

Documentación del proyecto de sistemas de inventarios

Tecnologías Utilizadas

El proyecto sigue una arquitectura full-stack, separando claramente las responsabilidades del frontend y el backend.

Backend

- **Lenguaje: Python 3**
- **Framework: Flask** (un micro-framework ligero para construir APIs web).
- **Extensiones de Flask:**
 - **Flask-SQLAlchemy:** ORM (Object-Relational Mapper) para interactuar con la base de datos usando objetos de Python.
 - **Flask-Bcrypt:** Para la encriptación (hasheo) segura de las contraseñas de los usuarios.
 - **Flask-JWT-Extended:** Para implementar la autenticación mediante tokens web JSON (JWT).
 - **Flask-Cors:** Para permitir las peticiones desde el dominio del frontend (Cross-Origin Resource Sharing).
- **Servidor de Base de Datos: PostgreSQL**
- **Adaptador de Base de Datos: psycopg2-binary**

Frontend

- **Librería: React 18** (para la construcción de la interfaz de usuario).
- **Entorno de Desarrollo: Vite** (un empaquetador y servidor de desarrollo rápido).
- **Lenguaje: JavaScript (JSX)** y **CSS3**.

Entornos de Desarrollo

- **Backend:** PyCharm
- **Frontend:** Visual Studio Code

Estructura del Proyecto

El proyecto está organizado en dos carpetas principales, backend y frontend, para mantener una separación limpia de los entornos.

```
proyecto-inventario/
├── backend/
│   ├── .venv/           # Entorno virtual de Python con las dependencias
│   ├── app.py           # Archivo principal de Flask (API, rutas, lógica)
│   └── requirements.txt  # Lista de dependencias de Python para instalar con pip
└── frontend/
    ├── node_modules/    # Dependencias de JavaScript (instaladas por npm)
    ├── public/           # Archivos estáticos
    ├── src/
    │   ├── components/  # Componentes reutilizables de React
    │   └── AddProductForm.jsx
```

—	Dashboard.jsx	
—	EditProductModal.jsx	
—	LoginForm.jsx	
—	ProductList.jsx	
—	RegisterForm.jsx	
—	App.css	# Hoja de estilos principal
—	App.jsx	# Componente raíz y controlador de vistas
—	main.jsx	# Punto de entrada de la aplicación React
—	.gitignore	# Archivos y carpetas ignorados por Git
—	index.html	# Plantilla HTML principal
—	package.json	# Metadatos y dependencias del proyecto frontend

Guía de Instalación y Uso

Seguir los siguientes pasos para configurar y ejecutar el proyecto en un entorno local.

Prerrequisitos

- Tener instalado **Python 3.10** o superior.
- Tener instalado **Node.js 18** o superior (incluye npm).
- Tener un servidor de **PostgreSQL** activo.

1. Configuración del Backend

1. Navega a la carpeta del backend:

Bash

```
cd C:\Users\USUARIO\Desktop\Sistema_inventarios\BackEnd
```

2. Crea y activa un entorno virtual:

Bash

```
# Crear el entorno
python -m venv .venv
```

```
# Activar en Windows
.\.venv\Scripts\activate
```

```
# Activar en macOS/Linux
source .venv/bin/activate
```

3. Instala las dependencias:

Bash

```
pip install -r requirements.txt
```

4. Configura la Base de Datos:

- Crea una nueva base de datos en PostgreSQL (ej. sistema_inventario_db).

- Abrir el archivo app.py y verificar que la línea `app.config['SQLALCHEMY_DATABASE_URI']` apunte a la base de datos con las credenciales correctas.

5. Inicia el servidor:

Bash

`flask run`

El backend estará disponible en `http://127.0.0.1:5000`.

2. Configuración del Frontend

1. **Abre una nueva terminal** y navega a la carpeta del frontend:

Bash

`cd C:\Users\USUARIO\Desktop\Sistema_inventarios\FrontEnd`

2. **Instala las dependencias de Node.js:**

Bash

`npm install`

3. **Inicia el servidor de desarrollo:**

Bash

`npm run dev`

El frontend estará disponible en `http://localhost:5173`.

3. Uso de la Aplicación

Para usar la aplicación, asegurarse de que **ambos servidores (Flask y Vite) estén corriendo** en sus respectivas terminales. Acceder a la aplicación a través de la URL del frontend (`http://localhost:5173`).