

## K6

**NOMBRE:** Roddy Steeven Zamora Rodríguez

### EJERCICIO:

Ejecutar el siguiente ejercicio práctico usando cualquier herramienta de rendimiento, deseable K6.

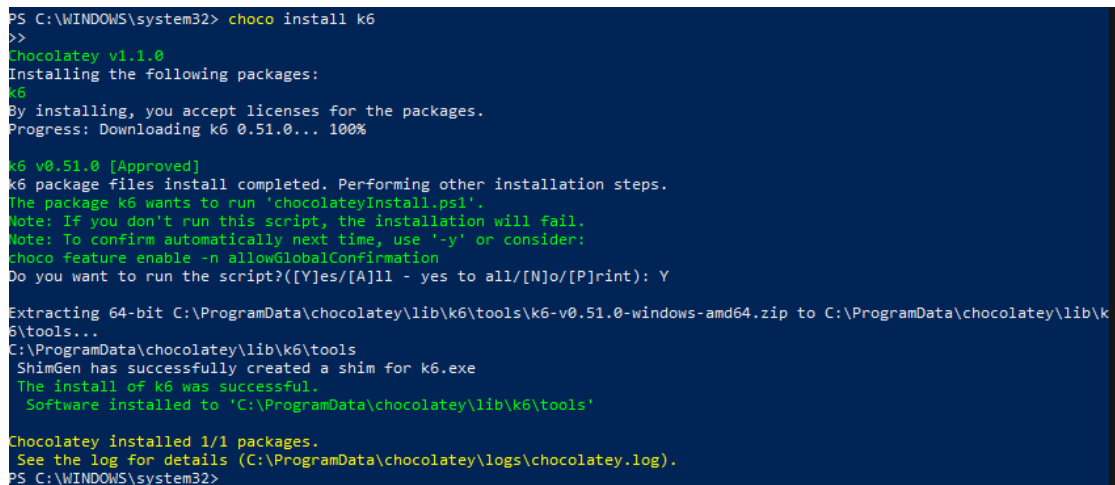
Se debe considerar lo siguiente:

- a) Generar un script que permita grabar la siguiente página:  
<https://petstore.octoperf.com/actions/Catalog.action>
- b) Acción del script: Navegación de la sección del menú Menú: Fish ProductID: FI-SW-01
- c) Ejecutar una prueba de carga con 20 hilos concurrentes durante 10 minutos
- d) Obtener los siguientes resultados:
  - i. Summay Report
  - ii. Graph Result
- e) Conclusiones de la prueba de acuerdo a los resultados obtenidos.
- f) incluir archivo README.md con instrucciones claras de compilación y ejecución

### PRERREQUISITOS:

- 1.- Instalar K6 en Windows.

#### *choco install k6*



```
PS C:\WINDOWS\system32> choco install k6
>>
Chocolatey v1.1.0
Installing the following packages:
k6
By installing, you accept licenses for the packages.
Progress: Downloading k6 0.51.0... 100%

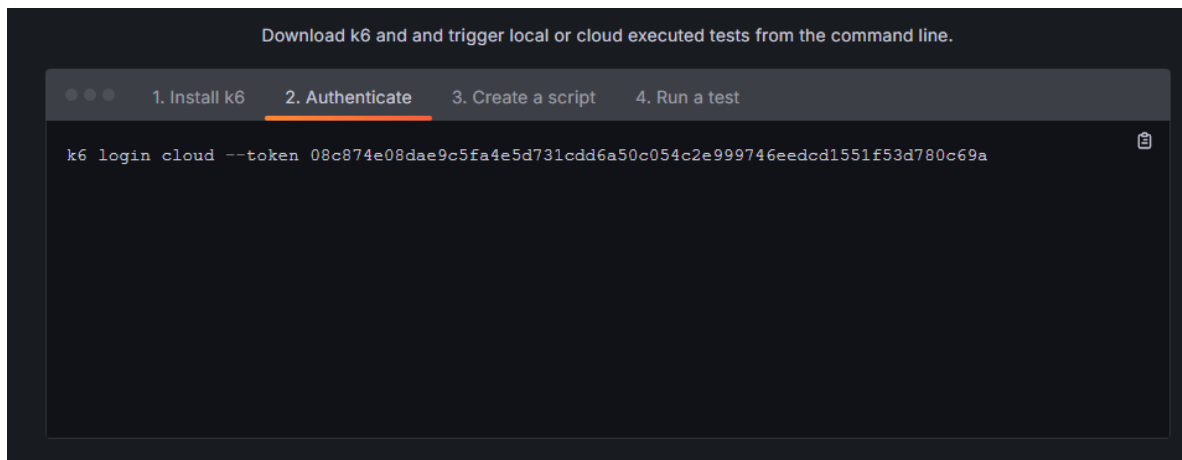
k6 v0.51.0 [Approved]
k6 package files install completed. Performing other installation steps.
The package k6 wants to run 'chocolateyinstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): Y

Extracting 64-bit C:\ProgramData\chocolatey\lib\k6\tools\k6-v0.51.0-windows-amd64.zip to C:\ProgramData\chocolatey\lib\k6\tools...
C:\ProgramData\chocolatey\lib\k6\tools
ShimGen has successfully created a shim for k6.exe
The install of k6 was successful.
Software installed to 'C:\ProgramData\chocolatey\lib\k6\tools'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32>
```

- 2.- Nos registramos en Grafana Cloud para que nuestra prueba genere un reporte aprovechando la nube <https://grafana.com/products/cloud/k6/>.

Una vez ahí nos autenticamos para al ejecutar el test se cree el reporte respectivo.



