Automatización de procesos de gestión contable en la Electrificadora de Santander S.A. (ESSA) a través de herramientas de Microsoft y Scripts de Python.

Julián David León Quintero

Trabajo de Grado para optar al título de Ingeniero de Sistemas

Director

Andrés Leonardo González Gómez

PhD (c). Ciencias de la Computación

Tutor

Román Alexis Suárez Caliz

MSc. Ingeniería de sistemas e informática

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingeniería de Sistemas e Informática

Ingeniería de Sistemas

Bucaramanga

2024

Dedicatoria

Primeramente quiero dedicar esto a Dios, cuya presencia en todo mi proceso ha sido incondicional, es quien conoce todo mi proceso y a quien puedo volver siempre sabiendo que estará para mi, sin él nada de esto sería una realidad.

Quiero dedicar este trabajo a cada persona que de alguna manera me aportó en mi desarrollo personal y profesional, sin duda alguna a mi madre Janeth Quintero Rojas, cuyo pilar principal representa en mi vida, si mi carrera educativa, profesional y personal fueran una mesa, sería de una sola pata, y esta sería representada por ella. A mi padre Pompilio León Rodriguez por otro lado de quien también estoy profundamente agradecido, por construir desde el principio de todo hasta la actualidad, un lugar donde situar esta mesa que es mi persona.

Quiero hacer una dedicatoria y agradecimiento especial a mi hermano Fabián Camilo León Quintero, pues fue, es y siempre será el espejo al que me he aferrado, es sin duda alguna mi héroe, mi figura de alguien que quiero ser, y de quien estoy muy orgulloso por todo lo que ha logrado sin tener alguien a quien seguir. Un caminante para hacer caminos.

Quiero dedicar también a dos personas que cumplieron un papel importante en toda mi vida aunque no tienen la noción de esto, a mi abuelo Jaime Quintero Soler y a mi primo Diego Andrés Vargas Quintero, cuya figuras me inspiraron a ser quien soy, a esforzarme de la manera en la que lo hago, y de mantener una sonrisa siempre.

Quiero dedicar a toda mi familia, cada uno ha hecho parte de esto, mi tío Oscar que me preparó para enfrentarme a entornos empresariales, mi tía Smith que siempre me motivó a seguir adelante, mi

abuela Elvira, que siempre me se ha preocupado por mi, mi tío Gerardo que me enseñó discresión y seriedad, mi tío Eder que siempre me mantuvo despierto en la realidad, y a cada uno que dió de su parte para hacer esto realidad.

Quiero dedicar este trabajo a la familia Quevedo Solano, en especial a Mary y Danilo, personas que me brindaron un hogar, con mucho cariño y amor, sin tener razones para hacerlo. Les estaré agradecido de por vida.

Quiero dedicarle esto a Katara, que ya no está conmigo, pero hizo parte de mi en cada momento, esto es para ella, y para Kurama, he sentido mucho consuelo de su parte.

Finalmente dedico este trabajo a los "Kunis"tantas risas y tantos recuerdos.

Agradecimientos

Este trabajo ha sido posible gracias a la Universidad Industrial de Santander, fue la encargada de toda mi formación personal y profesional, quiero agradecer a la UIS por permitirme desarrollarme, y hacerme apto para el futuro.

Quiero agradecer a la Electrificadora de Santander S.A. por darme la oportunidad de desarrollar y afianzar mis conocimientos en un entorno empresarial, por aportar a mi hoja de vida aprendizajes importante en el contexto organizacional, y por poner en mi camino a profesionales innatos y de calidad.

Quiero agradecer a mi tutor, por estar para mi en cada momento, enseñarme a aprender de mis errores, guiarme en un camino que en un principio parecía inescrutable, y finalmente fue enriquecedor.

Quiero agradecer a mi director Andrés Leonardo Gonzáles Gómez por la guía que me brindó, tanto personal como profesional, por confiar en mis capacidades y darme la oportunidad de realizar todo el trabajo representado en este documento.

Quiero hacer un agradecimiento especial a el director de escuela Luis Carlos Gómez Flórez, por confiar en mí cuando fue mi profesor, y haber dado un voto de confianza en mi desarrollo, todo lo que hizo en la gestión y pasión para hacerme soñar, y el coraje para llevar a cabo mis sueños.

Por último pero no menos importante, quiero agradecerme a mí. Quiero agradecerme por creer en mí. Quiero agradecerme por hacer todo este trabajo duro. Quiero agradecerme por no tomarme días libres. Quiero agradecerme por nunca rendirme. Quiero agradecerme por ser siempre generoso, y

por intertar dar más de lo que recibo. Quiero agradecerme por tratar de hacer más cosas bien que mal. Quiero agradecerme por simplemente ser yo en todo momento. Julián León, eres un crack.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Glosario

Automatización: Uso de tecnologías para realizar tareas o procesos sin intervención humana, con el objetivo de aumentar la eficiencia y reducir errores manuales.

Microsoft Power Platform: Conjunto de herramientas de Microsoft, que incluye Power Automate, Power Apps y Power BI, que permite la creación de soluciones empresariales con poco o ningún código.

Python: Lenguaje de programación utilizado para crear scripts que permiten la conexión, extracción y tratamiento de información de bases de datos.

RPA (Automatización Robótica de Procesos): Tecnología que permite automatizar tareas repetitivas utilizando robots de software.

SAC (Sistema de Administración Comercial): Sistema utilizado para la administración de la información comercial en las empresas.

JD Edwards: Sistema de planificación de recursos empresariales (ERP) que gestiona finanzas, proyectos y otros procesos comerciales.

GUI (Interfaz Gráfica de Usuario): Interfaz que permite la interacción del usuario con la solución de software de manera visual e intuitiva, facilitando la recopilación y ejecución de parámetros.

Encriptación: Proceso de codificación de información para protegerla de accesos no autorizados.

Conexión Segura: Método de conexión a bases de datos utilizando protocolos de seguridad como SSL/TLS para proteger la información durante la transmisión.

Flujo de nube: Automatización basada en la nube, utilizada para ejecutar procesos automáticamente desde un servidor en la nube.

Flujo de escritorio: Automatización que se ejecuta directamente en la computadora del usuario, procesando tareas locales.

Paquete Python: Conjunto de scripts que están estructurados para trabajar juntos como una aplicación o sistema.

Módulo: Archivo de código en Python que contiene definiciones de clases, funciones y variables, y puede ser importado por otros scripts.

Biblioteca (Library): Conjunto de módulos o paquetes que proporcionan funcionalidades específicas y están diseñados para ser reutilizados.

Pandas: Biblioteca de Python utilizada para la manipulación y análisis de datos, particularmente en formato de tablas o "DataFrames".

Oracle Client: Conjunto de librerías que permiten la conexión a bases de datos Oracle para ejecutar consultas SQL y otros procesos.

AES-256: Algoritmo de encriptación simétrica utilizado para proteger datos sensibles mediante el uso de una clave de cifrado.

SSL/TLS: Protocolos de seguridad utilizados para cifrar la transmisión de datos entre un cliente y un servidor.

VPN (Red Privada Virtual): Tecnología que crea una conexión segura y cifrada sobre una red menos segura, como Internet.

Data Wrangling: Proceso de limpieza y transformación de datos para que sean utilizables en análisis.

DataFrame: Estructura de datos bidimensional, similar a una tabla, que organiza los datos en filas y columnas, usada en Python con la biblioteca Pandas.

Query SQL: Instrucción de lenguaje de consulta estructurado utilizada para realizar operaciones como la extracción o manipulación de datos en una base de datos.

SID (System Identifier): Identificador único de una instancia específica de una base de datos Oracle.

Service Name: Identificador utilizado en bases de datos Oracle para permitir la conexión a un servicio específico en entornos de múltiples instancias.

JSON: Formato de intercambio de datos que utiliza texto plano y estructurado en clave-valor, comúnmente utilizado en configuraciones de software y APIs.

Orquestación: Proceso de coordinar y gestionar flujos de trabajo automatizados que involucran múltiples sistemas o servicios.

Cifrado en tránsito: Proceso de proteger la información mientras se transmite entre dos puntos, generalmente utilizando protocolos como SSL/TLS.

Cifrado en reposo: Protección de datos almacenados en un sistema, asegurando que la información esté encriptada cuando no está siendo utilizada.

Microsoft Power Automate: Herramienta de automatización de flujos de trabajo que permite conectar aplicaciones y servicios para ejecutar tareas repetitivas automáticamente.

Microsoft Power Apps: Plataforma que permite desarrollar aplicaciones empresariales con poco o ningún código, enfocándose en la creación de interfaces gráficas de usuario (GUI).

Microsoft Teams: Herramienta de comunicación y colaboración en equipo de Microsoft, utilizada frecuentemente en entornos empresariales.

Power Automate Desktop: Versión de escritorio de Power Automate, que permite la automatización de tareas locales en el equipo del usuario.

SQL (Structured Query Language): Lenguaje de consulta estructurado utilizado para gestionar bases de datos relacionales.

Data Wrangling: Proceso de limpieza y transformación de datos en bruto en un formato utilizable para el análisis.

AES (Advanced Encryption Standard): Algoritmo de cifrado utilizado para proteger la información sensible con una clave simétrica.

VPN (Virtual Private Network): Red que permite conexiones seguras a través de redes públicas, cifrando la comunicación para proteger la información.

TLS (Transport Layer Security): Protocolo criptográfico diseñado para proporcionar seguridad en las comunicaciones a través de redes informáticas.

Resumen

Título: Automatización de procesos de gestión contable en la Electrificadora de Santander S.A. (ESSA) a través de herramientas de Microsoft y Scripts de Python. *

Autores: Julián David León Quintero **

Palabras Clave: Automatización, Microsoft Power Platform, Python

Descripción: La generación manual de reportes contables en una empresa consume tiempo excesivo y afecta la eficiencia operativa. Este trabajo automatiza el proceso de generación y envío de reportes contables, reduciendo significativamente el tiempo y aumentando la precisión. Se integraron herramientas de Microsoft Power Platform (Power Automate y Power Apps) con scripts en Python, conectándose a bases de datos, procesando datos, y generando reportes automatizados. Esta implementación redujo el tiempo de ejecución de 135 horas trimestrales a un máximo de 20 minutos, disminuyendo el esfuerzo anual de 540 horas a aproximadamente 2 horas. La automatización que integra Microsoft con Python, ha demostrado ser una mejora significativa en la eficiencia operativa. Esta combinación permitió tratar grandes volúmenes de datos, liberar recursos y optimizar los tiempos de entrega de los reportes.

---

* Trabajo de grado

** Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería de Sistemas e Informática.
Director: Andrés Leonardo Gonzáles Gómez, Magister en Ingeniería de Sistemas e Informática

Abstract

Title: Automation of accounting management processes in Electrificadora de Santander S.A. (ESSA) using Microsoft tools and Python scripts. *

Authors: Julián David León Quintero **

Keywords: Automation, Microsoft Power Platform, Python.

Description: The manual generation of accounting reports in a company consumes excessive time and affects operational efficiency. This work automates the process of generating and sending accounting reports, reducing time and increasing accuracy. Microsoft Power Platform tools (Power Automate and Power Apps) were integrated with Python scripts, connecting to databases, processing data, and generating automated reports. This implementation reduced the execution time from 135 hours per quarter to a maximum of 20 minutes, reducing the annual effort from 540 hours to approximately 2 hours. Automation integrating Microsoft with Python has proven to be a significant improvement in operational efficiency. This combination made it possible to process large volumes of data, free up resources and optimize report turnaround times.

---

* Bachelor Thesis

** Faculty of Physicomechanical Engineering. School of Systems and Computer Engineering.
Director: Andrés Leonardo Gonzáles Gómez, Master in Systems and Computer Engineering

Introducción

En el contexto empresarial, la generación eficiente de reportes es crucial en la toma de decisiones y cumplmiento de obligaciones regulatorias, sin embargo, implica un recurso empresarial laborioso, debido a que la información tiene diversos orígenes. Este enfoque no solo está propenso a errores manuales sino también a la duplicidad de esfuerzos.

Este trabajo presenta una propuesta para solventar este problema, que comprende la automatización de la generación de reportes contables, cuya arquitectura integra herramientas de Microsoft Power Platform y Python. Según el trabajo de (Narayn, 2023a) donde propone el uso de herramientas de Microsoft Power Platform, donde su integración genera un gran valor añadido a sus capacidades (Narayn, 2023a). Microsoft Power Automate ofrece gran facilidad al orquestar flujos de trabajo automatizados (Vartiainen, 2024a), el uso de Scripts de Python capaces de hacer conexión a los diferentes orígenes de datos así como su tratamiento (Rattenbury et al., 2017), y Microsoft Power Apps para la mejora en la interacción y control del usuario sobre la solución (Simon, 2022). Se explora la implementación técnica de esta solución, evaluando el impacto de la integración de estas tecnologías por medio del desarrollo de servicios, partiendo de la flexibilidad que ofrece Python a la hora de tratar datos, en la eficiencia operativa de la gestión contable.

En consecuencia, se presenta el siguiente problema.

1. Planteamiento y Justificación del Problema

La Contaduría General de la Nación (CGN) requiere el reporte de cuentas contables de las entidades oficiales suministrado por la Electrificadora de Santander S.A (ESSA) con frecuencia trimestral que a su vez es enviada a las partes interesadas. Para la entrega de esta información, la ESSA designó un recurso (empleado) con actividades manuales propensas al error y que demandan tiempo que podría utilizarse en tareas con una mayor relevancia, que ofrezcan un valor agregado superior al que se obtiene de estos procesos repetitivos.

Surge entonces la siguiente pregunta problemática: £Cómo puede la ESSA implementar una solución de RPA para mejorar la eficiencia en la generación y envío de reportes a CGN, minimizando errores y optimizando el uso del tiempo del personal asignado?

Con la intención de contribuir a la solución de esta problemática, se contempla generar una solución a través de la implementación de Automatización Robótica de Procesos (RPA) debido a que son tareas repetitivas que cumplen con unos criterios específicos que se mantienen en el tiempo, como las reglas de negocio estipuladas para entregar y solicitar información, la frecuencia transaccional, entre otros. Actualmente la ESSA tiene un licenciamiento de Microsoft E3 M365 que permite desarrollar soluciones low-code y de automatización, que se ajusta al planteamiento del problema, por lo que se estima que la solución puede ser desarrollada por medio de la herramienta Microsoft PowerApps para la creación de una interfaz gráfica de usuario (GUI) con el fin de que

guíe intuitivamente la recopilación de información requerida para el desarrollo. Además de lo anterior, se propone el uso de Microsoft Power Automate para automatizar el proceso de creación y carga del reporte de cuentas contables de entidades oficiales, así como para su posterior envío por correo a las partes interesadas. Esta herramienta, integrada dentro del ecosistema de Microsoft, facilita la automatización de flujos de trabajo repetitivos, optimizando la entrega de información de manera precisa y oportuna.

Debido a la complejidad de esta solicitud y garantizando la calidad de la información, es necesario implementar herramientas como Python para la conexión, extracción y tratamiento de información con los requisitos expedidos por la CGN. Considerando el uso de dichas herramientas, se espera que la solución automatice en su totalidad la creación del reporte de entidades oficiales y el envío de esta información a cada entidad vía correo de lo reportado a CGN

Con el propósito de abordar este dilema, este estudio tiene los siguientes objetivos.

2. Objetivos

2.1. Objetivo General

- Automatizar el proceso de digitalización de información obtenida por los sistemas de información utilizados en ESSA, requerida para el reporte de entidades oficiales definidas por la CGN.

2.2. Objetivos Especificos

- Crear Scripts que permitan la conexión, extracción y tratamiento de información de las bases de datos del Sistema de Administración Comercial (SAC) y el Sistema JD Edwards. Información expedida por la CGN.
- Implementar Automatización Robótica de Procesos (RPA) por medio de flujos de nube y de escritorio con Microsoft para la creación y cargue del reporte en la página oficial de CGN incluyendo el envío a las partes interesadas.
- Construir interfaz gráfica de usuario (GUI) usando Microsoft PowerApps, para la captura de parámetros necesarios para la ejecución del RPA.
- Desarrollar un sistema de seguridad usando encriptación avanzada y conexiones seguras (SSL/TLS), para proteger las credenciales y controlar el acceso a bases de datos mediante VPN y firewalls.

- Ejecutar escenarios de pruebas funcionales y técnicas para el despliegue y estabilización de la solución implantada incluyendo la documentación de estos.

Para lograr estos objetivos, el alcance del proyecto abarca una serie de elementos esenciales que deben ser cuidadosamente planificados y ejecutados.

2.3. Alcance

El alcance de la solución está limitado por los datos y por qué tan parametrizable puede ser su tratamiento para la generación del reporte. Sin embargo, el objetivo es la automatización total del reporte, en la medida en que la parametrización lo permita. Las herramientas utilizadas y su seguridad imponen límites de implementación, dado que el enfoque del estudio está dirigido principalmente a la lógica manejada por Python. Por ello, son las herramientas de Microsoft las que marcan el camino en la ejecución de la solución, asumiendo la responsabilidad total de su implementación.

Para comprender completamente el contexto y la necesidad de este proyecto, es esencial analizar el estado del arte.

3. Estado del Arte

3.1. Antecedentes

La Automatización Robótica de Procesos (RPA, por sus siglas en inglés) ha ganado popularidad en diversas industrias debido a su capacidad para automatizar tareas repetitivas y basadas en reglas, incrementando la eficiencia operativa y reduciendo costos. En el trabajo de (Cooper et al., 2019), se destaca cómo las firmas de contabilidad han adoptado RPA para automatizar la generación de reportes financieros y otras tareas rutinarias, mejorando la precisión y reduciendo los errores humanos. Este tipo de tecnologías permite a las empresas liberar a los empleados de tareas manuales para que puedan enfocarse en actividades de mayor valor añadido.

Un aspecto clave de la RPA es su capacidad para integrarse sin necesidad de modificar los sistemas subyacentes (Aalst et al., 2018), lo que facilita su implementación en una amplia variedad de entornos empresariales. Sin embargo, como mencionan (Syed et al., 2020), los desafíos actuales incluyen la necesidad de integrar capacidades cognitivas, como el aprendizaje profundo, para mejorar la adaptabilidad de los bots en entornos cambiantes. En este sentido, las plataformas de RPA han comenzado a evolucionar hacia la Automatización de Procesos Inteligentes (IPA, por sus siglas en inglés), que combina RPA con Inteligencia Artificial (IA) y Machine Learning (ML), ofreciendo soluciones más robustas (A. Gami y Singh, 2019).

3.2. Desarrollo Actual

La plataforma Microsoft Power Automate es una herramienta clave en la automatización de procesos de negocios, especialmente en entornos corporativos que ya utilizan la suite de Microsoft 365. Su capacidad para automatizar flujos de trabajo a través de aplicaciones como SharePoint, OneDrive y Outlook permite a las organizaciones integrar datos de múltiples fuentes y realizar tareas complejas sin intervención humana (Microsoft, 2024g). Además, Power Automate ofrece dos tipos principales de automatización: asistida y no asistida. En la automatización asistida, los bots replican acciones humanas en tiempo real, mientras que la automatización no asistida ejecuta tareas programadas sin supervisión (Xerox, 2023). Esto es particularmente útil para la automatización de procesos contables que requieren la generación y envío de reportes de manera recurrente.

Python, por su parte, ha sido ampliamente utilizado para la manipulación y tratamiento de datos en procesos de automatización. Herramientas como Pandas y NumPy permiten limpiar, transformar y analizar grandes volúmenes de datos, lo que resulta esencial para asegurar la calidad de los reportes generados automáticamente (Rattenbury et al., 2017). Python también ofrece librerías especializadas para la automatización de tareas de escritorio, como ‘pywinauto‘ y ‘pyautogui‘, las cuales facilitan la interacción automatizada con aplicaciones en sistemas Windows (Team, 2024). Estas bibliotecas permiten combinar el poder de la automatización robótica con las capacidades de procesamiento de datos de Python, ofreciendo una solución integral para la automatización de procesos contables.

