

En primer lugar, incorporamos un gzip al proyecto

```
//const { fork } = require("child_process");
const isAuth = require ('../middlewares/isAuth.js');
const util = require("util");
const compression = require ("compression");
const logger = require ("../logger.js");

module.exports = function(passport){
  routes.get('/info',compression(), (req,res)=>{util
    res.json(
      `Titulo del proceso: ${process.title}
      Sistema operativo: ${process.platform}
      Version de Node: ${process.version}
      Memoria total reservada: ${util.inspect(process.memoryUsage(), {
        showHidden: false,
        depth: null,
        colors: true})}
      Path de ejecución: ${util.inspect(process.execPath)}
      Process id: ${process.pid}
      Carpeta del proyecto: ${process.cwd()}
      Procesadores presentes: ${process.pid}`
    )});
  routes.get('/', isAuth,(req,res)=>{
    res.sendFile(path.join(__dirname, "../public/home.html"))
  })
  routes.get('/login',(req, res)=>{
    if(req.isAuthenticated()){
      res.redirect('/')
    }else{
      res.sendFile(path.join(__dirname, "../public/login.html"));
    }
  })
}
```

En el proyecto lo deje en el final del
export, y comente el child_process

Comprobamos su uso en la ruta /info

Sin comprimir:

The screenshot shows the Chrome DevTools Network tab with the 'Network' panel selected. The left sidebar displays the process information and the console output. The main panel shows a list of network requests. The 'info' request is selected, and its details are shown in the right pane. The 'Waterfall' view shows the request timeline. The status bar at the bottom indicates that 2 requests were made, with a total of 961 B transferred and 589 B of resources. The 'Finish' time is 393 ms, 'DOMContentLoaded' is 331 ms, and 'Load' is 324 ms.

Titulo del proceso: C:\WINDOWS\system32\cmd.exe \n Sistema operativo: win32 \n Version de Node: v16.16.0 \n Memoria total reservada: { \n rss: \n \u001b[33m65028096\u001b[39m, \n heapTotal: \n \u001b[33m38445056\u001b[39m, \n heapUsed: \n \u001b[33m26301872\u001b[39m, \n external: \n \u001b[33m24238150\u001b[39m, \n arrayBuffers: \n \u001b[33m18768991\u001b[39m \n \n Path de ejecuci\u00f3n: 'C:\\\\Program Files\\\\nodejs\\\\node.exe' \n Process id: 5956 \n Carpeta del proyecto: C:\\xampp\\htdocs\\Coder\\SegundaEntrega \n Procesadores presentes: 5956"

Elements Console Sources Network Performance >>

Filter ☐ Invert ☐ Hide data URLs

All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other ☐ Has blocked cookies

☐ Blocked Requests ☐ 3rd-party requests

50 ms 100 ms 150 ms 200 ms 250 ms 300 ms 350 ms 400 ms

Name	Status	Type	Initiator	Size	Time	Waterfall
info	200	docum...	Other	783 B	69 ms	
favicon.ico	304	text/html	Other	178 B	22 ms	

2 requests 961 B transferred 589 B resources Finish: 393 ms DOMContentLoaded: 331 ms Load: 324 ms

Comprimido:

The screenshot shows the Chrome DevTools Network tab. The left sidebar displays the process title and memory usage. The main panel shows the Network tab with a filter set to 'All'. A single request is visible in the list, with a status of 200 and a size of 808 B. The waterfall view shows the request taking 11 ms. The bottom status bar indicates that the resource is compressed (808 B / 986 B transferred) and that the DOMContentLoaded event occurred 268 ms after the request.

Process Title: C:\WINDOWS\system32\cmd.exe \n Sistema operativo: win32 \n Version de Node: v16.16.0 \n Memoria total reservada: \n rss: \n \u001b[33m55767040\u001b[39m \n heapTotal: \n \u001b[33m25075712\u001b[39m \n heapUsed: \n \u001b[33m23498728\u001b[39m \n external: \n \u001b[33m19363185\u001b[39m \n arrayBuffers: \n \u001b[33m18345032\u001b[39m \n Path de ejecuci\u00f3n: 'C:\\\\Program Files\\\\nodejs\\\\node.exe' \n Process id: 11088 \n Carpeta del proyecto: C:\\xampp\\htdocs\\coder\\SegundaEntrega \n Procesadores presentes: 11088"

DevTools is now available in Spanish!

