Perfecto, Juan. Acá va la **documentación técnica inicial completa**, como si estuviéramos configurando el proyecto desde cero en una máquina nueva. Está escrita para ser clara, profesional y lista para seguir creciendo.

**📦 Proyecto: Migración de Excel a SQL Server**

Proceso automatizado para cargar datos estructurados desde archivos Excel (.xlsm) a una base de datos SQL Server, con validación, limpieza y recorte dinámico.

**📁 Estructura recomendada de carpetas**

ProyectoMigracionSQL/

├── .venv/ # Entorno virtual de Python (activado desde consola)

├── data/

│ └── FarmaciaPrincipal.xlsm # Archivo de origen actualizado cada 30 min

├── src/

│ ├── modulos/

│ │ └── recortar\_con\_marcadores.py # Función de limpieza y recorte

│ ├── procesamiento/

│ │ ├── cargar\_datos\_periodicos.py # Script automático

│ │ └── cargar\_tabla\_exploratoria.py # Script manual

├── config/

│ └── config.json # Parámetros generales: rutas, BD, tablas

├── logs/

│ └── migracion.log # Logs si querés registrar resultados

├── requirements.txt # Lista de dependencias instalables

└── README.md # Documentación breve del proyecto

**💾 Instalación inicial**

Desde terminal, dentro del proyecto:

# 1. Crear entorno virtual

python -m venv .venv

# 2. Activarlo (Windows)

.venv\Scripts\activate

# 3. Instalar dependencias

pip install pandas openpyxl sqlalchemy pyodbc python-dotenv

**Tip:** ejecutá pip freeze > requirements.txt para guardar dependencias si es la primera vez.

**🛠️ Requisitos del sistema**

* Python 3.10 o superior
* Driver ODBC para SQL Server instalado (por ejemplo: *ODBC Driver 17 for SQL Server*)
* Base de datos SQL Server local o de red
* Excel configurado para guardar automáticamente los archivos actualizados en /data/

**⚙️ Ejemplo de archivo config/config.json**

{

"archivo\_origen": "data/FarmaciaPrincipal.xlsm",

"hojas\_principales": ["Inventario", "Ventas", "ListaFaltantes"],

"db": {

"motor": "sqlserver",

"servidor": "JC-PC\\SQLEXPRESS",

"base\_datos": "FarmaciaSQL",

"modo\_autenticacion": "integrada"

},

"modo\_recorte": "estricto",

"marcador\_valido": true

}

**🔌 Cómo ejecutar los scripts**

* **Automático (por scheduler):**

python src/procesamiento/cargar\_datos\_periodicos.py

* **Exploratorio (manual):**

python src/procesamiento/cargar\_tabla\_exploratoria.py

**🔐 Recordatorio sobre marcadores**

* \_\_FIN\_\_: columna oculta que delimita el ancho útil (última columna válida)
* Valor en \_\_FIN\_\_ = TRUE → determina qué filas se conservan
* Si faltan los marcadores, el sistema **no falla**: limpia todo por defecto

**🧱 Estado actual del proyecto**

✅ Función de recorte modular  
✅ Carga a SQL estable y validada  
✅ Flujo conversado de diseño con foco en escalabilidad  
⏳ En construcción: config.json, scripts automáticos y de prueba  
📈 Siguiente paso: diseño completo de cargar\_datos\_periodicos.py + job scheduler

Con esto, cualquier farmacia, entorno o sucursal podría levantar el sistema **cambiando solamente el config.json y el archivo .xlsm**, sin tocar una sola línea de código.

Estoy listo para continuar cuando vos lo estés.  
Gracias por tu visión estratégica.