AUTOMATIZACIÓN DE LA REVISIÓN DE DATOS PROCESADOS POR UNA ETL PARA EL ÁREA DE PRUEBAS DEL BANCO DE OCCIDENTE

RAY ANDRES AHUMADA CAMPOS

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA DE TELECOMUNICACIONES
PROYECTO DE PRÁCTICAS PROFESIONALES
BOGOTÁ D.C.
2023

AUTOMATIZACIÓN DE LA REVISIÓN DE DATOS PROCESADOS POR UNA ETL PARA EL ÁREA DE PRUEBAS DEL BANCO DE OCCIDENTE

RAY ANDRES AHUMADA CAMPOS

PROYECTO DE PASANTÍA EN BANCO DE OCCIDENTE

DIRIGIDO POR:
PEDRO ALEJANDRO MANCERA LAGOS

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA DE TELECOMUNICACIONES
PROYECTO DE PRACTICAS PROFESIONALES
BOGOTÁ D.C.
2023

Nota de Aceptación

_____

_____

_____

_____

Presidente del Jurado

_____

Jurado

_____

Jurado

26 de octubre del 2023, Bogotá D.C

AGRADECIMIENTOS

Quiero agradecer a todas las personas que han sido parte fundamental en mi vida y en la realización de este proyecto. Especialmente:

A mi familia, por su amor incondicional, su apoyo constante y por haber sido mi mayor inspiración para perseguir mis sueños académicos. Gracias por siempre creer en mí.

A mi tutor, Pedro Mancera, por su invaluable orientación y conocimientos compartidos durante el desarrollo de este proyecto. Su guía y sugerencias fueron fundamentales para enriquecer este proyecto.

A mis profesores y profesoras, quienes han contribuido significativamente a mi formación académica y profesional. Agradezco su dedicación y paciencia en transmitirme conocimientos que han sido vitales para mi crecimiento como ingeniero.

A la Universidad Santo Tomás, por darme la oportunidad de pertenecer tanto a la institución como a la facultad de ingeniería de telecomunicaciones, lugar donde encontré mi vocación y adquirí los conocimientos necesarios para realizar mis objetivos.

A todos aquellos que, de una u otra manera, contribuyeron a la realización de este proyecto, mi más sincero agradecimiento.

CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

Pág.

RESUMEN

El presente documento tiene como propósito evidenciar el proceso de creación de una herramienta automatizada para revisar y asegurar la calidad de los datos procesados por una ETL (Extrac, Transform and Load), la cual es utilizada por el Banco de Occidente para migrar información desde múltiples bases de datos hacia una base de datos central.

Durante este proceso de migración, las tablas de datos están expuestas al riesgo de corrupción debido a posibles errores en el desarrollo de la ETL. Dada la voluminosa cantidad de datos involucrados, realizar esta revisión de manera manual se torna inviable, ya que estas tablas pueden contener millones de registros. Ante esta necesidad, el equipo de pruebas del banco identificó la urgencia de contar con una herramienta que permitiera verificar la correctitud de los archivos antes de ser cargados en la base de datos destino.

Como solución, se desarrolló una herramienta utilizando el lenguaje de programación Python. Esta herramienta toma archivos en formato .csv extraídos de las bases de datos y los compara con los datos almacenados en hojas de cálculo de Excel (.xlsx), proporcionando indicadores de similitud entre ambas fuentes. De esta manera, se verifica la integridad de los archivos una vez procesados. Además, la herramienta ofrece la opción de descargar un archivo en formato .xlsx con una tabla booleana que detalla qué datos han sido modificados, lo cual resulta de gran utilidad para los Analistas de Control de Calidad (QA) al reportar fallos en las ETL suministradas.

En conclusión, se logró desarrollar una interfaz gráfica de usuario que cumple con los objetivos establecidos para asegurar la calidad de los datos procesados por la ETL. La herramienta también posee un potencial significativo para ser implementada en otros equipos de pruebas de software y migración de datos, ofreciendo una solución eficiente y confiable para asegurar la exactitud de los datos en procesos similares.

1. INTRODUCCIÓN

El presente trabajo de grado surge a raíz de una experiencia de aprendizaje llevada a cabo en el Banco de Occidente, donde se colaboró estrechamente con el equipo de pruebas. Durante este proceso, se trabajó con un recurso esencial conocido como ETL (Extract, Transform and Load), el cual permite combinar datos provenientes de distintas fuentes en un repositorio central. Este recurso es creado por desarrolladores que se ajustan a criterios específicos, adaptándose a las necesidades particulares de cada proyecto.

La ejecución de una ETL sigue un flujo previamente establecido que involucra el uso de diversas aplicaciones, entre ellas AWS GLUE, S3, GoAnywhere, entre otras, cada una cumpliendo funciones determinadas para garantizar que los datos extraídos sean correctamente cargados en una base de datos destino.

Sin embargo, durante este recorrido, los datos pueden verse expuestos a posibles problemas de corrupción, lo cual exige que los analistas verifiquen minuciosamente la integridad de la data cargada. Dada la gran cantidad de datos involucrados, que en ocasiones puede alcanzar cifras millonarias, una verificación manual resultaría ineficiente. Como respuesta a esta necesidad actual, el presente proyecto propone el desarrollo de una aplicación que automatice la revisión de los datos cargados por la ETL, brindando una solución efectiva para el equipo de pruebas del Banco de Occidente.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Desarrollar una herramienta en lenguaje de programación Python que automatice los procesos de revisión y aseguramiento de los datos procesados por una ETL.

2.2 OBJETIVOS ESPECÍFICOS

- Analizar los requerimientos y funcionalidades necesarias para la automatización del proceso de revisión de datos en una ETL, en consulta con los analistas de datos y el equipo de pruebas del Banco de Occidente.

- Diseñar, implementar y desplegar una interfaz de usuario intuitiva y amigable utilizando metodologías ágiles SCRUM. Esto incluirá la investigación y selección de las librerías y módulos de Python más apropiados para el procesamiento eficiente de archivos .csv y la lectura de hojas de cálculo en formato .xlsx.

- Verificar la precisión y eficacia de la herramienta desarrollada, asegurándose de que cumpla con la funcionalidad requerida.

3. PLANTEAMIENTO DEL PROBLEMA

3.1 DEFINICIÓN DEL PROBLEMA

El proceso de migración de datos en una ETL (Extract, Transform and Load) constituye una actividad crítica para el Banco de Occidente, ya que permite consolidar información de diversas fuentes en una base de datos central. Sin embargo, este proceso conlleva el riesgo de corrupción de datos debido a posibles errores en el desarrollo de la ETL, lo que afecta la integridad y exactitud de la información almacenada.

El equipo de pruebas del Banco de Occidente ha identificado que la revisión manual de los datos procesados resulta ineficiente y poco factible, especialmente considerando la alta cantidad de datos que pueden estar involucrados en el proceso, pudiendo llegar a millones de registros. Esta situación dificulta la detección temprana de posibles discrepancias o fallas en el proceso de transformación, lo que puede afectar negativamente la calidad de los datos y, por ende, la toma de decisiones y análisis realizados con base en ellos.

Adicionalmente, no se dispone actualmente de una herramienta automatizada que permita verificar de manera efectiva la correctitud de los datos procesados por la ETL antes de ser cargados en la base de datos destino. Esta carencia limita las capacidades de aseguramiento de la calidad de los datos y conlleva a un incremento en los tiempos y costos de revisión y corrección de errores, afectando la eficiencia y eficacia del proceso de migración.

3.2 JUSTIFICACIÓN

Este proyecto se realiza para cubrir la necesidad de una solución tecnológica que automatice el proceso de revisión y aseguramiento de los datos procesados por la ETL en el Banco de Occidente. Esta herramienta deberá ser capaz de comparar los datos extraídos de las bases de datos de origen con los datos transformados por la ETL, identificando discrepancias y errores de manera precisa y oportuna. Además, se requerirá que la herramienta genere un informe detallado de las modificaciones realizadas a los datos, proporcionando información valiosa para el equipo de pruebas y facilitando el proceso de reporte de fallos encontrados.

El desarrollo de esta herramienta no solo optimizará la eficiencia y calidad del proceso de migración de datos en el Banco de Occidente, sino que también tiene potencial para ser implementada en otros equipos de pruebas de software y migración de datos, aportando beneficios significativos en términos de control y aseguramiento de la calidad de los datos procesados por ETLs en distintos contextos organizacionales.

4. MARCO TEÓRICO

4.1 BASES TEORICAS

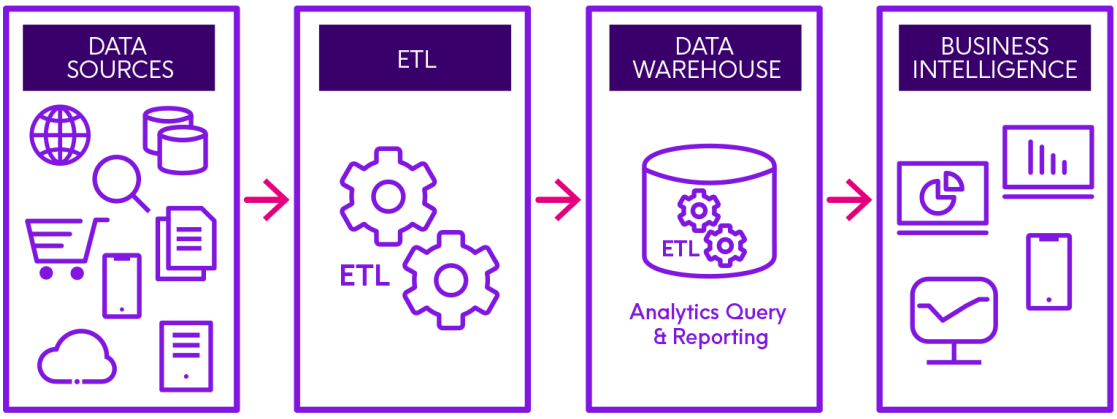
4.1.1 ETL

Extracción, Transformación y Carga (ETL) es un proceso que implica la combinación de datos procedentes de diversas fuentes en un gran repositorio central conocido como almacenamiento de datos. ETL emplea un conjunto de reglas comerciales para limpiar y estructurar los datos en bruto, preparándolos para su almacenamiento, análisis de datos y aplicaciones de aprendizaje automático (Machine Learning).

Este proceso puede satisfacer necesidades específicas de inteligencia empresarial, como la predicción de resultados en decisiones de negocios, generación de informes y paneles, así como la mejora de la eficiencia operativa, entre otros. ETL opera en tres etapas:

1. Extracción: Consiste en obtener los datos relevantes de la base de datos de origen.
2. Transformación: Implica la modificación de los datos para que sean más adecuados para el análisis y uso posterior.
3. Carga: En esta etapa, los datos transformados se cargan en la base de datos de destino de manera periódica.

Figura 1. Flujo general ETL



Fuente: Precisely

4.1.2 AUTOMATIZACION DE PROCESOS

La automatización de procesos implica la transformación operativa de una tarea previamente realizada de forma manual, mediante la aplicación de tecnología y la sincronización de sistemas y datos.

Este proceso ostenta una relevancia significativa en la administración empresarial, dado que conlleva la capacidad de fomentar competitividad, eficiencia y añadir valor al cliente.

