**Backend inicial**

-En una terminal correr mongod, que es la conexión a la bd

-En la otra terminal abrir el compilador de typescript desde la carpeta del proyecto: tsc -watch

-En la otra terminal desde la carpeta del proyecto abrir nodemon dist/

-El comando - -save es una biblioteca importante porque se necesita para que el proyecto funcione.

-npm es un gestor de paquetes de nodejs

-typescript: da mas ayuda para encontrar fácilmente el error en el código.

1.-tsc - -init

2.-npm init desde el directorio de la terminal (con esto se instala el packe.json)

corese: es para que corra de igual forma que corre en mi maquina, en otra, intercambio de recursos de origen cruzado, si el usuario envía un método get y el server tiene un método rebota al usuario.

Body parser: le da formato a una petición (request) que hace el usuario para que sea mas legible, dándole formato json.

para cambiar de directorio escribir en la consola: cd espacio arrastrar la carpeta del proyecto y darle enter, para estar dentro de la carpeta del proyecto.

1.- Crear la carpeta del proyecto

2.- Abrir la carpeta en el visual studio code

3.- Abrir la terminal de comando o el CMD de Windows, navegar al directorio que acaban de crear e inicializar el proyecto con el comando “npm init”, llenar los datos que solicita el comando.

4.- Instalar express con el comando “npm install express”.

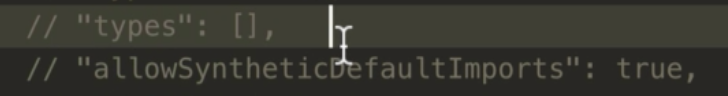
5.- Instalar CORS (Cross Domain Origin) con el comando “npm install cors”.

6.- Instalar Body Parser con el commando “npm install body-parser”.

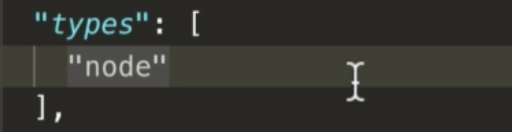
7.- npm install -g typescript

Con esto tenemos inicializado el proyecto, como vamos a trabajar directamente con typescript, en la terminal debemos escribir “tsc - -nit” para inicializar el proyecto de typescript.

Una vez hecho esto, desde el visual studio code debemos buscar el archivo “tsconfig.json” y vamos a decomentar types:



Dentro de los corchetes debemos agregar a node, tiene que quedar de la siguiente manera:



Una vez hecho esto, dentro del archivo buscamos la linea que hace referencia al directorio de exportación, esta marcado como “outDir”:

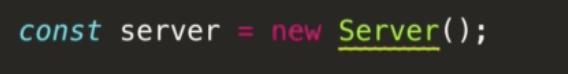


Con esto indicamos en que directorio queremos que nuestra aplicación sea compilada, es decir, todo lo que hagamos en typescript va a ser traducido a su sinonimo de javascript, para seguir el orden que llevamos en los proyectos,, la linea tiene que quedar de la siguiente manera:



Despues es necesario crear el archivo en el que nuestra aplicación sera lanzada, para esto, en la raiz del proyecto crearemos el archivo “index.ts”.

Dentro del index.ts, creamos una constante llamada server que va a llamar a la clase Server que haremos mas adelante, de esta manera:



abrimos una terminal en el proyecto y ejecutamos

$ npm install @types/express (son ayudas para que no nos mande errores al exportar el express).

Posteriormente es necesario crear en la raiz del proyecto la carpeta “clases” en esta colocaremos todas la clases que vayamos desarrollando.

Dentro de la carpeta “clases” crearemos un archivo llamado “server.ts” dentro de este crearemos nuestra clase:



Debemos definir “SERVER\_PORT”, para esto, en la raiz del proyecto creamos una carpeta llamada “global”, dentro creamos un archivo llamado “environment.ts” en el vamos a crear una constante de tipo numero, que nos indicara en que puerto nuestro servidor esta a la escucha de peticiones, asi mismo vamos a exportarla para poder usarla desde cualquier parte de nuestra aplicacion:



Una vez finalizado esto, volvemos al archivo “index.ts” e importamos la clase que acabamos de crear y arrancamos nuestro servidor:



Antes de correr el servidor, asegurense de tener nodemon instalado de manera global, esto se hace con el comando “npm install -g nodemon” para windows deben abrir la terminal como administrador, en mac basta con colocar “sudo” antes del npm install.

Dicho esto, abran 2 terminales y en ambas dirijanse a la carpeta del proyecto, en una van a colocar “tsc –watch” y en la otra “nodemon dist/” con esto deberia correr su servidor.

**Conexión a base de datos MongoDB**

Pasos para conectar el backend con la base de datos:

1. Abrir una terminal.
2. Ir al directorio del proyecto.
3. Instalar mongoose con el comando “npm install mongoose - -save”
4. Ir al archivo “index.ts”.
5. Importar la bilbioteca de mongoose. Escribiendo

var mongoose = require(‘mongoose’);

1. Por ultimo, agregar al “index.ts” las siguientes líneas



Nota: este pedazo de código debe ir antes de “server.start”.

Si manda error en la variable ( err ) => declararla como tipo any ( err: any ) =>

Con esto ya tenemos nuestro backend listo para conectarse a la base de datos.

-Crear la carpeta middlewares, dentro de ahí crear el archivo autentication.ts

instalar npm install jsonwebtoken

instalar npm instal @types/jsonwebtoken

si todo esta correcto no debe marcar error en las 3 consolas

Middleware; intercepta y verifica peticiones, sirve como mediador entre el usuario y el backend.

Express: es una biblioteca que te da las herramientas para hacer a node js un servidor.

Enviroment: se usa el enviroment para estar importando lo ahí escrito en otros lugares

Constructor: inicializa todo lo que se necesita para que tu clase funcione

Cerar rutas:

Crear una carpeta en la raíz del proyecto con el nombre interfaces y dentro un archivo usuario.ts con lo siguiente:

export interface Iusuario {

nombre: string,

apellido: string,

email: string,

password: string

}

crear otra carpeta en la raiz llamada modelos y dentro de ella usuario.ts,

instalar: npm install mongoose-unique-valdator - -save

instalar: npm install @types/validator

instalar: npm install @types/mongoose-unique-validator

instalar: npm install @types/cors

crear una carpeta en la raíz llamada ruta, crear el archivo usuarios.ts

importar las rutas en insdex.ts

login de usuario:

instalar: npm install bcrypt --save