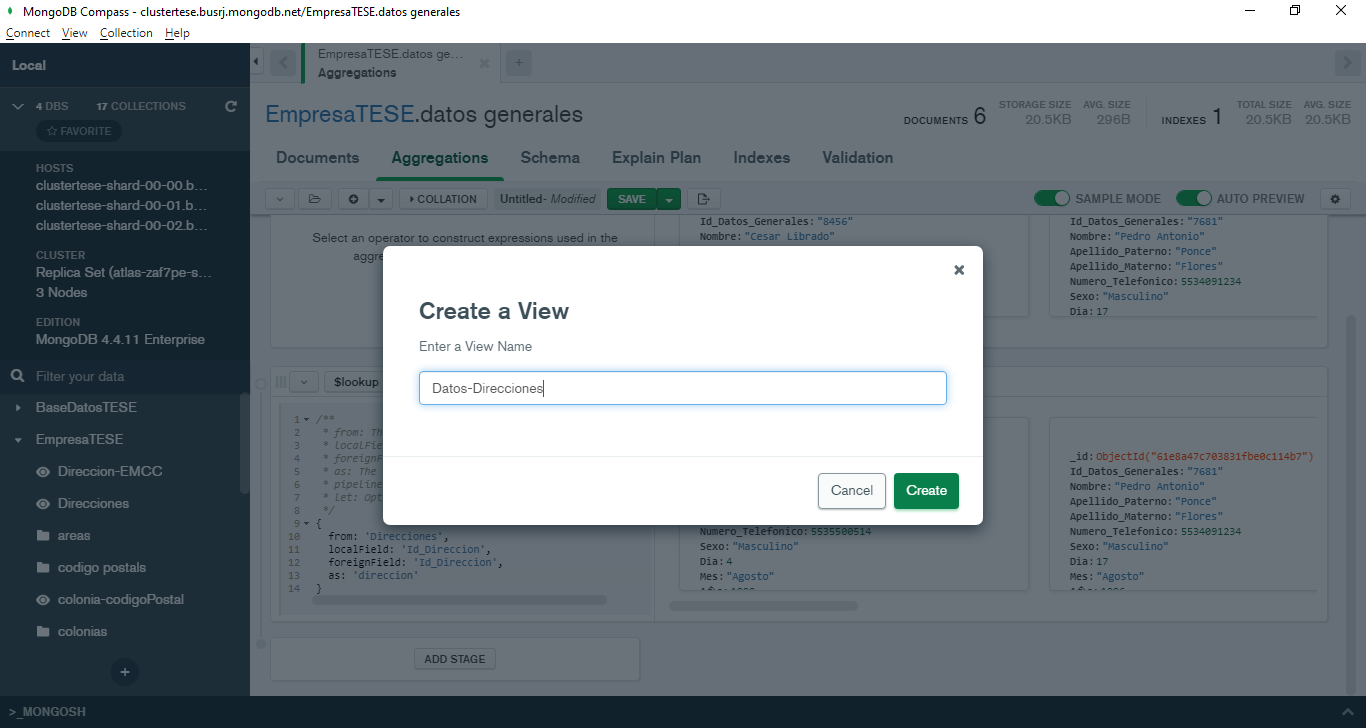


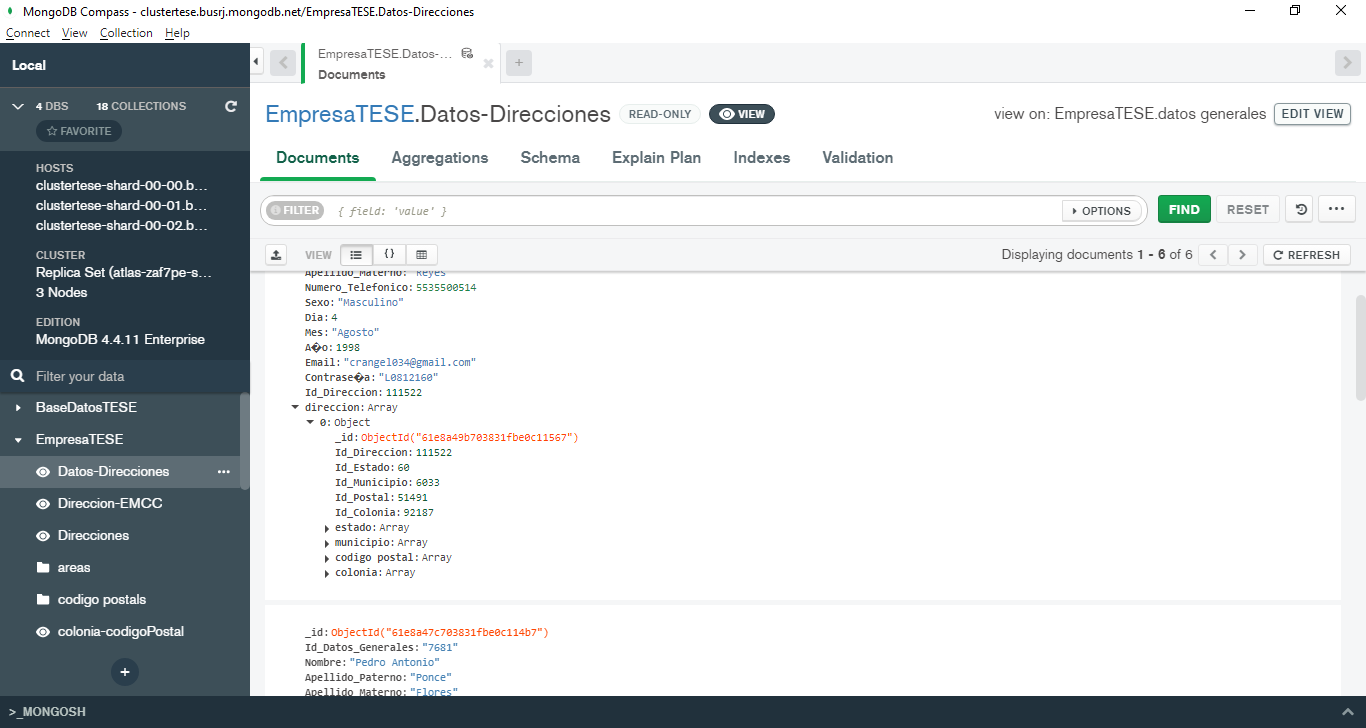


Lo vamos a guardar como tipo Views para poder observar las vistas.

