Requerimientos Funcionales (RF)

RF-01: Importación de Datos

- El sistema deberá permitir la carga de facturas desde un archivo en formato JSON.

RF-02: Validación de Datos

- El sistema deberá validar que el invoice_number de cada factura importada sea único.
- El sistema deberá validar la consistencia monetaria de cada factura, comparando el total_amount con la suma de los subtotales de sus productos.

RF-03: Clasificación de Datos

- El sistema deberá clasificar las facturas como Consistent o Inconsistent basándose en la validación monetaria.
- El sistema deberá calcular y asignar un Estado de Factura (Issued, Partial, Cancelled) a cada factura basándose en sus notas de crédito.
- El sistema deberá calcular y asignar un Estado de Pago (Paid, Overdue, Pending) a cada factura basándose en sus fechas.

RF-04: Gestión de Facturas

- El sistema deberá mostrar una lista de todas las facturas Consistent.
- El sistema deberá permitir la búsqueda de facturas por su invoice_number.
- El sistema deberá permitir el filtrado de facturas por Estado de Factura y Estado de Pago.
- El sistema deberá mostrar una vista con el detalle completo de una factura seleccionada.

RF-05: Gestión de Notas de Crédito

- El sistema deberá permitir agregar notas de crédito a una factura existente.
- El sistema deberá validar que el monto de una nueva nota de crédito no exceda el saldo pendiente de la factura.

RF-06: Generación de Reportes

- El sistema deberá generar y mostrar en pantalla los siguientes tres reportes:
    - Un listado de facturas consistentes con más de 30 días de vencimiento.
    - Un resumen (total y porcentual) de facturas por estado de pago.
    - Un listado de facturas inconsistentes, detallando la discrepancia de montos.

- El sistema deberá permitir la exportación de los datos de los reportes en formato CSV.

RF-07: Registro de Errores

- El sistema deberá registrar en un log los errores de importación, como los invoice_number duplicados.

Requerimientos No Funcionales (RNF)

RNF-01: Pila Tecnológica (Stack)

- Backend: La API deberá ser desarrollada en .NET 8.
- Frontend: La interfaz de usuario deberá ser desarrollada con React.
- Base de Datos: El sistema deberá utilizar SQLite como motor de base de datos.
- ORM: El acceso a datos deberá gestionarse a través de Entity Framework Core.

RNF-02: Arquitectura y Diseño

- La API deberá exponerse como una API REST.
- La solución deberá implementar un patrón de diseño reconocido (se ha elegido Arquitectura en Capas con Patrón Repositorio).
- La API deberá utilizar los códigos de estado HTTP apropiados para las respuestas (ej. 200, 201, 400, 404).

RNF-03: Documentación

- La API deberá contar con documentación generada a través de Swagger (OpenAPI).
- El proyecto deberá entregarse con un archivo `README.md` que contenga instrucciones claras de instalación y ejecución.
- El README.md deberá mencionar y explicar el patrón de diseño utilizado.

RNF-04: Usabilidad y Rendimiento

- La interfaz de usuario deberá ser intuitiva.
- Los formularios deberán incluir validaciones para guiar al usuario.
- El sistema deberá responder a las interacciones del usuario (búsquedas, filtros, carga de reportes) de manera fluida y sin demoras perceptibles.