Planificación Técnica de Desarrollo

Sistema de Gestión de Inventario y Cuadratura Comercial

Consultoría de Software - Flask/SQLite Stack

12 de enero de 2026

Introducción

Este documento detalla la planificación para la migración de un sistema basado en hojas de cálculo hacia una aplicación web centralizada utilizando el micro-framework **Flask** y **SQLite**. El objetivo es eliminar el desorden de datos y automatizar la fórmula:

$$\text{Stock Actualizado} = \text{Stock Inicial} + \text{Compras} - \text{Ventas}$$

Módulos Críticos y Flujo de Datos

2.1. 1. Módulo de Abastecimiento y Lotes

Basado en la hoja "Detalle Stock", se implementará un sistema de trazabilidad por lotes.

- **Gestión de Lote PD (Product Data):** Registro de fecha de recepción, elaboración y vencimiento.
- **Control de Proveedor:** Almacenamiento de Código de Proveedor y N° de Factura de entrada.
- **Estado de Facturación:** Marcado de facturas como "Pagado" o "Pendiente" para el flujo de caja.

2.2. 2. Módulo de Logística y Reparto

Este módulo digitaliza la "Hoja de Repartidor", optimizando el seguimiento físico:

- **KPIs de Ruta:** Registro de Kilometraje (Km) y tiempo de ruta para medir eficiencia.
- **Geo-referenciación:** Campos de Sector y Ciudad para futuros mapas de calor de ventas.
- **Vínculo de Venta:** Cada salida de bodega debe estar amarrada a una Nota de Venta o Factura.

2.3. 3. Motor de Cuadratura

Es el corazón del sistema. Debe comparar las ventas de plataformas externas (como Jumpseller) con el inventario físico disponible.

- **Conciliación:** Validación automática de montos y cantidades facturadas vs. despachadas.

Especificaciones Técnicas

3.1. Modelado de Base de Datos (SQLAlchemy)

Se utilizarán las siguientes tablas principales:

a) **User:** Control de acceso para el personal.

b) **Product:** Maestro de artículos con stock crítico.

c) **LotDetail:** Almacén de fechas de vencimiento y códigos específicos.

d) **Sale:** Registro de ventas (integrando campos de e-commerce).

e) **LogisticsRoute:** Registro de despachos y métricas de transporte.

3.2. Interfaz de Usuario (UX/UI)

- **Estilo:** Minimalista basado en Tailwind CSS, priorizando la legibilidad en dispositivos móviles para el personal de bodega y repartidores.
- **Responsividad:** Tablas con scroll horizontal para datos extensos de stock y botones de acción rápida para entrada/salida.

Cronograma de Implementación

Semana 1: Configuración del entorno Flask y migración inicial (ETL) de los Excels actuales a SQLite.

Semana 2: Desarrollo de formularios de ingreso (Abastecimiento) y lógica de lotes.

Semana 3: Implementación de vistas de logística y reportes de cuadratura.

Semana 4: Despliegue en servidor (Render/Railway) y capacitación de usuario final.

Implementación Técnica Realizada

5.1. Esquema de Base de Datos

Se ha implementado el siguiente modelo relacional en SQLite:

- **User:** Gestión de usuarios con roles (admin, bodega, driver).

- **Supplier:** Proveedores de productos.
- **Product:** Catálogo maestro (SKU, Nombre, Stock Crítico). Relacionado 1 a N con `Lot`.
- **InboundOrder:** Cabecera de recepción de productos (Facturas de compra).
- **Lot:** Trazabilidad de lotes. Incluye `lot_code`, fecha de vencimiento y cantidades.
- **Sale:** Cabecera de pedido/venta. Estados: pending, assigned, in_transit, delivered.
- **SaleItem:** Detalle de productos por venta. Normaliza la estructura plana de los Excel antiguos.
- **LogisticsRoute:** Agrupación de ventas para despacho asignadas a un conductor.

5.2. Manual de Operación (Walkthrough)

5.2.1. Ejecución del Entorno

Para iniciar el sistema en entorno local:

1. Activar el entorno virtual:

```
source venv/bin/activate
```

2. Ejecutar la aplicación Flask:

```
export FLASK_APP=run.py
flask run
```

La aplicación estará disponible en `http://127.0.0.1:5000/`.

5.2.2. Gestión de Base de Datos

La base de datos se encuentra en `instance/inventory.db`. Para actualizar el esquema ante nuevos cambios:

```
flask db migrate -m "Descripción"
flask db upgrade
```