PRUEBA TÉCNICA - API DE COLABORADORES

BACK-END

Dependencias Instaladas

Las siguientes dependencias fueron instaladas para el correcto funcionamiento del proyecto:

- **dotenv**: Permite cargar variables de entorno desde un archivo .env.
- **express**: Framework para construir aplicaciones web y APIs.
- **mysql2**: Cliente para conectar y trabajar con bases de datos MySQL.
- **cors**: Middleware que habilita el intercambio de recursos entre diferentes orígenes.
    - **VERIFICAR**: En caso de haya error desde front-end al tratar de conectare al api, sea para hacer consultar o realizar cambios en el api asegurarse de configurar correctamente CORS, ya que actualmente solo hay 2 puertos los cuales pueden hacer peticiones a la api, (3000, 3001), En mi caso levante en el puerto 3000 el back-end y En el puerto 3001 el front-end
- **express-validator**: Middleware para la validación y sanitización de datos en las peticiones HTTP.

Para instalarlas:

npm install dotenv express mysql2 cors express-validator

Configuración del Proyecto

1. **Archivo de Variables de Entorno (.env)**
2. DB_HOST=localhost
3. DB_USER=tu_usuario
4. DB_PASSWORD=tu_contraseña
5. DB_NAME=TEST
6. PORT=3000

Conexión a la Base de Datos

El proyecto se conecta a MySQL utilizando los datos definidos en el archivo. env. Asegúrate de tener una base de datos configurada y de que la tabla COLABORADOR esté creada.

Ejecución del Proyecto

Para ejecutar el proyecto, utilizar el siguiente comando.

node index.js

FRONT-END

Sistema de Gestión de Colaboradores - Frontend

Dependencias Instaladas

Las siguientes dependencias se han instalado para el correcto funcionamiento del proyecto:

- **React**: Biblioteca principal para construir la interfaz de usuario.
- **Next.js**: Framework de React para renderizado del lado del servidor y generación de rutas.
- **react-hook-form**: Manejo y validación de formularios de forma sencilla y eficiente.
- **axios**: Realización de peticiones HTTP para comunicarse con la API del backend.
- **lucide-react**: Biblioteca de íconos utilizada en la interfaz.
- **UI Components**: Conjunto de componentes UI personalizados (Button, Card, Collapsible, Form, Input, Select, Table, Toaster, Alert, Dialog, etc.) ubicados en @/components/ui.
- **Tailwind CSS**: Utilizado para estilizar la aplicación mediante clases utilitarias.

Para instalarlas, ejecuta:

npm install

Configuración

1. **API Base URL:**
   La variable API_URL se ha definido en el código con el valor http://localhost:3000. Para que funcione primero el bakc-end tiene que estar corriendo, también si se realizan cambios respecto al puerto al que se desea trabajar hay actualiza la variable.

Ejecución del Proyecto

Para iniciar el servidor de desarrollo, ejecutar:

npm run dev

Luego, abre tu navegador en con la dirección e localhost en la que se levanto el proyecto, esta información la observamos en la consola.

Funcionalidades Principales

- **Agregar Colaborador:**
  Permite registrar un nuevo colaborador mediante un formulario con validación en tiempo real usando **react-hook-form** y **zod**.
- **Editar Colaborador:**
  Habilita la actualización de la información de un colaborador existente. Se activa el modo edición al hacer clic en el botón correspondiente.
- **Eliminar Colaborador:**
  Permite la eliminación de un colaborador, mostrando un diálogo de confirmación antes de ejecutar la acción.
- **Nivel de Riesgo:**
  Evalúa y muestra un mensaje de nivel de riesgo en función de la edad ingresada, utilizando diferentes alertas visuales.
- **Carga de Datos:**
  Realiza la consulta a la API para obtener y normalizar los datos de los colaboradores, mostrando la lista en una tabla interactiva.

Estructura del Código

- **Formularios y Validación:**
  Se utiliza react-hook-form junto con zod para gestionar la entrada de datos y aplicar validaciones en tiempo real, garantizando la integridad de la información.
- **Componentes UI:**
  La interfaz se compone de componentes reutilizables (como Button, Card, Collapsible, etc.) que facilitan la creación de una experiencia de usuario consistente y moderna.
- **Comunicación con el Backend:**
  Se utiliza axios para hacer peticiones GET, POST, PUT y DELETE hacia la API de colaboradores, permitiendo la interacción completa con el sistema.
- **Gestión de Estados y Feedback:**
  Se emplea useState para gestionar estados como la carga de datos, el modo de edición, y para controlar la visualización de notificaciones y diálogos mediante el hook useToast.