Automatización de Gestión Documental de RRHH

Proyecto: Generador Automático de Boletas de Vacaciones (Bureau Veritas & Inspectorate)

Rol: Desarrollador de Automatización / Asistente Técnico

Tecnologías: Python, Pandas, Jinja2, WeasyPrint, Tkinter, PyInstaller.

1. El Desafío (Problem Statement)

En el área administrativa, la gestión de vacaciones para un equipo de más de **50 empleados** se realizaba mediante un proceso manual. El flujo de trabajo anterior consistía en:

- Copiar datos individuales de una lista.
- Pegar datos en una plantilla de Excel.
- Guardar manualmente como PDF.
- Repetir el proceso para cada trabajador (lotes de 15 a 50 personas).

Puntos de dolor: Este proceso era propenso a errores humanos (copiar/pegar mal), tedioso y consumía tiempo que podían dedicarse a tareas analíticas. Además, el manejo de dos razones sociales distintas (Bureau Veritas e Inspectorate) complicaba la gestión de archivos.

2. La Solución Técnica

Desarrollé una aplicación de escritorio en **Python** que automatiza el ciclo completo de generación documental. El sistema toma un archivo Excel masivo como *input* y genera certificados individuales en PDF y organizados por división corporativa.

Arquitectura del Software

El código se estructura en tres capas principales:

1. **Ingesta de Datos (pandas):** Lectura y limpieza del dataset desde Excel.
2. **Motor de Renderizado (jinja2 + weasyprint):**
    - Utilicé **HTML/CSS** como base para las plantillas, permitiendo un diseño pixel-perfect mucho más flexible que Excel.
    - La librería Jinja2 inyecta los datos dinámicos (Nombres, Fechas, Cargos) en el HTML.
    - WeasyPrint convierte el HTML renderizado en un PDF de alta calidad listo para impresión o firma.
3. **Interfaz de Usuario (tkinter + threading):**
    - Diseñé una GUI amigable para que el usuario seleccione el archivo.
    - Implementé multithreading para ejecutar la lógica pesada en segundo plano, manteniendo la barra de progreso fluida y evitando que la aplicación se "congele" durante el procesamiento masivo.

3. Innovación: Accesibilidad y "Bus Factor".

Un requisito crítico fue la sostenibilidad del proyecto tras mi salida de la empresa. El equipo de RRHH no poseía conocimientos de programación para ejecutar scripts de Python (.py).

Solución Implementada:

- Empaqueté la aplicación como un ejecutable **portable (.exe)** utilizando **PyInstaller**.
- Implementé una función de manejo de rutas (resource_path) que permite al ejecutable acceder a sus recursos internos (imágenes, plantillas HTML) sin depender de carpetas externas.
- **Resultado:** El equipo ahora utiliza una herramienta "doble clic", sin necesidad de instalar Python ni librerías, garantizando la continuidad operativa de la mejora.

4. Diagrama de Flujo (Input/Output)

Etapa	**Descripción Técnica**
Input	Archivo Excel (.xlsx) con columnas estandarizadas: *Nombres, Cargo, División (BV/INS), Fechas Inicio/Fin/Retorno*.
Procesamiento	Iteración de filas -> Cálculo de lógica de negocio (asignación de fechas, selección de logo BV vs INS) -> Renderizado HTML.
Output	Carpeta raíz con subcarpetas automáticas /BV y /INS conteniendo archivos PDF nombrados: Vacaciones - [Nombre_Apellido].pdf.

5. Resultados e Impacto

- **Eficiencia:** Reducción del tiempo de procesamiento de **~3 horas** (manual) a **menos de 2 minutos** (automático) para lotes de 50 trabajadores.
- **Precisión:** Eliminación del 100% de errores de tipeo o asignación equivocada de logos/fechas.
- **Escalabilidad:** La herramienta es capaz de procesar cientos de registros sin costo adicional de tiempo.
- **Adopción:** La herramienta fue entregada como un ejecutable independiente, permitiendo su uso inmediato por cualquier miembro del equipo administrativo sin dependencia técnica.