Universidad de los Andes

Facultad de Ingeniería

Desarrollo de Soluciones Cloud

Plan de pruebas de carga

Grupo 1

Juan Sebastian Colmenares - 202312351

William Fernando Alarcón - 202423127

Jhon Mario Forero – 20123466

Contenido

Plan de Pruebas de Carga

1. Descripción de la Aplicación

La aplicación consta de un backend en FastAPI y un frontend en Angular. El backend gestiona la carga, análisis y resumen de documentos con PostgreSQL y Redis. Se usa el modelo LLM web de Google (Gemini Flash 2.0) para el análisis de documentos. El frontend permite a los usuarios interactuar con la plataforma de análisis.

2. Entorno de Pruebas

Las pruebas de proyecto "LLM Analyze Documents" se ejecutarán en una plataforma implementada sobre Google Cloud Platform, la cual tiene los siguientes recursos:

- **Web Server** (Compute Engine - 4 vCPU, 4 GiB RAM, 20 GiB disco): Encargado de exponer la interfaz web y recibir peticiones REST API de los usuarios.
- **Worker Server** (Compute Engine - 4 vCPU, 4 GiB RAM, 50 GiB disco): Realiza procesamiento intensivo, embedding y chunking de los documentos.
- **File Server (NFS)** (Compute Engine - 2 vCPU, 2 GiB RAM, 20 GiB disco): Proporciona almacenamiento compartido para los documentos subidos.
- **DB Server** (PostgreSQL - 2 vCPU, 2 GiB RAM, 20 GiB disco): Guarda metadatos, logs y resultados del análisis de documentos.

Gemini Flash 2.0 API: LLM externo que recibe los chunks de texto para su análisis

3. Criterios de Aceptación

- Tiempo de respuesta óptimo para cargas estándar.
- Capacidad de procesamiento adecuada para múltiples solicitudes concurrentes.
- Uso eficiente de los recursos de hardware.

4. Escenarios de Prueba

Se han definido escenarios representativos que cubren rutas críticas de usuario tanto en la capa web como en el procesamiento por lotes. La variedad y realismo de los escenarios garantizan una evaluación precisa del rendimiento del sistema.

Capa Web:

- **Autenticación:**
    - Registro de usuario (/api/v1/auth/register)
    - Inicio de sesión (/api/v1/auth/login)
- **Gestión de Documentos:**
    - Carga de documentos (/api/v1/documents/upload)
    - Listado de documentos (/api/v1/documents/listDocuments)

- **Análisis de Documentos:**
    - Resumen de documentos (/api/v1/analysis/summarize/{document_id})
    - Preguntas sobre documentos (/api/v1/analysis/ask/{document_id})
- **Conversaciones:**
    - Listado de conversaciones (/api/v1/conversations/list)
    - Detalle de conversación (/api/v1/conversations/byId/{conversation_id})
- **Monitoreo del servicio:**
    - Verificación del estado (/health)

Procesamiento por Lotes:

- Simulación de carga masiva en el backend para evaluar el impacto del procesamiento en lotes sobre el rendimiento del sistema.
- Análisis del tiempo de ejecución para múltiples peticiones simultáneas de resumen y preguntas sobre documentos.

5. Parámetros de Configuración

Las pruebas utilizarán Apache JMeter con la siguiente configuración:

- **Número de usuarios simulados:** 50-100
- **Duración de las pruebas:** 10 minutos por escenario
- **Monitoreo del uso de CPU y memoria**

6. Justificación de la Herramienta Seleccionada

Se ha elegido **Apache JMeter** por su adecuación al contexto del ejercicio y su facilidad de configuración para pruebas de carga y estrés. Entre sus ventajas destacan:

- Soporte para pruebas de múltiples protocolos (HTTP, WebSockets, JDBC, etc.).
- Facilidad para simular cargas concurrentes y obtener métricas detalladas.
- Integración con herramientas de monitoreo y generación de reportes.
- Flexibilidad para realizar pruebas en entornos locales y en la nube.

7. Definición de los Recursos de la Instancia de Pruebas

Para garantizar resultados representativos, se han definido los siguientes recursos:

- **CPU:** 4 núcleos para manejar múltiples hilos simultáneamente.
- **Memoria RAM:** 4 para soportar grandes volúmenes de peticiones sin degradación.
- **Disco SSD:** 512GB para optimizar tiempos de acceso a datos y almacenamiento temporal.

Estos recursos han sido seleccionados considerando la carga esperada y las pruebas de estrés que se realizarán en JMeter.

8. Identificación y Explicación de Métricas Clave

Para evaluar el rendimiento del sistema, se analizarán las siguientes métricas clave:

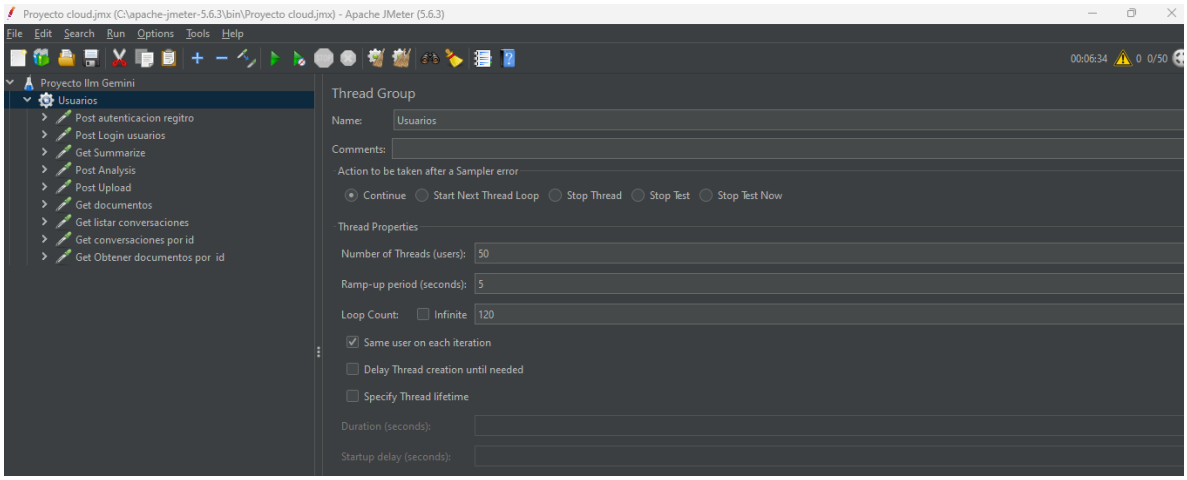
- **Throughput:** Número de peticiones procesadas por segundo. Indicador clave de la capacidad del sistema.
- **Response Time:** Tiempo promedio de respuesta de cada endpoint bajo diferentes niveles de carga.
- **Utilización de Recursos:** Consumo de CPU, memoria y disco durante las pruebas.
- **SLIs relevantes:** Se definirán umbrales aceptables de tiempo de respuesta y tasa de éxito de solicitudes para evaluar el cumplimiento de los acuerdos de nivel de servicio (SLAs).

