Manual Técnico - Frontend GradoChain

1. **Descripción General**
   El frontend de GradoChain es una aplicación desarrollada con React, diseñada para proporcionar una interfaz moderna y responsive para gestionar autenticación, proyectos de grado y trazabilidad documental. Utiliza Material UI como librería de componentes para una experiencia de usuario consistente y se integra con una API RESTful backend para realizar operaciones como registro, login, subida de documentos y seguimiento de trazabilidad. Este manual detalla la estructura, instalación, configuración e integración con la API.

2. **Estructura del Proyecto**
   La organización del proyecto está diseñada para facilitar el mantenimiento y la escalabilidad. A continuación, se describen las carpetas y archivos clave:

- src/pages/: Contiene las páginas principales de la aplicación, como:
  - Login.jsx: Página de inicio de sesión.
  - Register.jsx: Página de registro de usuarios.
  - Dashboard.jsx: Panel principal con vistas de proyectos y trazabilidad.
  - DocumentUpload.jsx: Formulario para subir documentos.
- src/components/: Almacena componentes reutilizables, incluyendo:
  - Sidebar.jsx: Barra lateral de navegación.
  - Layout.jsx: Estructura general de la aplicación.
  - Alert.jsx: Notificaciones y mensajes al usuario.
- src/contexts/: Gestiona el estado global y la autenticación mediante contextos de React:
  - AuthContext.jsx: Almacena tokens JWT y estado de usuario.
  - AppContext.jsx: Estado global para datos de proyectos y documentos.
- src/App.jsx: Configura las rutas con react-router-dom y monta la estructura principal de la aplicación.
- public/: Contiene archivos estáticos como imágenes, íconos y el archivo index.html.

3. **Instalación y Ejecución**
   Sigue estos pasos para configurar y ejecutar el frontend localmente:

1. Instala las dependencias:
   a. Ejecuta el siguiente comando en la raíz del proyecto:
      text
      Copiar
      npm install
   Esto instalará React, Material UI y otras dependencias listadas.

2. Ejecuta el proyecto en modo desarrollo:
   a. Usa el comando:
      text
      Copiar

npm run dev

La aplicación estará disponible en http://localhost:5173 (el puerto puede variar según la configuración de Vite; verifica la consola para el puerto exacto).

3. Acceso:
    - Abre un navegador y navega a la URL indicada (por defecto, http://localhost:5173).

4. **Integración con la API**

   El frontend se comunica con una API RESTful backend (probablemente en http://localhost:5000/api) para realizar las siguientes operaciones:

- Autenticación:
    - /api/auth/login: Inicia sesión y devuelve tokens JWT.
    - /api/auth/register: Registra nuevos usuarios.
    - Tokens JWT se almacenan en AuthContext y se envían en el header Authorization: Bearer <token> para rutas protegidas.
- Gestión de Documentos:
    - /api/upload_document: Sube archivos y registra eventos de trazabilidad.
    - Usa FormData para enviar archivos al servidor.
- Trazabilidad:
    - /api/trazabilidad/documento/<documento_id>: Consulta el historial de un documento.
    - /api/trazabilidad/documento/<documento_id>/evento: Registra nuevos eventos.
- Streams de MultiChain:
    - /api/streams: Lista los streams disponibles.
    - /api/publish/<stream_name>: Publica datos en un stream, incluyendo generación de QR.
- Consideraciones:
    - Configura la URL base de la API en una variable de entorno (por ejemplo, .env) como VITE_API_URL=http://localhost:5000/api.
    - Maneja errores HTTP (401, 403, 500) con mensajes al usuario.

5. **Personalización**

   El frontend es altamente personalizable:

- Dashboards y Formularios:
    - Edita los archivos en src/pages/ para modificar vistas como el dashboard o formularios de subida.
    - Usa componentes de src/components/ para mantener consistencia.
- Agregar Nuevas Funcionalidades:
    - Crea un nuevo archivo en src/pages/ (ej. NewPage.jsx).
    - Añade la ruta en src/App.jsx usando react-router-dom: jsx
      Copiar
      <Route path="/new-page" element={<NewPage />} />
- Roles y Vistas:

- Implementa lógica de roles en AuthContext para mostrar vistas según permisos (ej. director, vicerrectoría).

6. **Dependencias Principales**

El proyecto depende de las siguientes librerías:

- React: Biblioteca principal para construir la interfaz.
- React-router-dom: Gestión de rutas en el frontend.
- @mui/material: Componentes de interfaz de Material Design.
- @mui/icons-material: Íconos para la interfaz.

7. **Buenas Prácticas**

- Mantenimiento: Documenta cambios en el código y usa comentarios claros.
- Pruebas: Implementa pruebas unitarias con @testing-library/react si es posible.
- Seguridad: Nunca almacenes tokens JWT en localStorage sin cifrado; considera sessionStorage o cookies HttpOnly.

8. **Solución de Problemas**

- Error 401/403: Verifica el token JWT y renueva con /api/auth/refresh.
- Carga Lenta: Asegúrate de que la API responda correctamente y optimiza las peticiones.
- Puertos Conflictivos: Cambia el puerto en vite.config.js si hay conflictos (ej. server: { port: 5174 }).