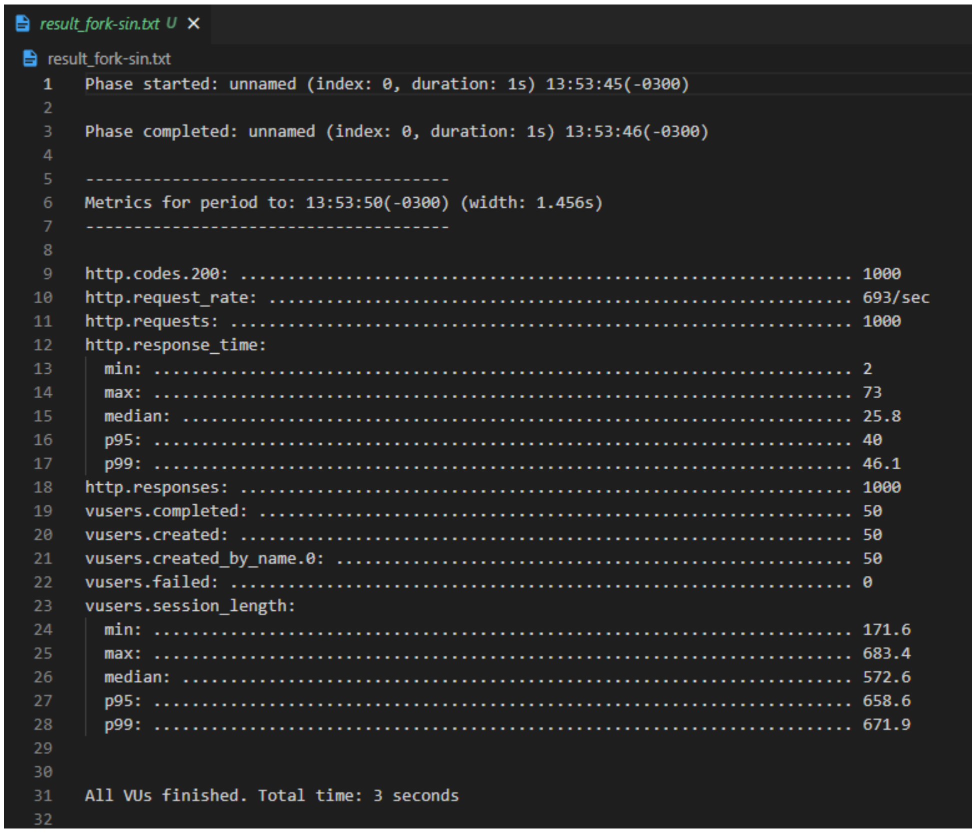
Informe

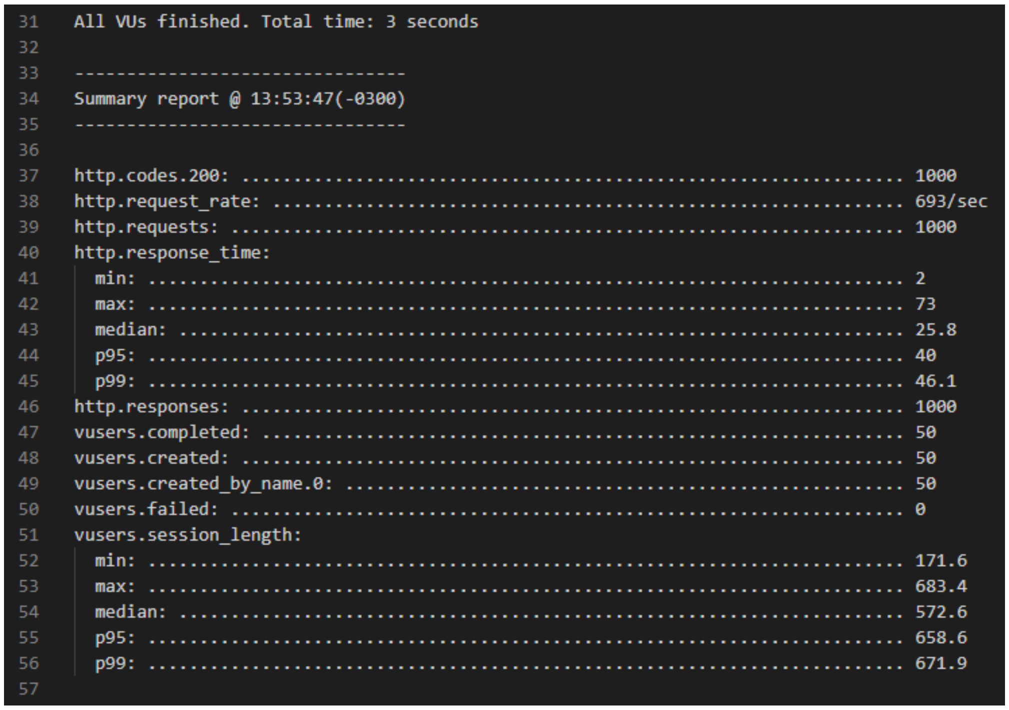
El primer test que realizamos fue con *artillery*, emulando un total de 50 conexiones con 20 *requests* en cada una de ellas. Lo hice sobre un *endpoint* que contiene (o no) un *console log* demandante, para ver la diferencia entre las respuestas. Para tal fin, utilicé el siguiente comando:

```
artillery quick --count 50 -n 20 http://localhost:8081/info-sin > result_fork-sin.txt
```

El resultado sobre la ruta sin console log se ve en las siguientes dos capturas.



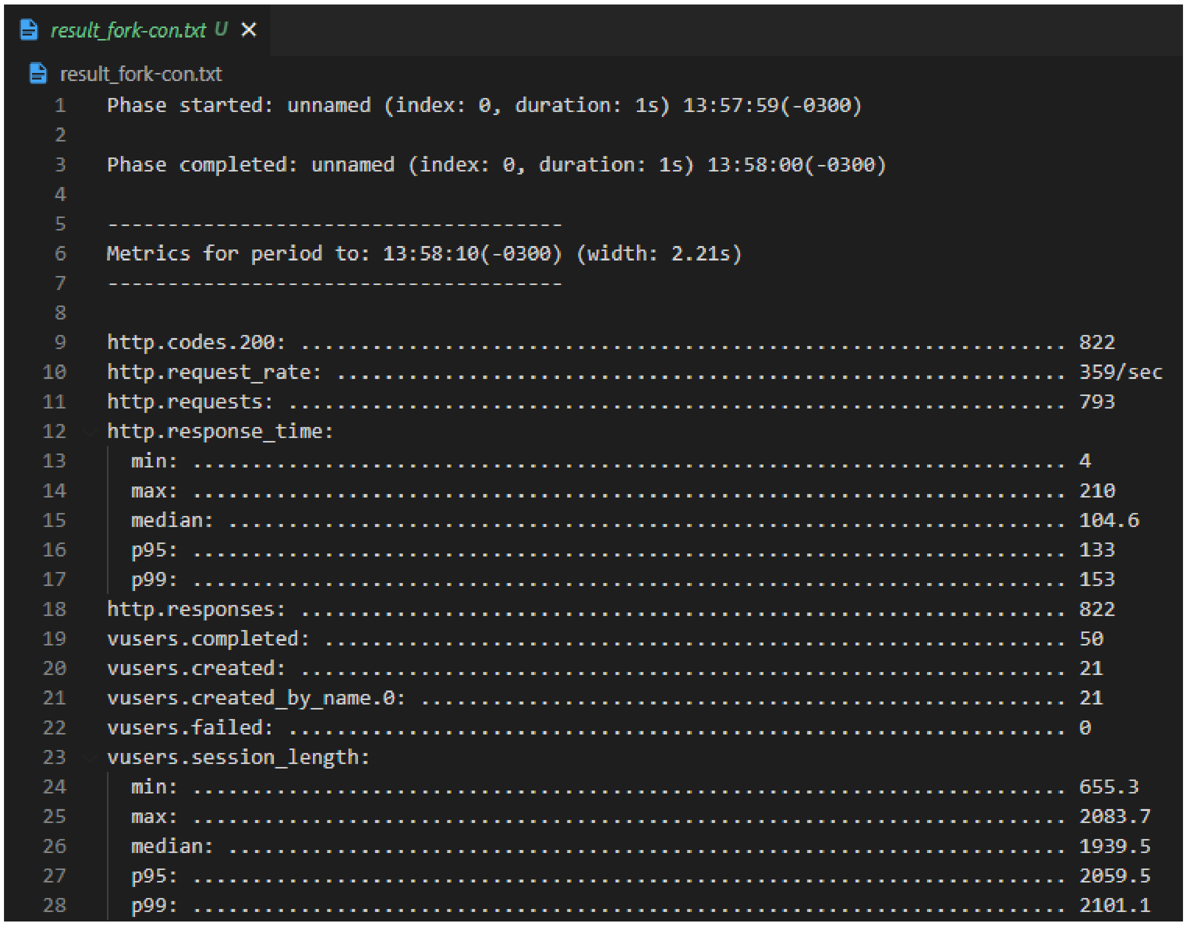
```
Phase started: unnamed (index: 0, duration: 1s) 13:53:45(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 13:53:46(-0300)

--------------------------------------
Metrics for period to: 13:53:50(-0300) (width: 1.456s)
--------------------------------------

http.codes.200: .............................................................. 1000
http.request_rate: ........................................................... 693/sec
http.requests: ............................................................... 1000
http.response_time:
  min: ....................................................................... 2
  max: ....................................................................... 73
  median: .................................................................... 25.8
  p95: ....................................................................... 40
  p99: ....................................................................... 46.1
http.responses: .............................................................. 1000
vusers.completed: ............................................................ 50
vusers.created: .............................................................. 50
vusers.created_by_name.0: .................................................... 50
vusers.failed: ............................................................... 0
vusers.session_length:
  min: ....................................................................... 171.6
  max: ....................................................................... 683.4
  median: .................................................................... 572.6
  p95: ....................................................................... 658.6
  p99: ....................................................................... 671.9


All VUs finished. Total time: 3 seconds
```



```
All VUs finished. Total time: 3 seconds

--------------------------------
Summary report @ 13:53:47(-0300)
--------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 693/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 2
  max: ......................................................................... 73
  median: ...................................................................... 25.8
  p95: ......................................................................... 40
  p99: ......................................................................... 46.1
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 171.6
  max: ......................................................................... 683.4
  median: ...................................................................... 572.6
  p95: ......................................................................... 658.6
  p99: ......................................................................... 671.9
```