En empresas como Xerox, Power Automate se ha utilizado para generar reportes de facturación automatizados, eliminando la necesidad de generación manual y reduciendo significativamente los tiempos de procesamiento (Xerox, 2023). La integración de estas herramientas con tecnologías de Microsoft como Azure Key Vault y SharePoint asegura la seguridad y confiabilidad de los procesos, al utilizar encriptación de 256 bits para la protección de credenciales y datos sensibles.

En resumen, la Automatización Robótica de Procesos y las plataformas de bajo código como Microsoft Power Automate, combinadas con el poder de Python para el tratamiento de datos, representan una solución robusta y eficaz para la optimización de procesos contables. Estas tecnologías no solo reducen el esfuerzo humano y el riesgo de errores, sino que también mejoran la eficiencia operativa y facilitan el cumplimiento de normativas mediante la automatización de tareas repetitivas y la generación de reportes automáticos.

4. Marco de Referencia

La optimización de procesos es crucial para la competitividad de las organizaciones en el mercado, ya que busca minimizar costos y maximizar el rendimiento, la productividad y la eficiencia. Con el avance de la revolución tecnológica, las empresas han comenzado a implementar la transformación digital para optimizar los procesos en la cadena de valor. Este proyecto está enfocado en el desarrollo de soluciones digitales con el uso de tecnologías de Automatización Robótica de Procesos (RPA), Plataformas de Desarrollo de Bajo Código (LCDPs) y Análisis de Datos (Data Analysis).

4.1. Automatización Robótica de Procesos (RPA)

La Automatización Robótica de Procesos (RPA) en los últimos años se ha posicionado como una herramienta esencial para la optimización de proceso, debido a paradigmas de automatización de tareas repetitivas en diferentes industrias (Doguc, 2020). Su arquitectura es un software que imita las acciones humanas al usar interfaces de usuario de diferentes aplicaciones, permitiendo así una integración sin modificar sistemas subyacentes. En el trabajo de (Doguc, 2020) se describe a RPA como una tecnología clave que reduce costos y mejora la eficiencia operativa al automatizar tareas basadas en reglas que anteriormente realizaban los humanos. Esta tecnología no solo aumenta la precisión, sino que también tiene la gran capacidad de liberar a los empleados de tareas manuales, para que se centren en tareas más estratégicas y de mayor valor añadido . Sin embargo, en lo propuesto por (Syed et al., 2020) se analizan los temas contemporáneos y desafíos a los que

se enfrenta RPA como nueva tecnología en auge, destacando así la integración de capacidades cognitivas y de aprendizaje profundo. Estos avances no solo permiten a los bots ir más allá de la ejecución de tareas repetitivas, sino también la capacidad de adaptarse al entorno cambiante. Por otro lado, menciona la relevancia de la gobernanza centralizada y la preservación del conocimiento del proceso para asegurar la efectividad de RPA.

Según (Aalst et al., 2018) el enfoque de afuera hacia adentro de RPA lo diferencia de otros métodos de automatización. Este enfoque permite que los sistemas de información existentes permanezcan en el tiempo sin cambios, de la misma manera que los bots asumen las tareas humanas. Esto contrasta con enfoques tradicionales como el Procesamiento Directo (STP), que requerían modificaciones en los sistemas subyacentes. En su investigación (Madakam et al., 2019) exploran cómo RPA está transformando la fuerza laboral digital del futuro. La automatización de tareas rutinarias permite que los empleados inviertan su tiempo laboral en actividades que aporten mayor valor añadido a las empresas. Además de esto, se discuten los impactos en la estructura organizacional y la necesidad de gestionar el cambio para asegurar la implementación exitosa de estas tecnologías.

Finalmente, (M. Gami et al., 2019) ofrecen una revisión exhaustiva del estado del arte de la implementación de la tecnología RPA y su impacto en las organizaciones empresariales. Destacan los beneficios de su implementación como la reducción de costos y la mejora en la calidad del trabajo, también analizan el encaminamiento de estas tecnologías con dirección a la Automatización de Procesos Inteligentes (IPA), que combina RPA con tecnologías avanzadas como la Inteligencia Artificial (IA) y el Aprendizaje Automático para ofrecer soluciones más robustas y eficaces.

La adopción de la Automatización Robótica de Procesos representa una evolución crucial en la manera en que las organizaciones optimizan sus procesos, en aras de la reducción de costo y la mejora constante en la calidad del trabajo. Entre las herramientas de RPA, Microsoft Power Automate se destaca por su capacidad para automatizar flujos de trabajo entra aplicaciones y servicios, permitiendo a las empresas aumentar su eficiencia y productividad mediante la simplificación de tareas diarias.

El software de automatización Microsoft Power Automate es un servicio de automatización basado en la nube y un conjunto de herramientas de desarrollo de bajo código, diseñado para facilitar la creación de flujos de trabajo automatizados, flujos que a su vez automatizan procesos empresariales. Esta herramienta de bajo código que hace parte de la suite de Microsoft Power Platflorm, tiene como objetivo optimizar las operaciones empresariales al reducir las herramientas manuales y aumenta la eficiencia. Según (Najjar, 2023) Power Automate facilita la programación autodidacta para usuarios empresariales y no desarrolladores, permitiéndoles construir automatizaciones funcionales si necesidad de conocimientos avanzados de programación. La herramienta soporta varios tipos de automatización, incluyendo flujos en la nube, flujos de escritorio y flujos de procesos de negocio, cada uno adaptado a diferentes necesidades de automatización. Añadido a esto, Power Automate cuenta con más de 400 conectores que permiten integrar servicios como SharePoint, Onedrive, Outlook y Teams, lo que maximiza su utilidad para las organizaciones que ya disponen de herramientas ofrecidas por Microsoft 365.

En este contexto, (Vartiainen, 2024b) enfatiza cómo Power Automate puede mejorar la experiencia

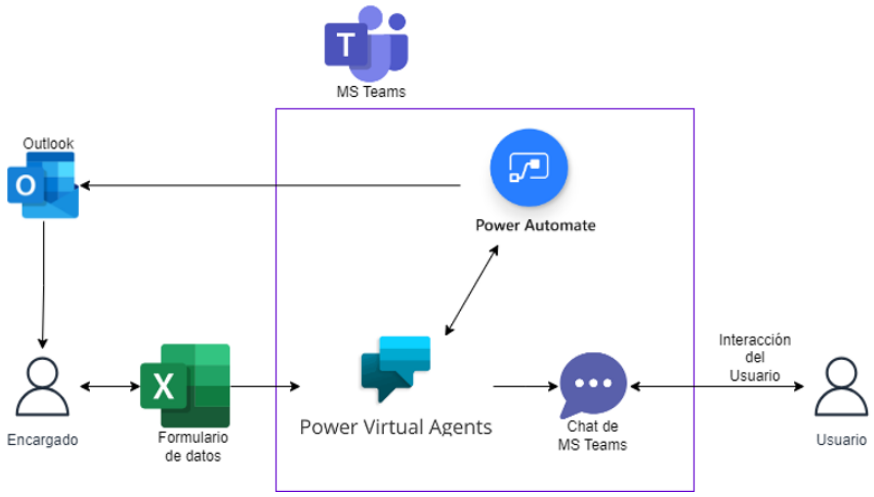
de los empleados mediante RPA. Afirma que la plataforma facilita la automatización de flujos de trabajo completos, integrando datos de diversas fuentes y mejorando la productividad y eficiencia operativa. No solo es útil para automatizar tareas rutinarias, puesto que también es una herramienta poderosa para la integración de datos que mejora la toma de decisiones y la gestión de procesos empresariales. Esta herramienta con su capacidad para manejar flujos en la nube y flujos de escritorio puede adaptarse a una amplia gama de escenarios empresariales, desde la automatización de procesos simples hasta la gestión de flujos de trabajo complejos que involucren múltiples aplicaciones y servicios.

El asistente virtual interno LUISA de la ESSA, nace de la necesidad de agilizar el acceso a información clave por parte de los trabajadores. Tiene el objetivo de guiar a la Gente ESSA en la resolución de dudas sobre cesantías, vacaciones, reemplazos, pensiones, nómina, permisos, seguridad social y beneficios convencionales. De esta manera mejorar la eficiencia y productividad de los empleados al simplificar la obtención de datos relevantes proporcionando una herramienta integral para la gestión eficaz de temas laborales y prestaciones en todos los niveles y departamentos de la organización. El asistente virtual LUISA está implementado en Microsoft Teams, el cual es un medio de comunicación interno de la empresa. Este asistente virtual fue diseñado con la integración de tecnologías como Microsoft Power Automate, Microsoft Copilot Studio, Microsoft Teams, Microsoft Outlook y Microsoft Excel. (ver Figura 1)

La infraestructura de LUISA usa Microsoft Copilot Studio como plataforma principal para diseñar las conversaciones, definir flujos de trabajo y configurar respuestas. Microsoft Teams actúa

Figura 1.

Arquitectura LUISA.



Nota: Arquitectura de Luisa, RPA exitoso en la ESSA. León, J. (2024). Diseño propio.

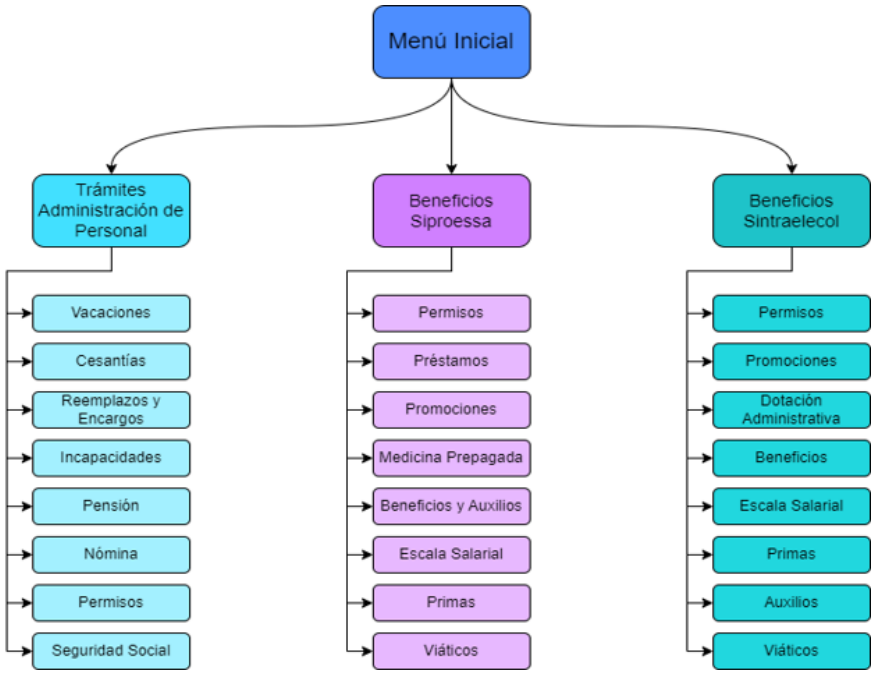
como el canal de interacción principal, brindando a los usuarios un espacio familiar para acceder a las capacidades del asistente virtual de manera eficiente. En la integración con Microsoft Power Automate usa palabras desencadenadoras, las cuales al ser accionadas accionan un flujo creado previamente el cual tomará una serie de respuestas o acciones de envío de correos para dar soluciones al usuario. La estructura de los temas hallados en LUISA (ver Figura 2). está ramificado según el sindicato, pues el tipo de solicitud y sus respuestas son diferentes en cada una.

4.2. Plataformas de Desarrollo de Bajo Código (LCDP's)

Las plataformas de desarrollo de bajo código (LCDPs) han emergido como una solución innovadora para enfrentar la creciente demanda de aplicaciones. En el trabajo de (Alsaadi et al., 2021) se indica que las LCDPs permiten a las organizaciones crear aplicaciones rápida y fácilmente por

Figura 2.

Estructura LUISA.



Nota: Estructura de Luisa, RPA exitoso en la ESSA. León, J. (2024). Diseño propio.

medio de interfaces visuales y componente preconstruidos, disminuyendo significativamente la necesidad de codificación manual. Este enfoque libera el desarrollo de software, pues permite a empleados participar en desarrollos de aplicaciones sin tener un estudio técnico avanzado previo, a ellos se les conoce como desarrolladores ciudadanos. En consecuencia, (Prinz et al., 2021) destacan el aumento en la relevancia de las LCDP y cómo están transformando las operaciones y la estructura organizativa al agilizar el desarrollo, consecuente a esto la reducción de los costos asociados a la contratación y formación de personal en el cargo de desarrolladores especializados.

Las LCDPs ofrecen varios beneficios importantes para las empresas, (Sharma y Arora, 2023) indican que entre estos beneficios se encuentran la rapidez y eficiencia en el desarrollo, lo que permite

desplegar aplicaciones en tiempos verdaderamente despreciables comparándolos con métodos que se realizaban anteriormente. Además, la facilidad de integración con diferentes orígenes de datos y sistemas, mejorando así la cooperación operativa en las organizaciones. No obstante, Käss, Strahringer y Westner et al. (2023) afirman que la accesibilidad y facilidad de uso de estas plataformas permiten a los desarrolladores ciudadanos crear soluciones personalizadas que atienden necesidades específicas del negocio, promoviendo una mayor innovación interna. Sin embargo, la adopción de LCDP también enfrenta desafíos.

Finalmente, (Alsaadi et al., 2021) mencionan la necesidad establecer estructuras claras de gobernanza y medidas de seguridad robustas para así garantizas en gran parte la calidad y protección de los datos.

En conclusión, las plataformas de desarrollo de bajo código están reestructurando el entorno del desarrollo de software empresarial, permitiendo a las organizaciones generar más agilidad e innovación en su cadena de valor. El aumento de la inclusión de estas plataformas en las empresas trae consigo la necesidad de implementar procesos de gobernanza y seguridad en aras de la maximización de su potencial. Una de las herramientas más relevantes en este contexto es Microsoft Power Apps, pues ofrece una robusta plataforma de bajo código, lo cual permite a los usuarios empresariales no solo desarrollar aplicaciones rápidamente, sino también integrar soluciones en su ecosistema.

La herramienta Microsoft Power Apps no deja de revolucionar cada vez más el desarrollo de apli-

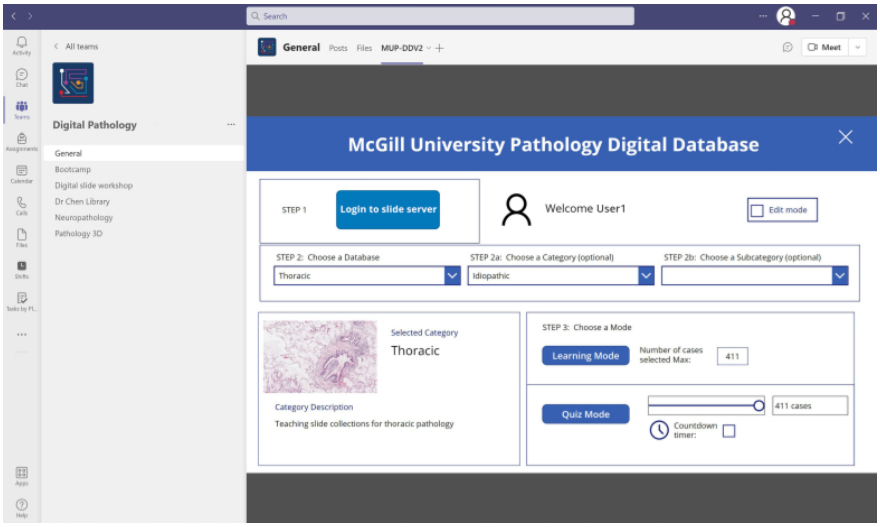
caciones por medio de su plataforma de bajo código, la cual permite a los usuarios crear aplicaciones empresariales sin la necesidad de conocimientos avanzados en programación por parte de los empleados. Según (Pandit, 2024), Power Apps destaca en el contexto de LCDPs por su interfaz intuitiva la cual se trata de arrastrar y soltar, sus conectores preconstruidos y plantilla, y su gran capacidad de integración con diversas fuentes de datos. Esta plataforma no solo facilita a los usuarios la creación rápida de aplicaciones personalizadas, sino que también se destaca por fomentar la innovación tecnológica dentro de las organizaciones. En este sentido (Narayn, 2023b) destaca la importancia de Power Apps en el entorno del trabajo moderno, integrándolo con herramientas como SharePoint Online, Power Automate y Microsoft Teams para con ello ofrecer soluciones completas de colaboración y gestión de contenido. Por otro lado, (Di Ruscio et al., 2022) subraya cómo esta plataforma, en combinación con la ingeniería dirigida de modelos (MDE), proporciona un enfoque eficiente y flexible para el desarrollo de software, abordando tanto los aspectos técnicos como organizativos de la creación de aplicaciones.

La Universidad McGill es un notable caso de éxito en su implementación de Microsoft Power Apps y Microsoft Teams, para el desarrollo de una plataforma educativa virtual de la disciplina patología. Este caso ha sido descrito por (Rajaram et al., 2022). El departamento de Patología Universidad McGill en respuesta a la pandemia de COVID-19, implementó Microsoft Power Apps y Microsoft Teams (ver Figura 3) para continuar la educación de los estudiantes de manera remota. Esta plataforma personalizada permitió a los residentes acceder a imágenes digitalizadas de diapositivas completas, participar en sesiones educativas y realizar exámenes a distancia (ver Figura

4).

Figura 3.

Plataforma virtual Universidad McGuill.



Nota: Visión de la aplicación Patología Digital. Rajaram et al., 2022. Adaptación.

La integración con Microsoft Teams facilitó la organización y el acceso a todos los recursos educativos, mejorando así significativamente la continuidad y calidad de la educación durante esta etapa. Esta implementación demuestra el éxito de Power Apps, y cómo puede ser usado para desarrollar soluciones educativas innovadoras y eficiente, incluso fuera de este contexto. La gran capacidad de adaptación a las necesidades cambiantes y con ello asegurando la continuidad educativa en tiempos de crisis.

4.3. Tratamiento de datos (Data Wrangling)

El proceso de limpieza al que se someten los datos, conocido como Data Wrangling, es parte importante en el ciclo de vida de los datos, puesto que este proceso asegura que los conjuntos de

Figura 4.

Plataforma virtual Universidad McGuill.

Nota: Aplicación de Patología digital que muestra: a) Lista de imágenes de portaobjetos completos con sus diagnósticos que se pueden lanzar. b) Modo de cuestionario dentro de la aplicación. c) Visualización de las respuestas evaluadas por los usuarios. Rajaram et al., 2022. Adaptación.

datos sean precisos, consistentes y compactos, esto los vuelve utilizables. Según (Rattenbury et al., 2017) la identificación y corrección de errores, la eliminación de duplicados y el tratamiento de valores atípicos y faltantes, es lo que implica este proceso. No obstante, (Kazil y Jarmul, 2016) aseguran que la implementación efectiva permite que los análisis se basen en datos confiables, lo cual es crucial para obtener resultados significativos en su utilización. Las técnicas más utilizadas en el Data Wrangling son la atribución de valores faltantes, la normalización de datos numéricos, y la codificación de variables categóricas. Estas técnicas ayudan a mejorar la calidad de los datos, asegurando decisiones empresariales basadas en estos datos cohesionados, sean precisas y confiables.

En este contexto, Python es una de las herramientas más usadas en la limpieza de datos debido a

su versatilidad y robustez del compendio de bibliotecas. Herramientas como Pandas, NumPy tienen una colección de funciones avanzadas que permiten la manipulación y transformación de los datos, consiguiendo facilitar tareas como la eliminación de duplicados, la gestión de valores faltantes, irregularidades en los registros y normalización de características numéricas (Van Der Aalst et al., 2018) , (Jafari, 2024). La biblioteca Pandas, permite la lectura, transformación y limpieza de grandes conjuntos de datos, que hace llamar DataFrame, de manera eficiente, lo cual es parte fundamental para la preparación de los datos a análisis con mayor complejidad y modelado predictivo (Provost y Fawcett, 2013). Además, Python es un lenguaje de programación con facilidad de uso, y su comunidad de usuarios la hacen la opción más accesible con un sector objetivo de oscila entre principiantes y profesionales experimentados en ciencia de datos. En conclusión, el uso de Python no solo optimiza el proceso de cleaning, sino que también ofrecen una base sólida para realizar análisis de datos más profundos y precisos.

