CENTENARIO PEREIRA

Manual Técnico

Versión 1.0

2023 - 2024

ÍNDICE

INTRODUCCIÓN

La Institución Educativa Centenario de Pereira enfrenta la urgente necesidad de establecer un medio de comunicación oficial y fácil de usar.

La falta de este canal dificulta la transparencia, la planificación y la participación de la comunidad educativa, generando desinformación y rumores o desinformación.

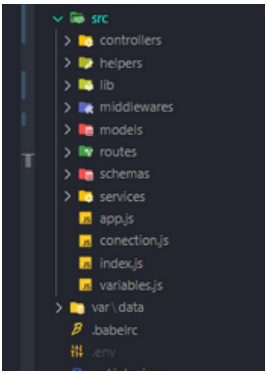
La implementación de un medio de comunicación oficial es esencial para combatir la desinformación, mejorar el acceso a recursos educativos y fortalecer la comunidad educativa en general.

Los objetivos de este proyecto incluyen establecer un medio de comunicación oficial para distribuir información de manera efectiva, facilitar la comunicación entre directivos y la comunidad, anunciar actividades de manera oportuna y proporcionar una búsqueda fácil y confiable de información para todos los interesados en la comunidad educativa.

ARQUITECTURA DEL SISTEMA

BACK-END

Arquitectura del Software (Estructura de Carpetas) se esta utilizando el modelo ***MVC Extendida***



Capa raíz:

- app.js: Archivo principal del proyecto que ejecuta el servidor y configura las rutas.

Carpetas principales:

- js: Contiene los archivos JavaScript del proyecto.
- lib: Contiene librerías externas o personalizadas utilizadas en el proyecto.
- public: Contiene archivos estáticos como imágenes, CSS y JavaScript que se sirven al cliente.

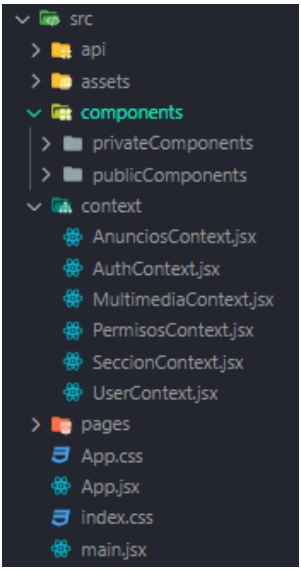
Subcarpetas dentro de "js":

- controllers: Contiene los controladores que manejan las solicitudes HTTP.
- helpers: Contiene funciones auxiliares para el proyecto.
- middlewares: Contiene middleware para procesar solicitudes HTTP antes de llegar a los controladores.
- models: Contiene los modelos de datos del proyecto.
- routes: Contiene las rutas para las solicitudes HTTP.
- services: Contiene los servicios que encapsulan la lógica de negocio del proyecto.
- upload: Contiene funciones para la carga de archivos.

ARQUITECTURA DEL SISTEMA

FRONT-END

Arquitectura del Software (Estructura de Carpetas)



Capa raíz:

- main.jsx: Archivo Principal del proyecto que ejecuta todo el proyecto

Subcarpetas dentro de "src":

- **api:** Contiene los controladores que nos ayudan a conectar nuestro código back-end y de esta forma traer los datos de la base de datos.
- **assets:** Contiene archivos públicos que son usados de forma estática dentro del proyecto, como imágenes, data con json, etc.
- **components:** Contiene los componentes que vamos a estar utilizando a lo largo del proyecto.
- **context:** Contiene los contextos y proveedores para gestionar la autenticación de usuarios y la información relacionada.
- pages: Contiene las paginas que vamos a utilizar.

