Test Scenario\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1 - Create a few UI automation tests (TestCafe)

2 - Create a few API Automation tests (SuperTest/Mocha)

3 - Implement Instanbul for api test coverage

4 - Implement Stryker.io for mutation testing to make sure our api test coverage is accurate

5 - Implement Lighthouse CI for In Sprint Performance testing

6 - Implement Artillery.io for in Sprint Load Testing

7 - Implement [https://clinicjs.org/](https://apc01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fclinicjs.org%2F&data=02%7C01%7Cvandana.kumari3%40wipro.com%7Ca36b09eb0faf43fc4c3208d7b1547c2a%7C258ac4e4146a411e9dc879a9e12fd6da%7C0%7C0%7C637172848723091021&sdata=RH%2BBQnUPHN%2BQCbO%2BbdVsSRdXK2Odu%2F0ZFOlvk%2FLlISc%3D&reserved=0) for monitoring against the load tests

Pre requisite \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*:

1) Node.js and NPM should install

1) Download Node js and NPM(node -v)

2) set NODE\_HOME Environment Variable

- Create folder under C drive C:\Shoppingtest\

- Clone the below project here

2) Clone : <https://github.com/jeffersonRibeiro/react-shopping-cart>

3) Follow below:

/\* First, Install the needed packages \*/

npm install

npm rebuild node-sass

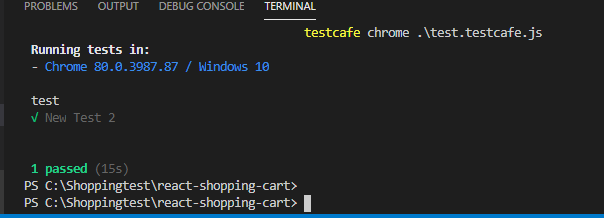
/\* Then start both Node and React \*/

npm start

1) TestCafe \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*UI **File name : test.testcafe.js**

1. Create project in your selected editor and run : testcafe chrome .\test.testcafe.js

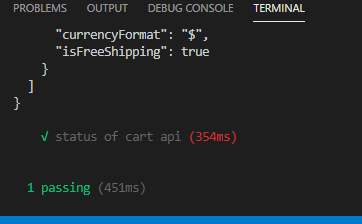
**Output**



2) Mocha\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*API test **file: test.js(under test folder)**

1. npm test

**OutPut:**

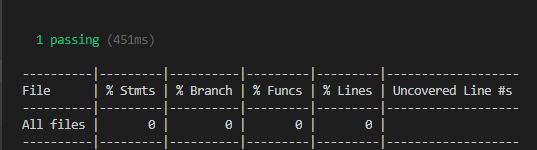


3) Instanbul \*\*\*\*\*\*\*\*test coverage

**Uses : Code overage**

* command : npm i --save-dev nyc
* add in package.json : "test": "nyc mocha"
* HTML report
* add : "nyc --repoter=lcow --repoter=text-lcow nmp test"
* it will create coverage folder(index.html)
* run command : npm test

