Ministerio del Poder Popular para la Educación Superior

Universidad Católica Andrés Bello - Extensión Guayana

Facultad de Ingeniería - Ingeniería Informática

**Libreta de Salud del niño - Manual de Sistema**

|  |  |
| --- | --- |
| Realizado por | Sotillo Vallejo, César Enrique  Naim Arcoiza, Carmelo Jesús |
| Profesora: | Medina Cuida, Luz Esperanza |
| Fecha | Abril, 2023 |

Índice

[**Estructura principal Frontend.** 3](#_Toc151563663)

[**Componentes**. 4](#_Toc151563664)

[Tabla Componentes. 4](#_Toc151563665)

[**Hooks.** 5](#_Toc151563666)

[Tabla Hooks. 5](#_Toc151563667)

[**Images.** 6](#_Toc151563668)

[Tabla Images. 6](#_Toc151563669)

[**Pages.** 7](#_Toc151563670)

[Tabla babyList. 7](#_Toc151563671)

[Tabla babyProfile. 8](#_Toc151563672)

[Tabla KidBirthRecord. 8](#_Toc151563673)

[Tabla login. 9](#_Toc151563674)

[Tabla singup. 9](#_Toc151563675)

[**Theme.** 10](#_Toc151563676)

[**App.js.** 11](#_Toc151563677)

[**Ejecución del sistema.** 11](#_Toc151563678)

[Recomendación para IDE Setup 11](#_Toc151563679)

[Setup del proyecto 11](#_Toc151563680)

[Compilar y preparar para desarrollo 11](#_Toc151563681)

[Compilar para producción 11](#_Toc151563682)

[**Estructura principal Backend.** 12](#_Toc151563683)

[**Checker**. 13](#_Toc151563684)

[Tabla Checker. 13](#_Toc151563685)

[**Config.** 13](#_Toc151563686)

[Tabla Config. 13](#_Toc151563687)

[**Controllers**. 14](#_Toc151563688)

[Tabla Config. 14](#_Toc151563689)

[**Db.** 15](#_Toc151563690)

[Tabla Db. 15](#_Toc151563691)

[**Middlewares**. 16](#_Toc151563692)

[Tabla Middlewares. 16](#_Toc151563693)

[**Routes**. 16](#_Toc151563694)

[Tabla Routes. 17](#_Toc151563695)

[**Schemas.** 17](#_Toc151563696)

[Tabla Schemas. 18](#_Toc151563697)

[**Ejecución del sistema.** 18](#_Toc151563698)

[Recomendación para IDE Setup 18](#_Toc151563699)

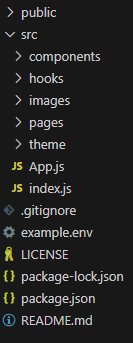
[Setup del proyecto 18](#_Toc151563700)

[Compilar y preparar para desarrollo 18](#_Toc151563701)

[Compilar para producción 18](#_Toc151563702)

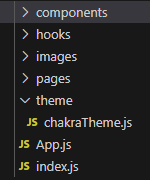
## Estructura principal Frontend.

Estructura de los archivos del front-end. Los archivos que se explicarán a continuación son los que están dentro de la carpeta “SRC” ( Source ).



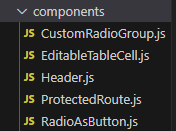
Las carpetas y archivos de la carpeta “SRC” son:

* components
* hooks
* images
* pages
* theme
* App.js
* index.js



## Componentes.

La carpeta de “components” es una carpeta importante en Javascript. En ella hay componentes “.js” que se crean para el sistema. Los archivos “.js” son archivos que contienen HTML y JavaScript que está relacionado con una determinada parte de la página, como podría ser el formulario de crear un nuevo progenitor y los progenitores en las listas de los enfermeros, por ejemplo.



