Dirección Nacional de Niñez adolescencia y familia (DINAF)



Proyecto modulo migración

Presentado por

Héctor Josué Ardón Morga

Fecha

Miercoles 11 de Agosto del 2021

Contenido

[Introducción 3](#_Toc79579312)

[Herramientas utilizadas 4](#_Toc79579313)

[Pantallas diseñadas 4](#_Toc79579314)

[Back End 7](#_Toc79579315)

[Front End 11](#_Toc79579316)

[Cuadro comparativo entre datos de restitución internacional y base de datos sismo 12](#_Toc79579317)

[Propuesta modelo relacional Restitución Internacional 13](#_Toc79579318)

[Glosario de términos 14](#_Toc79579319)

[Bibliografía 14](#_Toc79579320)

# Introducción

Se ha planteado desarrollar un sistema para DINAF con el cual se pretende satisfacer las necesidades que requiere la gente que trabaja en migración generando reportes que satisfagan la información que se necesita para poder ayudar a las personas ya sean niños, niñas, y adolescentes (nna) con información que pertenece a ellos y poder facilitar la conexión entre los padres o encargados de ellos con los niños, cabe destacar que esto es algo que contribuye a mejorar la comunicación y que la información este lo más organizada y detallada posible para que el servicio que se le brinda a los niños pueda contribuir a que haya armonía con la familia que se le pretende integrar.

La plataforma principal de SISMO se encuentra utilizando el lenguaje de programación PHP, descubrimos que hay una forma de poder integrar el modulo que desarrolle a la plataforma de SISMO con una herramienta llamada servicios, esto brinda la facilidad de poder hacer el sistema en un lenguaje de programación diferente y poder adaptarlo a la plataforma de sismo, el lenguaje de programación utilizado ha sido JavaScript tanto para el Back End como para el Front End, el lenguaje JavaScript ha tenido mejoras al pasar de los años la última fue en el 2015 con la integración de Ecma Script 6, a esto se le conoce como desarrollo Full Stack, lo cual facilita un poco más la curva de aprendizaje al tener que aprender un solo lenguaje para integrarlo todo.

Hay muchas necesidades que satisfacer y es un proceso continuo de prueba y error, al final el objetivo es buscar la excelencia en nuestras labores y que las personas que utilicen el sistema como las personas que se ingresen con su información poder facilitar el trabajo que hay entre ellos y así contribuir al desarrollo de la nación.

Como dato adicional, quizás en el proceso de lectura de este informe puedan haber dudas o cosas difíciles de comprender entonces dejaré mi número de teléfono para dudas o sugerencias que puedan tener, es 99533283, estoy a la disposición de ayudarles a comprenderlo y también escuchar las criticas que tengan hacia mi persona.