De esta manera, se abordará en esta ocasión la comprensión detallada del funcionamiento y el propósito subyacente de la automatización de procesos, así como su implementación en la gestión de procesos de negocio (BPM), y las ventajas inherentes para la organización.

Con el objetivo de simplificar la asimilación, trasladaremos la noción de automatización de procesos a nuestra vida cotidiana. En diversas ocasiones aprovechamos la tecnología para economizar tiempo y energía, por ejemplo, al utilizar aplicaciones de entrega de alimentos a domicilio o al gestionar nuestra vivienda mediante dispositivos inteligentes, que permiten regular la temperatura del aire acondicionado mediante el teléfono móvil.

En una analogía sencilla, podríamos concebir la automatización de procesos como una idea similar, si bien aplicada al ámbito empresarial concreto.

La primordial finalidad de la automatización de procesos radica en la optimización del flujo de trabajo dentro de una organización.

Por medio de la automatización se puede disminuir costos, tiempo y desperdicio, incrementar la productividad, reducir fallos y supervisar, en tiempo real, todas las operaciones comerciales.

Es viable reemplazar actividades manuales por procedimientos automatizados o emplear programas y sistemas para respaldar una variedad de actividades.

4.1.3 CALIDAD DE DATOS

La excelencia de los datos trasciende meramente la percepción. No existen normativas estandarizadas ni un enfoque único en lo concerniente a la calidad de la información. Sustentar la precisión y la integridad de todas las categorías de datos a lo largo de toda la entidad implica laborar en pro de su idoneidad para cumplir su propósito en un contexto específico. Esto conlleva asegurar que cada dato posea todos los atributos necesarios, que abarcan:

1. Exactitud
2. Integridad
3. Actualización
4. Pertinencia
5. Coherencia
6. Fiabilidad
7. Presentación adecuada
8. Accesibilidad

En el seno de una organización, la calidad de los datos ostenta una importancia crucial para la uniformidad de los informes, la confianza de los usuarios y la eficacia de los procesos operativos y transaccionales. La toma de decisiones basada en inteligencia empresarial demanda la cimentación en datos de alta calidad. Para garantizar que estos datos alcancen el nivel deseado, resulta fundamental supervisar cada interacción con los mismos, desde su entrada hasta su almacenamiento y gestión.

La garantía de la calidad de los datos es el procedimiento que verifica la confiabilidad y eficacia de los datos, llevado a cabo de manera periódica, que engloba acciones como la actualización, normalización y eliminación de duplicados. Cada organización debe buscar alcanzar una visión coherente de la realidad, aun cuando para edificar su comprensión requiera emplear datos diversos, almacenados en sistemas heterogéneos o procedentes de fuentes dispares.

Con miras a asegurar la calidad de los datos, el mercado presenta múltiples soluciones que facilitan los procesos de depuración, perfilado y conciliación de datos. Estas contribuyen a obtener resultados superiores en un lapso menor, gracias a la automatización, que simultáneamente reduce la tasa de errores en el proceso.

4.1.4 LENGUAJE DE PROGRAMACION PYTHON Y AUTOMATIZACIÓN

Python es un lenguaje de programación de uso generalizado en aplicaciones web, desarrollo de software, ciencia de datos y aprendizaje automático (machine learning, ML). Los profesionales optan por Python debido a su eficiencia y facilidad de adquisición, además de su capacidad para operar en múltiples plataformas. El software Python se halla disponible gratuitamente, se integra de manera fluida con diversos sistemas y potencia la velocidad de desarrollo.

El lenguaje Python se emplea en múltiples contextos en el desarrollo de aplicaciones, entre ellos, la automatización.

Automatización mediante Scripts en Python

Un lenguaje de script es aquel que automatiza tareas comúnmente realizadas por personas. Los programadores recurren ampliamente a los scripts en Python para automatizar diversas labores cotidianas, tales como:

1. Renombrar múltiples archivos en conjunto.
2. Convertir un archivo a otro formato.
3. Eliminar duplicados de archivos de texto.
4. Realizar operaciones matemáticas elementales.
5. Enviar correos electrónicos.
6. Descargar contenido.
7. Llevar a cabo análisis básicos de registros.
8. Identificar fallas en diversos archivos.

Automatización de Pruebas de Software

Las pruebas de software abarcan la validación de si los resultados reales del software coinciden con los resultados anticipados, con el objetivo de asegurar la ausencia de errores.

Para probar las funciones que escriben, los desarrolladores emplean marcos de pruebas unitarias en Python, como Unittest, Robot y PyUnit.
Para elaborar casos de prueba para diversos escenarios, los evaluadores de software utilizan Python. Esto abarca pruebas de la interfaz de usuario de aplicaciones web, componentes de software y nuevas características.

Los programadores cuentan con distintas herramientas para ejecutar scripts de pruebas de manera automatizada. Estas herramientas se reconocen como herramientas de integración y despliegue continuo (CI/CD). Los equipos de evaluación y desarrollo de software recurren a herramientas CI/CD como Travis CI y Jenkins para automatizar pruebas. Estas herramientas ejecutan automáticamente los scripts de pruebas en Python y comunican los resultados cada vez que los programadores introducen cambios en el código.

4.2 BASES LEGALES

En el contexto de Colombia, no se encontraba previamente una regulación específica que abordara de forma singular los procedimientos de ETL (Extract, Transform and Load). No obstante, la instauración de los procesos ETL engloba facetas vinculadas a la salvaguarda de datos personales y la preservación de la seguridad de la información. Por lo tanto, es esencial acatar las regulaciones generales que imperan en el país respecto a la protección de datos y la ciberseguridad. A continuación, se enumeran algunas normativas de relevancia en esta materia:

1. Ley 1581 de 2012 - Ley de Protección de Datos Personales: Esta ley establece las disposiciones generales para la protección de los datos personales en Colombia. Se aplica a todas las entidades que traten datos personales y establece los principios, derechos y deberes para el manejo adecuado de esta información.

2. Decreto 1074 de 2015 - Reglamentario del Sector Comercio, Industria y Turismo: Este decreto regula aspectos específicos relacionados con la protección de datos personales y detalla los procedimientos para el cumplimiento de la Ley 1581 de 2012.

3. Resolución 20171100156285 de 2017 - Política de Seguridad de la Información: Esta resolución establece las pautas y lineamientos para la implementación de la Política de Seguridad de la Información en las entidades públicas del orden nacional.

4. Normas técnicas sobre seguridad de la información ISO/IEC 27000: Estas normas internacionales proporcionan directrices y mejores prácticas para la implementación de sistemas de gestión de seguridad de la información (SGSI), lo que es relevante para garantizar la integridad y confidencialidad de los datos manejados en un proceso ETL.

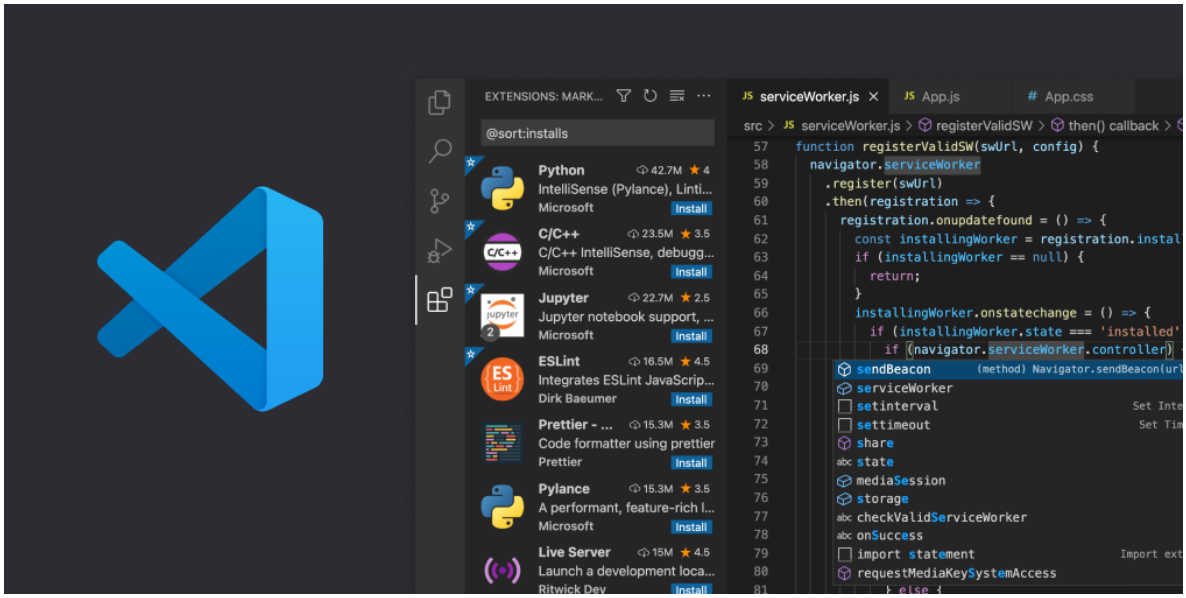
5. MATERIALES Y MÉTODOS

5.1 HERRAMIENTAS Y TECNOLOGIAS USADAS

5.1.1 VISUAL STUDIO CODE

Visual Studio Code es un editor de código fuente desarrollado por Microsoft. Es ampliamente utilizado por desarrolladores para escribir código en diversos lenguajes de programación. Aunque se llama "Visual Studio", no es una versión completa del entorno de desarrollo integrado (IDE) Visual Studio, sino un editor de texto con funciones avanzadas para programadores.

Figura 2. Visual Studio Code



Fuente: Microsoft

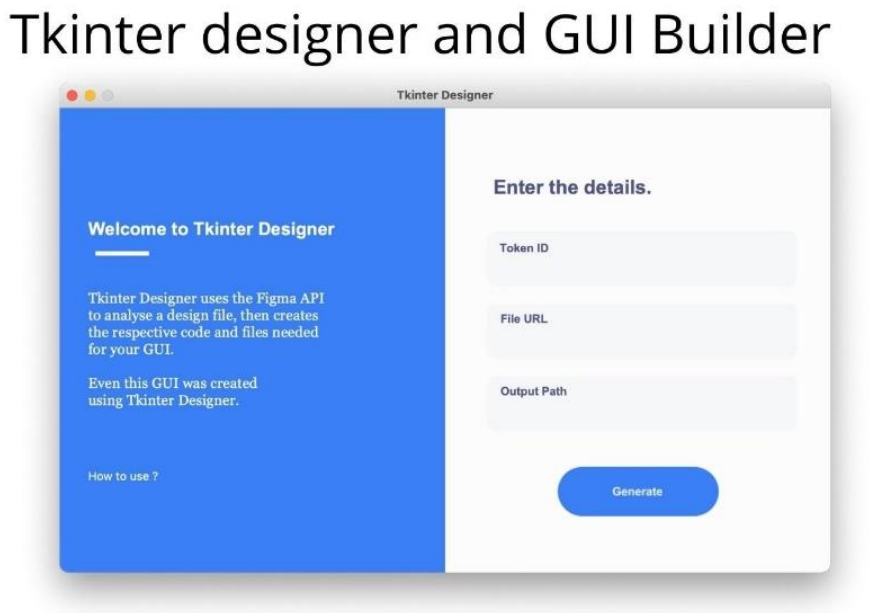
Visual Studio Code (VS Code) es principalmente un editor de código, pero no ofrece una funcionalidad nativa para el diseño de interfaces de usuario de manera visual, como lo haría un entorno de desarrollo integrado (IDE) dedicado a la creación de aplicaciones con interfaces gráficas. Sin embargo, es posible utilizar VS Code junto con extensiones y bibliotecas para desarrollar interfaces de usuario como Tkinter.