4.4. Seguridad

En el contexto de las soluciones digitales, la seguridad en la protección de datos y credenciales es primordial. Por ello, aquellos trabajos que pretenden manejar información sensible deben implementar mecanismos sólidos de seguridad para evitar brechas que expongas información con alto riesgo, como credenciales o incluso información de terceros involucrados. Teniendo en cuenta esto, existen diversas herramientas y tecnologías cuyas características de seguridad integradas permiten la gestión y protección de la información durante su ciclo de vida útil. Algunos métodos de estos son:

4.4.1. Encriptación de Datos Sensibles

Uno de los más importantes métodos para proteger los datos sensibles, es la encriptación de estos. La encriptación simétrica, especialmente mediante el algoritmo AES-256, se ha consolidado como uno de los métodos más seguros y eficientes para proteger datos en reposo. Este tipo de encriptación utiliza una única clave para cifrar y descifrar la información, garantizando así que solo se pueda acceder a los datos protegidos por medio de autorización. Los archivos de configuración, como aquellos que contienen credenciales de acceso a bases de datos, deben ser cifrados antes de ser almacenados para protegerse de accesos no autorizados (Cryptography, 2024).

Un punto importante parte de la necesidad de que la clave de encriptación debe estar gestionada de manera correcta, evitando que sea vulnerada. Existen herramientas de gestión de secretos como Azure Key Vault, que gestiona claves en entornos locales. Estas herramientas permiten almacenar y tener control sobre el acceso a estas claves de manera segura (Microsoft, 2024a).

4.4.2. Conexiones Seguras a Bases de Datos

En proyectos que dependen de consultas y operaciones sobre bases de datos, establecer conexiones seguras es crucial para proteger la información durante su transmisión. Bases de datos comerciales, como Oracle y SQL Server, ofrecen soporte para conexiones cifradas a través de protocolos SSL/TLS. Estos protocolos garantizan que los datos se transmitan de forma segura, evitando que sean interceptados o modificados por terceros no autorizados.

En la práctica, el uso de la librería Python-oracledb en modo Thick, por ejemplo, facilita la configuración de conexiones seguras entre la solución y la fuente de datos, aprovechando las librerías cliente de Oracle para implementar Capa de Sockets Seguros/Seguridad de la capa de transporte (SSL/TLS) (Oracle, 2024a). Este proceso asegura el tránsito de información entre el servidor y las bases de datos, para solidificar aún más la solución, algunos trabajos refuerzan la seguridad haciendo uso de redes privadas virtuales (VPN) y firewalls para restringir el acceso a la base de datos únicamente a usuarios y direcciones IP autorizadas, brindando una capa adicional de seguridad (Microsoft, 2024f).

4.4.3. Seguridad en la plataforma Microsoft Power Platform

Microsoft Power Platform, que incluye herramientas como Power Apps y Power Automate, contiene prácticas de seguridad integradas para controlar los accesos a las soluciones creadas y garantizar la protección de los datos usados en los entornos. La plataforma implementa un modelo de control de acceso basado en roles (RBAC), que permite asignar permisos específicos a los usuarios en función de su rol. Esta segmentación asegura que solo los usuarios autorizados puedan acceder y manipular aplicaciones, flujos o datos sensibles dentro del entorno (Microsoft, 2024e).

Adicional a esto, contiene capacidades de cifrado de datos en tránsito y en reposo, utilizando protocolos como TLS para la transmisión de información y AES-256 para el almacenamiento de datos sensibles (Microsoft, 2024c). Garantizando que las conexiones y los datos almacenados en la nube estén protegidos de accesos no autorizados y manipulaciones.

Basándonos en esta base conceptual, se debe profundizar en la metodología de investigación.

5. Metodología

En esta sección se aborda de qué manera se van a llevar a cabo los objetivos planteados inicialmente, en la cual se notarán las configuraciones que se deben realizar en los softwares, las tecnologías que se van a usar, y qué se va a realizar con ellas. Consta de 4 etapas, las cuales son la conexión a las bases de datos y extracción de la información por medio de Python, la automatización de este proceso con Power Automate, la realización de la GUI con PowerApps y finalmente, la implementación de pruebas, documentación y manuales técnicos y de soporte.

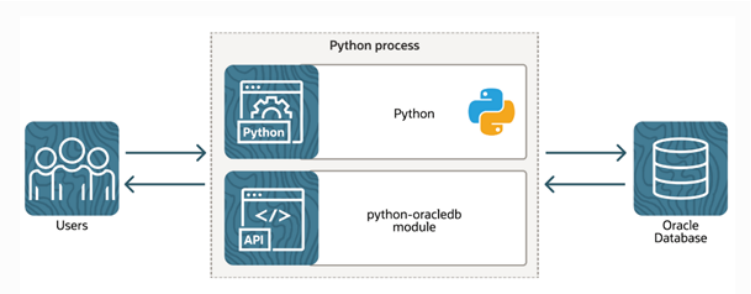
5.1. Script de Python

Se propone crear un script mediante el cual se utiliza la librería de Python para conectar con las bases de datos usadas en los sistemas de información. Estas bases de datos están alojadas en Oracle, por lo que se requiere llevar a cabo la conexión a través de la librería `Python-oracledb`. Esta librería, en particular, tiene dos modos de conexión: la arquitectura del modo "Thin"(ver Figura 5) permite la conexión directa a la base de datos, la cual puede estar en la misma máquina o puede ser remota (Oracle, 2024a).

El otro modo es el Thick (ver Figura 6). Python-oracledb opera en este modo cuando se enlaza con las librerías de Oracle Client, lo que le proporciona funcionalidad adicional. Dado que las bases de datos con las que se requiere establecer la conexión son de versiones antiguas, se propone utilizar Python-oracledb en modo Thick para garantizar compatibilidad y aprovechar sus capacidades

Figura 5.

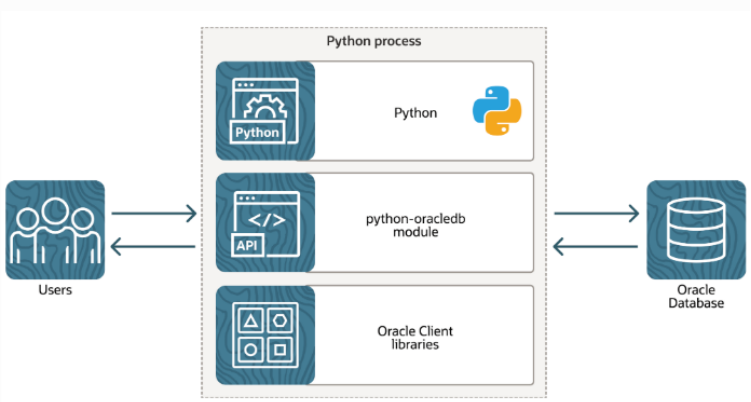
Modo Thin.



Nota:Arquitectura del controlador python-oracledb en modo Thin. (Oracle, 2024a). Adaptación.

Figura 6.

Modo Thick.



Nota: Arquitectura del controlador Python-oracledb en modo Thick. (Oracle, 2024a). Adaptación.

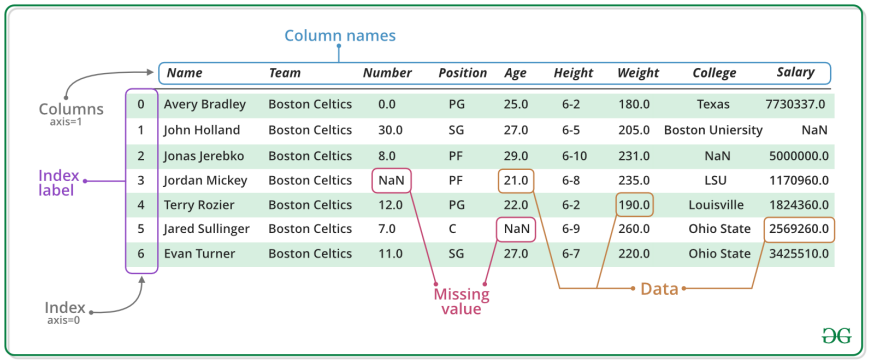
avanzadas (Oracle, 2024a).

Posterior a la conexión, se hace una consulta a la base de datos, por medio de un query, normalmente está hospedado en un archivo con formato sql, esta consulta va a traer aquella información que se especifique, y para mayor facilidad en el siguiente proceso, se convertirá a un formato DataFrame (ver Figura 7). Este formato permite a la librería Pandas de Python limpiar y procesar la información con todas las herramientas que posee. Hay que hacer procesos de limpieza: eliminar

duplicados, eliminar nulos, cruzar información con archivos de Excel, cambiar los valores de una columna en específico y cambiar el tipo de dato para operarlo con otros. Esta librería permite obtener un reporte con el formato establecido, y el tipo de archivo necesario (.txt) (pandas development team, 2024).

Figura 7.

DataFrame `Pandas`.



Nota: Estructura de un DataFrame de Pandas. (pandas development team, 2024). Adaptación.

Finalmente, este Script de Python resulta en un archivo plano que contiene la información lista para reportar.

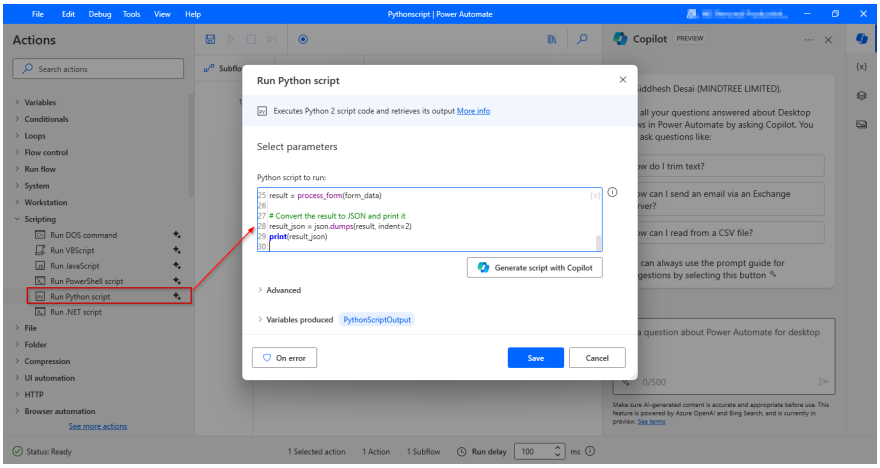
5.2. Automatización Robótica de Procesos

Como consiguiente de este proceso, alguien debe tener la responsabilidad de ejecutar este Script, y aquí se introduce el papel de la Automatización Robótica de Procesos (RPA) por medio de la herramienta Microsoft Power Automate, se crean flujos de nube, cuyo desencadenador es la captura de parámetros por medio de una interfaz. A partir de esto, la herramienta low-code proporciona gran facilidad para la creación de los flujos, para este caso en particular se requiere crear un flujo

que ejecute el Script de Python por medio de Power Automate Desktop (ver Figura 8). Este flujo pondrá los parámetros capturados en el momento que se desencadena, y los pondrá en el Script, para seguir con la ejecución de este, ya que este Script da como resultado un archivo plano, el siguiente flujo buscará este archivo plano en la ruta donde se hospedó, para cargarlo en la página web requerida, para ello tendrá que validar con el usuario, por ello enviará un flujo de aprobación (Stack Overflow, 2024).

Figura 8.

Ejecución Script con Power Automate



Nota: Ejemplo de la ejecución de un Script de Python desde un flujo de Power Automate Desktop. (Stack Overflow, 2024). Adaptación.

Siendo la aprobación efectiva, se construye el flujo que carga la información, por lo que irá a obtener las credenciales en un archivo cuya ruta conoce y hará el respectivo cargue. Una vez hecho lo anterior, se enviará un correo masivo a las entidades por medio de Outlook, este se enviará mediante un conector de Power Automate, cuyo contenido incluirá una tabla HTML dinámica, reflejando la información reportada a cada entidad.

Siendo la aprobación efectiva, se construye el flujo que carga la información, entonces irá a obtener las credenciales en un archivo cuya ruta conoce, y hará el respectivo cargue. Hubo hecho lo anterior, este enviará un correo masivo a las entidades por medio de Outlook, este es un conector de Power Automate, cuyo contenido tendrá una tabla HTML dinámica, reflejando la información reportada a cada entidad.

5.3. Interfaz Gráfica de Usuario

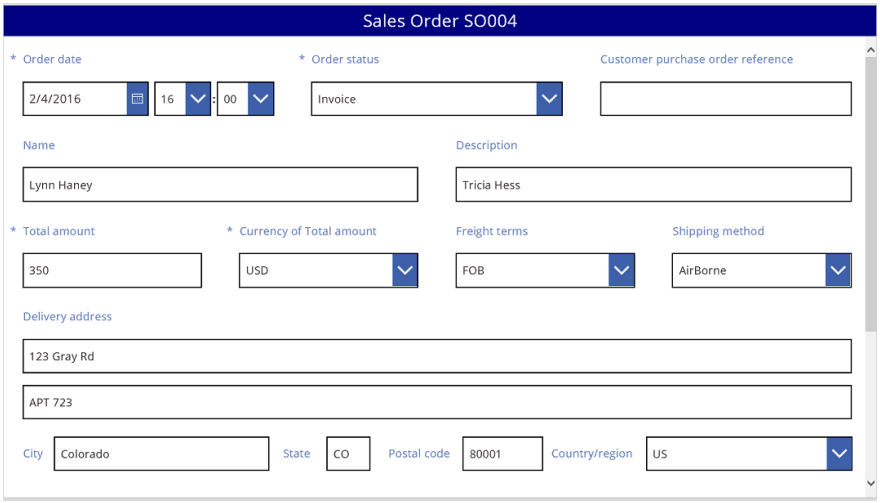
Para finalizar el desarrollo, los parámetros necesarios para la creación y cargue del reporte, se deben capturar por medio de una interfaz gráfica de usuario (GUI), para ello se utiliza la herramienta de Microsoft, Power Apps. Gracias a la facilidad que esta herramienta presta para la creación de interfaces se propone crear una de manera que sea un formulario (ver Figura 9) (tapanm-MSFT, 2024). Este formulario dará la posibilidad al usuario de digitar los parámetros requeridos para el reporte, y un botón cuya funcionalidad será ser el desencadenador de los flujos posteriores a estos, los cuales mantienen la lógica del proceso (ver Figura 10) (tapanm-MSFT, 2024).

5.4. Implementación de seguridad

Con el fin de proporcionar una seguridad eficiente al proyecto y proteger tanto las credenciales como las conexiones entre la máquia y las bases de datos, se investigó las capacidades de seguridad proporcionadas por Microsoft Power Platform y la implementación de encriptación de credenciales mediante librería en Python.

Figura 9.

Ejemplo Form Power Apps.



Nota: Ejemplo de la creación de un formulario para la captura de parámetros en Power Apps (tapanm-MSFT, 2024). Adaptación.

5.4.1. Roles y Permisos en Microsoft Power Platform

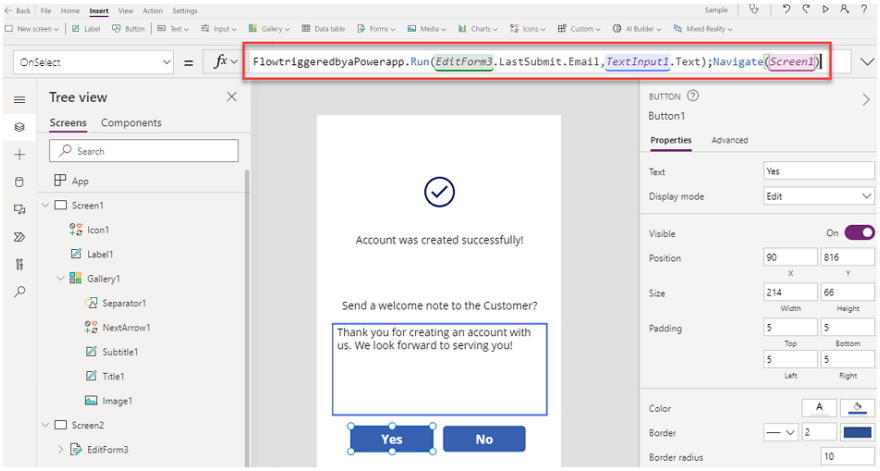
Microsoft Power Platform utiliza un sistema de roles basado en permisos que indica qué acciones puede realizar un usuario en cada entorno (Microsoft, 2024d). Los entornos son creados para fines de desarrollo, pruebas o producción (Ver Figura 11). Estos roles se dividen en 3 principales categorías:

5.4.1.1. Administrador de entorno. Este rol tiene acceso completo al entorno, lo que incluye capacidad para gestionar usuario, flujos, aplicaciones e incluso configuraciones de seguridad.

5.4.1.2. Creador de aplicaciones. Un creador de aplicaciones puede crear y personalizar aplicaciones en el entorno, pero su capacidad, pero su alcance en otros aspectos del entorno es limitado.

Figura 10.

Botón que ejecuta un flujo



Nota: Ejemplo de la configuración de un botón como desencadenador de un flujo de Power Automate (tapanm-MSFT, 2024). Adaptación.

5.4.1.3. Usuario básico. Tienen permisos restringidos y solo pueden utilizar las aplicaciones y flujos a los que se les haya otorgado acceso, un administrador de entorno.

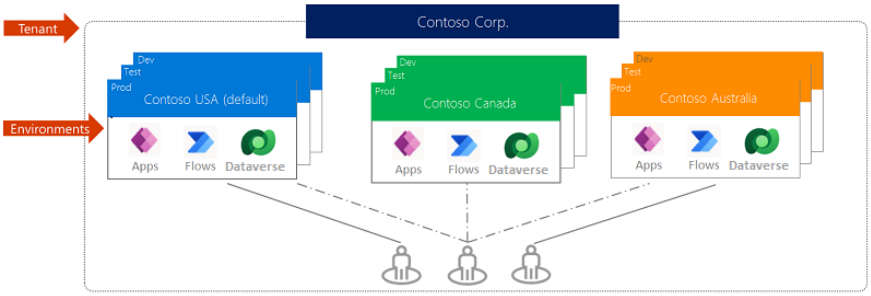
5.4.2. Conexión de Servicios

La comunicación entre los servicios de Microsoft, como Power Automate, Power Apps, Outlook o Sharepoint está protegida por medio de cifrado de datos en tránsito (TLS) (Ver Figura 12) y datos en reposo (AES-256), garantizando así que la información se mantenga protegida frente a accesos no autorizados en la transmisión y almacenamiento de esta (Microsoft, 2024c).

Por ello, es vital configurar de manera correcta los roles de los usuarios, con el fin de evitar la exposición no deseada de datos sensibles. En entorno de desarrollo y pruebas, es común tener

Figura 11.

Gestión de entornos en Power Platform.



Nota: Implementación de seguridad por medio de entornos administrados en Power Platform (Microsoft, 2024d). Adaptación.

un mayor acceso, pero en entornos de producción es recomendable restringir los permisos solo a usuarios esenciales (Microsoft, 2024d).

5.4.3. Encriptación de credenciales

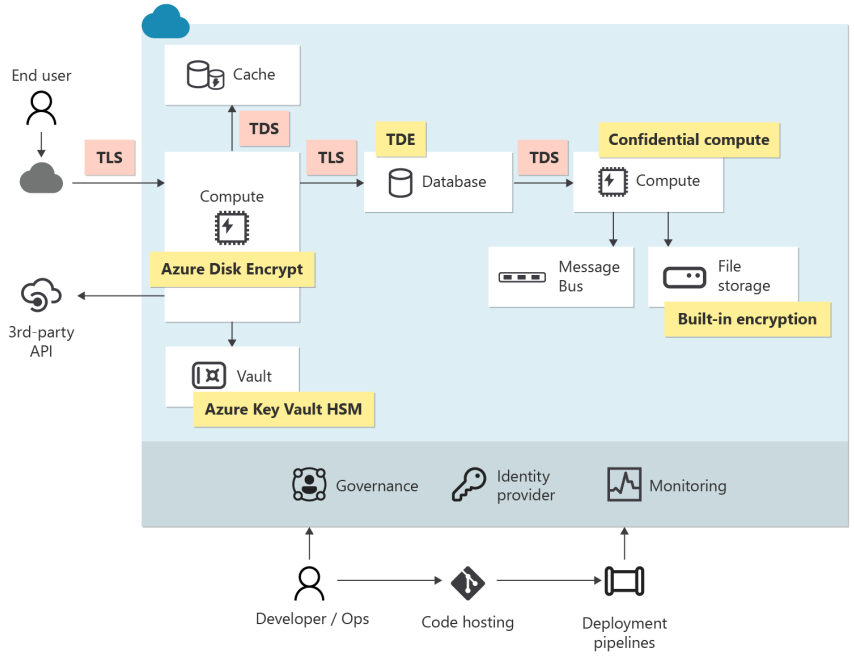
Para asegurar las credenciales de acceso a las bases de datos como SAC o JD Edwards, se propone un sistema de encriptación basado en Python que utiliza cifrado simétrico (AES-256) a través de la biblioteca cryptography (Ver Figura 13). Este enfoque asegura que las credenciales se almacenen en un archivo JSON encriptado, y que únicamente sean desencriptadas durante la ejecución del Script (Microsoft, 2023).

