PLAN DE DESARROLLO: SISTEMA SAAS PARA GESTIÓN DE COMERCIOS TÉCNICOS

1. VISIÓN DEL PRODUCTO

Definición del producto

Desarrollaremos un sistema Software as a Service (SaaS) enfocado en la gestión integral de pequeños comercios de servicio técnico y venta de productos tecnológicos. La plataforma permitirá administrar el ciclo completo de operaciones, desde la gestión de clientes y ventas hasta el control de stock y servicio técnico, todo en un único ecosistema integrado y accesible vía web.

Propuesta de valor

- **Integración completa**: Unificación de todos los procesos críticos del negocio en una sola plataforma
- **Escalabilidad**: Adaptable desde micro-emprendimientos hasta pequeñas cadenas
- **Accesibilidad**: Interfaz intuitiva que no requiere conocimientos técnicos avanzados
- **Modelo SaaS**: Sin inversión inicial en infraestructura, pago por uso
- **Especialización**: Diseñado específicamente para comercios de tecnología y servicio técnico

Público objetivo

Pequeños comercios y emprendedores del sector tecnológico que:

- Venden productos electrónicos y tecnológicos
- Ofrecen servicios de reparación y mantenimiento
- Operan con personal limitado (1-10 empleados)
- Necesitan centralizar sus operaciones
- Buscan profesionalizar su gestión sin grandes inversiones

2. ARQUITECTURA Y STACK TECNOLÓGICO

Backend

- **FastAPI**: Framework Python de alto rendimiento para APIs
- **Poetry**: Gestión de dependencias y empaquetado
- **Ruff**: Linter para asegurar calidad del código
- **Pre-commit**: Hooks de git para controles de calidad automáticos
- **TestCoverage**: Herramienta para análisis de cobertura de pruebas

Frontend

- **Angular 16**: Framework robusto para aplicaciones SPA
- **PrimeNG**: Biblioteca de componentes UI rico y responsive

Base de datos

- **PostgreSQL**: Sistema de base de datos relacional
- **Modelo multitenant con esquemas compartidos y tenant ID**: Eficiencia y simplicidad en la gestión de datos

Infraestructura

- **Heroku**: Despliegue principal para aplicación (alternativa a Railway)
- **Supabase**: Servicio para base de datos PostgreSQL gestionada
- **GitHub**: Repositorio de código y gestión de CI/CD
- **Dev Containers**: Entorno de desarrollo estandarizado

3. CONSIDERACIONES TÉCNICAS CLAVE

Modelo de datos multitenant con esquema compartido

El enfoque de esquema compartido con discriminador de tenant (tenant ID) ofrece:

- **Ventajas**:
  - Mayor eficiencia en el uso de recursos de base de datos
  - Simplificación en actualizaciones de esquema (una sola operación)
  - Mejor rendimiento para operaciones de consulta entre tenants
- **Desafíos**:
  - Necesidad de implementar control de acceso riguroso a nivel de aplicación
  - Diseño cuidadoso para prevenir filtración de datos entre tenants
  - Estrategias específicas para consultas que atraviesan datos de múltiples tenants

Estrategia de migraciones

Implementación de un sistema robusto que:

- Coordine migraciones a través de múltiples esquemas
- Permita rollbacks selectivos por tenant
- Gestione versiones de esquema diferentes por cliente

Optimización de rendimiento

- Implementación de caché por tenant

- Conexiones pooling inteligente
- Monitoreo específico de rendimiento por esquema
- Estrategia para "tenants calientes" vs "tenants fríos"

Seguridad

- Autenticación multi-factor
- Aislamiento completo de datos entre tenants
- Cifrado a nivel de base de datos
- Auditoría de acciones por usuario y tenant

4. MÓDULOS FUNCIONALES

Gestor de usuarios

- Jerarquía de roles (administrador, vendedor, técnico, etc.)
- Permisos granulares por funcionalidad
- Auditoría de actividades
- Auto-registro limitado con aprobación

Gestor de cajas

- Apertura y cierre de cajas con balance
- Múltiples cajas simultáneas
- Gestión de diferentes medios de pago
- Arqueo de caja y reportes

Gestor de clientes

- Perfil completo con historial de interacciones
- Segmentación y categorización
- Integración con comunicaciones (correo, SMS)
- Gestión de crédito y confianza

Gestor de cuentas corrientes

- Estado de cuenta por cliente
- Historial de movimientos
- Configuración de límites de crédito
- Notificaciones automáticas de vencimientos

Gestor de stock

- Control de inventario en tiempo real
- Múltiples depósitos
- Alertas de stock mínimo
- Trazabilidad de productos (serial, lote)

Gestor de servicio técnico

- Seguimiento de órdenes de servicio
- Estados personalizables del proceso
- Presupuestos con aprobación
- Historial técnico por dispositivo

Gestor de ventas

- Proceso completo desde cotización hasta factura
- Descuentos y promociones
- Integración con stock y caja
- Reportes de rendimiento de ventas

5. DEFINICIÓN DEL MVP (PRODUCTO MÍNIMO VIABLE)

Alcance del MVP

Incluirá las funcionalidades esenciales para la operativa básica de un comercio:

1. **Gestor de usuarios**:
   - Roles básicos (admin, vendedor, técnico)
   - Autenticación segura
2. **Gestor de clientes**:
   - Registro de datos básicos
   - Historial simplificado
3. **Gestor de stock**:
   - Alta, baja y modificación de productos
   - Control básico de inventario
4. **Gestor de servicio técnico**:
   - Registro de órdenes de servicio
   - Seguimiento de estado (recibido, en proceso, terminado)
5. **Gestor de ventas**:

- Proceso de venta simplificado
- Generación de comprobantes básicos

6. **Gestor de cuentas corrientes simple**:
   - Registro básico de movimientos por cliente
   - Consulta de saldo actual
   - Registro de pagos a cuenta

Elementos diferidos para post-MVP

- Cuentas corrientes avanzadas (múltiples condiciones de pago, intereses)
- Múltiples cajas y depósitos
- Reportes complejos
- Integraciones con sistemas externos

6. ROADMAP DE DESARROLLO

Fase 1: Preparación y fundamentos (1-2 meses)

- Configuración del entorno de desarrollo con Dev Containers
- Definición detallada de arquitectura y modelo de datos
- Implementación de sistema base multitenant
- Desarrollo de sistema de autenticación y autorización
- Configuración de CI/CD en GitHub

Fase 2: Desarrollo del MVP (3-4 meses)

- Implementación incremental de módulos del MVP:
  1. Gestor de usuarios (2 semanas)
  2. Gestor de clientes (3 semanas)
  3. Gestor de stock (3 semanas)
  4. Gestor de servicio técnico (4 semanas)
  5. Gestor de ventas simple (4 semanas)
  6. Gestor de cuentas corrientes simple (3 semanas)
- Pruebas de integración continua
- Validaciones con usuarios potenciales

Fase 3: Refinamiento y lanzamiento del MVP (1-2 meses)

- Ajustes basados en feedback de pruebas
- Optimización de rendimiento

- Implementación de monitoreo y alertas
- Documentación de usuario
- Preparación de onboarding
- Lanzamiento controlado con clientes beta

Fase 4: Expansión post-MVP (4-6 meses)

- Desarrollo de gestor de cajas completo
- Implementación de cuentas corrientes avanzadas
- Mejoras en reportes e inteligencia de negocio
- Implementación de gestión de múltiples sucursales
- Expansión de integraciones (fiscal, pagos online)

Fase 5: Producto completo y escalamiento (6+ meses)

- Desarrollo de módulos avanzados:
  - Fidelización de clientes
  - Gestión de compras y proveedores
  - Módulo de marketing integrado
  - Planificación de recursos
- Mejoras continuas basadas en métricas de uso
- Escalabilidad para mayor número de usuarios

7. CONSIDERACIONES ESTRATÉGICAS

Pruebas y calidad

- **Enfoque de TDD**: Pruebas desde el inicio del desarrollo
- **TestCoverage**: Seguimiento riguroso de la cobertura de código con metas de >80%
- **Pruebas automáticas**: Unitarias, integración, e2e
- **Control de calidad**: Pre-commit hooks, CI/CD, revisión de código
- **Revisiones de calidad**: Sesiones periódicas de revisión de métricas de calidad

Experiencia de usuario

- Diseño centrado en el flujo de trabajo real de comercios
- Optimización para dispositivos móviles y tablets
- Tiempos de respuesta optimizados
- Offline capabilities para operaciones críticas

Escalabilidad

- Arquitectura preparada para crecimiento horizontal
- Monitoreo de rendimiento por tenant
- Estrategia para gestionar "tenants problemáticos"
- Plan de escalado para DB cuando aumente el número de esquemas

Consideraciones legales

- Cumplimiento de GDPR/LGPD/regulaciones locales
- Acuerdos de nivel de servicio (SLA) claros
- Gestión de responsabilidad sobre datos fiscales
- Políticas de retención y eliminación de datos

8. MÉTRICAS DE ÉXITO

Métricas técnicas

- Tiempo de respuesta < 200ms para operaciones comunes
- Disponibilidad > 99.9%
- Cobertura de pruebas > 85%
- Tiempo de recuperación ante fallos < 10 minutos

Métricas de negocio

- Tasa de conversión de pruebas gratuitas
- Retención de clientes > 90% anual
- NPS > 40
- Crecimiento mensual de usuarios activos

9. RIESGOS Y MITIGACIONES

Riesgos técnicos

- **Escalabilidad del modelo multitenant**: Implementar monitoreo temprano y plan de migración a arquitectura híbrida si necesario
- **Complejidad de actualizaciones**: Desarrollar sistema automatizado de migraciones por lotes
- **Rendimiento degradado**: Implementar benchmarks continuos por tenant

Riesgos de negocio

- **Adopción lenta**: Programa de onboarding asistido y migración desde sistemas legacy
- **Competencia**: Enfoque en verticales específicas dentro del nicho de servicio técnico

- **Rentabilidad**: Estructura de costos clara con monitoreo de uso por tenant

10. CONCLUSIONES Y PRÓXIMOS PASOS

Este plan establece las bases para el desarrollo incremental de un SaaS especializado para comercios de servicio técnico y venta de productos tecnológicos. El enfoque en un MVP claramente definido permitirá validar el concepto rápidamente, mientras que la arquitectura seleccionada brinda flexibilidad para crecer y adaptarse a las necesidades del mercado.

Los próximos pasos inmediatos incluyen:

1. Establecer el entorno de desarrollo con Dev Containers
2. Definir la estructura detallada de la base de datos multitenant
3. Implementar el sistema base de autenticación y gestión de tenants
4. Comenzar el desarrollo del primer módulo del MVP: Gestor de usuarios

Con un enfoque ágil e iterativo, el sistema podrá evolucionar basándose en feedback real, manteniendo el foco en resolver los problemas específicos de los pequeños comercios del sector tecnológico.