### Tabla Componentes.

|  |  |
| --- | --- |
| CustomRadioGroup.js | Componente de React que implementa un grupo de radio personalizado utilizando Chakra UI y React Hook Form |
| EditableTableCell.js | Componente de React que representa una celda editable en una tabla. La celda puede contener diferentes tipos de entradas dependiendo del tipo de datos que se esté editando |
| Header.js | Componente de React que representa la barra de encabezado de una aplicación. |
| ProtectedRoute.js | Archivo que define un componente de ruta protegida (ProtectedRoute) que actúa como un contenedor para páginas que requieren autenticación. |
| RadioAsButton.js | Archivo que define un componente llamado RadioAsButton que utiliza Chakra UI para renderizar un botón que actúa como una opción de radio. |

## **Hooks.**

La carpeta “Hooks”, Los Hooks son funciones que te permiten “enganchar” el estado de React y el ciclo de vida desde componentes de función. Los hooks no funcionan dentro de las clases, nos permiten usar React sin clases.



### Tabla Hooks.

|  |  |
| --- | --- |
| useLocalStorage.js | Este hook, proporciona una manera conveniente de interactuar con el almacenamiento local del navegador (localStorage). Permite almacenar y recuperar valores desde el almacenamiento local mientras mantiene el estado actualizado en React. |

## **Images.**

La carpeta “Images”, capeta donde van a estar las imágenes paea decorar las pantallas del sistema.

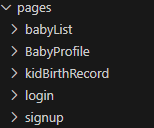
****

### Tabla Images.

|  |  |
| --- | --- |
| login-background | Imagen que va a estar localizada en la página del login |
| signup-background | Imagen que va a estar localizada en la página del crear cuenta |

## **Pages.**

La carpeta “Pages”, capeta que indica donde van a estar la generación de las páginas, con sus respectivos controladores y rutas.

****

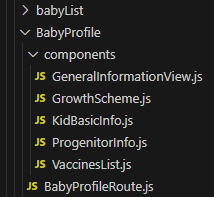
La carpeta babyList contiene la página donde se va a mostrar la lista de bebes que estan en el sistema y con sus respectivas rutas.



### Tabla babyList.

|  |  |
| --- | --- |
| BabyListRoute.js | Representa una lista de niños |

La carpeta BabyProfile, contiene dentro una carpeta llamada “componentes” donde están las componentes que devuelven funciones utilizando Javascript para realizar acciones sobre la aplicaciones en el Perfil de los bebes. Esta carpeta se va a encargar de representar el perfil de un bebé.



### Tabla babyProfile.

|  |  |
| --- | --- |
| GeneralInformationView.js | Componente que representa una pestaña de información general en un perfil de niño |
| GrowthScheme.js | Componente que representa una interfaz de usuario para gestionar y visualizar registros de crecimiento (como evaluaciones médicas) relacionados con el desarrollo de un niño. |
| KidBasicInfo.js | Componente que representa la información básica de un niño |
| ProgenitorInfo.js | Componente que representa la información básica de un progenitor |
| VaccineList.js | Componente que muestra una lista de vacunas y permite editar la información relacionada con la aplicación de esas vacunas |
| BabyProfileRoute.js | Representa la página de perfil de un niño |

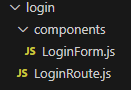
La Carpeta kadBirthRecord, contiene dentro una carpeta llamada “componentes” donde están las componentes que devuelven funciones utilizando Javascript para realizar acciones sobre la aplicaciones en libreta de salud de los bebes. Esta carpeta se va a encargar de representar el la libreta de salud de un bebé.

****

### Tabla KidBirthRecord.

|  |  |
| --- | --- |
| BirthRecordForm.js | Componente que se encarga de la generación de un formulario que será llenado para el registro de libreta de salud de un niño |
| BirthRecorRoute.js | componente que se utiliza para visualizar y editar los registros de nacimiento de niños |

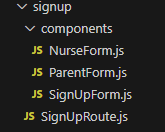
La carpeta Login, contiene dentro una carpeta llamada “componentes” donde están las componentes que devuelven funciones utilizando Javascript para realizar acciones sobre la aplicacion en el login de los usuarios. Esta carpeta se va a encargar de representar el login de los usuarios.



### Tabla login.

|  |  |
| --- | --- |
| LoginForm.js | Componente que representa un formulario de inicio de sesión en una aplicación web. |
| LoginRoute.js | Ruta que representa la página de inicio de sesión en una aplicación web. |

La carpeta Singup, contiene dentro una carpeta llamada “componentes” donde están las componentes que devuelven funciones utilizando Javascript para realizar acciones sobre la aplicacion en el crear nueva cuenta de los usuarios. Esta carpeta se va a encargar de representar el crear nueva cuenta de los usuarios.