## PRUEBA:

```
JS test.js x
JS test.js > [o] opciones
1  import http from 'k6/http';
2  import { sleep, check } from 'k6';
3
4  export let opciones = {
5    vus: 20, // cantidad Users
6    duration: '10m',
7    summaryTrendStats: ['avg', 'min', 'med', 'max', 'p(90)', 'p(95)'],
8  };
9
10 export default function () {
11   // 1. Página principal
12   let res = http.get('https://petstore.octoperf.com/actions/Catalog.action');
13   check(res, {
14     'status was 200': (r) => r.status === 200,
15   });
16   sleep(1);
17
18   // 2. Sección Fish
19   res = http.get('https://petstore.octoperf.com/actions/Catalog.action?viewCategory=&categoryId=FISH');
20   check(res, {
21     'status was 200': (r) => r.status === 200,
22   });
23   sleep(1);
24
25   // 3. Producto FI-SW-01
26   res = http.get('https://petstore.octoperf.com/actions/Catalog.action?viewProduct=&productId=FI-SW-01');
27   check(res, {
```

Se inicia sesión en Graphana

```
PS C:\Users\roddy\OneDrive\Escritorio\Roddy\SofkaPruebaRZA\retoAutK6-RZA\k6-rza> k6 login cloud --token 08c874e08dae9c5fa4e5d731cdd6a50c054c2e999746eedcd1551f53d780c69a
token: 08c874e08dae9c5fa4e5d731cdd6a50c054c2e999746eedcd1551f53d780c69a
Logged in successfully, token saved in C:\Users\roddy\AppData\Roaming\loadimpact\k6\config.json
```

Y se ejecuta la prueba apuntando a la nube.

*k6 cloud test.js*



The image shows a VS Code editor with a file named `test.js` open. The file contains a JavaScript script that uses the `http` module from the `k6` library. The script defines a function `testjs` that performs a series of HTTP requests and returns a summary of the results. The script is executed in a terminal window, and the output shows the results of the execution, including the number of iterations, the duration of the test, and the status of the requests.

```
1 import http from 'k6/http';
2 import { sleep, check } from 'k6';
3
4 export let opciones = {
5   vus: 20, // cantidad Users
6   duration: '10m',
7   summaryTrendStats: ['avg', 'min', 'med', 'max', 'p(90)', 'p(95)'],
8 };
9
10 export default function () {
11   // 1. Página principal
```

The terminal output shows the following results:

```
data_received.....: 17 kB 4.0 kB/s
data_sent.....: 852 B 200 B/s
http_req_blocked.....: avg=177.4ms min=0s med=0s max=532.21ms p(90)=425.77ms p(95)=478.99ms
http_req_connecting.....: avg=72.64ms min=0s med=0s max=220.94ms p(90)=176.75ms p(95)=108.84ms
http_req_duration.....: avg=227.09ms min=220.54ms med=223.01ms max=237.71ms p(90)=234.77ms p(95)=236.24ms
http_req_failed.....: 0.00% ✓ 0 X 3
http_req_receiving.....: avg=0s min=0s med=0s max=0s p(90)=0s p(95)=0s
http_req_sending.....: avg=168.43µs min=0s med=0s max=595.29µs p(90)=404.24µs p(95)=454.76µs
http_req_tls_handshaking.....: avg=82.75ms min=0s med=0s max=248.25ms p(90)=198.6ms p(95)=223.43ms
http_req_waiting.....: avg=226.92ms min=220.54ms med=222.5ms max=237.71ms p(90)=234.66ms p(95)=236.19ms
http_reqs.....: 3 0.704889/s
iteration_duration.....: avg=4.25s min=4.25s med=4.25s max=4.25s p(90)=4.25s p(95)=4.25s
iterations.....: 1 0.234963/s
vus.....: 1 min=1 max=1
vus_max.....: 1 min=1 max=1
```

running (00m04.3s), 0/1 VUs, 1 complete and 0 interrupted iterations  
default ✓ [=====] 1 VUS 00m04.3s/10m0s 1/1 iters, 1 per VU  
PS C:\Users\roddy\OneDrive\Escritorio\Roddy\SofkaPruebaRZA\retoAutK6-RZA\k6-rza>

The image shows a JSON file named `summary.json` with the following content:

```
{
  "root_group": {
    "path": "",
    "id": "d41d8cd98f00b204e9800998ecf8427e",
    "groups": {},
    "checks": {
      "status was 200": {
        "name": "status was 200",
        "path": ">::status was 200",
        "id": "1461660757a913d4fb82ac4c5e1009de",
        "passes": 3,
        "fails": 0
      }
    },
    "name": ""
  },
  "metrics": {
    "http_req_receiving": {
      "max": 0,
      "p(90)": 0,
      "p(95)": 0,
      "avg": 0,
      "min": 0,
      "med": 0
    },
    "iterations": {
      "count": 1,
      "rate": 0.23496292496510685
    }
  }
}
```

### **Conclusiones:**

- La tasa de solicitudes es relativamente baja, alcanzando un máximo de alrededor de 0.67 solicitudes por segundo (reqs/s).
- Esta tasa de solicitudes indica que la prueba no está generando una alta carga de tráfico hacia el servidor.
- El tiempo de respuesta máximo es de 118 milisegundos (ms). Esto sugiere que las solicitudes están siendo manejadas de manera rápida, lo cual es positivo en términos de rendimiento del servidor.
- La tasa de fallos (Failure Rate) parece ser inexistente o extremadamente baja, ya que no se observa ninguna línea roja en el gráfico.