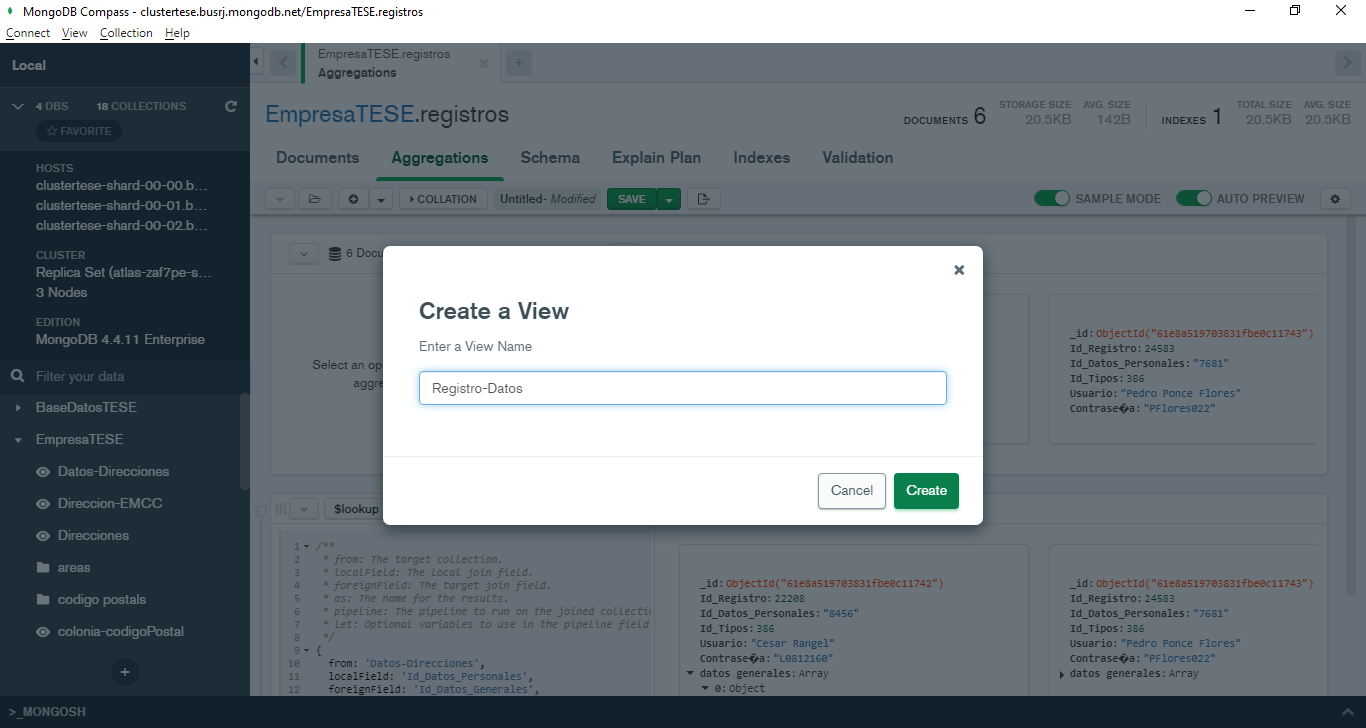


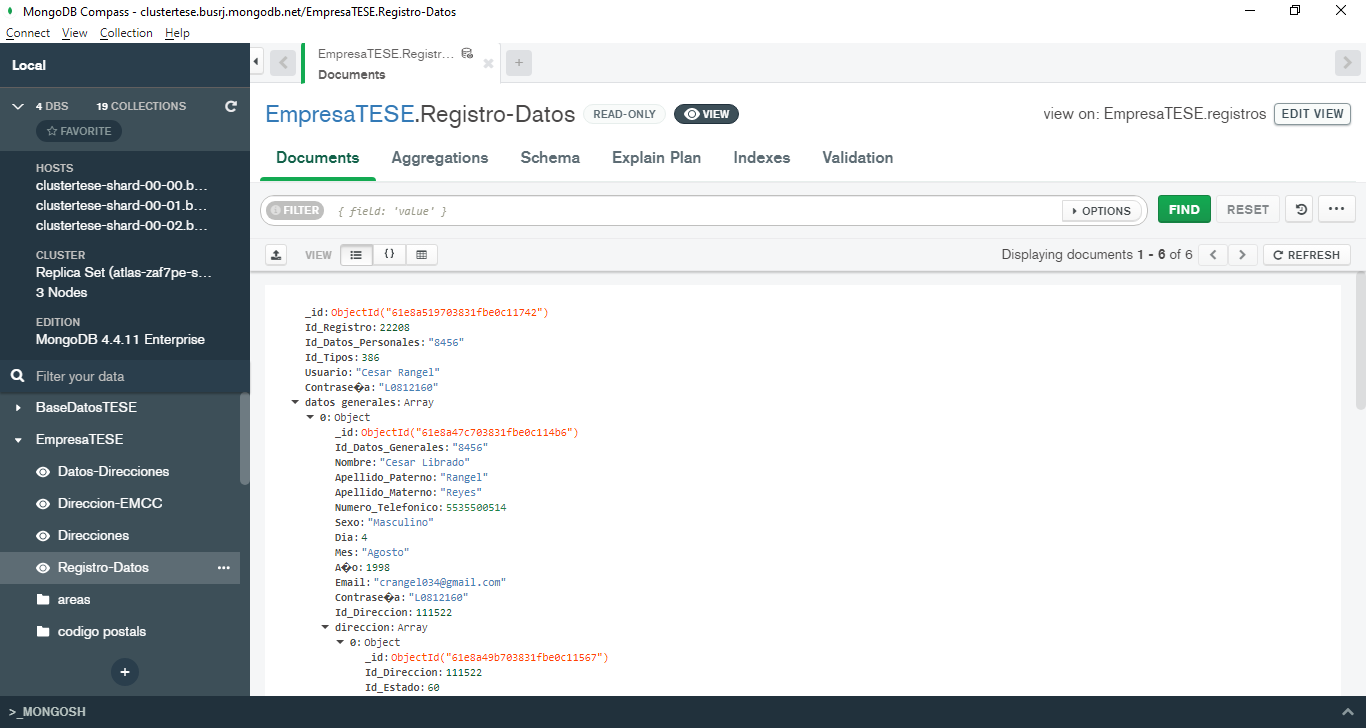
Lo que sigue es hacer relación de Direcciones con datos generales. Vamos hacer los mismos pasos anteriores para poder crear la vista.





Y por último vamos hacer a vista de Datos-Direcciones con registro, vamos