### Tabla singup.

|  |  |
| --- | --- |
| NurseForm.js | Componente es un formulario que recoge la información relacionada con el código del centro para una enfermera. |
| ParentForm.js | Componente que representa un formulario que recoge información relacionada con un padre o progenitor. |
| SignUpForm.js | Componente representa un formulario de registro de usuario. |
| SignUpRoute.js | Ruta que representa la página de registro. |

## **Theme.**

Esta carpeta contiene un archivo Javascript. Este archivo exporta un tema personalizado para la biblioteca de componentes de interfaz de usuario Chakra UI en una aplicación de React.



|  |  |
| --- | --- |
| chakraTheme.js | Archivo define un tema personalizado que puede ser importado y utilizado en una aplicación Chakra UI para proporcionar un estilo visual coherente y personalizado en toda la interfaz de usuario. |

## **App.js.**

Este archivo es la entrada principal de la aplicación React, configura las rutas y proporciona el contexto global de la aplicación, como el tema y la configuración regional. Además, maneja la lógica de redirección inicial basada en la autenticación del usuario.



## **Ejecución del sistema.**

### Recomendación para IDE Setup

[VSCode](https://code.visualstudio.com/)

### Setup del proyecto

~ npm install

### Compilar y preparar para desarrollo

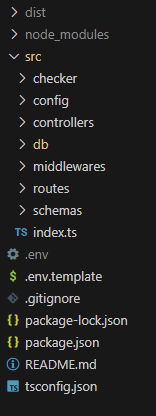
~ npm run dev

### Compilar para producción

~npm run build

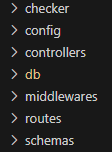
## **Estructura principal Backend.**

Estructura de los archivos del back-end. Los archivos que se explicarán a continuación son los que están dentro de la carpeta “SRC” ( Source ).

****

Las carpetas y archivos de la carpeta “SRC” son:

* Checker
* config
* controllers
* db
* middlewares
* routes
* schemas
* index.js



## Checker.

Esta carpeta contiene un archivo Typescript llamado Checker. Este archivo es el encargado de enviar notificaciones y realiza la configuración inicial para enviar mensajes de texto.



### Tabla Checker.

|  |  |
| --- | --- |
| Checker.ts | Utiliza las bibliotecas node-cron y nodemailer para realizar una tarea programada (cron job) y enviar correos electrónicos con Node.js. |

## Config.

Esta carpeta contiene un archivo Typescript llamado Database. Este archivo es el encargado de realizar las conexiones con la base de datos.

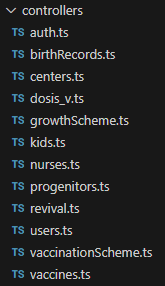
****

### Tabla Config.

|  |  |
| --- | --- |
| Database.ts | Configura y exporta un objeto Pool de PostgreSQL utilizando las credenciales y la configuración de conexión proporcionadas a través de las variables de entorno, lo que facilita la conexión y ejecución de consultas a una base de datos PostgreSQL desde otras partes del código. |

## Controllers.

**L**a carpeta controllers. Carpeta donde se organizan y almacenan los controladores de la aplicación, que son componentes clave para gestionar la lógica y el flujo de datos en la aplicación. La estructura específica puede variar según las preferencias del desarrollador y la arquitectura de la aplicación.

