Universidad Provincial del Sudoeste

(UPSO)

Documentación del Proyecto

Sistema de Gestión de Inventario

Materia: Proyecto Informático

Alumno: Rodrigo Matías Molina Navarro

Profesor: Carlos Enrique Berger

Año: 2025

Introducción

Este proyecto tiene como objetivo desarrollar un sistema de gestión de inventario que facilite el control de productos, categorías, stock, proveedores, órdenes de compra y generación de reportes. El sistema está diseñado siguiendo una arquitectura cliente-servidor moderna, con separación clara entre frontend y backend.

Requisitos del Sistema

- Python 3.x - https://www.python.org/downloads/
- Git - https://git-scm.com/
- XAMPP - https://www.apachefriends.org/es/download.html

Estructura del Proyecto

Carpetas y Archivos Principales:

backend/

- .venv/: Entorno virtual de Python con las dependencias del proyecto.
- api/: Contiene la configuración de base de datos, modelos, rutas y seguridad.
- settings/: Scripts SQL para configuración inicial y creación de la base de datos.
- .env: Variables de entorno (credenciales de base de datos).
- main.py: Punto de entrada principal que inicia el servidor Flask.

frontend/

- css/: Archivos de estilos CSS para la interfaz.
- js/: Scripts JavaScript para interacción con la API.
- pages/: Páginas HTML del sistema (dashboard, productos, etc.).
- login.html: Página de inicio de sesión.
- register.html: Página de registro de usuarios.

diagramas/

- Diagramas UML de secuencia para cada módulo del sistema.

Diagramas UML

Diagramas de Secuencia

El proyecto incluye diagramas de secuencia para cada módulo del sistema, ubicados en la carpeta /diagramas/:

- **User.png**: Flujo de registro e inicio de sesión con autenticación JWT.
- **Category.png**: Operaciones CRUD de categorías.
- **Product.png**: Gestión completa de productos.
- **Stock.png**: Control de inventario y alertas de stock bajo.
- **Supplier.png**: Gestión de proveedores.
- **Order.png**: Órdenes de compra con confirmación y actualización de stock.

Cada diagrama muestra la interacción entre el Usuario, la Interfaz (Frontend), el Backend (API) y la Base de Datos MySQL, incluyendo los métodos HTTP utilizados y las consultas SQL ejecutadas.

Base de Datos: Descripción de Tablas

1. users

Tabla para gestionar los usuarios del sistema.

- id: Identificador único (clave primaria)
- username: Nombre de usuario único
- email: Correo electrónico
- password: Contraseña cifrada (hash)

2. categories

Tabla para clasificar los productos en categorías.

- id: Identificador único (clave primaria)
- name: Nombre de la categoría
- descripcion: Descripción de la categoría
- user_id: Referencia al usuario (clave foránea)

3. products

Tabla para registrar los productos disponibles.

- id: Identificador único (clave primaria)
- name: Nombre del producto
- price: Precio del producto
- category_id: Referencia a la categoría (clave foránea)
- user_id: Referencia al usuario (clave foránea)

4. stock

Tabla para gestionar el inventario de productos.

- product_id: Identificador del producto (clave primaria y foránea)
- quantity: Cantidad disponible
- user_id: Referencia al usuario

5. suppliers

Tabla para registrar los proveedores.

- id: Identificador único (clave primaria)
- name_supplier: Nombre del proveedor
- phone: Teléfono de contacto
- mail: Correo electrónico
- user_id: Referencia al usuario

6. suppliers_products

Tabla intermedia para relación proveedores-productos.

- supplier_id: ID del proveedor (clave foránea)
- product_id: ID del producto (clave foránea)
- user_id: Referencia al usuario

7. purchase_orders

Tabla para órdenes de compra.

- id: Identificador único (clave primaria)
- order_date: Fecha de creación
- received_date: Fecha de recepción
- status: Estado (pending, completed, deleted)
- user_id: Referencia al usuario

8. order_products

Tabla para productos en cada orden.

- id: Identificador único
- order_id: ID de la orden (clave foránea)
- product_id: ID del producto (clave foránea)
- quantity: Cantidad solicitada

Frontend

Diseño Responsivo

La interfaz de usuario está diseñada con CSS responsive, adaptándose a diferentes tamaños de pantalla (desktop, tablet, móvil). Se utilizan variables CSS para mantener consistencia en colores y estilos.

Páginas del Sistema

- **login.html / register.html**: Autenticación de usuarios
- **dashboard.html**: Panel principal con acceso a todos los módulos
- **products.html**: Gestión de productos (CRUD completo)
- **categories.html**: Gestión de categorías
- **stock.html**: Control de inventario con alertas
- **suppliers.html**: Gestión de proveedores
- **orders.html**: Órdenes de compra
- **reports.html**: Reportes y estadísticas

Objetos JavaScript Globales

- **API_CONFIG**: Configuración de URL base del backend
- **Auth**: Manejo de autenticación (token, usuario, logout)
- **UI**: Notificaciones y feedback visual

Flujo de Usuario

1. Registro e Inicio de Sesión

El usuario accede a register.html para crear una cuenta. Después del registro exitoso, es redirigido al login donde ingresa sus credenciales. El sistema valida los datos y genera un token JWT que se almacena en localStorage.