# Herramientas utilizadas

Se han utilizado herramientas de software tanto en Front End como en Back End

Lenguaje de Programación

* JavaScript

Base de Datos

* Postgre SQL -> PgAdmin

Editor de código

* Visual Studio Code

Back End

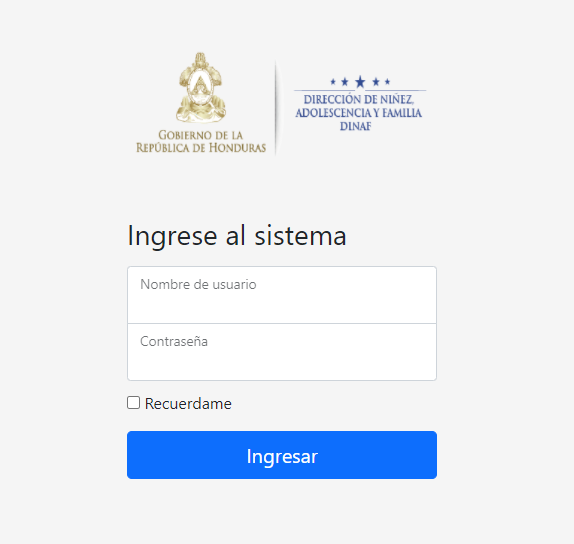
* Node JS -> Framework Express

Front End

* JavaScript -> Framework React JS

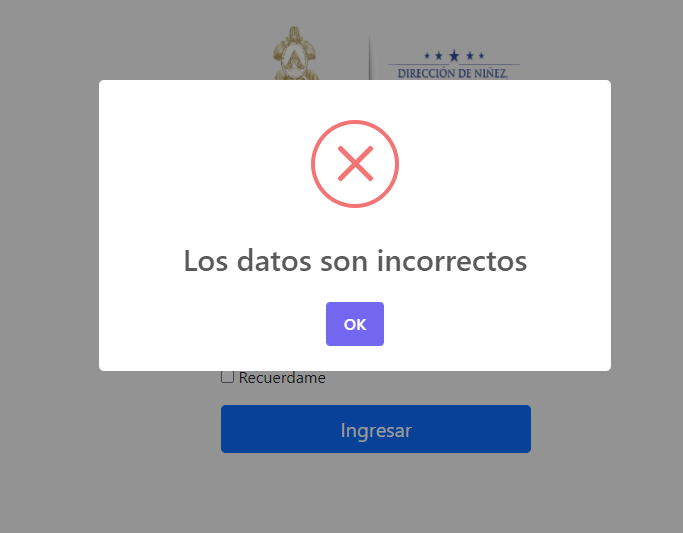
# Pantallas diseñadas

Login



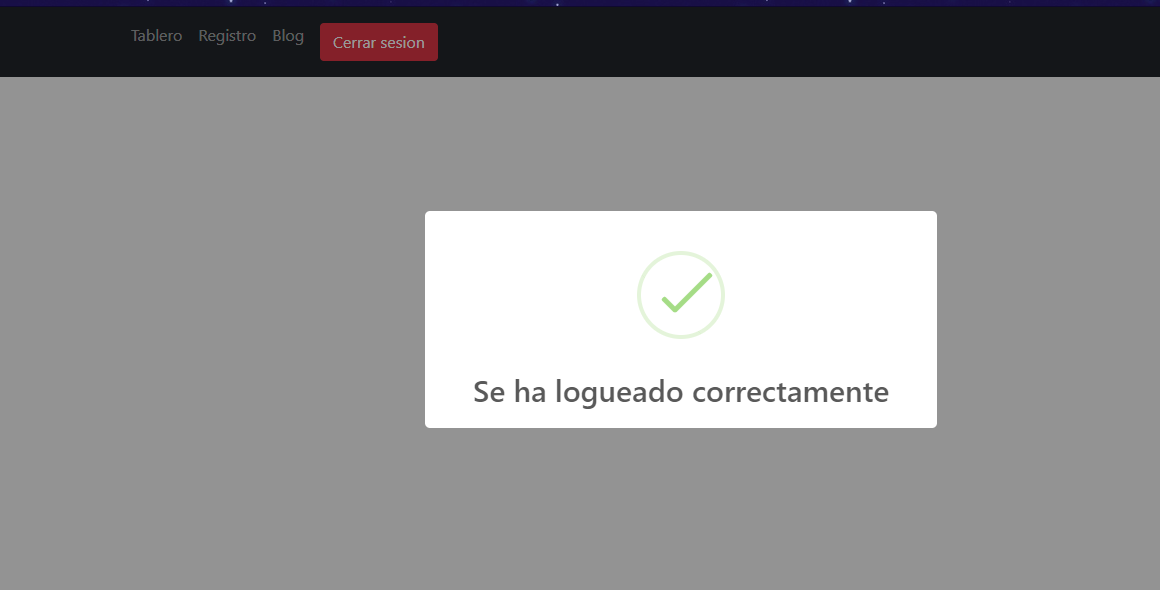
Login Incorrecto

Si el usuario no ingresa el usuario correcto y la contraseña correcta, entonces le mostrará la siguiente pantalla