5.4.3.1. Generación de clave. Una clave de encriptación se genera una única vez y se almacena en un archivo protegido (Microsoft, 2023).

5.4.3.2. Almacenamiento seguro. El archivo que contiene la clave, debe ser guardado en una ubicación segura, cuya autorización será restringida para su acceso (Microsoft, 2023).

Figura 12.

Cifrado de datos en Power Platform.



Nota: Cifrado de datos en tránsito (TLS) usado por Microsoft Power Platform (Microsoft, 2024c). Adaptación.

5.4.3.3. Encriptación de credenciales. Durante el desarrollo del proyecto, las credenciales se encriptan usando esta clave previamente gestionada y se almacenan en un archivo JSON encriptado (Microsoft, 2023) y (Foundation, 2024).

5.4.3.4. Desencriptación. Cuando se ejecuta el módulo del Script que requiere las credenciales, el archivo de claves es utilizado para desencriptar el contenido del archivo JSON (Microsoft, 2023) y (Foundation, 2024).

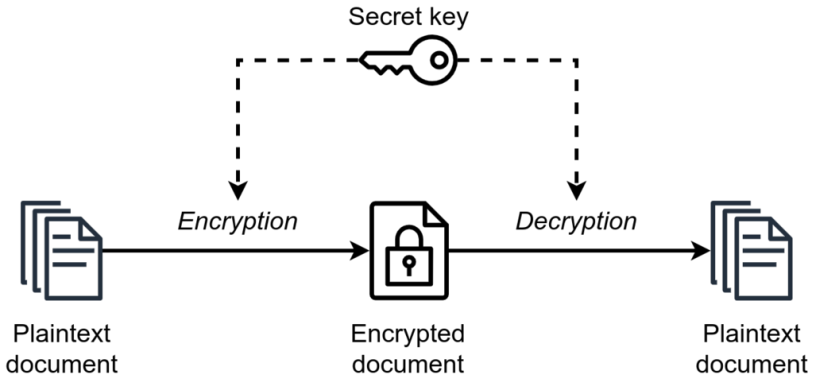
Es fundamental que el archivo de claves esté protegido con permisos adecuados. Solo el servi-

dor que ejecuta el Script debe tener acceso a estos archivos, lo cual puede gestionarse mediante permisos del sistema de archivos o almacenamiento de secretos locales (Oracle, 2024b).

Como alternativa, se propone el uso de variables de entorno con el fin de gestionar la clave de encriptación evitando almacenar la clave en un archivo. Esto reduce los riesgos de exposición en sistemas compartidos. Las variables de entorno se configuran al nivel del sistema operativo, de modo que las claves nunca se almacenan directamente en los archivos de código (Microsoft, 2024b).

Figura 13.

Encriptación Simétrica.



Nota: Encriptación simétrica usada por la librería Criptography Fernet (Cryptography, 2024). Adaptación.

5.4.4. Conexión entre máquinas

Con el fin de garantizar la seguridad de la conexión entre el servidor donde será hospedada la solución y las bases de datos (SAC y JD Edwards) como se definió anteriormente, el uso de la librería Python-oracledb en modo Thick que soporta el protocolo SSL/TLS para el cifrado de las

conexiones (Microsoft, 2024a).

Debido a que la solución será desplegada en un entorno corporativo, el cual usa redes privadas virtuales (VPN) o túneles seguros para la conexión a bases de datos, especialmente en entornos remotos. Y el uso de firewalls para limitar qué IPs pueden acceder a la base de datos, consolidad la seguridad eficiente de la conexión (Open Web Application Security Project (OWASP), 2024) y (Real Python, 2024).

5.5. Pruebas y Conclusiones

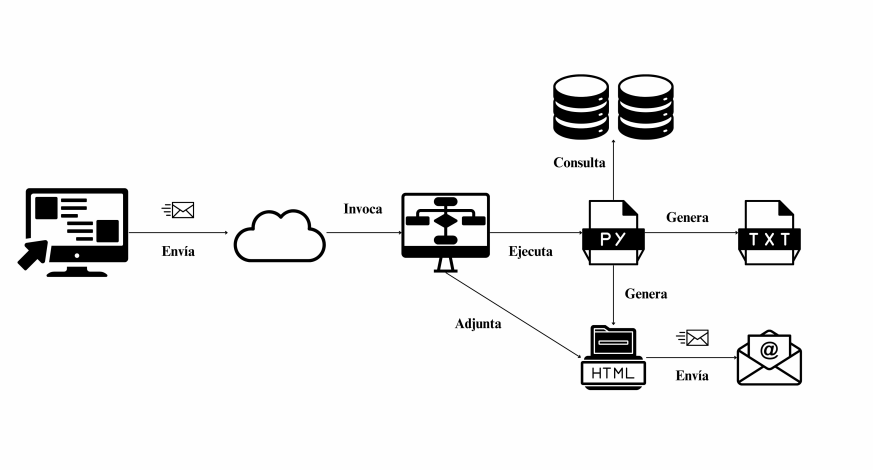
Finalmente, teniendo la estructura planteada, se harán pruebas funcionales, verificando información con reportes contraídos y cargados anteriormente, atrapando excepciones y posibles errores que los flujos, o el Script puedan presentar, esto con el fin de que el mantenimiento de esta solución a futuro sea simple. Teniendo un testeo perfecto, se propone buscar técnicas para obtener un despliegue eficaz junto con la estabilización del proyecto en producción, dando lugar al usuario generar los reportes y sus respectivas verificaciones. Como trayecto final, con el proyecto en producción, la creación de un manual técnico de uso es vital ya que esto se procurará en el tiempo, permitiendo un mantenimiento preventivo útil dejando la puerta abierta para futuros cambios.

6. Desarrollo

La arquitectura que se llevó a cabo para la realización de la solución fue la siguiente:

Figura 14.

Arquitectura.



Nota: Arquitectura completa de la solución desarrollada. León, J. (2024). Diseño propio.

Para el desarrollo de este trabajo se realizó una investigación y validación a fondo con el fin de garantizar la integración entre las herramientas y su correcto uso, teniendo en cuenta impedimentos a nivel empresarial, técnico y basándose en las reglas de negocio previamente socializadas con el usuario y su necesidad. Esta investigación da cabida a la integración del proyecto cuyo punto de partida es la lógica de la información, iniciando así con la construcción y diseño del Paquete Python.

6.1. Paquete Python

Inicialmente se tuvo planteado la creación de un Script de Python que llevara la lógica completa, sin embargo, en aras de la organización, captura de errores y buenas prácticas, se desarrolla un **Paquete Python** (Beazley, 2013; Lutz, 2013). Se refiere a una colección o conjuntos de Scripts que están estructuralmente conectados. Algunos términos que permiten describir la estructura son:

- **Proyecto Python:** Se refiere al conjunto de Scripts o módulos que, al trabajar integrados, conforman una aplicación o sistema. Puede incluir archivos de Python `.py` con archivos individuales encargados de otras funcionalidades, como archivos de configuración, conexión, documentación y otros recursos (Martelli, 2006).
- **Módulo:** Cada Script `.py` es técnicamente hablando, un módulo en Python. Un módulo es cualquier archivo de código Python, que define clases, funciones y variables, y puede ser importado por otros scripts para su uso (Pilgrim, 2004).
- **Paquete:** Un paquete en Python es una colección de módulos organizados en directorios, que incluyen el archivo `init.py` para que Python los trate como un paquete importable. Los paquetes permiten estructurar proyectos de manera que las funcionalidades estén bien organizadas (Beazley, 2009).
- **Biblioteca (Library):** Se refiere a un conjunto de módulos o paquetes que proporcionan funcionalidades específicas y están diseñados para ser reutilizables en otros proyectos (van Rossum, 2001).

Un paquete Python ofrece las siguientes ventajas sobre un Script monolítico:

1. **Mantenibilidad**

    - **Código más organizado:** Dividir la lógica en módulos hace que el código sea más fácil de entender y mantener. Cada módulo tiene una responsabilidad específica, lo que facilita realizar modificaciones o correcciones sin afectar otras partes del sistema (Lutz, 2013).
    - **Modularización:** Al tener diferentes funcionalidades separadas, es más sencillo localizar errores o añadir nuevas funcionalidades sin necesidad de revisar o modificar todo el código (Beazley, 2013).

2. **Reutilización de código**

    - **Modularidad:** Los módulos pueden reutilizarse en otros proyectos; solo se requiere exportación, evitando la duplicidad de código y esfuerzos (Pilgrim, 2004).
    - **Paquetes compartidos:** Los paquetes tienen la posibilidad de ser compartidos con otros desarrolladores o bien distribuirse públicamente, lo que facilita la creación de librerías utilizables (van Rossum, 2001).

3. **Escalabilidad**

    - **Facilidad de expansión:** A medida que el proyecto crece en tamaño y complejidad, es más fácil agregar nuevas funcionalidades creando nuevos módulos en lugar de aumentar la longitud de un único Script (Beazley, 2013).

- **Evitar sobrecarga de funciones:** Con un solo Script, las funciones acaban por ser bastante largas y difíciles de gestionar, incluso llegando al punto de recrear funciones que ya estaban diseñadas (Lutz, 2013).

4. **Pruebas y depuración**

    - **Pruebas unitarias:** Es más sencillo probar individualmente cada módulo y sus funcionalidades en un paquete modulado (Pilgrim, 2004).
    - **Depuración sencilla:** Llegado el caso en que suceda un error, localizar el problema es mucho más sencillo en un sistema modular. Es posible depurar un módulo específicamente en lugar de todo el sistema (Lutz, 2013).

Teniendo en cuenta la viabilidad de realizar la lógica del Script modularmente en un Paquete, se da por inicio la creación de cada uno de estos, cada módulo está ligado con el principal, cuyo módulo mantiene el flujo de trabajo principal. Es este quien llama a cada uno de los módulos consiguientes para procesar la información. (Ver Figura 15)

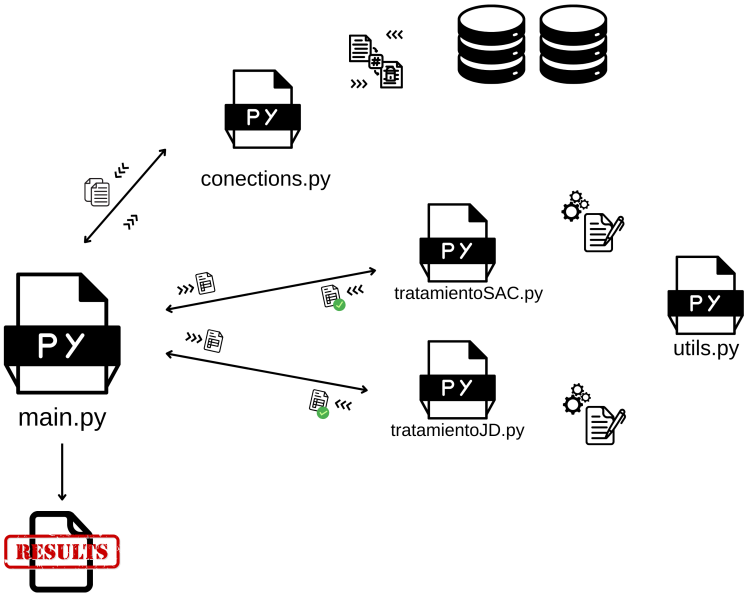
6.1.1. Módulo Principal

Este módulo debe llevar la lógica completa del proceso según lo planteado por un paquete Python, por ello se desarrolla con la siguiente estructura:

El módulo principal `main.py` contiene en su cuerpo la referencia de todos los módulos, pues utiliza clases, funciones y variables de estos. Este módulo importa librerías como `Pandas` como

Figura 15.

Gráfico de la estructura del Paquete Python.



Nota: Gráfico estructural de la integración modular del paquete de Python. León, J. (2024). Diseño propio.

`pd`, `Python-Oracledb` como `odb` y `Json`.

`ReportGenerator` es la clase principal de este módulo, cuya función principal recibe como parámetros:

- `year` - VARIABLE DE ENTRADA - *Tipo de dato: Numérico*
- `trimester` - VARIABLE DE ENTRADA - *Tipo de dato: Numérico*
- `sac_config_encrypted_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*
- `jd_config_encrypted_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*

- `params_sac_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*
- `params_sac_str_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*
- `params_jd_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*
- `params_jd_F0902_path` - VARIABLE DE RUTA - *Tipo de dato: Texto*

Esta clase consta de los siguientes métodos o funciones para su funcionamiento:

1. **Función Principal `__init__():`** Esta función se encarga de crear el objeto `self`, es un parámetro especial que permite ser utilizado por los métodos de la clase. Representa la inspancia actual que está siendo manipulada, esto permite que cualquier método pueda acceder a los atributos definidos en su creación. Teniendo en cuenta esto, la función carga archivos `.json` hospedados en una carpeta de configuración ubicada en el paquete, por medio de funciones hospedadas en el módulo `utils.py`, obteniendo así en variables tipo diccionario, los parámetros, queries y credenciales. Esta función llama a otras dos funciones para actualizar los parámetros de las consultas.
2. **Función actualización parámetros SAC `update_sac_params():`** Esta función parte de una función anidada que calcula los parámetros requeridos para cada query, ya que estos son generalmente: `fecha_inicio` y `fecha_final`, y el parámetro `trimester` y `year` indica únicamente el trimestre y año de la consulta, respectivamente. Adicional esta función toma

los nuevos la salida de la función `calculate_sac_params()` y actualiza los parámetros en el archivo `.json`

3. **Función actualización parámetros JD `update_jd_params()`:** De manera similar a la función de actualización de parámetros SAC, este llama a la función `calculate_jd_params()`, que calcula los parámetros requeridos por el query partiendo desde las variables de entrada `year` y `trimester`, actualizando directamente a la variable tipo diccionario que contiene los párámetros que se usarás posteriomente en la consulta a la base de datos.

4. **Función generación reporte SAC `generate_sac_report()`:** Esta función inicialmente garantiza que no exista un archivo con los argumentos proporcionados, ya que de existir, tomaría este en lugar de hacer consulta, por temas de migración de datos en las bases de datos, en sesiones posteriores se explicará mejor. La conexión a la base de dato para hacer consultas, es inicializada por esta función, llamando a la clase principal del módulo `conections.py` llamada `OracleConnection`, haciendo enlace y consulta con el método `execute_query()`. Este módulo devuelve en formato DataFrame (tipo de dato en la librería Pandas) y este para fines de limpieza, tranformación, ajuste y modelado, se envía como argumento a la función principal `process()` de la clase princpal del módulo `tratamiento_sac.py` llamada `SACProcessor`, devuelve el dataframe listo para concatenarse con la parte faltante de JD y usa la función `save_report()` para guardar el archivo en una ruta interna y una compartida con el usuario.

5. **Función generación reporte JD `generate_sac_report()`:** Esta función inicialmente

garantiza que no exista un archivo con los argumentos proporcionados, ya que de existir, tomaría este en lugar de hacer consulta, por temas de migración de datos en las bases de datos. La conexión a las bases de datos para hacer consultas, es inicializada por esta función, llamando a la misma clase usada por el SAC, el proceso posterior persigue la misma lógica que el proceso del SAC.

6. **Función inicializadora flujo completo `run():`** Esta función está encargada de enlazar, ejecutar la generación de los reportes y la combinación de estos para obtener como resultado el reporte limpio y en el formato exigido por la entidad regulatoria **Condaturía General de la Nación (CGN)**, adicional a esto los archivos son guardados en una ruta compartida específica en la organización, cuyo ingreso es autorizado y solo tiene acceso el usuario técnico donde está desplegada la solución, Posterior a esto, se ejcutar un método que instanciado del módulo `utils.py` el cual crea una tabla HTML por cada cuenta contable asociada a una entidad en específico, lo guarda en una ruta específica con el Código de Entidad CGN. Este método será mejor explicado en la sesión dedicada a ese módulo.

7. **Punto de entrada de la ejecución `if __name__ == "__main__":`** Este bloque permite ejecutar el Script desde la línea de comandos, recibiendo como argumentos el año y el trimestre para generar los reportes correspondientes. Tiene uso en la línea de compandos con la siguientes sentencia de ejemplo:

```
python main.py --year 2024 --trimester 1
```

El uso de este punto de entrada es una buena práctica en Python para separar la lógica que debe ejecutarse al correr el script desde la línea de comandos, versus cuando solo quieres importar el archivo para reutilizar sus funciones o clases en otro lugar. Esto ayuda a modularizar el código.

Se hace uso del módulo `argparse` ya que es la manera estándar en Python para manejar parámetros de línea de comandos.

```
parser = argparse.ArgumentParser(description="Generador de
↪ Reportes")
```

Se crea un objeto `ArgumentParser` que maneja los argumentos de entrada del programa. Se especifica un texto descriptivo del script, que se muestra si el usuario ejecuta el script con la opción `--help`

```
    parser.add_argument("--year", type=int, required=True, help="Año del reporte (por ejemplo, 2024)")
    parser.add_argument("--trimester", type=int, required=True, help="Trimestre del reporte (por ejemplo, 1)")
```

`--year`: Define que el script debe recibir ese argumento que será un entero (representando el año del reporte) y es obligatorio (`requiered=True`)

`--trimester`: Define que el script debe recibir ese argumento que será un entero (representando el trimestre del reporte) y es obligatorio (`requiered=True`)

```
    args = parser.parse_args()
```

Aquí, argparse toma los argumentos que el usuario introduce en la línea de comandos y los almacena en el objeto args, donde `args.year` y `args.trimester` contienen los valores que el usuario ha introducido.

argparse facilita la captura de parámetros y su validación, permitiendo que el script sea flexible y reutilizable sin necesidad de modificar el código cada vez que se requiere cambiar los parámetros.

8. **Función generadora `ReportGenerator()`:** Se encarga de recibir los archivos de configuración, procesar los parámetros y generar el reporte.

```
    generator = ReportGenerator(
        'F:/Reporte Entidades
        ↪ CGN/archivos/config/db_sac_encrypted.json',
        'F:/Reporte Entidades CGN/archivos/config/db_jd_encrypted.json',
        # ...
        args.year,  # Año ingresado por el usuario
        args.trimester  # Trimestre ingresado por el usuario
    )
```

Partiendo de este módulo principal y siguiendo la lógica de se trabajo de ejecución, se desarrolla el módulo de conexión:

6.1.2. Módulo Conexión

El módulo `conections.py` contiene en su cuerpo la importa las librerías `Pandas` y `Python-Oracledb`.

