**Análisis de Resultados :**

**Antecedentes**

1. Dado el siguiente cuadro de resultados de una prueba de carga detallados con los valores obtenidos luego de la ejecución de la prueba (Archivo: textSummary.txt), necesitamos realizar un análisis de los resultados encontrados:

Imagen que contiene Tabla

El contenido generado por IA puede ser incorrecto.

**Información Adicional:**

**El monitoreo** de la prueba, arrojó el siguiente diagrama:

Imagen que contiene Gráfico

El contenido generado por IA puede ser incorrecto.

Resultados:

**Tasa de éxito y fallos**

Checks exitosos: 97.55 % (269 891 de 276 650 solicitudes).

Solicitudes fallidas: 2.44 % (≈ 6 759 peticiones).

Esto indica un nivel aceptable de confiabilidad, aunque el porcentaje de fallos no es despreciable:

Hallazgos:

**Rendimiento de red y latencia**

Duración promedio de solicitud (http\_req\_duration): 861.68 ms

**Duración esperada de respuesta (expected\_response:true):**

Promedio 735.84 ms, con valores máximos de hasta 26.72 s.

Se observa cierta variabilidad, lo que sugiere momentos de sobrecarga.

**Tráfico total**

Datos recibidos: 842 MB (223 kB/s) y Datos enviados: 588 MB (156 kB/s)

El volumen de tráfico indica que el sistema maneja una cantidad considerable de intercambio, útil para evaluar la capacidad de red y el backend.

**Usuarios virtuales (VUs)**

Máximo concurrente: 140 usuarios virtuales.

Mínimo: 2.

La prueba alcanzó picos sostenidos de carga máxima (140 VUs) con fases intermedias de reducción de tráfico (como se aprecia en el gráfico).

**Errores HTTP**

4xx (errores de cliente): 769 fallos (~0.2 %)

5xx (errores de servidor): 5987 fallos (~1.5 %)

Esto refleja que la mayoría de los problemas fueron del lado del servidor, señal de posibles cuellos de botella en la aplicación o infraestructura.

**Análisis del diagrama (VUs vs. http\_reqs/s)**

El gráfico muestra la relación entre usuarios virtuales (VUs) y peticiones por segundo (http\_reqs/s).

Se observan tres fases:

1.- Al inicio: Se observa un incremento rápido de usuarios hasta 140 VUs; la tasa de peticiones crece rápidamente.

2.- Caída intermedia ( aprox. 02:00 h): reducción drástica de peticiones aprox. 20 req/s, mientras se mantienen 140 VUs activos.

Probablemente bloqueos o tiempos de espera excesivos del sistema bajo carga máxima.

3.-Recuperación: tras la caída, la tasa se estabiliza en torno a 80–90 req/s, manteniéndose un promedio contante hasta el final de la prueba.

Esto sugiere que el sistema se recupera correctamente después de una fase de saturación temporal, pero existe un punto de degradación al límite de concurrencia.

Conclusiones

* El **sistema soporta adecuadamente hasta 100 req/s con 140 usuarios concurrentes**, manteniendo tiempos de respuesta inferiores a 1 s en la mayoría de los casos.
* Sin embargo, **se detectaron picos de latencia** (hasta 26 s) y **fallos del 2.4 %**, lo cual puede impactar negativamente la experiencia de usuario en escenarios de alta concurrencia.
* Los **errores 5xx** son los más preocupantes, ya que indican que el servidor no pudo procesar correctamente las solicitudes en determinados momentos de carga.
* La **recuperación posterior a la degradación** sugiere una buena gestión de recursos, pero es probable que la aplicación o base de datos alcance un límite de throughput en torno a los 80–90 req/s.

Recomendaciones

1. **Optimizar el backend**

* Revisar la gestión de conexiones a base de datos o servicios externos.
* Implementar pooling, caching o asynchronous I/O donde sea posible.

1. **Revisar límites de infraestructura**

* Analizar el rendimiento del servidor (CPU, memoria, ancho de banda) durante el pico de 140 VUs.
* Escalar horizontalmente (más instancias) si el cuello de botella es de hardware.

1. **Monitoreo y logs**

* Correlacionar los picos de latencia con registros del servidor o métricas de infraestructura (CPU, GC, I/O).
* Identificar endpoints específicos con mayor incidencia de errores 5xx o tiempos altos.