**Objetivo del Proyecto:** Desarrollar un sistema de Punto de Venta (POS) integral y multicanal, diseñado para optimizar las operaciones de ventas, inventario y gestión de pedidos, con especial énfasis en la usabilidad, seguridad, escalabilidad y adaptabilidad a diversos modelos de negocio (restaurantes/comida rápida y tiendas/venta de productos), integrándose con plataformas de delivery líderes en Colombia.

**Inspiración de Diseño (Referencia Visual):** La interfaz del sistema se inspirará en el diseño moderno, limpio y funcional del dashboard adjunto `

que proporciona una base sólida para la estética y la organización de la información.

**Requisitos Funcionales Generales:**

1. **Módulo de Ventas (POS Central):**
   * Interfaz táctil intuitiva para agilizar el proceso de venta.
   * Registro de ventas con desglose detallado de productos, precios, descuentos e impuestos.
   * Funcionalidad para aplicar descuentos por artículo o al total de la compra.
   * Soporte para múltiples métodos de pago (efectivo, tarjeta de crédito/débito, QR, etc.).
   * Generación e impresión de facturas/recibos (físicos y digitales).
   * Gestión de propinas.
   * Capacidad para pausar y reanudar transacciones.
   * Búsqueda rápida de productos por código, nombre o categoría.
   * Registro de clientes para programas de fidelización (opcional, pero deseable).
2. **Módulo de Gestión de Inventario:**
   * Alta, baja y modificación de productos (nombre, descripción, precio, SKU, stock, categoría, unidad de medida, imagen).
   * Control de stock en tiempo real con alertas de bajo inventario.
   * Gestión de variantes de productos (ej. talla, color, ingredientes para restaurantes).
   * Registro de entradas y salidas de inventario (compras a proveedores, ajustes, devoluciones).
   * Reportes de inventario (valoración, rotación, productos más vendidos).
3. **Módulo de Gestión de Pedidos (Restaurantes/Comida Rápida):**
   * **Interfaz Móvil (para Toma de Pedidos):** Aplicación dedicada para dispositivos móviles (smartphones/tablets) para que el personal pueda tomar pedidos en mesa o para llevar/recoger.
     + Menú interactivo y personalizable.
     + Modificadores de productos (sin cebolla, extra queso, cocción).
     + Envío de comandas a la cocina/estación de preparación.
     + Gestión de mesas y estados (ocupada, libre, pagando).
     + Visualización del estado de los pedidos.
   * **Estación Central (Recepción y Procesamiento de Comandas):** Dashboard en la aplicación principal para visualizar, gestionar y procesar comandas recibidas desde los dispositivos móviles y plataformas de delivery.
     + Panel de comandas en tiempo real (KDS - Kitchen Display System opcional o lista de pedidos).
     + Actualización de estado de pedidos (en preparación, listo, entregado).
     + Asignación de comandas a personal de cocina.
     + Impresión de tickets de cocina/comandas.
4. **Conectividad y Sincronización:**
   * **Web y Móvil Integrados:** El sistema debe funcionar tanto como aplicación web (para la estación central POS y administración) como aplicación móvil nativa o PWA (para toma de pedidos y gestión simplificada). Ambas plataformas deben estar completamente sincronizadas en tiempo real.
   * Comunicación bidireccional entre la estación central y los dispositivos móviles de toma de pedidos.
5. **Seguridad y Gestión de Usuarios:**
   * **Sistema de Login Autenticado y Seguro:** Implementación de autenticación robusta (ej. JWT, OAuth 2.0).
   * **Roles de Usuario:**
     + **Administrador:** Acceso completo a todas las funcionalidades del sistema (ventas, inventario, reportes, gestión de usuarios y roles, configuración, integración APIs).
     + **Usuario Regular (Vendedor/Mesero):** Acceso limitado a las funciones de venta, toma de pedidos, visualización de inventario (sin edición).
   * **Gestión de Usuarios y Roles (Administrador):** Capacidad para crear, modificar y eliminar usuarios, y asignar/cambiar sus roles y permisos.
6. **Base de Datos Robusta:**
   * Diseño de base de datos relacional o NoSQL (según la evaluación de escalabilidad y tipo de datos) para asegurar la integridad, consistencia y rendimiento de los datos (productos, inventario, ventas, usuarios, pedidos).
7. **Integración con APIs de Domicilios (Colombia):**
   * **Rappi, Didi Food, Uber Eats:** Integración directa con las APIs de estas plataformas de delivery para:
     + Recepción automática de pedidos de delivery en el sistema POS.
     + Actualización del menú y disponibilidad de productos en las plataformas desde el POS.
     + Sincronización de estados de pedidos.
     + Consolidación de ventas de delivery en los reportes del POS.
8. **Reportes y Analíticas (para Administrador):**
   * Reportes de ventas por período, producto, empleado, método de pago.
   * Reportes de inventario y margen de ganancia.
   * Dashboard de rendimiento general del negocio (ventas diarias/mensuales, productos más vendidos, etc.), similar al diseño de inspiración.

**Tecnologías Sugeridas (Opcional, para el contexto del LLM):**

* **Backend:** Node.js (Express), Python (Django/Flask), .NET Core o Java (Spring Boot).
* **Base de Datos:** PostgreSQL, MySQL, MongoDB.
* **Frontend Web:** React, Angular, Vue.js.
* **Frontend Móvil:** React Native, Flutter, Swift/Kotlin nativo (para mayor rendimiento).
* **Contenedorización:** Docker.
* **Cloud:** AWS, Google Cloud, Azure.

**Consideraciones Adicionales:**

* **Escalabilidad:** El sistema debe ser diseñado pensando en la expansión futura y el manejo de un alto volumen de transacciones y datos.
* **Mantenibilidad:** Código limpio, modular y bien documentado.
* **Experiencia de Usuario (UX):** Interfaces intuitivas, rápidas y eficientes para minimizar la curva de aprendizaje y optimizar el flujo de trabajo del personal.
* **Rendimiento:** Tiempos de respuesta rápidos para todas las operaciones críticas.
* **Resiliencia:** Manejo de errores y tolerancia a fallos.