5.1.2 TKINTER

El módulo "tkinter" (abreviatura de "interfaz Tk") constituye la interfaz estándar de Python para el conjunto de herramientas de interfaz gráfica de usuario (GUI) conocido como Tk. Tanto Tk como la biblioteca "tkinter" se encuentran disponibles en una amplia variedad de plataformas Unix, así como en sistemas Windows. Es importante destacar que Tk en sí no es un componente nativo de Python, sino que es mantenido por la entidad ActiveState.

Aunque "Tkinter" no es una implementación liviana, introduce diversas capas de lógica adicional que favorecen la coherencia con el estilo de programación en Python. Esta documentación se enfocará en analizar estas adiciones y modificaciones, remitiendo a la documentación oficial de Tcl-Tk para abordar aspectos que no han experimentado cambios.

Figura 3. Ejemplo interfaz gráfica de usuario hecha con Tkinter



Fuente: Electroica Blog

5.1.3 PANDAS

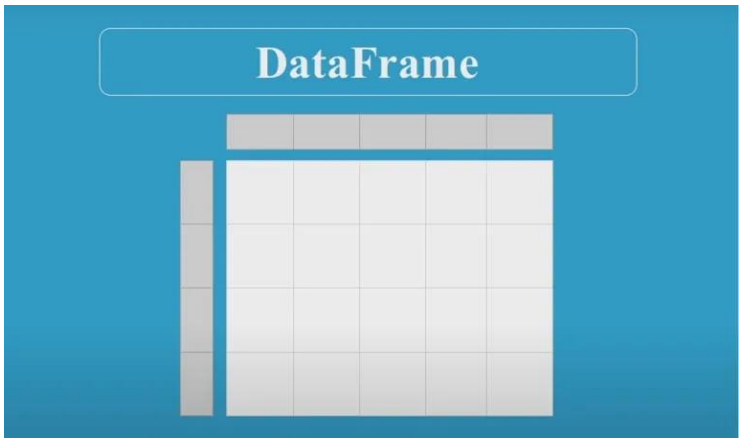
La biblioteca de código abierto Pandas está específicamente concebida para la manipulación y análisis de datos dentro del entorno del lenguaje de programación Python. Sus atributos de potencia, flexibilidad y facilidad de uso hacen de Pandas una herramienta fundamental en este ámbito.

La utilidad de Pandas radica en permitir la carga, alineación, manipulación e incluso la combinación de datos utilizando Python. La eficacia de su rendimiento es particularmente notable cuando el código de su núcleo está desarrollado en C o Python.

El término "Pandas" deriva de "Panel Data", aludiendo a conjuntos de datos que contienen observaciones a lo largo de múltiples períodos temporales. La biblioteca se originó con la intención de ser una solución de nivel superior para el análisis en el entorno Python.

Pandas opera sobre DataFrames, que son estructuras de datos bidimensionales similares a tablas, donde cada columna representa una variable y cada fila contiene un conjunto de valores. Los datos almacenados pueden ser numéricos o caracteres. Los profesionales en ciencia de datos y programadores familiarizados con R emplean DataFrames para organizar información de manera eficiente, motivo por el cual Pandas es ampliamente utilizado en aplicaciones de Machine Learning. Esta herramienta permite importar/exportar datos en diversos formatos, como CSV o JSON, y también ofrece capacidades de limpieza de datos. Pandas es especialmente eficaz para el análisis de datos estadísticos, información tabular como tablas SQL o Excel, series temporales y matrices con etiquetas de filas y columnas.

Figura 4. DataFrame



Fuente: DataScientest

5.1.4 NUMPY

NumPy representa una biblioteca de programación en el lenguaje Python de suma relevancia en el ámbito de la computación científica. El término "NumPy" deriva de la conjunción de las palabras "Python numérico", denotando así su enfoque en la manipulación y análisis de datos numéricos. La descripción oficial de esta librería, como se halla en su página web, brinda un esclarecedor panorama acerca de sus atributos y utilidades.

NumPy se erige como el pilar esencial para la ejecución de tareas de computación científica en el contexto de Python. Su amplio conjunto de capacidades abarca, entre otros elementos:

- Un versátil y robusto objeto de arreglo con dimensiones N, propiciando la representación y manipulación eficiente de datos complejos.
- Funcionalidades avanzadas de radiodifusión, que permiten la aplicación de operaciones sobre arreglos de diferentes tamaños y formas con una elevada eficacia.
- Instrumentos para la integración de código desarrollado en lenguajes como C/C++ y Fortran, facilitando la optimización y aceleración de algoritmos.
- Potentes capacidades en ámbitos como el álgebra lineal, las transformaciones de Fourier y la generación de números aleatorios, contribuyendo así a la resolución de una amplia gama de problemas científicos y matemáticos.

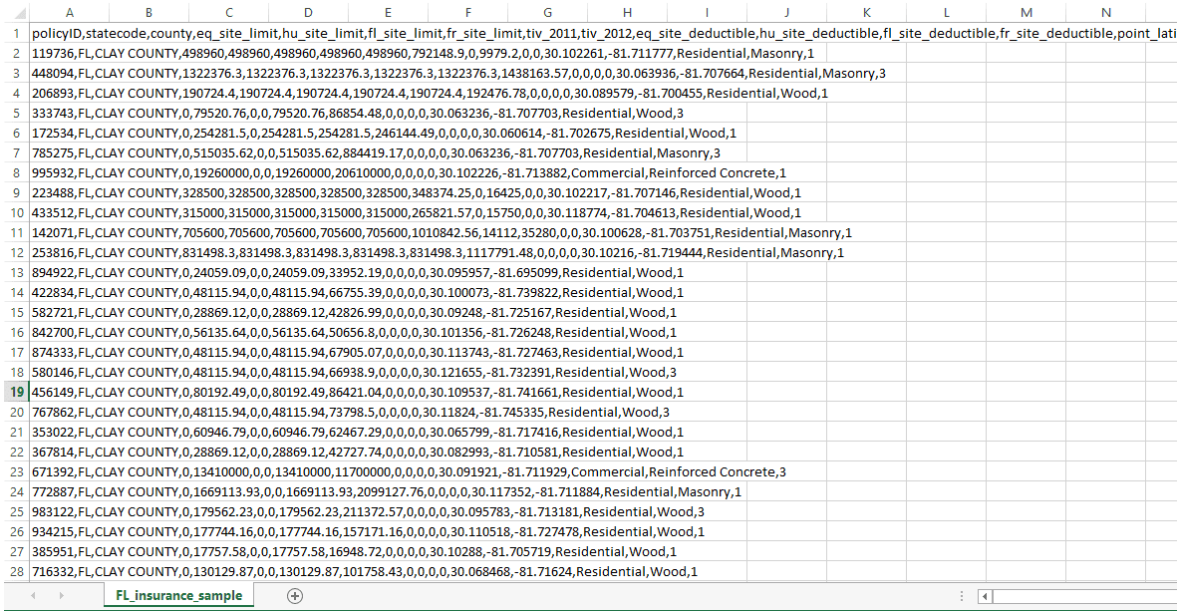
5.1.5 ARCHIVOS .CSV

Un archivo CSV, acrónimo de "Comma Separated Values" o "Valores Delimitados por Comas", constituye un formato de archivo que representa un conjunto de datos mediante una estructura tabular dividida por comas, o en determinados contextos geográficos, puntos y comas.

Los archivos de texto confeccionados en el formato CSV adoptan un diseño en el cual la información se dispone en filas y columnas. Cada columna es identificada por un encabezado que define la categoría correspondiente a un tipo de dato específico. La separación entre estas columnas se efectúa mediante comas. Por su parte, las filas son representativas de las líneas individuales en la tabla, y contienen los datos que integran el conjunto de información bajo consideración.

Un atributo distintivo del formato CSV es su amplia capacidad de visualización en diversas aplicaciones, lo que refleja su cualidad de ser compatible con múltiples plataformas. Esto implica que los archivos en formato CSV son susceptibles de ser procesados, editados y almacenados en distintos entornos y programas informáticos, contribuyendo así a su alta versatilidad y utilidad en diversas situaciones y contextos.

Figura 5. Archivo .CSV



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	policyID,statecode,county,eq_site_limit,hu_site_limit,fl_site_limit,fr_site_limit,tiv_2011,tiv_2012,eq_site_deductible,hu_site_deductible,fl_site_deductible,fr_site_deductible,point_lati													
2	119736,FL,CLAY COUNTY,498960,498960,498960,498960,498960,792148.9,0,9979.2,0,0,30.102261,-81.711777,Residential,Masonry,1													
3	448094,FL,CLAY COUNTY,1322376.3,1322376.3,1322376.3,1322376.3,1322376.3,1438163.57,0,0,0,0,30.063936,-81.707664,Residential,Masonry,3													
4	206893,FL,CLAY COUNTY,190724.4,190724.4,190724.4,190724.4,190724.4,192476.78,0,0,0,0,30.089579,-81.700455,Residential,Wood,1													
5	333743,FL,CLAY COUNTY,0,79520.76,0,0,79520.76,86854.48,0,0,0,0,30.063236,-81.707703,Residential,Wood,3													
6	172534,FL,CLAY COUNTY,0,254281.5,0,254281.5,254281.5,246144.49,0,0,0,0,30.060614,-81.702675,Residential,Wood,1													
7	785275,FL,CLAY COUNTY,0,515035.62,0,0,515035.62,884419.17,0,0,0,0,30.063236,-81.707703,Residential,Masonry,3													
8	995932,FL,CLAY COUNTY,0,19260000,0,0,19260000,20610000,0,0,0,0,30.102226,-81.713882,Commercial,Reinforced Concrete,1													
9	223488,FL,CLAY COUNTY,328500,328500,328500,328500,328500,348374.25,0,16425,0,0,30.102217,-81.707146,Residential,Wood,1													
10	433512,FL,CLAY COUNTY,315000,315000,315000,315000,315000,265821.57,0,15750,0,0,30.118774,-81.704613,Residential,Wood,1													
11	142071,FL,CLAY COUNTY,705600,705600,705600,705600,705600,1010842.56,14112,35280,0,0,30.100628,-81.703751,Residential,Masonry,1													
12	253816,FL,CLAY COUNTY,831498.3,831498.3,831498.3,831498.3,831498.3,1117791.48,0,0,0,0,30.10216,-81.719444,Residential,Masonry,1													
13	894922,FL,CLAY COUNTY,0,24059.09,0,0,24059.09,33952.19,0,0,0,0,30.095957,-81.695099,Residential,Wood,1													
14	422834,FL,CLAY COUNTY,0,48115.94,0,0,48115.94,66755.39,0,0,0,0,30.100073,-81.739822,Residential,Wood,1													
15	582721,FL,CLAY COUNTY,0,28869.12,0,0,28869.12,42826.99,0,0,0,0,30.09248,-81.725167,Residential,Wood,1													
16	842700,FL,CLAY COUNTY,0,56135.64,0,0,56135.64,50656.8,0,0,0,0,30.101356,-81.726248,Residential,Wood,1													
17	874333,FL,CLAY COUNTY,0,48115.94,0,0,48115.94,67905.07,0,0,0,0,30.113743,-81.727463,Residential,Wood,1													
18	580146,FL,CLAY COUNTY,0,48115.94,0,0,48115.94,66938.9,0,0,0,0,30.121655,-81.732391,Residential,Wood,3													
19	456149,FL,CLAY COUNTY,0,80192.49,0,0,80192.49,86421.04,0,0,0,0,30.109537,-81.741661,Residential,Wood,1													
20	767862,FL,CLAY COUNTY,0,48115.94,0,0,48115.94,73798.5,0,0,0,0,30.11824,-81.745335,Residential,Wood,3													
21	353022,FL,CLAY COUNTY,0,60946.79,0,0,60946.79,62467.29,0,0,0,0,30.065799,-81.717416,Residential,Wood,1													
22	367814,FL,CLAY COUNTY,0,28869.12,0,0,28869.12,42727.74,0,0,0,0,30.082993,-81.710581,Residential,Wood,1													
23	671392,FL,CLAY COUNTY,0,13410000,0,0,13410000,11700000,0,0,0,0,30.091921,-81.711929,Commercial,Reinforced Concrete,3													
24	772887,FL,CLAY COUNTY,0,1669113.93,0,0,1669113.93,2099127.76,0,0,0,0,30.117352,-81.711884,Residential,Masonry,1													
25	983122,FL,CLAY COUNTY,0,179562.23,0,0,179562.23,211372.57,0,0,0,0,30.095783,-81.713181,Residential,Wood,3													
26	934215,FL,CLAY COUNTY,0,177744.16,0,0,177744.16,157171.16,0,0,0,0,30.110518,-81.727478,Residential,Wood,1													
27	385951,FL,CLAY COUNTY,0,17757.58,0,0,17757.58,16948.72,0,0,0,0,30.10288,-81.705719,Residential,Wood,1													
28	716332,FL,CLAY COUNTY,0,130129.87,0,0,130129.87,101758.43,0,0,0,0,30.068468,-81.71624,Residential,Wood,1													