Con este enfoque, se asegura una evaluación completa del rendimiento del sistema bajo distintos niveles de carga y escenarios de uso.

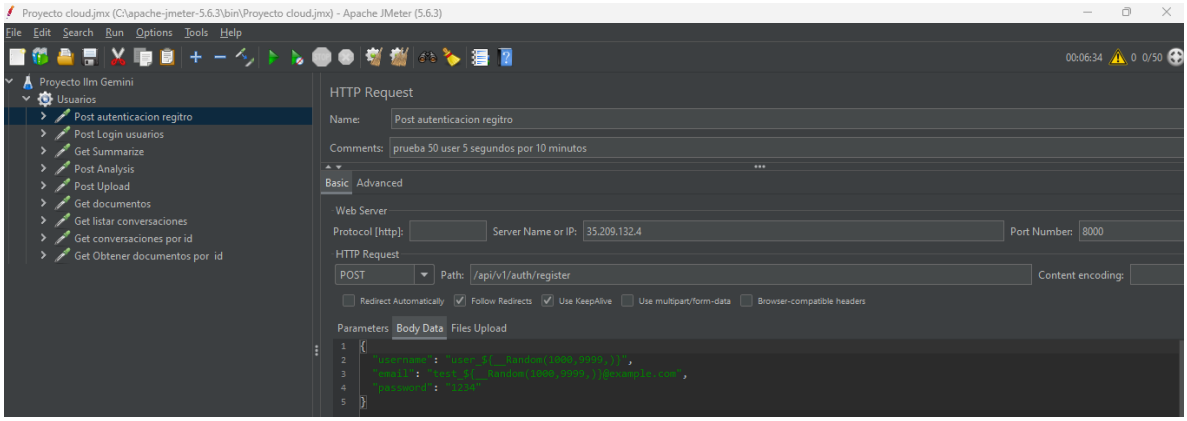
Resultados pruebas de estrés

Criterios de las pruebas de cada uno de los servicios:

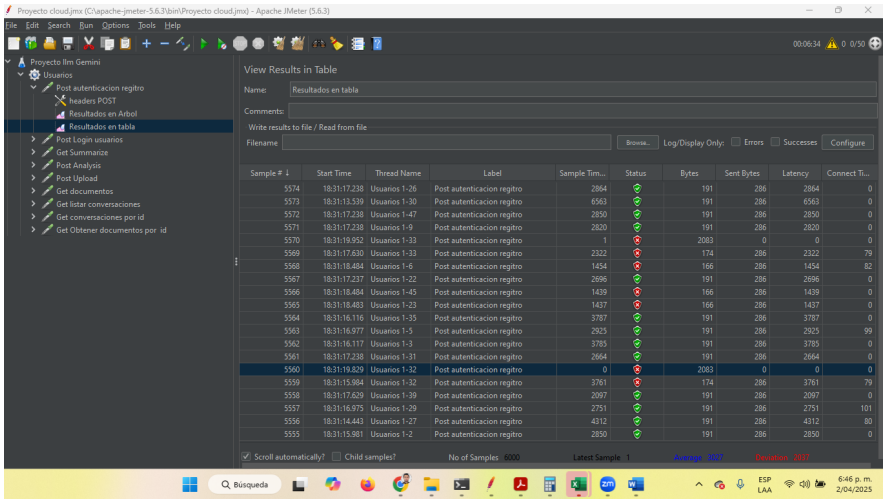
50 peticiones cada 5 segundos durante 120 ciclos para un total de 6000 peticiones en 10 minutos.



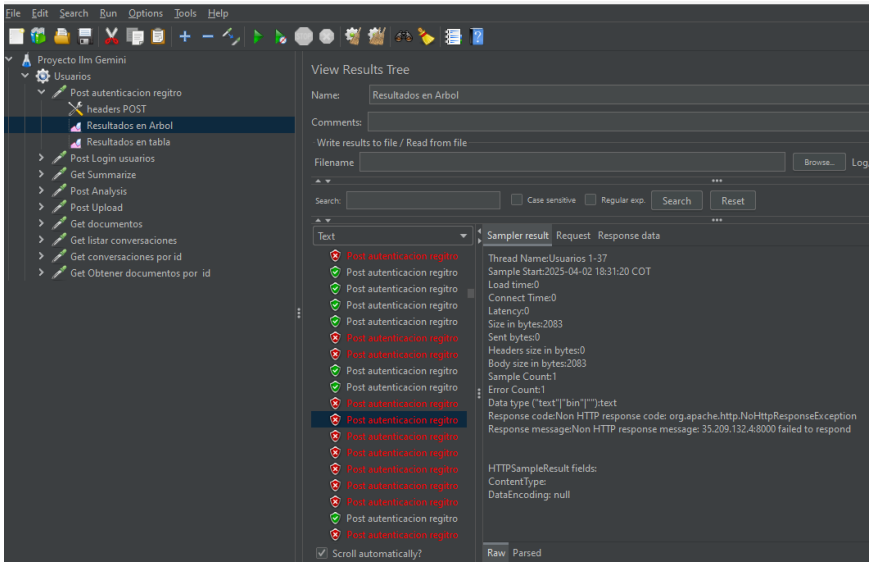
- Registro de usuario:
  http://35.209.132.4:8000/api/v1/auth/register
  Prueba jmeter registro de usuarios, request