a repetir los mismos pasos para crear la vista.





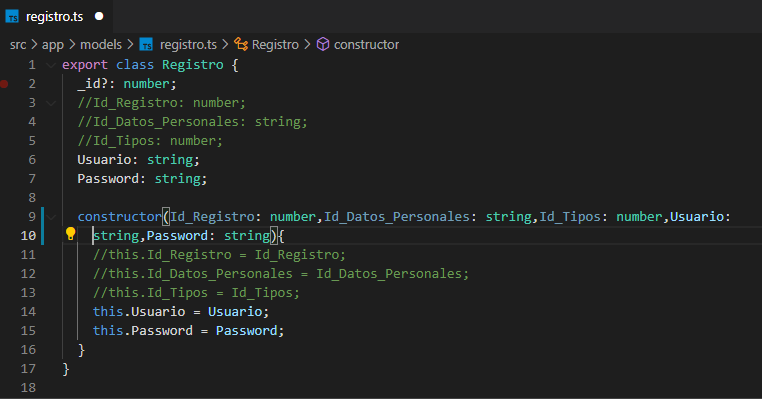
# **Visualización de datos en su Frontend**

Lo que sigue es conectar nuestro FrontEnd (La página que creamos) con el BackEnd (El servidor que creamos) para que podamos trabajar al mismo tiempo.