FL_insurance_sample

Fuente: Profesional Review

5.1.6 ARCHIVOS .XLSX

Un archivo XLSX es un formato de archivo ampliamente reconocido y utilizado en el ámbito de la informática y la gestión de datos. La denominación "XLSX" alude a "eXceL eXtensible" y representa una evolución del formato previo "XLS", vinculado a hojas de cálculo generadas por el software Microsoft Excel y afines. El archivo XLSX, diseñado según las pautas del estándar Open XML, se caracteriza por su capacidad para contener y estructurar datos de naturaleza diversa, principalmente en forma de tablas y hojas de cálculo.

En su esencia, un archivo XLSX se configura como una colección organizada de información dispuesta en filas y columnas, similar a una matriz bidimensional. Cada columna y fila puede albergar valores numéricos, textuales o de otro tipo, con la posibilidad de asignar fórmulas, funciones y formatos específicos a las celdas

individuales. La estructura de un archivo XLSX permite la creación de hojas de cálculo complejas que incluyen cálculos automatizados, representaciones gráficas, tablas dinámicas y otras características avanzadas de análisis y presentación de datos.

5.2 METODOLOGÍA

La metodología adoptada para la elaboración del proyecto encuentra sus fundamentos en los conceptos adquiridos y aplicados en el seno del equipo de pruebas perteneciente al Banco de Occidente. El propósito subyacente radica en alcanzar una apropiada preparación de los elementos indispensables para la ejecución del proyecto, conjuntamente con la asimilación de conceptos inherentes a las metodologías ágiles, las cuales se revelan como instrumentos sumamente valiosos para forjar, de manera expedita, la interfaz de usuario.

A continuación, vemos un diagrama que muestra las fases y su respectivo orden:

Figura 6. Fases del proyecto

Fuente: Elaboración propia

6. DESARROLLO DEL PROYECTO

6.1 ACTIVIDADES PREVIAS

6.1.1 PLAN DE PROYECTO

El plan de proyecto es el pilar fundamental del proyecto. Actúa como un mapa detallado que guía cada paso de mi proyecto, desde la definición de objetivos hasta la prueba de la interfaz gráfica. Sin este plan, sería fácil perderse en la complejidad de mi proyecto, lo que podría llevar a retrasos, confusiones y resultados insatisfactorios. Gracias al plan de proyecto, puedo mantener un enfoque claro, asignar recursos de manera eficiente, identificar y mitigar riesgos, y asegurarme de que mi proyecto sea un éxito tanto en términos académicos como profesionales. Es la herramienta esencial que me lleva de manera efectiva desde la idea inicial hasta la conclusión exitosa del proyecto.

Tabla 1. Plan de proyecto

Tareas	Fecha de inicio	Fecha final	Días	Estado
Actividades previas				
Definición de objetivos y requisitos	7/31/2023	8/1/2023	1	**Sin empezar**
Recursos y herramientas	8/1/2023	8/2/2023	1	**Sin empezar**
Plan de proyecto	8/3/2023	8/4/2023	1	**Sin empezar**
Preparación entorno desarrollo	8/7/2023	8/8/2023	1	**Sin empezar**
Diseño			0	
Plan de casos de prueba	8/9/2023	8/11/2023	2	**Sin empezar**
Capas	8/14/2023	8/16/2023	2	**Sin empezar**
Preparación de datos			0	
Archivos .csv para pruebas	8/17/2023	8/18/2023	1	**Sin empezar**
Programación			0	

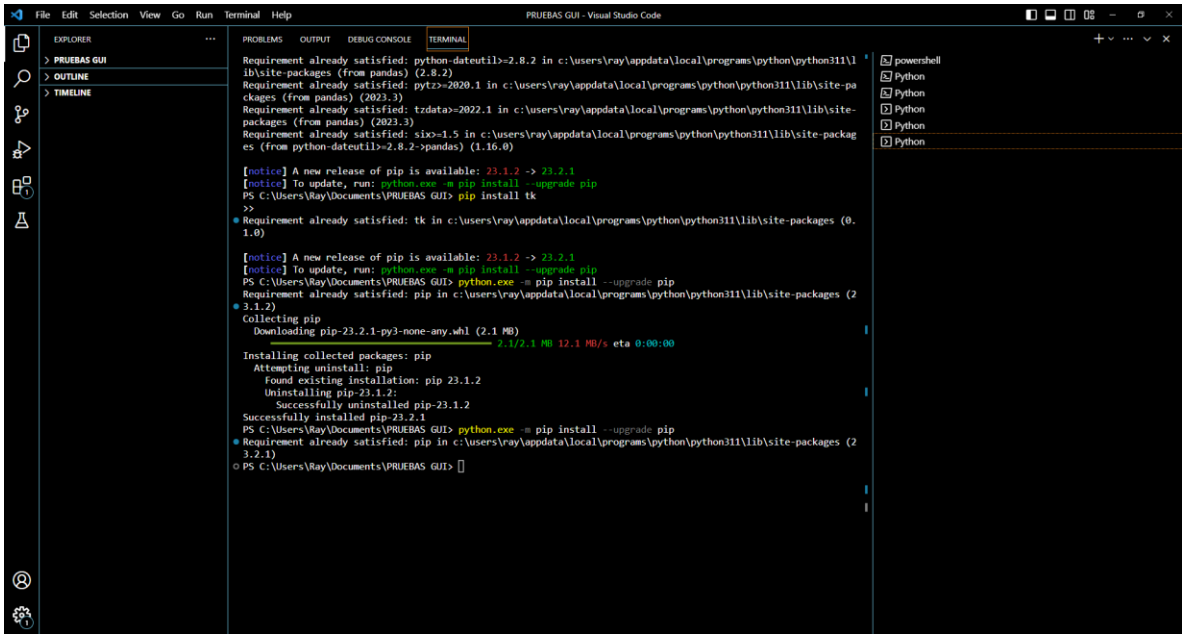
Desarrollo funcionalidad	8/21/2023	8/25/2023	4	**Sin empezar**
Desarrollo Interfaz grafica	8/28/2023	9/1/2023	4	**Sin empezar**
Pruebas	9/4/2023	9/5/2023	1	**Sin empezar**

Fuente: Elaboración propia

6.1.2 PREPARACION ENTORNO DE DESARROLLO

Este apartado detalla el proceso de preparación del entorno de desarrollo en Visual Studio Code (VS Code) para el proyecto. Se ha realizado la instalación de las bibliotecas NumPy, Pandas y Tkinter para asegurar que el entorno esté listo y configurado adecuadamente para comenzar el desarrollo.

Figura 7. Instalación librerías



Fuente: Elaboración propia

Con la instalación exitosa de Visual Studio Code y las bibliotecas NumPy, Pandas y Tkinter, el entorno de desarrollo está completamente preparado para el inicio del proyecto. Este entorno proporciona las herramientas necesarias para la manipulación de datos, análisis y creación de interfaces gráficas de usuario de manera efectiva y eficiente. Con esta configuración, el equipo de desarrollo puede

comenzar a trabajar en el proyecto con todas las bibliotecas y herramientas necesarias a su disposición.

6.2 DISEÑO

6.2.1 PLAN DE CASOS DE PRUEBA

El plan de casos de prueba desempeña un papel crítico en el proyecto. Este plan actúa como una hoja de ruta esencial que me guía en la evaluación sistemática de la GUI para asegurarme de que funcione de manera efectiva y cumpla con los objetivos del proyecto.

El plan de casos de prueba es un componente crítico de una GUI. No solo garantiza la calidad y el cumplimiento de los requisitos, sino que también mejora la experiencia del usuario final al identificar y resolver problemas antes de su implementación.

Tabla 2. Diseño de casos de prueba

Tipo de prueba	Descripción	Ejecución desde Visual Studio Code	Cantidad	Funcionalidad / Característica
Pruebas de funciones GUI	Verifica que todas las funciones de la GUI funcionen según lo previsto	Si	1	Cumpla con todos los requerimientos de la GUI
Pruebas de carga	Evalúa como la GUI se comporta bajo cargas de trabajo elevadas	Si	1	Funcione correctamente con archivos pesados
Pruebas de validación de datos	Asegura que los datos ingresados por el usuario sean validos	Si	1	Verificar que las tablas se procesan correctamente

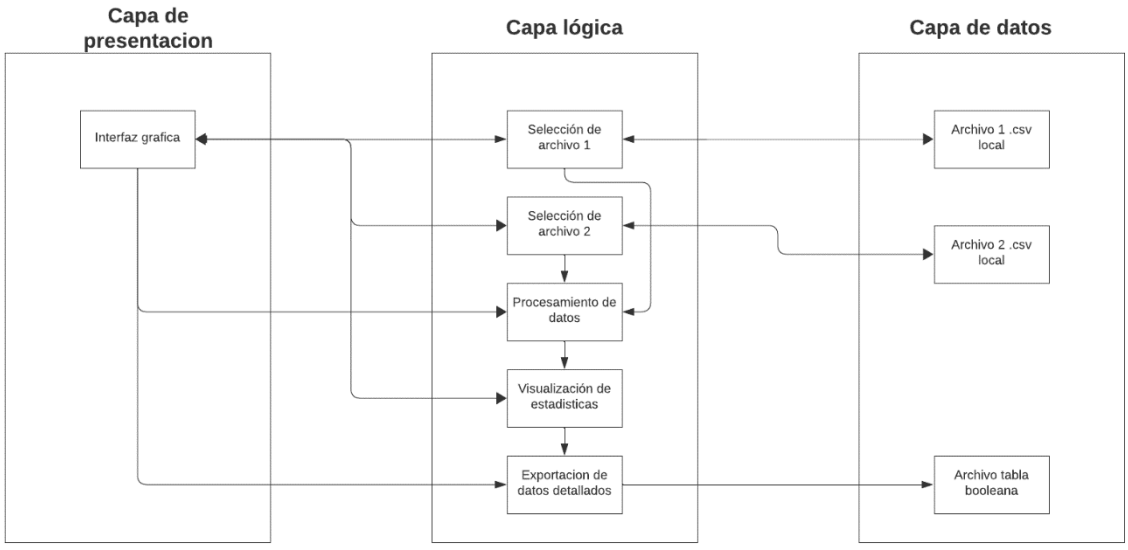
Pruebas de flujo de trabajo	Evalúa si los flujos de trabajo en la GUI son intuitivos y eficientes	Si	1	Validación de un tercero sobre las funciones de la GUI

Fuente: Elaboración propia

6.2.2 CAPAS

El diseño de capas en una GUI es esencial para garantizar una experiencia de usuario eficiente, simplificar el mantenimiento, permitir la escalabilidad, fomentar la reutilización de código y fortalecer la seguridad de las aplicaciones. Es un enfoque clave en el desarrollo de software que conduce a sistemas más robustos y efectivos.

