

Informe de Desarrollo Guiado por IA: Refactorización y Escalamiento

Proyecto: Sistema de Procesamiento de Ventas

Herramienta IA: Claude Code (Simulado) / Gemini

Objetivo: Transformar un script legacy en una aplicación de producción mantenible y extensible.

1. Línea de Tiempo de Versiones

● Versión 1: Código Legacy (**script_ventas_v1.py**)

Estado Inicial: Script funcional pero frágil.

El código original realizaba la tarea básica (leer CSV, calcular IVA, guardar JSON), pero presentaba deuda técnica crítica.

- **Deficiencias Detectadas:**
 - **Gestión de Recursos:** Uso de `open()/close()` manual, alto riesgo de dejar archivos abiertos si ocurría un error.
 - **Estructura Monolítica:** Todo el código vivía en el ámbito global (sin funciones), haciendo imposible las pruebas unitarias.
 - **Valores "Hardcoded":** El IVA (1.19) y los nombres de archivo estaban incrustados en la lógica.
 - **Ausencia de Tipado:** Imposible saber qué datos se esperaban sin leer todo el código.

● Versión 2: Refactorización Estructural (**script_ventas_v2.py**)

Intervención de la IA: Limpieza, seguridad y estandarización.

Se solicitó a la IA actuar como Ingeniero de Software para aplicar principios SOLID y buenas prácticas de Python moderno.

- **Cambios Implementados:**
 - **Programación Modular:** Se dividió la lógica en funciones con responsabilidad única (`leer_csv`, `procesar_fila`, `guardar_json`).
 - **Seguridad de Tipos:** Implementación de *Type Hints* (`List`, `Dict`, `Path`) para mejorar la legibilidad y el debug.
 - **Manejo de Errores:** Inclusión de bloques `try-except` y sistema de `logging` para no romper la ejecución ante datos corruptos.
 - **Configuración Centralizada:** Uso de `dataclass` (`ConfigReporte`) para manejar parámetros y rutas, eliminando valores mágicos.
 - **Manejo de Rutas:** Implementación de librería `pathlib` para compatibilidad entre sistemas operativos.

● Versión 3: Módulo de Inteligencia de Negocio (`script_ventas_v3.py`)

Intervención de la IA: Creación de nueva funcionalidad (Feature Request).

Se solicitó un módulo adicional que no existía en el alcance original para aportar valor analítico a los datos.

- **Nuevo Módulo IA Incorporado: Análisis Estadístico.**
 - Se creó la clase `EstadisticasVentas` para encapsular métricas.
 - **Funcionalidad Agregada:** Cálculo automático de KPIs (Ticket promedio, venta máxima/mínima, total recaudado).
 - **Visualización:** Generación de un reporte visual en formato tabla ASCII para la terminal (`generar_resumen_texto`).
 - **Persistencia:** Las estadísticas se guardan junto con los datos crudos en el JSON final.

2. Cuadro Comparativo de Mejoras

| Característica | Versión 1 (Original) | Versión 3 (Final IA) | Impacto |
|----------------------|--------------------------------------|---|---|
| Legibilidad | Código espagueti, difícil de seguir. | Estructurado, documentado (Docstrings). | Alto: Facilita el onboarding de nuevos devs. |
| Robustez | Se rompe con una fila vacía o texto. | Ignora filas corruptas y loguea el error. | Crítico: Apto para producción. |
| Seguridad | Archivos pueden quedar bloqueados. | Context Managers (<code>with</code>) garantizan cierre. | Medio: Previene fugas de memoria. |
| Escalabilidad | Difícil de agregar cambios. | Fácil de extender (ej. cambiar IVA, rutas). | Alto: Código preparado para crecer. |

| | | | |
|--------------|------------------------|---|---|
| Valor | Solo transforma datos. | Genera conocimiento (Estadísticas/KPIs). | Alto: Convierte datos en información útil. |
|--------------|------------------------|---|---|

3. Conclusión Técnica

El proceso de desarrollo guiado demostró cómo la IA puede elevar la calidad del código de un nivel "principiante" a "profesional" en iteraciones rápidas. No solo se corrigieron errores sintácticos y de seguridad, sino que la IA propuso una arquitectura orientada a objetos (**Dataclasses**) y generó lógica de negocio adicional (Módulo de Estadísticas) que enriqueció el producto final sin intervención manual compleja.