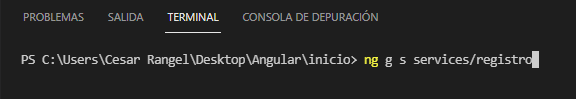
Creamos dos carpetas nuevas una con nombre de models y services.



En la carpeta models creamos un archivo con el nombre registro.ts lo abrimos y va tener las siguientes líneas de comandos. Con ellos nos va ayudar poder trabajar con la tabla de registro del servidor que creamos.



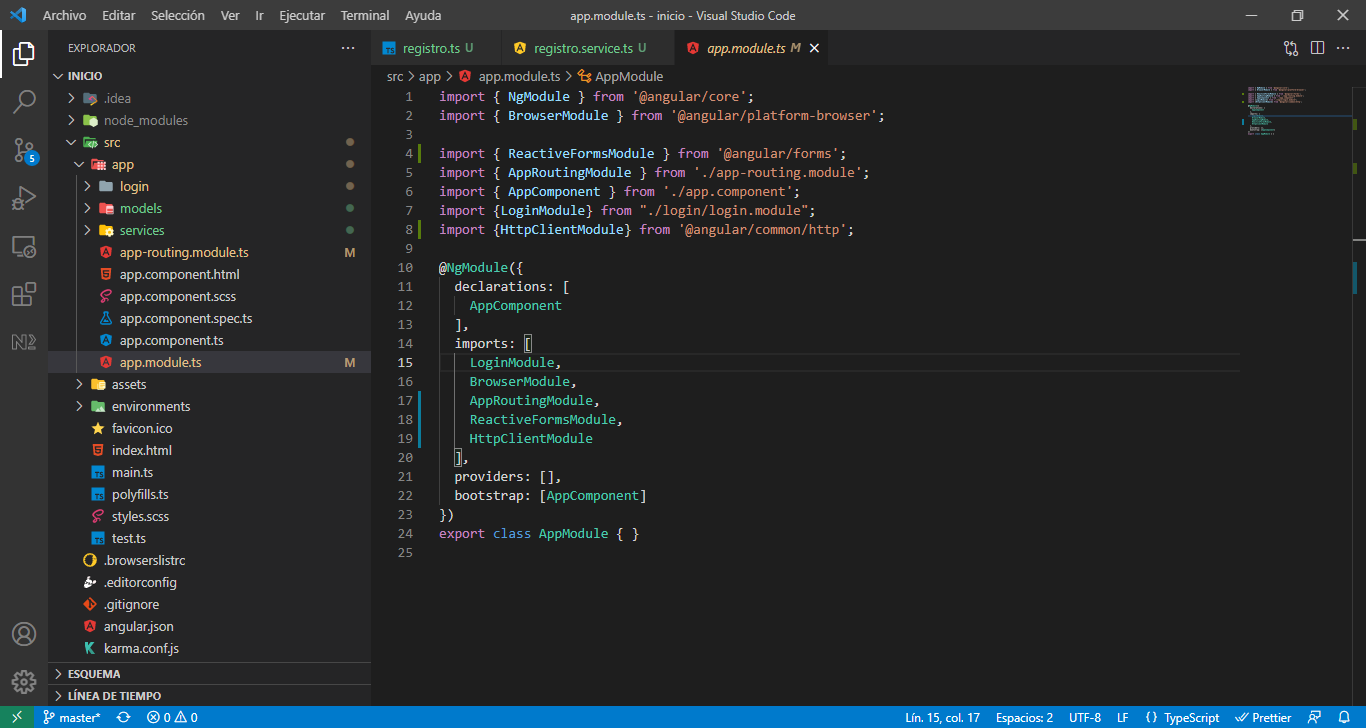
En el terminar ejecutamos el siguiente comando para crear en la carpeta de services un nuevo archivo.



