Load Testing – k6

**El código debe ser utilizando k6 y con javascript.**

1. **Responder las siguientes preguntas**
   1. **¿Qué es una prueba de carga?**
   2. **¿Cuándo se utilizan?**
   3. **¿Cómo es el proceso de diseño?**
2. **¿Qué tipos de pruebas de carga existen?**
   1. **Crear un cuadro comparativo con descripción, objetivos**
3. **Diseñar una prueba de carga tipo Stress**
   1. **Duración total 2 minutos**
   2. **Envía en el body un string con el nombre de un país**
   3. **Valida que la respuesta es exitosa.**
   4. **Valida que en la respuesta hay un string.**
   5. **Incluir una impresión de pantalla de los resultados de la prueba**
4. **Diseñar una prueba de carga que realice 27 request por segundo.**
   1. **Incluir una impresión de pantalla de los resultados de la prueba.**
5. **Diseñar una prueba de carga que simule 300 usuarios por minuto.**
   1. **Incluir una impresión de pantalla de los resultados.**

**1. Responder las siguientes preguntas**

**1.1¿Qué es una prueba de carga?**

Las pruebas de carga son simulaciones diseñadas para evaluar cómo se comporta una aplicación o sistema\* bajo condiciones específicas de uso. Ayudan a los desarrolladores a identificar problemas como retrasos del sistema, tiempos de carga de páginas lentos o bloqueos cuando diferentes niveles de tráfico acceden a la aplicación

* 1. **¿Cuándo se utilizan?**

Las pruebas de carga se utilizan en varias etapas clave del ciclo de vida del desarrollo de software y son especialmente importantes cuando se necesita asegurar que una aplicación o sistema\* pueda manejar un volumen de usuarios o tráfico significativo sin perder rendimiento**.**

-Antes del Lanzamiento de una Aplicación, para asegurarse de que la aplicación o sistema esté preparado para usuarios reales y pueda manejar el tráfico esperado sin problemas.

-Después de Realizar Cambios Significativos (ej. Actualizaciones, migración a nuevas bases de datos, servidores), los cambios en la infraestructura, el código, la configuración o la arquitectura de la aplicación pueden afectar el rendimiento.

-Previo a Eventos de Alto Tráfico, ventos especiales, como campañas de ventas, promociones, o lanzamientos, el tráfico puede aumentar considerablemente, y la aplicación debe poder manejar ese pico sin caídas.

- Periódicamente en el Ciclo de Vida de la Aplicación.

* 1. **¿Cómo es el proceso de diseño?**

* Definir los Objetivos de la Prueba - ¿Qué se quiere medir?
* Identificar el Entorno de Prueba - Configurar un entorno representativo, es decir, reflejar el ambiente real de producción
* Definir el Escenario de Carga - Determinar cuántos usuarios virtuales simularás y cómo se comportarán.
* Desarrollar los Scripts de Prueba –
* Configurar y Ejecutar la Prueba –
* Recopilación y Análisis de Resultados
* Optimización y Ajustes