Figura 8. Capas de la GUI



Fuente: Elaboración propia

6.3 PREPARACION DE DATOS

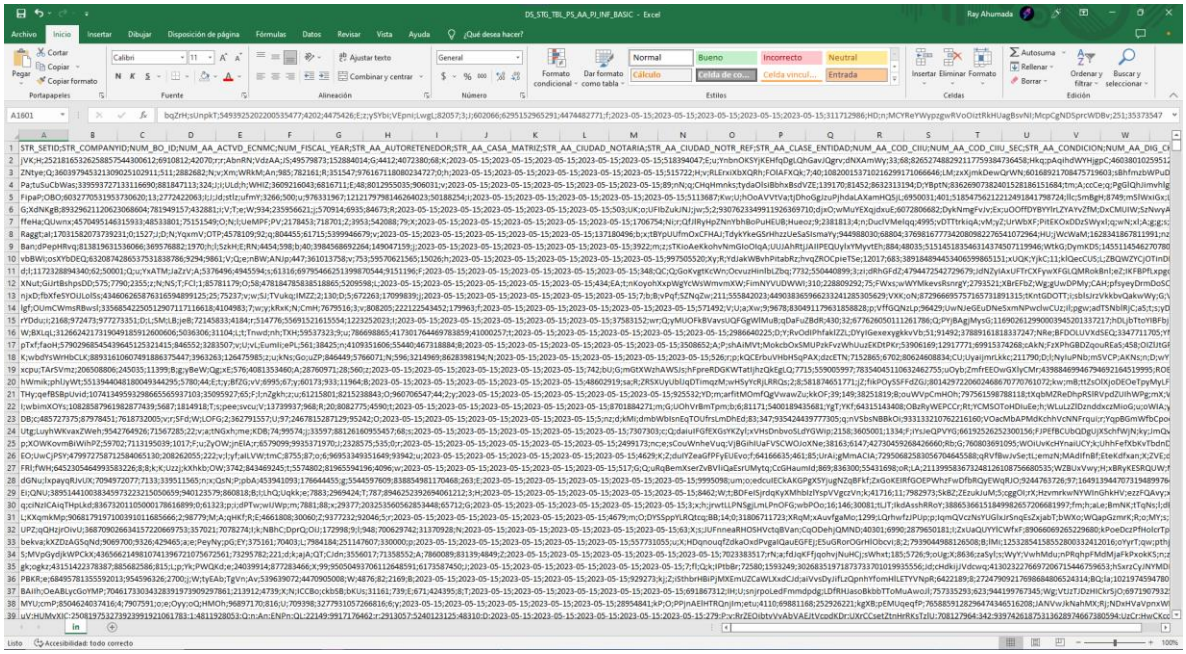
La adecuada preparación de los datos se erige como un elemento de vital importancia, dado que constituye el insumo fundamental destinado a la ejecución de las pruebas previamente definidas, con el propósito de garantizar la calidad de la interfaz de usuario. Es esencial contar con un archivo base que desempeñe el rol de emular el procedimiento de verificación de los datos, y este archivo debe respetar

rigurosamente la estructura característica de los datos que habitualmente son procesados por sistemas de Extracción, Transformación y Carga (ETL).

Para ello se va a realizar la ejecución de los casos de prueba con una ayuda de una carpeta previamente preparada con el insumo base que simula la data cargada en una base de datos destino. Este archivo de nombre “DS_STG_TBL_PS_AA_PJ_INF_BASIC” fue suministrado en el proyecto de profesional en entrenamiento trabajando como analista QA para el proyecto CRM SIEBEL del Banco de Occidente. Es un archivo Excel que contiene datos extraídos de la base de datos origen.

Con ese archivo tenemos un grupo de datos reales para ejecutar pruebas y certificar el correcto funcionamiento de la interfaz. A este archivo lo vamos a definir como “Archivo base” para la descripción de los datos preparados para cada prueba.

Figura 9. Archivo base



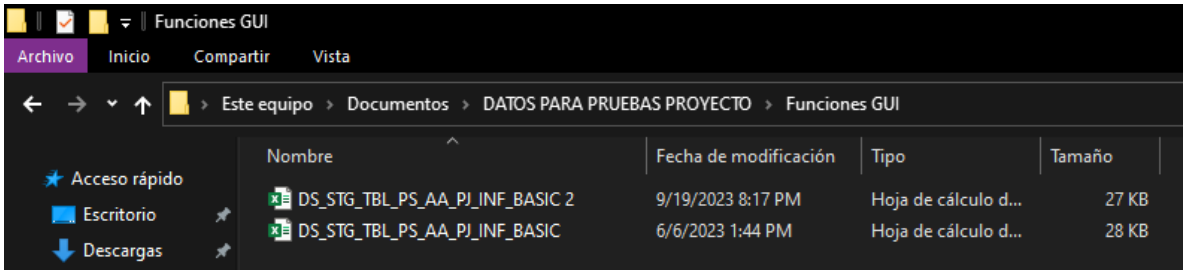
Fuente: Elaboración propia

6.3.1 PRUEBAS FUNCIONALIDAD GUI

Para los datos de prueba de funcionalidad se va a tener dos archivos en el cual el primero va a ser idéntico al Archivo base y el otro va a tener 5 filas modificadas de las 50. El criterio de aceptación es que la interfaz debe arrojar que los archivos

tienen un 90% de correctitud ya que únicamente se alteró el 10% de los datos. Adicionalmente, debe generar y exportar el archivo .xlsm con la tabla booleana que contiene el detalle de la revisión de los datos.

Figura 10. Carpeta Data Funcionalidad



Fuente: Elaboración propia

6.3.2 PRUEBAS VALIDACION DE DATOS

Para los datos de prueba de validación de datos se va a tener dos archivos gemelos. El criterio de aceptación es que la interfaz debe arrojar que los archivos tienen un 100% de correctitud ya que al ser archivos gemelos no debería haber modificación en los datos al ser procesados por la herramienta.

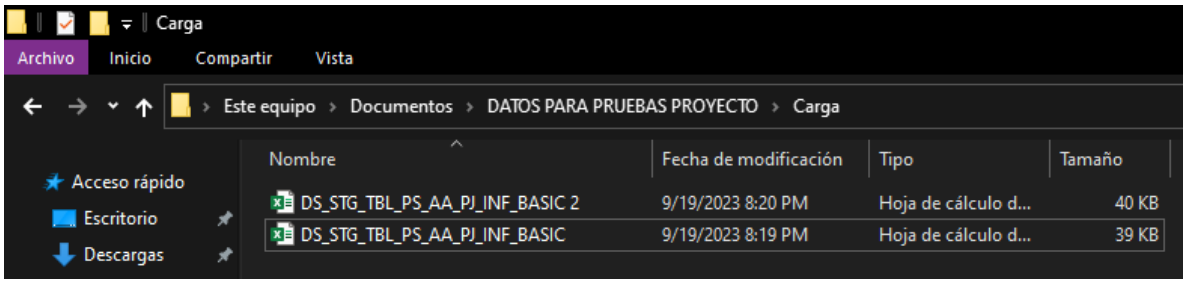
Figura 11. Carpeta Data Validación de Datos

Fuente: Elaboración propia

6.3.3 PRUEBAS DE CARGA

Para los datos de prueba de carga se va a tener dos archivos que hayan aumentado de 50 filas de datos a 1600 cada uno. El criterio de aceptación es que la interfaz debe funcionar con normalidad en tiempos prudentes ya que el peso de los archivos no debería afectar la funcionalidad.

Figura 12. Carpeta Data Carga



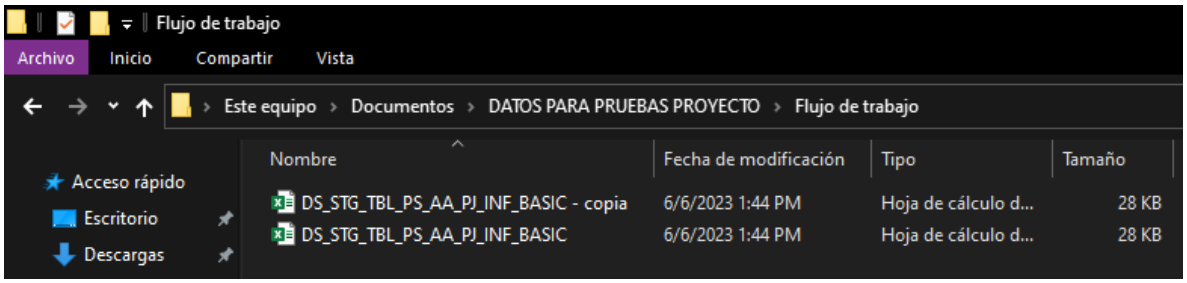
Fuente: Elaboración propia

6.3.4 PRUEBAS FLUJO DE TRABAJO

Para los datos de prueba de carga se va a tener dos archivos. El criterio de aceptación es que un usuario ejecute la herramienta y valide la funcionalidad de las capas entre sí y que demuestre que el flujo de ejecución es intuitivo.

Usuario: Cristian Alberto Diaz – Analista QA del Banco de Occidente

Figura 13. Carpeta Data Flujo de Trabajo



Fuente: Elaboración propia

6.4 PROGRAMACIÓN

Durante el inicio de la fase de desarrollo de la interfaz, se consideraron todas las definiciones previamente establecidas, con el propósito de seguir sus directrices. La exposición del código desarrollado se llevará a cabo por secciones específicas, las cuales abarcan las siguientes áreas: Librerías, Creación de la Ventana, Funciones y Contenido de Interfaz. Este enfoque permitirá garantizar la transparencia en relación con la arquitectura subyacente de la interfaz gráfica.

6.4.1 LIBRERIAS

En esta sección se importan las librerías Tkinter, Pandas y Numpy previamente descritas con su respectivo funcionamiento.