Always match Chrome's language Switch DevTools to Spanish Don't show again

Elements Console Sources Network Performance Memory »

Preserve log Disable cache No throttling

Filter ☒ Invert ☒ Hide data URLs

All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other ☐ Has blocked cookies

☐ Blocked Requests ☐ 3rd-party requests

Name	Status	Type	Initiator	Size	Time	Waterfall
info	200	docum...	Other	808 B	11 ms	

1 / 2 requests 808 B / 986 B transferred 548 B / 591 B resources Finish: 315 ms DOMContentLoaded: 268 ms

Por alguna raz\u00f3n el archivo incremento su peso. Prob\u00e9 anteriormente con otras rutas y funcion\u00f3 adecuadamente

Implementado el loggeo con Winston

```
JS logger.js > buildProdLogger
1  const winston = require("winston");
2  require("dotenv").config()
3
4  const buildProdLogger = () => {
5    const prodLogger = winston.createLogger({
6      transports: [
7        new winston.transports.File({ filename: "warn.log", level: "warn" }),
8        new winston.transports.File({ filename: "error.log", level: "error" }),
9      ],
10   });
11
12   return prodLogger;
13 };
14
15 const buildDevLogger = () => {
16   const devLogger = winston.createLogger({
17     transports: [new winston.transports.Console({ level: "info" })],
18   });
19
20   return devLogger;
21 };
22
23 let logger;
24
25 if (process.env.NODE_ENV.toLocaleUpperCase() === "PROD") {
26   logger = buildProdLogger();
27 } else {
28   logger = buildDevLogger();
29 }
30
31 module.exports = logger;
```

Prender el servidor en modo prof

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> node --prof server.js
8080
{"level":"info","message":"Servidor http escuchando en el puerto 8080 - PID 10540"}
Servidor escuchando: 8080
█
```

Uso de artillery con 50 conexiones y 20 request

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> artillery quick --count 50 --num 20 http://localhost:8080/info > result_fork.txt
PS C:\xampp\htdocs\Coder\SegundaEntrega> █
```

Uso de --prof-process

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> node --prof-process prof_info.log > result_prof.txt
(node:4532) ExperimentalWarning: VM Modules is an experimental feature. This feature could change at any time
(Use `node --trace-warnings ...` to show where the warning was created)
Could not find function 0xec540cf90
Could not find function 0xec540cf90
Could not find function 0xec540d408
Could not find function 0x30482ffce70
Could not find function 0xec540d458
Could not find function 0xec540cf90
Could not find function 0xec540d408
Could not find function 0x30482ffce70
Could not find function 0xec540d458
Could not find function 0xec540cf90
Could not find function 0xec540d408
Could not find function 0x30482ffce70
Could not find function 0xec540d408
Could not find function 0x3fda9e5940
Could not find function 0xec540cf90
Could not find function 0xec540d408
Could not find function 0x30482ffce70
Could not find function 0x30482fcf918
Could not find function 0x15df9578fc0
Could not find function 0xad5e051570
Could not find function 0xad5e0509a0
Could not find function 0xad5e051018
Could not find function 0x15df9578fc0
Could not find function 0xad5e06ad10
Could not find function 0xad5e064ce0
Could not find function 0xad5e050700
Could not find function 0xad5e0445a0
Could not find function 0xad5e052208
PS C:\xampp\htdocs\Coder\SegundaEntrega> |
```

Uso de autocannon, emulando 100 conexiones en un lapso de 20 segundos

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> npm test
npm WARN config global `--global`, `--local` are deprecated. Use `--location=global` instead.
```

```
> backend@1.0.0 test
> node benchmark.js
```

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

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	82 ms	122 ms	467 ms	664 ms	160.1 ms	108.86 ms	795 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	197	197	571	1062	623.1	278.78	197
Bytes/Sec	165 kB	165 kB	479 kB	891 kB	523 kB	234 kB	165 kB

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

```
13k requests in 20.28s, 10.5 MB read
PS C:\xampp\htdocs\Coder\SegundaEntrega> █
```