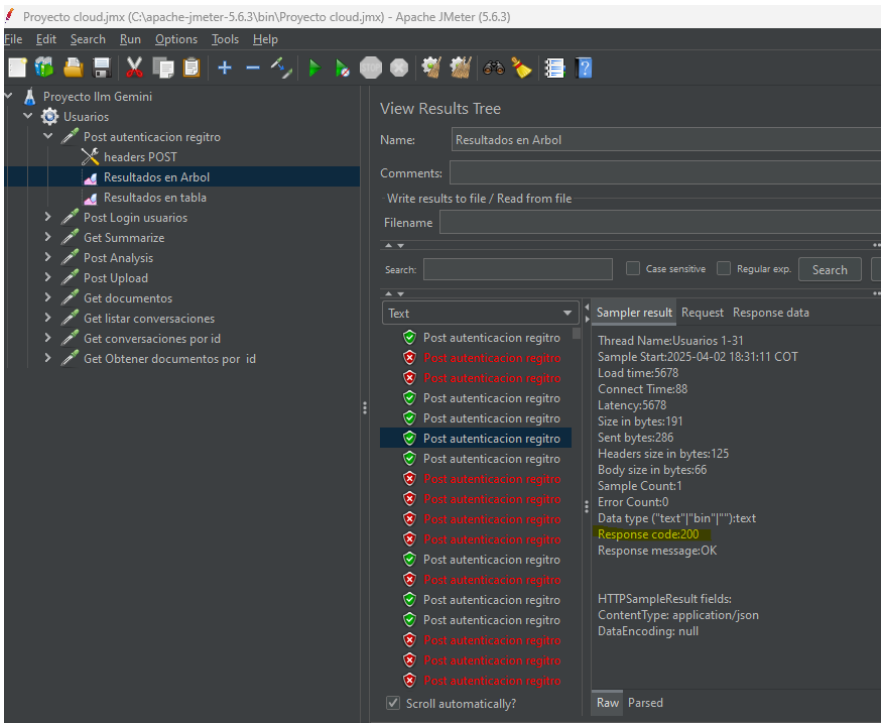
Resultados:
En la siguiente imagen se puede evidenciar que en 10 minutos hubieron 3027 registros exitosos, 2037 request rechazados por el servidor y 936 que no fueron fallas si no que los usuarios ya existían.



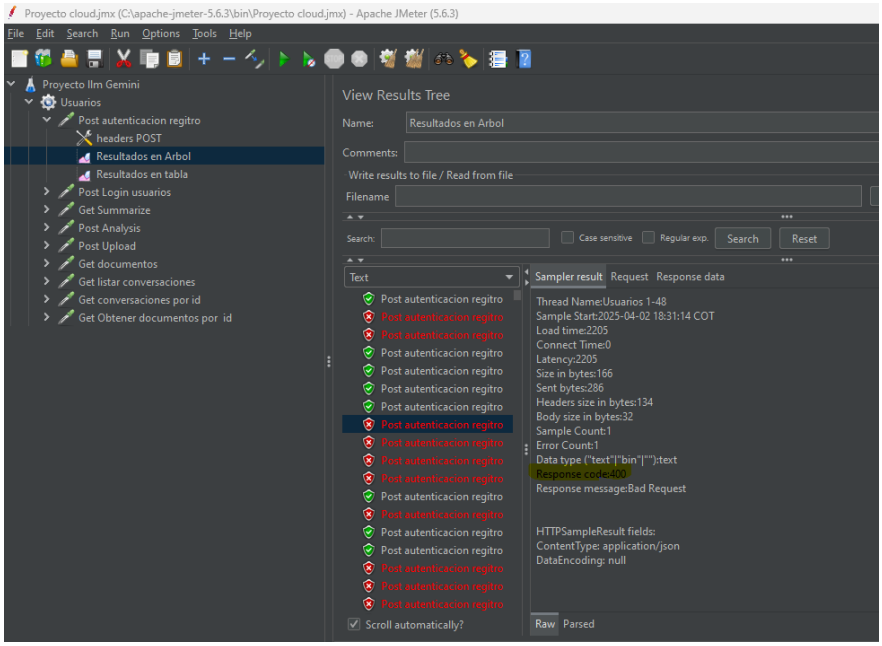
Respuesta servidor El error **org.apache.http.NoHttpResponseException** con el mensaje **"35.209.132.4:8000 failed to respond"** significa que el servidor no respondió a la solicitud de JMeter:



Respuesta exitosa como se muestra:



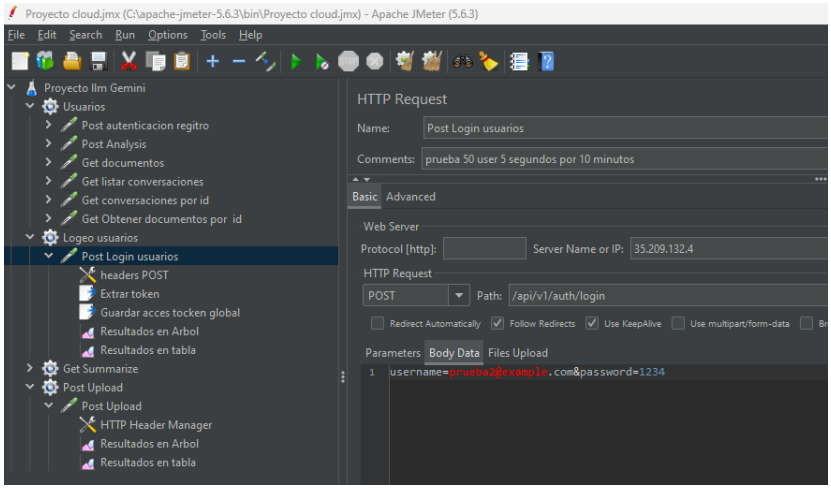
Respuesta de error, el servidor El error **400 - Bad Request** indica que el servidor rechazó la solicitud porque está malformada o tiene datos incorrectos.:



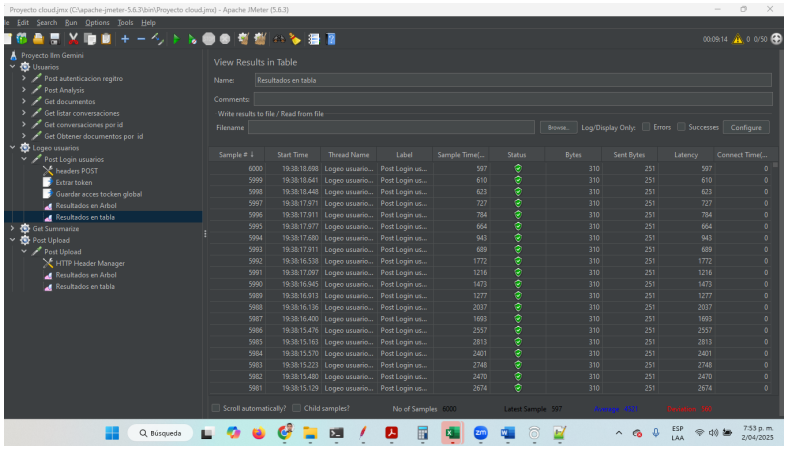
- Login de usuarios:

http://35.209.132.4:8000/api/v1/auth/login

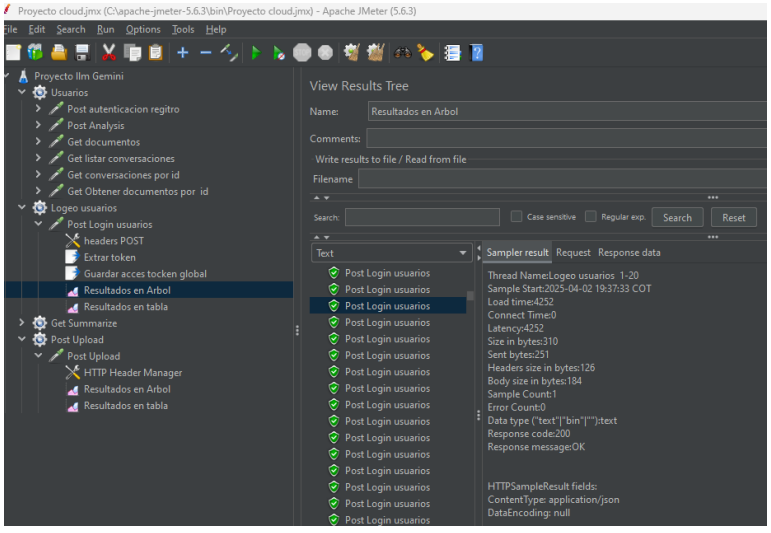
Request que se ejecuto:



Se puede evidenciar que las 6000 peticiones fueron exitosas donde la petición no genero ningún error genero error , sin embargo hay una desviación de 500 ms lo cual indica que algunas peticiones tardan más en responder.



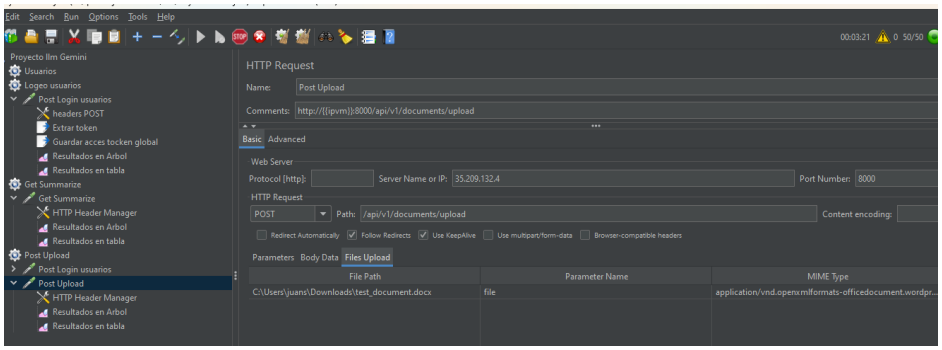
Todos los response fueron exitosos sin fallas



- Cargar archivo:

http://35.209.132.4:8000/api/v1/documents/upload

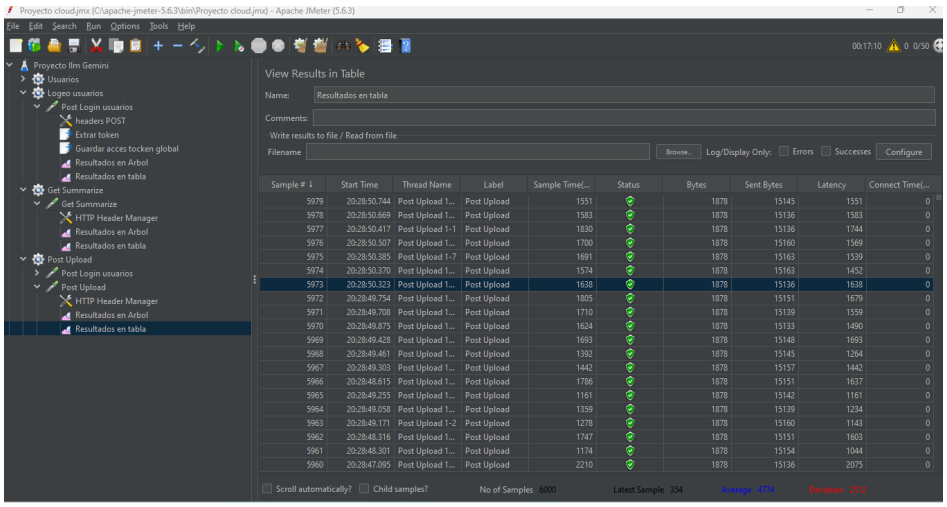
Request que se ejecutó en este se está realizando la carga de archivos masivos:



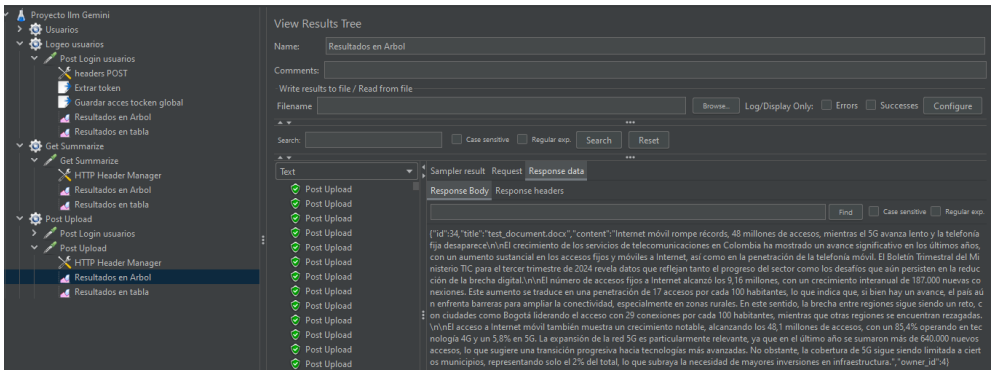
Cargue exitoso de todas las peticiones:

El promedio (4774 ms) es alto, lo que puede indicar que el servidor está tardando en procesar las solicitudes.