2. **¿Qué tipos de pruebas de carga existen?**

**1.Crear un cuadro comparativo con descripción, objetivos**

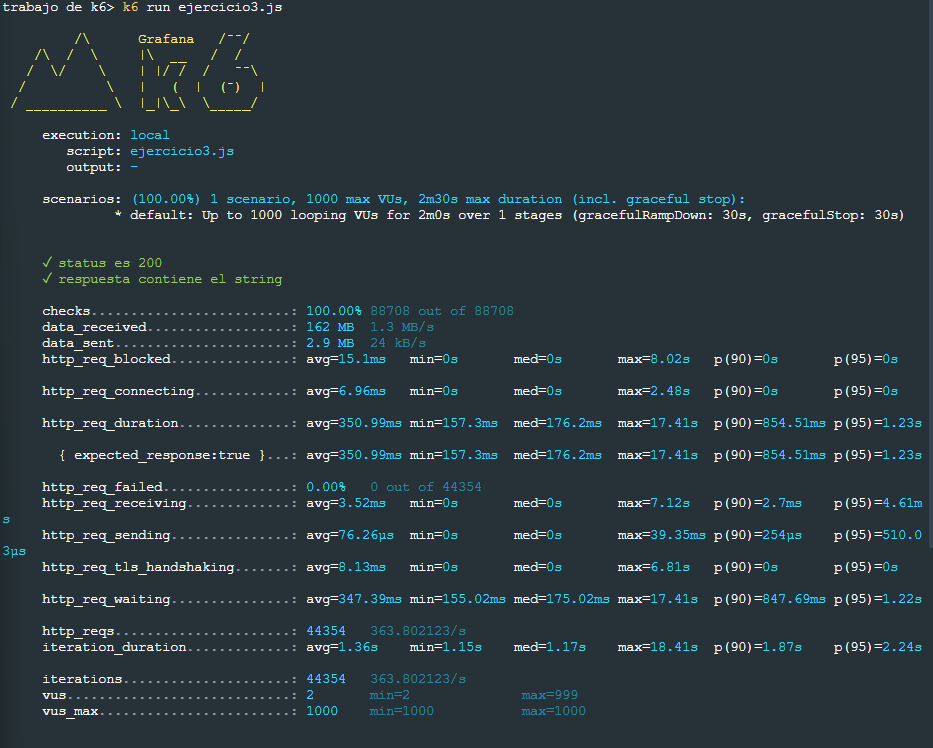
|  |  |  |
| --- | --- | --- |
| **Tipo de prueba** | **Descripción** | **Objetivo** |
| Prueba de Carga Básica | Simula usuarios constantes para probar el rendimiento en condiciones normales. | Verificar que el sistema funcione adecuadamente bajo la carga esperada. |
| Prueba de Estrés | Aumenta la carga gradualmente hasta que el sistema falla o alcanza su límite. | Identificar el punto de ruptura y los cuellos de botella. |
| Prueba de Resistencia | Mantiene una carga constante durante un largo período de tiempo. | Evaluar la estabilidad del sistema bajo uso prolongado. |
| Prueba de Pico | Aplica picos de carga de manera repentina para probar la reacción del sistema. | Comprobar cómo se comporta el sistema ante incrementos repentinos de carga y su recuperación. |
| Prueba de Escalabilidad | Incrementa la carga para probar la capacidad del sistema de escalar. | Verificar si el sistema puede escalar eficientemente a medida que crece el tráfico. |
| Prueba de Capacidad | Determina el máximo de usuarios que el sistema puede manejar sin degradación. | Definir la capacidad máxima del sistema antes de que su rendimiento se vea afectado. |

**3. Diseñar una prueba de carga tipo Stress**

* 1. **Duración total 2 minutos**
  2. **Envía en el body un string con el nombre de un país**
  3. **Valida que la respuesta es exitosa.**
  4. **Valida que en la respuesta hay un string.**
  5. **Incluir una impresión de pantalla de los resultados de la prueba**