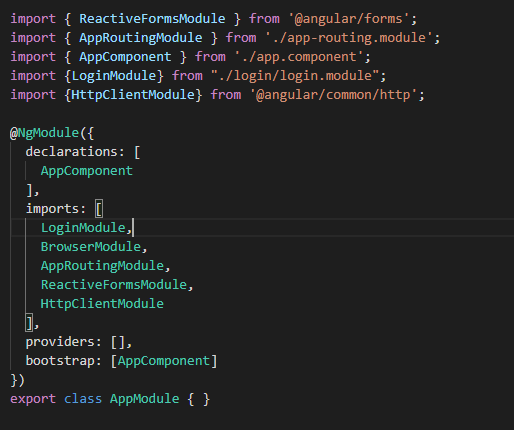
Cuando se crea nos coloca dos nuevos archivos, pero eliminamos uno y nos quedamos nada más con el que dice registro.services.ts



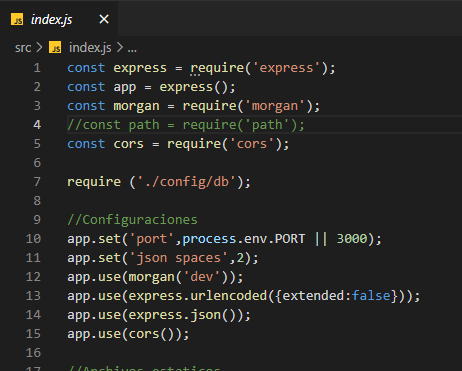
Después abrimos el archivo de app.module.ts y nos vamos al apartado de imports e importamos el que se llama ReactiveFormsModule.



Después nos vamos al apartado de app.module.ts e importamos la HttpClientModule. Con ellos permite que lea la liga del servidor que nosotros llegáramos ocupar dentro de la página.



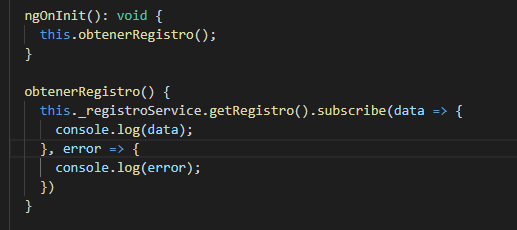
Después en la carpeta de donde creamos el servidor instalamos cors para no tener problemas al momento de comunicarse. Y lo importamos en nuestro archivo index.js



Nos dirigimos el archivo de registro.servece.ts y agregamos la url y los demás apartados para el get y el post con los que vamos a trabajar.



Nos dirigimos al archivo acceso.component.ts y agregamos el siguiente código. Esta parte nos muestra lo que contiene la tabla de registro.



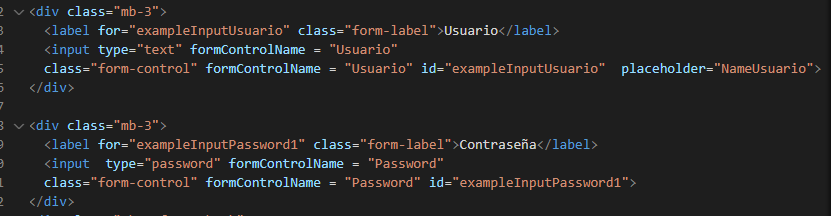
Después vamos a usar la siguiente línea de comando. Con ellos podemos trabajar con los campos de la parte de iniciar sesión.



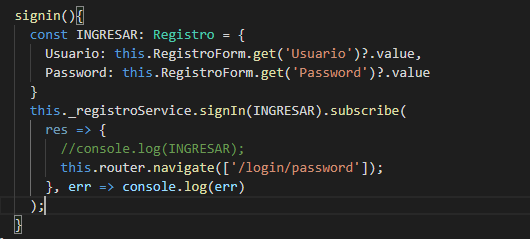
Abrimos acceso.component.html y la línea de form lo mandamos a llamar.



Mandamos a llamar los espacios que vamos a ocupar para podernos loguear dentro de la página ya creada.



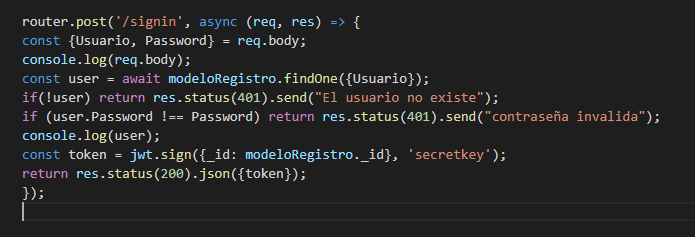
En acceso.component.ts agregamos el siguiente código. Con ellos nos permite trabajar con los campos de Usuario y Contraseña para podernos loguear y especificamos a que liga o que parte lo valla a re direccionar al momento de iniciar sesión.



