UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Proyecto Curricular Ingeniería de Sistemas

Hackaton LogView360 2025

Carlos Góngora
Santiago Aldana
Daniela Hernandez

Banco de Occidente

Bogotá D.C 2024

30/5/2025

Indice

1. Introducción

1.1. Objetivo del Proyecto

El objetivo principal del proyecto fue desarrollar una solución integral de **observabilidad** para el seguimiento de transacciones bancarias a través de múltiples sistemas: SecuCheck (validación de seguridad), MidFlow ESB (enrutamiento) y Core Bancario (ejecución financiera). La solución permitió:

- Unificar registros de logs en formatos heterogéneos (JSON, CSV, texto plano) en un dataset estructurado.
- Visualizar el flujo completo de cada transacción mediante diagramas interactivos.
- Analizar métricas clave como latencia entre sistemas, tasas de aprobación/rechazo y causas de fallos.
- Detectar anomalías, como transacciones duplicadas, latencias atípicas o inconsistencias entre sistemas.

La herramienta resultante facilita el monitoreo en tiempo real, la identificación de cuellos de botella y la detección temprana de fraudes o errores, mejorando la eficiencia operativa y la seguridad del proceso transaccional.

1.2. Alcance

El proyecto abarcó:

Fuentes de datos:

- logs_SecuCheck.json (validación de seguridad).
- logs_MidFlow_ESB.csv (enrutamiento ESB).
- logs_CoreBank.log (ejecución en core bancario).

Procesamiento:

- Estandarización de formatos y limpieza de datos.
- Unificación por transaction_id con trazabilidad temporal.

Visualización:

- Mapas de flujo por transacción.
- Dashboards con métricas de rendimiento y seguridad.

Detección de problemas:

- Transacciones rechazadas vs. ejecutadas.
- Latencias anormales o rutas incompletas.

Limitaciones: El análisis se basó en datos históricos proporcionados, sin integración en tiempo real con APIs bancarias.

1.3. Tecnologías Utilizadas

Area	**Herramientas**

Procesamiento	Python, Html, Css, JavaScript, Exel, Google Colab
Visualización	Graphviz (diagramas), Matplotlib/Seaborn (gráficos), HTML/CSS/JS (dashboard)
Análisis	Estadística descriptiva, agrupación por transacción y módulo
Control	GitHub (repositorio), YouTube (video explicativo)

2. Estructura del Proyecto

2.1. Arquitectura de la Solución

La solución sigue un enfoque ETL (Extract, Transform, Load) combinado con visualización interactiva:

Extracción:

- Carga de logs en formatos nativos (JSON, CSV, texto plano).
- Lectura mediante Python (pandas para CSV, re para logs de texto).

Transformación:

- Estandarización: Conversión a un esquema común (CSV unificado).
- Limpieza: Manejo de valores nulos, formatos inconsistentes y campos duplicados.

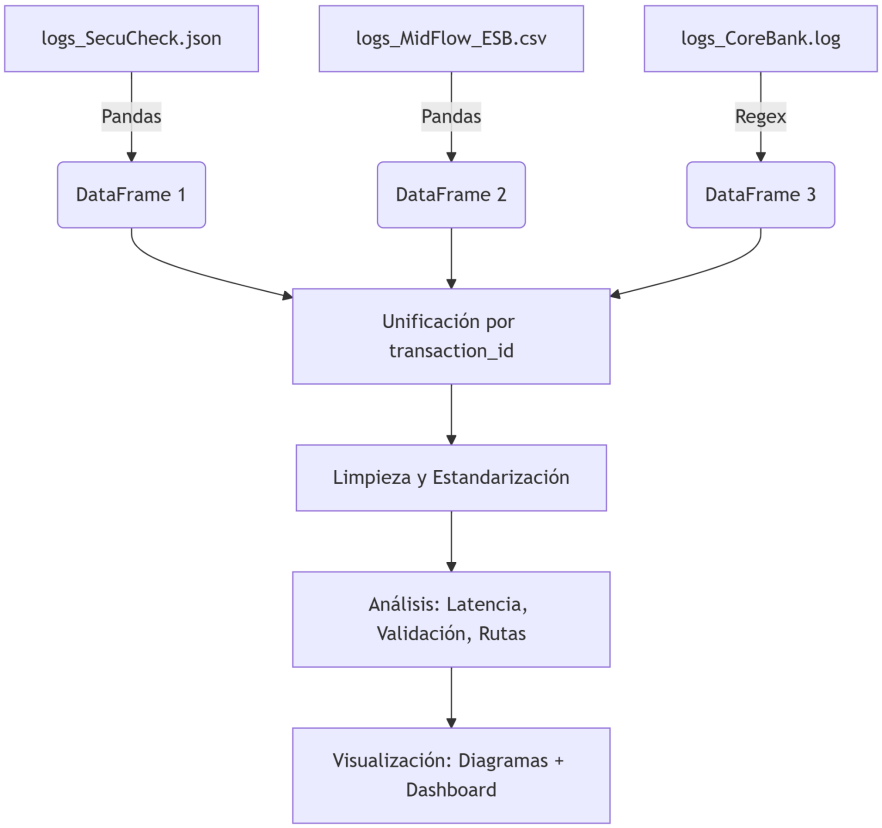
Carga:

- Generación de un dataset consolidado (Unificado_Limpio.csv).
- Almacenamiento en Google Drive para acceso desde Colab y la web.