`OracleConection` es la clase principal del módulo, cuya función principal recibe los siguientes parámetros

- `config` - VARIABLE TIPO DICCIONARIO - *Tipo de dato: Compuesto*

El diccionario de config tiene una estructura de la siguiente manera (Ejemplo es la configuración de la base de datos SAC):

```
{
    "sac_query": "F:/Reporte Entidades CGN/archivos/queries/query_sac.sql",
    "sac_params": {
        "firstdate": "202404",
        "lastdate": "202406"
    },
    "sac_report_name": "//essa-file08/6210-E-GC/ARCHIVOS SCRIPT - REPORTE CGN/Reportes SAC/SAC-{{firstdate}}-{{lastdate}}.xlsx",
    "sac_report_name_grouped": "F:/Reporte Entidades CGN/reportes/Reportes SAC/SAC-{{firstdate}}-{{lastdate}}.txt"
}
```

El query tiene la siguiente forma (Consulta de ejemplo de la base de datos SAC):

```
SELECT s.cliente_id,
       s.codigo_entidad,
       TO_CHAR(TO_DATE(v.fecha_generacion, 'J'), 'YYYY-MM') AS periodo,
       SUM(DECODE(v.corr_cambio, 0, DECODE(v.empresa, 'E', v.valor_concepto,
        ↪  0), 0)) AS valor_empresa
FROM   v_multitabla b,
       oficiales o,
       v_cli_fac_dia_his v,
       clientes s
WHERE  b.tabla            (+) = 'BARRIOS'
AND    b.codigo_car       (+) = s.municipio
AND    b.codigo_num       (+) = s.barrio
AND    v.cliente_id       = s.cliente_id
AND    v.fecha_generacion BETWEEN TO_CHAR(TO_DATE(:firstdate, 'YYYYMM'), 'J')
                          AND TO_CHAR(LAST_DAY(TO_DATE(:lastdate,
                           ↪  'YYYYMM')), 'J')
AND    s.codigo_entidad IS NOT NULL
AND    o.codigo_entidad (+) = s.codigo_entidad
GROUP BY s.cliente_id,
         s.codigo_entidad,
         TO_CHAR(TO_DATE(v.fecha_generacion, 'J'), 'YYYY-MM')
ORDER BY s.codigo_entidad,
         s.cliente_id
```

Esta clase consta de los siguientes métodos o funciones para su funcionamiento:

1. **Función Principal `__init__()`:** esta función se encarga de crear el objeto self, y acceder a la configuración que ya viene desde el módulo principal, accede a las credenciales de este, como el usuario y contraseña de conexión, determina si se debe usar el Identificador de Sistema (SID) o el Nombre del servicio (Service Name). El SID es un identificador único e una instancia de base de datos Oracle. Una instancia es el conjunto de procesos y memoria

que permiten la ejecución de la base de datos. Este identificador es usado particularmente en bases de datos de una sola instancia, por el contrario el Service Name es usado el entornos de múltiples instancias asociadas a la base de datos, permitiendo así conexiones de manera más flexible, ya que el servicio puede distribuirse en varias instancias. Por lo que esta función permite conexión con una o múltiples instancias. (Ver Tabla 1)

Tiene en cuenta también el modo de la conexión, indicada en la configuración, permitiendo así la conexión de modo grueso o de modo delgado, dependiendo de las características de la base de datos, e inicializando las librerías instantáneas de cliente llegado el caso en el que se especifique el modo grueso, que es por medio de un valor booleano en el diccionario de configuración.

Aspecto	**SID**	**Service Name**
Significado	Identificador único de una instancia específica.	Identifica el servicio lógico de la base de datos.
Uso principal	Para instancias individuales de bases de datos.	Para bases de datos en entornos multi-instancia o RAC.
Escenarios comunes	Entornos single-instance (bases de datos simples).	Entornos multi-instance como Oracle RAC o alta disponibilidad.
Capacidad de redirección	No permite redirección a otras instancias.	Permite redirección a otras instancias si la original falla.
Balanceo de carga	No soporta balanceo de carga.	Soporta balanceo de carga entre instancias en RAC.

Tabla 1
Comparación entre SID y Service Name.

2. **Función ejecución consulta `execute_query()`:** Este método recibe como parámetros la consulta SQL y los parámetros que son inicializados como `None`, ya que no es un valor

requerido si la consulta no los requiere. Esta función inicializa la conexión y el curso en `None` por buenas prácticas, y realiza un `try - except` para atrapar cualquier error de conexión directo con la base de datos. En try hace la conexión por medio de la función `connect()` de la librería `odb` diligenciando las credenciales: usuario, dontraseña, dsn: SID o ServiceName. El dsn es creado por una función nativa de la librería `odb`, `makedsn()` la cual recibe como argumentos el **Host**, el **Puerto** y el **SID** o **ServiceName**

Una vez establecida la conexión, se obtiene un cursor, que es un objeto que permite ejecutar comandos SQL sobre la conexión. Realiza la ejecución de la consulta SQL, llegado el caso en que le pase en el argumento de la función los parámetros de la consulta, los pasa en la ejecución, de lo contario hace la consulta solamente.

```
    # Establecer la conexión con la base de datos
    connection = odb.connect(user=self.username, password=self.password,
    ↪  dsn=self.dsn)
    cursor = connection.cursor()

    # Ejecutar la consulta, con o sin parámetros
    if parameters is not None:
        cursor.execute(query, parameters)
    else:
        cursor.execute(query)

    # Obtener los resultados de la consulta
    rows = cursor.fetchall()

    # Obtener los nombres de las columnas para construir el DataFrame
    columns = [col[0] for col in cursor.description]
    df = pd.DataFrame(rows, columns=columns)
```

Posterior a esto obtiene los resultado de la consulta en filas y columnas y lo convierte en un DataFramer por medio de la `Pandas`.

Finalmente cierra el cursor y la conexión a la base de datos, por buenas prácticas.

6.1.3. Módulo Tratamiento SAC

El módulo `tratamiento_sac.py` contiene en su cuerpo la referencia del módulo `utils.py` e importa la librería `Pandas` como `pd`

`SACProcessor` es la clase principal de este módulo, cuyo método principal recibe como parámetros:

- `df` - VARIABLE DE RESULTADO - *Tipo de dato: DataFrame (pandas)*
- `df_str` - VARIABLE DE RESULTADO - *Tipo de dato: DataFrame (pandas)*

Para entender el funcionamiento de este método se requiere tener en cuenta qué información trae consigo los DataFrames, la información proveniente de la base de datos SAC (Sistema de Administración Financiera) son tablas que recopilan información sobre cuentas de energía a nombre de entidades en específico, para efectos de reporte de lo facturado en determinado periodo de tiempo, adicional a esto tiene información referente bien sea a la cuenta o a la entidad, tiene las siguiente columnas:

- `CLIENTE_ID` - Cuenta de Energía - *Tipo de dato: object*
- `CODIGO_ENTIDAD` - Código de Entidad Interno - *Tipo de dato: object*
- `PERIODO` - Periodo de Facturación - *Tipo de dato: date*
- `VALOR_EMPRESA` - Valor Facturado - *Tipo de dato: int*

Este método se encarga principalmente de tratar los datos con métodos nativos de la librería `pandas`, lo métodos más usados son:

- Remolición de filas nulas en una columna en específico. `pd.dropna(subset=COLUMN)`
- Carga de archivos adicionales de relación. `pd.read_excel()`

- División de DataFrames por un valor de una columna en específico.
- Concatenación de DataFrames. `pd.concat()`
- Cruce con otros archivos externos por una o varias columnas que los relaciona. `pd.merge([COLUMNS])`
- Agrupación por una o varias columnas en específico. `pd.groupby([COLUMN])`
- Eliminación de duplicados. `pd.drop_duplicates()`
- Cambio de tipo de dato.

Ya que este método cruza la información con tablas puente que relacionan un código con otro que viene siendo un identificador para la entidad regulatoria, carga estos archivos Excel (.xlsx) inicialmente con la función `readExcel()` del módulo `utils.py` poniendo la ruta donde este se ubica y el nombre de la hoja donde está la información. Relaciona por medio de una columna las dos tablas y se trae el dato necesario.

Debido a que hay algunas cuentas registrada cuyo entidad a su nombre se desconoció en el momento de registrarse, se ingresó un **Código Ficticio** o también conocido como **Dummy Value**, por ello se separa el dataframe por estos valores y se cruzan con una tabla que hace un relacionamiento directo de la cuenta con el código de entidad, haciendo inválido este valor Dummy para identificar.

La otra parte del DataFrame, se cruza su valor identificador de entidad por medio de una tabla que relaciona este valor con el código de entidad oficial, estos relacionamientos son hechos en estas tablas con valores expedidos por la entidad regulatoria.

Posterior a este proceso, se deben filtrar las entidades por un archivo llamado **Directorio ECP** expedido directamente por la entidad regulatoria, para indicar cuales son las entidades oficiales que se deben reportar, todas aquellas que no estén allí, no deben reportarse. Estas cuentas homologan a la cuenta contable: 4.3.15.20 expedidad por la entidad regulatoria.

Para que estos cruces se den sin problema, se deben mantener el mismo tipo de datos de la columna que relaciona el cruce, hay que garantizarlo antes de.

Finalmente en el procesamiento se llama a la función `str_acc()` la cual hace tratamiento similar al anterior, pero a diferencia de que el DataFrame que procesa `df_str` es extraído de la base de datos del SAC pero por medio de otro query que consulta a otras tablas, son cuentas adicionales que no son mostradas por medio del otro query, se hace el procesamiento y el formato, finalmente se concatena al DataFrame inicial.

Adicional al método incial esta clase tiene los siguientes métodos:

1. **Función Cuentas Adicionales `str_acc()`:** Esta función trata un DataFrame que es extraído por una consulta diferente que extrae información de la misma base de datos pero crea una vista que viene de otras tablas, esta información adicional tiene las siguientes columnas:

    - `CLIENTE_ID` - Cuenta de Energía - *Tipo de dato: object*
    - `NIT` - Código NIT de entidad - *Tipo de dato: object*
    - `PERIODO` - Periodo de Facturación - *Tipo de dato: date*

- `VALOR_EMPRESA` - Valor Facturado - *Tipo de dato: int*

A diferencia de las otras cuentas, este tratamiento de datos se hace búsqueda del valor del Código de Entidad por medio de una tabla que relaciona el NIT con el Código CGN. Posterior a ello se trae se le da formato y se crea una columna llamada CUENTA CGN, ya que todo se reporta a la cuenta de energía 4.3.15.19.

2. **Función Final `finalize()`:** Esta función se encarga de dar formato al DataFrame, un formato expedido por la CGN el cual lleva la siguiente estructura:

   - `CUENTA CGN`: Cuenta contable de reporte a CGN
   - `CODIGO CGN`: Código de entidad de reporte a CGN
   - `VALOR CORRIENTE`: Valor de importe a corto plazo
   - `VALOR NO CORRIENTE`: Valor de importa a largo plazo

   Debido a que la particularidad de las cuentas de energía es a largo plazo, los valores de `VALOR_EMPRESA` van a `VALOR NO CORRIENTE`, y el `VALOR CORRIENTE` va en ceros.

6.1.4. Módulo Tratamiento JD

El módulo `tratamiento_jd.py` contiene en su cuerpo la referencia del módulo `utils.py` e importa la librería `Pandas` como `pd`

`JDProcessor` es la clase principal de este módulo, cuyo método principal recibe como parámetros:

- `df` - VARIABLE DE RESULTADO - *Tipo de dato: DataFrame (pandas)*
- `df_cuentas_faltantes` - VARIABLE DE RESULTADO - *Tipo de dato: DataFrame (pandas)*

Para entender el funcionamiento de este método se requiere tener en cuenta qué información trae consigo los DataFrames, la información proveniente de la base de datos JD Edwards son tablas que recopilan información sobre cuentas contables a nombre de entidades en específico, para efectos de reporte de los saldos en determinado periodo de tiempo, adicional a esto tiene información referente a la cuenta o a la entidad, para más exactitud, el DataFrame tiene las siguiente columnas:

- `CUENTA OBJETO` - Cuenta Interna - *Tipo de dato: object*
- `CUENTA AUXILIAR` - Cuenta Auxiliar Identificador - *Tipo de dato: object*
- `PERIODO` - Periodo de facturación - *Tipo de dato: date*
- `TIPO DE DOCUMENTO` - Documento donde se factura la cuenta - *Tipo de dato: object*
- `NÚMERO DE DIRECCIÓN` - Tercero al que se le factura - *Tipo de dato: object*
- `CÓDIGO DE CATEGORÍA 25` - Cuenta Contable CGN - *Tipo de dato: object*
- `IMPORTE` - Valor de facturazión- *Tipo de dato: int*

Este método se encarga principalmente de tratar los datos con métodos nativos de la librería `pandas`, lo métodos más usados son:

- Remolición de filas nulas en una columna en específico. `pd.dropna(subset=COLUMN)`
- Carga de archivos adicionales de relación. `pd.read_excel()`
- División de DataFrames por un valor de una columna en específico.
- Concatenación de DataFrames. `pd.concat()`
- Cruce con otros archivos externos por una o varias columnas que los relaciona. `pd.merge([COLUMNS])`
- Agrupación por una o varias columnas en específico. `pd.groupby([COLUMN])`
- Eliminación de duplicados. `pd.drop_duplicates()`
- Cambio de tipo de dato.

Ya que este método cruza la información con tablas puente, archivos de conciliación relacionadas por medio de varios campos, carga los siguientes archivo Excel (.xlsx)

- `Reglas de eliminación.xlsx` - Conjunto de cuentas contables que se deben reportar según la entidad regulatoria.
- `Directorio ECP.xlsx` - Conjunto de entidades que se deben reportar según la entidad regulatoria.
- `Tabla Puente SAC - JD.xlsx` - Este archivo contiene en diferentes hojas las tablas que relacionan identificadores internos de cuenta (Cuenta Objeto) o entidad (Tercero, An8, Nú-

mero de Dirección, Nit) con identificadores externos para efectos de reporte, expedidos por la entidad regulatoria.

- `cuenta 580140.xlsx` - Cuenta especial debido a su particularidad debe digitarse manualmente. (su naturaleza no permite parametrización)
- `cuenta 230615.xlsx` - Cuenta especial debido a su particularidad debe digitarse manualmente. (su naturaleza no permite parametrización)

Por otro lado también carga información de conciliación ya que algunas cuentas de algunas entidades deben ser conciliadas por trámites internos con la organización, que es entregada por otros organismos de la organización:

- `Conciliaciones Filiales.xlsx` - Conjunto de cuentas 1, 2, 4, 5 y 7 conciliadas a nombre de las filiales de la organización.
- `Conciliaciones Municipios.xlsx` - Conjunto de saldos a la cuenta 244004 a nombre de los municipios a los que le factura la organización.
- `Conciliaciones Cuentas por Cobrar` - Conjunto de cuentas 1 por cobrar a nombre de diferentes entidades que requieren ser conciliadas.
- `Conciliaciones Cuentas por Pagar` - Conjunto de cuentas 1 y 2 por pagar a nombre de diferentes entidades que requieren ser conciliadas.

- `Conciliaciones Costos` - Conjunto de saldos a la cuenta 636005 a nombre de entidades que requieren conciliación.

De este trámite se encarga la función principal, adicional a esto esta función llama a los siguientes métodos para su funcionamiento:

1. **Proceso Inicial `proceso_inicial()`:** En el proceso incial se le incluye el Código de Entidad CGN al DataFrame por medio de el cruce con la tabla puente que relaciona **Tercero** con **Código CGN** y elimina aquellas filas cuyo tercero no figure en este cruce. Adicional a esto quita cuentas o tipos de documento que no se deben reportar, ya que son información que no es requerida. Garantiza en este proceso mantener el valor de las cuentas, puesto que las cuentas 2 y 4 son de naturaleza negativa, y las cuentas 1, 5, y 7 son de naturaleza positiva, por lo tanto depura teniendo en cuenta esta condición. Por regulación de la entidad CGN se requiere que todas las cuentas 7 homologuen a la cuenta contable 636005 y garantiza que los importes sea valor corriente o valor no corriente dependiendo del tipo de cuenta. Las cuentas 4, 5, y 6 ahora que las 7 homologan, son valor no corriente (a largo plazo) mientras que las cuentas 1 y 2 son valor corriente (a corto plazo).

   Cruce con tabla relación Tercero - Código CGN

```
    df = pd.merge(df, terceros, on='Numero de Direccion', how='left')
    df = df.dropna(subset=['Codigo Consolidacion CGN'])
```

Depuración por Signo:

```
mask1 = df['CUENTA OBJETO'].str.startswith(('1', '5', '6')) & (df['SALDO'] < 0)
df = df[~mask1]
mask2 = df['CUENTA OBJETO'].str.startswith(('2', '4')) & (df['SALDO'] > 0)
df = df[~mask2]
```

Homologación cuentas 7:

```
df['CUENTA CGN'] = df['CUENTA CGN'].apply(lambda x: '636005' if x.startswith('7') else x)
```

Ajuste del saldo para `VALOR CORRIENTE` y `VALOR NO CORRIENTE`

```
# Si la cuenta objeto es 4, 5, 6 o 7 su valor es no corriente
df.loc[df['CUENTA OBJETO'].str.startswith(('4', '5', '6', '7')), 'VALOR NO CORRIENTE'] = df['SALDO']

# Si la cuenta objeto es 1 o 2, su valor es no corriente
df.loc[df['CUENTA OBJETO'].str.startswith(('4', '5', '6', '7')), 'VALOR CORRIENTE'] = 0.0
```

Finalmente minimiza el dataframe únicamente a las columnas necesarias , filtra cuentas con

movimientos en cero y agrupa las cuentas por cuenta y entidad.

```
# Minimizacion del dataframe
df = Dataframe_Minimization(df, ['CUENTA CGN','CUENTA OBJETO','CODIGO CGN','VALOR CORRIENTE', 'VALOR NO CORRIENTE', 'TERCERO', 'AUXILIAR'])

df = NoAccountMove(df)

df = df.groupby(['CUENTA CGN', 'CODIGO CGN']).agg({
    'VALOR CORRIENTE': 'sum',
    'VALOR NO CORRIENTE':'sum',
    'CUENTA OBJETO': 'first',
    'TERCERO': 'first',
    'AUXILIAR': 'first'
}).reset_index()
```

Debido a que hay cuentas que van a ser conciliadas, se deben eliminar del DataFrame para posterior inclusión.

Cada conciliación requiere tener los campos del DataFramer para hacer una exitosa inclusión, sin embargo la mayoría de estos no la tiene, por ello cada método se encarga de ajustar la conciliación para su concatenación.

2. **Excepción Cuentas Acueducto `cuentas_acueducto()`:** Este método extrae las cuentas que están a nombre del acueducto y les hace un tratamiento distinto el cual está conciliado entre las entidades y la organización, de acuerdo al monto que asume cada entidad, se hace una desagregación por cuenta auxiliar para definir a que entidad se le reporta el saldo. Estos

son temas internos de la organización, por ende no se revelan. Sin embargo el ejemplo del procedimiento es así:

```
df['CODIGO CGN'] = df['Cuenta Auxiliar'].apply(lambda x: '239868001' if x in ['01010102','01020102'] else '923270864' if x == '01060101' else None)
df_511117 = df_511117.dropna(subset=['CODIGO CGN'])
```

3. **Proceso de conciliación con filiales `conciliacion_filiales()`:** Este médotodo recibe como parámetro el DataFrame, la conciliación, el puente de Terceros y de Cuenta CGN. Para el procesamiento de esta conciliación se requiere limpiar inicialmente tomando únicamente las columnas útiles, valores en cero o registros no útiles, también asegurando los tipos de datos para cruces posteriores, usando métodos definidos en anteriores módulos. Con la limpieza lista y un reporte de conciliación de filiales listo para cruzar, se obtiene el Código de Entidad CGN por medio del Tercero, y la Cuenta Contable CGN por medio de la cuenta objeto. Se realizan eliminación de lo no encontrado, eliminación de duplicados y agrupación de valores. Finalmente se minimiza el DataFrame de conciliación para que sea igual al original y se hace la concatenación.

4. **Proceso de conciliación con municipios `conciliacion_municipios()`:** Este método recibe como parámetro el DataFrame, la conciliación y la tabla puente de Terceros. Ya que esta conciliación es únicamente de la Cuenta Contable 244004, por ende solo se requiere

obtener el Códido de Entidad CGN, se hace por medio de la tabla puente relacionando el An8 con el Código de Entidad CGN, posterior a la limpieza de la conciliación, se minimiza el DataFrame de la conciliación y se concatena al DataFrame original.