Configuración encontrada en benchmark.js

Prender el servidor mediante inspect

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> node --inspect server.js
Debugger listening on ws://127.0.0.1:9229/b6898b3e-9606-458f-9e11-4a9233eb43f9
For help, see: https://nodejs.org/en/docs/inspector
8080
{"level":"info","message":"Servidor http escuchando en el puerto 8080 - PID 664"}
Servidor escuchando: 8080
```

Pantallazo de Chrome://inspect

Self Time	Total Time	Function
10509.6 ms	10509.6 ms	(idle)
31.3 ms 21.43 %	78.3 ms 53.57 %	▸ deserializeObject deserializer.ts:117
21.0 ms 14.37 %	21.0 ms 14.37 %	(program)
5.9 ms 4.05 %	11.0 ms 7.54 %	▸ serialize bson.ts:137
5.4 ms 3.73 %	6.0 ms 4.13 %	▸ writeBuffer
4.4 ms 3.02 %	4.4 ms 3.02 %	▸ Long long.ts:136
3.5 ms 2.38 %	5.1 ms 3.49 %	▸ nextTick node:internal/p...ask_queues:104
3.2 ms 2.22 %	88.1 ms 60.32 %	callbackTrampoline node:internal/async_hooks:118
3.2 ms 2.22 %	4.3 ms 2.94 %	emitHook node:internal/async_hooks:228
2.6 ms 1.75 %	2.6 ms 1.75 %	▸ utf8Slice
2.3 ms 1.59 %	4.9 ms 3.33 %	▸ slice node:buffer:593
1.9 ms 1.27 %	3.2 ms 2.22 %	▸ emitInitScript node:internal/async_hooks:485
1.7 ms 1.19 %	83.4 ms 57.06 %	▸ onStreamRead node:internal/s...se_commons:171
1.6 ms 1.11 %	1.6 ms 1.11 %	(garbage collector)
1.6 ms 1.11 %	76.0 ms 51.98 %	▸ processIncomingData message_stream.ts:135
1.6 ms 1.11 %	68.4 ms 46.83 %	▸ (anonymous) connection.ts:256
1.6 ms 1.11 %	90.1 ms 61.67 %	▸ addChunk node:internal/s...s/readable:304
1.5 ms 1.03 %	6.0 ms 4.13 %	▸ serializeInto serializer.ts:750
1.4 ms 0.95 %	1.7 ms 1.19 %	▸ slice node:buffer:1115
1.3 ms 0.87 %	51.6 ms 35.32 %	▾ parse commands.ts:643
1.3 ms 0.87 %	51.6 ms 35.32 %	▸ onMessage connection.ts:381
1.3 ms 0.87 %	5.2 ms 3.57 %	▸ Long.comp long.ts:420
1.0 ms 0.71 %	1.0 ms 0.71 %	▸ FastBuffer node:internal/buffer:958
1.0 ms 0.71 %	171.2 ms 117.14 %	▸ emit node:events:340
1.0 ms 0.71 %	66.8 ms 45.71 %	▸ onMessage connection.ts:381
1.0 ms 0.71 %	6.3 ms 4.29 %	▸ serverUpdateHandler topology.ts:608
1.0 ms 0.71 %	3.8 ms 2.62 %	▸ Long.compare long.ts:403
0.9 ms 0.63 %	5.1 ms 3.49 %	processTicksAndRejections node:internal/p...task_queues:68
0.9 ms 0.63 %	3.4 ms 2.30 %	▸ onwrite node:internal/s...s/writable:425

3) El diagrama de flama con 0x, emulando la carga con Autocannon con los mismos parámetros anteriores.

Aquí supondré que se refiere
iniciar el servidor con 0x,
como lo deje configurado en
el package.json

```
▷ Debug  
"scripts": {  
  "test": "node benchmark.js",  
  "start": "0x server.js"  
},
```

```
PS C:\xampp\htdocs\Coder\SegundaEntrega> npm start  
npm WARN config global `--global`, `--local` are deprecated. Use `--location=global` instead.  
  
> backend@1.0.0 start  
> 0x server.js  
  
Profiling8080  
{ "level": "info", "message": "Servidor http escuchando en el puerto 8080 - PID 5876" }  
Servidor escuchando: 8080  
█
```