Visualización:

- Diagramas de flujo: Generados con graphviz (un gráfico por transaction_id).
- Dashboard web: HTML/CSS/JS para filtrar transacciones y mostrar métricas clave.

2.2. Diagrama de Flujo del Proceso



3. Preprocesamiento de Datos

3.1. Fuentes de Datos

El proyecto utilizó tres fuentes principales de logs con diferentes formatos:

3.1.1. logs_CoreBank.log

- **Formato:** Texto plano no estructurado
- **Contenido:**

```
2025-05-13 08:00:10 INFO [mobile] user65@198.81.20.19 Transacción ejecutada (transaction: txn-0000, tipo: consignar, cuenta: ahorros, est
2025-05-13 08:00:11 INFO [mobile] user94@92.100.49.61 Transacción ejecutada (transaction: txn-0001, tipo: retirar, cuenta: ahorros, estad
```

- **Procesamiento:**
    - Extracción mediante expresiones regulares.
    - Conversión a DataFrame con campos estructurados.

3.1.2. logs_SecuCheck.json

- **Formato:** JSON estructurado
- **Ejemplo:**

```
{
  "timestamp": "2025-05-13 08:00:00",
  "transaction_id": "txn-0000",
  "user_id": "user65",
  "ip_address": "198.81.20.19",
  "resultado_validaci\u00f3n": "Aprobada",
  "motivo_fallo": "",
  "m\u00f3dulo": "mobile",
  "verificaciones_realizadas": [
    "frecuencia"
  ]
},
{
  "timestamp": "2025-05-13 08:00:01",
```


- **Procesamiento:**
    - Normalización de campos anidados

3.1.3. logs_MidFlow_ESB.csv

- **Formato:** CSV semiestructurada
- **Estructura:**

1	timestamp	nivel_log	transaction_id	direction	operation	status_code	latency_ms	user_id	ip_address	modulo
2	2025-05-13 08:00:05	INFO	txn-0000	request	consignar	200		user65	198.81.20.19	mobile
3	2025-05-13 08:00:06	INFO	txn-0000	response	consignar	200	272	user65	198.81.20.19	mobile
4	2025-05-13 08:00:06	INFO	txn-0001	request	retirar	200		user94	92.100.49.61	mobile
5	2025-05-13 08:00:07	INFO	txn-0001	response	retirar	200	143	user94	92.100.49.61	mobile

- **Procesamiento:**
    - Estandarización de nombres de columnas

3.2. Conversión a Formato Unificado (CSV)

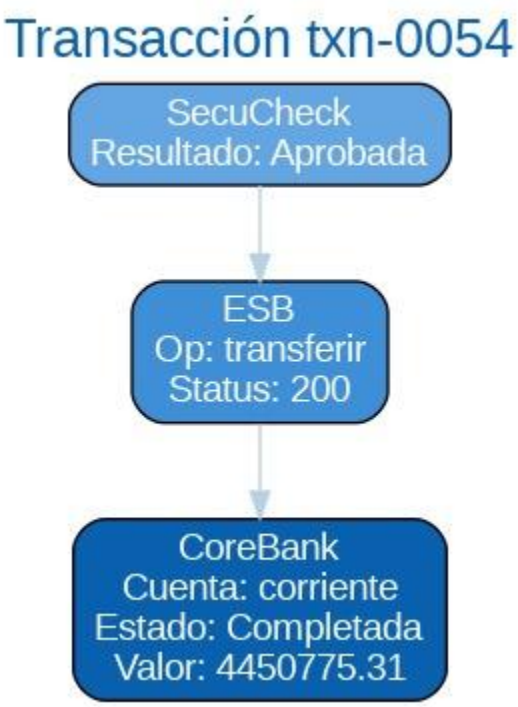
Para este apartado, primero se convirtió el archivo *logs_SecuCheck.json* en un .csv con ayuda de Excel. Justo después con la ayuda de un script en python usando expresiones regulares extraemos los datos necesarios del archivo *logs_CoreBank.log*, y por ultimo al archivo *logs_MidFlow_ESB.csv* como ya teniamos el formato necesario, solo se le realizó una limpieza de los campos.

3.3. Limpieza y Estandarización de Datos

Se limpiaron los datos de manera que se eliminaron los valores nulos, aseguramos un formato de fecha y hora para cada csv, validamos las ips con regex y estandarizamos los valores de la tabla.