Login correcto

Si el usuario ingreso su usuario valido y contraseña correcta le mostrará la siguiente pantalla



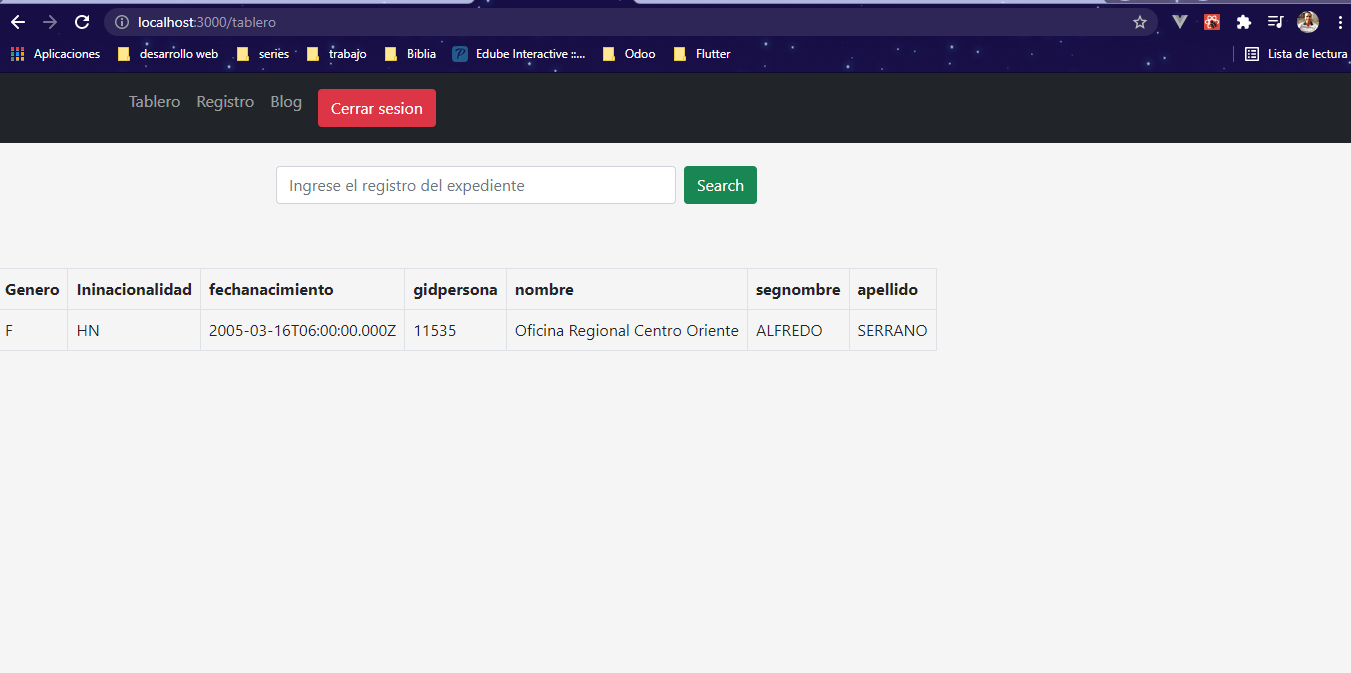
Pantalla de Registro

En este caso este formulario sirve para guardar personas, es reutilizable eso significa que si en el sistema se quieren guardar otro tipo de información, puede tomarse como base este formulario