5. **Proceso de conciliación con cuentas por cobrar** `conciliacion_cuentas_por_cobrar():` Este método recibe como parámetros el DataFrame, la conciliación, la tabla puente de Cuenta CGN y la tabla puente de NIT. Hace la limpieza respectiva siguiendo la lógica de los anteriores métodos, hace el cruce entre la Cuenta Objeto por medio de la tabla puente para traer la Cuenta Contable CGN, y elimina aquellas que no aparezcan en esta tabla, asímismo trae el Código de Entidad CGN por medio del relacionamiento del NIT, y elimina aquellos registro cuyo relacionamiento no se dió y trajo vacío el valor del Código de Entidad CGN. Similar a los anteriores métodos, minimiza el DataFrame de la conciliación, agrupa por cuenta y código para concatenarlo con el DataFrame Original.

6. **Proceso de conciliación con cuentas por pagar** `conciliacion_cuentas_por_pagar():` Este método recibe como parámetros, de manera similar al método de conciliación de cuenta por cobrar, el DataFrame, la conciliación,la tabla puente de Cuenta CGN y la tabla puente de NIT. Hace la limpieza respectiva siguiendo la lógica de los anteriores métodos, hace el cruce entre la Cuenta Objeto por medio de la tabla puente para traer la Cuenta Contable CGN, y elimina aquellas que no aparezcan en esta tabla, asímismo trae el Código de Entidad CGN por medio del relacionamiento del NIT, y elimina aquellos registro cuyo relacionamiento no se dió y trajo vacío el valor del Código de Entidad CGN. Similar a los anteriores métodos,

minimiza el DataFrame de la conciliación, agrupa por cuenta y código para concatenarlo con el DataFrame Original.

7. **Proceso de conciliación con costos `conciliacion_costos()`:** Esta conciliación debido a su naturaleza y arreglos internos, por ello ya viene lista para concatenar al DataFrame, y solo se le asegura los tipos de dato para que no genere problemas al concatenar.

```
    costos['CUENTA CGN'] = costos['CUENTA CGN'].astype(str).str.strip()
    costos['CODIGO CGN'] = costos['CODIGO CGN'].astype(str).str.strip()
    costos['VALOR CORRIENTE'] = costos['VALOR CORRIENTE'].astype(float)
    costos['VALOR NO CORRIENTE'] = costos['VALOR NO
    ↪  CORRIENTE'].astype(float)
    df_final = pd.concat([df, costos], ignore_index=True)
```

8. **Proceso final `finalize()`:** Este método hace la inclusión de la mayoría de las excepciones que se pueden encontrar en este tipo de reportes, haciendo concatenación de estos. Llama a métodos que se encuentran en el módulo `utils.py` que son inclusiones de cuentas que tienen desagregaciones específicas, incluye las cuentas adicionales que no salen en el query relacionado para esto, y por el contrario les hace un tratamiento bastante similar al de esta Clase, ordena los valores a petición de formato.

9. **Proceso de exclusión de cuentas de ajuste al peso `exclusion_cuentas_ajustes_al_pesos()`:** Ya que existen algunas cuentas que su naturaleza es para ajustar el peso de otros valores reportados para balance total, donde su reporte son montos muy pequeños para evitar errores

de contabilidad, esta cuentas según la entidad regulatoria no deben reportar, por ello este método recibe el DataFrame como argumento y le quita estas cuentas, de la siguiente manera:

```
df = df[~df['CUENTA CGN'].isin(['4.8.08.90', '5.1.11.79',
↪  '5.1.20.17', '5.8.90.90','4.8.08.63'])]
```

incluye una lista de cuentas donde se declara que el DataFrame no tenga valores en la columna CUENTA CGN que se encuentren allí.

6.1.5. Módulo Utilidades

Este módulo es un conjunto de métodos que son utilizados en los otros módulos, Son métodos creados con el fin de ser exportados y usados en otros contextos ya que organizan, facilitan y agilizan los procesos realizados en estos, Algunas de las funciones más usadas son:

1. **Lectura de archivo plano (`.json`) `load_json()`:** Este método permite leer archivos `.json` en una ruta específica.

```
def load_json(self, path):
    with open(path, 'r') as file:
        return json.load(file)
```

2. **Lectura de consulta SQL `.sql` `load_query()`:** Este método permite leer archivos con consultas `.sql` en una ruta específica.

```
    def read_sql_file(self, file_path):
        with open(file_path, 'r') as file:
            return file.read()
```

3. **Lectura de un archivo Excel `.xlsx` `readExcel()`:** Este método permite leer archivos de excel `.xlsx`, en una hoja de trabajo y ruta en específico, incluso controlar el tipo de dato que se requiere de lectura.

```
    def readExcel(self, path, sheet):
        df = pd.read_excel(path, sheet_name=sheet, dtype=str)
        return df

```

4. **Cambio de formato `cambio_formato()`:** Este método permite reemplazar un caracter por otro para ajustar el formato de algún tipo de código, el cual es usado en el Paquete Python para cambiar el formato de **CUENTA CGN** para fines regulatorios.

```
    def cambiar_formato(self, df, column):
        df[column] = df[colum].str.replace('.', '', regex=False)
        return df
```

También es usado para el nit, manteniendo la misma lógica

```
    def ajuste_nit(self, df, column):
        df[column] = df[column].str.replace('.', '',
        ↪  regex=False).str.replace('-', ':', regex=False)
        return df
```

5. **Creación y guardado de tablas HTML `generar_reportes_y_tablas()`:** Este método recorre el reporte final y va tomanto todas aquellas filas donde su valor en la columnas `CODIGO_CGN` es único, Crea una tabla html que contiene todas las filas que pertenezcan a un Código CGN específico. Esta tabla tiene CSS embebido por lo que será interpretable para el cuerpo de un correo electrónico. Finalmente las guarda en una carpeta, esta ruta será utilizada por un flujo de escritorio para efectos de envío a cada entidad, se debe actualizar el Excel que relaciona el Código CGN con el correo electrónico.

```
    html_template = """
        <html>
        <head>
            <title>Reporte para Entidad Oficial: {codigo_cgn}</title>
            <style>
                body {{
                    font-family: Arial, sans-serif;
                    margin: 20px;
                    background-color: #f9f9f9;
                }}
                h1 {{
                    color: #333;
                }}
                table {{
                    width: 100%;
                    border-collapse: collapse;
                    margin-top: 20px;
                    box-shadow: 0 2px 3px rgba(0,0,0,0.1);
                }}
                th, td {{
                    border: 1px solid #ddd;
                    padding: 8px;
                    text-align: left;
                }}
                th {{
                    background-color: #4CAF50;
                    color: white;
                }}
                tr:nth-child(even) {{
                    background-color: #f2f2f2;
                }}
                tr:hover {{
                    background-color: #e2e2e2;
                }}
            </style>
        </head>
        <body>
            <h1>Reporte para Entidad Oficial: {codigo_cgn}</h1>
            <table>
                <tr>
                    <th>CUENTA CGN</th>
                    <th>CODIGO CGN</th>
                    <th>VALOR CORRIENTE</th>
                    <th>VALOR NO CORRIENTE</th>
                </tr>
                {rows}
            </table>
        </body>
        </html>
        """
```

```
    def generar_reportes_y_tablas(df, ruta_salida, html_template):
        'Asegurarse de que la ruta de salida exista'
        if not os.path.exists(ruta_salida):
            os.makedirs(ruta_salida)

        'Agrupar el DataFrame por CODIGO_CGN y generar un HTML por cada grupo'
        for codigo_cgn, group in df.groupby('CODIGO CGN'):
            # Crear las filas de la tabla HTML con los datos de cada cuenta asociada a este CODIGO_CGN
            rows = ""
            for _, row in group.iterrows():
                rows += f"""
                <tr>
                    <td>{row['CUENTA CGN']}</td>
                    <td>{row['CODIGO CGN']}</td>
                    <td>{row['VALOR CORRIENTE']}</td>
                    <td>{row['VALOR NO CORRIENTE']}</td>
                </tr>
                """

            'Formatear el HTML con las filas generadas'
            html_content = html_template.format(codigo_cgn=codigo_cgn, rows=rows)

            'Guardar el archivo HTML'
            archivo_html = os.path.join(ruta_salida, f"{codigo_cgn}.html")
            with open(archivo_html, 'w', encoding='utf-8') as file:
                file.write(html_content)
```

En este módulo hay más excepciones específicas las cuales solo para este contexto son útiles, por ello se podrán visualizar en los anexos y su explicación de ellas.

6.1.6. Módulo Encriptación

Este módulo de encriptación importa la librería cryptography.fernet trayendose Fernet, un método de cifrado autenticado, lo que significa que los dartos cifrados también se verifican para garantizar que no han sido manipulados o alterados, este método genera una clave simétrica, la cual deve ser compartida entre las parte que cifran y descifran los datos. Esto se explica con mejor claridad en el módulo de seguridad. El módulo iene dos clases, la clase `Encriptador` y la clase `Desencriptador`

La clase `Encriptador` tiene un método inicial que recibe como argumento la ruta donde se encuentra la llave para el cargue de esta. Adicional a esto tiene los siguientes métodos:

- **Cargue de llave `cargar_clave()`:** Este método carga la llave buscando en la ruta donde se ubica, leyendo el archivo y devuelve la llave.

```
    def cargar_clave(self, key_path):
        """Cargar la clave de encriptacion desde un archivo."""
        with open(key_path, 'rb') as key_file:
            return key_file.read()
```

- **Encriptación `encriptar()`:** Este método verifica si el archivo que se desea encriptar existe, y lo carga, luego lo encripta por medio del método nativo de la librería Fernet `encrypt()` que recibe como parámetro el archivo y luego los guarda en una ruta específica.

```python
    def encriptar(self, data, output_json_path, is_file=False):
        """Encriptar los datos. Puede ser un archivo o un diccionario."""
        fernet = Fernet(self.key)

        if is_file:
            # Si es un archivo, leemos el contenido
            with open(data, 'rb') as file:
                data = file.read()
        else:
            # Si es un diccionario, lo convertimos a JSON y luego a bytes
            data = json.dumps(data).encode('utf-8')

        # Encriptamos los datos
        data_encrypted = fernet.encrypt(data)

        # Guardamos los datos encriptados en el archivo de salida
        with open(output_json_path, 'wb') as file:
            file.write(data_encrypted)
```

Para la clase `Desencriptador` que persigue la misma lógica que la clase `Encriptador` a diferencia de que usa la función `decrypt()` de Fernet.

```python
class Desencriptador:
    def __init__(self, key_path='F:/Reporte Entidades CGN/archivos/config/key.key'):
        self.key = self.cargar_clave(key_path)

    def cargar_clave(self, key_path):
        """Cargar la clave de encriptacion desde un archivo."""
        with open(key_path, 'rb') as key_file:
            return key_file.read()

    def desencriptar(self, encrypted_json_path):
        """Desencriptar el archivo JSON y cargar las credenciales."""
        with open(encrypted_json_path, 'rb') as file:
            data_encrypted = file.read()

        fernet = Fernet(self.key)
        data_decrypted = fernet.decrypt(data_encrypted)

        # Convertimos los bytes desencriptados a un diccionario
        return json.loads(data_decrypted)
```

Para accionar ese módulo, ya que el módulo principal `main.py` solo accede a este para realizar la desencriptación, se debe hacer por medio de un Script, que solo está a manos del encargado de infraestructura. Este Script contiene la referencia del módulo `Encriptacion.py` e importa la librería `Fernet` de `cryptography.fernet` donde tiene un único método llamado `generar_clave()`

```python
def generar_clave():
    key = Fernet.generate_key()
    with open('archivos/config/key.key', 'wb') as key_file:
        key_file.write(key)

    # Genera la clave
    generar_clave()

    encriptador = Encriptador()
    encriptador.encriptar('archivos/config/db_sac.json',
    ↪  'archivos/config/credenciales_sac_encriptadas.json', is_file=True)
    encriptador.encriptar('archivos/config/db_jd.json',
    ↪  'archivos/config/credenciales_jd_encriptadas.json', is_file=True)
```

Posterior a la generación de la clave, debe guardar los archivos de configuración en la ruta preestablecida para que el el Paquete Python pueda acceder a la desencriptación de estos. El encargado de hacer esto debe llevar el archivo de credenciales y ponerlos en la ruta específica, carga el Script encriptador y accionarlo para generar la clave, eliminar los archivos de credenciales sin encriptar y dejar únicamente los encriptados, eliminar el módulo encriptador y salir del servidor donde se encuentra hospedada la solución. De esta manera solo este sabrá las verdaderas credenciales, a las que ni siquiera el desarrollador o el encargado de soporte pueda tener acceso a ellas, por buenas prácticas es el deber ser.

Teniendo la creación del Paquete Python exitosamente desarrollada, pasamos a la lógica que ejecutará el proyecto, se da lugar al desarrollo de los flujos de trabajos comandados por Microsoft Power Automate:

6.2. Automatización Robótica de Procesos

Se desarrolla un flujo de trabajo con la lógica para ejecutar el Paquete Pytho, sin embargo, £cómo este flujo se va a disparar? Se propuso inicialmente que sea por medio de un correo electrónico vía Outlook que es de por si un conector de Microsoft Power Automate, el proceso de construcción de la lógica se divide en dos módulos de Automate, flujos de nube para el desencadenamiento de estos, ya que los flujos de escritorio no tienen trigger, y los correos se hace por medio de la nube de Microsoft, y por otro lado el flujo de escritorio que es el encargado de la ejecución directa en la línea de comandos del Paquete Python.

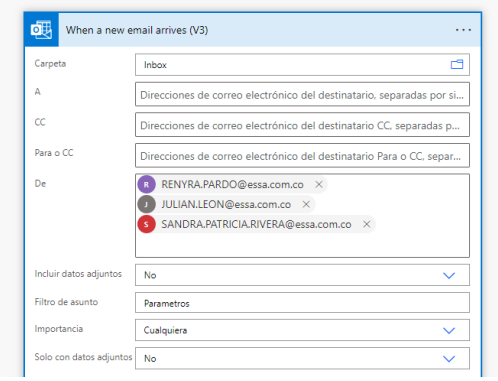
6.2.1. Flujos de nube

Se crea un flujo de nube donde el desencadenador es cuando llega un correo electrónico de Outlook, Este mensaje debe cumplir con los filtros que permite establecer este trigger, como autor del correo, asunto, carpeta. (Ver Figura 16)

Posterior a la acción de desencadenamiento del flujo, se deben obtener los parámetros enviados en el cuerpo del correo. Outlook por defecto maneja el cuerpo de los correos con lógica HTML, para limpiar esta lógica y obtener los parámetros en bruto de pasa por una acción Compose, la cual permite intervenir la información con fórmulas, y por medio de un Compose adicional se eliminan aquellas sentencias HTML en las cuales se encapsula cualquier mensaje en el cuerpo del correo. (Ver figura 17)

Figura 16.

Trigger flujo de nube ***GetParameters***.



Nota: Filtros del Trigger del flujo de nube encargado de capturar parámetros y ejecutar el Paquete Python. León, J. (2024). Diseño propio.

El compose **Creal HMTL** obtiene los parámetros limpios, sin embargo estos parámetros vienen en una matriz de 2x2 cuya ejemplo de visualización es:

year:	2024
trimester:	1

Tabla 2
Ejemplo de los parámetros enviados en el cuerpo del correo.

Por ello se utiliza un acción **Compose** para cada parámetros, el cual usa la siguiente fórmula:

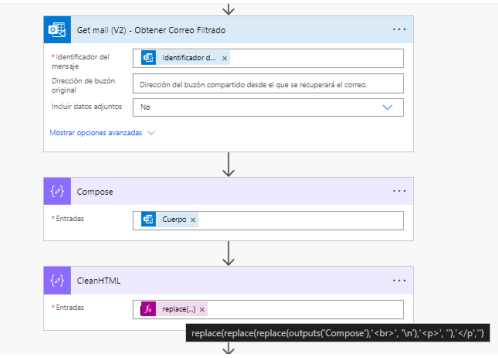
Para obtener el año:

```
int(trim(split(split(outputs('CleanHTML'), 'year: ')[1], '\n')[0]))
```

Para obtener el trimestre:

Figura 17.

Obtención de parámetros del correo. ***GetParameters***.



Nota: Extracción de parámetros del cuerpo del correo y limpieza del HTML por defecto que trae. León, J. (2024). Diseño propio.

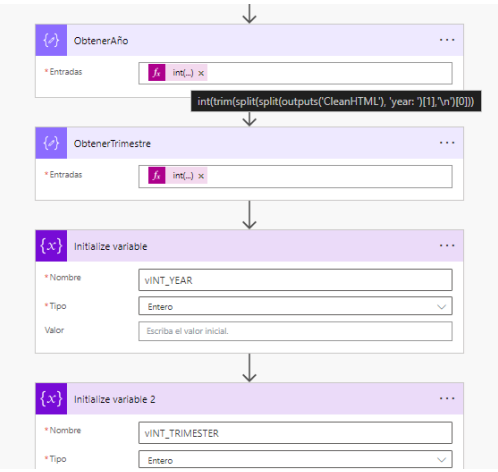
```
int(trim(split(split(outputs('CleanHTML'), 'trimester: ')[1], '\n')[0]))
```

Esta fórmula obtiene la salida de la limpieza del HTML, quita espacios vacíos y saltos tabulares, y obtiene el primer valor de la fila, para cada caso. Posterior a ello inicializa las variables para poder pasarles el valor obtenido en los Compose. (Ver Figura 18)

Teniendo las variables inicializadas, es posibles pasarles el valor obteniedo, teniendo las variables con el valor correspondiente, se pasan a las variables de entrada que contiene el flujo de escritorio que contiene la lógica de ejecución de Paquete Python, por medio de la función **Run a flow built with Power Automate for desktop**, esta acción permite conectarse a la máquina nativa donde se encuentra este flujo, e incluso decidir sobre su ejecución y parámetros de entrada. (Ver Figura 19)

Figura 18.

Inicialización de variables. ***GetParameters****.*



Nota: Obtención de los parámetros por separado e inicialización de variables para capturar los parámetros. León, J. (2024). Diseño propio.

Ahora bien, hay un flujo adicional que se desarrollo para desencadenar el flujo de escritorio que realiza el envío de las tablas HTML a cada entidad. Este flujo mantiene la misma lógica del anterior pero evitando la transferencia de parámetros, solo acudiendo a los filtros de Trigger y la invocación del flujo de escritorio (Ver Figura 20)

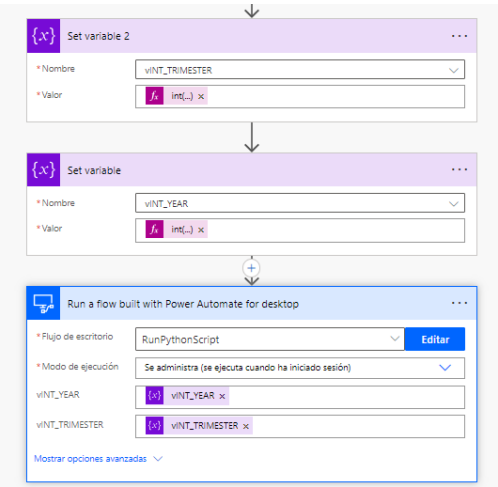
6.2.2. Flujos de escritorio

Inicialmente se le establecieron 2 variables de entrada al flujo de escritorio `RunPythonFile`:

- `vINT_trimester`: variable que recibe el parámetro trimestre por parte del flujo de nube que lo invoca.
- `vINT_year`: variable que recibe el parámetro año por parte del flujo de nube que lo invoca.

Figura 19.

Invocación y paso de parámetros. ***GetParameters****.*



Nota: Invocación del flujo de escritorio que lleva la lógica de ejecución, y transferencia de parámetros previamente obtenidos. León, J. (2024). Diseño propio.

El flujo de escritorio encargado de la ejecución del Paquete Python tiene las siguientes acciones: (Ver Figura 21)

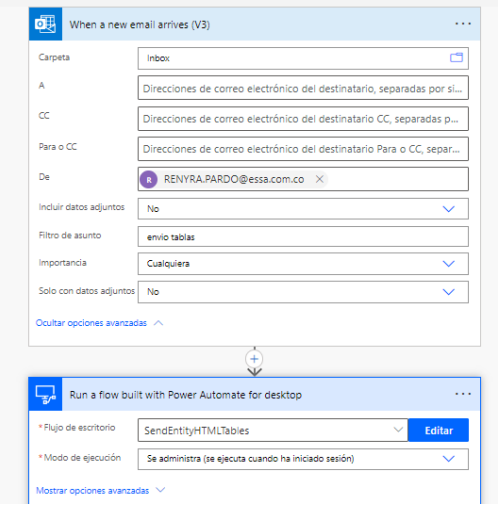
1. **Establecer Variable** `vPath_Python`**:** Esta variable se establece para definir la ruta donde se encuentra el ejecutable de python, con el fin de poder acceder a la ejecución en la línea de comandos.