Luego repetí el proceso para la ruta que sí tiene el console log en su flujo. El resultado se ve en las siguientes tres capturas.

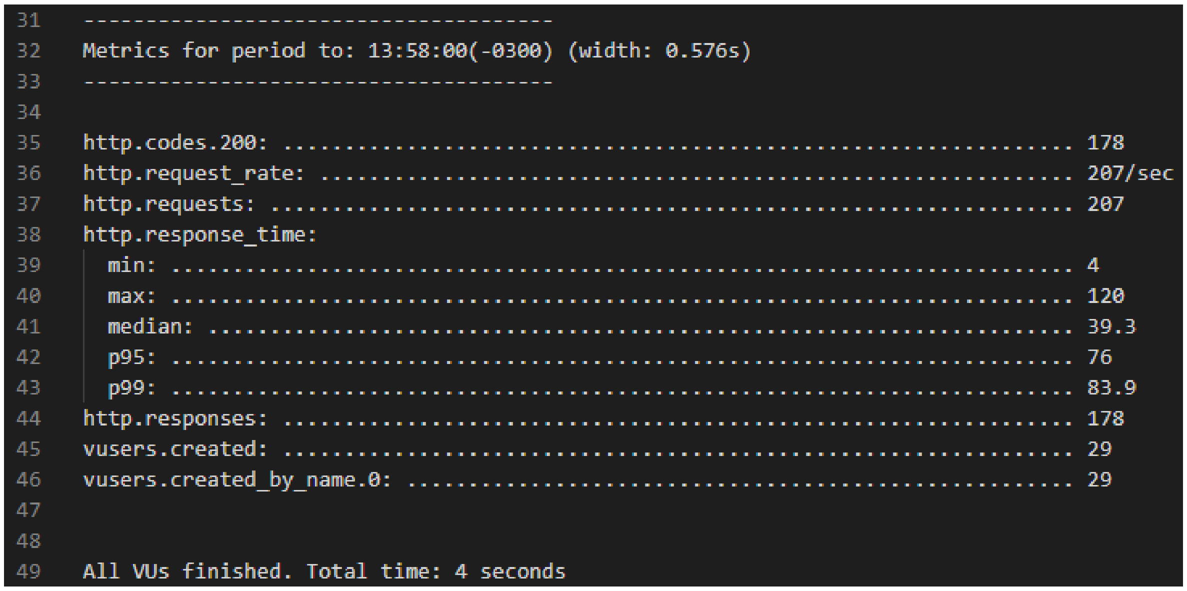
```
artillery quick --count 50 -n 20 http://localhost:8081/info-con > result_fork-con.txt
```



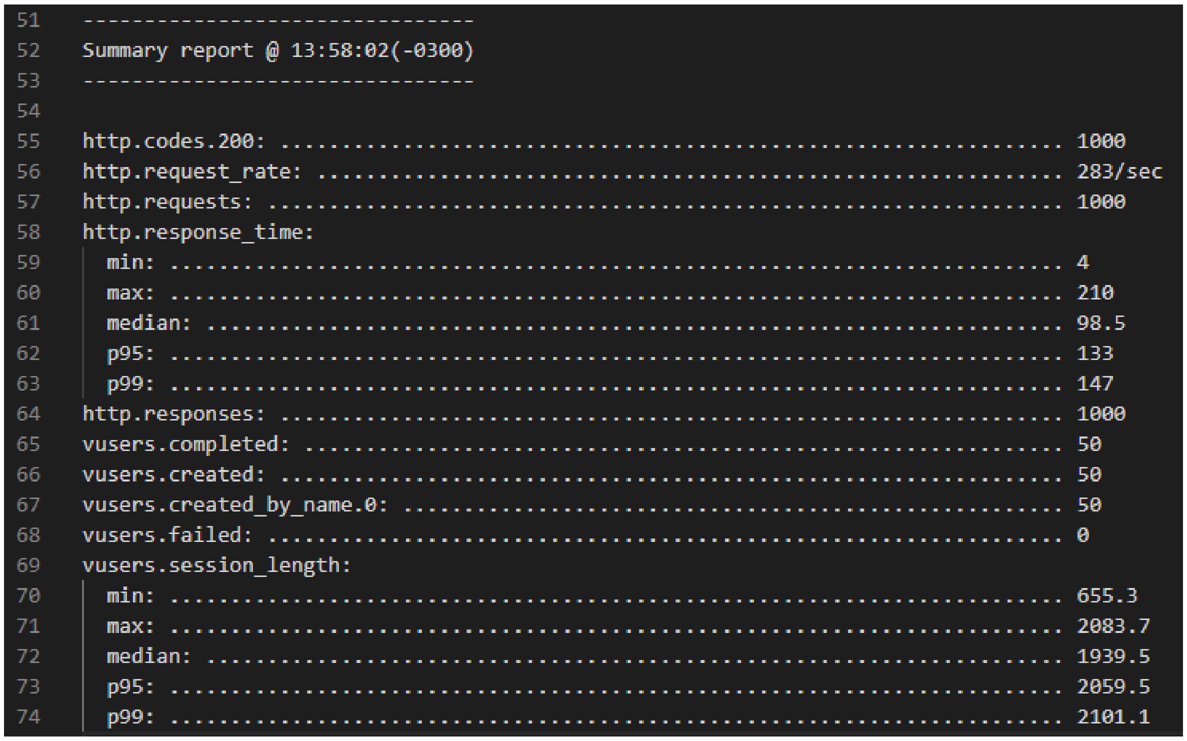
result_fork-con.txt U ×

result_fork-con.txt

```
Phase started: unnamed (index: 0, duration: 1s) 13:57:59(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 13:58:00(-0300)

--------------------------------------
Metrics for period to: 13:58:10(-0300) (width: 2.21s)
--------------------------------------

http.codes.200: ................................................................ 822
http.request_rate: ............................................................. 359/sec
http.requests: ................................................................. 793
http.response_time:
  min: ......................................................................... 4
  max: ......................................................................... 210
  median: ...................................................................... 104.6
  p95: ......................................................................... 133
  p99: ......................................................................... 153
http.responses: ................................................................ 822
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 21
vusers.created_by_name.0: ...................................................... 21
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 655.3
  max: ......................................................................... 2083.7
  median: ...................................................................... 1939.5
  p95: ......................................................................... 2059.5
  p99: ......................................................................... 2101.1
```



```
--------------------------------------
Metrics for period to: 13:58:00(-0300) (width: 0.576s)
--------------------------------------

http.codes.200: ................................................................ 178
http.request_rate: ............................................................. 207/sec
http.requests: ................................................................. 207
http.response_time:
  min: ......................................................................... 4
  max: ......................................................................... 120
  median: ...................................................................... 39.3
  p95: ......................................................................... 76
  p99: ......................................................................... 83.9
http.responses: ................................................................ 178
vusers.created: ................................................................ 29
vusers.created_by_name.0: ...................................................... 29


All VUs finished. Total time: 4 seconds
```



```
--------------------------------
Summary report @ 13:58:02(-0300)
--------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 283/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 4
  max: ......................................................................... 210
  median: ...................................................................... 98.5
  p95: ......................................................................... 133
  p99: ......................................................................... 147
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 655.3
  max: ......................................................................... 2083.7
  median: ...................................................................... 1939.5
  p95: ......................................................................... 2059.5
  p99: ......................................................................... 2101.1
```