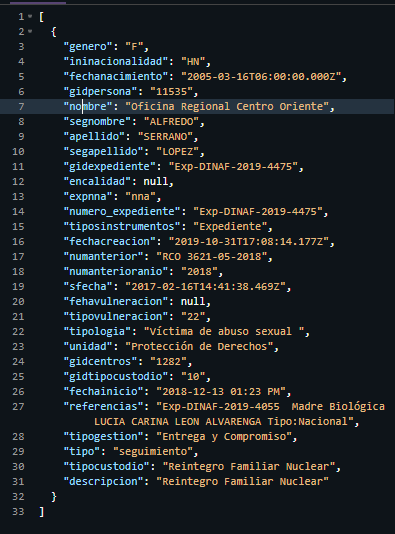


Pantalla de búsqueda de expedientes

Esta es la pantalla que se muestra al momento de loguearse, el objetivo es que al utilizar el buscador que sale en esa pantalla, busque y muestre en pantalla información con respecto a ese expediente, expediente que es de un nna, en ese caso ahí los muestra en una tabla, la tabla solamente es un ejemplo de cómo se podría visualizar la información, aún no está del todo terminado, porque si trae la información del expediente pero no la trae al momento de usar el buscador, eso es porque en la ruta que busca el expediente con la información aun no funciona, pero si el expediente se deja explícitamente en la ruta ahí sí funciona, esto creo que es algo complejo de entender así que cualquier duda estoy a la disposición de explicarles.



Los datos relacionados al expediente que podrían mostrarse en pantalla, pueden ser que no sea necesario que se muestren todos los datos, en todo caso los datos serian los que se ven en el siguiente JSON, JSON es un objeto de JavaScript



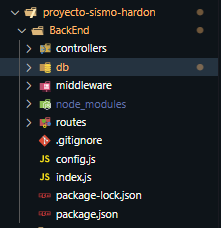
Estructura de carpetas y directorios del proyecto

El proyecto se ha dividido en dos carpetas una que se llama Back End y otra que se llama Front End



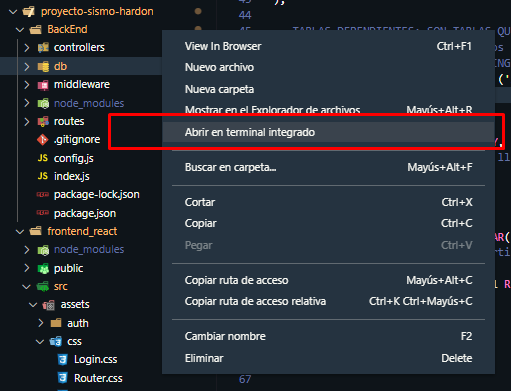
Vamos a empezar a explicar la estructura como se encuentra la carpeta Back End