Figura 14. Librerías

```
1   import tkinter
2   from tkinter import *
3   from tkinter import filedialog
4   import pandas as pd
5   import numpy as ps
```

Fuente: Elaboración propia

6.4.2 CREACIÓN DE LA VENTANA

Esta sección de código en Python con la biblioteca Tkinter están creando una ventana de aplicación gráfica con un conjunto de características específicas:

1. `ventana = tkinter.Tk()`: Crea una ventana principal para la interfaz gráfica de la aplicación.

2. `ventana.geometry("1000x500")`: Define la geometría de la ventana, lo que significa que se establece el tamaño de la ventana en píxeles con un ancho de 1000 píxeles y una altura de 500 píxeles.

3. `ventana.configure(background="black")`: Establece el color de fondo de la ventana en negro.

4. `etiqueta = tkinter.Label(ventana, text="Prueba Data Auto", font=("Arial", 14), bg="Black", fg="#009092")`: Crea una etiqueta de texto en la ventana que muestra el texto "Prueba Data Auto". Se especifica la fuente como "Arial" con un tamaño de 14 puntos, el color de fondo como negro y el color de texto como un tono de verde azulado.

5. `etiqueta.pack()`: Coloca la etiqueta en la ventana, usando el método `pack()` para la disposición predeterminada. El método `pack()` organiza los widgets en bloques antes de colocarlos en la ventana principal.

Figura 15. Ventana

```
ventana = tkinter.Tk()
ventana.geometry("1000x500")
ventana.configure(background="black")
etiqueta = tkinter.Label(ventana, text = "Prueba Data Auto", font=("Arial", 14), bg="Black", fg="#009092",)
etiqueta.pack()
```

Fuente: Elaboración propia

6.4.3 FUNCIONES

En esta sección vamos a dividir las funciones por bloques del 1 al 4 para precisar la definición de cada uno.

Figura 16. Funciones

```
def abrirArchivo():                                                                                    1

    global s3
    global s2
    archivo1 = filedialog.askopenfilename(title="abrir", initialdir="C:/")
    s= pd.read_excel(archivo1)
    s2= ps.array(s)
    s3 = pd.DataFrame(s2, columns=['DATA'])
    label_nombre_archivo1.config(text="Archivo cargado: " + archivo1)

def abrirArchivo2():                                                                                   2

    global a3
    global a2
    archivo2 = filedialog.askopenfilename(title="abrir", initialdir="C:/")
    a= pd. read_excel(archivo2)
    a2 =ps.array(a)
    a3 = pd.DataFrame(a2, columns=['DATA'])
    label_nombre_archivo2.config(text="Archivo cargado: " + archivo2)

def leerExcel():                                                                                       3
    global archivoFinal
    contador = 0
    global tablaBooleana
    tablaBooleana = s3 == a3

    vector = s2 == a2
    valor1 = sum(vector)
    valor2 = len(vector)
    operacion = ((valor1 / valor2)*100)

    mensaje = (f"Los archivos tienen un {operacion} % de correctitud")
    lblMensaje = tkinter.Label(ventana, text = mensaje,  bg="black", fg="#39ff14", font=("Arial", 12)).pack()

def exportarArchivo():                                                                                 4
    df = pd.DataFrame(tablaBooleana)
    nombre_archivo = 'DATA_ETL_REVISADA.xlsx'
    print(tablaBooleana)
    df.to_excel(nombre_archivo, sheet_name='Datos de comparación', index=False, engine='openpyxl')
```

Fuente: Elaboración propia

Bloque 1

Esta función de Python, `abrirArchivo()`, está diseñada para llevar a cabo las siguientes tareas:

1. Declara las variables globales `s3` y `s2`.
2. Utiliza `filedialog.askopenfilename()` para permitir al usuario seleccionar un archivo con una interfaz de diálogo y asigna la ruta del archivo seleccionado a la variable `archivo1`.
3. Utiliza `pd.read_excel()` de Pandas para leer el archivo Excel seleccionado y almacenarlo en el dataframe `s`.
4. Convierte el dataframe `s` en un array de NumPy y lo almacena en la variable `s2`.
5. Crea un nuevo dataframe de Pandas `s3` con la información del array `s2`, y define la columna como 'DATA'.
6. Actualiza el texto del widget de interfaz gráfica `label_nombre_archivo1` con la información del archivo cargado.

Bloque 2

Esta función de Python, `abrirArchivo2()`, realiza las siguientes operaciones:

1. Declara las variables globales `a3` y `a2`.
2. Utiliza `filedialog.askopenfilename()` para permitir al usuario seleccionar un archivo utilizando una interfaz de diálogo y asigna la ruta del archivo seleccionado a la variable `archivo2`.
3. Utiliza `pd.read_excel()` de Pandas para leer el archivo Excel seleccionado y almacenarlo en el dataframe `a`.
4. Convierte el dataframe `a` en un array de NumPy y lo almacena en la variable `a2`.
5. Crea un nuevo dataframe de Pandas `a3` utilizando la información del array `a2` y define la columna como 'DATA'.
6. Actualiza el texto del widget de interfaz gráfica `label_nombre_archivo2` con la información del archivo cargado.

Bloque 3

Esta función en Python, `leerExcel()`, realiza las siguientes operaciones:

1. Declara la variable global `archivoFinal`.

2. Inicializa el contador en 0.
3. Declara la variable global `tablaBooleana` y la inicializa con el resultado de la comparación entre `s3` y `a3`.
4. Realiza una comparación elemento a elemento entre los arrays `s2` y `a2` y almacena el resultado en el vector.
5. Calcula la cantidad de elementos iguales en el vector y la longitud total del vector.
6. Realiza una operación de cálculo para determinar el porcentaje de coincidencia entre los dos conjuntos de datos.
7. Crea un mensaje que indica el porcentaje de coincidencia calculado.
8. Crea un widget de etiqueta en la interfaz gráfica `ventana` con el mensaje generado y lo muestra en la ventana.

Bloque 4

La función `exportarArchivo()` en Python realiza lo siguiente:

1. Crea un dataframe de Pandas `df` utilizando la información almacenada en la variable `tablaBooleana`.
2. Define el nombre del archivo de salida como 'DATA_ETL_REVISADA.xlsx'.
3. Imprime el contenido de la variable `tablaBooleana`.
4. Exporta el dataframe `df` a un archivo Excel con el nombre especificado en el formato 'DATA_ETL_REVISADA.xlsx', y lo guarda en una hoja llamada 'Datos de comparación'. Se asegura de que el índice de las filas no se incluya en el archivo exportado, y utiliza el motor 'openpyxl' para realizar la exportación.

6.4.4 CONTENIDO DE INTERFAZ

Esta sección del código está creando una interfaz de usuario con varias etiquetas y botones que realizan diferentes funciones en la aplicación.
Esta sección contribuye a la creación de una interfaz de usuario interactiva que permite al usuario cargar archivos, realizar comparaciones y descargar los resultados en un formato de archivo específico.

Figura 17. Contenido Interfaz

```
Label(ventana, text="Esta inferfaz de usuario es la demostración del codigo que automatiza la revision de los datos procesados por una ETL", font
Label(ventana, text="1. Cargue el primer archivo .xlsx", font=("Arial", 12), bg="black", fg="white").pack()
Button(ventana, text="Abrir Archivo", font=("Arial", 12), bg="#009092", fg="white", relief="flat", command=abrirArchivo).pack()

label_nombre_archivo1 = tkinter.Label(ventana, text="Archivo cargado: ", font=("Arial", 12), bg="black", fg="yellow")
label_nombre_archivo1.pack()

Label(ventana, text="2. Cargue el segundo archivo .xlsx con el que va a hacer la comparación", font=("Arial", 12), bg="black", fg="white").pack()
Button(ventana, text="Abrir Archivo 2", font=("Arial", 12), bg="#009092", fg="white", relief="flat", command=abrirArchivo2).pack()

label_nombre_archivo2 = tkinter.Label(ventana, text=("Archivo cargado: "), font=("Arial", 12), bg="black", fg="yellow")
label_nombre_archivo2.pack()

Label(ventana, text="3. Haga click en el boton de Leer Archivos para ejecutar la revisión", font=("Arial", 12), bg="black", fg="white").pack()
Button(ventana, text="Leer archivos", font=("Arial", 12), bg="#009092", fg="white", relief="flat", command=leerExcel).pack()
Label(ventana, text="4. Haga click en Descargar si desea descargar un archivo .xlsx con la informacion completa", font=("Arial", 12), bg="black",
Button(ventana, text="Descargar", font=("Arial", 12), bg="#009092", fg="white", relief="flat", command=exportarArchivo).pack()

ventana.mainloop()
```

Fuente: Elaboración propia

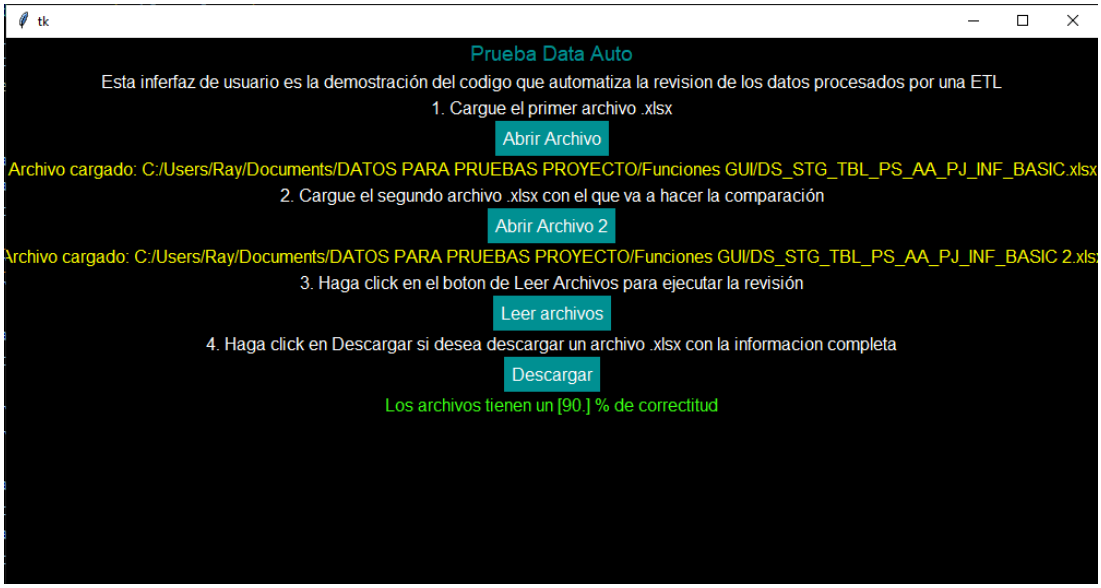
6.5 PRUEBAS

Para dar inicio a la fase de pruebas, se emplearán los datos previamente preparados con el propósito de ejecutar cada uno de los casos de prueba delineados en el diseño. En el transcurso de cada prueba, se establecerá si se satisfacen los criterios de aceptación, proporcionando además el porcentaje de cobertura y precisión correspondiente a cada caso.