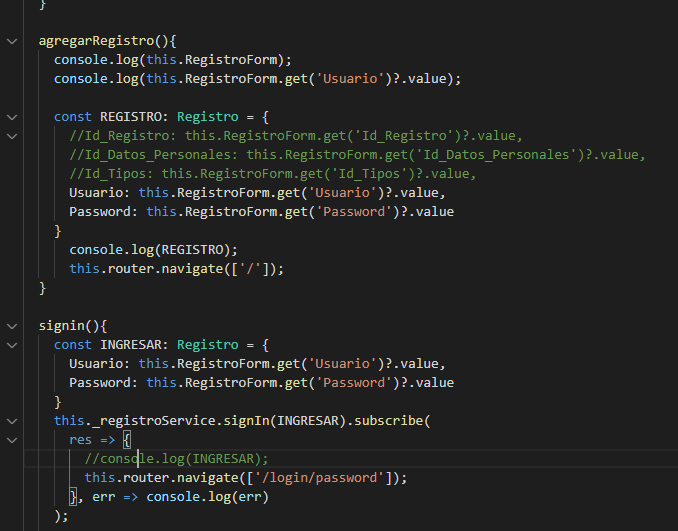
En la parte de acceso.component.html en la parte que agregamos formGroup agregamos el siguiente comando. Con ellos nos permite trabajar con el código del paso anterior.



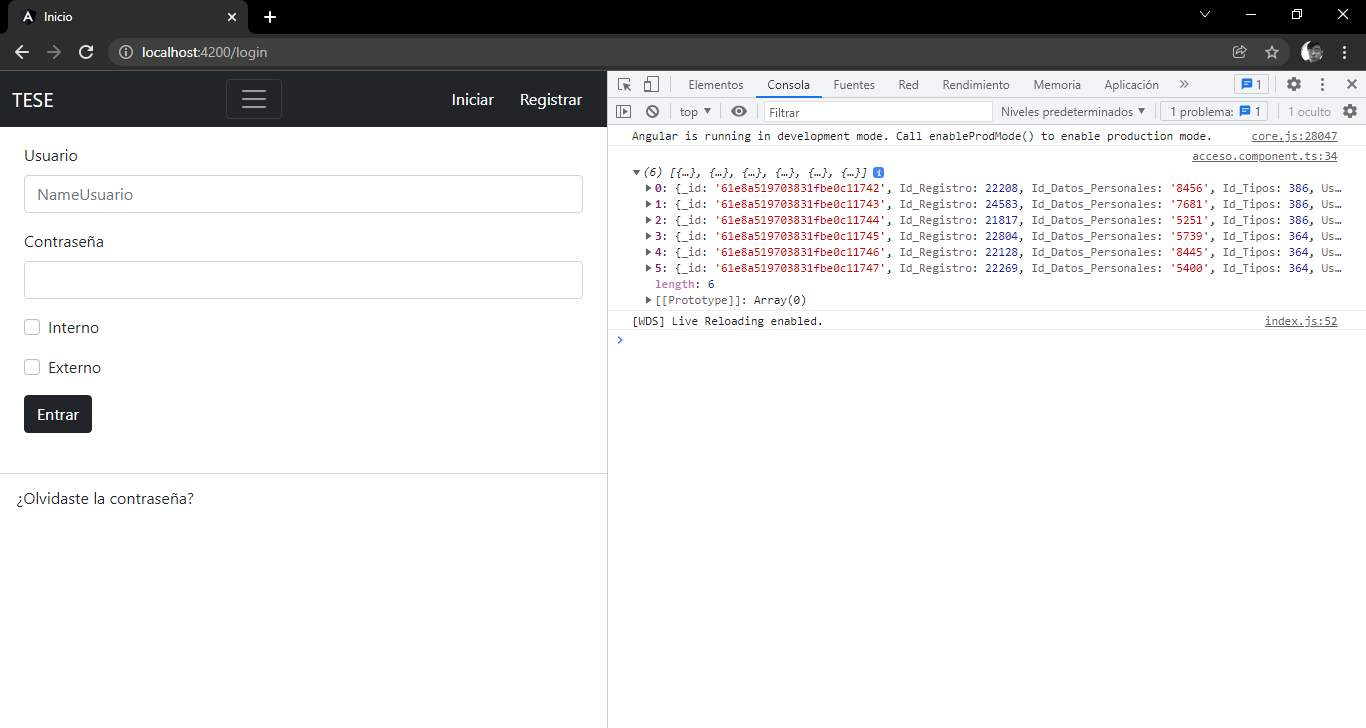
Después vamos a usar el siguiente comando para poder usar una dirección que creamos en el servidor. En la primera imagen vamos a mostrar lo que hace el código der servidor para poder iniciar sesión dentro de nuestra página.