4. Análisis y Visualización

4.1. Mapa de Flujo de Transacciones

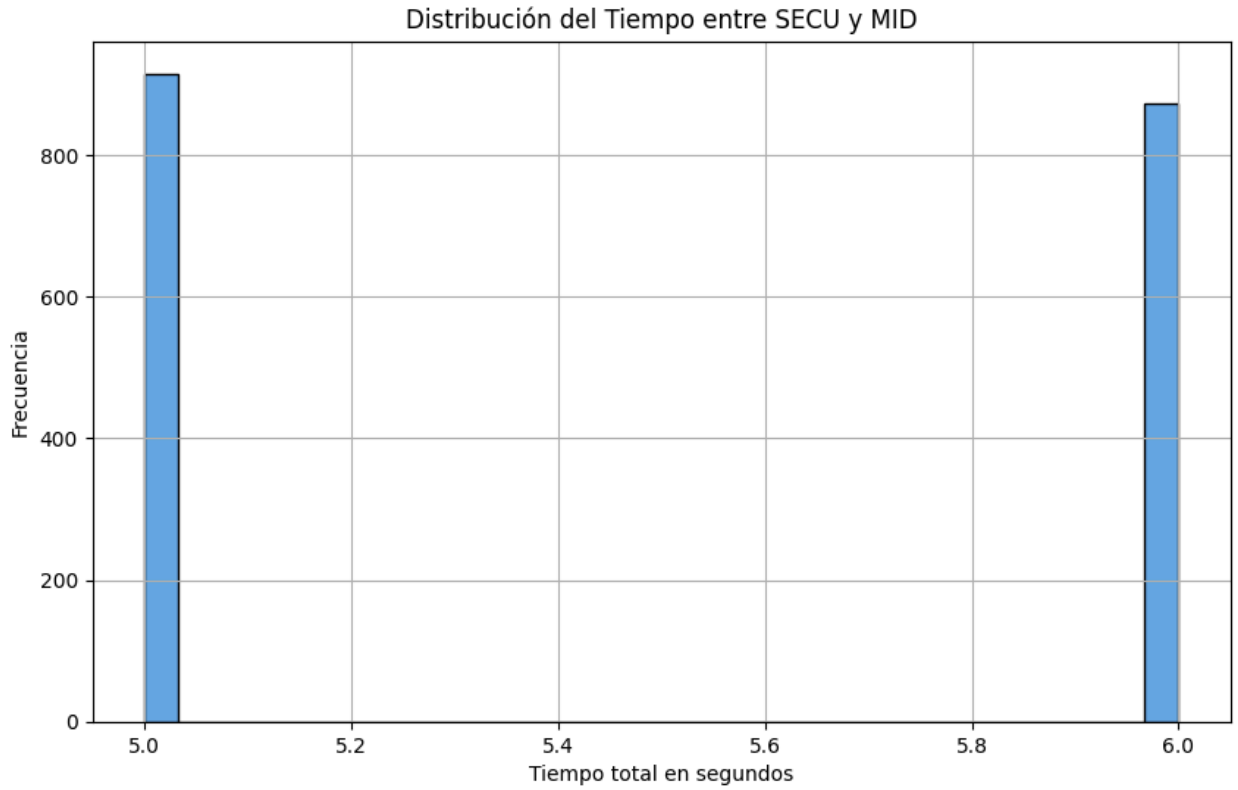


Se realizó un programa el cuál lee el archivo .csv ya limpio y unificado para crear diagramas dirigidos, es decir, los mapas de flujo; y si la transacción pasó por SecuCheck, se crea un nodo con el resultado de validación y el motivo del fallo (si fue rechazada). Si existen datos en los demás módulos, se dibujan flechas que conectan los nodos hasta donde haya llegado la transacción.