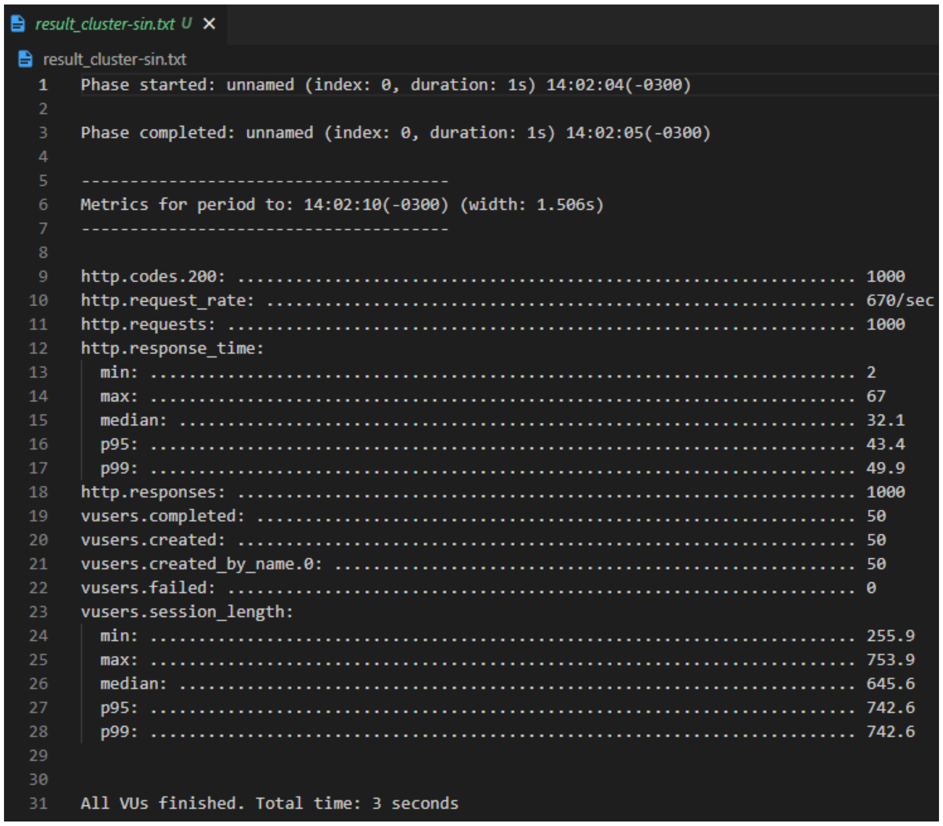
Si comparamos ambos resultados, vemos que el valor de ***median*** (valor que corresponde a la mediana en la distribución de milisegundos de latencia) es menor en la ruta sin console log (**25.8**) que en la ruta con console log (**98.5**). Estos valores los extraigo del *summary report* en ambos casos. Eso indica que el proceso sin console log es más eficiente. A su vez, el valor de ***request_rate*** fue de **693/s** para la ruta sin console log, y de **283/s** para la ruta con console log. Esto también me indica que la ruta sin console log es más eficiente, ya que este servidor despliega una mayor cantidad de respuestas por segundo.

Luego utilizamos el *Node built-in profiler*. Creamos el archivo isolate, necesario para el análisis. Corrimos nuevamente los tests con *artillery*, y se observaron resultados similares a los anteriores:

```
// NODE BUILT-IN PROFILER
// node --prof server.js
```

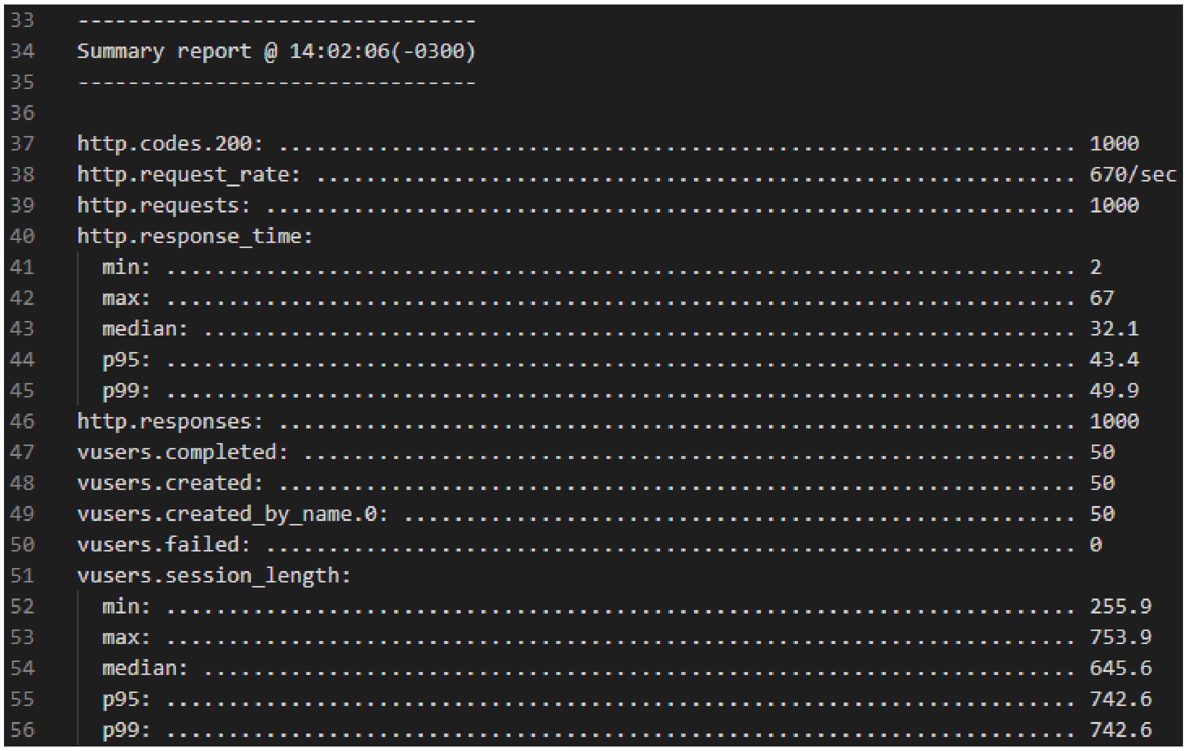
isolate-000001E04C... U

```
/ EN OTRA CONSOLA:
// artillery quick --count 50 -n 20 http://localhost:8081/info-sin > result_cluster-sin.txt
```


result_cluster-sin.txt U ×

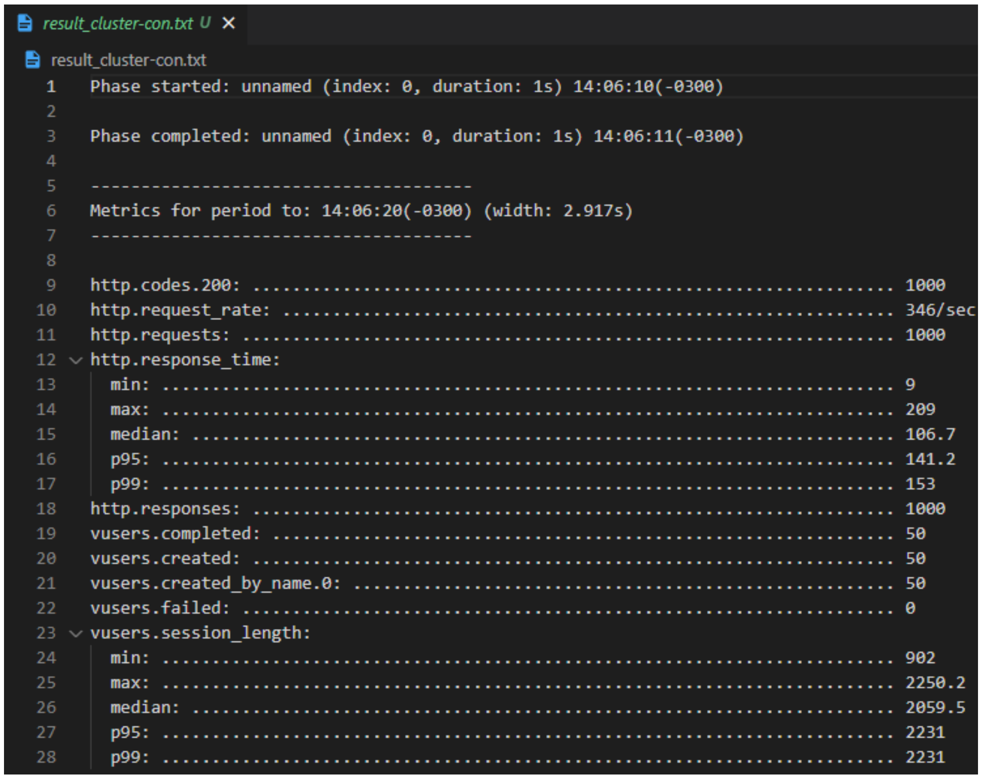
result_cluster-sin.txt

```
Phase started: unnamed (index: 0, duration: 1s) 14:02:04(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 14:02:05(-0300)

--------------------------------------
Metrics for period to: 14:02:10(-0300) (width: 1.506s)
--------------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 670/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 2
  max: ......................................................................... 67
  median: ...................................................................... 32.1
  p95: ......................................................................... 43.4
  p99: ......................................................................... 49.9
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 255.9
  max: ......................................................................... 753.9
  median: ...................................................................... 645.6
  p95: ......................................................................... 742.6
  p99: ......................................................................... 742.6


All VUs finished. Total time: 3 seconds
```



```
--------------------------------
Summary report @ 14:02:06(-0300)
--------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 670/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 2
  max: ......................................................................... 67
  median: ...................................................................... 32.1
  p95: ......................................................................... 43.4
  p99: ......................................................................... 49.9
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 255.9
  max: ......................................................................... 753.9
  median: ...................................................................... 645.6
  p95: ......................................................................... 742.6
  p99: ......................................................................... 742.6
```

