**MODELO TECNICO LISTADOR DE ARCHIVOS**

**10/2025**

Índice

[1. Visión General 1](#_Toc212549219)

[2. Características Principales 2](#_Toc212549220)

[3. Instalación y Ejecución 2](#_Toc212549221)

[3.1. Opción A (Recomendada): Usar el Ejecutable (.exe) 2](#_Toc212549222)

[3.2. Opción B: Ejecutar desde el Código Fuente 3](#_Toc212549223)

[4. Estructura de Carpetas 3](#_Toc212549224)

[4.1. Estructura de Entrada 3](#_Toc212549225)

[4.2. Estructura de Salida 4](#_Toc212549226)

[5. Flujo de Ejecución 4](#_Toc212549227)

[6. Archivos de Salida Detallados 5](#_Toc212549228)

[6.2. resumen\_completo.xlsx 5](#_Toc212549229)

[6.2. detalle\_nulos.csv 6](#_Toc212549230)

[7. Diccionario 6](#_Toc212549231)

[7.1. Componentes Principales 6](#_Toc212549232)

[7.2. Estructura y Archivos (Entrada) 7](#_Toc212549233)

[7.3. Estructura y Archivos (Salida) 7](#_Toc212549234)

[7.4. Instalación y Ejecución 7](#_Toc212549235)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Controles mínimos aplicados a GSTI** | | | | |
| Solicitante: | | | | Felipe Acuña |
|  | | |  | |
| Versión: | 1.0 | Fecha: 24-10-2025 | | Ejecutor: Dariac Saldias |
| Versión: | 2.0 | Fecha: 28-10-2025 | | Ejecutor: Dariac Saldias |
|  | | | | |
| Detalle: | 1.0 | Estructura principal e información de la documentación | | |
| Detalle: | 2.0 | Ajuste de estructura profesional de documentación | | |

# **1. Visión General**

El **Listador de Archivos Mensuales** es una herramienta de escritorio desarrollada en Python con una interfaz gráfica (GUI) que permite inventariar y consolidar la información de archivos distribuidos en una estructura de carpetas mensuales.

El propósito principal de la aplicación es escanear una carpeta base que contiene subcarpetas con formato MMYYYY, procesar todos los archivos que se encuentren dentro (incluyendo los comprimidos) y generar un **único reporte maestro en Excel** con el listado completo de los archivos, su cantidad de filas, tamaño y ubicación. Adicionalmente, genera un informe con los archivos que no pudieron ser procesados, detallando el error. A diferencia de otras herramientas de análisis, su objetivo es la **consolidación de un inventario**, no la comparación de tendencias entre períodos.

# **2. Características Principales**

* **Inventario Centralizado:** Genera un único archivo Excel (resumen\_completo.xlsx) que actúa como un inventario maestro de todos los archivos encontrados en todas las carpetas mensuales.
* **Interfaz Gráfica Sencilla:** Permite al usuario seleccionar la carpeta de trabajo, iniciar el proceso y monitorear el progreso general y por carpeta en tiempo real.
* **Manejo Automático de Archivos Comprimidos:** Soporta la descompresión automática de formatos .zip, .tar, .gz, .bz2, .7z, permitiendo analizar el contenido de forma transparente.
* **Soporte de Múltiples Formatos de Datos:** Capaz de leer y contar las filas de archivos .xlsx, .xls, .csv, .json, .parquet, .txt y .html.
* **Reporte de Errores:** Crea un archivo detalle\_nulos.csv que lista todos los archivos que no pudieron ser leídos, especificando la razón del fallo, lo cual facilita la depuración y corrección de los datos de origen.
* **Procesamiento Eficiente:** Utiliza un modo de lectura optimizado para archivos grandes, evitando un alto consumo de memoria RAM al solo contar las filas en lugar de cargar el archivo completo.
* **Gestión de Recursos:** Incluye un mecanismo para manejar errores de "espacio insuficiente en disco" durante la descompresión, guiando al usuario para liberar espacio y continuar.
* **No Destructivo:** La aplicación no modifica ni elimina los archivos originales. Tampoco interrumpe al usuario por archivos de reportes previos; simplemente los sobrescribirá al finalizar la nueva ejecución.

# **3. Instalación y Ejecución**

Existen dos métodos para ejecutar la aplicación.

## **3.1. Opción A (Recomendada): Usar el Ejecutable (.exe)**

Se proporciona un archivo ejecutable (.exe) para máxima simplicidad.

* **Ventajas:**
  + No requiere instalar Python ni ninguna biblioteca.
  + Totalmente portable: solo necesitas el archivo para ejecutarlo en cualquier PC con Windows.
* **Instrucciones:**
  1. Ubica el archivo ejecutable (lista.exe).
  2. **Haz doble clic** sobre él para abrir la aplicación.
  3. (Opcional). Para más fácil acceso:
* Click derecho en lista.exe
* Crear acceso directo. (Mostrar más opciones en Windows 11)
* Mover este acceso directo al escritorio (No borrar ni manipular la carpeta principal).

## **3.2. Opción B: Ejecutar desde el Código Fuente**

Este método es para usuarios con un entorno de Python configurado.

* **Requisitos:**
  + **Python 3.x** instalado.
  + Bibliotecas de Python: pandas, openpyxl, xlrd, y opcionalmente py7zr (solo para archivos .7z).
* Instalación de Dependencias:

Abre una terminal o símbolo del sistema y ejecuta:

pip install -r requirements.txt

* Ejecución:

Navega hasta el directorio del script y ejecútalo.

# **4. Estructura de Carpetas**

## **4.1. Estructura de Entrada**

La herramienta espera una carpeta base que contenga subcarpetas por cada mes, usando el formato MMYYYY.