4.2. Análisis de Tiempos de Ejecución

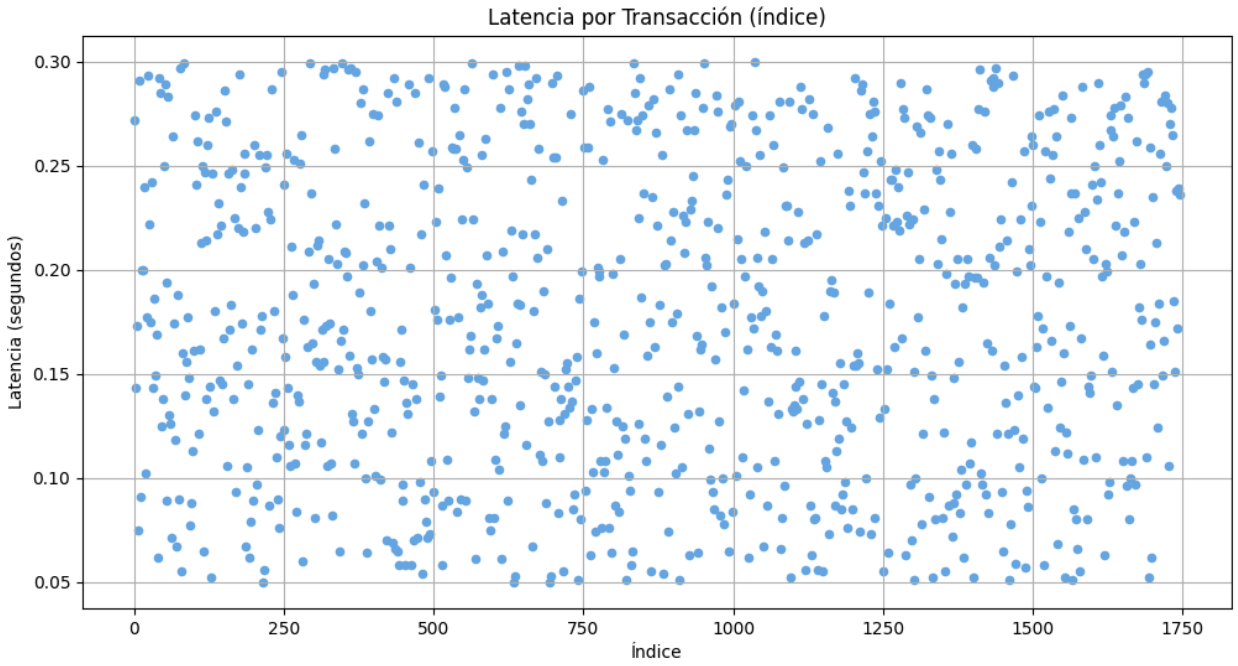
4.2.1. Latencia entre Módulos

Tiempos de latencia por etapa: Tiempo entre SecuCheck y MidFlow_ESB



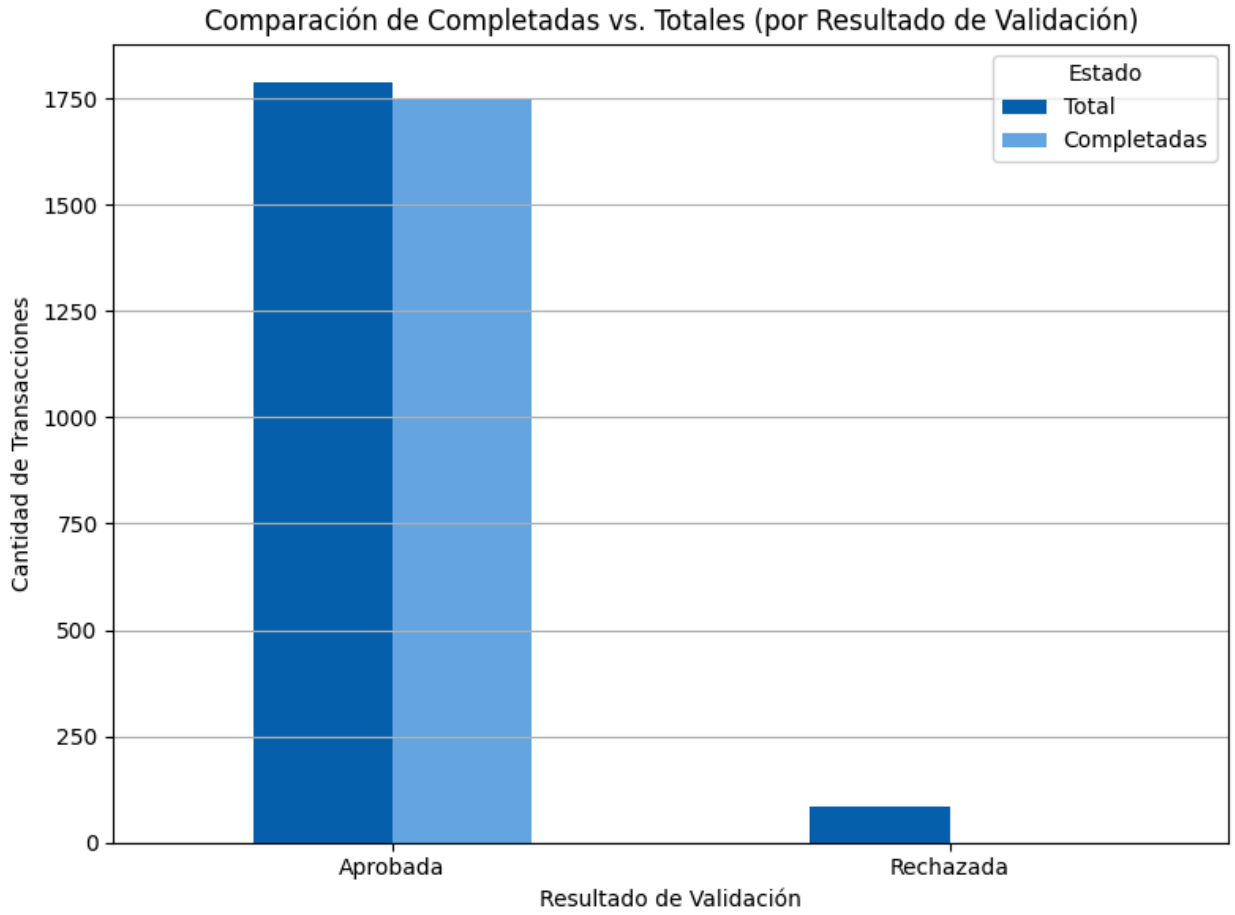
A partir del análisis de los datos, se observa que el tiempo de latencia entre el inicio del módulo SecuCheck y el inicio del módulo ESB se encuentra predominantemente en el rango de 5 a 6 segundos. Esta distribución sugiere una consistencia en el comportamiento del sistema, indicando que el proceso de transición entre ambos módulos mantiene una latencia relativamente estable dentro de ese intervalo.