2. **Establecer Variable** `vPath_Script`**:** Esta variable se establece para definir la ruta donde se encuentra el módulo principal del Paquete Python, para ejecutarlo por medio de la línea de comando.

3. **Ejecutar script de PowerShell** `PowerShellOutput` **y** `ScriptError`**:** Son las variables de la salida de la ejecución del módulo principal del Paquete Python, la configuración de esta

Figura 20.

Trigger y Ejecución. ***GetSignalEntities****.*



Nota: Trigger, filtros de correo e invocación de flujo de escritorio para envío de tablas HTML a cada entidad. León, J. (2024). Diseño propio.

acción se realiza ubicando la ruta del ejecutable de pytho, la ruta del módulo principal y los parámetros usando las variables de entrada anteriormente inicializadas en 0, todo en la línea de comandos. (Ver Figura 22)

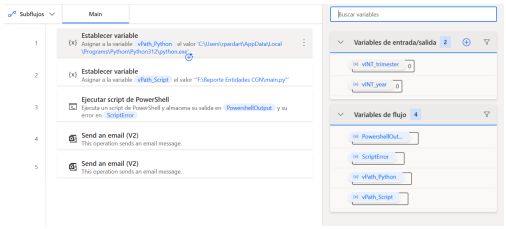
Las variables de salida tendrán bien sean impresiones directas del Paquete Python, o errores que este presente, respectivamente.

4. **Enviar un correo** Esta acción es para firmes de notificación, pues envía una alerta al usuario y una al usuario de soporte, el primero indica que la ejecución fue exitosa y ofrece la ruta donde está hospedado el reporte creado. Por otro lado el otro envía los errores o peligros que presente el Paquete Python. (Ver Figuras 23 , 24)

El flujo de escritorio **SendEntities** no posee variables de entrada, ya que solo requiere de su invo-

Figura 21.

Estructura del flujo de escritorio ***RunPythonFile***



Nota: Estructura diseñada para el flujo de escritorio encargado de la ejecución del Paquete Python. León, J. (2024). Diseño propio.

cación por parte del flujo de nube para desecadenarse. Este flujo contiene las siguientes acciones para llevar a cabo su ejecución y funcionamiento: (Ver Figura 25)

1. **Iniciar Excel `ExcelInstance`:** Esta acción lee un archivo excel en una ruta específica y entrega una instancia para su lectura. Se requiere obtener el archivo Excel que relaciona el Código de Entidad CGN y su correo.

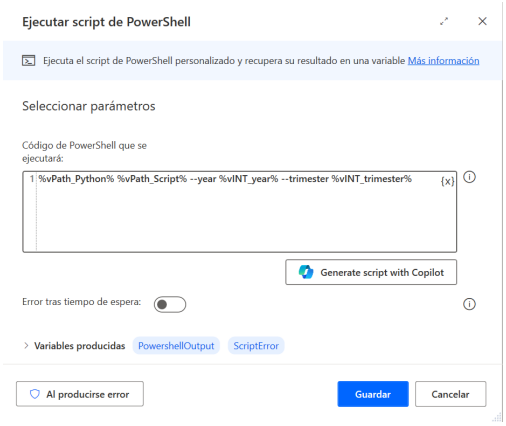
2. **Obtener primer fila libre de una columna `FirstFreeRowOnColumn`:** Esta acción busca la primera fila libre en un Excel, lo hace por medio de la instancia, eso con el fin de saber el rango que debe recorres posteriormente.

3. **Leer hoja de cálculo de Excel `ExcelData`:** Esta acción lee la instancia y toma la información en el rango A:1 (primera celda) hasta B:`FirstFreeRowOnColumn` (última celda)

4. **Obtener archivos de la carpeta `Files`:** Esta acción obtiene todos los archivos de una carpeta en específico y lo guarda en la variable `Files`.

Figura 22.

Configuración acción de ejecución. ***RunPythonFile***



Nota: Configuración de la acción de ejecución del Script, ubicando la línea de comandos los parámetros. León, J. (2024). Diseño propio.

Para recorrer estos archivos se utiliza un `For each`, donde la iteración se hace sobre cada archivo, y sigue con la acción:

5. **Obtener parte de ruta de archivo `CGNcode`:** Esta acción se realiza para obtener el nombre del archivo, que como se mencionó anteriormente, es el Código de Entidad CGN.

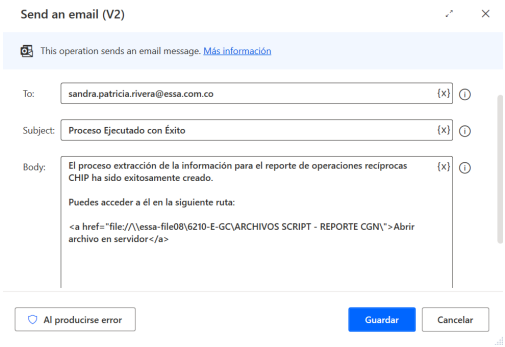
6. **Establecer Variable `vBool_Found`:** Esta variable de tipo booleano se establece falsa hasta encontrar la coincidencia y enviar el correo respectivo, y no gaste recursos recorriendo posterior al encuentro.

   Para lograr esto se debe recorrer la variable `ExcelData` que contiene la tabla puente, y una condición que consulte si el código CGN que se está evaluando coincide con el que está en la tabla, con ello pasa a la siguiente acción:

7. **Leer texto del archivo `FileContent`:** Esta acción va a acceder al contenido del archivo,

Figura 23.

Configuración acción de enviar notificación de exito. ***RunPythonFile***



Nota: Configuración de la acción de envío de notificación de exitosa ejecución al usuario. León, J. (2024). Diseño propio.

en este caso obtendrá el HTML de la tabla y lo guardará en la variable `FileContent`.

8. **Enviar un correo:** Esta acción emite un correo, teniendo el Código CGN, accediendo al correo posterior a la relación en la tabla, y el contenido del archivo HTML como cuerpo del correo. También utiliza un asunto parametrizado **Detalle Reporte CGN**

9. **Establecer Variable `vBool_Found`:** Se establece la variable como verdadera con el fin de evaluarla y salir del bluce teniendo el envío exitosamente realizado.

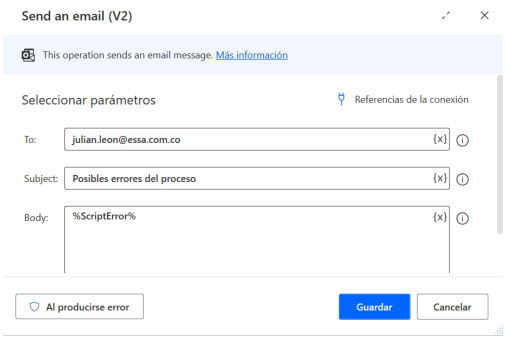
10. **Salir del Bucle:** Esta acción dentro de una condición que evalúa si la variable `vBool_Found` es verdadera, termina el bucle.

6.3. Interfaz Gráfica de Usuario

La interfaz fue creada en Microsoft Power Apps la cual consta de una sola pantalla, teniendo en cuenta que la interfaz es para efectos de generación de la lógica fuerte del proyecto, se decidió

Figura 24.

Configuración acción de ejecución. ***RunPythonFile***



Nota:Configuración de la acción de envío de notificación de posibles errores en la ejecución al usuario de soporte. León, J. (2024). Diseño propio.

mantener pantallas emergente donde la lógica de visibilidad es basada en botones.

Fue desarrollada con parametrización a la hora de elegir los parámetros de ejecución, usando drop-box en lugar de cajas de texto propensas a errores de digitación e incluso una doble confirmación de ejecución para evitar mala diligencia de parámetros.

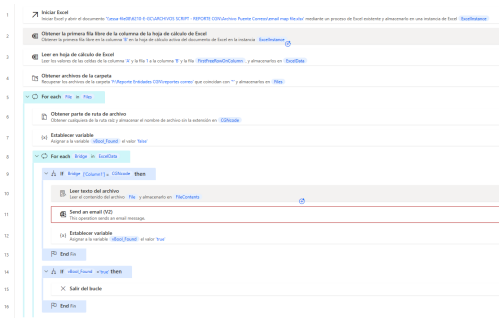
Inicialmente se tiene una primera pantalla con dos botones, uno pretende guiar al usuario a la digitación de los parámetros, y el otro al envío de las tablas HTML vía correo a cada entidad. (Ver Figura 27)

La acción del botón **Digitar Parámetros**, es hacer visible eel grupo de componentes que desarrollan la siguiente pantalla de digitalización, cambiando el valor booleano de las variables de visualización de cada componente. (Ver Figuras 28, 29)

En la pantalla de diligenciamiento de parámetros se tienen dos componentes **DropBox** o **Caja**

Figura 25.

Estructura flujo. ***SendEntities***



Nota: Estructura completa del flujo de envío a entidades. León, J. (2024). Diseño propio.

Desplegable que permite al usuario elegir el año y el trimestre para el que requiere la generación del reporte.

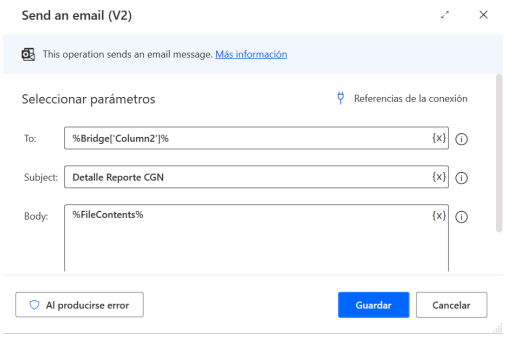
Posterior a esta acción, el usuario confirma la efecución por medio del botón **Confirmar ejecución**, que manteniendo la lógica del anterior botón, hace visible una pantalla emergente que resumen los parámetros escogidos y re confirma la generación del reporte. Ver figuras (30, 31)

Este botón de **Generar Reporte** tiene en su propiedad **OnSelect** la ejecucuón de un flujo de nube que tiene como parámetros los valores de las cajas desplegables, y las envía por correo a la cuenta técnica que contiene la solución. (Ver Figuras 32,33)

Adicional a esto la acción y el flujo de envío de las tablas HTML, son generador por el botón **Envío a entidades**, cuya acción es desencadenar un flujo que envía una señal vía correo a la cuenta técnica que hospeda el flujo con tal lógica. (Ver Figuras 34,35)

Figura 26.

Configuración Acción envío de correo. ***SendEntities***



Nota: Configuración de la acción encargada de enviar las tablas HTML vía correo a las entidades. León, J. (2024). Diseño propio.

6.4. Implementación de seguridad

La implementación de seguridad se realizó inicialmente en la encriptación de credenciales para realizar la conexión a las bases de datos, por medio de un módulo del paquete de python, presentado en sesiones anteriores, donde su realización es de la siguiente manera:

1. Las credenciales deben llevar un formato tipo diccionario el cual tiene items como : usuario, contraseña, sid o service name, también un valor tipo booleano que indique si la conexión se debe hacer en modo grueso o delgado.

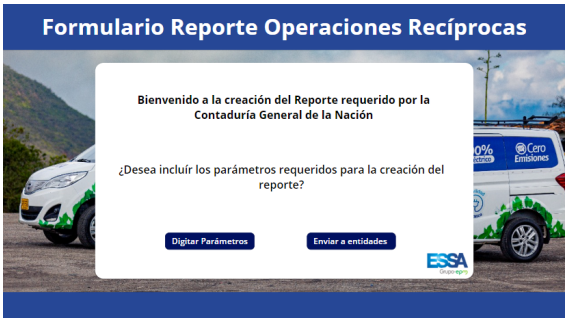
2. Hospedar las credenciales en un archivo `.json` en la ruta específica donde el encriptador va a buscar

3. Accionar el encriptador.

   El encriptador es un Script que estará a manos del encargado de arquitectura el cual tiene la

Figura 27.

Pantalla inicial GUI. ***Form CGN****.*



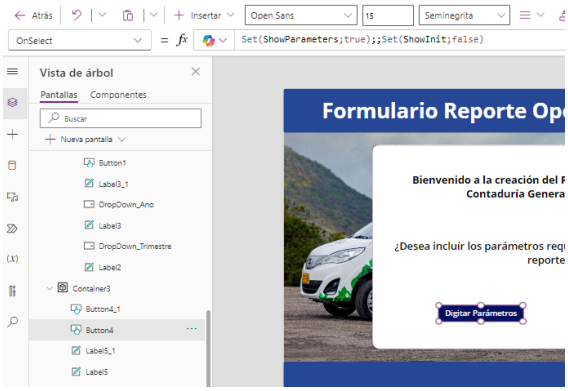
Nota: Visualización de la composición de la pantalla inicial de la interfaz gráfica. León, J. (2024). Diseño propio.

siguiente estructura:

```
from modulos.encriptation import Encriptador
import json

from cryptography.fernet import Fernet

def generar_clave():
    key = Fernet.generate_key()
    with open('archivos/config/key.key', 'wb') as key_file:
        key_file.write(key)

# Genera la clave
generar_clave()

encriptador = Encriptador()
encriptador.encriptar('archivos/config/db_sac.json', 'archivos/config/db_sac_encrypted.json', is_file=True)
encriptador.encriptar('archivos/config/db_jd.json', 'archivos/config/db_jd_encrypted.json', is_file=True)

```

Figura 28.

*Acción Botón "Digitar Parámetros". **Form CGN**.*



Nota: Acción que realiza el botón "Digitar Parámetros". León, J. (2024). Diseño propio.

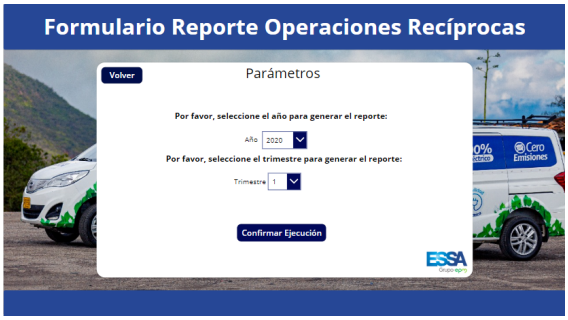
> Ejecutando el Script, se va a crear una clave o llave que usa para hacer la encriptación, y guarda en la ruta los archivos encriptados. El encargado debe eliminar las credenciales, retirar el Script, y el Paquete Python tendrá uso de las credenciales sin problema.

También se realizar implementación de seguridad en la conexión a las bases de datos, ya que la librería `Python-oracledb` la cual hace enlace mediante el protocolo SSL/TLS para el cifrado de las conexiones, por otro lado la organización maneja el uso de firewalls y VPN como seguridad en las conexiones.

Por otra parte la implementación de la seguridad en los servicios explicada en sesiones anteriores, se hace uso de esta implementación ortorgando permisos a los usuarios debido y manteniendo una gestión del flujo completo de la solución.

Figura 29.

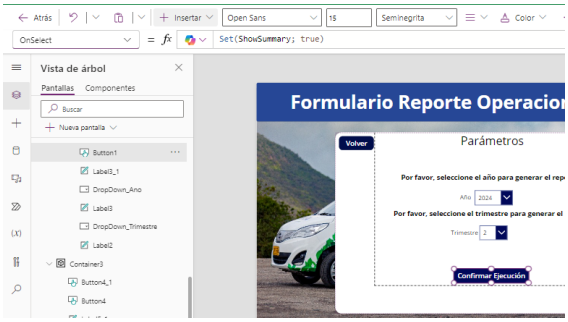
Pantalla de diligenciamiento de parámetros ***Form CGN****.*



Nota: Pantalla dedicada al diligenciamiento de parámetros para la generación del reporte. León, J. (2024). Diseño propio.

Figura 30.

*Acción del botón de confirmar ejecución.****Form CGN****.*



Nota: Acción que realiza el botón Çonfirmar Ejecución". León, J. (2024). Diseño propio.

6.5. Despliegue y Estabilización

El despliegue de esta solución se realizó transfiriendo el Paquete Python al servidor donde se va a hospedar. Incluyendo requerimientos como la instalación de las librerías:

- `pandas`
- `Python-Oracledb`

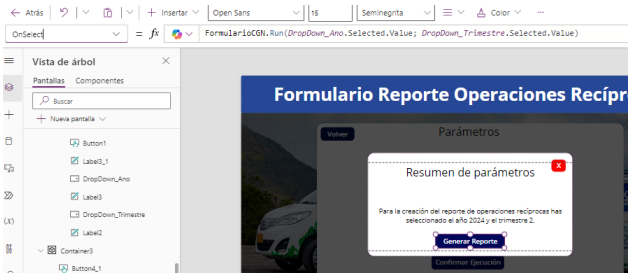
Figura 31.

Pantalla emergente de re confirmación de generación. ***Form CGN****.*

Nota: Pantalla emergente que resume los parámetros escogidos y re confirma la ejecución. León, J. (2024). Diseño propio.

Figura 32.

*Acción del botón Generar Reporte.****Form CGN****.*



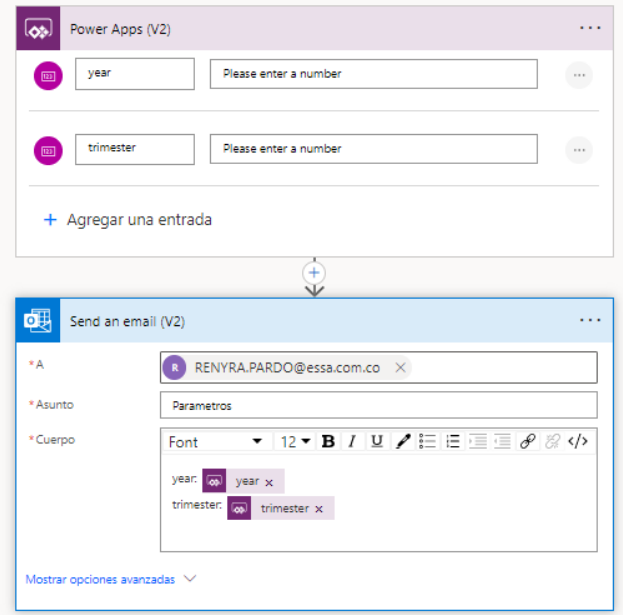
Nota: Acción que realiza el botón "Generar Reporte". León, J. (2024). Diseño propio.

- `json`
- `cryptography.Fernet`
- `Oracle Instant Client`

Ya que esta solución requiere de un lugar donde hospedar archivos, bien sea los resultantes, o los necesario para la ejecución, como lo son las conciliaciones y excepciones. Estos archivos se hospedan en una carpeta local compartida, que tiene acceso restringido, por ello tuvo que tramitarse

Figura 33.

*Flujo desencadenado por el botón "Generar Reporte". **Form CGN**.*



Nota: Flujo cuyo trigger es el botón de la GUI de Power Apps, recibe parámetros y los envía por correo. León, J. (2024). Diseño propio.

los permisos únicamente del usuario del servidor donde está hospedada la solución, y el usuario que debe cargar los archivos u obtener los resultantes.

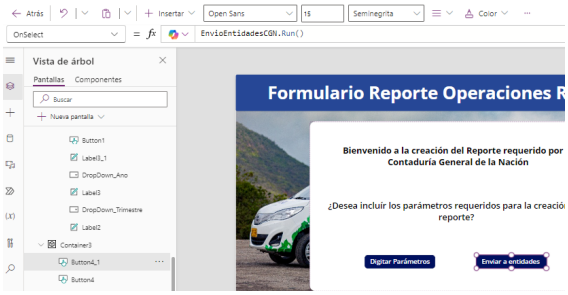
Se realiza despliegue a las conexión de las bases de datos, cambiando el ambiente al que se hace conexión, cambiando el host al que se apunta, el puerto y el sid o service name.

Para el despliegue de la parte de la solución desarrollada por herramientas de Microsoft, se debe crear una solución, integrando todos los elementos (Ver Figura 36):

- aplicación

Figura 34.

