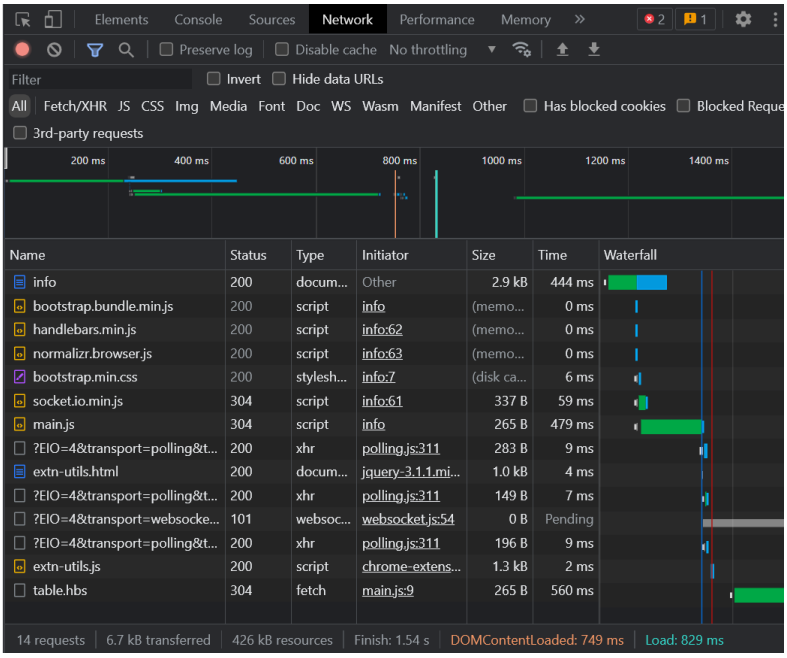
Performance improvement report:

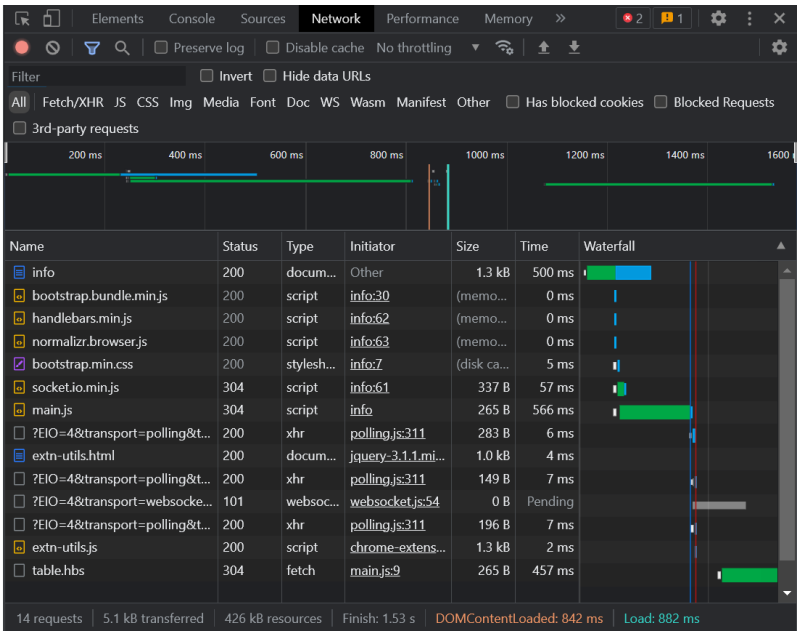
/info route using GZIP: 6.7kB transferred

Key	Value
Title	node
Arguments	
Path	C:\Users\raranguren\Desktop\backend-portfolio\14_logs-and-profiling
Operating System	win32
Process ID	14824
Node.js version	v16.18.0
Folder	C:\Users\raranguren\Desktop\backend-portfolio\14_logs-and-profiling\src
Reserved memory	78.92992 MB