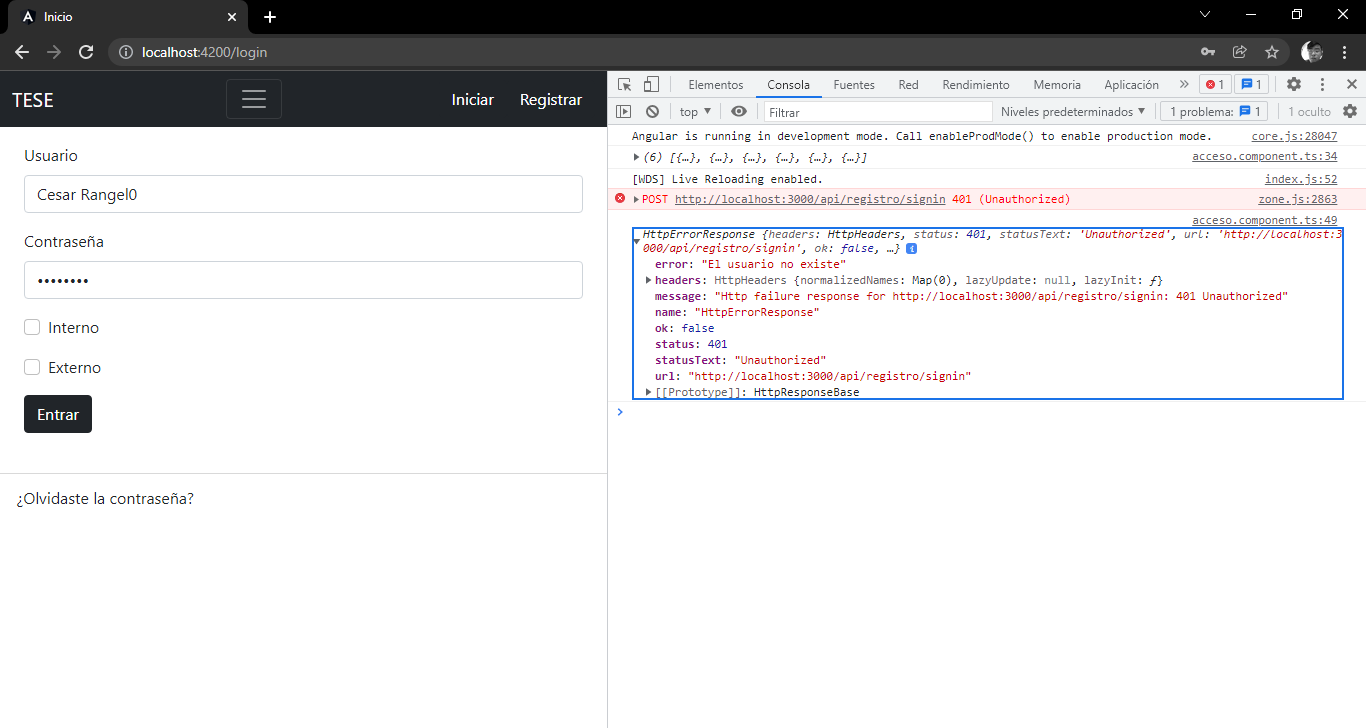
En la segunda imagen nos dirigimos al apartado de acceso.component.ts para agregar los dos apartados que agregamos de registro.service.ts.