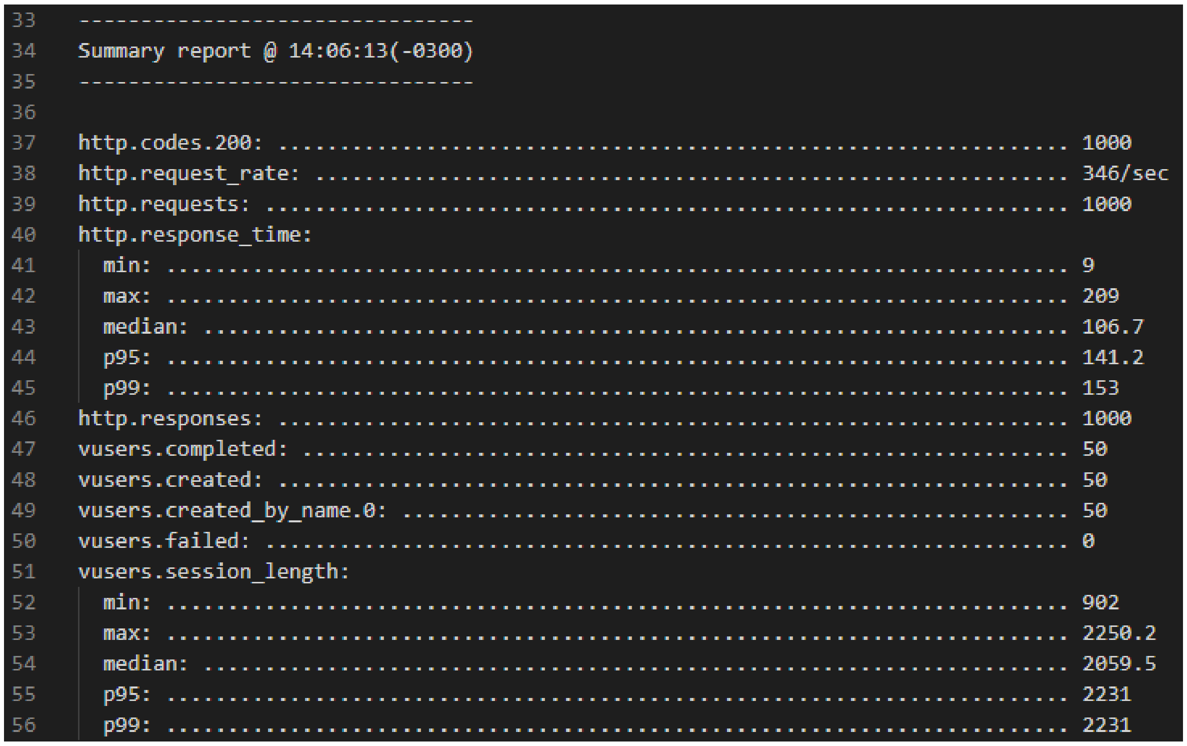
```
// EN OTRA CONSOLA:

// artillery quick --count 50 -n 20 http://localhost:8081/info-con > result_cluster-con.txt
```



result_cluster-con.txt U ×

result_cluster-con.txt

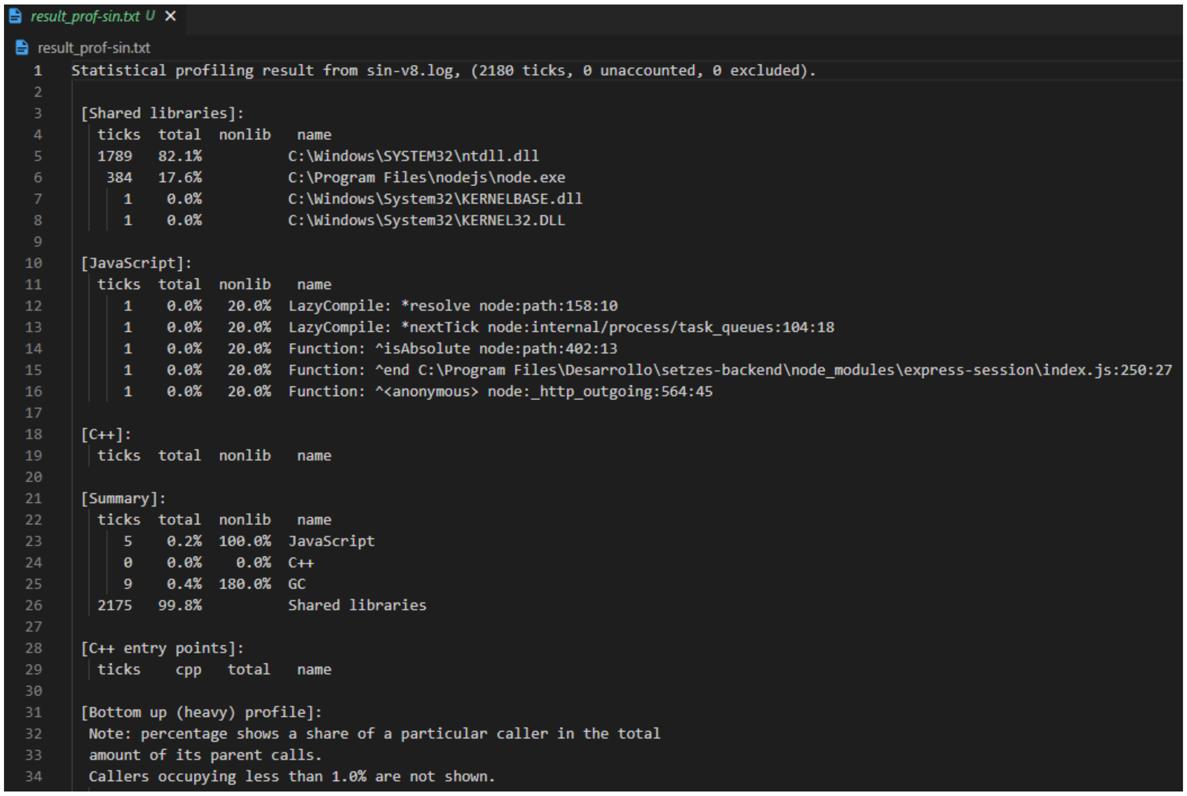
```
Phase started: unnamed (index: 0, duration: 1s) 14:06:10(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 14:06:11(-0300)

--------------------------------------
Metrics for period to: 14:06:20(-0300) (width: 2.917s)
--------------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 346/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 9
  max: ......................................................................... 209
  median: ...................................................................... 106.7
  p95: ......................................................................... 141.2
  p99: ......................................................................... 153
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 902
  max: ......................................................................... 2250.2
  median: ...................................................................... 2059.5
  p95: ......................................................................... 2231
  p99: ......................................................................... 2231
```