2. Dashboard

Una vez autenticado, el usuario accede al dashboard que muestra un menú lateral con acceso a todos los módulos del sistema. El nombre del usuario se muestra en la cabecera junto con el botón de cerrar sesión.

3. Gestión de Productos

Permite crear, editar y eliminar productos. Cada producto tiene nombre, precio y categoría opcional. Al crear un producto, automáticamente se crea un registro en la tabla stock mediante un trigger de base de datos.

4. Gestión de Categorías

Permite organizar los productos en grupos. Cada categoría tiene nombre y descripción. Al eliminar una categoría, los productos asociados quedan sin categoría.

5. Gestión de Stock

Muestra el inventario actual con indicadores visuales de estado (Normal, Stock Bajo, Sin Stock). Permite actualizar la cantidad de cada producto. Genera alertas automáticas para productos con menos de 5 unidades.

6. Gestión de Proveedores

Permite registrar y eliminar proveedores con sus datos de contacto (nombre, teléfono, email).

7. Órdenes de Compra

Permite crear órdenes de compra especificando producto y cantidad. Al confirmar una orden, el stock del producto se actualiza automáticamente sumando las unidades pedidas. Las órdenes pueden tener estado: pendiente, completada o eliminada.

8. Reportes

Muestra estadísticas del inventario: total de productos, unidades totales, productos con stock bajo y sin stock. También lista los productos más solicitados en las órdenes.

Instalación y Configuración

Requisitos Previos

- Python 3.x instalado
- Git instalado
- XAMPP instalado y MySQL corriendo

Pasos de Instalación

1. Clonar el repositorio:

```
git clone https://github.com/Rodrimolina10/sistema-gestion-inventario.git
```

2. Navegar al directorio backend:

```
cd proyecto-gestion-inventario-upso/PROYECTO-FINAL-COMPLETO/backend
```

3. Crear entorno virtual:

```
py -3 -m venv .venv
```

4. Activar entorno virtual:

```
.venv/Scripts/activate
```

5. Instalar dependencias:

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

6. Crear base de datos:

Ejecutar los scripts SQL en settings/ usando phpMyAdmin o MySQL CLI

7. Configurar variables de entorno:

Crear archivo .env en backend/ con:

```
DB_HOST=localhost
DB_PORT=3306
DB_USER=tu_usuario
DB_PASSWORD=tu_password
DB_NAME=flask_app_db
PORT=5000
HOST=localhost
```

8. Ejecutar el servidor:

```
python main.py
```

9. Abrir el frontend:

Abrir frontend/login.html en el navegador

Tecnologías Utilizadas

Backend

- **Python 3.x**: Lenguaje de programación principal
- **Flask**: Microframework web para crear la API REST
- **Flask-CORS**: Manejo de Cross-Origin Resource Sharing
- **PyJWT**: Generación y validación de tokens JWT
- **mysql-connector-python**: Driver de conexión a MySQL
- **python-dotenv**: Carga de variables de entorno
- **Werkzeug**: Utilidades de seguridad (hash de contraseñas)

Frontend

- **HTML5**: Estructura de las páginas
- **CSS3**: Estilos y diseño responsive
- **JavaScript (Vanilla)**: Lógica del cliente y comunicación con API
- **Fetch API**: Peticiones HTTP al backend

Base de Datos

- **MySQL**: Sistema gestor de base de datos relacional
- **XAMPP**: Paquete que incluye MySQL y phpMyAdmin

Herramientas

- **Git**: Control de versiones
- **GitHub**: Repositorio remoto
- **Visual Studio Code**: Editor de código
- **PlantUML**: Generación de diagramas UML

Conclusión

El Sistema de Gestión de Inventario desarrollado proporciona una solución completa y funcional para el control de productos, categorías, stock, proveedores y órdenes de compra. El proyecto implementa una arquitectura cliente-servidor moderna con las siguientes características:

- API RESTful con autenticación JWT
- Interfaz de usuario responsive y amigable
- Base de datos relacional bien estructurada
- Documentación técnica y de usuario completa
- Diagramas UML para cada módulo
- Código organizado siguiendo buenas prácticas

El sistema cumple con todos los requisitos establecidos en el enunciado del proyecto y está preparado para futuras mejoras y extensiones, como se documenta en el archivo de mejoras pendientes.

Agradecimientos

Agradezco al profesor Carlos Enrique Berger por su orientación y apoyo durante el desarrollo de este proyecto, lo cual ha sido fundamental para su éxito.