

RESUMEN COMPLETO DEL SISTEMA (BACKEND + FRONTEND)

1. VISIÓN GENERAL DEL SISTEMA

Backend

- Django 5 + Django REST Framework.
- Apps: accounts, catalog, sales, inventory, promos, cashdesk, audit, dte, reports.
- Autenticación: SimpleJWT.
- Base: SQLite.
- Router DRF para productos, ventas, caja, usuarios, etc.
- Rutas extras: login, refresh, reports, preview ventas, boletas PDF.

Frontend

- React + Vite.
- pages/: POS, Stock, Caja, VentasDia, Dashboard, AdminPanel.
- components/: Header, PrivateRoute, AdminRoute, ProductRow, BarcodeScanner.
- utils/: formatos.
- App.jsx define rutas con react-router-dom.
- Axios centralizado en api.js (manejo de tokens y refresh).
- Autenticación en auth.js y useMe.js.

Comunicación

- Todo el front llama el backend vía api.js.
- Boletas: descarga binaria desde /dte/boleta//.

2. BACKEND POR APP

ACCOUNTS

- Modelo User con rol OWNER/ADMIN/SELLER.
- Permisos: IsOwnerOrAdmin, ReadOnlyOrAdmin.
- Vistas: MeView (/auth/me/), UserAdminViewSet (/users/).

CATALOG

- Modelos: Category, Product (SKU, precio, stock, top_seller).
- ProductViewSet con búsquedas y auditoría de cambios de precio.
- CategoryViewSet maneja errores al eliminar categorías usadas.

SALES

- Sale y SaleItem.
- SaleSerializer recibe items y crea venta + ítems.
- SaleViewSet: checkout_sale actualiza stock, crea DTE y AuditLog.
- /sales/{id}/void/ revierte stock.
- /sales/preview/ calcula descuentos sin grabar venta.

CASHDESK

- CashSession (abre/cierra caja).
- ViewSet maneja cierre con diferencias de monto.
- Front pide PIN 1234 para abrir/cerrar.

PROMOS

- Modelo Promotion: tipo (porcentaje/monto), valor, categoría, productos.
- best_unit_discount calcula mejor descuento.
- apply_promotions_to_sale aplica descuentos a la venta.

INVENTORY

- InventoryMovement registra entradas/salidas.
- register_movement actualiza stock y registra el movimiento.

DTE

- DTE OneToOne con Sale.
- Boleta PDF generada en memoria por _build_boleta_pdf.
- Front descarga boleta desde VentasDelDia.

AUDIT

- AuditLog guarda actor, acción, modelo, cambios.
- Admin ve todo; vendedores solo lo propio.

REPORTS

- SalesReportView, InventoryReportView.
- ExportView genera CSV/XLSX.

3. FRONTEND DETALLADO

Estructura general

- App.jsx + Layout + Header definen navegación y permisos.
- Header usa useMe para mostrar rutas según rol.

Auth y API

- Tokens en localStorage (auth.js).
- Axios agrega Authorization y maneja refresh automático.
- useMe carga /auth/me/.

Páginas clave

- POS: búsqueda por SKU/nombre, carrito, descuentos, checkout.
- Stock: lista, métricas, top_seller, edición según rol.
- Promos: CRUD completo de promociones.
- Caja: abrir/cerrar con PIN.
- VentasDelDia: ventas desde apertura de caja, anular y boletas.
- Dashboard: polling cada 10s, métricas y gráfico 7 días.
- AdminPanel: CRUD de usuarios.

4. FLUJO TÍPICO

1. Login → /auth/login/ devuelve tokens.
2. useMe obtiene usuario actual y su rol.
3. POS:
 - Búsqueda → carrito → preview descuentos → checkout → stock y DTE.
4. Caja:
 - Abrir sesión → ventas del día → cerrar sesión.
5. Reportes y boletas desde /dte/boleta//.

5. AUTENTICACIÓN

Backend

- SimpleJWT: login y refresh.
- Permisos DRF y roles accounts.User.

Frontend

- Tokens en localStorage.
- Axios interceptor usa refresh si access expira.
- PrivateRoute y AdminRoute protegen vistas.

6. CHULETA PARA DEFENSA

- Backend = 9 apps separadas por responsabilidad.
- Modelo central = Sale + SaleItem.
- Stock siempre pasa por register_movement.
- Promos: best_unit_discount y apply_promotions_to_sale.
- Auth: JWT + useMe.
- Front: React + Vite + Axios centralizado.
- Flujo: login → POS → ventas → caja → reportes/boleta.

7. PREGUNTAS TÍPICAS

- ¿Por qué tantas apps? → Separación de responsabilidades.
- ¿Cómo manejas auth? → JWT con refresh automático.
- ¿Cómo garantizas stock correcto? → Siempre vía register_movement.
- ¿Cómo aplicas descuentos? → best_unit_discount + apply_promotions_to_sale.
- ¿Qué pasa si falla boleto? → Flujo de venta no se afecta.
- ¿Cómo defines “ventas del día”? → Desde opened_at de la CashSession.