

INFORME PRUEBAS SERVIDOR

→node --prof server.js

→artillery quick --count 40 -n 50 "http://localhost:8080/info" > result_test.txt

Phase started: unnamed (index: 0, duration: 1s) 10:14:20(-0300)

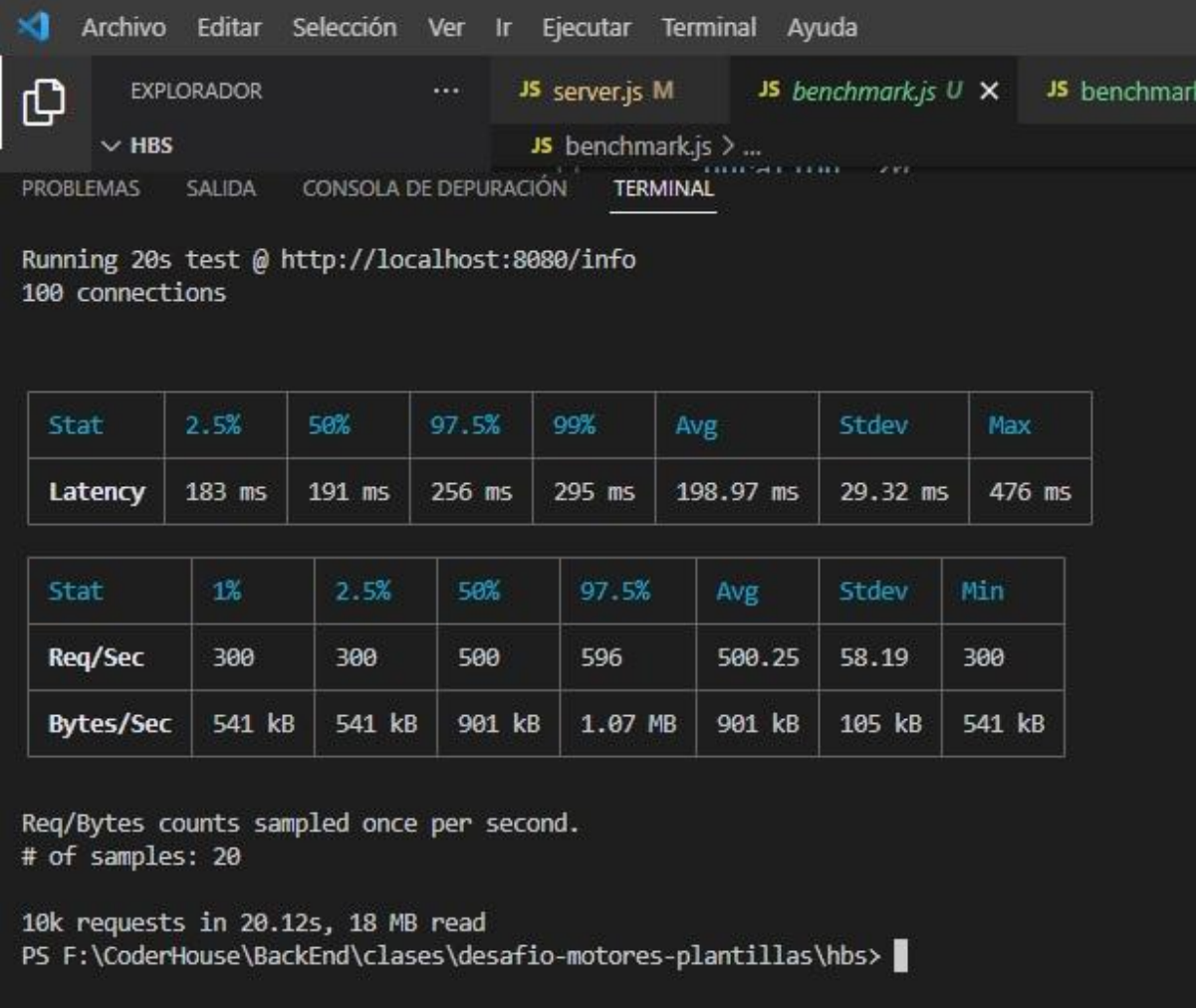
Phase completed: unnamed (index: 0, duration: 1s) 10:14:21(-0300)

All VUs finished. Total time: 10 seconds

Summary report @ 10:14:27(-0300)

http.codes.200:	2000
http.request_rate:	189/sec
http.requests:	2000
http.response_time:	
min:	5
max:	132
median:	96.6
p95:	115.6
p99:	125.2
http.responses:	2000
vusers.completed:	40
vusers.created:	40
vusers.created_by_name.0:	40
vusers.failed:	0
vusers.session_length:	
min:	4345.8
max:	4718.8
median:	4583.6
p95:	4676.2
p99:	4676.2

AUTOCANNON



The screenshot shows the Visual Studio Code interface with the following elements:

- Menu Bar:** Archivo, Editar, Selección, Ver, Ir, Ejecutar, Terminal, Ayuda.
- Explorer (EXPLORADOR):** Shows a file tree with 'HBS' expanded, containing 'server.js' and 'benchmark.js'.
- Terminal:** Displays the output of the AUTOCANNON benchmark.

Terminal Output:

```
Running 20s test @ http://localhost:8080/info
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	183 ms	191 ms	256 ms	295 ms	198.97 ms	29.32 ms	476 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	300	300	500	596	500.25	58.19	300
Bytes/Sec	541 kB	541 kB	901 kB	1.07 MB	901 kB	105 kB	541 kB

Req/Bytes counts sampled once per second.
of samples: 20

10k requests in 20.12s, 18 MB read
PS F:\CoderHouse\BackEnd\clases\desafio-motores-plantillas\hbs> |

0x y Autocannon

→ autocannon -d 20 -c 100 http://localhost:8080/info



Conclusión

Según la información proporcionada por las distintas pruebas, podemos determinar que nuestro servidor está optimizado. Hay ciertas librerías que provocan una demora en los tiempos de carga, pero no son de consideración.