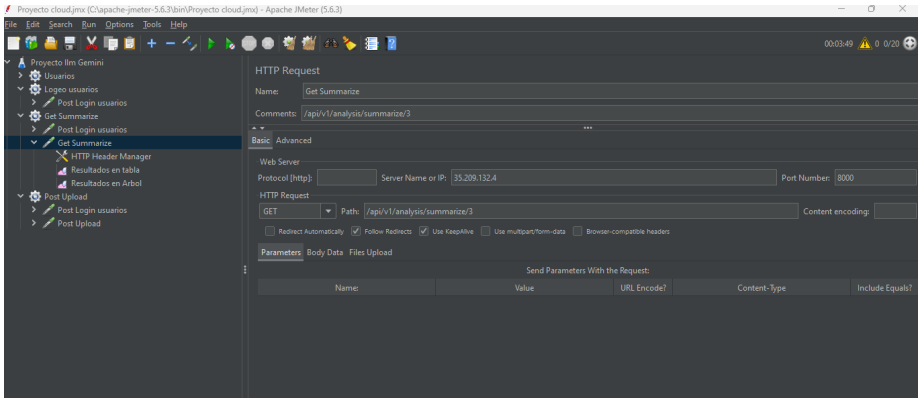
La desviación estándar es alta (2512 ms), lo que significa que los tiempos de respuesta varían mucho entre las solicitudes. Algunas son rápidas, pero otras son mucho más lentas.



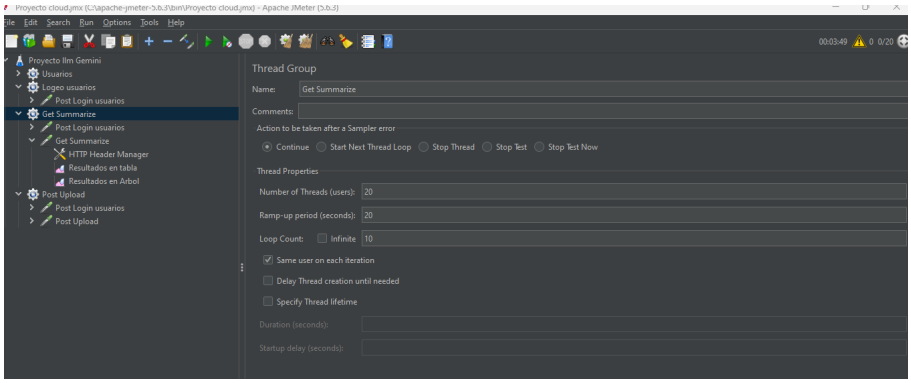
Response de la data del archivo:



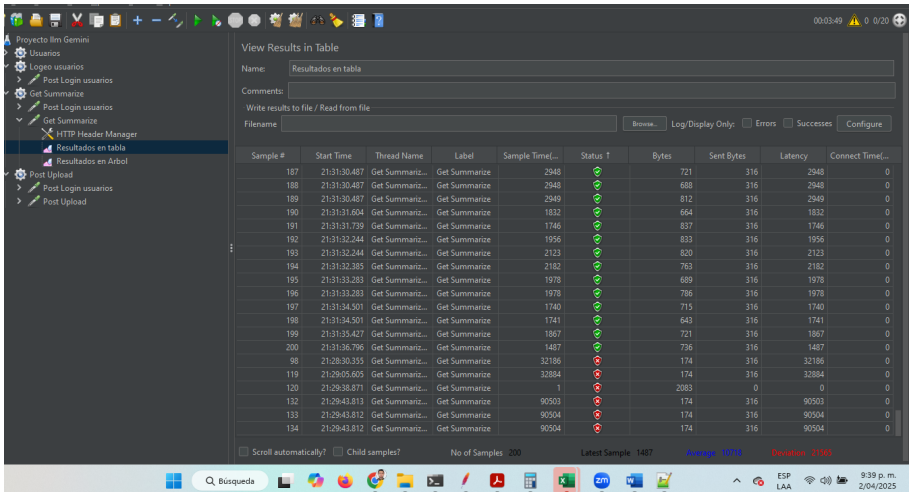
- Resumen del archivo cargado:
  http://35.209.132.4:8000/api/v1/analysis/summarize/3
  Al ser un método Get trae la información de acuerdo con el archivo que se cargó previamente:



Toco realizar una configuración diferente, en los parámetros de carga debido a que bajaba el servidor, numero de usuarios 20 peticiones por segundo 20 y 10 ciclos para un total de 200 peticiones en 200 segundos.



