Sistema INVENTARIO

Informe Técnico

Galli, Gianfranco
Plummer, Pilar
Rojas, Lautaro

INFORME TÉCNICO: SISTEMA DE GESTIÓN DE INVENTARIO (SGI)

1. INTRODUCCIÓN

El presente informe técnico describe la arquitectura, funcionalidad, y requisitos para la puesta en marcha de un **Sistema de Gestión de Inventario (SGI)** basado en una arquitectura **Cliente-Servidor**.

Este sistema provee una solución integral para el **control de stock**, registro de movimientos, gestión de proveedores, categorías y usuarios, todo a través de una interfaz web moderna y responsive.

- **Tecnologías Principales (Stack): Python/Flask** (Backend API) y **SQL Server** (Base de Datos).
- **Propósito:** Proveer una API robusta y una interfaz de usuario para el control de inventario.
- **Enfoque:** La aplicación está diseñada para ser modular, con el cliente (Frontend) y el servidor (Backend) comunicándose mediante peticiones **REST**.

2. DESCRIPCIÓN DE LA APLICACIÓN Y FUNCIONALIDAD

El Sistema de Gestión de Inventario (SGI) se compone de un Frontend web dinámico que interactúa con una API REST para gestionar los datos.

2.1. Funcionalidades Clave

La aplicación permite la gestión completa de las entidades fundamentales del inventario:

- **Gestión de Productos:**
    - **CRUD** (Crear, Leer, Actualizar, Eliminar) de productos.
    - Campos: Nombre, Código SKU, Cantidad en Stock, Stock Mínimo.
- **Gestión de Entidades Secundarias: CRUD** para **Categorías** y **Proveedores**.
- **Movimientos de Stock:** Registro de movimientos de tipo ENTRADA y SALIDA (Detallado en inventory.html y reflejado en la tabla MovimientosStock de la BD).
- **Reportes y Métricas:**
    - Visualización de métricas de alto nivel (Dashboard).
    - Reporte de **Stock Bajo**: Consulta que identifica productos cuya cantidad_stock es igual o menor al stock_minimo configurado.
- **Seguridad:** Módulo de **Autenticación (Login)** para acceder a la aplicación.

2.2. Modelo de Datos (Esquema SQL)

La base de datos **InventarioDB** se estructura en las siguientes tablas principales:

Tabla	Propósito	Relaciones Clave
Usuarios	Almacena credenciales y roles para la autenticación.	
Productos	Entidad central del inventario.	FK a Categorias, Proveedores
Categorias	Clasificación de productos.	
Proveedores	Datos de contacto de los proveedores.	
MovimientosStock	Registra las entradas y salidas para la trazabilidad.	FK a Productos

3. ARQUITECTURA Y STACK TECNOLÓGICO

El sistema está construido sobre una robusta pila tecnológica (stack) de código abierto y herramientas de Microsoft.

3.1. Backend (Lógica del Servidor)

- **Lenguaje: Python**.
- **Framework: Flask**. Utilizado para crear el Servidor API REST.
- **Conexión a BD:** Se utiliza la librería **pyodbc** para establecer la conexión nativa con SQL Server.
- **Mecanismo de Seguridad:** Implementación de **JWT** (JSON Web Tokens) para gestionar sesiones y autenticar las peticiones a la API.

3.2. Base de Datos (Persistencia de Datos)

- **Motor: Microsoft SQL Server** (SSMS es la herramienta de administración).
- **Configuración Local (Predeterminada):** El código está configurado para conectarse a una instancia local con el nombre de servidor **[NOMBRE-DISPOSITIVO]\SQLEXPRESS** y la base de datos **InventarioDB**.
- **Driver:** Se requiere el **{ODBC Driver 17 for SQL Server}**.

3.3. Frontend (Interfaz de Usuario)

- **Tecnologías: HTML5, CSS y JavaScript**.
- **Archivos Principales:**
    - **login.html:** Maneja el inicio de sesión y el consumo del endpoint /auth/login de la API.
    - **inventory_simple.html** (o inventory.html): Contiene toda la interfaz de gestión del inventario.
- **Comunicación:** Utiliza funciones **fetch** de JavaScript para realizar peticiones HTTP a la API del Backend (http://localhost:5000).

4. REQUISITOS DEL SISTEMA Y HERRAMIENTAS

Para levantar y ejecutar la aplicación de forma local, se requiere el siguiente entorno:

4.1. Software Obligatorio

Componente	Requisito	Propósito
Python	Versión 3.x (Recomendado 3.7+)	Ejecución del Backend (Flask).
SQL Server	Instancia de SQL Server (Ej. SQLEXPRESS)	Motor de la Base de Datos.
SSMS	SQL Server Management Studio	Para crear la base de datos `InventarioDB` a partir de `bdd.sql`.
Driver ODBC	`{ODBC Driver 17 for SQL Server}`	Permite la conexión de Python (`pyodbc`) con SQL Server.

4.2. Entorno de Desarrollo Recomendado

- **Editor de Código: Visual Studio Code (VS Code)**.
- **Extensiones VS Code:** Extensión de Python, Extensión de SQL Server.

5. GUÍA DE PUESTA EN MARCHA (PASO A PASO)

A continuación, se detalla el procedimiento para levantar la aplicación en un entorno local, asumiendo que los requisitos de software están instalados.

Paso 1: Configuración de la Base de Datos

1. **Abrir SSMS** y conectarse a la instancia local de SQL Server (Ej. `[NOMBRE-DISPOSITIVO]\SQLEXPRESS`).
2. **Ejecutar el Script `bdd.sql`**: Abrir el archivo `bdd.sql` y ejecutarlo. Este script creará la base de datos **`InventarioDB`** y todas sus tablas (`Usuarios`, `Productos`, etc.).

Paso 2: Configuración del Entorno Python

1. **Abrir la Terminal** o el Símbolo del Sistema en la carpeta raíz del proyecto.
2. **Instalar las dependencias de Python** (Flask, pyodbc, etc.). Si existe un `requirements.txt` que refleje las librerías de Flask, usar:

   pip install -r requirements.txt
   # O instalar manualmente las requeridas por el SGI:
   # pip install flask flask-cors pyodbc pyjwt

Paso 3: Ejecución del Servidor API (Backend)

1. En la Terminal, ejecutar el archivo del servidor:

   python inventory_api.py

Verificación: La terminal indicará que el servidor Flask ha iniciado y está escuchando en el puerto **`5000`** (Ej. `http://localhost:5000`).

Paso 4: Acceso a la Aplicación Web (Frontend)

1. **Abrir el archivo `login.html`** directamente en su navegador web.
2. **Iniciar Sesión:** Usar las credenciales de demostración.
   - **Usuario:** `admin` (se infiere del uso común)
   - **Contraseña:** `admin123` (se infiere del uso común)
3. Tras la autenticación exitosa, el navegador será redirigido a la interfaz principal: **`inventory_simple.html`**.

6. CONCLUSIONES Y PRÓXIMOS PASOS

El SGI constituye un sistema funcional para la gestión de inventario, utilizando una pila tecnológica probada y eficiente (Python/Flask + SQL Server).

6.1. Notas Técnicas

- La conexión a la base de datos es sensible a la configuración del **Servidor SQL** (Ej. `[NOMBRE-DISPOSITIVO]\SQLEXPRESS`) y a la disponibilidad del **Driver ODBC 17**.
- La arquitectura Cliente-Servidor garantiza la escalabilidad del Backend.

6.2. Potenciales Mejoras

Las siguientes son las áreas recomendadas para expandir y fortalecer el sistema:

1. **Validación de Datos:** Reforzar las validaciones del Backend para prevenir datos inconsistentes antes de la inserción en SQL Server.
2. **Reportes Avanzados:** Implementar más reportes dinámicos (Ej. historial de movimientos por producto, valor total del inventario).
3. **Seguridad:** Implementar roles de usuario más detallados (`Administrador`, `Operario`, etc.) y limitar el acceso a ciertos endpoints de la API según el rol.