# Back End

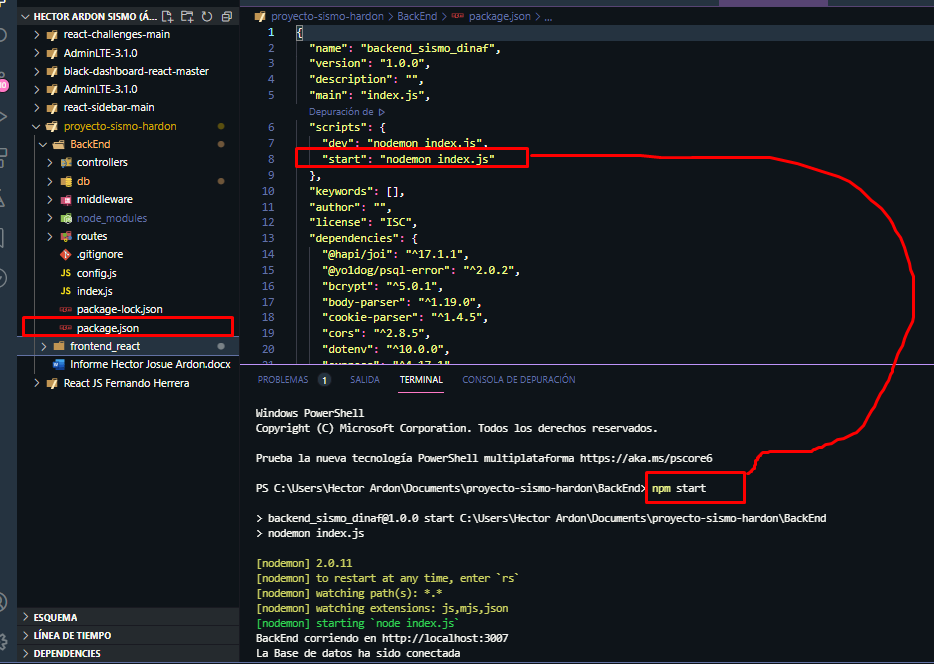


El Back End es lo que interactua con la base de datos de Sismo, para iniciar la ejecución del Back End, dar click derecho sobre la carpeta Back End y dar click donde

dice Abrir en terminal integrado



Una vez que abrimos la carpeta Back End con la terminal de Visual Studio Code, escribir el comando **npm start.** Para arrancar la ejecución del Back End, tener en cuenta que debe estar abierto el servidor de Postgre SQL en este caso con la herramienta PG Admin para que pueda funcionar

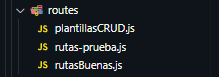


Al ejecutar el comando **npm start** lo que hará será ejecutar la palabra que dice start en el archivo package.json, esto es si está en modo de desarrollo, una vez que el proyecto este terminado y listo para subirse a la nube, se utilizará el comando **npm run build**, que minificará el proyecto con el objetivo de que pese lo menos posible para hacer una aplicación eficaz y rápida. Donde dice **“start”: nodemon index.js**, lo que esto hace es arrancar la ejecución del Back End con el archivo index.js

**Archivo index.js**: En este archivo se encuentra la instalación del framework express, este archivo se encarga de llamar a los routers que interactúan con la base de datos, esto de los routers se encuentra en la carpeta routes.

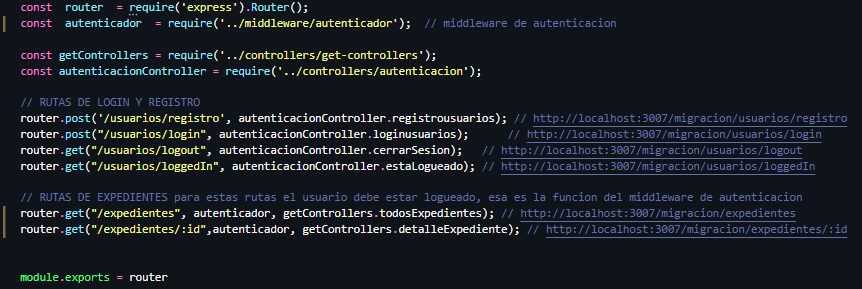
**Archivo config.js:** Este archivo contiene las credenciales que indican a la base de datos que se va a conectar, también la url que se va a conectar, por ahora se encuentra en localhost, cuando la aplicación este terminada se subirá a la nube y ya no será localhost el servidor en el que se encuentre alojada.

**Carpeta routes**: Esta carpeta se encuentra estructurada de la siguiente forma, es una carpeta muy importante y la explicaré un poco más



Estos archivos cumplen ciertas funciones comenzaré con el que dice plantillasCRUD

en este archivo se encuentran tal como dice su nombre plantillas, eso significa que si se quiere hacer una operación como guardar, obtener, editar o eliminar datos de la base de datos, entonces se toma la plantilla de ahí y lo que básicamente se modifica seria la consulta a la base de datos, se toma la plantilla y se le hacen pruebas en el archivo rutas-prueba.js, para que cuando este buena, entonces moverla a donde dice rutasBuenas, quiero compartir ahora el archivo rutasBuenas.js ya que lo que hay ahí son funcionalidades que ya están buenas en términos de Back End, hay algunas que se han logrado integrar con el Front End y hay otras que no, mostraré la imagen de las funcionalidades correctas hasta el día de hoy.