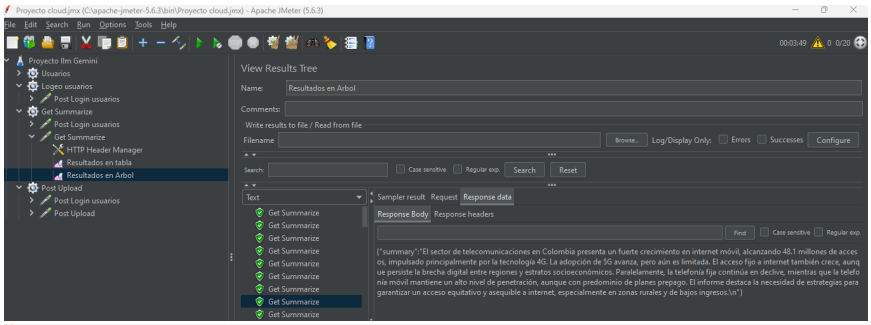
Se evidenciaron 7 peticiones fallidas y 193 correctas con esta configuración de prueba de estrés:

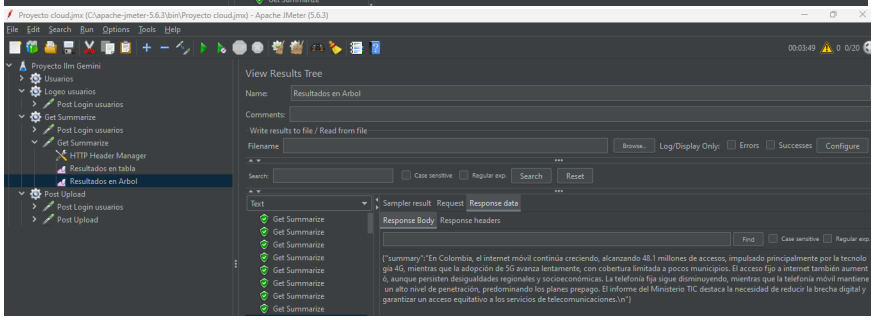


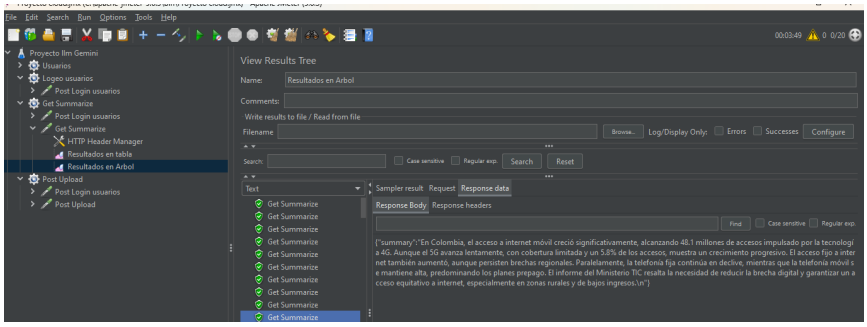
Conclusión:

1. Promedio alto (10.7 segundos): Esto indica que el servidor está tardando mucho en responder.
2. Desviación estándar muy alta (21.5 segundos): Hay una gran inconsistencia en los tiempos de respuesta, lo que sugiere que algunas solicitudes son rápidas y otras extremadamente lentas.
3. Errores en algunas solicitudes (90504 ms, 32.8 s, etc.): Algunas peticiones han fallado o han tomado demasiado tiempo.

Al momento de realizar el response este trae la información diferente de cada petición del mismo texto debido a que el resumen se hace de forma diferente en cada petición:



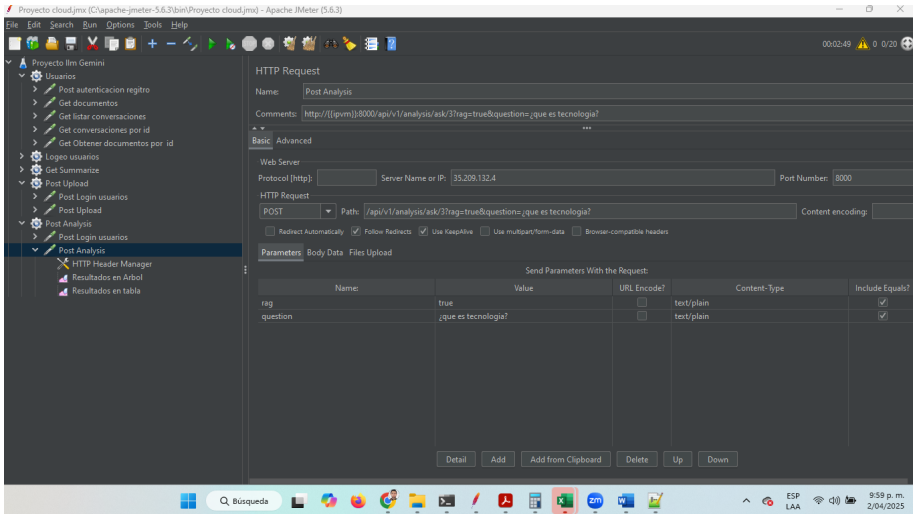




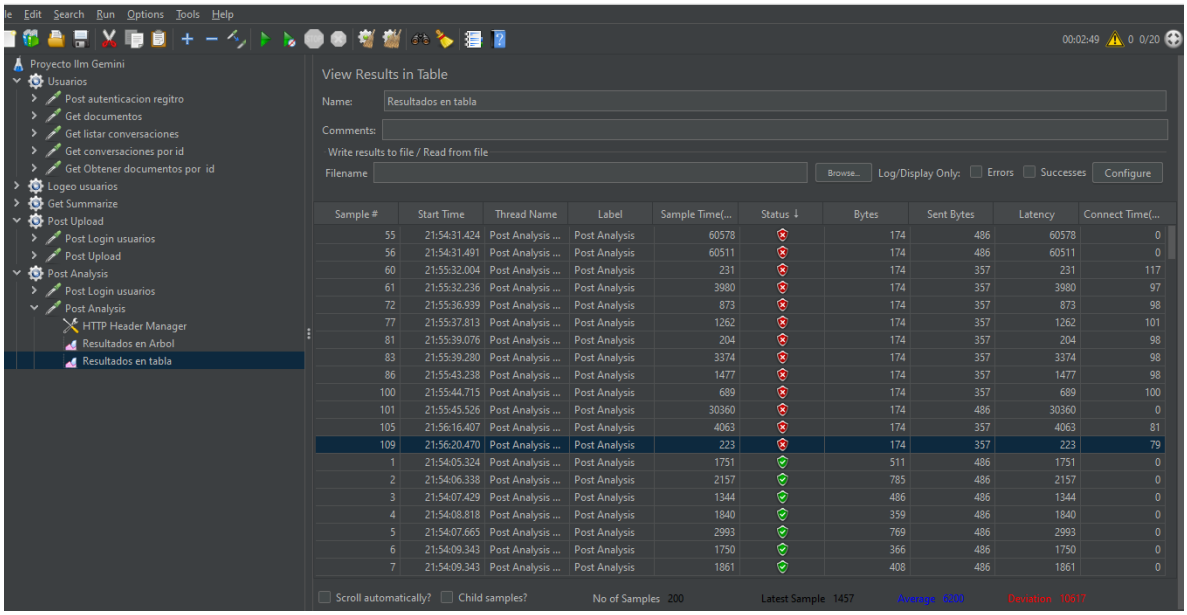
Posibles problemas y soluciones:

Sobrecarga del servidor: El backend puede estar saturado. Se recomienda monitorear CPU, memoria y latencia en el servidor.
Cuellos de botella en la base de datos: Si este endpoint depende de consultas, revisa los tiempos de ejecución en la base de datos.
Problemas de concurrencia: Si hay muchas solicitudes simultáneas, prueba con menos hilos para ver si mejora.
Optimización del código: Usa caching, mejora la estructura de datos o implementa procesamiento en segundo plano si es posible.

- Pregunta sobre el archivo cargado:
  http://35.209.132.4:8000/api/v1/analysis/ask/3?rag=true&question=¿que es tecnologia?
  Envía una pregunta referente al archivo que se cargó previamente, con 20 peticiones cada 20 segundos en un ciclo de 10:



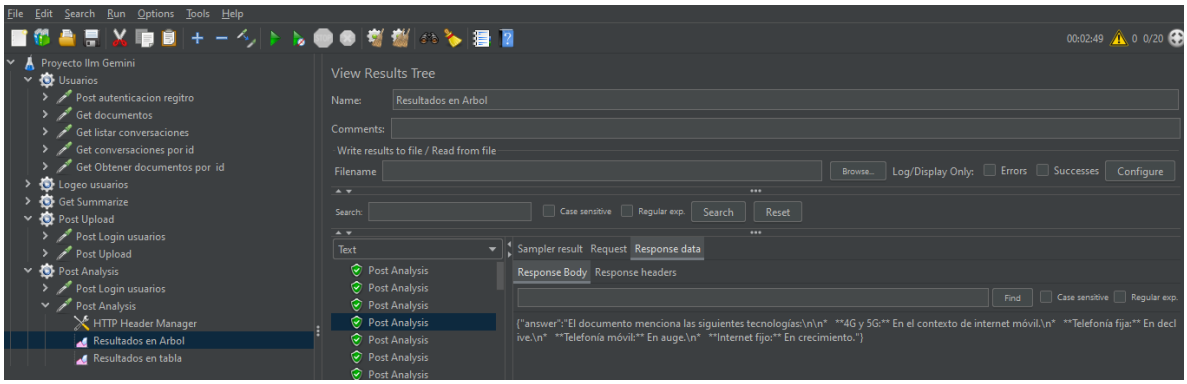
Genero 187 casos exitosos y 13 fallidos al momento de realizar los diferentes envíos de petición:

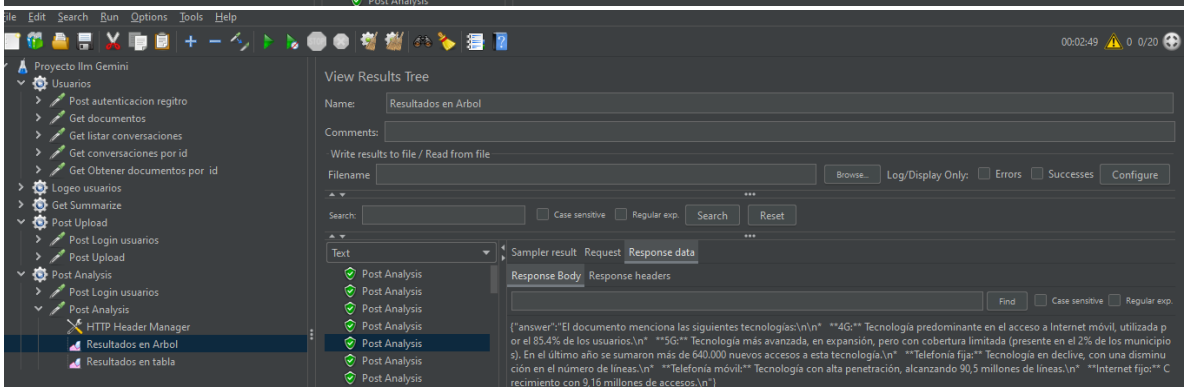


Conclusiones:

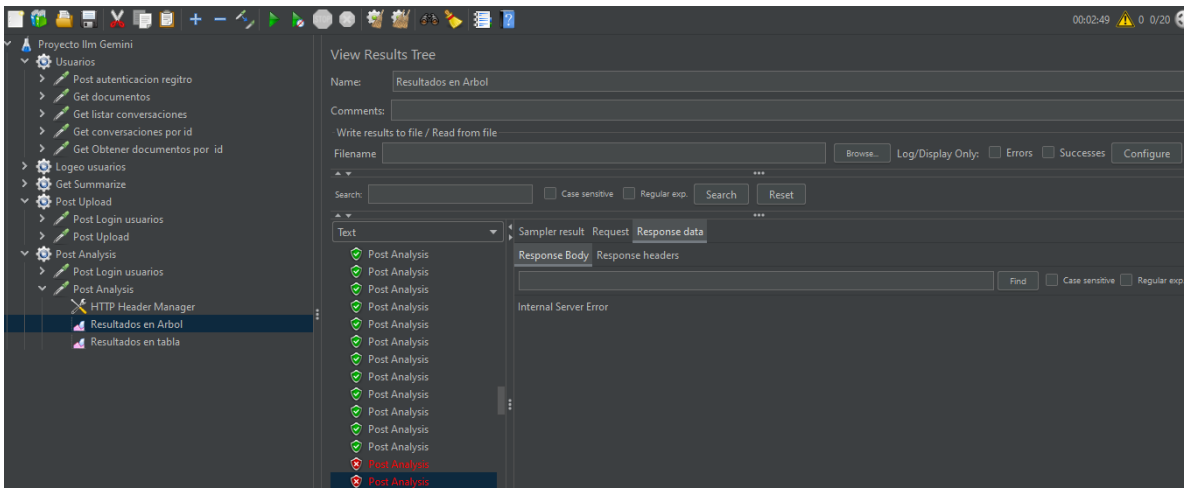
1. Hay tiempos de respuesta extremadamente altos, como 60,578 ms (~60 segundos) en algunas solicitudes.
2. Otras respuestas tardan solo unos cientos de milisegundos.
3. Esto sugiere que algunas solicitudes se quedan atascadas antes de fallar.
4. Tiempo promedio: 6200 ms (~6.2 segundos).
5. Desviación estándar: 10,617 ms, lo que indica que los tiempos de respuesta son muy variables.