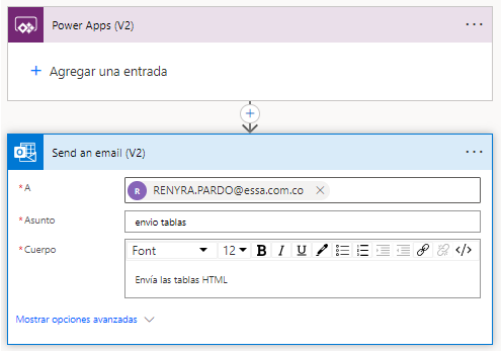
*Acción del botón Envío a Entidades.**Form CGN**.*



Nota: Acción que realiza el botón Envío a entidades. León, J. (2024). Diseño propio.

Figura 35.

*Flujo desencadenado por el botón Envío a entidades. **Form CGN**.*

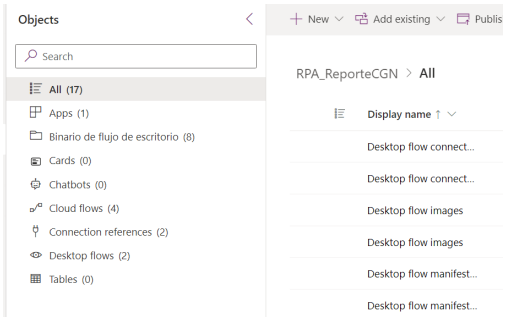


Nota: Flujo cuyo trigger es el botón de la GUI de Power Apps y envía una señal de ejecución por correo. León, J. (2024). Diseño propio.

- flujos de nube
- flujos de escritorio
- referencias de conexión
- conectores

Figura 36.

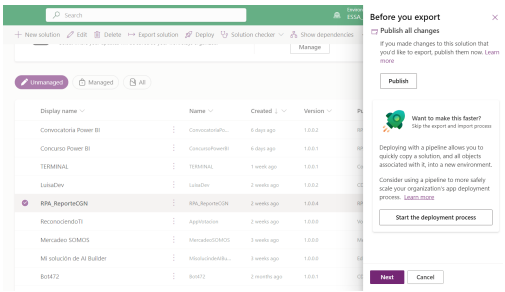
Componentes de la solución. ***RPA_ReporteCGN****.*



Nota: Componentes integrados en la solución RPA_ReporteCGN. León, J. (2024). Diseño propio.

Figura 37.

Exportación de la solución. ***RPA_ReporteCGN****.*



Nota: Exportación de la solución RPA_ReporteCGN en modo managed. León, J. (2024). Diseño propio.

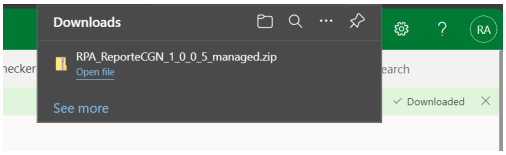
Teniendo la solución creada en el entorno de desarrollo, se procede a exportarla en modo **Managed** por buenas prácticas. (Ver Figura 37)

Esta solución exporta todos los componentes en una carpeta comprimida (`.zip`). (Ver Figura 37)

Posterior a la descarga se realiza el cambio de entorno a producción y se importa la solución (Ver Figura 39)

Figura 38.

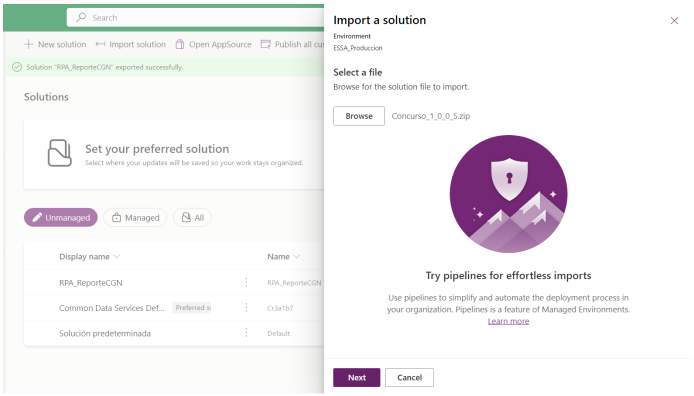
Paquete de la solución descargado. ***RPA_ReporteCGN****.*



Nota: Descarga de la carpeta compartida de los componentes de la solución RPA_ReporteCGN. León, J. (2024). Diseño propio.

Figura 39.

Importación de la solución. ***RPA_ReporteCGN****.*



Nota: Importación de la solución RPA_ReporteCGN en el entorno de producción. León, J. (2024). Diseño propio.

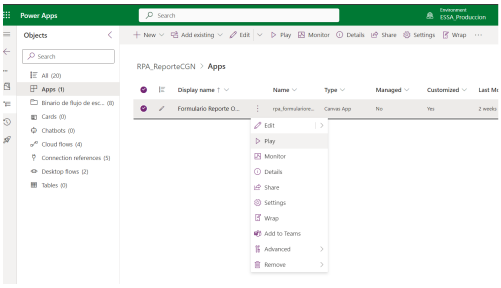
Finalmente se ejecuta la solución en producción (Ver Figura 40

6.6. Pruebas

Se realizaron pruebas desde el punto más bajo de la solución (el paquete Python), hasta la solución en conjunto, desde la interfaz gráfica para el reporte del primer y segundo trimestre del presente año.

Figura 40.

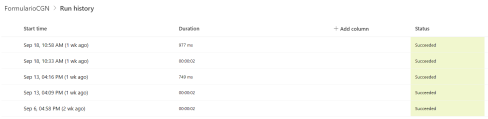
Solución en producción. ***RPA_ReporteCGN****.*



Nota: Solución RPA_ReporteCGN ejecutada en producción. León, J. (2024). Diseño propio.

Figura 41.

Pruebas flujo de nube que desencadena la solución



Nota: Prueba de las ejecuciones por parte del flujo desencadenado por la interfaz gráfica y desencadenador del flujo que invoca el flujo de escritorio. León, J. (2024). Diseño propio.

1. Se ejecutan los parámetros desde la aplicación y se probaron las ejecuciones del flujo desecadenador que enviará los parámetros vía correo (Ver Figura 41)

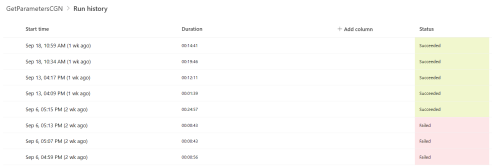
2. Con la ejecución del flujo, se estudia la ejecución del flujo que recibe los parámetros y ejecuta el flujo de escritorio a cargo del Paquete Python. (Ver Figura 42)

3. Ahora bien, teniendo pruebas se revisan las ejecuciones del flujo de escritorio que ejecuta el Paquete Python (Ver Figura 43)

4. Se hacen verificaciones de la creación del reporte y se verifican sus valores los cuales coinciden, ya que se hace comparativa con los reportes ya realizados (Ver Figuras 44,45)

Figura 42.

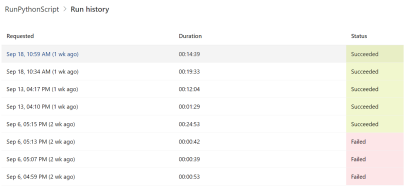
Pruebas flujo de nube hospedado en cuenta técnica



Nota: Prueba de ejecución del flujo que recibe los parámetros e invoca el flujo de escritorio. León, J. (2024). Diseño propio.

Figura 43.

Pruebas flujo de escritorio



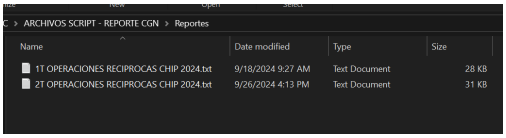
Nota: Prueba de ejecución del flujo de escritorio que ejecuta el Paquete Python. León, J. (2024). Diseño propio.

5. Para el canal de envío del detalle (tablas HTML) a cada entidad se realiza de la misma manera, y fue exitoso el envío (Ver Figura 47)

Finalmente con cada ejecución se puede hacer una proyección de las ejecuciones y una comparativa referente al gatos previo a la implementación de la automatización. Los resultados de ejecución tienen un promedio de 20 minutos, esta ejecución es necesaria únicamente 4 veces al año, ya que su reporte se maneja con frecuencia trimestral. (Ver Figura 46)

Figura 44.

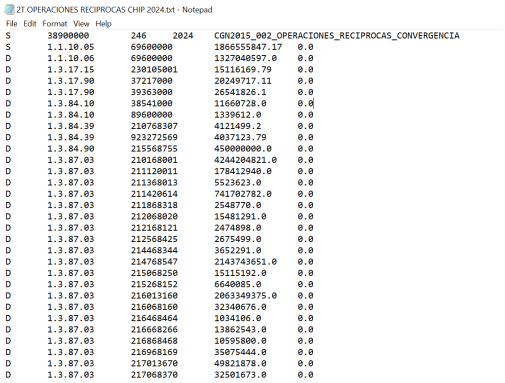
Verificación reportes ejecutados



Nota: Verificación de la creación del reporte en la ruta específica. León, J. (2024). Diseño propio.

Figura 45.

Verificación reporte resultante



Nota: Verificación del formato del reporte y los valores. León, J. (2024). Diseño propio.

7. Conclusiones

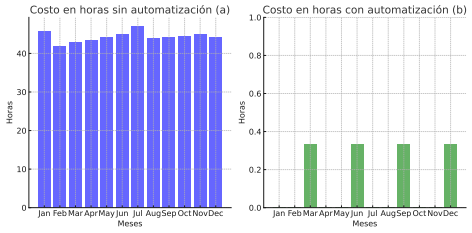
La integración de Python en entornos de Microsoft ha demostrado ser una solución eficiente y adaptable para la automatización de procesos complejos. Su capacidad para manejar grandes volúmenes de datos, junto con su amplia gama de bibliotecas especializadas como Pandas, ha facilitado el tratamiento y análisis de datos que, de otro modo, serían costosos y difíciles de gestionar utilizando únicamente herramientas nativas de Microsoft. La facilidad de integración con bases de

datos permite realizar operaciones de manera rápida y eficaz, optimizando significativamente el tiempo de procesamiento de datos.

Como se muestra en la Figura 46 (a), el tratamiento manual de datos, que tradicionalmente llevaba alrededor de 540 horas al año, ha sido reducido en un 99.7 % mediante la automatización, disminuyendo el tiempo operativo a solo 80 minutos al año (Figura 46 (b)). Este ahorro de tiempo implica una liberación considerable de recursos humanos y tecnológicos, que ahora pueden ser reorientados hacia actividades de mayor valor añadido para la empresa, impulsando la productividad y la eficiencia operativa. Además, este enfoque refuerza la escalabilidad y flexibilidad de las soluciones basadas en Python en entornos corporativos, abriendo nuevas oportunidades para la mejora continua en la gestión de datos y la automatización de tareas.

Figura 46.

Costo en horas antes y después de la automatización



Nota: (a) Representa el costo en horas sin automatización en un año, con valores que oscilan entre 41 y 48 horas mensuales. (b) Muestra el costo en horas con automatización aplicado solo en el corte de cada trimestre, con una ligera variación de tiempos que promedian alrededor de 20 minutos, representados en horas. León, J. (2024). Diseño propio.

Figura 47.

Correo con detalle enviado.

Nota: Correo de llegada enviado por la cuenta técnica con la tabla HTML, exito de ejecución. León, J. (2024). Diseño propio.

7.1. Trabajo Futuro

A futuro, la automatización de procesos contables en la Electrificadora de Santander S.A. (ESSA) podría beneficiarse de la integración de Inteligencia Artificial (IA) y Machine Learning (ML), lo que permitiría que los bots adaptaran su comportamiento a cambios inesperados en los datos o procesos, mejorando la toma de decisiones y evolucionando hacia la Automatización de Procesos Inteligentes (IPA). Asimismo, la incorporación de tecnologías de Procesamiento de Lenguaje Natural (NLP) facilitaría la interpretación de información no estructurada, reduciendo la intervención humana.

Otra área de mejora es la seguridad de los datos. Aunque ya se emplean herramientas como Azure Key Vault, el uso de criptografía homomórfica permitiría procesar datos cifrados sin necesidad de desencriptarlos, aumentando así la protección de la información sensible. Además, la metodología desarrollada puede extenderse a otros departamentos de la empresa, como recursos humanos o finanzas, optimizando la eficiencia en varias áreas.

Finalmente, explorar otras plataformas de bajo código como Appsmith o Retool ofrecería mayor flexibilidad en los flujos de trabajo, y un análisis del impacto en la sostenibilidad podría alinear estos esfuerzos de automatización con los objetivos medioambientales de la empresa.

Bibliografía

Aalst, W. M. P. v. d., La Rosa, M., Mendling, J., & Reijers, H. A. (2018). *Fundamentals of Business Process Management*. Springer. https://doi.org/10.1007/978-3-662-56509-4

Alsaadi, H., Radain, D., Alzahrani, M., Alshammari, W., Alahmadi, D., & Fakieh, B. (2021). Factors that affect the utilization of low-code development platforms: Survey study. *Revista Român de Informatic i Automatic*, *31*, 123-140. https://doi.org/10.33436/v31i3y202110

Beazley, D. (2009). *Python Essential Reference*. Addison-Wesley Professional.

Beazley, D. (2013). *Python Cookbook: Recipes for Mastering Python 3*. O'Reilly Media.

Cooper, L. A., Holderness, D. K., Sorensen, T. L., & Wood, D. A. (2019). Robotic Process Automation in Public Accounting. *Accounting Horizons*, *33*(4), 15-35. https://doi.org/10.2308/acch-52466

Cryptography. (2024). *Fernet - symmetric encryption*. https://cryptography.io/en/latest/fernet/

Di Ruscio, D., Kolovos, D., Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? *Software and Systems Modeling*, *21*. https://doi.org/10.1007/s10270-021-00970-2

Doguc, O. (2020). Robot Process Automation (RPA) and Its Future. En *Advances in e-business research series* (pp. 469-492). https://doi.org/10.4018/978-1-7998-1125-1.ch021

Foundation, P. S. (2024). *Cryptography documentation*. https://cryptography.io/en/latest/

Gami, A., & Singh, R. (2019). Intelligent Process Automation (IPA) and its Impact on Business Efficiency. *Journal of Business Automation*, *5*(3), 45-60. https://doi.org/10.1017/jba.2019.12

Gami, M., Jetly, P., Mehta, N., & Patil, S. (2019). Robotic Process AutomationFuture of Business Organizations: A Review. https://doi.org/10.2139/ssrn.3370211

Jafari, R. (2024). *Hands-On Data Preprocessing in Python*. O'Reilly Media.

Kazil, J., & Jarmul, K. (2016). *Data Wrangling with Python: Tips and Tools to Make Your Life Easier*. O'Reilly Media.

Lutz, M. (2013). *Learning Python*. O'Reilly Media.

Madakam, S., Holmukhe, R., & Jaiswal, D. (2019). The future digital workforce: Robotic Process Automation (RPA). *Journal Of Information Systems And Technology Management*, *16*, 1-17. https://doi.org/10.4301/s1807-1775201916001

Martelli, A. (2006). *Python in a Nutshell*. O'Reilly Media.

Microsoft. (2023). *Security and governance in Power*. https://learn.microsoft.com/en-us/power-platform/admin/security

Microsoft. (2024a). *Azure Key Vault - secure key management*. https://learn.microsoft.com/en-us/azure/key-vault/general/basic-concepts

Microsoft. (2024b). *Best practices for securely storing API keys and credentials in python*. https://docs.microsoft.com/en-us/azure/key-vault/general/best-practices

Microsoft. (2024c). *Encryption in Power Platform*. https://learn.microsoft.com/en-us/power-platform/admin/encryption

Microsoft. (2024d). *Entornos: Descripción general*. https://learn.microsoft.com/es-es/power-platform/admin/environments-overview

Microsoft. (2024e). *Security roles in Power Platform - Microsoft Learn*. https://learn.microsoft.com/en-us/power-platform/admin/security

Microsoft. (2024f). *Virtual Private Network (VPN) Overview*. https://learn.microsoft.com/en-us/security/zero-trust/network-security/vpn-overview

Microsoft. (2024g). *What is RPA (Robotic Process Automation)?* [Accessed: 2024-09-26]. https://www.microsoft.com/en-us/solutions/robotic-process-automation

Najjar, A. (2023). *Microsoft Power Automate Cookbook*. O'Reilly Media, Inc.

Narayn, H. (2023a). *Building the Modern Workplace with SharePoint Online: Solutions with SPFx, JSON Formatting, Power Automate, Power Apps, Teams, and PVA*. APRESS.

Narayn, H. (2023b). *Building the Modern Workplace with SharePoint Online: Solutions with SPFx, JSON Formatting, Power Automate, Power Apps, Teams, and PVA*. Apress.

Open Web Application Security Project (OWASP). (2024). Cryptographic Storage Cheat Sheet. https://cheatsheetseries.owasp.org/cheatsheets/Cryptographic_Storage_Cheat_Sheet.html

Oracle. (2024a). *Introduction to the Python driver for Oracle Database* [https://pythonoracledb.readthedocs.io/en/latest/user_guide/introduction.html]. Oracle.

Oracle. (2024b). *Introduction to the Python driver for Oracle Database*. https://pythonoracledb.readthedocs.io/en/latest/user_guide/introduction.html

pandas development team, T. (2024). *pandas documentation* [https://pandas.pydata.org/docs/index.html]. pandas 2.2.2 documentation.

Pandit, A. (2024). *Exploring Low-Code and No-Code Development with Power Apps*.

Pilgrim, M. (2004). *Dive Into Python*. Apress.

Prinz, N., Rentrop, C., & Huber, M. (2021). Low-Code Development Platforms: A Literature Review. *AMCIS*.

Provost, F., & Fawcett, T. (2013). *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. O'Reilly Media.

Rajaram, A., Olory, C., Leduc, V., Evaristo, G., Coté, K., Isenberg, J., & Fiset, P. (2022). An integrated virtual pathology education platform developed using Microsoft Power Apps and Microsoft Teams. *Journal of Pathology Informatics*, *13*, 100117. https://doi.org/10.1016/j.jpi.2022.100117

Rattenbury, T., et al. (2017). *Principles of Data Wrangling: Practical Techniques for Data Preparation*. O'Reilly Media.

Real Python. (2024). How to securely store and access secrets in Python applications. https://realpython.com/python-environment-variables/

Sharma, R., & Arora, R. (2023). *Low-code development with appsmith: Building internal tools and business applications* (1st). Apress.

Simon, P. (2022). *Low-Code/No-Code: Citizen Developers and the Surprising Future of Business Applications*. Racket Publishing.

Stack Overflow. (2024). Running Python Scripts in Microsoft Power Automate Cloud [Accessed: 2024-09-27]. https://stackoverflow.com/questions/77751515/running-python-scripts-in-microsoft-power-automate-cloud

Syed, S., Ahmed, A., & Iqbal, M. (2020). Towards a Method for Automated Testing in Robotic Process Automation Projects. *14th International Workshop on Automation of Software Test (AST)*, 42-47. https://doi.org/10.1109/AST.2019.00012

tapanm-MSFT. (2024). *Documentación oficial de Microsoft Power Apps - Power Apps*. https://learn.microsoft.com/es-es/power-apps/

Team, P. (2024). *Pywinauto Documentation* [Accessed: 2024-09-26]. https://pywinauto.readthedocs.io/en/latest/

van Rossum, G. (2001). *Python Reference Manual* [Release 2.1]. Python Software Foundation.

Van Der Aalst, W., Bichler, M., & Heinzl, A. (2018). Robotic Process Automation. *Business & Information Systems Engineering*, *60*(4), 269-272. https://doi.org/10.1007/s12599-018-0542-4

Vartiainen, H. (2024a). Lean Management Empowerment: Elevating Employee Experience through RPA Implementation with Microsoft Power Automate.

Vartiainen, H. (2024b). *Lean Management Empowerment: Elevating Employee Experience through RPA Implementation with Microsoft Power Automate*.

Xerox. (2023). *Robotic Process Automation with Power Automate | How we built it: Xerox* [Accessed: 2024-09-26]. https://techcommunity.microsoft.com/t5/roboprocessautomation/rpa-xerox/ba-p/3653921