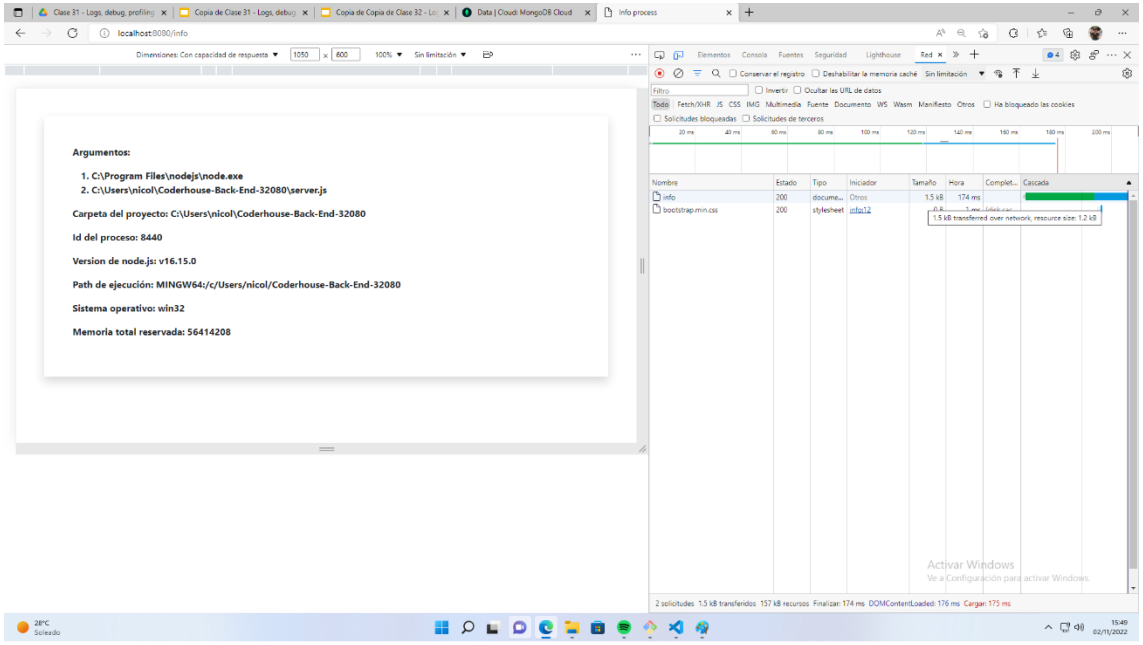
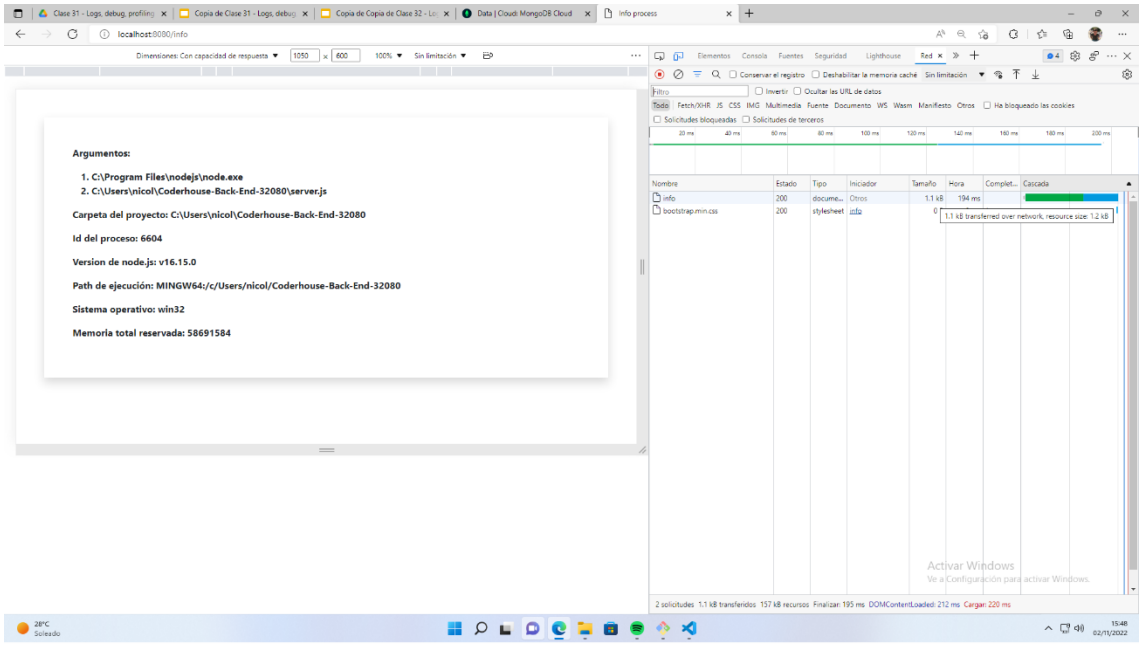
Con el compress