6.5.1 PRUEBA DE FUNCIONALIDAD

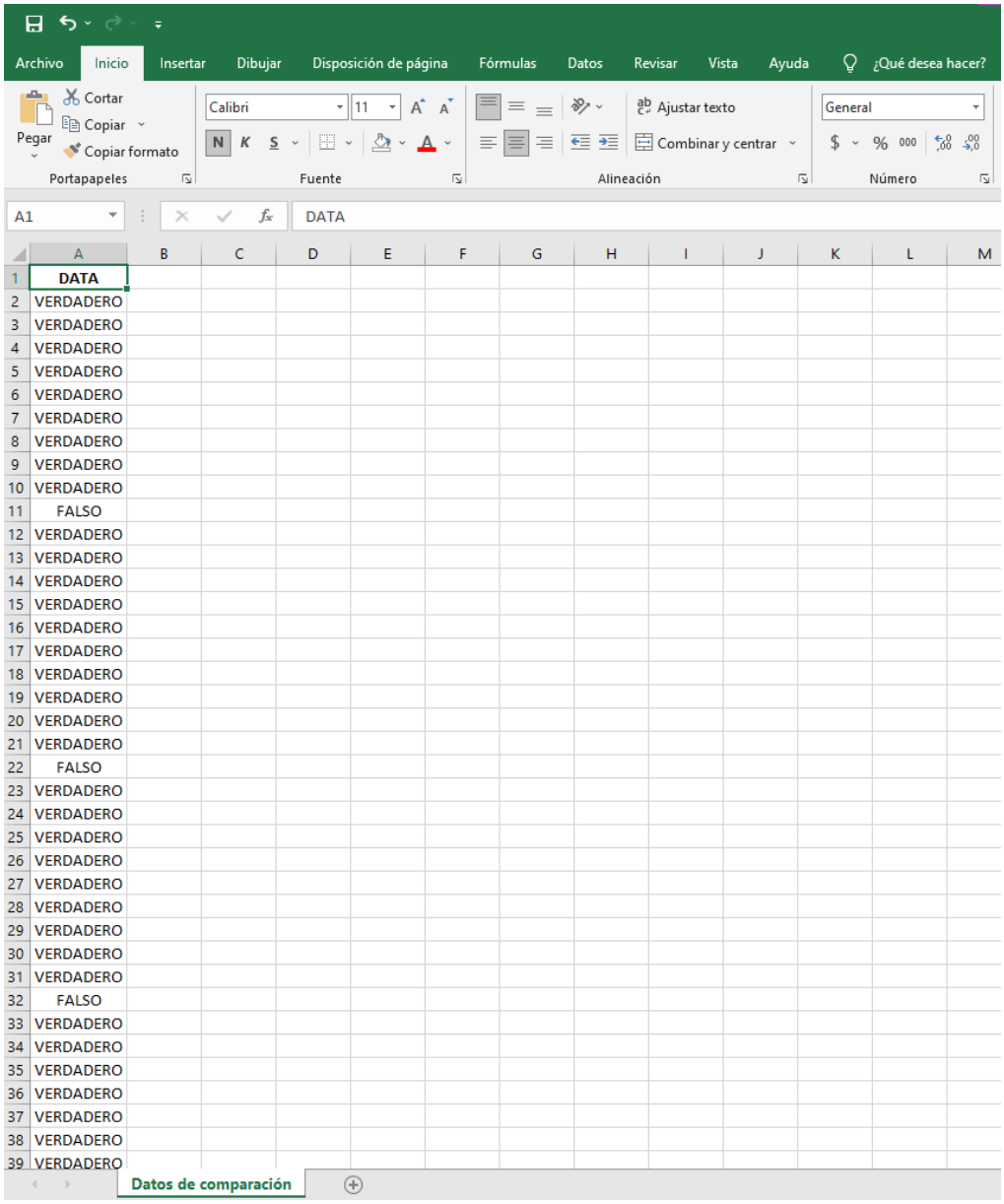
Criterios de aceptación: Cumplidos
Cobertura: 100%
Correctitud: 100%

Figura 18. Evidencia Funcionalidad Interfaz



Fuente: Elaboración propia

Figura 19. Evidencia Funcionalidad Archivo Exportado



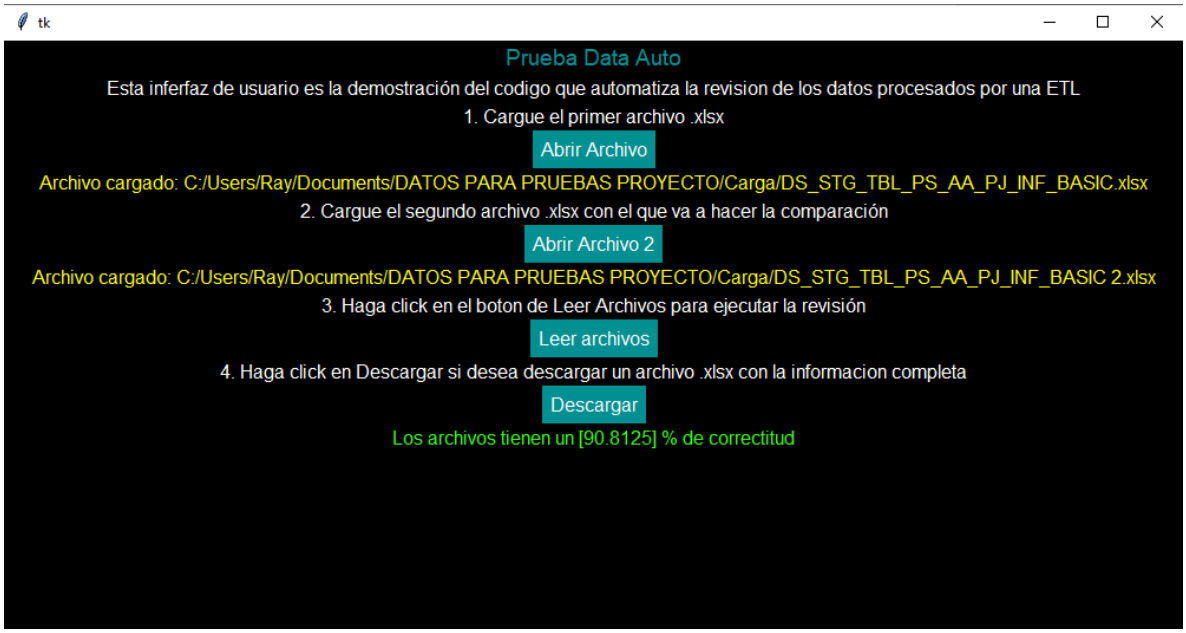
	A
1	DATA
2	VERDADERO
3	VERDADERO
4	VERDADERO
5	VERDADERO
6	VERDADERO
7	VERDADERO
8	VERDADERO
9	VERDADERO
10	VERDADERO
11	FALSO
12	VERDADERO
13	VERDADERO
14	VERDADERO
15	VERDADERO
16	VERDADERO
17	VERDADERO
18	VERDADERO
19	VERDADERO
20	VERDADERO
21	VERDADERO
22	FALSO
23	VERDADERO
24	VERDADERO
25	VERDADERO
26	VERDADERO
27	VERDADERO
28	VERDADERO
29	VERDADERO
30	VERDADERO
31	VERDADERO
32	FALSO
33	VERDADERO
34	VERDADERO
35	VERDADERO
36	VERDADERO
37	VERDADERO
38	VERDADERO
39	VERDADERO

Fuente: Elaboración propia

6.5.2 PRUEBA DE CARGA

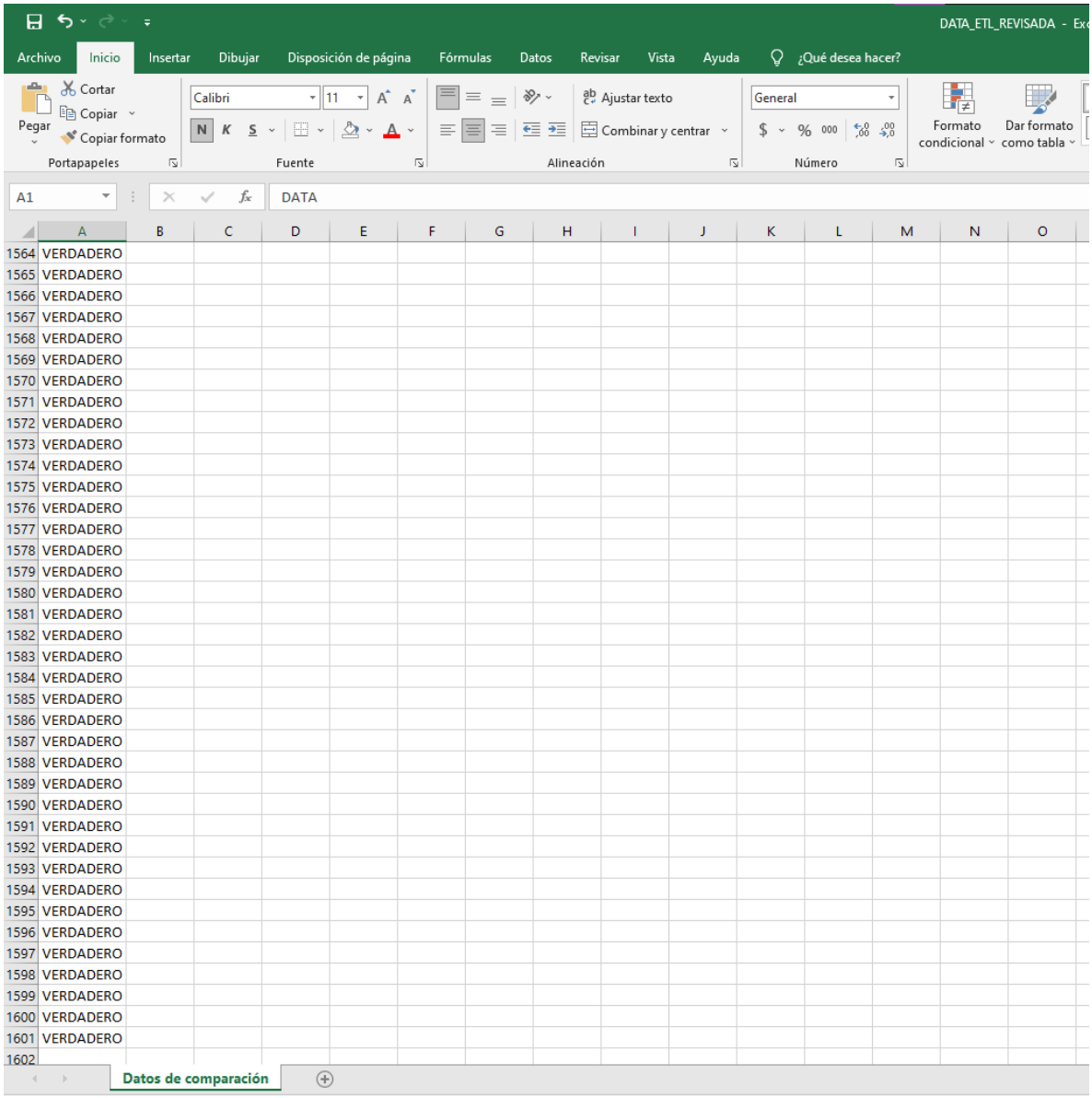
Criterios de aceptación: Cumplidos
Cobertura: 100%
Correctitud: 100%