Subcarpetas dentro de “Components”:

- **publicComponents:** Contiene los componentes públicos que se van a utilizar en las paginas
- **privateComponents:** Contiene los componentes privados que solo seran usados en la pagina “AdminDashboard”

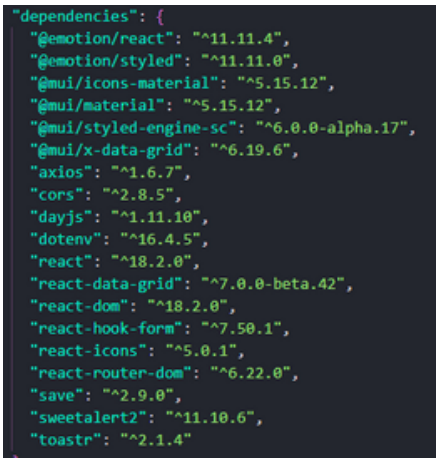
TECNOLOGIAS UTILIZADAS

FRONT-END

Como base se esta utilizando React + Vite en sus versiones más actuales

React: 18.2.0
Vite: 5.1.0

Dependencias utilizadas



```
"dependencies": {
  "@emotion/react": "^11.11.4",
  "@emotion/styled": "^11.11.0",
  "@mui/icons-material": "^5.15.12",
  "@mui/material": "^5.15.12",
  "@mui/styled-engine-sc": "^6.0.0-alpha.17",
  "@mui/x-data-grid": "^6.19.6",
  "axios": "^1.6.7",
  "cors": "^2.8.5",
  "dayjs": "^1.11.10",
  "dotenv": "^16.4.5",
  "react": "^18.2.0",
  "react-data-grid": "^7.0.0-beta.42",
  "react-dom": "^18.2.0",
  "react-hook-form": "^7.50.1",
  "react-icons": "^5.0.1",
  "react-router-dom": "^6.22.0",
  "save": "^2.9.0",
  "sweetalert2": "^11.10.6",
  "toastr": "^2.1.4"
```

Se esta utilizando PostgreSQL con el entorno de trabajo PgAdmin4

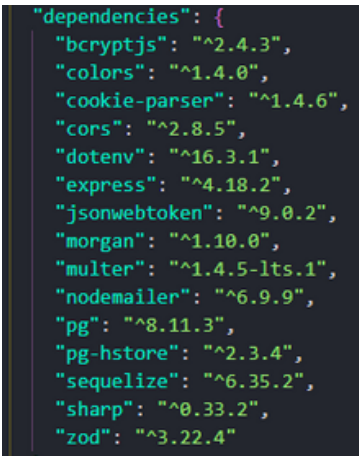
Versión 16

BACK-END

Como base se esta utilizando NodeJS + Express en sus versiones mas actuales

NodeJs 20.10.0
Express: 4.18.2

Dependencias utilizadas



```
"dependencies": {
  "bcryptjs": "^2.4.3",
  "colors": "^1.4.0",
  "cookie-parser": "^1.4.6",
  "cors": "^2.8.5",
  "dotenv": "^16.3.1",
  "express": "^4.18.2",
  "jsonwebtoken": "^9.0.2",
  "morgan": "^1.10.0",
  "multer": "^1.4.5-lts.1",
  "nodemailer": "^6.9.9",
  "pg": "^8.11.3",
  "pg-hstore": "^2.3.4",
  "sequelize": "^6.35.2",
  "sharp": "^0.33.2",
  "zod": "^3.22.4"
```

Dependencias de Desarrollo

Babel
Nodemon

DISEÑO DEL SISTEMA

01

DIAGRAMAS

Link hacia el Drive con los diferentes diagramas

Clic Aquí

02

MOCK UPS

Link hacia el Mock Up

Click Aquí

03

APLICACIONES UTILIZADAS

Diagramas: StarUML

Mock Up: FIGMA

Nota:

En el repositorio se explica que variables de entorno lleva el proyecto para funcionar correctamente.

BASE DE DATOS

Tablas en pgAdmin

PostgreSQL

Para trabajar con PostgreSQL se requiere se recomienda tener de la versión 16 en adelante ya que el proyecto fue construido usando esta misma.

pgAdmin4

pgAdmin4 es un gestor de base de datos de PostgreSQL que es necesario para administrar el instancia o servicio instalador de PostgreSQL en pocas palabras trabajarlo de manera visual.

Recomendaciones

Cuando vayas a correr en servidor de la aplicación asegúrate de contar con los datos necesarios y correctos en las variables de entorno de nuestro proyecto (.env)

Nota:

El archivo no puede tener ningún nombre antes del punto ya que esto genera errores de lectura.