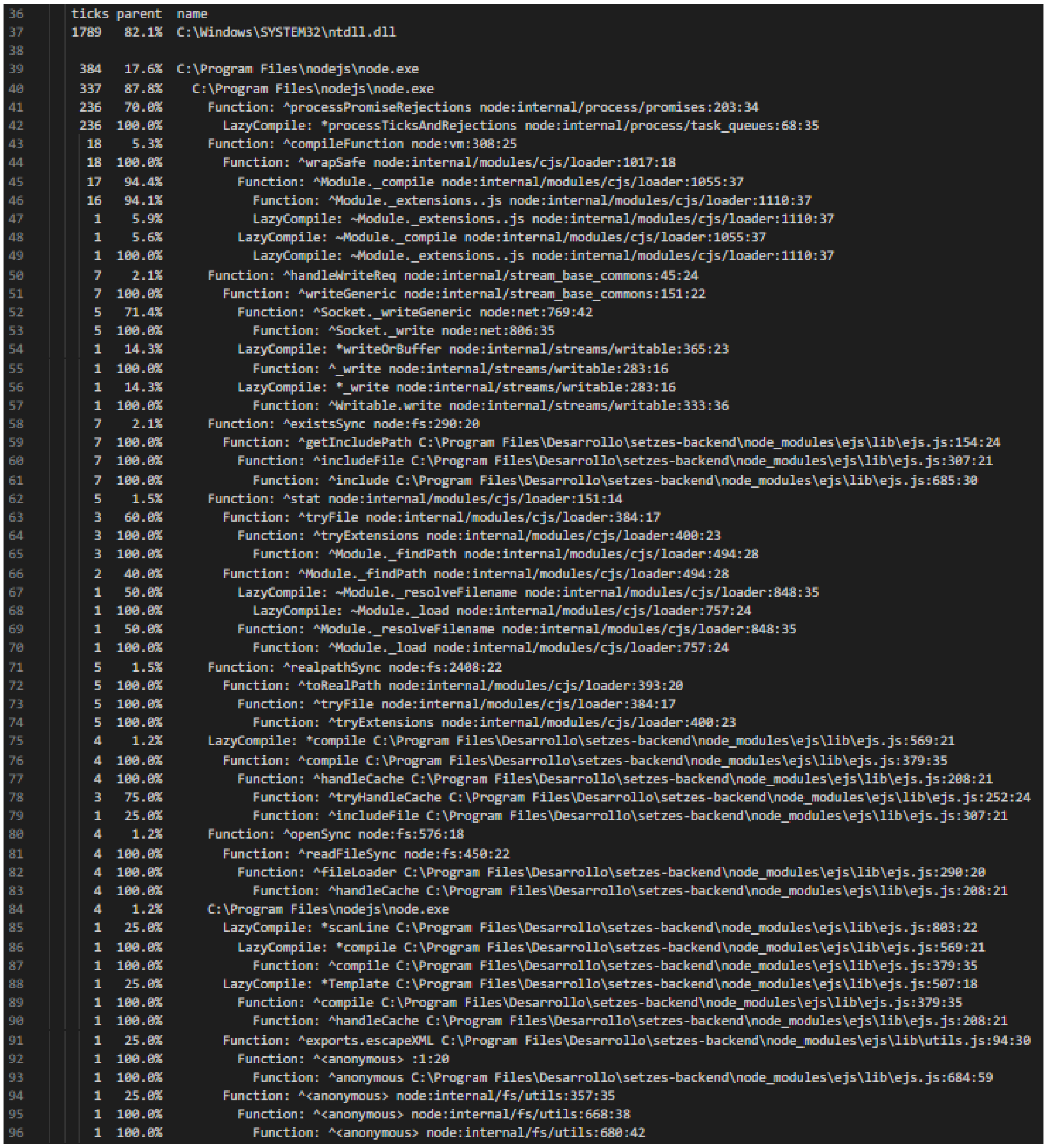
```
--------------------------------
Summary report @ 14:06:13(-0300)
--------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................. 346/sec
http.requests: ................................................................. 1000
http.response_time:
  min: ......................................................................... 9
  max: ......................................................................... 209
  median: ...................................................................... 106.7
  p95: ......................................................................... 141.2
  p99: ......................................................................... 153
http.responses: ................................................................ 1000
vusers.completed: .............................................................. 50
vusers.created: ................................................................ 50
vusers.created_by_name.0: ...................................................... 50
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ......................................................................... 902
  max: ......................................................................... 2250.2
  median: ...................................................................... 2059.5
  p95: ......................................................................... 2231
  p99: ......................................................................... 2231
```

Con el siguiente commando, logramos obtener un perfil estadístico de ambas respuestas. En el mismo, analizamos los ticks que presentó cada una de las respuestas:

```
// node --prof-process sin-v8.log > result_prof-sin.txt
```

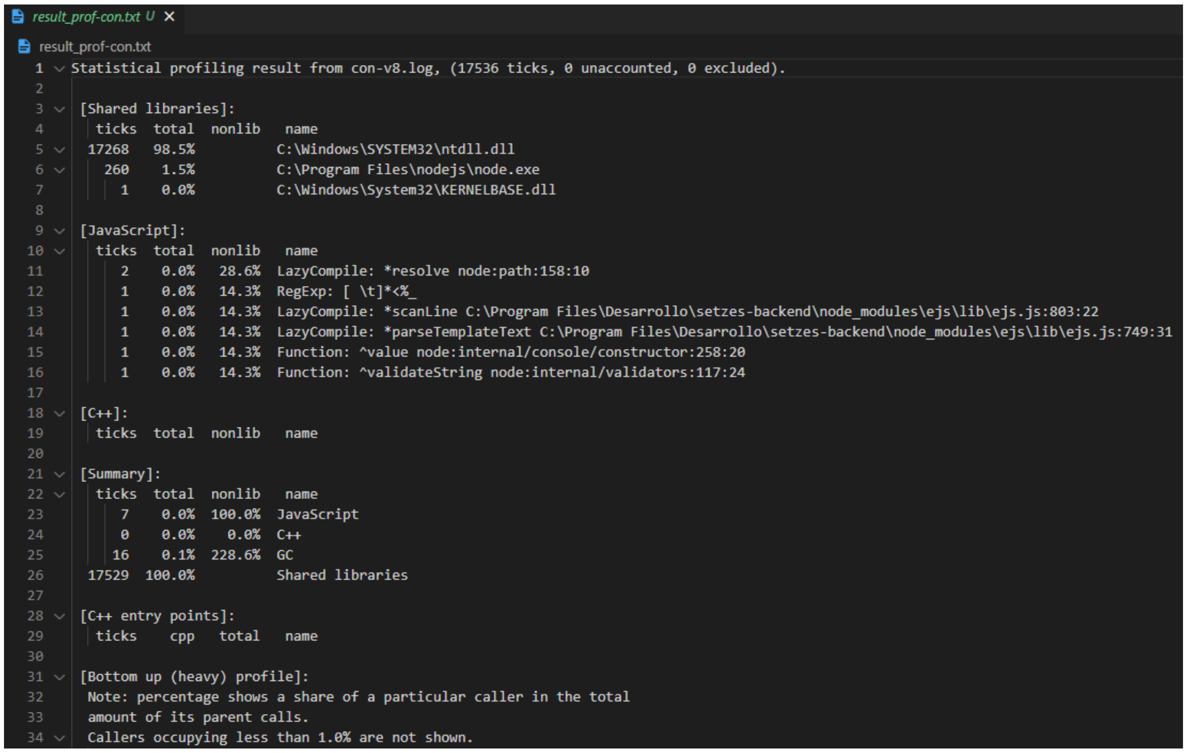


```
Statistical profiling result from sin-v8.log, (2180 ticks, 0 unaccounted, 0 excluded).

 [Shared libraries]:
   ticks  total  nonlib   name
   1789   82.1%          C:\Windows\SYSTEM32\ntdll.dll
    384   17.6%          C:\Program Files\nodejs\node.exe
      1    0.0%          C:\Windows\System32\KERNELBASE.dll
      1    0.0%          C:\Windows\System32\KERNEL32.DLL

 [JavaScript]:
   ticks  total  nonlib   name
      1    0.0%   20.0%  LazyCompile: *resolve node:path:158:10
      1    0.0%   20.0%  LazyCompile: *nextTick node:internal/process/task_queues:104:18
      1    0.0%   20.0%  Function: ^isAbsolute node:path:402:13
      1    0.0%   20.0%  Function: ^end C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:250:27
      1    0.0%   20.0%  Function: ^<anonymous> node:_http_outgoing:564:45

 [C++]:
   ticks  total  nonlib   name

 [Summary]:
   ticks  total  nonlib   name
      5    0.2%  100.0%  JavaScript
      0    0.0%    0.0%  C++
      9    0.4%  180.0%  GC
   2175   99.8%          Shared libraries

 [C++ entry points]:
   ticks    cpp   total   name

 [Bottom up (heavy) profile]:
  Note: percentage shows a share of a particular caller in the total
  amount of its parent calls.
  Callers occupying less than 1.0% are not shown.
```