Sin el compress



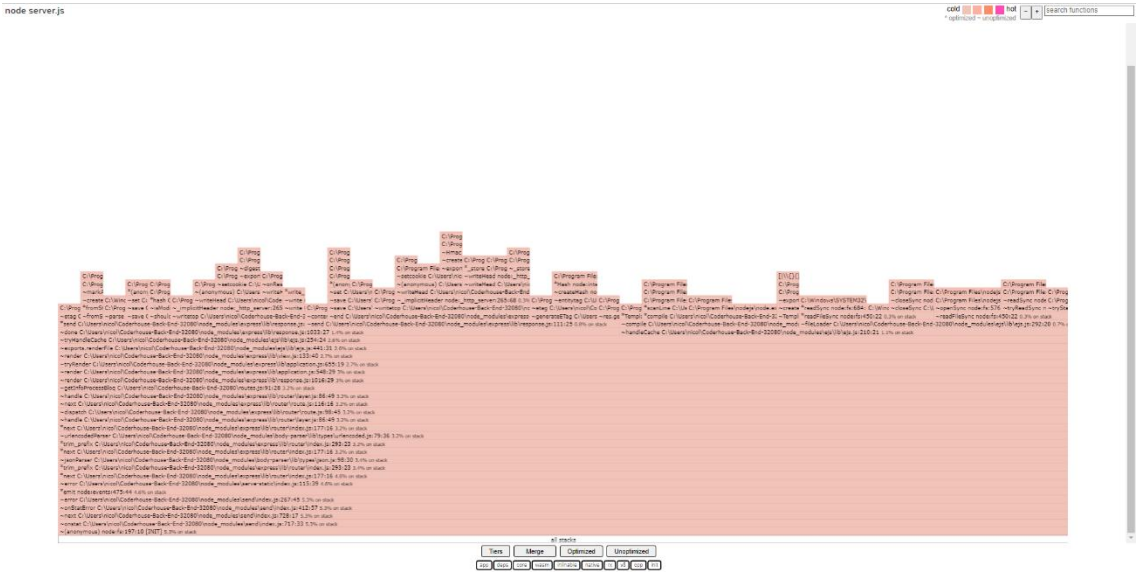
InfoBloqueante

```
-------------------------------
Summary report @ 23:20:35(-0300)
-------------------------------

http.codes.200: ..................................................................... 1000
http.request_rate: .................................................................. 26/sec
http.requests: ...................................................................... 1000
http.response_time:
  min: .............................................................................. 2
  max: .............................................................................. 1639
  median: ........................................................................... 61
  p95: .............................................................................. 757.6
  p99: .............................................................................. 788.5
http.responses: ..................................................................... 1000
vusers.completed: ................................................................... 20
vusers.created: ..................................................................... 20
vusers.created_by_name.0: ........................................................... 20
vusers.failed: ...................................................................... 0
vusers.session_length:
  min: .............................................................................. 18386.4
  max: .............................................................................. 20077.9
  median: ........................................................................... 19346.7
  p95: .............................................................................. 20136.3
  p99: .............................................................................. 20136.3
```

Info No Bloqueante

```
-------------------------------------
Metrics for period to: 23:17:10(-0300) (width: 1.036s)
-------------------------------------

http.codes.200: ..................................................................... 39
http.request_rate: .................................................................. 48/sec
http.requests: ...................................................................... 49
http.response_time:
  min: .............................................................................. 1
  max: .............................................................................. 479
  median: ........................................................................... 22
  p95: .............................................................................. 175.9
  p99: .............................................................................. 175.9
http.responses: ..................................................................... 39
vusers.created: ..................................................................... 20
vusers.created_by_name.0: ........................................................... 20
```

Info Bloquante



Info No Bloqueante

Info Bloqueante

```
C:\Users\nicol\Coderhouse-Back-End-32080\utils>node autocannon.js
Running all benchmarks in parallel...
Running 20s test @ http://localhost:8080/infoBloq
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	746 ms	1008 ms	1751 ms	2017 ms	1048.29 ms	220.06 ms	2800 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	33	33	94	132	93.05	16.09	33
Bytes/Sec	50.3 kB	50.3 kB	143 kB	201 kB	142 kB	24.5 kB	50.3 kB

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

2k requests in 20.14s, 2.84 MB read
```

Info No Bloqueante

```
C:\Users\nicol\Coderhouse-Back-End-32080\utils>node autocannon.js
Running all benchmarks in parallel...
Running 20s test @ http://localhost:8080/infoNobloq
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	729 ms	1003 ms	1833 ms	2014 ms	1040.32 ms	235.12 ms	2992 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	45	45	95	110	92.5	11.53	45
Bytes/Sec	68.5 kB	68.5 kB	145 kB	168 kB	141 kB	17.6 kB	68.5 kB

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

2k requests in 20.13s, 2.82 MB read
```

Info Bloqueante

```
104
105          function getInfoProcessNoBloq(req, res) {
106   1.7 ms   info = {
107   1.0 ms     args: process.argv,
108   0.5 ms     cwd: process.cwd(),
109   0.5 ms     pid: process.pid,
110              version: process.version,
111   1.8 ms     title: process.title,
112   0.3 ms     os: process.platform,
113   0.8 ms     memoryUsage: process.memoryUsage().rss,
114              };
115   5.8 ms   res.render("pages/infoProcess.ejs", { info });
116          }
117
```

Info No Bloqueante

```
91           function getInfoProcessBloq(req, res) {
92    2.2 ms   info = {
93    1.0 ms     args: process.argv,
94    0.2 ms     cwd: process.cwd(),
95    0.3 ms     pid: process.pid,
96    0.3 ms     version: process.version,
97    2.4 ms     title: process.title,
98               os: process.platform,
99    1.5 ms     memoryUsage: process.memoryUsage().rss,
100              };
101   5.4 ms   console.log(info);
102  12.3 ms   res.render("pages/infoProcess.ejs", { info });
103   0.1 ms }
104
```

En Conclusión, se necesita de herramientas que permita mejorar la performance del servidor. Una de la primeras que se probó es la librería compress que nos dejo comprimir el tamaño de las respuestas de los endpoints y con eso disminuir un el tráfico de los datos.

Luego aplicamos varias herramientas de análisis de perfomance, llegando a la conclusión que tener console.log por nuestro código es algo feo si se habla de rendimiento del servidor, ya que al ser una operación síncrona, se le va a dar prioridad cuando se la ejecute retrasando las demás peticiones importantes. La solución que se planteó para esto es disponer de una librería para generar Logs, en este caso, log4js.