**OutPut**

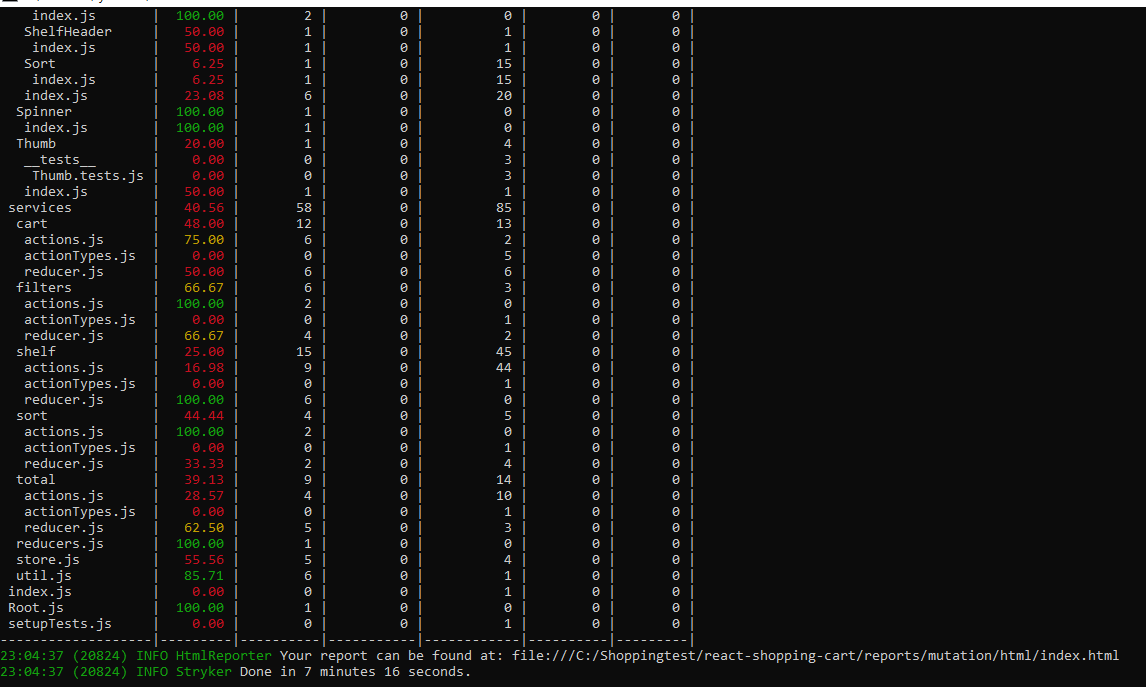


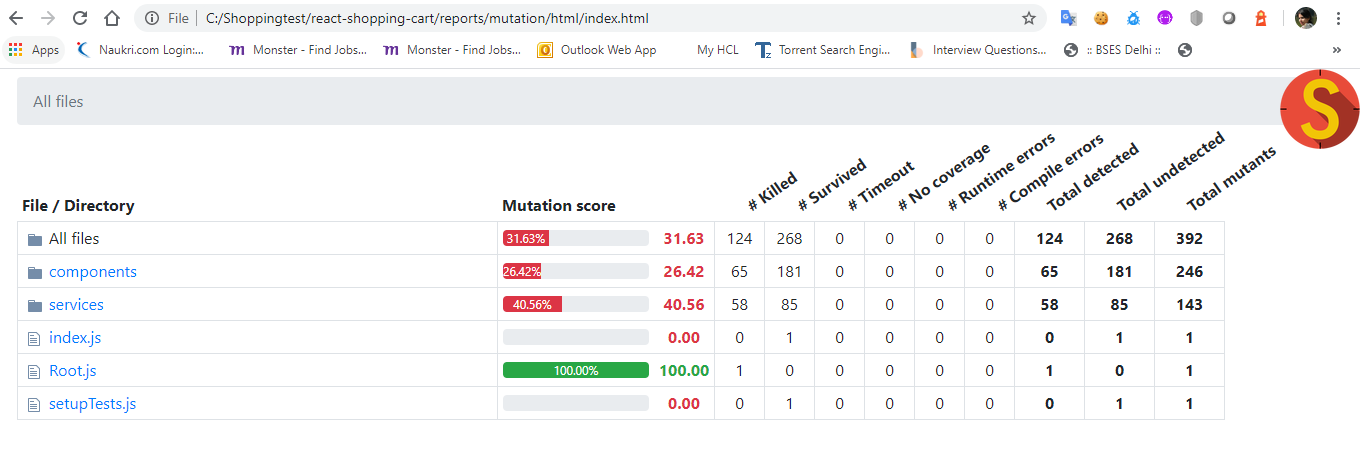
4)Stryker.io \*\*\*\*\*\*\*\*\*\*\*\*\*\*Mutation testing

**Uses : Mutation testing** –.

* install - npm install -g stryker-cli
* config : stryker init - stryker.conf.js created
* run : stryker run

**OutPut :**





**Find the report here** : C:\Shoppingtest\react-shopping-cart\reports\mutation\html\index.html

5)Lighhouse CI \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Performance, Quality, correctness testing

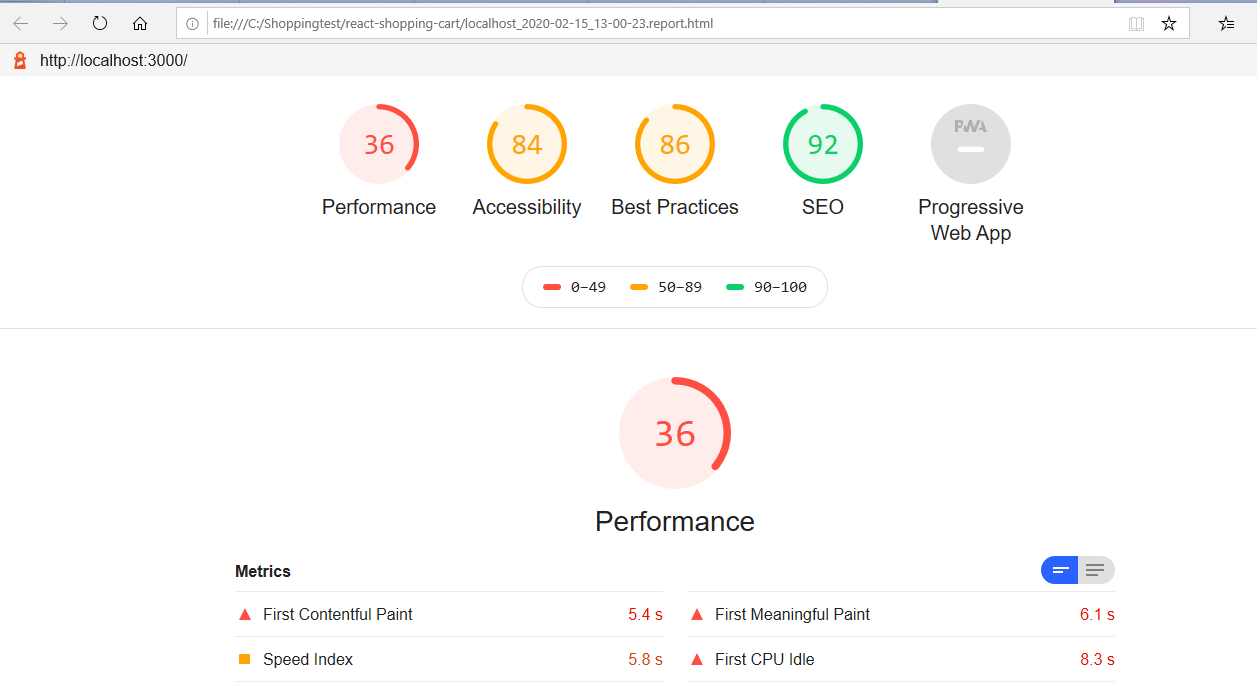
1. run command : npm install -g lighthouse
2. run command : lighthouse <http://localhost:3000/>
3. go to C:\Shoppingtest\react-shopping-cart path and check ./report.html file will be generated ( i.e : localhost\_2020-02-15\_13-00-23.report.html)



**Output**



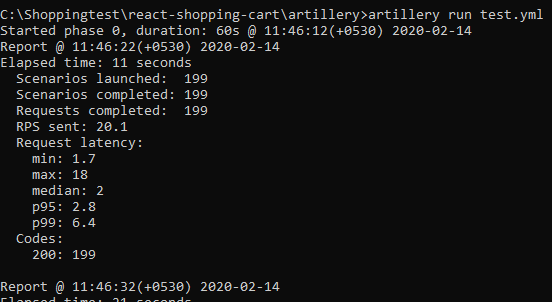
**HTML report Output**



6)Artillery.io \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* In sprint Load testing File name : test.yml

1. Install : npm install -g artillery
2. To check that the installation succeeded, run: artillery –V
3. Create yml file (artillery-> test.yml)
4. run test script: artillery run filename.yml -

**OutPut :**



Or run

1. artillery quick -count 10 -n 20 http://localhost:3000/

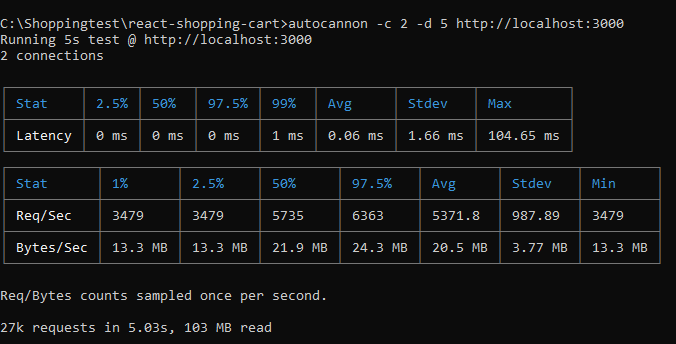
**response description:**

1. **Scenarios launched** is the number of virtual users created in the preceding 10 seconds (or in total)
2. **Scenarios completed** is the number of virtual users that completed their scenarios in the preceding 10 seconds (or in the whole test). Note: this is the number of completed sessions, not the number of sessions started and completed in a 10 second interval.
3. **Requests completed** is the number of HTTP requests and responses or WebSocket messages sent
4. **RPS sent** is the average number of requests per second completed in the preceding 10 seconds (or throughout the test)
5. **Request latency** is in milliseconds, and p95 and p99 values are the 95th and 99th [percentile](https://en.wikipedia.org/wiki/Percentile) values (a request latency p99 value of 500ms means that 99 out of 100 requests took 500ms or less to complete).
6. **Codes** provides the breakdown of HTTP response codes received.

7)Clinicjs.org \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* monitoring against Load testing

1. npm install -g clinic
2. For cannon report : npm i autocannon -g
3. npm i autocannon --save
4. autocannon <http://localhost:3000>

**OutPut**



1. run command : clinic doctor --autocannon [ / --method POST ] -- node server.js