/info route using GZIP: 5.1kB transferred

Key	Value
Title	node
Arguments	
Path	C:\Users\raranguren\Desktop\backend-portfolio\14_logs-and-profiling
Operating System	win32
Process ID	21188
Node.js version	v16.18.0
Folder	C:\Users\raranguren\Desktop\backend-portfolio\14_logs-and-profiling\src
Reserved memory	79.290368 MB



/info without console.log()

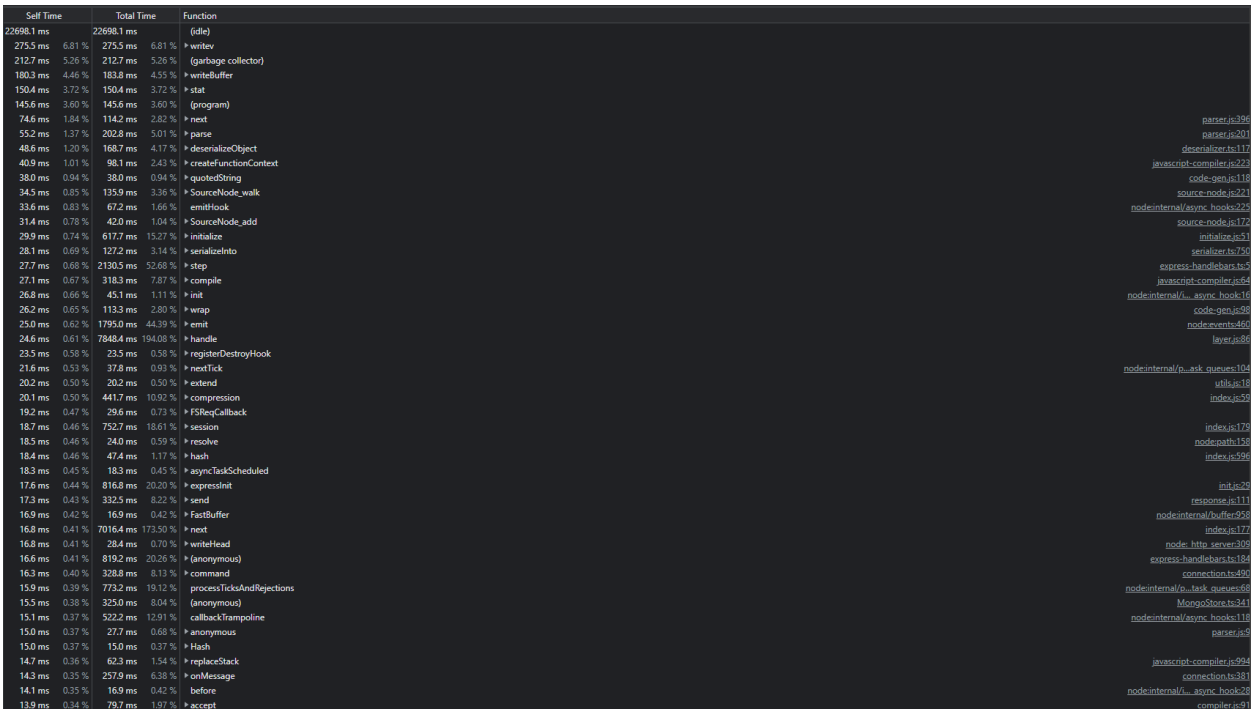
Artillery output:

```
Running scenarios...
Phase started: unnamed (index: 0, duration: 1s) 09:55:48(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 09:55:49(-0300)

All VUs finished. Total time: 15 seconds

--------------------------------
Summary report @ 09:56:02(-0300)
--------------------------------

errors.ETIMEDOUT: ..................................................... 50
http.codes.200: ....................................................... 50
http.request_rate: .................................................... 13/sec
http.requests: ........................................................ 50
http.response_time:
  min: ................................................................ 3
  max: ................................................................ 72
  median: ............................................................. 4
  p95: ................................................................ 32.1
  p99: ................................................................ 67.4
http.responses: ....................................................... 50
vusers.created: ....................................................... 50
vusers.created_by_name.0: ............................................. 50
vusers.failed: ........................................................ 50
```