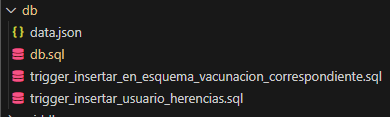
****

### Tabla Config.

|  |  |
| --- | --- |
| Auth.ts | Este controlador se encarga de manejar la lógica asociada al proceso de inicio de sesión (signin) en la aplicación, PostgreSQL como base de datos, y está configurada para utilizar bcrypt para el almacenamiento seguro de contraseñas y jsonwebtoken (JWT) para la generación de tokens de autenticación. |
| birthRecords.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada registros\_nacimiento |
| centers.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada centros |
| dosis\_v.ts | Este controlador maneja la obtención de dosis de vacunaciones según ciertos criterios, principalmente la edad óptima de aplicación. |
| growthScheme.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada esquemas\_control\_crecimiento |
| kids.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada ninos |
| nurses.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada efermeros |
| Progenitors.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada progenitores |
| revivals.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada reanimaciones |
| Users.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada usuarios |
| vaccinationScheme.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada esquemas\_vacunacion |
| Vaccines.ts | Este controlador maneja operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una entidad llamada vacunaciones |

## Db.

La carpeta llamada “db”. en ella se van a contener todos los archivos “.sql” necesarios para la la estructuracion y creacion de las tablas de la base de datos, asimismo como los triggers.



### Tabla Db.

|  |  |
| --- | --- |
| Data.json | Archivo donde se guardan los json a utilizar para la pruebas en la base de datos |
| Db.sql | Archivo que contiene las tablas de la base de datos |
| trigger\_insertar\_en\_esquema\_vacunacion\_correspondiente.sql | Trigger donde al crear un esquema de vacunaciones dependiendo su clasificación inserta su clave primaria en su tabla correspondiente |
| trigger\_insertar\_usuario\_herencias.sql | Trigger donde se insertan la clave primaria del usuario si es un enfermero en la tabla enfermeros y si es un progenitor en la tabla progenitores |

## Middlewares.

Esta es la carpeta Middlewares. Esta carpeta es la encargada de proteger rutas específicas, asegurándose de que solo usuarios autenticados y autorizados (según sus roles) tengan acceso a ciertos recursos o acciones.

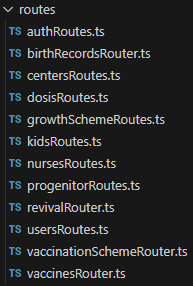


### Tabla Middlewares.

|  |  |
| --- | --- |
| authMiddleware.ts | programa es un middleware en TypeScript que verifica la autenticación y autorización de un token JWT (JSON Web Token) |

## Routes.

Esta es la carpeta Routes. Esta capeta contiene los archivos que definen las rutas y los controladores asociados para cada recurso o funcionalidad de la aplicación. Las rutas son las URL que los clientes (navegadores web, aplicaciones móviles, etc.) utilizan para acceder a recursos específicos en la aplicación. La estructura y organización de esta carpeta dependen de las preferencias del desarrollador y del diseño de la aplicación, pero a menudo sigue algunos patrones comunes.



### Tabla Routes.

|  |  |
| --- | --- |
| authRoutes.ts | Esta ruta define las operaciones en el inicio de sesión |
| birthRecordsRouter.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "registro\_nacimientos" |
| centersRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "centros" |
| dosisRoutes.ts | Este ruta maneja la obtención de dosis de vacunaciones |
| growthSchemeRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "Esquema\_crecimiento" |
| kidsRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "ninos" |
| nursesRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "enfermeros" |
| progenitorRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "Progenitores" |
| revivalRouter.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "Reanimaciones" |
| usersRoutes.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "Usuarios" |
| vaccinationSchemeRouter.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "Esquema\_vacunacion" |
| vaccinesRouter.ts | Esta ruta define las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para el recurso "vacunas" |

## Schemas.

La carpeta “Schemas”. dentro de esta carpeta hay un archivo que se llama userDecodedData.ts, que hace uso de la librería “zod” para validar y estructurar los datos decodificados de un usuario. La definición del esquema se exporta para su uso en otras partes del código, y se proporciona un tipo TypeScript asociado para representar el tipo de datos que cumplen con este esquema.

### Tabla Schemas.

|  |  |
| --- | --- |
| userDecodedData.ts | Archivo que utiliza la biblioteca “zod” para definir un esquema (schema) para los datos decodificados de un usuario |

## **Ejecución del sistema.**

### Recomendación para IDE Setup

[VSCode](https://code.visualstudio.com/)

### Setup del proyecto

~ npm install

### Compilar y preparar para desarrollo

~ npm run dev

### Compilar para producción

~npm run build