Tal como se ve en la imagen las funcionalidades que están bien son

* registro
* login
* logout (Cerrar sesión)
* loggedin (Verifica si el usuario está logueado o no)
* expedientes: Retorna todos los expedientes que hay en la base de datos
* expedientes/:id: Retorna un expediente en especifico, el que se le indica donde dice: id

La funcionalidad de expedientes/:id es algo que tengo pendiente integrarlo al Front End, porque debo estudiar de algo que se llama Redux para que funcione bien y estoy en ese proceso, Redux es para manejar el estado dinámicamente, eso significa que si el expediente cambia en el campo de búsqueda, en la pantalla debe renderizar (mostrar) los datos de ese expediente, es algo que cambia conforme al expediente que se ingrese y el estado como tal es equivalente al expediente.

Carpeta controllers

La carpeta controllers contiene las funciones que van a ejecutar cada una de las rutas del Back End, las funciones contienen consultas de tipo SELECT, INSERT, UPDATE o DELETE

Carpeta middleware

La carpeta middleware solo tiene un archivo, este archivo es para hacer un filtro entre las rutas publicas y privadas, es para darle seguridad a la base de datos, de tal forma que la información que es privada para los usuarios al estar logueados, otros usuarios no la puedan ver.

Carpeta db

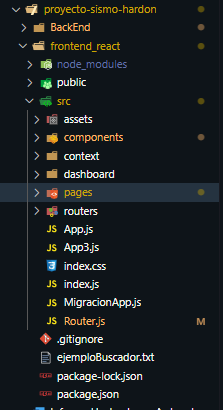
La carpeta db está para almacenar las tablas de la base de datos adicionales que vaya haciendo, la tengo con un archivo que dice sismo2.sql que es precisamente para eso, y el archivo que dice ensayos.js, son pruebas que he hecho al momento de insertar datos.

Carpeta node\_modules

La carpeta node\_modules son todos los modulos o dependencias que se han instalado en el proyecto, se puede eliminar esta carpeta y volverse a restablecer con el comando npm install, que instalara los módulos que se indican en el archivo package.json

Ya con esto explico las funcionalidades necesarias como dato importante el archivo packcage-lock.json no tocarlo para nada

# Front End



El Front End también se debe ejecutar con el comando npm start en la terminal de visual studio code, el cmd o la terminal de git bash pero yo recomiendo usar la terminal del visual studio code para mayor facilidad.

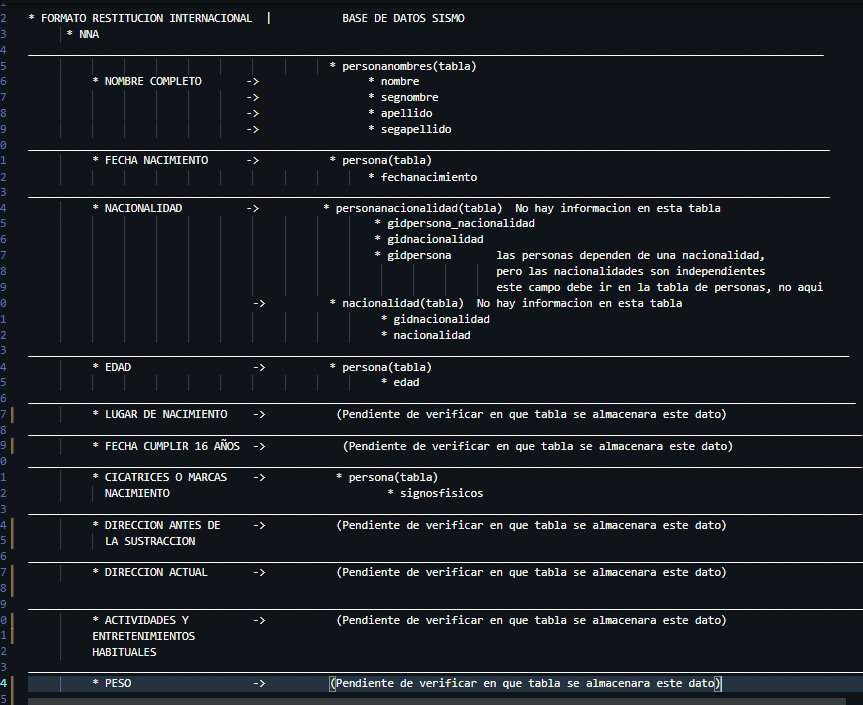
Cuando se arranca la ejecución del Front End, tiene que estar corriendo también el Back End simultáneamente para que la aplicación pueda funcionar, el Front End arranca la ejecución con el archivo index.js

