

Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Software Avanzado

Documentación – Práctica 6

Andrea María López Flores

201404134

Guatemala, 10 de marzo de 2021

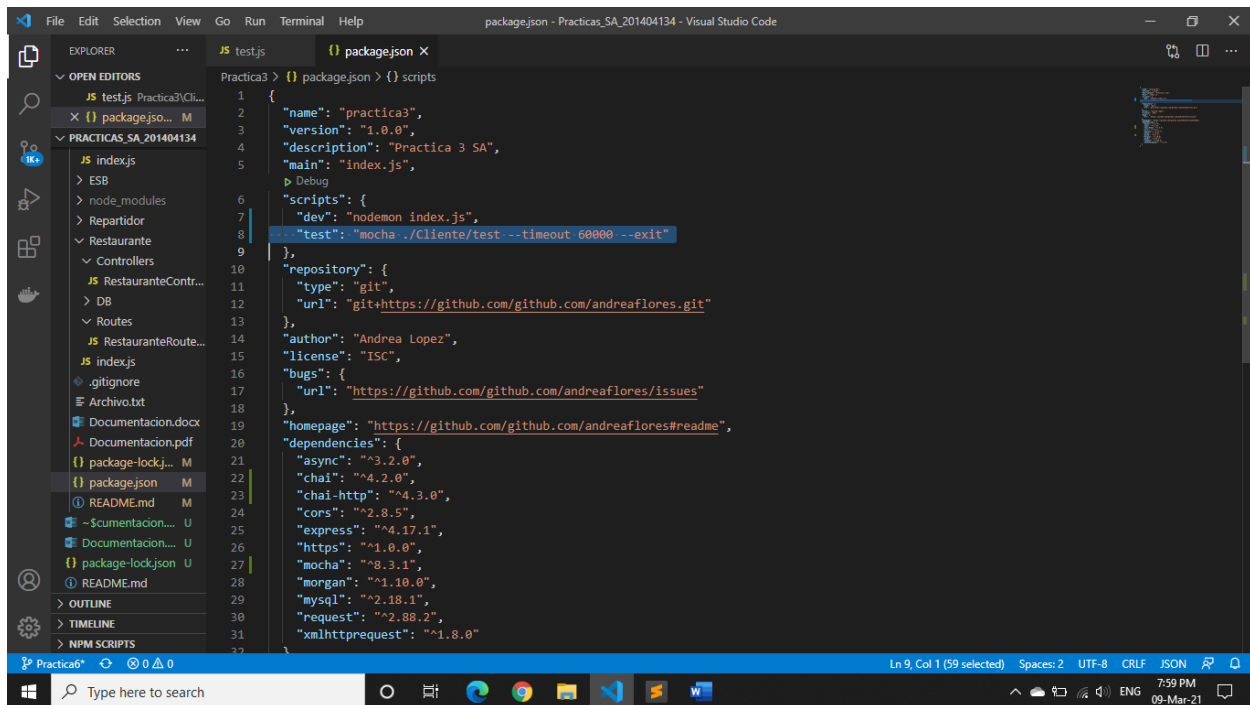
Pruebas unitarias

Las pruebas fueron realizadas con mocha y chai. Se instalaron ambas herramientas en el proyecto.

Se hicieron sobre el servicio del Cliente el cual tiene los siguientes métodos

- Realizar pedido
- Consultar estado del pedido en el restaurante
- Consultar estado del pedido con el repartidos

Para ejecutar las pruebas solo se deben agregar en el package.json en la parte de test.



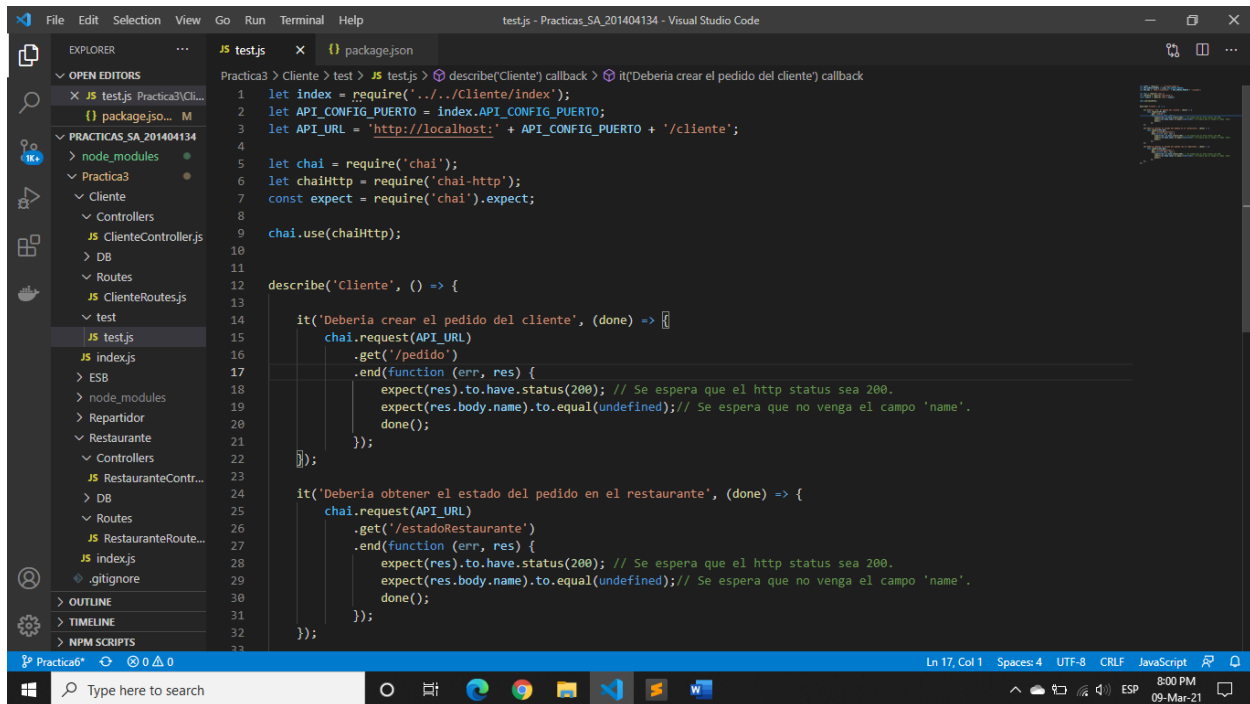
The screenshot shows the Visual Studio Code editor with the `package.json` file open. The `scripts` section is highlighted, showing the following configuration:

```
{
  "name": "practica3",
  "version": "1.0.0",
  "description": "Practica 3 SA",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon index.js",
    "test": "mocha ./Cliente/test --timeout 60000 --exit"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/github.com/andreaflares.git"
  },
  "author": "Andrea Lopez",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/github.com/andreaflares/issues"
  },
  "homepage": "https://github.com/github.com/andreaflares#readme",
  "dependencies": {
    "async": "^3.2.0",
    "chai": "^4.2.0",
    "chai-http": "^4.3.0",
    "cors": "^2.8.5",
    "express": "^4.17.1",
    "https": "^1.0.0",
    "mocha": "^8.3.1",
    "morgan": "^1.10.0",
    "mysql": "^2.18.1",
    "request": "^2.88.2",
    "xmlhttprequest": "^1.8.0"
  }
}
```

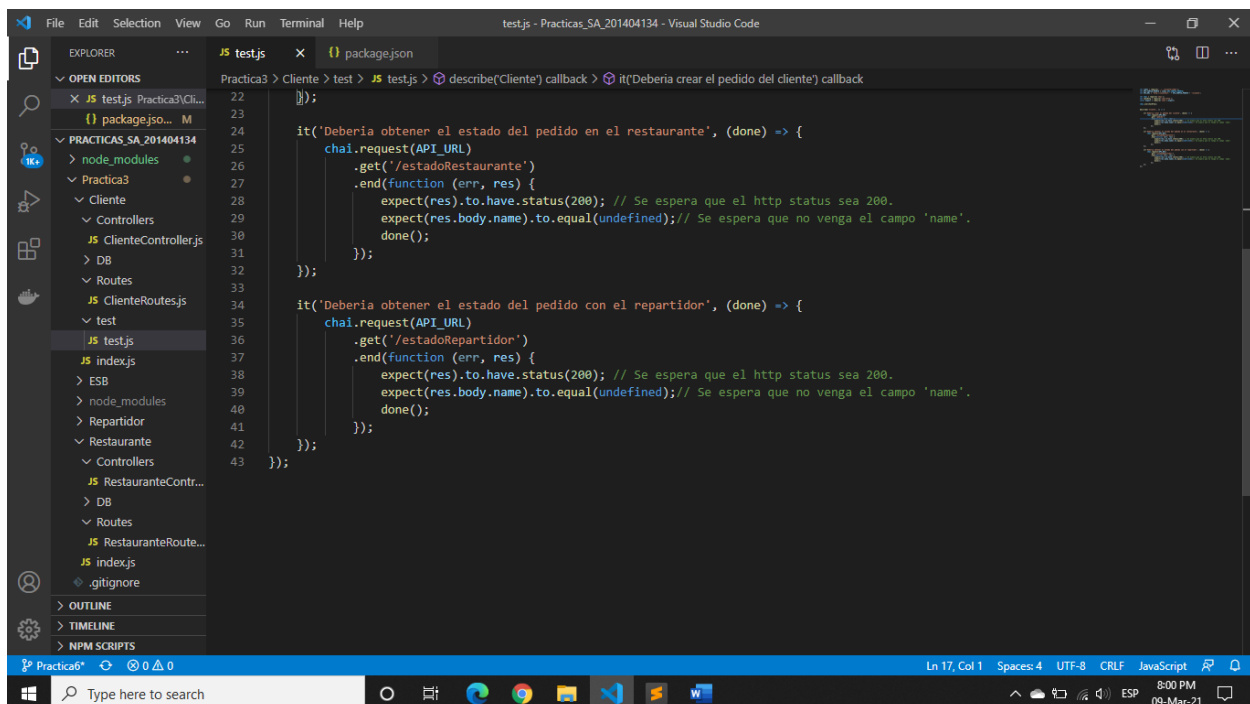
The `test` script is set to run `mocha` on the test files in the `./Cliente/test` directory, with a timeout of 60000ms and the `--exit` flag.

Test.js

Se creó un archivo que contiene las tres pruebas.



```
Practica3 > Cliente > test > JS test.js describe('Cliente') callback > it('Deberia crear el pedido del cliente') callback
1 let index = require('../Cliente/index');
2 let API_CONFIG_PUERTO = index.API_CONFIG_PUERTO;
3 let API_URL = 'http://localhost:' + API_CONFIG_PUERTO + '/cliente';
4
5 let chai = require('chai');
6 let chaiHttp = require('chai-http');
7 const expect = require('chai').expect;
8
9 chai.use(chaiHttp);
10
11
12 describe('Cliente', () => {
13
14     it('Deberia crear el pedido del cliente', (done) => {
15         chai.request(API_URL)
16             .get('/pedido')
17             .end(function (err, res) {
18                 expect(res).to.have.status(200); // Se espera que el http status sea 200.
19                 expect(res.body.name).to.equal(undefined); // Se espera que no venga el campo 'name'.
20                 done();
21             });
22     });
23
24     it('Deberia obtener el estado del pedido en el restaurante', (done) => {
25         chai.request(API_URL)
26             .get('/estadoRestaurante')
27             .end(function (err, res) {
28                 expect(res).to.have.status(200); // Se espera que el http status sea 200.
29                 expect(res.body.name).to.equal(undefined); // Se espera que no venga el campo 'name'.
30                 done();
31             });
32     });
33 });
```



```
Practica3 > Cliente > test > JS test.js describe('Cliente') callback > it('Deberia crear el pedido del cliente') callback
22 });
23
24 it('Deberia obtener el estado del pedido en el restaurante', (done) => {
25     chai.request(API_URL)
26         .get('/estadoRestaurante')
27         .end(function (err, res) {
28             expect(res).to.have.status(200); // Se espera que el http status sea 200.
29             expect(res.body.name).to.equal(undefined); // Se espera que no venga el campo 'name'.
30             done();
31         });
32 });
33
34 it('Deberia obtener el estado del pedido con el repartidor', (done) => {
35     chai.request(API_URL)
36         .get('/estadoRepartidor')
37         .end(function (err, res) {
38             expect(res).to.have.status(200); // Se espera que el http status sea 200.
39             expect(res.body.name).to.equal(undefined); // Se espera que no venga el campo 'name'.
40             done();
41         });
42 });
43 });
```

Resultado de las pruebas

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Displays the project structure for 'Practicas_SA_201404134', including folders for 'node_modules', 'Practica3', 'Cliente', 'Controllers', 'DB', 'Routes', and 'RestaurantRoute...'. The 'test' folder is selected.
- TESTS:** Shows a list of test files: 'test.js', 'index.js', 'ESB', 'node_modules', 'Repartidor', 'Restaurant', 'Controllers', 'RestaurantContr...', 'DB', 'Routes', 'RestaurantRoute...', 'index.js', and '.gitignore'. The 'test.js' file is selected.
- TEST RESULTS:** Displays the following output:

```
> practica3 1.0.0 test C:\Users\Andrea\Documents\USAC\2021\primer_semestre\Software\Lab\Practicas\Practicas_SA_201404134\Practica3
> mocha ./Cliente/test --timeout 60000 --exit

Cliente corriendo en el puerto:8090

  Cliente
    Conectado con el identificador 39
    GET /cliente/pedido 200 21.067 ms - 43
      ✓ Deberia crear el pedido del cliente (163ms)
    GET /cliente/estadoRestaurante 200 10.926 ms - 351
      ✓ Deberia obtener el estado del pedido en el restaurante
    GET /cliente/estadoRepartidor 200 4.385 ms - 245
      ✓ Deberia obtener el estado del pedido con el repartidor

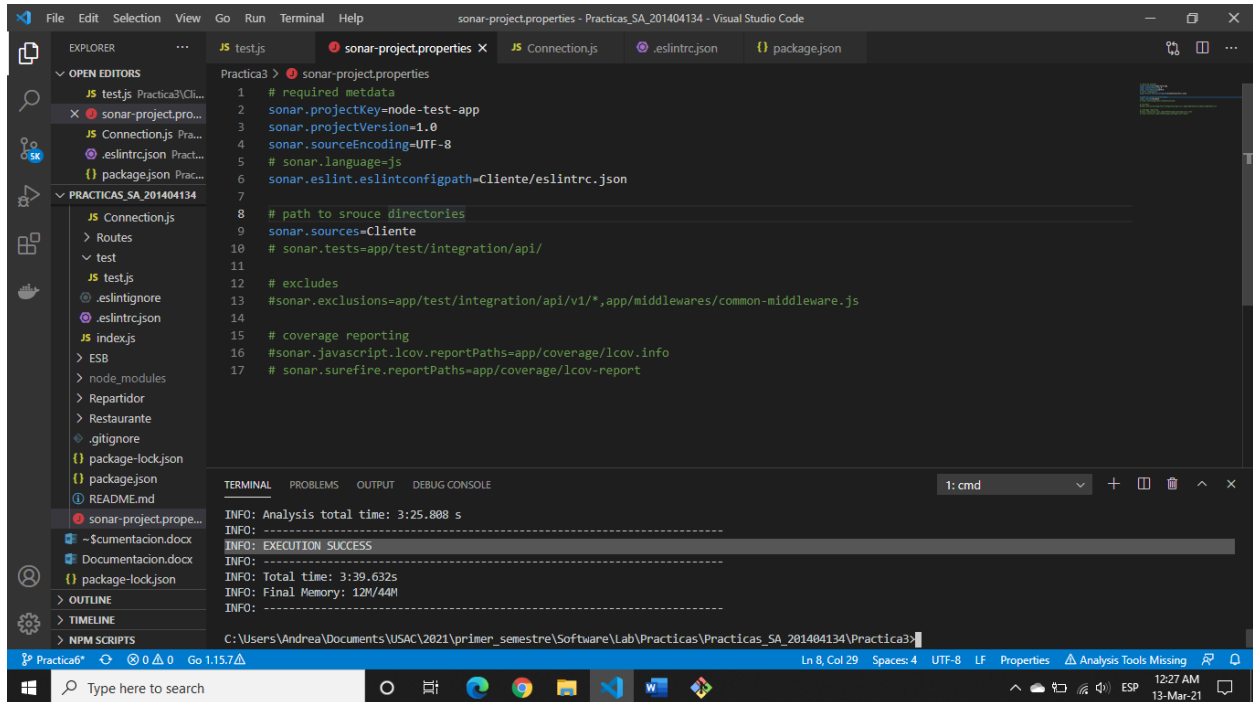
  3 passing (271ms)
```
- TERMINAL:** Shows the command prompt output for running the tests:

```
C:\Users\Andrea\Documents\USAC\2021\primer_semestre\Software\Lab\Practicas\Practicas_SA_201404134\Practica3>cd Restaurante
C:\Users\Andrea\Documents\USAC\2021\primer_semestre\Software\Lab\Practicas\Practicas_SA_201404134\Practica3>npm run test
```

Todas las pruebas pasaron con éxito.

Sonarqube

Archivos



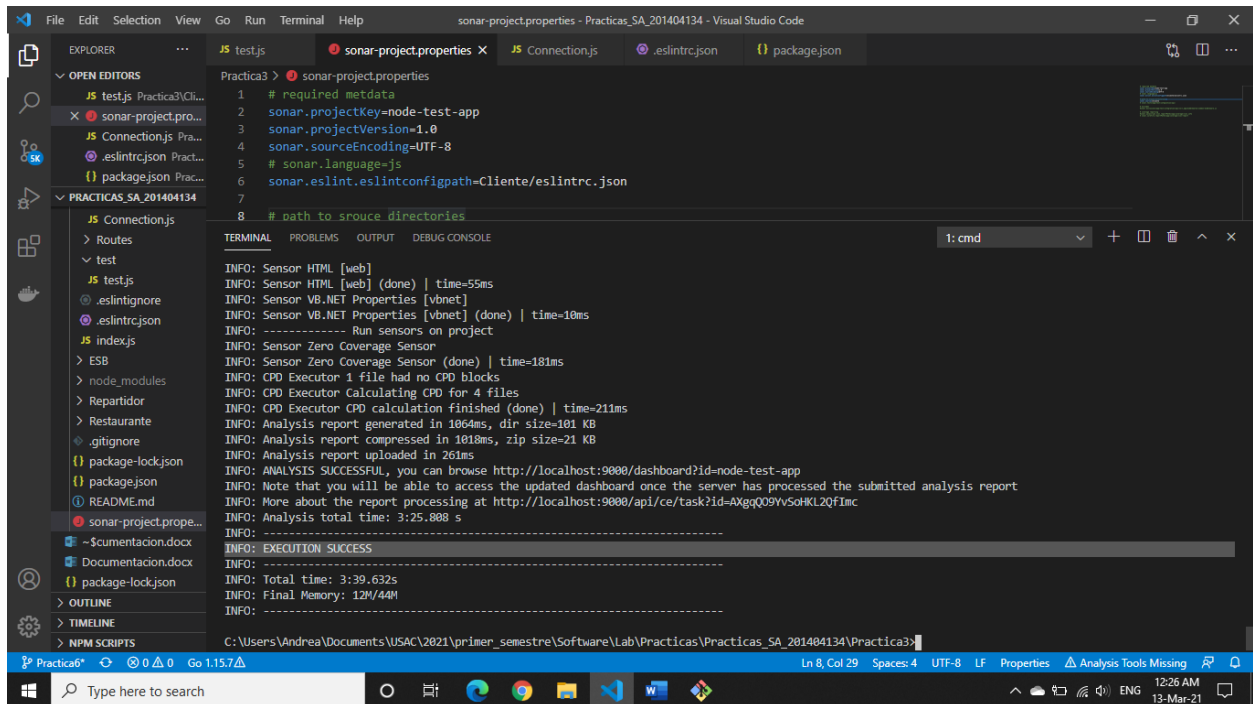
The screenshot shows the Visual Studio Code interface with the `sonar-project.properties` file open. The file contains the following configuration:

```
1 # required metadata
2 sonar.projectKey=node-test-app
3 sonar.projectVersion=1.0
4 sonar.sourceEncoding=UTF-8
5 # sonar.language=js
6 sonar.eslint.eslintconfigpath=Cliente/eslintrc.json
7
8 # path to source directories
9 sonar.sources=Cliente
10 # sonar.tests=app/test/integration/api/
11
12 # excludes
13 #sonar.excludes=app/test/integration/api/v1/*,app/middlewares/common-middleware.js
14
15 # coverage reporting
16 #sonar.javascript.lcov.reportPaths=app/coverage/lcov.info
17 # sonar.surefire.reportPaths=app/coverage/lcov-report
```

The terminal output shows the analysis results:

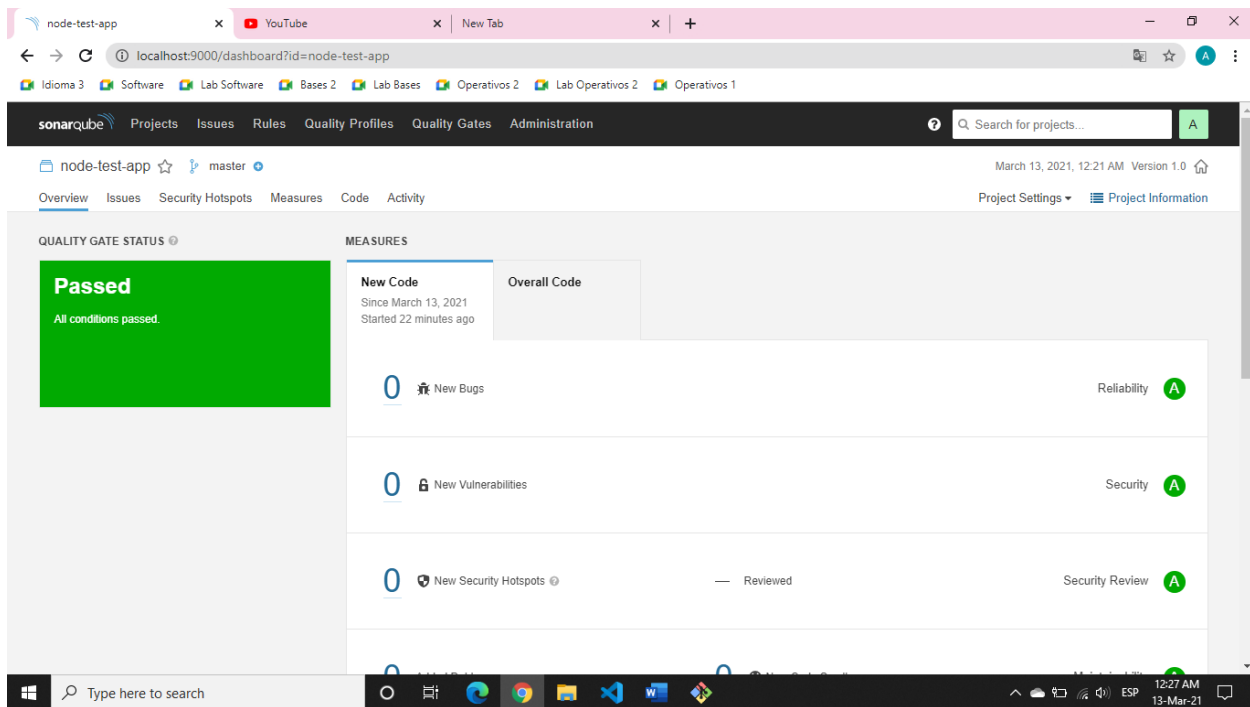
```
INFO: Analysis total time: 3:25.888 s
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 3:39.632s
INFO: Final Memory: 12M/44M
INFO: -----
```

Resultado del análisis



The screenshot shows the Visual Studio Code interface with the terminal output of the SonarQube analysis. The output includes the following information:

```
INFO: Sensor HTML [web]
INFO: Sensor HTML [web] (done) | time=55ms
INFO: Sensor VB.NET Properties [vbnet]
INFO: Sensor VB.NET Properties [vbnet] (done) | time=10ms
INFO: ----- Run sensors on project
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=181ms
INFO: CPD Executor 1 file had no CPD blocks
INFO: CPD Executor Calculating CPD for 4 files
INFO: CPD Executor CPD calculation finished (done) | time=211ms
INFO: Analysis report generated in 1064ms, dir size=101 KB
INFO: Analysis report compressed in 1010ms, zip size=21 KB
INFO: Analysis report uploaded in 261ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard?id=node-test-app
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=4xgq09VvSoHKL2QfImc
INFO: Analysis total time: 3:25.888 s
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 3:39.632s
INFO: Final Memory: 12M/44M
INFO: -----
```



Archivos de Sonarqube utilizados

