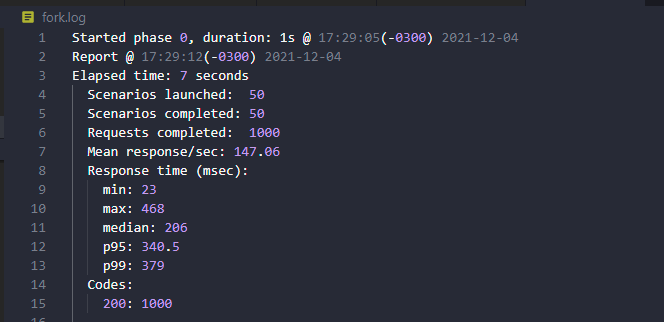
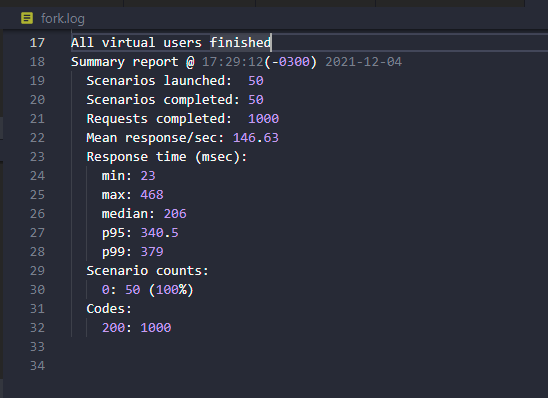
**Ruta \info – modo fork con Console Log**

Ejecuté el artillery según lo solicitado en el desafío en la primera instancia con el console log activado para mostrar la información en la ruta /ecommerce/info



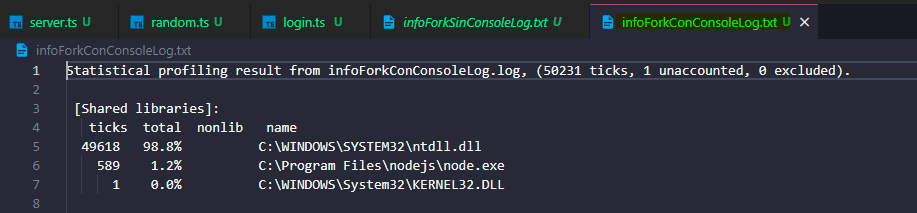
Resultados del artillery:

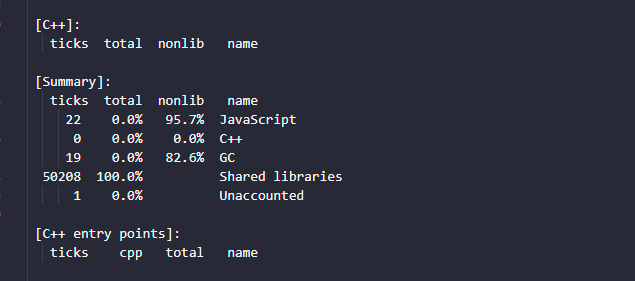




**Profiler**

Ejecuté el server.js con el profiler y estos son los resultados:



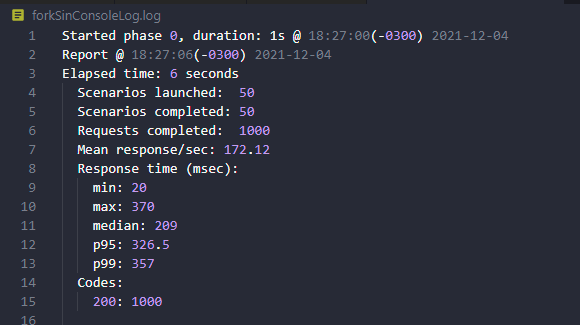


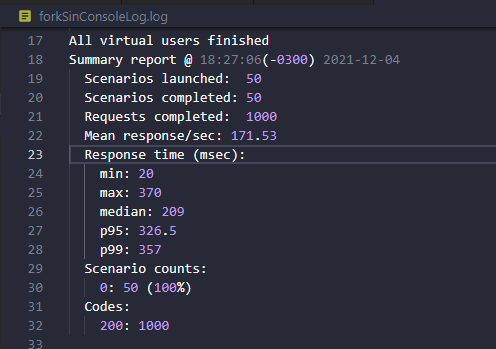
**Ruta \info – modo fork sin Console Log**