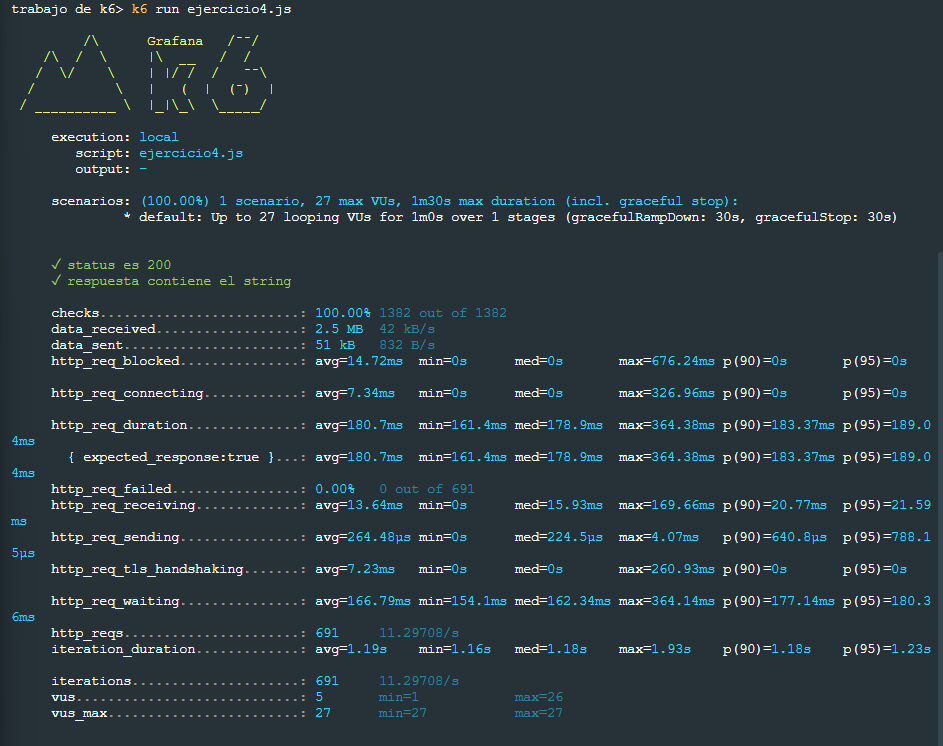
1. import http from 'k6/http';
2. import { check, sleep } from 'k6';
3. export *let* options = {
4. stages: [
5. { duration: '2m', target: 1000 },
6. ],
7. };
8. export default *function* () {
10. *let* res = http.get('https://restcountries.com/v3.1/name/argentina');
11. check(res, {
12. 'status es 200': (*r*) *=>* *r*.status === 200,
13. 'respuesta contiene el string': (*r*) *=>* *r*.body.includes('Argentina'),
14. });
15. sleep(1);
16. }

Salida por consola:



1. **Diseñar una prueba de carga que realice 27 request por segundo.**
   1. **Incluir una impresión de pantalla de los resultados de la prueba.**
2. //Prueba de Carga Básica    Simula usuarios constantes para probar el rendimiento en condiciones normales.  Verificar que el sistema funcione adecuadamente bajo la carga esperada.
3. /\*
4. Diseñar una prueba de carga que realice 27 request por segundo.
5. Incluir una impresión de pantalla de los resultados de la prueba.
6. \*/
7. import http from 'k6/http';
8. import { check, sleep } from 'k6';
9. export let options = {
10. stages: [
11. { duration: '1m', target: 27 }, // Sostiene 27 solicitudes por segundo
12. ],
13. };
14. export default function () {
15. let res = http.get('https://restcountries.com/v3.1/name/argentina');
16. check(res, {
17. 'status es 200': (r) => r.status === 200,
18. 'respuesta contiene el string': (r) => r.body.includes('Argentina'),
19. });
20. sleep(1);
21. }
22. //para ejecutar el código usar: k6 run ejercicio4.js

Salida por consola:



**5. Diseñar una prueba de carga que simule 300 usuarios por minuto.**

* 1. **Incluir una impresión de pantalla de los resultados.**

1. // Prueba de Carga Básica   Simula usuarios constantes para probar el rendimiento en condiciones normales.  Verificar que el sistema funcione adecuadamente bajo la carga esperada.
2. /\*\*
3. Diseñar una prueba de carga que simule 300 usuarios por minuto.
4. Incluir una impresión de pantalla de los resultados.
5. \*/
6. import http from 'k6/http';
7. import { check, sleep } from 'k6';
8. export let options = {
9. stages: [
10. { duration: '1m', target: 5 },
11. { duration: '1m', target: 300 }, // Aumenta hasta 300 usuarios/minuto
12. ],
13. };
14. export default function () {
15. let res = http.get('https://restcountries.com/v3.1/name/argentina');
16. check(res, {
17. 'status es 200': (r) => r.status === 200,
18. 'respuesta contiene el string': (r) => r.body.includes('Argentina'),
19. });
20. sleep(1);
21. }
22. //para correr este test: k6 run ejercicio5.js

Salida por consola:

