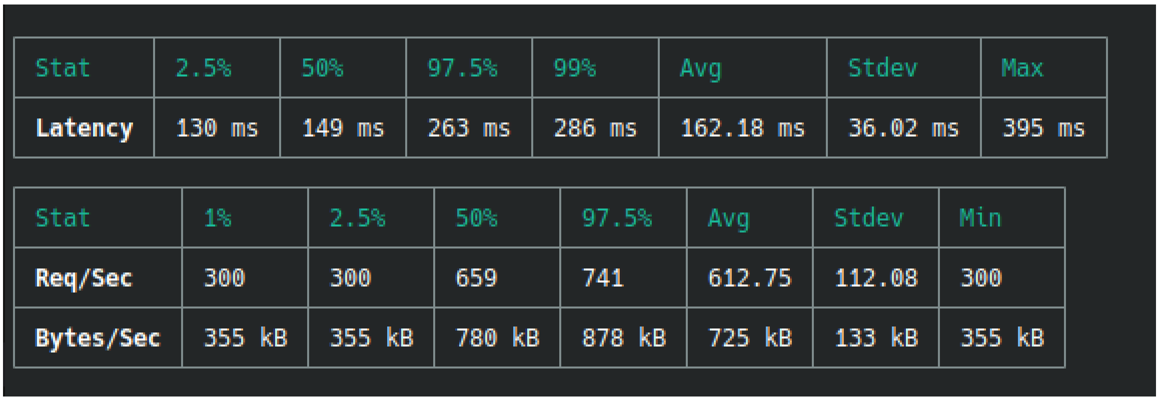
Los siguientes datos fueron obtenidos utilizando la funcion “console.log()” en la ruta /info (modo fork):

Artillery con params -c 20 -n 50:

```
Running scenarios...
Phase started: unnamed (index: 0, duration: 1s) 19:57:11(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 19:57:12(-0300)

All VUs finished. Total time: 5 seconds

--------------------------------
Summary report @ 19:57:15(-0300)
--------------------------------

http.codes.200: ............................................................. 1000
http.request_rate: ......................................................... 146/sec
http.requests: ............................................................... 1000
http.response_time:
  min: ............................................................................ 4
  max: ........................................................................... 83
  median: ..................................................................... 54.1
  p95: ........................................................................... 71.5
  p99: ........................................................................... 79.1
http.responses: ............................................................. 1000
vusers.completed: ........................................................... 20
vusers.created: ............................................................... 20
vusers.created_by_name.0: .................................................... 20
vusers.failed: ................................................................. 0
vusers.session_length:
  min: ............................................................................ 2392
  max: ........................................................................... 2745.6
  median: ..................................................................... 2618.1
  p95: ........................................................................... 2725
  p99: ........................................................................... 2725
```

Autocannon con params -d 20 -c 10:

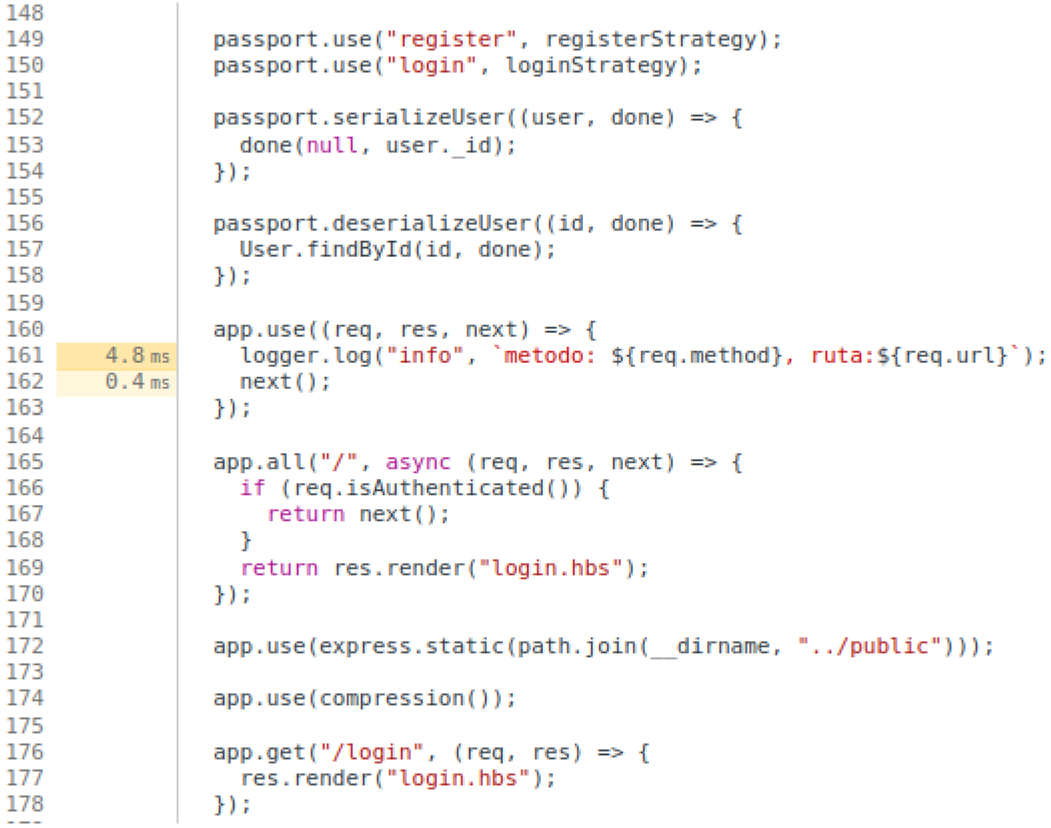


Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	130 ms	149 ms	263 ms	286 ms	162.18 ms	36.02 ms	395 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	300	300	659	741	612.75	112.08	300
Bytes/Sec	355 kB	355 kB	780 kB	878 kB	725 kB	133 kB	355 kB

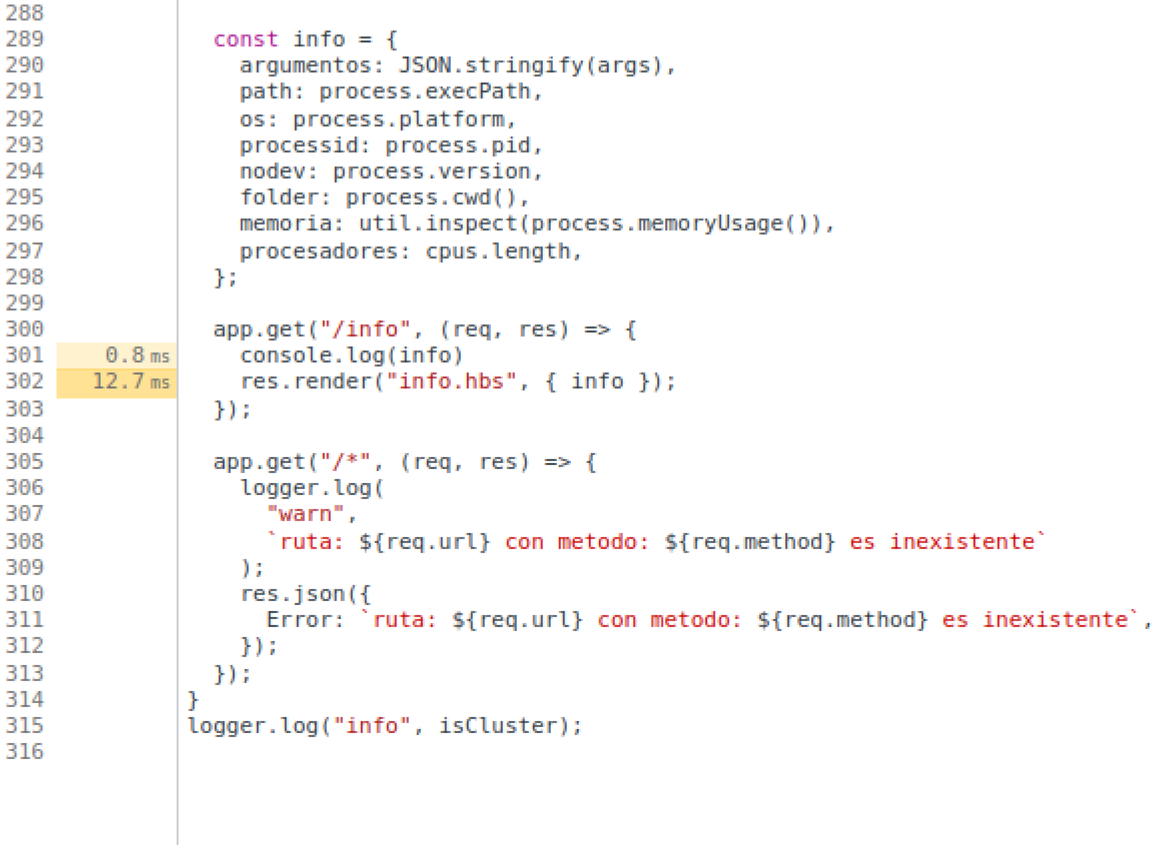
Node –inspect con chrome y artillery con params -c 20 -n 50:

Logger:



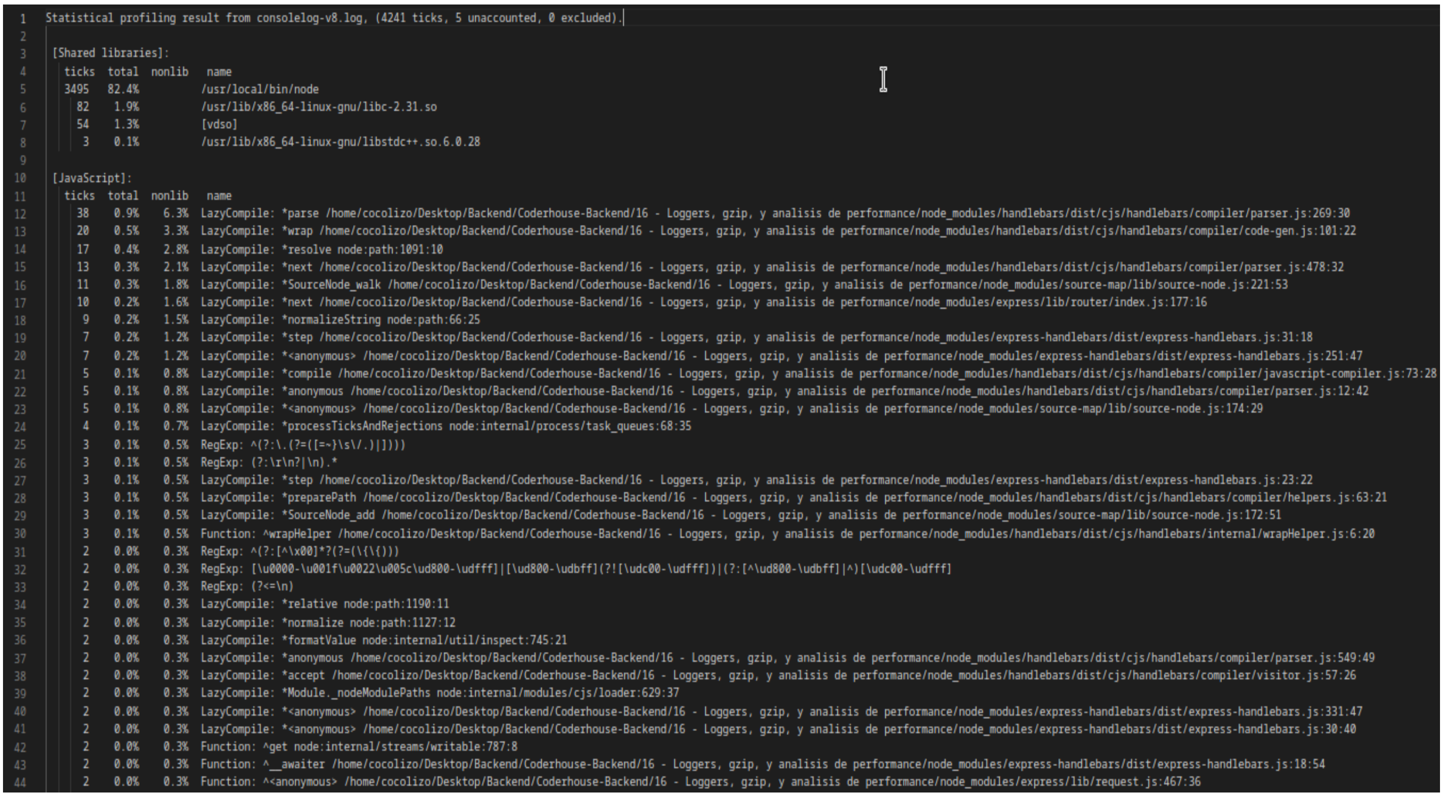
```

passport.use("register", registerStrategy);
passport.use("login", loginStrategy);

passport.serializeUser((user, done) => {
  done(null, user._id);
});

passport.deserializeUser((id, done) => {
  User.findById(id, done);
});

app.use((req, res, next) => {
  logger.log("info", `metodo: ${req.method}, ruta:${req.url}`);
  next();
});

app.all("/", async (req, res, next) => {
  if (req.isAuthenticated()) {
    return next();
  }
  return res.render("login.hbs");
});

app.use(express.static(path.join(__dirname, "../public")));

app.use(compression());

app.get("/login", (req, res) => {
  res.render("login.hbs");
});
```

Ruta /info:

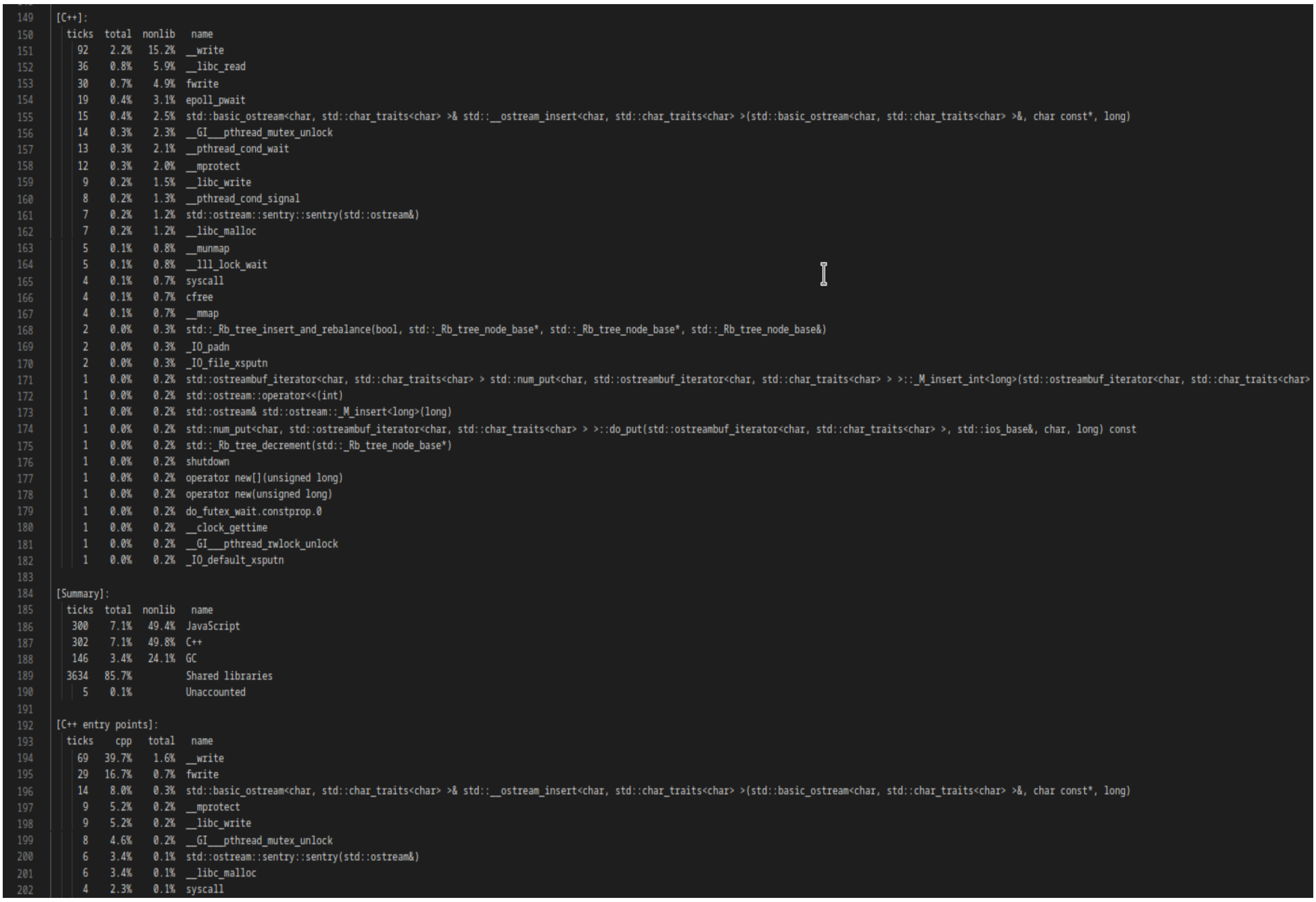


```

  const info = {
    argumentos: JSON.stringify(args),
    path: process.execPath,
    os: process.platform,
    processid: process.pid,
    nodev: process.version,
    folder: process.cwd(),
    memoria: util.inspect(process.memoryUsage()),
    procesadores: cpus.length,
  };

  app.get("/info", (req, res) => {
    console.log(info)
    res.render("info.hbs", { info });
  });

  app.get("/*", (req, res) => {
    logger.log(
      "warn",
      `ruta: ${req.url} con metodo: ${req.method} es inexistente`
    );
    res.json({
      Error: `ruta: ${req.url} con metodo: ${req.method} es inexistente`,
    });
  });
}
logger.log("info", isCluster);

```

El –prof de node.js procesado con –prof-process:
(partes mas relevantes)

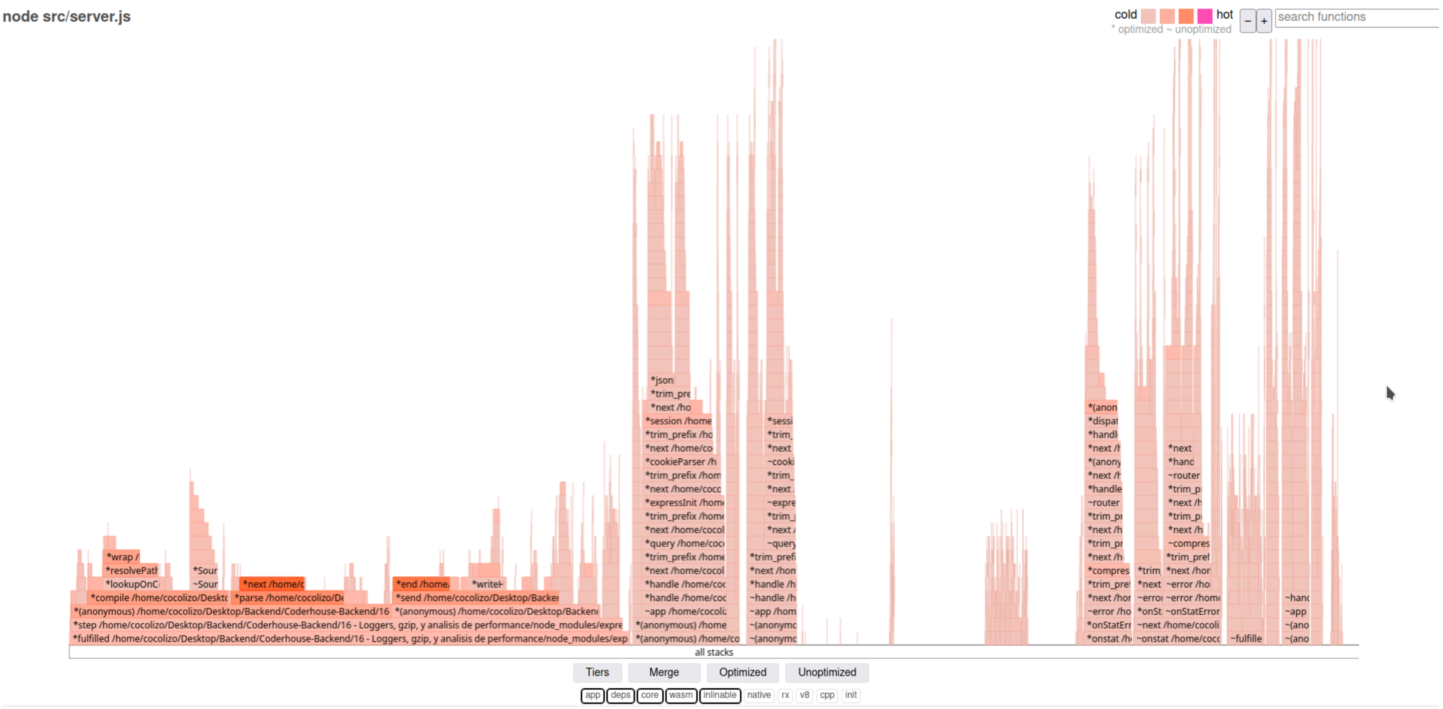


```
Statistical profiling result from consolelog-v8.log, (4241 ticks, 5 unaccounted, 0 excluded).

 [Shared libraries]:
   ticks  total  nonlib   name
   3495   82.4%          /usr/local/bin/node
     82    1.9%          /usr/lib/x86_64-linux-gnu/libc-2.31.so
     54    1.3%          [vdso]
      3    0.1%          /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28

 [JavaScript]:
   ticks  total  nonlib   name
     38    0.9%    6.3%  LazyCompile: *parse /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:269:30
     20    0.5%    3.3%  LazyCompile: *wrap /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/code-gen.js:101:22
     17    0.4%    2.8%  LazyCompile: *resolve node:path:1091:10
     13    0.3%    2.1%  LazyCompile: *next /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:478:32
     11    0.3%    1.8%  LazyCompile: *SourceNode_walk /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/source-map/lib/source-node.js:221:53
     10    0.2%    1.6%  LazyCompile: *next /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express/lib/router/index.js:177:16
      9    0.2%    1.5%  LazyCompile: *normalizeString node:path:66:25
      7    0.2%    1.2%  LazyCompile: *step /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:31:18
      7    0.2%    1.2%  LazyCompile: *<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:251:47
      5    0.1%    0.8%  LazyCompile: *compile /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/javascript-compiler.js:73:28
      5    0.1%    0.8%  LazyCompile: *anonymous /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:12:42
      5    0.1%    0.8%  LazyCompile: *<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/source-map/lib/source-node.js:174:29
      4    0.1%    0.7%  LazyCompile: *processTicksAndRejections node:internal/process/task_queues:68:35
      3    0.1%    0.5%  RegExp: ^(?:\.(?=([=~}\s\/.)|])))
      3    0.1%    0.5%  RegExp: (?:\r\n?|\n).*
      3    0.1%    0.5%  LazyCompile: *step /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:23:22
      3    0.1%    0.5%  LazyCompile: *preparePath /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/helpers.js:63:21
      3    0.1%    0.5%  LazyCompile: *SourceNode_add /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/source-map/lib/source-node.js:172:51
      3    0.1%    0.5%  Function: ^wrapHelper /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/internal/wrapHelper.js:6:20
      2    0.0%    0.3%  RegExp: ^(?:[^\x00]*?(?=(\{\{)))
      2    0.0%    0.3%  RegExp: [\u0000-\u001f\u0022\u005c\ud800-\udfff]|[\ud800-\udbff](?![\udc00-\udfff])|(?:[^\ud800-\udbff]|^)[\udc00-\udfff]
      2    0.0%    0.3%  RegExp: (?<=\n)
      2    0.0%    0.3%  LazyCompile: *relative node:path:1190:11
      2    0.0%    0.3%  LazyCompile: *normalize node:path:1127:12
      2    0.0%    0.3%  LazyCompile: *formatValue node:internal/util/inspect:745:21
      2    0.0%    0.3%  LazyCompile: *anonymous /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:549:49
      2    0.0%    0.3%  LazyCompile: *accept /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/visitor.js:57:26
      2    0.0%    0.3%  LazyCompile: *Module._nodeModulePaths node:internal/modules/cjs/loader:629:37
      2    0.0%    0.3%  LazyCompile: *<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:331:47
      2    0.0%    0.3%  LazyCompile: *<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:30:40
      2    0.0%    0.3%  Function: ^get node:internal/streams/writable:787:8
      2    0.0%    0.3%  Function: ^__awaiter /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:18:54
      2    0.0%    0.3%  Function: ^<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express/lib/request.js:467:36
```



```
 [C++]:
   ticks  total  nonlib   name
     92    2.2%   15.2%  __write
     36    0.8%    5.9%  __libc_read
     30    0.7%    4.9%  fwrite
     19    0.4%    3.1%  epoll_pwait
     15    0.4%    2.5%  std::basic_ostream<char, std::char_traits<char> >& std::__ostream_insert<char, std::char_traits<char> >(std::basic_ostream<char, std::char_traits<char> >&, char const*, long)
     14    0.3%    2.3%  __GI___pthread_mutex_unlock
     13    0.3%    2.1%  __pthread_cond_wait
     12    0.3%    2.0%  __mprotect
      9    0.2%    1.5%  __libc_write
      8    0.2%    1.3%  __pthread_cond_signal
      7    0.2%    1.2%  std::ostream::sentry::sentry(std::ostream&)
      7    0.2%    1.2%  __libc_malloc
      5    0.1%    0.8%  __munmap
      5    0.1%    0.8%  __lll_lock_wait
      4    0.1%    0.7%  syscall
      4    0.1%    0.7%  cfree
      4    0.1%    0.7%  __mmap
      2    0.0%    0.3%  std::_Rb_tree_insert_and_rebalance(bool, std::_Rb_tree_node_base*, std::_Rb_tree_node_base*, std::_Rb_tree_node_base&)
      2    0.0%    0.3%  _IO_padn
      2    0.0%    0.3%  _IO_file_xsputn
      1    0.0%    0.2%  std::ostreambuf_iterator<char, std::char_traits<char> > std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > >::_M_insert_int<long>(std::ostreambuf_iterator<char, std::char_traits<char>
      1    0.0%    0.2%  std::ostream::operator<<(int)
      1    0.0%    0.2%  std::ostream& std::ostream::_M_insert<long>(long)
      1    0.0%    0.2%  std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > >::do_put(std::ostreambuf_iterator<char, std::char_traits<char> >, std::ios_base&, char, long) const
      1    0.0%    0.2%  std::_Rb_tree_decrement(std::_Rb_tree_node_base*)
      1    0.0%    0.2%  shutdown
      1    0.0%    0.2%  operator new[](unsigned long)
      1    0.0%    0.2%  operator new(unsigned long)
      1    0.0%    0.2%  do_futex_wait.constprop.0
      1    0.0%    0.2%  __clock_gettime
      1    0.0%    0.2%  __GI___pthread_rwlock_unlock
      1    0.0%    0.2%  _IO_default_xsputn

 [Summary]:
   ticks  total  nonlib   name
    300    7.1%   49.4%  JavaScript
    302    7.1%   49.8%  C++
    146    3.4%   24.1%  GC
   3634   85.7%          Shared libraries
      5    0.1%          Unaccounted

 [C++ entry points]:
   ticks    cpp   total   name
     69   39.7%    1.6%  __write
     29   16.7%    0.7%  fwrite
     14    8.0%    0.3%  std::basic_ostream<char, std::char_traits<char> >& std::__ostream_insert<char, std::char_traits<char> >(std::basic_ostream<char, std::char_traits<char> >&, char const*, long)
      9    5.2%    0.2%  __mprotect
      9    5.2%    0.2%  __libc_write
      8    4.6%    0.2%  __GI___pthread_mutex_unlock
      6    3.4%    0.1%  std::ostream::sentry::sentry(std::ostream&)
      6    3.4%    0.1%  __libc_malloc
      4    2.3%    0.1%  syscall
```

Diagrama de flama con 0x, autocannon con params -d20 -c100:

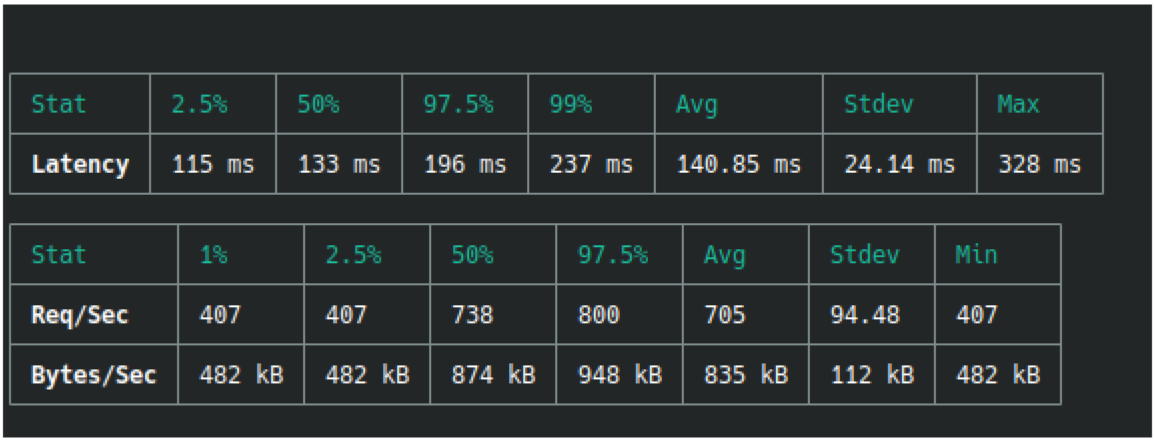


Los siguientes datos fueron obtenidos SIN utilizar la funcion “console.log()” en la ruta /info (modo fork):

Artillery con params -c 20 -n 50:

```
Running scenarios...
Phase started: unnamed (index: 0, duration: 1s) 19:56:11(-0300)

Phase completed: unnamed (index: 0, duration: 1s) 19:56:12(-0300)

All VUs finished. Total time: 5 seconds

--------------------------------
Summary report @ 19:56:16(-0300)
--------------------------------

http.codes.200: ................................................................ 1000
http.request_rate: ............................................................ 173/sec
http.requests: .................................................................. 1000
http.response_time:
  min: ........................................................................... 3
  max: .......................................................................... 91
  median: ...................................................................... 43.4
  p95: ........................................................................... 59.7
  p99: ........................................................................... 85.6
http.responses: ................................................................ 1000
vusers.completed: ............................................................. 20
vusers.created: ................................................................ 20
vusers.created_by_name.0: ................................................... 20
vusers.failed: ................................................................... 0
vusers.session_length:
  min: ........................................................................... 1934.2
  max: .......................................................................... 2279.2
  median: ...................................................................... 2143.5
  p95: ........................................................................... 2276.1
  p99: ........................................................................... 2276.1
```

Autocannon con params -d 20 -c 10:

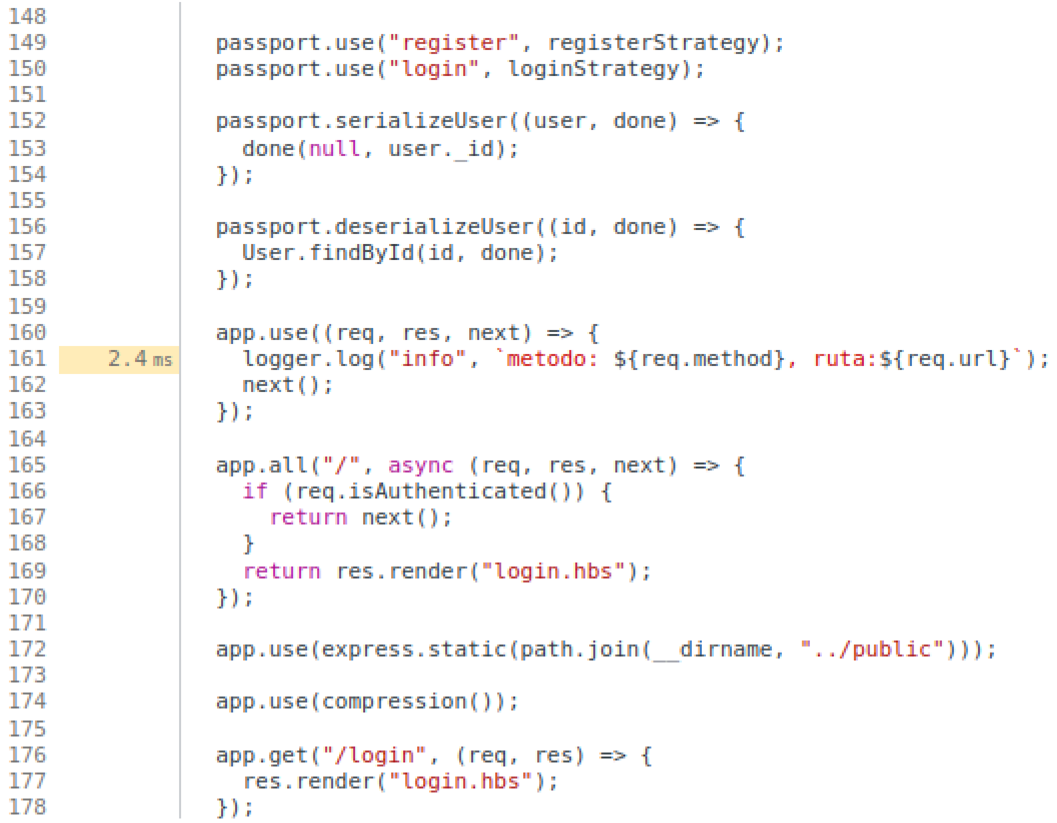


Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	115 ms	133 ms	196 ms	237 ms	140.85 ms	24.14 ms	328 ms

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	407	407	738	800	705	94.48	407
Bytes/Sec	482 kB	482 kB	874 kB	948 kB	835 kB	112 kB	482 kB

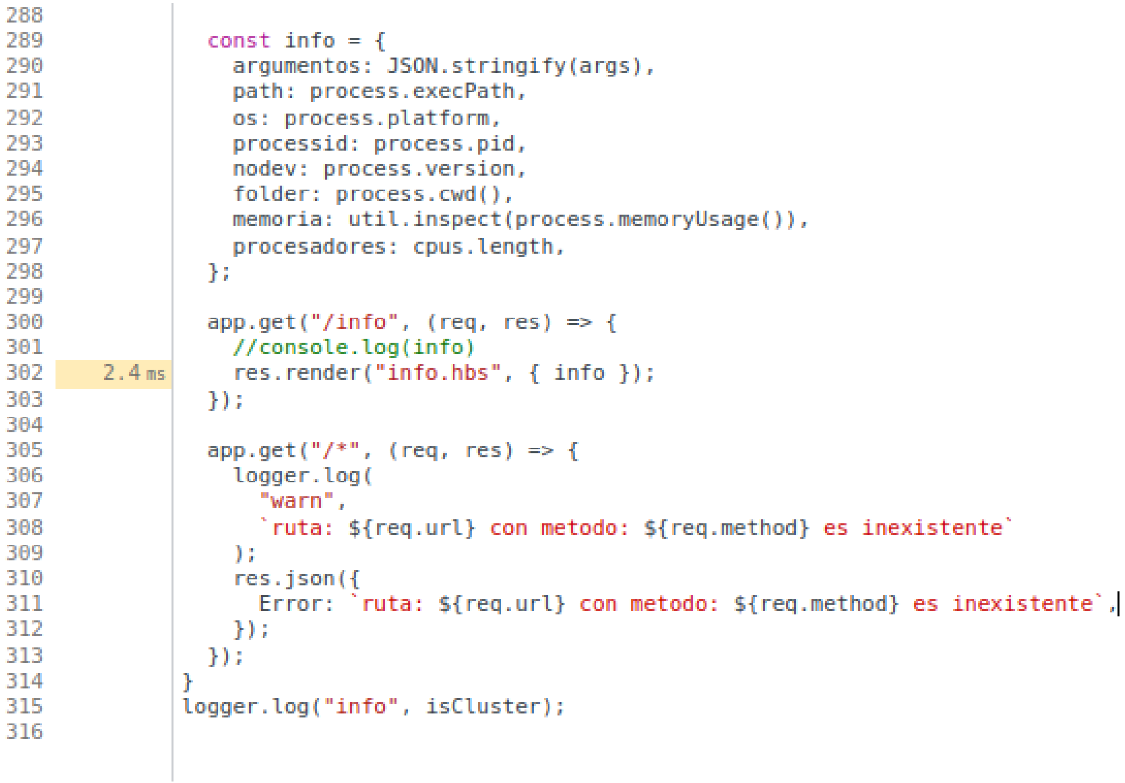
Node –inspect con chrome y artillery con params -c 20 -n 50:

Logger:



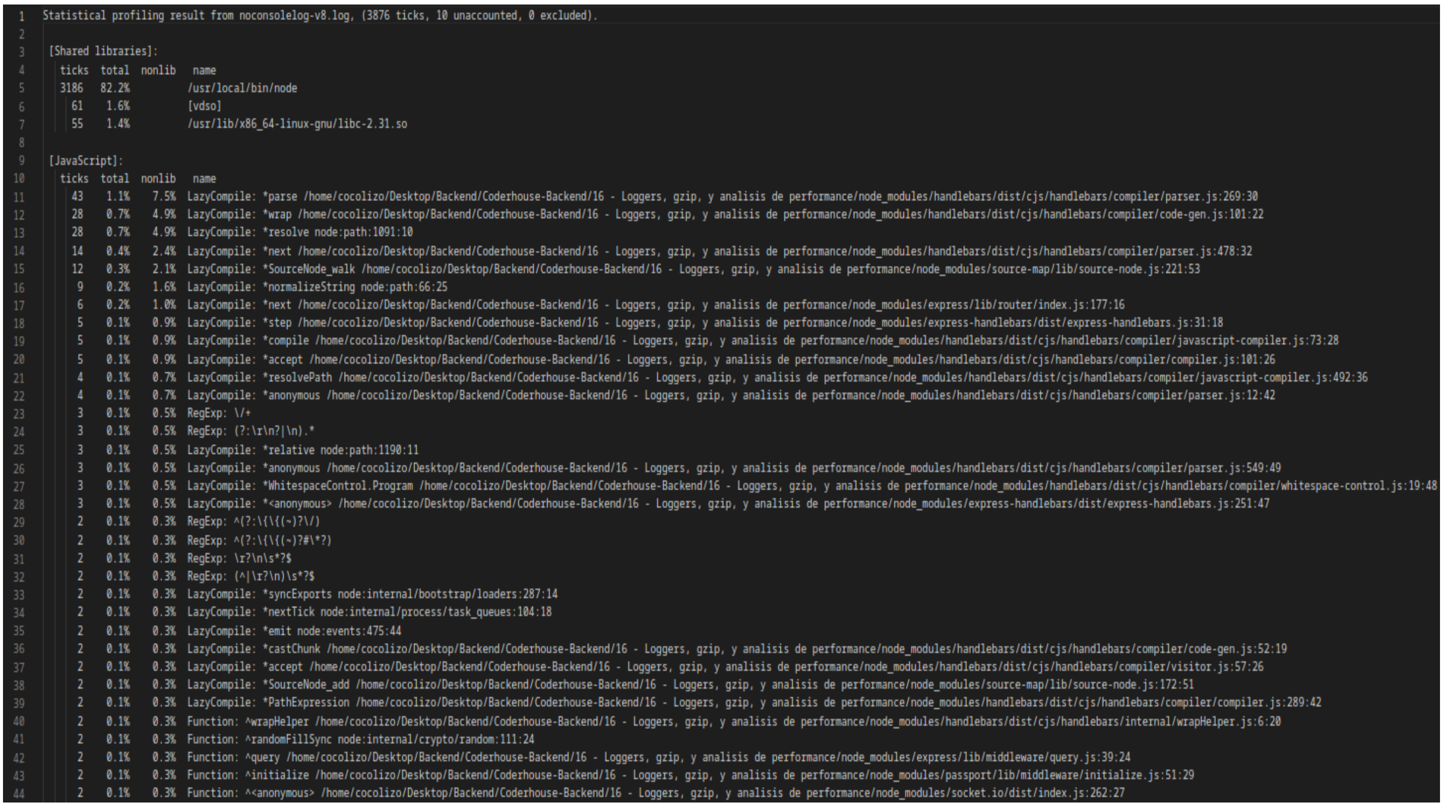
```
passport.use("register", registerStrategy);
passport.use("login", loginStrategy);

passport.serializeUser((user, done) => {
  done(null, user._id);
});

passport.deserializeUser((id, done) => {
  User.findById(id, done);
});

app.use((req, res, next) => {
  logger.log("info", `metodo: ${req.method}, ruta:${req.url}`);
  next();
});

app.all("/", async (req, res, next) => {
  if (req.isAuthenticated()) {
    return next();
  }
  return res.render("login.hbs");
});

app.use(express.static(path.join(__dirname, "../public")));

app.use(compression());

app.get("/login", (req, res) => {
  res.render("login.hbs");
});
```

Ruta /info:

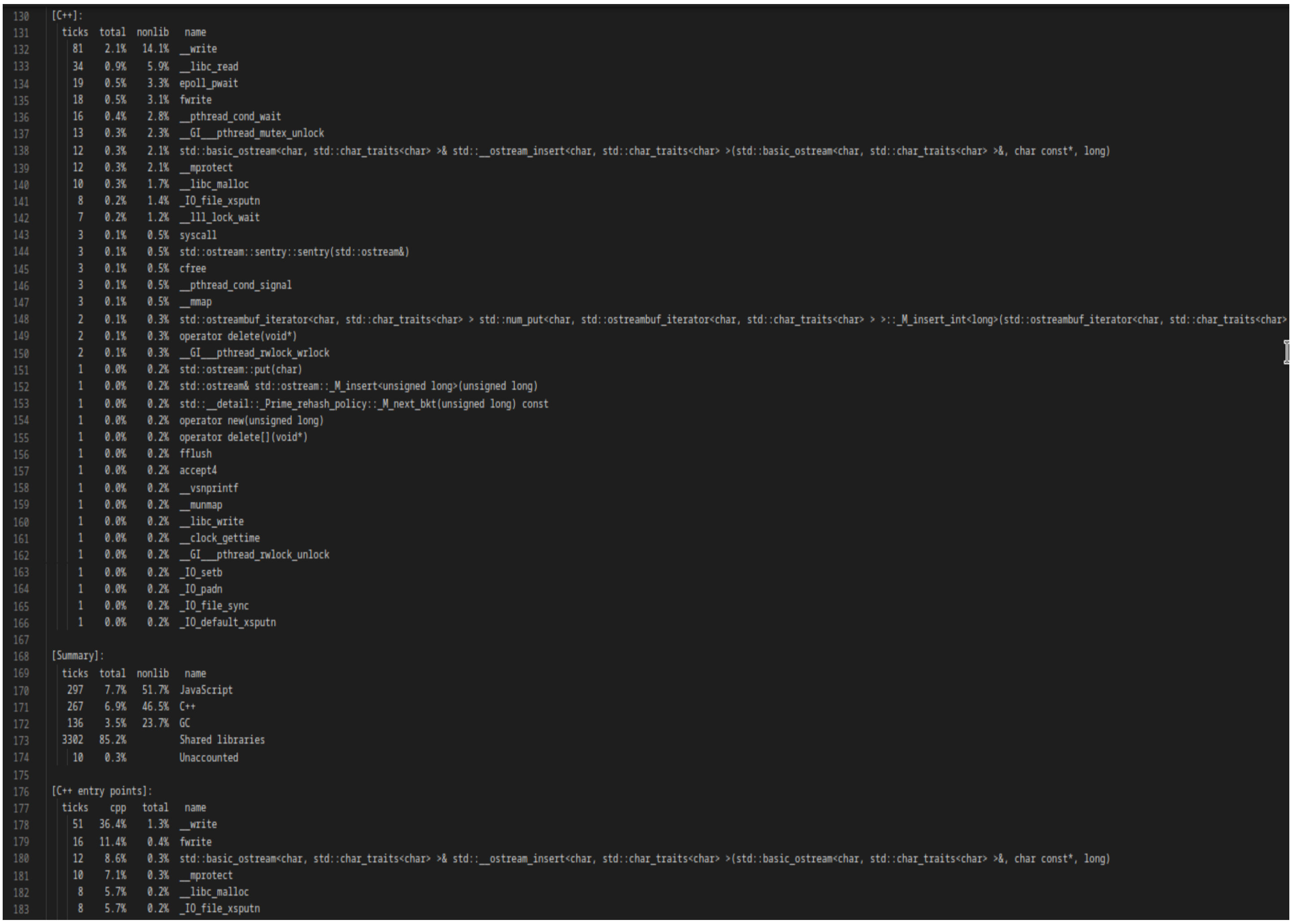


```
  const info = {
    argumentos: JSON.stringify(args),
    path: process.execPath,
    os: process.platform,
    processid: process.pid,
    nodev: process.version,
    folder: process.cwd(),
    memoria: util.inspect(process.memoryUsage()),
    procesadores: cpus.length,
  };

  app.get("/info", (req, res) => {
    //console.log(info)
    res.render("info.hbs", { info });
  });

  app.get("/*", (req, res) => {
    logger.log(
      "warn",
      `ruta: ${req.url} con metodo: ${req.method} es inexistente`
    );
    res.json({
      Error: `ruta: ${req.url} con metodo: ${req.method} es inexistente`,
    });
  });
}
logger.log("info", isCluster);
```

El –prof de node.js procesado con –prof-process: (partes mas relevantes)

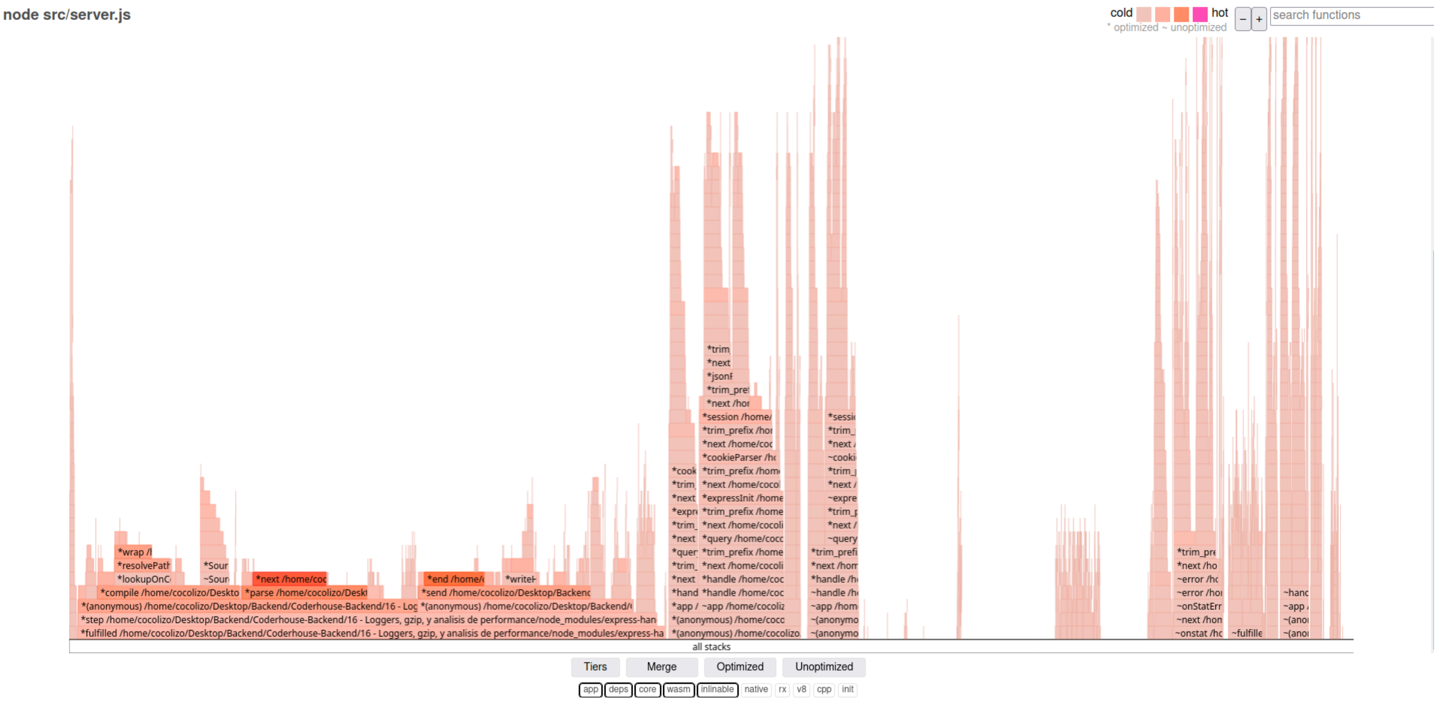


```
Statistical profiling result from noconsolelog-v8.log, (3876 ticks, 10 unaccounted, 0 excluded).

 [Shared libraries]:
   ticks  total  nonlib   name
   3186   82.2%          /usr/local/bin/node
     61    1.6%          [vdso]
     55    1.4%          /usr/lib/x86_64-linux-gnu/libc-2.31.so

 [JavaScript]:
   ticks  total  nonlib   name
     43    1.1%    7.5%  LazyCompile: *parse /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:269:30
     28    0.7%    4.9%  LazyCompile: *wrap /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/code-gen.js:101:22
     28    0.7%    4.9%  LazyCompile: *resolve node:path:1091:10
     14    0.4%    2.4%  LazyCompile: *next /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:478:32
     12    0.3%    2.1%  LazyCompile: *SourceNode_walk /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/source-map/lib/source-node.js:221:53
      9    0.2%    1.6%  LazyCompile: *normalizeString node:path:66:25
      6    0.2%    1.0%  LazyCompile: *next /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express/lib/router/index.js:177:16
      5    0.1%    0.9%  LazyCompile: *step /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:31:18
      5    0.1%    0.9%  LazyCompile: *compile /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/javascript-compiler.js:73:28
      5    0.1%    0.9%  LazyCompile: *accept /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/compiler.js:101:26
      4    0.1%    0.7%  LazyCompile: *resolvePath /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/javascript-compiler.js:492:36
      4    0.1%    0.7%  LazyCompile: *anonymous /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:12:42
      3    0.1%    0.5%  RegExp: \/+
      3    0.1%    0.5%  RegExp: (?:\r\n?|\n).*
      3    0.1%    0.5%  LazyCompile: *relative node:path:1190:11
      3    0.1%    0.5%  LazyCompile: *anonymous /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/parser.js:549:49
      3    0.1%    0.5%  LazyCompile: *WhitespaceControl.Program /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/whitespace-control.js:19:48
      3    0.1%    0.5%  LazyCompile: *<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express-handlebars/dist/express-handlebars.js:251:47
      2    0.1%    0.3%  RegExp: ^(?:\{\{(~)?\/)
      2    0.1%    0.3%  RegExp: ^(?:\{\{(~)?#\*?)
      2    0.1%    0.3%  RegExp: \r?\n\s*?$
      2    0.1%    0.3%  RegExp: (^|\r?\n)\s*?$
      2    0.1%    0.3%  LazyCompile: *syncExports node:internal/bootstrap/loaders:287:14
      2    0.1%    0.3%  LazyCompile: *nextTick node:internal/process/task_queues:104:18
      2    0.1%    0.3%  LazyCompile: *emit node:events:475:44
      2    0.1%    0.3%  LazyCompile: *castChunk /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/code-gen.js:52:19
      2    0.1%    0.3%  LazyCompile: *accept /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/visitor.js:57:26
      2    0.1%    0.3%  LazyCompile: *SourceNode_add /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/source-map/lib/source-node.js:172:51
      2    0.1%    0.3%  LazyCompile: *PathExpression /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/compiler/compiler.js:289:42
      2    0.1%    0.3%  Function: ^wrapHelper /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/handlebars/dist/cjs/handlebars/internal/wrapHelper.js:6:20
      2    0.1%    0.3%  Function: ^randomFillSync node:internal/crypto/random:111:24
      2    0.1%    0.3%  Function: ^query /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/express/lib/middleware/query.js:39:24
      2    0.1%    0.3%  Function: ^initialize /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/passport/lib/middleware/initialize.js:51:29
      2    0.1%    0.3%  Function: ^<anonymous> /home/cocolizo/Desktop/Backend/Coderhouse-Backend/16 - Loggers, gzip, y analisis de performance/node_modules/socket.io/dist/index.js:262:27
```



```
 [C++]:
   ticks  total  nonlib   name
     81    2.1%   14.1%  __write
     34    0.9%    5.9%  __libc_read
     19    0.5%    3.3%  epoll_pwait
     18    0.5%    3.1%  fwrite
     16    0.4%    2.8%  __pthread_cond_wait
     13    0.3%    2.3%  __GI___pthread_mutex_unlock
     12    0.3%    2.1%  std::basic_ostream<char, std::char_traits<char> >& std::__ostream_insert<char, std::char_traits<char> >(std::basic_ostream<char, std::char_traits<char> >&, char const*, long)
     12    0.3%    2.1%  __mprotect
     10    0.3%    1.7%  __libc_malloc
      8    0.2%    1.4%  _IO_file_xsputn
      7    0.2%    1.2%  __lll_lock_wait
      3    0.1%    0.5%  syscall
      3    0.1%    0.5%  std::ostream::sentry::sentry(std::ostream&)
      3    0.1%    0.5%  cfree
      3    0.1%    0.5%  __pthread_cond_signal
      3    0.1%    0.5%  __mmap
      2    0.1%    0.3%  std::ostreambuf_iterator<char, std::char_traits<char> > std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > >::_M_insert_int<long>(std::ostreambuf_iterator<char, std::char_traits<char>
      2    0.1%    0.3%  operator delete(void*)
      2    0.1%    0.3%  __GI___pthread_rwlock_wrlock
      1    0.0%    0.2%  std::ostream::put(char)
      1    0.0%    0.2%  std::ostream& std::ostream::_M_insert<unsigned long>(unsigned long)
      1    0.0%    0.2%  std::__detail::_Prime_rehash_policy::_M_next_bkt(unsigned long) const
      1    0.0%    0.2%  operator new(unsigned long)
      1    0.0%    0.2%  operator delete[](void*)
      1    0.0%    0.2%  fflush
      1    0.0%    0.2%  accept4
      1    0.0%    0.2%  __vsnprintf
      1    0.0%    0.2%  __munmap
      1    0.0%    0.2%  __libc_write
      1    0.0%    0.2%  __clock_gettime
      1    0.0%    0.2%  __GI___pthread_rwlock_unlock
      1    0.0%    0.2%  _IO_setb
      1    0.0%    0.2%  _IO_padn
      1    0.0%    0.2%  _IO_file_sync
      1    0.0%    0.2%  _IO_default_xsputn

 [Summary]:
   ticks  total  nonlib   name
    297    7.7%   51.7%  JavaScript
    267    6.9%   46.5%  C++
    136    3.5%   23.7%  GC
   3302   85.2%          Shared libraries
     10    0.3%          Unaccounted

 [C++ entry points]:
   ticks    cpp   total   name
     51   36.4%    1.3%  __write
     16   11.4%    0.4%  fwrite
     12    8.6%    0.3%  std::basic_ostream<char, std::char_traits<char> >& std::__ostream_insert<char, std::char_traits<char> >(std::basic_ostream<char, std::char_traits<char> >&, char const*, long)
     10    7.1%    0.3%  __mprotect
      8    5.7%    0.2%  __libc_malloc
      8    5.7%    0.2%  _IO_file_xsputn
```

Diagrama de flama con 0x, autocannon con params -d20 -c100:



Conclusiones:

Observando los datos obtenidos se puede entender que al no utilizar la funcion “console.log()” en la ruta de /info, los request a esta misma ruta tuvieron una respuesta mas rapida que al utilizar la funcion “console.log()” .
Asimismo se ve una mayor optimizacion en el codigo, si interpretamos el grafico de flama, ya que cuando no se utiliza “console.log()” se ven picos mas “delgados” a la derecha del diagrama.
Tanto Artillery como Autocannon nos dan una idea muy clara de los tiempos de respuesta con y sin console.log(), esta a la vista la velocidad de respuesta en cada caso.
En tanto en el modo -inspect con chrome se puede ver el tiempo de los procesos menos performantes se multiplican hasta 5 veces utilizando “console.log()”
En el –prof de node se puede tomar la diferencia de “ticks” lo que da a lugar como interpretacion de que se realizaron mas operaciones cuando el codigo se corre con la funcion “console.log()”, haciendo de este un codigo menos optimizado.

Sin lugar a dudas no utilizar console.log() en la ruta de /info nos da un mayor rendimiento al hacer requests a esta ruta, por lo que como conclusion puedo decir que no es aconsejable el utilizar console.log() en el codigo, en especial cuando no esta en etapa de desarrollo.