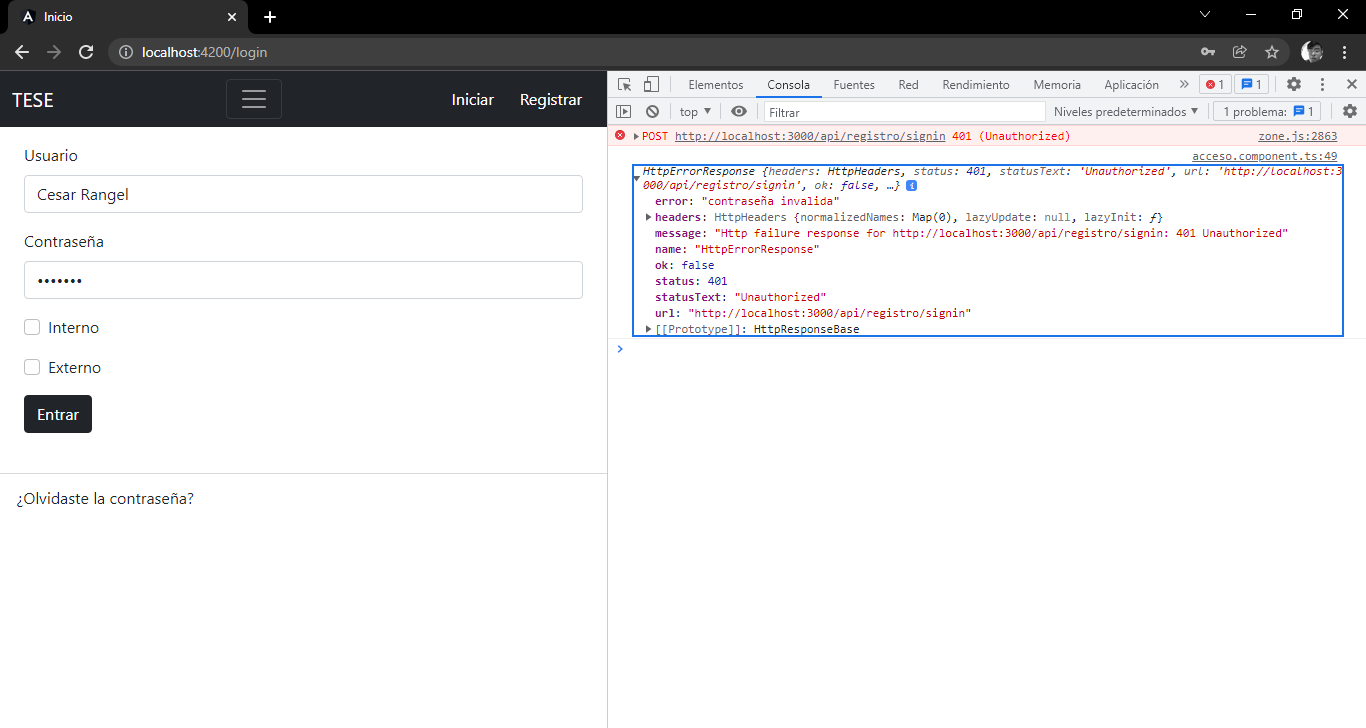
En la siguiente imagen nos dirigimos al programa que se anda ejecutando, le damos clic a iniciar y le damos clic derecho nos vamos a inspeccionar y le damos al apartado de consola y podemos ver que esta página trabaja con la tabla de registro.



Lo que sigue es agregar un usuario y ver que dentro de consola nos va arrojar un error y nos va indicar el error que es usuario no existe.



Luego perdonemos al usuario existente pero ahora ponemos la contraseña mal y nos va mostrar un mensaje de contraseña invalida.



Ya una vez que metamos los datos correctos nos va poder re direccionar a la liga que nosotros especificamos en el código al momento de iniciar sesión.

# **Enlace del Proyecto Login en GitHub**

El diseño de la pagina

<https://github.com/CesarLibradoRangelReyes/Login-Proyecto/tree/master>

El servidor

<https://github.com/CesarLibradoRangelReyes/ServidorLogin>

# **Manual Técnico**

Herramientas utilizadas para el desarrollo y el proceso.

|  |  |
| --- | --- |
| WebStorm Logo Vector (.SVG) Free Download | WebStorm: es el IDE de Programación (entorno de desarrollo integrado) de JavaScript de JetBrains, proporciona asistencia de codificación inteligente. es un IDE profesional JavaScript que es compatible con una amplia gama de tecnologías modernas relacionadas con el lenguaje de programación JavaScript, HTML y CCS, además ofrece la experiencia completa para el desarrollo Web productivo. |
| C:\Users\Cesar Rangel\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\2770CDF1.tmp | Draw.io es una herramienta de creación y edición de diagramas libre que permite la integración con diversas plataformas. ... Esta herramienta está disponible para trabajar en línea pudiendo guardar los diagramas en Google Drive, OneDrive o en local. |
| Bootstrap 4. ¿Vale la pena actualizar? « SEOCOM | Bootstrap es un framework front-end utilizado para desarrollar aplicaciones web y sitios mobile first, o sea, con un layout que se adapta a la pantalla del dispositivo utilizado por el usuario. Tiene varios recursos para configurar los estilos de los elementos de la página de una manera simple y eficiente, además de facilitar la construcción de páginas que, al mismo tiempo, están adaptadas para la web y para dispositivos móviles. |
| Angular (framework) - Wikipedia, la enciclopedia libre | Es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.  Angular se basa en clases tipo "Componentes", cuyas propiedades son las usadas para hacer el binding de los datos. En dichas clases tenemos propiedades (variables) y métodos (funciones a llamar).  Angular es la evolución de AngularJS, aunque incompatible con su predecesor. |