Prueba Técnica - SCL Consulting

Introducción

Esta prueba técnica está diseñada para evaluar sus habilidades en el desarrollo de aplicaciones web modernas utilizando Python como lenguaje principal. El proyecto consiste en crear una aplicación web completa para la gestión de inventario de una tienda de productos electrónicos, implementando una arquitectura que separa el frontend (Django) del backend (FastAPI), con una base de datos PostgreSQL alojada en Supabase y el despliegue de la aplicación en Railway.
La aplicación permitirá a los usuarios autenticarse mediante un sistema de login tradicional y posteriormente utilizar tokens JWT para las comunicaciones con la API.

Descripción del Negocio

La aplicación estará enfocada en la gestión de inventario de una **tienda de productos electrónicos** que comercializa dispositivos como smartphones, tablets, computadoras, accesorios y componentes electrónicos. Este tipo de negocio requiere un control preciso del inventario debido a que:

- Los productos suelen tener un valor elevado
- El ciclo de vida de los productos es relativamente corto
- Es necesario mantener un registro detallado de números de serie y modelos
- La rotación de stock debe ser monitoreada constantemente para evitar obsolescencia

Requisitos Técnicos

Tecnologías Requeridas

- **Frontend**: Django (templates, formularios, vistas)
- **Backend**: FastAPI (APIs RESTful)
- **Base de datos**: PostgreSQL (alojada en Supabase)
- **Autenticación**: Sistema de login con usuario/contraseña + JWT para APIs
- **Despliegue**: Railway para la aplicación web y las APIs

Requisitos Funcionales

1. Sistema de Autenticación

- Implementar un formulario de login en Django que permita a los usuarios autenticarse con usuario y contraseña
- Crear endpoints en FastAPI para la autenticación y generación de tokens JWT
- Configurar middleware para validar tokens JWT en todas las solicitudes a la API
- Implementar cierre de sesión y renovación de tokens

2. Gestión de Inventario

- Pantalla principal con listado de productos en inventario
- Funcionalidades CRUD (Crear, Leer, Actualizar, Eliminar) para los productos
- Filtros y búsqueda de productos por diferentes criterios
- Alertas de stock bajo y productos próximos a agotarse

3. Modelo de Datos

El sistema debe gestionar al menos las siguientes entidades:

- **Productos**: nombre, descripción, categoría, precio, stock actual, stock mínimo, código/SKU, número de serie
- **Categorías**: nombre, descripción
- **Proveedores**: nombre, contacto, dirección
- **Movimientos de Inventario**: entradas, salidas, ajustes, fecha, responsable
- **Usuarios**: información de autenticación y permisos

4. API RESTful con FastAPI

- Endpoints para todas las operaciones CRUD de productos
- Endpoints para consultar estadísticas de inventario
- Documentación automática con Swagger/OpenAPI
- Validación de datos con Pydantic
- Manejo adecuado de errores y excepciones

Ejemplo de Inventario

A continuación, se presenta un ejemplo de cómo podría estructurarse el inventario para la tienda de electrónicos:

Categorías de Productos

- Smartphones
- Tablets
- Laptops
- Componentes (placas base, procesadores, memoria RAM)
- Accesorios (cables, cargadores, fundas)
- Periféricos (teclados, ratones, monitores)

Ejemplo de Registro de Producto

Producto: Smartphone XYZ Pro

- **Código**: SM-XYZ-PRO-128
- **Descripción**: Smartphone de gama alta con 128GB de almacenamiento
- **Categoría**: Smartphones
- **Proveedor**: Electro Distribuciones S.A.
- **Precio de compra**: $450
- **Precio de venta**: $699
- **Stock actual**: 15 unidades
- **Stock mínimo**: 5 unidades
- **Ubicación en almacén**: Estante A3
- **Números de serie**: Lista de IMEI de cada unidad
- **Fecha de última entrada**: 15/04/2025
- **Observaciones**: Alta demanda, considerar aumentar stock mínimo

Requisitos de Despliegue

Base de Datos (Supabase)

- Crear proyecto en Supabase
- Configurar tablas y relaciones según el modelo de datos
- Configurar políticas de seguridad adecuadas
- Proporcionar credenciales de conexión para la aplicación

Aplicación (Railway)

- Desplegar frontend Django en Railway

- Desplegar backend FastAPI en Railway
- Configurar variables de entorno necesarias
- Asegurar la comunicación entre los servicios

Entregables

1. Repositorio Git con el código fuente completo
2. Documentación detallada de la API (generada automáticamente con Swagger)
3. Instrucciones para la instalación y ejecución local del proyecto
4. URLs de acceso a la aplicación desplegada
5. Credenciales de prueba para acceder al sistema

Criterios de Evaluación

- **Funcionalidad**: La aplicación cumple con todos los requisitos especificados
- **Calidad del código**: Estructura, legibilidad, mantenibilidad
- **Seguridad**: Implementación correcta de autenticación y autorización
- **Rendimiento**: Optimización de consultas a la base de datos
- **Usabilidad**: Interfaz intuitiva y responsiva
- **Documentación**: Claridad y completitud de la documentación

Bonus (Puntos Adicionales)

- Implementación de tests automatizados (unitarios y de integración)
- Configuración de CI/CD para despliegue automático
- Implementación de notificaciones por correo electrónico para alertas de stock
- Diseño responsive para acceso desde dispositivos móviles
- Implementación de gráficos y dashboards para visualización de datos
- **Integración con RAG (OpenAI)**: Implementar un endpoint que permita consultas en lenguaje natural sobre procedimientos de gestión de inventario, información técnica de productos, políticas de la empresa y mejores prácticas (opcional)

Plazo de Entrega

El candidato dispondrá de 7 días naturales para completar la prueba técnica desde el momento en que reciba las instrucciones.

Recursos Útiles

- Documentación oficial de Django
- Documentación oficial de FastAPI
- Guía de Supabase para PostgreSQL
- Documentación de Railway para despliegue

¡Buena suerte con la prueba técnica! Estamos ansiosos por ver su implementación y discutir su enfoque durante la entrevista.

SCL Consulting

Departamento de Tecnología