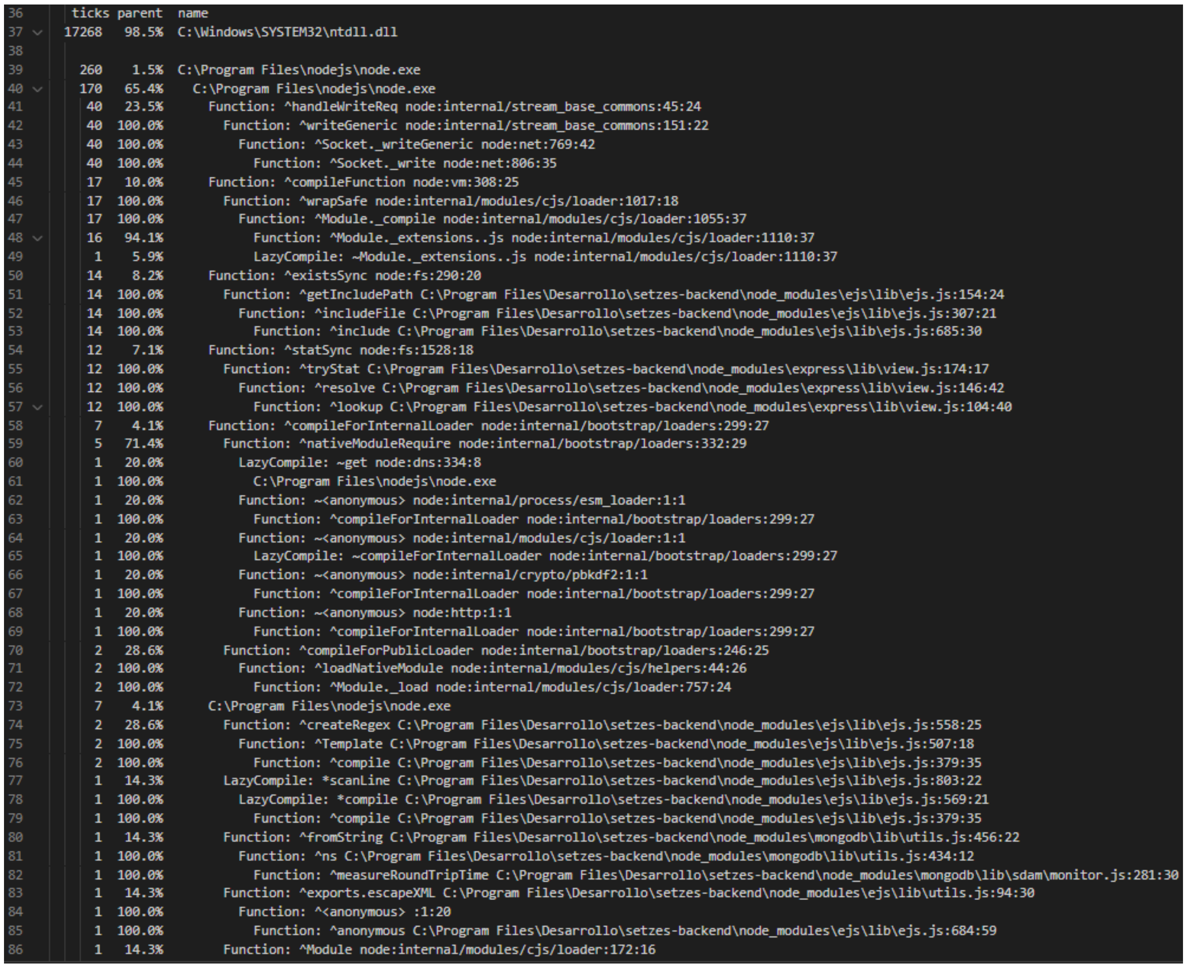
```
 ticks parent  name
 1789   82.1%  C:\Windows\SYSTEM32\ntdll.dll

  384   17.6%  C:\Program Files\nodejs\node.exe
  337   87.8%    C:\Program Files\nodejs\node.exe
  236   70.0%      Function: ^processPromiseRejections node:internal/process/promises:203:34
  236  100.0%        LazyCompile: *processTicksAndRejections node:internal/process/task_queues:68:35
   18    5.3%      Function: ^compileFunction node:vm:308:25
   18  100.0%        Function: ^wrapSafe node:internal/modules/cjs/loader:1017:18
   17   94.4%          Function: ^Module._compile node:internal/modules/cjs/loader:1055:37
   16   94.1%            Function: ^Module._extensions..js node:internal/modules/cjs/loader:1110:37
    1    5.9%            LazyCompile: ~Module._extensions..js node:internal/modules/cjs/loader:1110:37
    1    5.6%          LazyCompile: ~Module._compile node:internal/modules/cjs/loader:1055:37
    1  100.0%            LazyCompile: ~Module._extensions..js node:internal/modules/cjs/loader:1110:37
    7    2.1%      Function: ^handleWriteReq node:internal/stream_base_commons:45:24
    7  100.0%        Function: ^writeGeneric node:internal/stream_base_commons:151:22
    5   71.4%          Function: ^Socket._writeGeneric node:net:769:42
    5  100.0%            Function: ^Socket._write node:net:806:35
    1   14.3%          LazyCompile: *writeOrBuffer node:internal/streams/writable:365:23
    1  100.0%            Function: ^_write node:internal/streams/writable:283:16
    1   14.3%          LazyCompile: *_write node:internal/streams/writable:283:16
    1  100.0%            Function: ^Writable.write node:internal/streams/writable:333:36
    7    2.1%      Function: ^existsSync node:fs:290:20
    7  100.0%        Function: ^getIncludePath C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:154:24
    7  100.0%          Function: ^includeFile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:307:21
    7  100.0%            Function: ^include C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:685:30
    5    1.5%      Function: ^stat node:internal/modules/cjs/loader:151:14
    3   60.0%        Function: ^tryFile node:internal/modules/cjs/loader:384:17
    3  100.0%          Function: ^tryExtensions node:internal/modules/cjs/loader:400:23
    3  100.0%            Function: ^Module._findPath node:internal/modules/cjs/loader:494:28
    2   40.0%        Function: ^Module._findPath node:internal/modules/cjs/loader:494:28
    1   50.0%          LazyCompile: ~Module._resolveFilename node:internal/modules/cjs/loader:848:35
    1  100.0%            LazyCompile: ~Module._load node:internal/modules/cjs/loader:757:24
    1   50.0%          Function: ^Module._resolveFilename node:internal/modules/cjs/loader:848:35
    1  100.0%            Function: ^Module._load node:internal/modules/cjs/loader:757:24
    5    1.5%      Function: ^realpathSync node:fs:2408:22
    5  100.0%        Function: ^toRealPath node:internal/modules/cjs/loader:393:20
    5  100.0%          Function: ^tryFile node:internal/modules/cjs/loader:384:17
    5  100.0%            Function: ^tryExtensions node:internal/modules/cjs/loader:400:23
    4    1.2%      LazyCompile: *compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:569:21
    4  100.0%        Function: ^compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:379:35
    4  100.0%          Function: ^handleCache C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:208:21
    3   75.0%            Function: ^tryHandleCache C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:252:24
    1   25.0%            Function: ^includeFile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:307:21
    4    1.2%      Function: ^openSync node:fs:576:18
    4  100.0%        Function: ^readFileSync node:fs:450:22
    4  100.0%          Function: ^fileLoader C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:290:20
    4  100.0%            Function: ^handleCache C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:208:21
    4    1.2%      C:\Program Files\nodejs\node.exe
    1   25.0%        LazyCompile: *scanLine C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:803:22
    1  100.0%          LazyCompile: *compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:569:21
    1  100.0%            Function: ^compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:379:35
    1   25.0%        LazyCompile: *Template C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:507:18
    1  100.0%          Function: ^compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:379:35
    1  100.0%            Function: ^handleCache C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:208:21
    1   25.0%        Function: ^exports.escapeXML C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\utils.js:94:30
    1  100.0%          Function: ^<anonymous> :1:20
    1  100.0%            Function: ^anonymous C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:684:59
    1   25.0%        Function: ^<anonymous> node:internal/fs/utils:357:35
    1  100.0%          Function: ^<anonymous> node:internal/fs/utils:668:38
    1  100.0%            Function: ^<anonymous> node:internal/fs/utils:680:42
```

```
node --prof-process con-v8.log > result_prof-con.txt
```