/Carpeta\_de\_Trabajo/

├── 012025/

│ ├── reporte\_ventas.csv

│ ├── imagenes.zip

│ └── subdirectorio/

│ └── datos\_adicionales.xlsx

├── 022025/

│ ├── reporte\_ventas\_actualizado.csv

│ └── ...

└── 032025/

├── reporte\_ventas.csv

└── ...

## **4.2. Estructura de Salida**

Al finalizar el proceso, la aplicación crea los siguientes archivos directamente dentro de la **carpeta base** (/Carpeta\_de\_Trabajo/):

1. resumen\_completo.xlsx: El inventario maestro con todos los archivos procesados.
2. resumen\_completo.csv: El mismo inventario en formato CSV.
3. detalle\_nulos.csv: El reporte de errores y archivos no leídos.
4. process.log: Un archivo de registro técnico para seguimiento y depuración.

A diferencia de la versión de análisis, **no se crean** carpetas resumen\_generado dentro de los directorios de cada mes.

# **5. Flujo de Ejecución**

1. **Inicio y Selección:** El usuario inicia la aplicación y selecciona la carpeta base de trabajo.
2. **Inicio del Proceso:** Al hacer clic en "Iniciar procesamiento", se inicia un hilo de trabajo en segundo plano para no congelar la interfaz.
3. **Identificación de Meses:** El script localiza todas las subcarpetas MMYYYY y las ordena cronológicamente.
4. **Iteración por Mes:** La aplicación procesa cada carpeta mensual de forma secuencial.
5. **Expansión y Listado de Archivos:** Dentro de cada mes, se buscan todos los archivos. Si se encuentra un archivo comprimido, se extrae su contenido a una carpeta temporal en el Escritorio para su análisis.
6. **Lectura y Recolección de Datos:**
   * Se intenta leer cada archivo individualmente.
   * **Si la lectura es exitosa:** Se obtiene la cantidad de filas, el tamaño en KB y la ruta. Esta información se añade a una lista principal en memoria.
   * **Si la lectura falla:** El nombre del archivo, su ruta y el mensaje de error se añaden a una lista de errores separada.
7. **Actualización de Progreso:** Las barras de progreso en la interfaz se actualizan en tiempo real para reflejar el avance.
8. **Guardado de Resultados:** Una vez que se han procesado todas las carpetas:
   * La lista principal de datos se convierte en un DataFrame de Pandas y se guarda como resumen\_completo.xlsx (con formato) y resumen\_completo.csv.
   * La lista de errores se guarda como detalle\_nulos.csv.
9. **Limpieza Final:** Se elimina la carpeta temporal creada en el Escritorio.
10. **Finalización:** La interfaz notifica al usuario que el proceso ha terminado.

# **6. Archivos de Salida Detallados**

## **6.2. resumen\_completo.xlsx**

Este archivo contiene el inventario consolidado de todos los archivos procesados exitosamente.

| **Columna** | **Descripción** |
| --- | --- |
| **Mes** | El mes y año de la carpeta contenedora, en formato MM/YYYY. |
| **Archivo** | El nombre del archivo procesado. |
| **Cantidad filas** | El número total de filas con datos encontradas en el archivo. |
| **Tamaño KB** | El tamaño del archivo en Kilobytes, redondeado a dos decimales. |
| **Ruta completa** | La ruta absoluta en el disco donde se encuentra el archivo (puede ser una ruta temporal si fue extraído). |

## **6.2. detalle\_nulos.csv**

Este archivo es crucial para la auditoría, ya que lista todos los archivos que la aplicación no pudo procesar.

| **Columna** | **Descripción** |
| --- | --- |
| **Mes** | El mes y año de la carpeta contenedora (MM/YYYY). |
| **Archivo** | El nombre del archivo que generó el error. |
| **Ruta completa** | La ubicación exacta del archivo problemático. |
| **Error** | El mensaje de error específico devuelto por Python. |

# **7. Diccionario**

## 7.1. **Componentes Principales**

* **Listador de Archivos Mensuales:** Herramienta de escritorio (Python/Tkinter) para inventariar y consolidar listados de archivos desde una estructura de carpetas MMYYYY.
* **Inventario Centralizado:** Característica principal que consolida todos los archivos encontrados en un único reporte maestro (resumen\_completo.xlsx).
* **Reporte de Errores:** Función que genera un informe (detalle\_nulos.csv) listando los archivos que no pudieron ser leídos y la causa del error.

## 7.2. **Estructura y Archivos (Entrada)**

* **Carpeta Base / Carpeta de Trabajo:** Directorio principal seleccionado por el usuario que contiene las subcarpetas mensuales a procesar.
* **MMYYYY:** Formato requerido para las subcarpetas que representan cada mes (ej. "012025").

## 7.3. **Estructura y Archivos (Salida)**

* **resumen\_completo.xlsx:** Archivo Excel maestro generado en la carpeta base, que contiene el inventario de todos los archivos procesados, sus filas, tamaño y ubicación.
* **resumen\_completo.csv:** Versión en formato CSV del inventario maestro.
* **detalle\_nulos.csv:** Reporte generado en la carpeta base que lista todos los archivos que fallaron durante la lectura, detallando el error.
* **process.log:** Archivo de registro técnico (log) utilizado para depuración.

## 7.4. **Instalación y Ejecución**

* **lista.exe:** El archivo ejecutable (Opción A) que empaqueta la aplicación y sus dependencias, permitiendo su uso sin instalar Python.
* **requirements.txt:** Archivo de texto que lista las dependencias de Python (pandas, openpyxl, xlrd, py7zr) necesarias para ejecutar la aplicación desde el código fuente (Opción B).