Response de casos exitosos:





Response caso fallido:

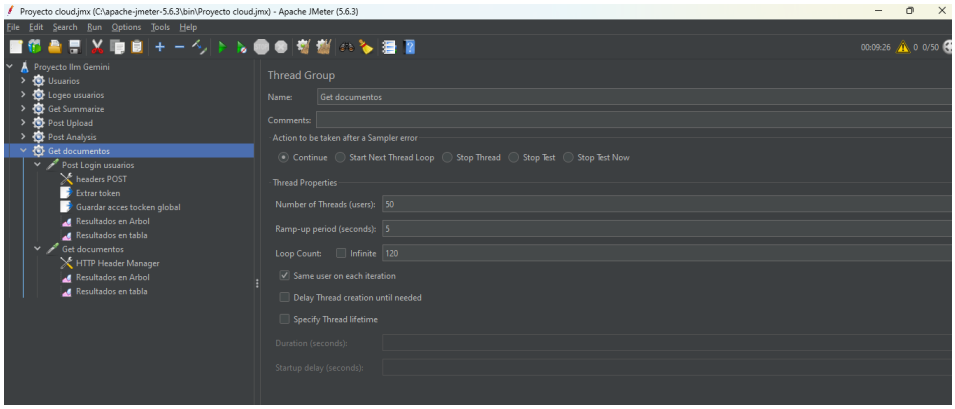


Posibles causas del problema

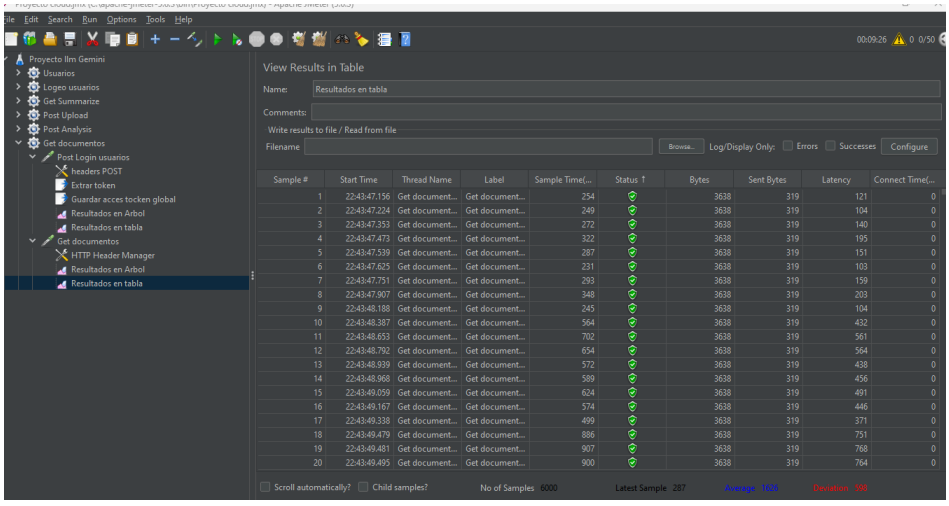
1. Sobrecarga del servidor: Puede estar procesando más solicitudes de las que soporta.

2. Problemas en la base de datos: Consultas lentas o bloqueos pueden estar retrasando las respuestas.
3. Errores en la API: La lógica de negocio o el backend pueden estar devolviendo errores internos.
4. Timeouts: Las peticiones pueden estar tardando demasiado y fallando por configuración de JMeter o el servidor.

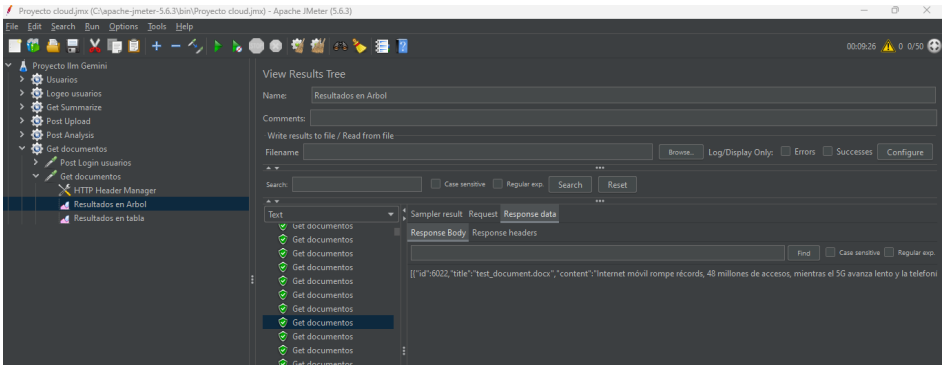
- Trae los documentos de la sesión correspondiente:
  http://35.209.132.4:80008000/api/v1/documents/listDocuments
  Se realizan 50 peticiones cada 50 segundos por 120 ciclos funcionando correctamente:



La ejecución fue exitosa en sus 6000 peticiones.



Podemos ver el titulo del documento cargado y su contenido en formato de texto.



Conclusiones:

1. Tiempo promedio: 1,832 ms (~1.8 segundos).
2. Desviación estándar: 596 ms, lo que significa que los tiempos de respuesta son relativamente consistentes.
3. La mayoría de los tiempos de respuesta están entre 1 y 2 segundos, con algunas variaciones.
4. Buen desempeño de la API: No hay errores y los tiempos de respuesta son aceptables.
5. Algunas variaciones en latencia: Aunque no son críticas, pueden mejorar con optimización de red o del backend.
6. Posible optimización: Si el objetivo es reducir tiempos de respuesta por debajo de 1 segundo, podrías revisar mejoras en la infraestructura o caché.