Hasta ahora estoy trabajando con el archivo App, aquí lo que quiero indicar es la ruta que el Front End va a tomar para ejecutarse, de esa forma no arruino las funcionalidades que están bien programadas de hecho los archivos App3 y MigracionApp pueden escribirse aquí y harian funcionalidades totalmente diferentes,

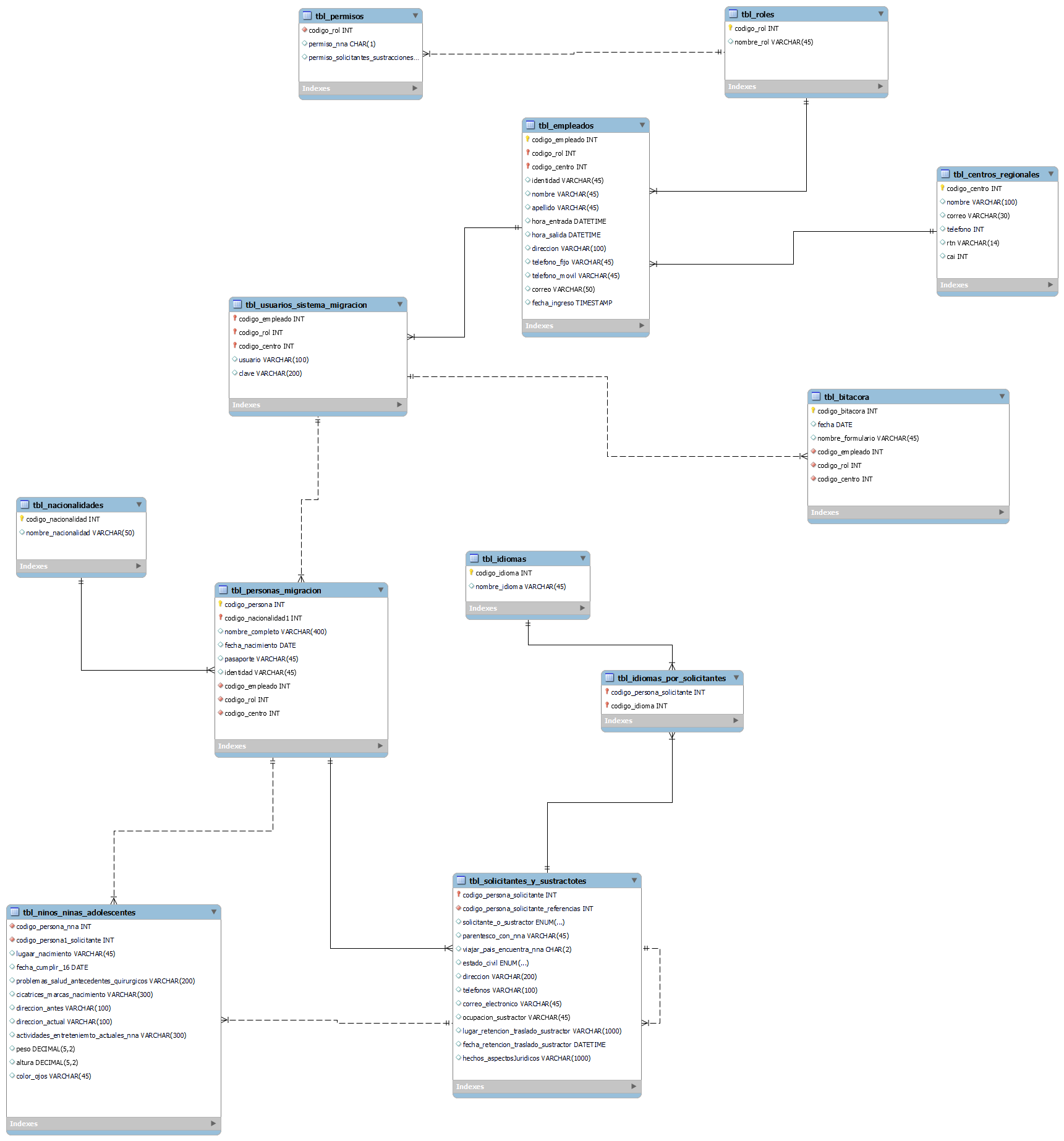
La carpeta components contiene trozos de código JSX que se van a renderizar o mostrar en las paginas que precisamente la carpeta pages es para almacenar las paginas a mostrar.