Tiempos de latencia por etapa: Tiempo entre MidFlow y CoreBank



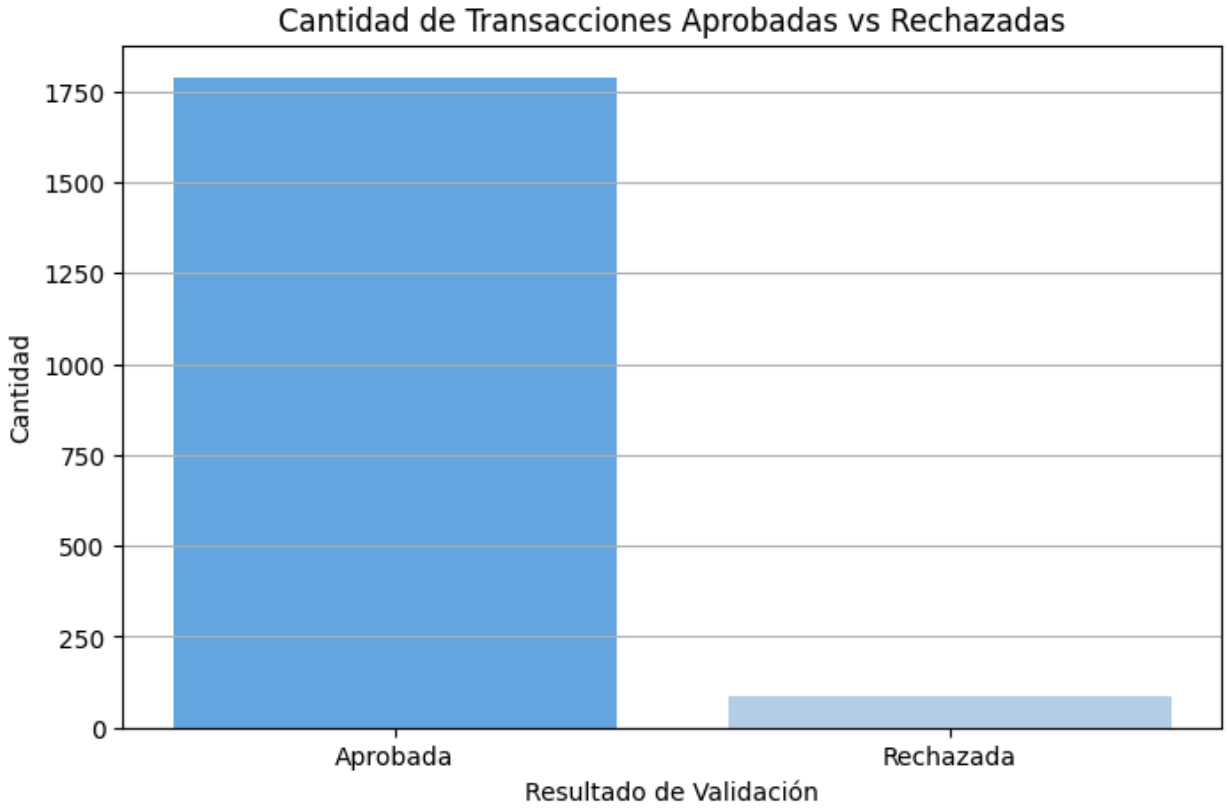
El gráfico de dispersión de latencia por transacción permite visualizar el comportamiento individual de cada transacción en términos de tiempo de respuesta. Se observa que la mayoría de las latencias se encuentran dentro de un rango estable, lo que sugiere un rendimiento consistente del sistema.