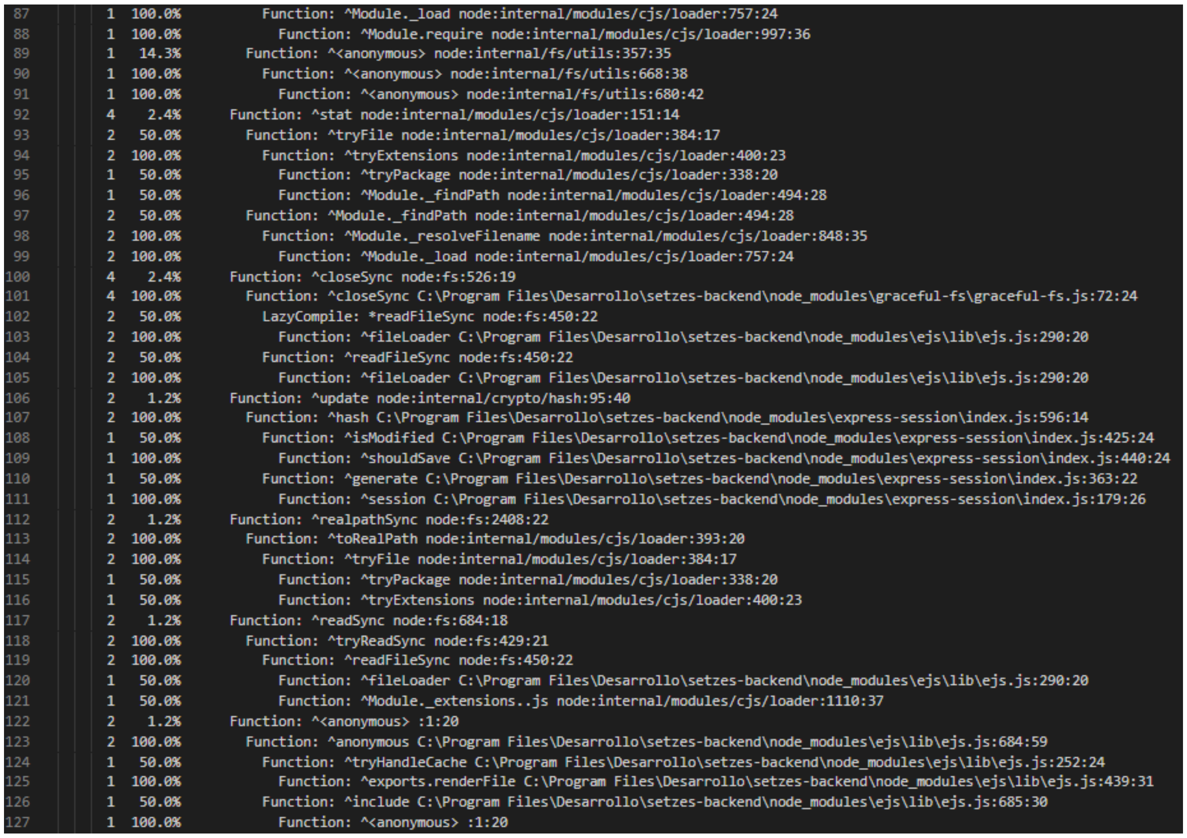

result_prof-con.txt U ×

result_prof-con.txt

```
Statistical profiling result from con-v8.log, (17536 ticks, 0 unaccounted, 0 excluded).

 [Shared libraries]:
   ticks  total  nonlib   name
  17268   98.5%          C:\Windows\SYSTEM32\ntdll.dll
    260    1.5%          C:\Program Files\nodejs\node.exe
      1    0.0%          C:\Windows\System32\KERNELBASE.dll

 [JavaScript]:
   ticks  total  nonlib   name
      2    0.0%   28.6%  LazyCompile: *resolve node:path:158:10
      1    0.0%   14.3%  RegExp: [ \t]*<%_
      1    0.0%   14.3%  LazyCompile: *scanLine C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:803:22
      1    0.0%   14.3%  LazyCompile: *parseTemplateText C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:749:31
      1    0.0%   14.3%  Function: ^value node:internal/console/constructor:258:20
      1    0.0%   14.3%  Function: ^validateString node:internal/validators:117:24

 [C++]:
   ticks  total  nonlib   name

 [Summary]:
   ticks  total  nonlib   name
      7    0.0%  100.0%  JavaScript
      0    0.0%    0.0%  C++
     16    0.1%  228.6%  GC
  17529  100.0%          Shared libraries

 [C++ entry points]:
   ticks    cpp   total   name

 [Bottom up (heavy) profile]:
  Note: percentage shows a share of a particular caller in the total
  amount of its parent calls.
  Callers occupying less than 1.0% are not shown.
```



```
   ticks parent  name
  17268   98.5%  C:\Windows\SYSTEM32\ntdll.dll

    260    1.5%  C:\Program Files\nodejs\node.exe
    170   65.4%    C:\Program Files\nodejs\node.exe
     40   23.5%      Function: ^handleWriteReq node:internal/stream_base_commons:45:24
     40  100.0%        Function: ^writeGeneric node:internal/stream_base_commons:151:22
     40  100.0%          Function: ^Socket._writeGeneric node:net:769:42
     40  100.0%            Function: ^Socket._write node:net:806:35
     17   10.0%      Function: ^compileFunction node:vm:308:25
     17  100.0%        Function: ^wrapSafe node:internal/modules/cjs/loader:1017:18
     17  100.0%          Function: ^Module._compile node:internal/modules/cjs/loader:1055:37
     16   94.1%            Function: ^Module._extensions..js node:internal/modules/cjs/loader:1110:37
      1    5.9%            LazyCompile: ~Module._extensions..js node:internal/modules/cjs/loader:1110:37
     14    8.2%      Function: ^existsSync node:fs:290:20
     14  100.0%        Function: ^getIncludePath C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:154:24
     14  100.0%          Function: ^includeFile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:307:21
     14  100.0%            Function: ^include C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:685:30
     12    7.1%      Function: ^statSync node:fs:1528:18
     12  100.0%        Function: ^tryStat C:\Program Files\Desarrollo\setzes-backend\node_modules\express\lib\view.js:174:17
     12  100.0%          Function: ^resolve C:\Program Files\Desarrollo\setzes-backend\node_modules\express\lib\view.js:146:42
     12  100.0%            Function: ^lookup C:\Program Files\Desarrollo\setzes-backend\node_modules\express\lib\view.js:104:40
      7    4.1%      Function: ^compileForInternalLoader node:internal/bootstrap/loaders:299:27
      5   71.4%        Function: ^nativeModuleRequire node:internal/bootstrap/loaders:332:29
      1   20.0%          LazyCompile: ~get node:dns:334:8
      1  100.0%            C:\Program Files\nodejs\node.exe
      1   20.0%          Function: ~<anonymous> node:internal/process/esm_loader:1:1
      1  100.0%            Function: ^compileForInternalLoader node:internal/bootstrap/loaders:299:27
      1   20.0%          Function: ~<anonymous> node:internal/modules/cjs/loader:1:1
      1  100.0%            LazyCompile: ~compileForInternalLoader node:internal/bootstrap/loaders:299:27
      1   20.0%          Function: ~<anonymous> node:internal/crypto/pbkdf2:1:1
      1  100.0%            Function: ^compileForInternalLoader node:internal/bootstrap/loaders:299:27
      1   20.0%          Function: ~<anonymous> node:http:1:1
      1  100.0%            Function: ^compileForInternalLoader node:internal/bootstrap/loaders:299:27
      2   28.6%        Function: ^compileForPublicLoader node:internal/bootstrap/loaders:246:25
      2  100.0%          Function: ^loadNativeModule node:internal/modules/cjs/helpers:44:26
      2  100.0%            Function: ^Module._load node:internal/modules/cjs/loader:757:24
      7    4.1%      C:\Program Files\nodejs\node.exe
      2   28.6%        Function: ^createRegex C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:558:25
      2  100.0%          Function: ^Template C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:507:18
      2  100.0%            Function: ^compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:379:35
      1   14.3%        LazyCompile: *scanLine C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:803:22
      1  100.0%          LazyCompile: *compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:569:21
      1  100.0%            Function: ^compile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:379:35
      1   14.3%        Function: ^fromString C:\Program Files\Desarrollo\setzes-backend\node_modules\mongodb\lib\utils.js:456:22
      1  100.0%          Function: ^ns C:\Program Files\Desarrollo\setzes-backend\node_modules\mongodb\lib\utils.js:434:12
      1  100.0%            Function: ^measureRoundTripTime C:\Program Files\Desarrollo\setzes-backend\node_modules\mongodb\lib\sdam\monitor.js:281:30
      1   14.3%        Function: ^exports.escapeXML C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\utils.js:94:30
      1  100.0%          Function: ^<anonymous> :1:20
      1  100.0%            Function: ^anonymous C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:684:59
      1   14.3%        Function: ^Module node:internal/modules/cjs/loader:172:16
```