Ejecuté el artillery según lo solicitado en el desafío en la primera instancia sin el console log que muestra en consola la información en la ruta /ecommerce/info



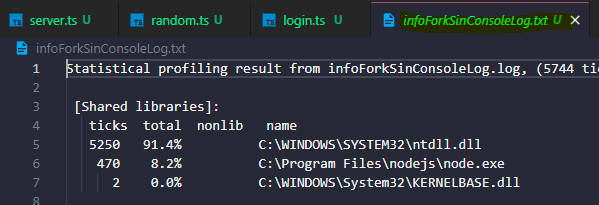
Resultados del artillery:

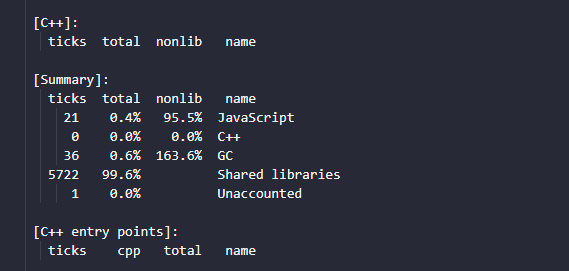




**Profiler**

Ejecuté el server.js con el profiler y estos son los resultados:

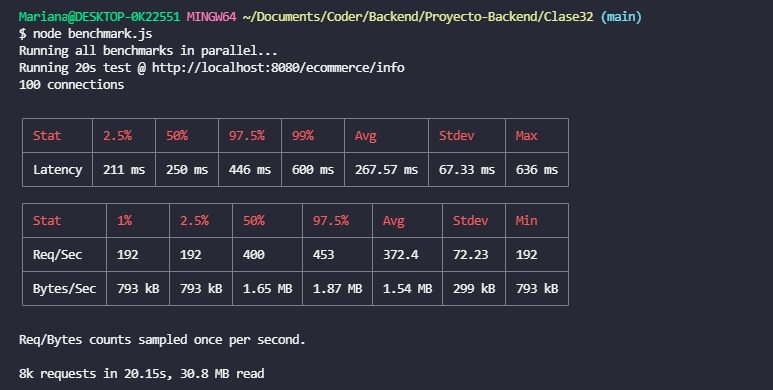




**Autocannon**

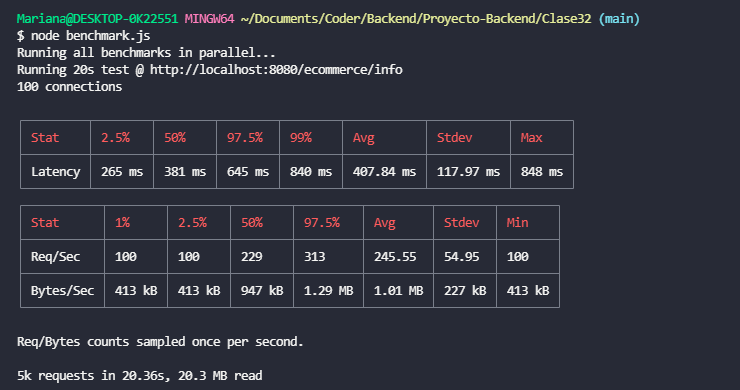
Ejecuté al autocannon según lo solicitado, en primera instancia sin el console log activado.

1. Sin Console Log



1. Con Console Log

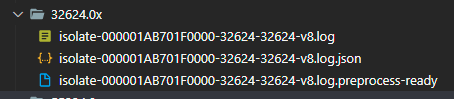
Resultados del autocannon con el console log activado:



Lo ejecuté con 0x, y me generó supuestamente el flamegraph.