Figura 20. Evidencia Carga Interfaz



Fuente: Elaboración propia

Figura 21. Evidencia Carga Archivo Exportado



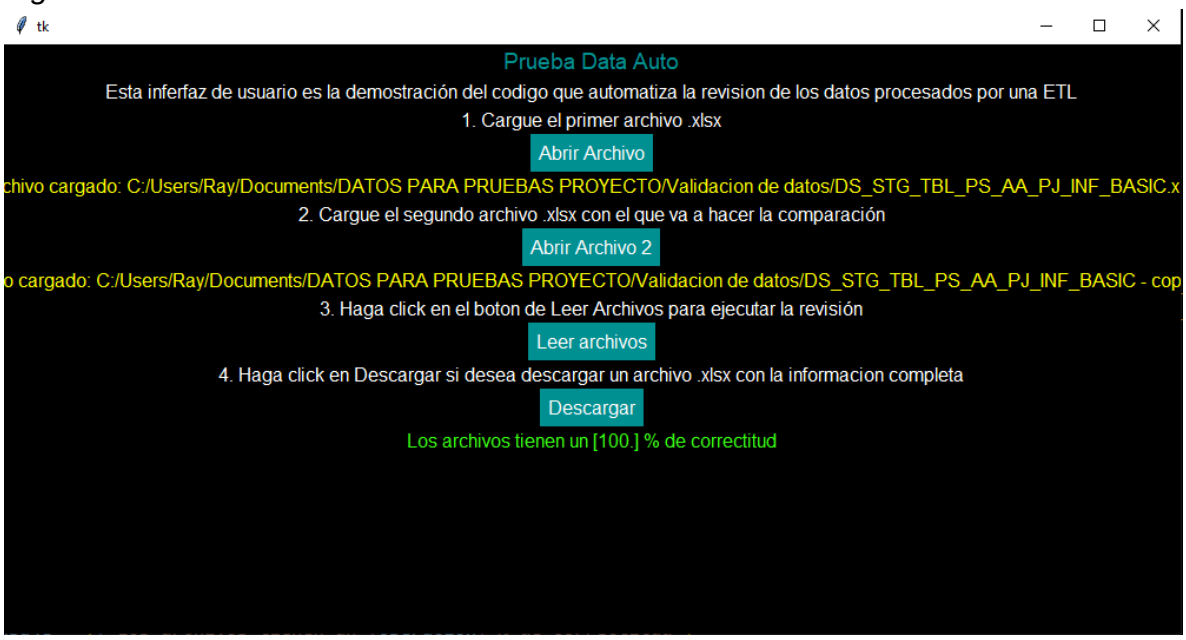
Fuente: Elaboración propia

6.5.3 PRUEBA DE VALIDACIÓN DE DATOS

Criterios de aceptación: Cumplidos
Cobertura: 100%

Correctitud: 100%

Figura 22. Evidencia Validación Datos



Fuente: Elaboración propia

6.5.4 PRUEBA DE FLUJO DE TRABAJO

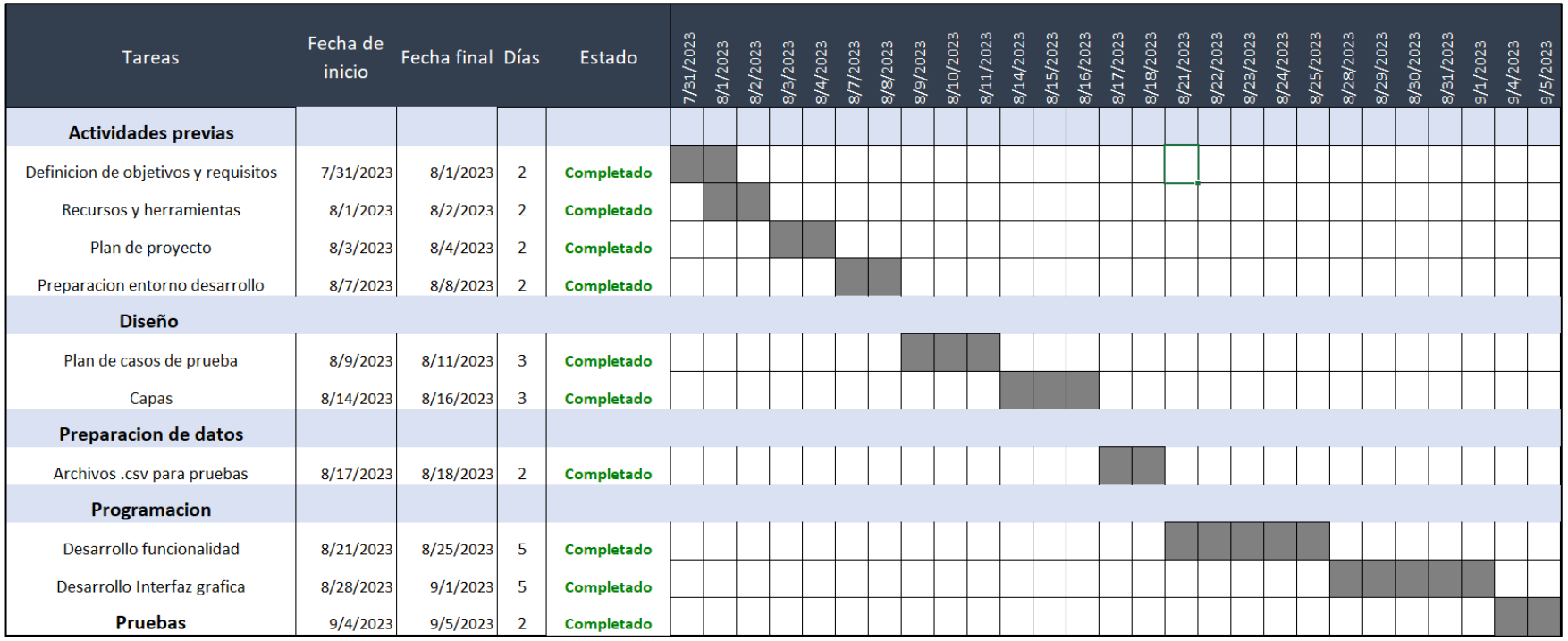
Criterios de aceptación: Cumplidos
Cobertura: 100%
Correctitud: 100%

El usuario asegura que el funcionamiento de la herramienta es correcto, es eficiente e intuitivo para el uso del equipo técnico y de pruebas de los proyectos de tecnología del Banco de Occidente.

6.6 CRONOGRAMA

Figura 23. Cronograma

Tareas	Fecha de inicio	Fecha final	Días	Estado
Actividades previas				
Definicion de objetivos y requisitos	7/31/2023	8/1/2023	2	Completado
Recursos y herramientas	8/1/2023	8/2/2023	2	Completado
Plan de proyecto	8/3/2023	8/4/2023	2	Completado
Preparacion entorno desarrollo	8/7/2023	8/8/2023	2	Completado
Diseño				
Plan de casos de prueba	8/9/2023	8/11/2023	3	Completado
Capas	8/14/2023	8/16/2023	3	Completado
Preparacion de datos				
Archivos .csv para pruebas	8/17/2023	8/18/2023	2	Completado
Programacion				
Desarrollo funcionalidad	8/21/2023	8/25/2023	5	Completado
Desarrollo Interfaz grafica	8/28/2023	9/1/2023	5	Completado
Pruebas	9/4/2023	9/5/2023	2	Completado



Fuente: Elaboración propia

7. CONCLUSIONES

- Tras un análisis exhaustivo de los requerimientos y funcionalidades en consulta con los analistas de datos y el equipo de pruebas del Banco de Occidente, se ha logrado identificar de manera precisa y completa las necesidades para la automatización del proceso de revisión de datos. La comprensión profunda de los requisitos ha permitido desarrollar una solución eficaz y precisa que satisface las demandas operativas de manera integral.

- Mediante el empleo de metodologías ágiles SCRUM, se ha logrado diseñar, implementar y desplegar una interfaz de usuario altamente intuitiva y amigable. La meticulosa investigación y selección de librerías y módulos de Python han asegurado un procesamiento eficiente de archivos .csv y la lectura de hojas de cálculo en formato .xlsx, permitiendo así un despliegue fluido y eficaz de la solución.

- La verificación exhaustiva de la precisión y eficacia de la herramienta desarrollada ha confirmado que cumple con todas las funcionalidades requeridas. La validación de su desempeño ha demostrado que la solución no solo cumple con los criterios establecidos, sino que también supera las expectativas al ofrecer un rendimiento óptimo y una precisión en línea con los estándares más exigentes del Banco de Occidente.

- El desarrollo de este proyecto deja bases de conocimiento para los estudiantes de la facultad de Ingeniería de Telecomunicaciones acerca del área de aseguramiento de calidad de software, aumentando así el espectro de conocimiento en el desarrollo de proyectos de ingeniería.

8. RECOMENDACIONES

Como entidad encargada de formular recomendaciones para futuros desarrolladores que prosigan con la evolución de este proyecto, sugerimos ampliar la carga de datos mediante la integración con bases de datos alojadas en la nube. Esta integración permitirá la extracción de información pertinente a través de consultas especializadas, garantizando así la verificación de la precisión de los datos, sin depender exclusivamente de la carga manual a través del sistema aplicativo.

Además, se aconseja incluir una mayor variedad de parámetros que puedan resultar de utilidad para los analistas de control de calidad (QA), con el fin de asegurar la exhaustividad de la evaluación de la integridad y confiabilidad de los datos en cuestión.

9. BIBLIOGRAFÍA

¿QUÉ ES ETL? - Explicación de extracción, transformación y carga (ETL) - AWS [Anónimo]. Amazon Web Services, Inc. [página web]. Disponible en Internet: <https://aws.amazon.com/es/what-is/etl/>.

SYDLE. Automatización de procesos: ¿cómo funciona? ¿Cuáles son los beneficios? Blog SYDLE [página web]. (9, abril, 2021). Disponible en Internet: <https://www.sydle.com/es/blog/automatizacion-de-procesos-6070ae4c9b901904c4349dcb>.

CALIDAD DE Datos. Cómo impulsar tu negocio con los datos. [Anónimo]. PowerData - Especialista en Gestión de Datos | MDM | Big Data | Cloud | Data Warehouse [página web]. Disponible en Internet: <https://www.powerdata.es/calidad-de-datos>.

USO DE Python para scripting y automatizaciÃ³n [Anónimo]. Microsoft Learn: Build skills that open doors in your career [página web]. Disponible en Internet: <https://learn.microsoft.com/es-es/windows/python/scripting>.

¿QUÉ ES Python? - Explicación del lenguaje Python - AWS [Anónimo]. Amazon Web Services, Inc. [página web]. Disponible en Internet: <https://aws.amazon.com/es/what-is/python/>.

TKINTER Â€” Interface de Python para Tcl/Tk [Anónimo]. Python documentation [página web]. Disponible en Internet: <https://docs.python.org/es/3/library/tkinter.html>.

LA LIBRERÍA Pandas | Aprende con Alf [Anónimo]. Aprende con Alf [página web]. Disponible en Internet: <https://aprendeconalf.es/docencia/python/manual/pandas/>.

CARDELLINO, Fernando. La guía definitiva del paquete NumPy para computación científica en Python. freeCodeCamp.org [página web]. (20, marzo, 2021). Disponible en Internet: <https://www.freecodecamp.org/espanol/news/la-guia-definitiva-del-paquete-numpy-para-computacion-cientifica-en-python/>.