Manual Técnico - Sistema de Préstamo de Equipos de Laboratorio

Requisitos Previos

- Node.js (v18 o superior recomendado)
- npm o pnpm (gestor de paquetes)
- MongoDB (local o en la nube)

Instalación y Puesta en Marcha

1. Clonar el repositorio

```
git clone <url-del-repositorio>
cd proyecto_final
```

2. Configuración del Backend

1. Ir a la carpeta `backend`:

   ```
   cd backend
   ```

2. Instalar dependencias:

   ```
   npm install --legacy-peer-deps
   ```

3. Configurar variables de entorno:
   - Crear un archivo `.env` (si es necesario) con la cadena de conexión de MongoDB y otras variables.
4. Iniciar el servidor backend:

   ```
   npm run dev
   # o
   node server.js
   ```

   El backend corre por defecto en `http://localhost:5000`.

3. Configuración del Frontend

1. Ir a la carpeta `client`:

```
cd ../client
```

2. Instalar dependencias:

```
npm install --legacy-peer-deps
```

3. Iniciar el frontend:

```
npm run dev
```

El frontend corre por defecto en `http://localhost:3030`.

Estructura General del Proyecto

- **backend/**: API REST en Node.js/Express. Modelos Mongoose para MongoDB.
- **client/**: Aplicación web en Next.js (React + TypeScript).

Explicación General del Código y Lógica

Backend

- **Modelos (models/):** Definen la estructura de los datos en MongoDB (usuarios, préstamos, equipos, laboratorios, notificaciones).
- **Controladores (controllers/):** Lógica de negocio y endpoints de la API. Ejemplo: `loanController.js` gestiona la creación, actualización y devolución de préstamos.
- **Rutas (routes/):** Definen los endpoints disponibles para cada recurso (equipos, préstamos, usuarios, etc).
- **Middleware:** Funciones intermedias para autenticación, manejo de errores, roles, subida de archivos, etc.
- **Utils:** Funciones auxiliares como generación de PDFs, validaciones y helpers.
- **server.js/app.js:** Punto de entrada del backend, configura Express, rutas y conexión a MongoDB.

Frontend

- **src/app/**: Páginas principales de la aplicación (Next.js). Cada carpeta es una ruta.
- **src/components/**: Componentes reutilizables (tablas, formularios, modales, selectores, etc).
- **src/services/**: Funciones para consumir la API del backend (fetch, update, create, etc).
- **src/hooks/**: Hooks personalizados para lógica de UI y autenticación.
- **src/styles/**: Archivos de estilos globales.
- **src/lib/**: Utilidades generales.

Lógica de Préstamos y Devoluciones

- Al crear un préstamo, se descuenta la cantidad prestada del inventario.

- Al devolver, se puede registrar una devolución parcial y una nota. El inventario se ajusta automáticamente:
    - Si no se devuelve todo, el sistema descuenta del total y suma solo lo devuelto al disponible.
- El historial muestra detalles, cantidades devueltas y notas.

Consejos para Desarrolladores Nuevos

- **Para entender la lógica de negocio:** Empieza revisando los modelos en `backend/models/` y los controladores en `backend/controllers/`.
- **Para modificar el frontend:** Los flujos principales están en `src/app/loans/`, `src/app/inventory/` y los componentes en `src/components/`.
- **Para agregar endpoints:** Crea la lógica en el controlador, expón la ruta y actualiza los servicios en el frontend si es necesario.
- **Para debuggear:** Usa los logs en consola y revisa los mensajes de error en la API y el frontend.
- **Para desarrollo local:** Asegúrate de tener MongoDB corriendo y ambos servidores (backend y frontend) activos.

Comandos Útiles

- `npm run dev` (en backend y client): Inicia ambos servidores en modo desarrollo.
- `npm run build` y `npm start` (en client): Para producción.

Notas

- El backend y frontend se comunican por HTTP (CORS habilitado por defecto para localhost).
- Las rutas y puertos pueden configurarse en los archivos de configuración.
- Si tienes problemas con dependencias, usa siempre `npm install --legacy-peer-deps`.

Cualquier duda técnica, revisar los archivos README.md o contactar al desarrollador principal.