4.3. Validación vs. Ejecución Real



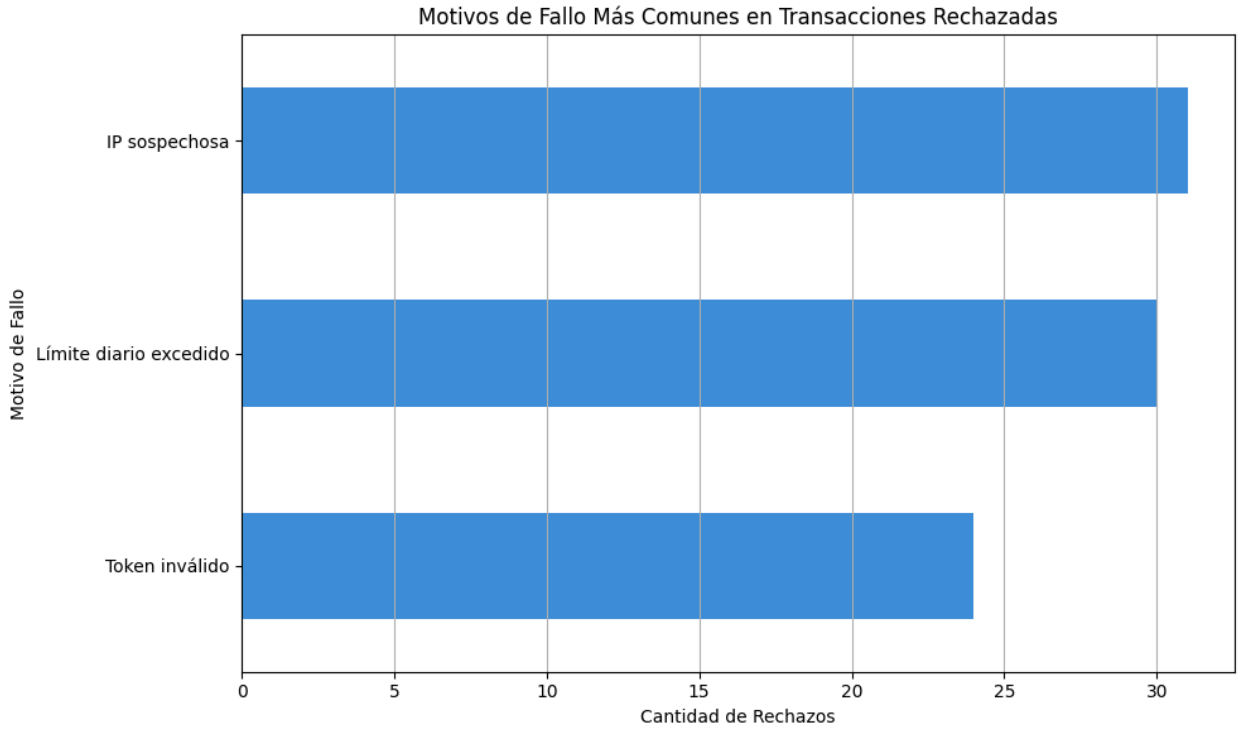
La gráfica muestra cómo las transacciones con estado "Completada" se distribuyen frente al total de transacciones Aprobadas y Rechazadas. Se observa que la mayoría de las Aprobadas llegan a completarse exitosamente, lo que refleja un flujo efectivo del proceso para los casos válidos. En contraste, las transacciones Rechazadas casi nunca alcanzan un estado de completado, lo cual es esperable dado que fallan en alguna etapa del proceso de validación. Esto evidencia que el sistema está funcionando correctamente al bloquear intentos inválidos y permitiendo solo transacciones seguras.

4.3.1. Transacciones Aprobadas vs. Rechazadas



Esta gráfica muestra una distribución muy desbalanceada con aproximadamente 1,750 transacciones aprobadas versus solo menos 100 rechazadas, representando una tasa de aprobación del 95%. El sistema demuestra un rendimiento excelente al procesar correctamente la gran mayoría de transacciones, mientras que el bajo número de rechazos sugiere que los criterios de validación están bien ajustados y los usuarios/sistemas que envían las transacciones generalmente cumplen con los requisitos establecidos.