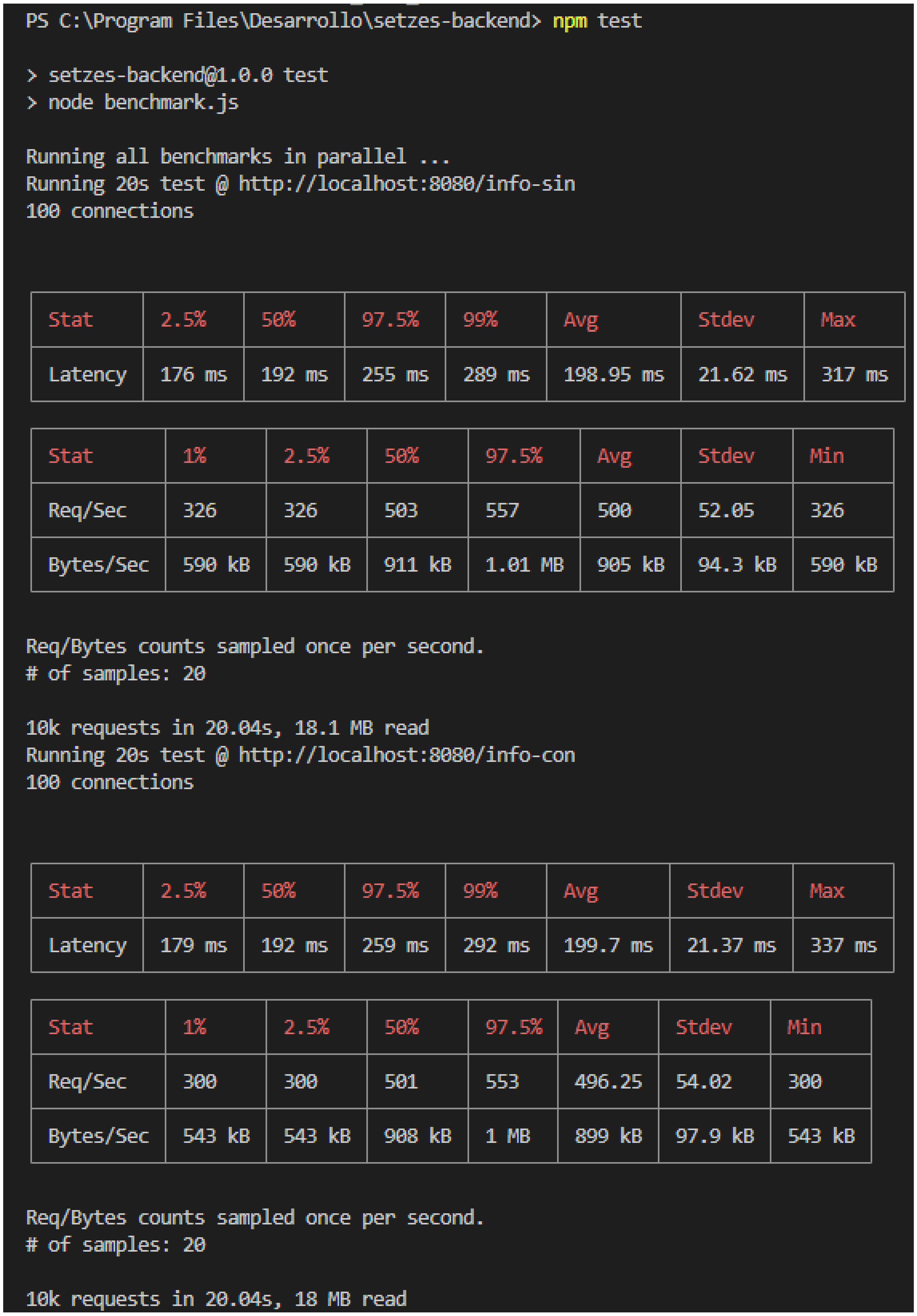


```
      1  100.0%          Function: ^Module._load node:internal/modules/cjs/loader:757:24
      1  100.0%            Function: ^Module.require node:internal/modules/cjs/loader:997:36
      1   14.3%        Function: ^<anonymous> node:internal/fs/utils:357:35
      1  100.0%          Function: ^<anonymous> node:internal/fs/utils:668:38
      1  100.0%            Function: ^<anonymous> node:internal/fs/utils:680:42
      4    2.4%      Function: ^stat node:internal/modules/cjs/loader:151:14
      2   50.0%        Function: ^tryFile node:internal/modules/cjs/loader:384:17
      2  100.0%          Function: ^tryExtensions node:internal/modules/cjs/loader:400:23
      1   50.0%            Function: ^tryPackage node:internal/modules/cjs/loader:338:20
      1   50.0%            Function: ^Module._findPath node:internal/modules/cjs/loader:494:28
      2   50.0%        Function: ^Module._findPath node:internal/modules/cjs/loader:494:28
      2  100.0%          Function: ^Module._resolveFilename node:internal/modules/cjs/loader:848:35
      2  100.0%            Function: ^Module._load node:internal/modules/cjs/loader:757:24
      4    2.4%      Function: ^closeSync node:fs:526:19
      4  100.0%        Function: ^closeSync C:\Program Files\Desarrollo\setzes-backend\node_modules\graceful-fs\graceful-fs.js:72:24
      2   50.0%          LazyCompile: *readFileSync node:fs:450:22
      2  100.0%            Function: ^fileLoader C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:290:20
      2   50.0%          Function: ^readFileSync node:fs:450:22
      2  100.0%            Function: ^fileLoader C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:290:20
      2    1.2%      Function: ^update node:internal/crypto/hash:95:40
      2  100.0%        Function: ^hash C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:596:14
      1   50.0%          Function: ^isModified C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:425:24
      1  100.0%            Function: ^shouldSave C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:440:24
      1   50.0%          Function: ^generate C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:363:22
      1  100.0%            Function: ^session C:\Program Files\Desarrollo\setzes-backend\node_modules\express-session\index.js:179:26
      2    1.2%      Function: ^realpathSync node:fs:2408:22
      2  100.0%        Function: ^toRealPath node:internal/modules/cjs/loader:393:20
      2  100.0%          Function: ^tryFile node:internal/modules/cjs/loader:384:17
      1   50.0%            Function: ^tryPackage node:internal/modules/cjs/loader:338:20
      1   50.0%            Function: ^tryExtensions node:internal/modules/cjs/loader:400:23
      2    1.2%      Function: ^readSync node:fs:684:18
      2  100.0%        Function: ^tryReadSync node:fs:429:21
      2  100.0%          Function: ^readFileSync node:fs:450:22
      1   50.0%            Function: ^fileLoader C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:290:20
      1   50.0%            Function: ^Module._extensions..js node:internal/modules/cjs/loader:1110:37
      2    1.2%      Function: ^<anonymous> :1:20
      2  100.0%        Function: ^anonymous C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:684:59
      1   50.0%          Function: ^tryHandleCache C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:252:24
      1  100.0%            Function: ^exports.renderFile C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:439:31
      1   50.0%          Function: ^include C:\Program Files\Desarrollo\setzes-backend\node_modules\ejs\lib\ejs.js:685:30
      1  100.0%            Function: ^<anonymous> :1:20
```

Como se aprecia en las capturas, la cantidad de *ticks* es mucho mayor en la ruta que sí tiene el console log (17529) en relación a la ruta que no lo tiene (2175). Esta información la extraigo del *summary report*, que se observa en las capturas para ambos casos. La menor cantidad de ticks en el proceso más "liviano" sigue la línea de lo que veníamos viendo en los procesos anteriores.

Por último, pruebo con *Autocannon*, una dependencia similar a *Artillery*. También usamos 0x, para perfilar y agregar más información al análisis. Utilizo un archivo llamado *benchmark js* para realizar el test, y modifico el *package json* de manera acorde.

```
// MODIFICO package.json //


// "scripts": {
//   "test": "node benchmark.js",
//   "start": "0x server.js"
// }

// npm start

// EN OTRA CONSOLA:

// npm test
```



```
PS C:\Program Files\Desarrollo\setzes-backend> npm test

> setzes-backend@1.0.0 test
> node benchmark.js

Running all benchmarks in parallel ...
Running 20s test @ http://localhost:8080/info-sin
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	176 ms	192 ms	255 ms	289 ms	198.95 ms	21.62 ms	317 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	326	326	503	557	500	52.05	326
Bytes/Sec	590 kB	590 kB	911 kB	1.01 MB	905 kB	94.3 kB	590 kB

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

10k requests in 20.04s, 18.1 MB read
Running 20s test @ http://localhost:8080/info-con
100 connections
```

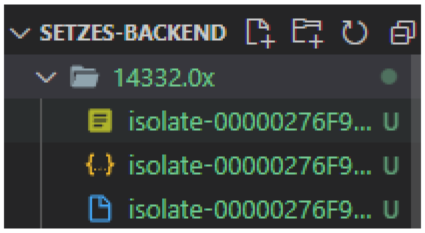
Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	179 ms	192 ms	259 ms	292 ms	199.7 ms	21.37 ms	337 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	300	300	501	553	496.25	54.02	300
Bytes/Sec	543 kB	543 kB	908 kB	1 MB	899 kB	97.9 kB	543 kB

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

10k requests in 20.04s, 18 MB read
```

Nuevamente vemos que el tiempo promedio de latencia en milisegundos es mayor en el proceso con console log (199.7 ms) que en el proceso sin (198.95 ms). A su vez, la cantidad de respuestas por segundo es mayor en el proceso más liviano (500 en promedio) que en el proceso más pesado (496.25). Esto sigue la línea de lo observado anteriormente.

Mediante 0x, se generan estos tres archivos en una carpeta con nombre aleatorio:



Uno de ellos debería permitir el análisis del diagrama de flama en el navegador, pero yo no fui capaz de hacerlo con ninguno de ellos. Desconozco si es un problema de mi navegador o del proceso, pero no fui capaz de visualizarlo.

Conclusión

El proceso más liviano sin console log exhibió en todos los casos, en promedio, 1) un mayor número de respuestas por segundo, 2) un menor número de ticks y 3) un menor número de latencia. Esto es coherente con lo anticipado previo a la realización de todos los *tests*.