Autocannon output:

```
raranguren@RARANGUREN-4G99 MINGW64 ~/Desktop/backend-portfolio/14_logs-and-profi
ling (master)
$ npm run autocannon

> logs-and-profiling@1.0.0 autocannon
> autocannon http://localhost:8080/info -c 100 -d 20

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


┌─────────┬────────┬─────────┬─────────┬─────────┬──────────┬──────────┬──────
──┐
│ Stat    │ 2.5%   │ 50%     │ 97.5%   │ 99%     │ Avg      │ Stdev    │ Max
  │
├─────────┼────────┼─────────┼─────────┼─────────┼──────────┼──────────┼──────
──┤
│ Latency │ 853 ms │ 1032 ms │ 2744 ms │ 2912 ms │ 1286.99 ms │ 523.25 ms │ 4280
 ms │
└─────────┴────────┴─────────┴─────────┴─────────┴──────────┴──────────┴──────
──┘

┌─────────┬──────┬──────┬──────┬──────┬──────┬──────┬──────┐
│ Stat    │ 1%   │ 2.5% │ 50%  │ 97.5% │ Avg  │ Stdev │ Min  │
├─────────┼──────┼──────┼──────┼──────┼──────┼──────┼──────┤
│ Req/Sec │ 24   │ 24   │ 88   │ 108  │ 75.55 │ 25.92 │ 24   │
├─────────┼──────┼──────┼──────┼──────┼──────┼──────┼──────┤
│ Bytes/Sec │ 72.5 kB │ 72.5 kB │ 266 kB │ 326 kB │ 228 kB │ 78.2 kB │ 72.5 kB │
└─────────┴──────┴──────┴──────┴──────┴──────┴──────┴──────┘

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

2k requests in 20.12s, 4.56 MB read
```

Chrome inspect output:



Self Time		Total Time		Function	
22698.1 ms		22698.1 ms		(idle)	
275.5 ms	6.81 %	275.5 ms	6.81 %	writev	
212.7 ms	5.26 %	212.7 ms	5.26 %	(garbage collector)	
180.3 ms	4.46 %	183.8 ms	4.55 %	writeBuffer	
150.4 ms	3.72 %	150.4 ms	3.72 %	stat	
145.6 ms	3.60 %	145.6 ms	3.60 %	(program)	
74.6 ms	1.84 %	114.2 ms	2.82 %	next	parser.js:396
55.2 ms	1.37 %	202.8 ms	5.01 %	parse	parser.js:201
48.6 ms	1.20 %	168.7 ms	4.17 %	deserializeObject	deserializer.ts:117
40.9 ms	1.01 %	98.1 ms	2.43 %	createFunctionContext	javascript-compiler.js:223
38.0 ms	0.94 %	38.0 ms	0.94 %	quotedString	code-gen.js:118
34.5 ms	0.85 %	135.9 ms	3.36 %	SourceNode_walk	source-node.js:221
33.6 ms	0.83 %	67.2 ms	1.66 %	emitHook	node:internal/async_hooks:225
31.4 ms	0.78 %	42.0 ms	1.04 %	SourceNode_add	source-node.js:172
29.9 ms	0.74 %	617.7 ms	15.27 %	initialize	initialize.js:51
28.1 ms	0.69 %	127.2 ms	3.14 %	serializeInto	serializer.ts:750
27.7 ms	0.68 %	2130.5 ms	52.68 %	step	express-handlebars.ts:5
27.1 ms	0.67 %	318.3 ms	7.87 %	compile	javascript-compiler.js:64
26.8 ms	0.66 %	45.1 ms	1.11 %	init	node:internal/i... async_hook:16
26.2 ms	0.65 %	113.3 ms	2.80 %	wrap	code-gen.js:98
25.0 ms	0.62 %	1795.0 ms	44.39 %	emit	node:events:460
24.6 ms	0.61 %	7848.4 ms	194.08 %	handle	layer.js:86
23.5 ms	0.58 %	23.5 ms	0.58 %	registerDestroyHook	
21.6 ms	0.53 %	37.8 ms	0.93 %	nextTick	node:internal/p...ask_queues:104
20.2 ms	0.50 %	20.2 ms	0.50 %	extend	utils.js:18
20.1 ms	0.50 %	441.7 ms	10.92 %	compression	index.js:55
19.2 ms	0.47 %	29.6 ms	0.73 %	FSReqCallback	
18.7 ms	0.46 %	752.7 ms	18.61 %	session	index.js:179
18.5 ms	0.46 %	24.0 ms	0.59 %	resolve	node:path:158
18.4 ms	0.46 %	47.4 ms	1.17 %	hash	index.js:596
18.3 ms	0.45 %	18.3 ms	0.45 %	asyncTaskScheduled	
17.6 ms	0.44 %	816.8 ms	20.20 %	expressInit	init.js:29
17.3 ms	0.43 %	332.5 ms	8.22 %	send	response.js:111
16.9 ms	0.42 %	16.9 ms	0.42 %	FastBuffer	node:internal/buffer:958
16.8 ms	0.41 %	7016.4 ms	173.50 %	next	index.js:177
16.8 ms	0.41 %	28.4 ms	0.70 %	writeHead	node: http_server:309
16.6 ms	0.41 %	819.2 ms	20.26 %	(anonymous)	express-handlebars.ts:184
16.3 ms	0.40 %	328.8 ms	8.13 %	command	connection.ts:490
15.9 ms	0.39 %	773.2 ms	19.12 %	processTicksAndRejections	node:internal/p...task_queues:68
15.5 ms	0.38 %	325.0 ms	8.04 %	(anonymous)	MongoStore.ts:341
15.1 ms	0.37 %	522.2 ms	12.91 %	callbackTrampoline	node:internal/async_hooks:118
15.0 ms	0.37 %	27.7 ms	0.68 %	anonymous	parser.js:9
15.0 ms	0.37 %	15.0 ms	0.37 %	Hash	
14.7 ms	0.36 %	62.3 ms	1.54 %	replaceStack	javascript-compiler.js:994
14.3 ms	0.35 %	257.9 ms	6.38 %	onMessage	connection.ts:381
14.1 ms	0.35 %	16.9 ms	0.42 %	before	node:internal/i... async_hook:28
13.9 ms	0.34 %	79.7 ms	1.97 %	accept	compiler.js:91

0x flame graph: see contents of /14944.0x

/info with console.log()

Artillery output:

```
Running scenarios...
Phase started: unnamed (index: 0, duration: 1s) 10:20:36(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 10:20:37(-0300)

-------------------------------------
Metrics for period to: 10:20:40(-0300) (width: 0.985s)
-------------------------------------

http.codes.200: ..................................................................... 50
http.request_rate: .................................................................. 50/sec
http.requests: ...................................................................... 50
http.response_time:
  min: .............................................................................. 3
  max: .............................................................................. 45
  median: ........................................................................... 4
  p95: .............................................................................. 8.9
  p99: .............................................................................. 44.3
http.responses: ..................................................................... 50
vusers.created: ..................................................................... 50
vusers.created_by_name.0: ........................................................... 50


All VUs finished. Total time: 25 seconds
```

Autocannon output:

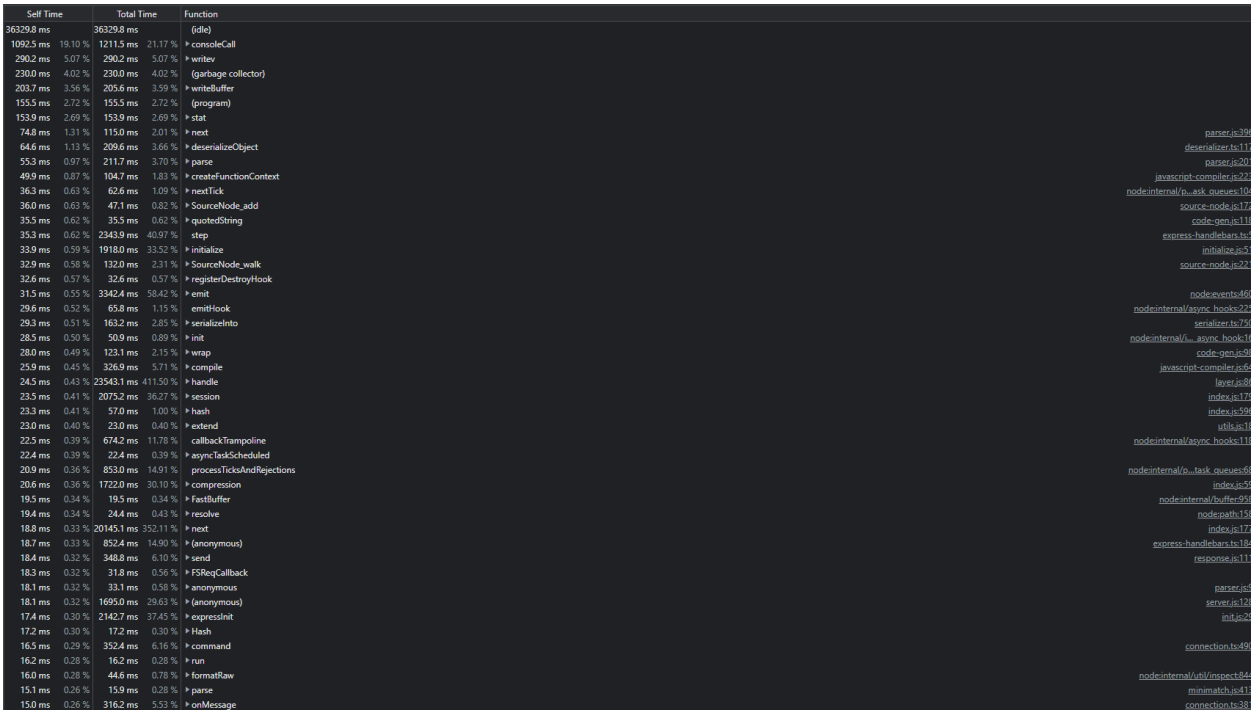
```
raranguren@RARANGUREN-4G99 MINGW64 ~/Desktop/backend-portfolio/14_logs-and-profiling (master)
$ npm run autocannon

> logs-and-profiling@1.0.0 autocannon
> autocannon http://localhost:8080/info -c 100 -d 20

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

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	836 ms	1022 ms	2006 ms	2420 ms	1143.54 ms	323.45 ms	3111 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	33	33	96	103	85.7	19.51	33
Bytes/Sec	99.5 kB	99.5 kB	289 kB	310 kB	258 kB	58.7 kB	99.4 kB

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

2k requests in 20.13s, 5.16 MB read
```

Chrome inspect output:



Self Time		Total Time		Function	
36329.8 ms		36329.8 ms		(idle)	
1092.5 ms	19.10 %	1211.5 ms	21.17 %	consoleCall	
290.2 ms	5.07 %	290.2 ms	5.07 %	writev	
230.0 ms	4.02 %	230.0 ms	4.02 %	(garbage collector)	
203.7 ms	3.56 %	205.6 ms	3.59 %	writeBuffer	
155.5 ms	2.72 %	155.5 ms	2.72 %	(program)	
153.9 ms	2.69 %	153.9 ms	2.69 %	stat	
74.8 ms	1.31 %	115.0 ms	2.01 %	next	parser.js:39
64.6 ms	1.13 %	209.6 ms	3.66 %	deserializeObject	deserializer.ts:117
55.3 ms	0.97 %	211.7 ms	3.70 %	parse	parser.js:20
49.9 ms	0.87 %	104.7 ms	1.83 %	createFunctionContext	javascript-compiler.js:22
36.3 ms	0.63 %	62.6 ms	1.09 %	nextTick	node:internal/p...ask_queues:104
36.0 ms	0.63 %	47.1 ms	0.82 %	SourceNode_add	source-node.js:17
35.5 ms	0.62 %	35.5 ms	0.62 %	quotedString	code-gen.js:11
35.3 ms	0.62 %	2343.9 ms	40.97 %	step	express-handlebars.ts:
33.9 ms	0.59 %	1918.0 ms	33.52 %	initialize	initialize.js:5
32.9 ms	0.58 %	132.0 ms	2.31 %	SourceNode_walk	source-node.js:22
32.6 ms	0.57 %	32.6 ms	0.57 %	registerDestroyHook	
31.5 ms	0.55 %	3342.4 ms	58.42 %	emit	node:events:460
29.6 ms	0.52 %	65.8 ms	1.15 %	emitHook	node:internal/async_hooks:22
29.3 ms	0.51 %	163.2 ms	2.85 %	serializeInto	serializer.ts:75
28.5 ms	0.50 %	50.9 ms	0.89 %	init	node:internal/a... async_hook:1
28.0 ms	0.49 %	123.1 ms	2.15 %	wrap	code-gen.js:9
25.9 ms	0.45 %	326.9 ms	5.71 %	compile	javascript-compiler.js:6
24.5 ms	0.43 %	23543.1 ms	411.50 %	handle	layer.js:8
23.5 ms	0.41 %	2075.2 ms	36.27 %	session	index.js:17
23.3 ms	0.41 %	57.0 ms	1.00 %	hash	index.js:59
23.0 ms	0.40 %	23.0 ms	0.40 %	extend	utils.js:1
22.5 ms	0.39 %	674.2 ms	11.78 %	callbackTrampoline	node:internal/async_hooks:11
22.4 ms	0.39 %	22.4 ms	0.39 %	asyncTaskScheduled	
20.9 ms	0.36 %	853.0 ms	14.91 %	processTicksAndRejections	node:internal/p...task_queues:6
20.6 ms	0.36 %	1722.0 ms	30.10 %	compression	index.js:5
19.5 ms	0.34 %	19.5 ms	0.34 %	FastBuffer	node:internal/buffer:95
19.4 ms	0.34 %	24.4 ms	0.43 %	resolve	node:path:15
18.8 ms	0.33 %	20145.1 ms	352.11 %	next	index.js:17
18.7 ms	0.33 %	852.4 ms	14.90 %	(anonymous)	express-handlebars.ts:18
18.4 ms	0.32 %	348.8 ms	6.10 %	send	response.js:11
18.3 ms	0.32 %	31.8 ms	0.56 %	FSReqCallback	
18.1 ms	0.32 %	33.1 ms	0.58 %	anonymous	parser.js:
18.1 ms	0.32 %	1695.0 ms	29.63 %	(anonymous)	server.js:12
17.4 ms	0.30 %	2142.7 ms	37.45 %	expressInit	init.js:2
17.2 ms	0.30 %	17.2 ms	0.30 %	Hash	
16.5 ms	0.29 %	352.4 ms	6.16 %	command	connection.ts:49
16.2 ms	0.28 %	16.2 ms	0.28 %	run	
16.0 ms	0.28 %	44.6 ms	0.78 %	formatRaw	node:internal/util/inspect:84
15.1 ms	0.26 %	15.9 ms	0.28 %	parse	minimatch.js:41
15.0 ms	0.26 %	316.2 ms	5.53 %	onMessage	connection.ts:38

0x flame graph: see contents of /2424.0x

Results: since the console.log() function is synchronous, calling it in every iteration of the test calls to the /info route slows down server performance. Hence why we should use asynchronous-based loggers, such as the Winston library.