# Cuadro comparativo entre datos de restitución internacional y base de datos sismo

Con este cuadro quiero dar a entender que en la columna de la izquierda son los datos que hay en el formulario de restitución internacional en papel, y en la columna de la derecha son donde podrían guardarse los datos en la base de datos de sismo, hay que saber donde se van a guardar los datos para plantear la consulta de tipo INSERT INTO la cual ejecutara el formulario al momento de enviar los datos.



# Propuesta modelo relacional Restitución Internacional



# Glosario de términos

Node JS: Node JS es la tecnología que nos permite utilizar el lenguaje JavaScript para conectar con la base de datos, Node JS como tal se vuelve demasiado complejo de implementar por esa razón se utilizan los frameworks para reducir el nivel de complejidad, el más popular es el framework express.

**Framework:** Framework es un conjunto de código que la gente ha reescrito de una forma diferente con el objetivo de facilitar un poco más la manipulación de lenguajes de programación o lenguajes para estilizar

Front End: Es la parte visual de la aplicación, formularios, paginas, todo lo que el usuario puede ver.

**Back End**; Es la parte funcional de la aplicación sin el Back End, todos los formularios no harian nada, es lo que se conecta con la base de datos y que le da vida a la aplicación.

# Bibliografía

https://www.youtube.com/watch?v=hRlApOXoVSs&list=RDCMUCXR7VjA26PcHP3vb6F2X3VQ&start\_radio=1&rv=hRlApOXoVSs&t=41

https://www.youtube.com/watch?v=JKenEw4S5BQ&t=1048s

https://www.youtube.com/watch?v=7ioabL1OAyM

https://github.com/jgbijlsma/mern-auth-template-tutorial

https://github.com/Borja95/reactLoginFakeApi

https://www.digitalocean.com/community/tutorials/how-to-add-login-authentication-to-react-applications

https://www.youtube.com/watch?v=oDQAzTJrbSI&t=127s

https://github.com/luismcabrera/react-router-dom/tree/main/src/pages

https://guias.makeitreal.camp/react/peticiones-http-con-axios

https://attachments.convertkitcdnm.com/211885/3ed242dc-1e14-4448-b978-d57aed9c1cb0/The\_React\_\_Axios\_Cheatsheet.pdf

https://rsuitejs.com/components/sidenav/

https://github.com/jonmircha/youtube-react

https://attachments.convertkitcdnm.com/211885/3ed242dc-1e14-4448-b978-d57aed9c1cb0/The\_React\_\_Axios\_Cheatsheet.pdf

https://medium.com/@simonhoyos/enrutando-en-react-cd9e4ad6e3d3

https://reactrouter.com/web/example/sidebar

https://www.salvia-kit.com/techno/react

https://bluuweb.github.io/node/07-jwt/#validaciones-hapi-joi

https://github.com/carlosazaustre/node-api-rest-2017

https://www.tabnine.com/code/javascript/functions/bcryptjs/compareSync