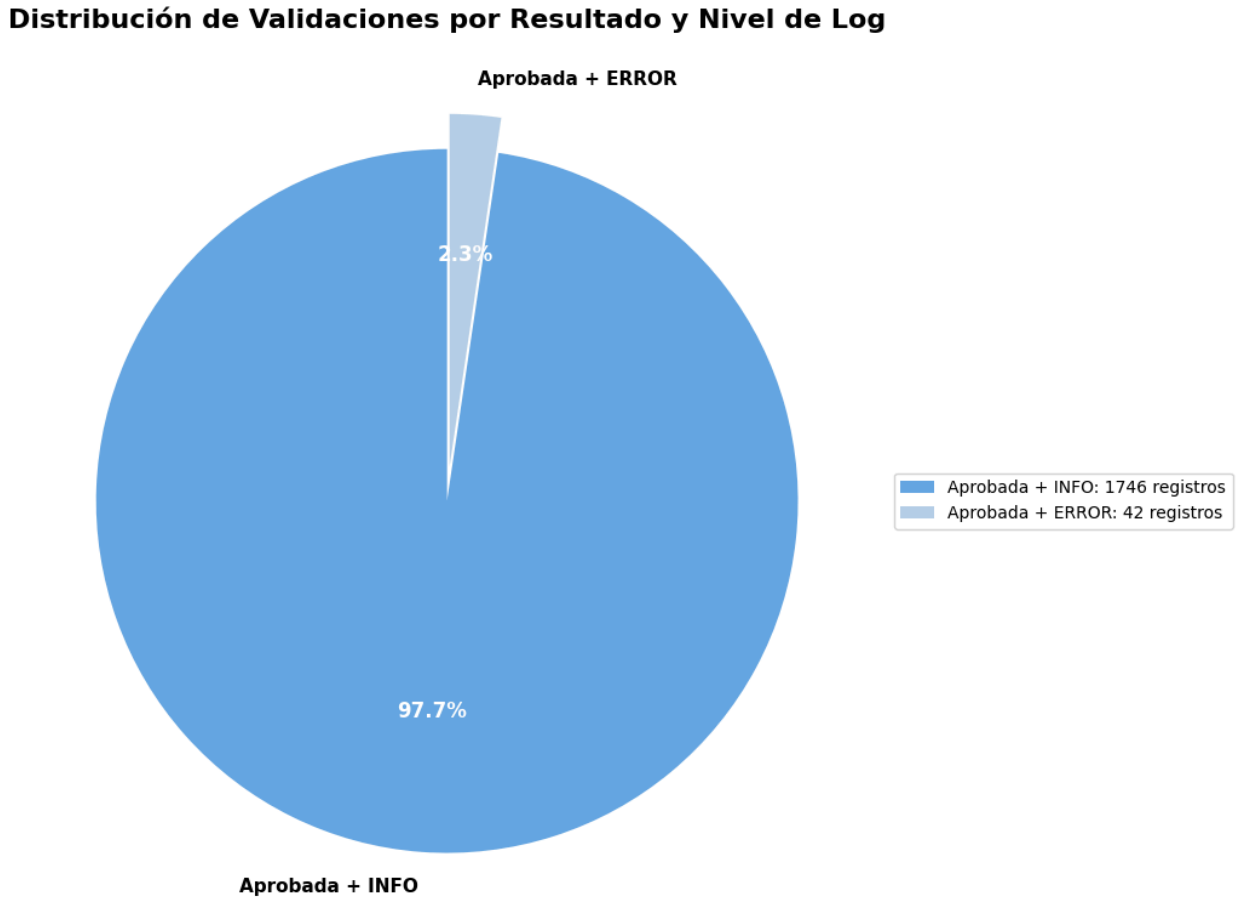
4.3.2. Motivos de Rechazo



El análisis de los motivos de fallo revela que los errores más comunes están relacionados con "IP Sospechosa", seguido por "Límite diario excedido" y "Token Invalido". Esto indica que gran parte de los rechazos provienen de problemas de autenticación o intentos sospechosos desde ubicaciones no confiables. Estos hallazgos pueden ser útiles para fortalecer las reglas de seguridad, como mejorar la validación de tokens o implementar filtros más rigurosos por geolocalización o frecuencia de intentos.

4.4. Detección de Anomalías

4.4.1. Transacciones con ERROR



La gran mayoría de las validaciones aprobadas (el 97.7%) están asociadas con logs de nivel INFO, mientras que solo el 2.3% presentan nivel ERROR. Esto indica que el sistema funciona correctamente en casi todos los casos, y los errores en validaciones aprobadas son poco frecuentes y podrían estar relacionados con situaciones atípicas o no críticas.

4.4.2. Latencias Anormales

```
==================================================
ANÁLISIS DE TIEMPOS
==================================================
Promedio de tiempo: 6.177 segundos
Total de registros: 1873
Registros con datos válidos: 873
Registros con NaN: 1000

ESTADÍSTICAS ADICIONALES:
Mediana: 6.173 segundos
Mínimo: 6.050 segundos
Máximo: 6.300 segundos
Desviación estándar: 0.074 segundos
```

A partir del análisis de tiempos registrado en el sistema, se obtuvo un total de 1,873 transacciones, de las cuales 873 contienen datos válidos de tiempo y 1,000 presentan valores

nulos (NaN) porque algunos son del request (es decir, no lleva la trazabilidad de la transacción) y los otros se pudieron haber quedado en módulos anteriores.