Esto fue lo que me generó el 0x:



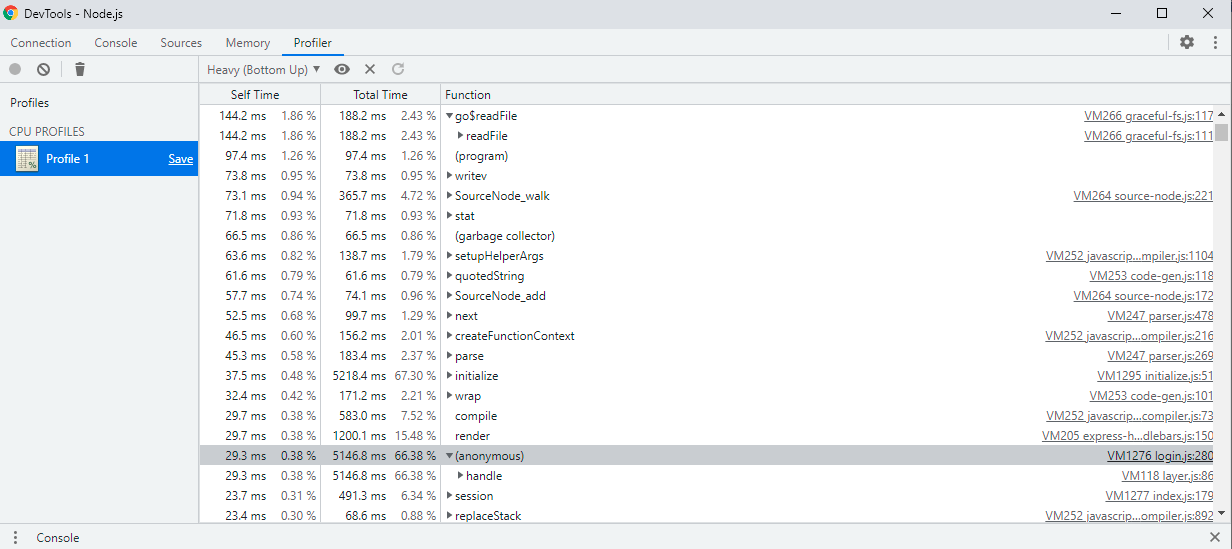
No me generó el html con el gráfico.

**Inspect de Chrome**

1. Con ConsoleLog







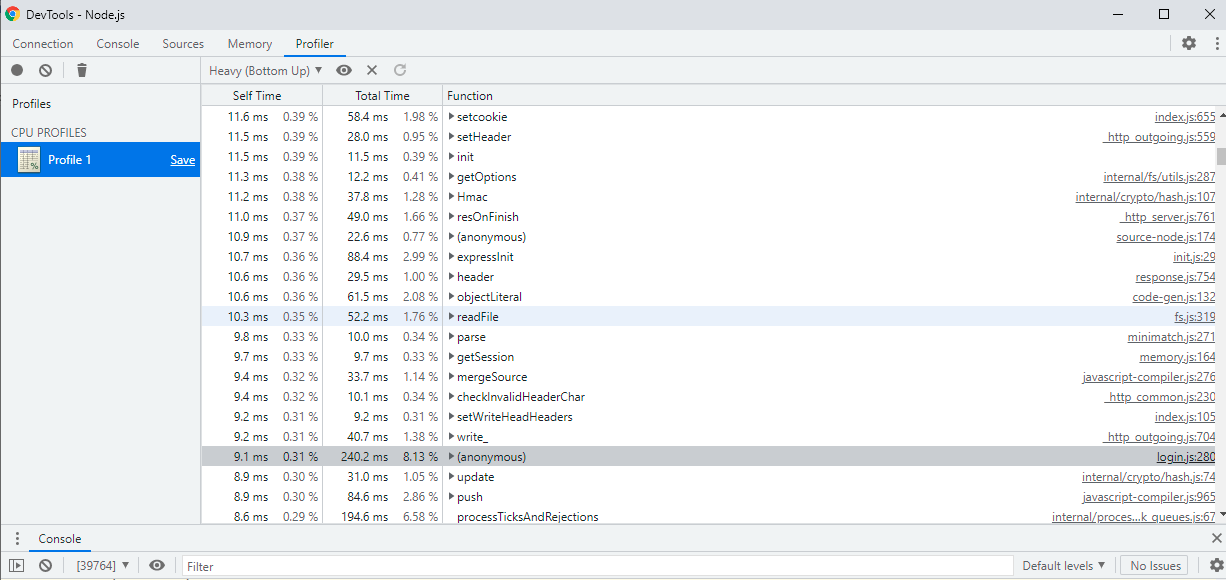
Encontré el login.js y me fijé que tardó bastante, y entré a ver el código.



