Arquitectura del Sistema
Proyecto Tenjin

Natalia Pérez
Juan Juarez

26 de mayo de 2023

1. Estructura del proyecto

Para el proyecto, utilizamos una arquitectura orientada a servicios. Nuestra estructura se ve de la siguiente manera.

1.0.1. Backend

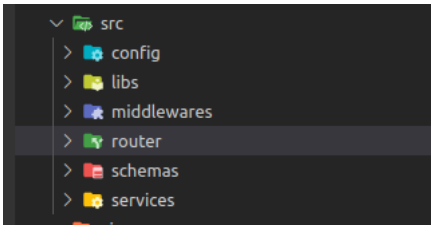

Figura 1: Estructura del proyecto

- **config**. En la carpeta config, encontramos las configuraciones con las variables de entorno para poder usarlas cuando sean necesarias.
- **libs**. Aqui incluimos librerias y funciones de utilidad que se utilizan. Por ejemplo, tenemos cludinary y multer para poder subir fotos. Y tenemos un archivo para trabajar con los tokens que generamos cuando un usuario inicia sesión.
- **middlewares**. Aqui incluimos los middleware necesarios para manipular el request antes de que pase por el callback del router.
- **router**. Aqui encontramos las rutas que definen la estructura de la API, las cuales se comunican con los servicios para completar las peticiones.

- **services**. La capa de servicios que se encarga de comunicarse con el manejador de base de datos y regresar los resultados al router.
- **schemas** Define los esquemas con los que los servicios realizaran las operaciones de la base de datos.

1.0.2. Frontend

Para el fronted, el router de views se encarga de mostrar la pantalla adecuada de acuerdo a la pagina a la que se quiere acceder, para lo que incluimos una carpeta de views donde se encuentran los archivos HTML que corresponden a las vistas de la aplicación.

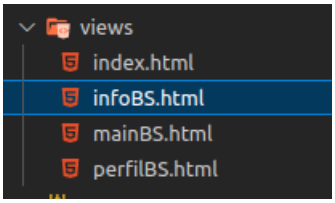

Figura 2:

En la carpeta public, encontramos los archivos CSS y javascript utilizados por el html que se encuentra en views.

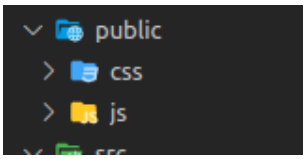

Figura 3:

2. Tecnologias utilizadas

2.1. Base de datos

Para la base de datos, utilizamos Mongo DB. Para hacer las peticiones y definir los esquemas que se utilizan, utilizamos el modulo **mongoose v8.18.0** para interactuar con la BD. En particular, son los servicios los que utilizan este modulo en conjunto con los esquemas definidos. En esta caso, el esquema de Usuario para definir la información de los usuarios de la plataforma, y el esquema

comentarios para definir la informacion de los comentarios que se pueden dejar en el perfil de un usuario.

3. Persistencia de Datos. Esquemas

3.1. Comentarios

```
const comentarioSchema = new Schema({
    comentario: {type: String, required: true},
    to: {type: String, required: true},
    calif: {type: Number, required: true}
});
```

Figura 4: Esquema del usuario

3.2. Usuarios

```
const usuarioSchema = new Schema(
    {
        nombre: {type: String, required: true},
        correo: {type: String, required: true, validate: custom},
        contrasena: {type: String, required: true},
        carrera: {type: String, required: true},
        // Esta es la foto por default
        fotoPerfilURL : {
            type: String,
            default: "https://res.cloudinary.com/dzya3fvwj/image/upload/v1674083608/noProfile_jaqk9o.png"
        },
        fotoPerfilId : {type: String, required: false},
        asesorias: [String],
        calif: Number,
        info: String
    }
);
```

Figura 5: Esquema del usuario

3.3. Lenguajes y Frameworks

3.3.1. Backend

Herramienta	Versión
Express Web framework for Node.js	4.18.2
bcrypt	5.0.1
cloudinary	1.37.0
cookie-parser	1.4.6
cors	2.8.5
dotenv	16.0.1
ejs	3.1.8
email-validator	2.0.4
jsonwebtoken	8.5.1
mongoose	6.3.4
multer	1.4.5-lts.1
nodemon	2.0.16
path	0.12.7
pug-cli	1.0.0-alpha6

3.3.2. Frontend

Herramienta	Versión
HTML	-
CSS	-
Javascript	-
bootstrap	5.2.3