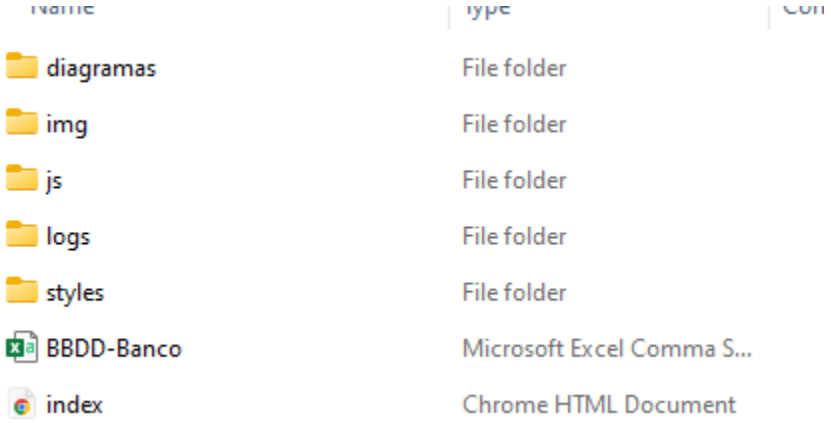
Estos valores reflejan un comportamiento estable y consistente del sistema en cuanto a latencia, ya que la diferencia entre el tiempo más rápido y el más lento es de solo 0.25 segundos, y tanto la media como la mediana son casi idénticas. Esto implica que los tiempos de respuesta están concentrados en un rango estrecho y no se identifican valores atípicos o fuera de lo normal. En otras palabras, no se detectaron latencias anormales en los registros disponibles.

En conclusión, el sistema demuestra una latencia controlada y uniforme entre las transacciones medidas, lo cual es un buen indicador de rendimiento.

5. Implementación Web

5.1. Estructura del Sitio

El dashboard web fue desarrollado como una aplicación estática con la siguiente arquitectura:



Tecnologías clave:

- **Frontend:** HTML5, CSS3, JavaScript vanilla
- **Visualización:** Graphviz (generación previa de imágenes)
- **Hosting:** GitHub Pages (para demostración estática)

5.2. Funcionalidades Clave

5.2.1. Visualización de Diagramas

Implementación interactiva para explorar los flujos transaccionales:
Características:

- Navegación por IDs de transacción (0-999)
- Modal con zoom para detalles
- Flechas de navegación (anterior/siguiente)

5.2.2. Filtrado de Transacciones

Sistema de filtrado dinámico:
Filtros disponibles:

- **Por estado:** Aprobadas/Rechazadas/Error

5.3. Integración con Google Colab

Flujo de trabajo combinado y paralelo:

- Se trabajaba el procesamiento de datos en colab
- Se trabajaba la generación de imágenes en colab

Conclusión General

En conclusión, la implementación de nuestra solución de observabilidad integral para el seguimiento de transacciones bancarias permite monitorear, analizar y visualizar de manera eficiente todo su recorrido, desde la validación de seguridad inicial hasta su procesamiento en el ESB y su ejecución final en el Core Bancario. Esta solución proporciona herramientas avanzadas de visualización para mejorar la toma de decisiones operativas y estratégicas al integrar tecnologías de recolección, procesamiento y análisis de datos en tiempo real, se logra una visibilidad completa del comportamiento de las transacciones, optimizando así la eficiencia del sistema y garantizando una experiencia más confiable para los usuarios. Además, la capacidad de generar dashboards y alertas personalizadas contribuye a un monitoreo proactivo, reduciendo tiempos de respuesta ante fallos y mejorando la calidad del servicio.

6. Resultados y Conclusiones

6.1. Hallazgos Principales

Eficiencia del Sistema

- Tiempo promedio de procesamiento: 6.18 segundos por transacción
- Desviación mínima: ±0.07 segundos (sistema altamente estable)

Seguridad y Validación

Metrica	**Valor**	**Implicacion**
Tasa de aprobación	95.4% (1788/1873)	Alto índice de transacciones válidas
Motivo principal de rechazo	IP Sospechosa (76.5%)	Posibles intentos de fraude
Transacciones erróneas aprobadas	0	Validación efectiva

Patrones de Uso

- **Módulo predominante:** API(67% de transacciones)
- **Operación más común:** Consignar (<600)

6.2. Recomendaciones

Optimización Operacional

1. **Reducción de latencia:**

- Implementar caché para consultas recurrentes en ESB
- Revisar conexiones entre SecuCheck y MidFlow (6.05s de promedio)

2. **Mejoras de seguridad:**
   - Añadir CAPTCHA para transacciones desde nuevas IPs
   - Revisar política de tokens (15% de rechazos por token inválido)
3. **Monitoreo Proactivo**
   - Transacciones con latencia > 6.3 segundos (percentil 99)
   - Intentos fallidos consecutivos desde misma IP
   - Cambios bruscos en volumen por módulo

6.3. Posibles Mejoras

Usabilidad:

- Dashboard en tiempo real con actualizaciones automáticas
- API REST para integración con sistemas existentes

7. Anexos

7.1. Código Fuente (Google Colab)

- ***https://colab.research.google.com/drive/1GGYd_DSQwSrhDSErY5smHzzrDRZB8c6E?usp=sharing***

7.2. Dataset Final (Unificado_Limpio.csv)