Proyecto: Generador y Almacenador de PDF de Gastos Comunes - Comunidad Habitacional

---

Objetivo del Proyecto

Desarrollar una aplicación **fullstack** (backend + frontend) capaz de:

- Recibir información de gastos comunes de distintas casas mediante una API.
- Generar archivos PDF con el detalle de los gastos.
- Almacenar los PDFs organizados en carpetas por mes y casa.
- Mantener un historial trazable de todos los documentos generados.
- Permitir al administrador **visualizar y navegar** por los PDFs generados a través del sitio web.
- Incorporar **autenticación de usuarios** para acceso seguro.

El sistema debe ser escalable, organizado y preparado para futuras ampliaciones.

Tecnologías a Utilizar

Backend:

- **Node.js**: Entorno de ejecución JavaScript para el servidor.
- **Express.js**: Framework minimalista para la creación de la API REST.
- **pdfkit**: Librería para generar archivos PDF de manera programática.
- **fs (File System) de Node.js**: Para creación de carpetas y almacenamiento de archivos.
- **Express-fileupload** (opcional futuro): Para manejo de archivos subidos.
- **Sequelize:** como ORM
- **Mysql2:** como base de datos
- **Cors**
- **Bcrypt: Para encriptar contraseñas**
- **Nodemailer:** Para enviar gastos comunes a los residentes
- **Winston:** Manejador de logs

- **Nodemon**
- **Json Web Token:** para autenticación

Frontend:

- **React.js**: Framework para construir la interfaz web.
- **Vite**: Herramienta de build para React.
- **Tailwind CSS**: Para estilizar rápidamente la aplicación.
- **React-PDF** o **PDF.js**: Librerías para visualizar archivos PDF en el navegador.
- **React-Router-dom:** Para manejar enrutamiento
- **React Icons:** Para agregar iconos
- **React Redux + Toolkit:** Para manejo de estados globales
- **React Spinner:** Para mostrar spiners de carga
- **Notistack:** Para notificaciones más amigables

Autenticación:

- **JWT (JSON Web Tokens)**: Para manejo de sesiones de usuarios de forma segura.

Funcionalidades a Implementar

Backend:

1. **API de creación de documentos PDF:**
    - **Ruta: POST /api/pdf/generar**
    - **Recibe:**
        - **Casa (Ej: "Casa A")**
        - **Fecha (Mes y Año)**
        - **Gastos fijos (Gastos comunes, Estacionamiento, Citófonos, Deuda anterior, Fondo de reserva)**
        - **Gastos adicionales (opcional).**
2. **Generación del PDF:**
    - **Crear documento con formato estándar basado en el diseño entregado.**

- **Incluir todos los gastos listados.**

3. **Manejo de Archivos:**
    - **Verificar si la carpeta correspondiente al mes existe.**
    - **Crear carpeta si no existe.**
    - **Guardar los PDFs en rutas /documentos/{YYYY-MM}/CasaX.pdf.**
4. **API para listar PDFs:**
    - **Ruta: GET /api/pdf/listar/:mes**
    - **Permite al frontend obtener la lista de documentos disponibles por mes.**
5. **API de autenticación:**
    - **Rutas de login (POST /api/auth/login) y validación de token.**
    - **Protección de rutas con middleware de autenticación.**

Frontend:

1. **Login de Usuario (Administrador):**
    - **Formulario de acceso protegido mediante JWT.**
    - **Persistencia de sesión.**
2. **Panel de Documentos:**
    - **Listar meses disponibles.**
    - **Mostrar lista de PDFs de cada mes.**
    - **Visualizar PDFs directamente en el navegador (sin descarga necesaria).**
3. **Visualizador de PDFs:**
    - **Integración de React-PDF o PDF.js para mostrar los documentos.**
    - **Navegación fluida entre archivos.**
4. **Manejo de Sesiones:**
    - **Redireccionar a Login si no hay sesión activa.**
    - **Proteger rutas de administración.**

Tareas Iniciales para el Equipo

Backend:

- **Tarea 1: Configurar el entorno de Node.js y Express.**
- **Tarea 2: Crear las rutas de autenticación y generación de PDF.**
- **Tarea 3: Implementar la generación dinámica de carpetas y almacenamiento de archivos.**
- **Tarea 4: Crear servicio para listar PDFs disponibles.**
- **Tarea 5: Aplicar middleware de autenticación en rutas protegidas.**

Frontend:

- **Tarea 6: Configurar proyecto React con Vite.**
- **Tarea 7: Crear el formulario de login y gestión de tokens.**
- **Tarea 8: Implementar panel de documentos y navegación entre PDFs.**
- **Tarea 9: Implementar visualizador de PDFs.**
- **Tarea 10: Proteger rutas de frontend basadas en sesión activa.**

Notas Adicionales

- **El diseño del PDF debe ser limpio, profesional y fácil de leer.**
- **El frontend debe ser responsivo para una buena experiencia en desktop y móvil.**
- **La autenticación debe ser segura y considerar expiración de tokens.**
- **Se debe prever la posibilidad de agregar nuevos gastos adicionales dinámicamente en el futuro.**