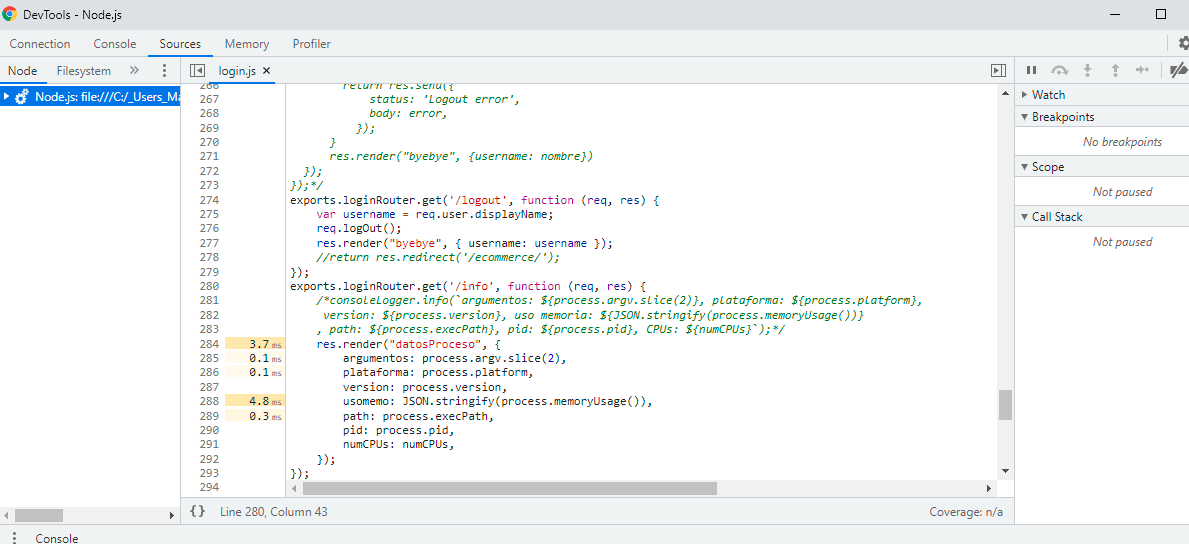
1. Sin ConsoleLog







Obviamente, sin el console log demoró mucho menos.

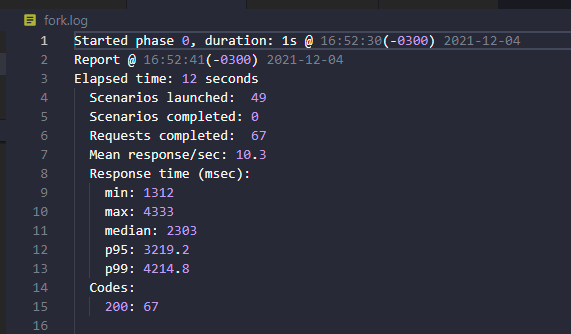


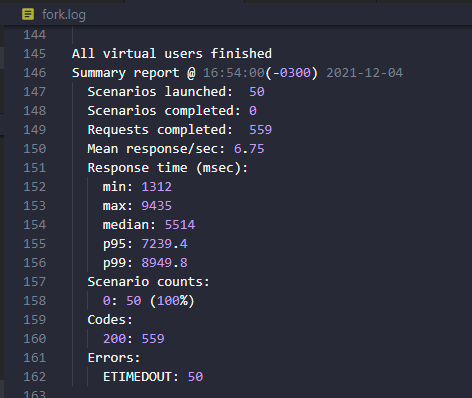
Esta parte no estaba en la consigna del desafío, pero son las primeras pruebas que hice de artillery en modo fork y cluster.

Obviamente, en cluster es más rápido, por lo que logra atender más cantidad de requests en menos tiempo.

FORK







CLUSTER

