Atlantbh d.o.o Sarajevo

Investigate Google Geocoding API service and create test plan

(Atlantbh d.o.o Sarajevo testing task)

Author:

Meris Bihorac

Sarajevo, November 2017.

Content

ABH Testing Task

Investigate Google Geocoding API service and create test plan

https://developers.google.com/maps/documentation/geocoding/intro#Geocoding

1. Write test cases for this service
2. From all the test cases, identify what you think represents a Smoke Test
3. Identify at least one positive and one negative test for both forward and reverse geocoding
4. With a tool or programming/scripting language (JMeter, soapUI, Ruby, Java…), automate the smoke test from #2
5. Create Github repository and push your assignment solution with the documentation on how to run the automated tests there

Note: Write test cases/test script for API (NOT for UI that is also available in documentation above)

Introduction

Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers on a map, or position the map.

Reverse geocoding is the process of converting geographic coordinates into a human-readable address.

You can also use the Google Maps Geocoding API to find the address for a given place ID.

The Google Maps Geocoding API provides a direct way to access these services via an HTTP request. [1]

For testing Geocoding API services it will be used SoapUI 5.3.0 application and testing environment:

- **Computer:** *Toshiba Satellite C660*
- **Processor**: *Intel Core i3 CPU M380 2.53GHz, x64*
- **OS**: *Windows 10 Pro, 64 bit*
- **RAM**: *3GB*

1. Test Cases

After running SoapUI 5.3.0 there is basic scenario:

1. File/Create new REST project
2. Enter URL request that demonstrates using the JSON or XML flags: https://maps.googleapis.com/maps/api/geocode/json https://maps.googleapis.com/maps/api/geocode/xml [1]
3. Add required parameters (address or components, key, etc.) to the parameter table
4. Create Test Case
5. Add Test Step (inset value for parameters)
6. Add assertion

1.1. Test Case without API key

Without API key:

Testing worked many times, with status “OK”, as it’s shown on picture below:

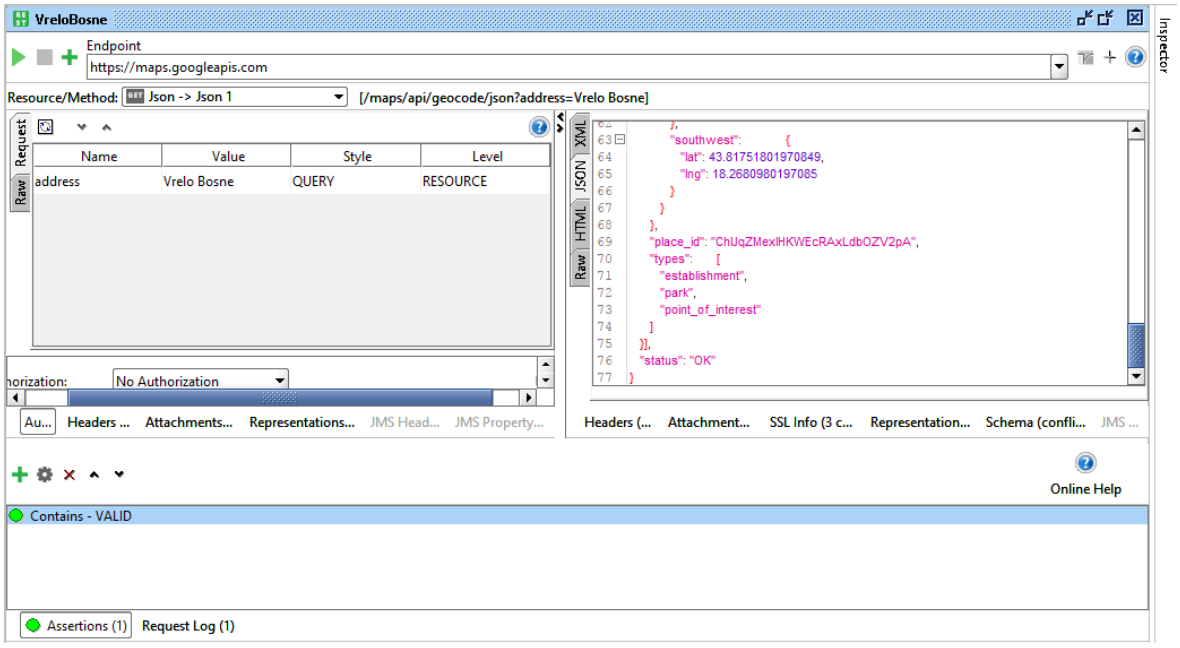

Figure 1 Response without using API key, status „OK“

After many requests there is JSON response below with status “OVER_QUERY_LIMIT”.

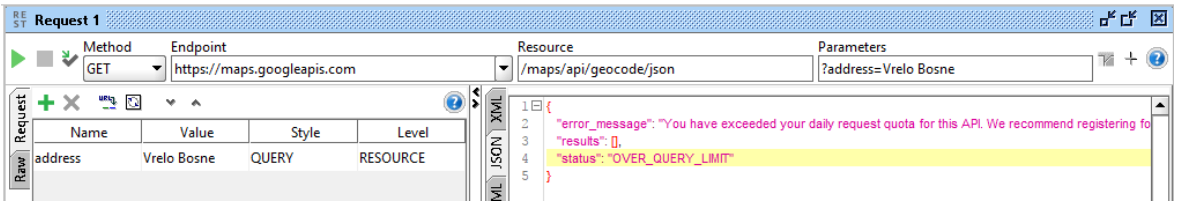

Figure 2 Response without using API key, status „OVER_QUERY_LIMIT“


```
{
   "error_message": "You have exceeded your daily request
quota for this API. We recommend registering for a key at the
Google Developers Console:
https://console.developers.google.com/apis/credentials?projec
t=_",
   "results": [],
   "status": "OVER_QUERY_LIMIT"
}
```


Conclusion: API key should be created!

1.2. Test Case with API key, geocoding example

In link below it is described how to create API key that is needed for testing geocoding.

https://developers.google.com/maps/documentation/geocoding/get-api-key

After getting API key, tester is able to complete request by putting key value parameter.

Below are shown links for getting JSON and XML response for a query “1600 Amphitheatre Parkway, Mountain View, CA”.

For JSON :

https://maps.googleapis.com/maps/api/geocode/json?address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&key=YOUR_API_KEY

For XML:

https://maps.googleapis.com/maps/api/geocode/xml?address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&key=YOUR_API_KEY

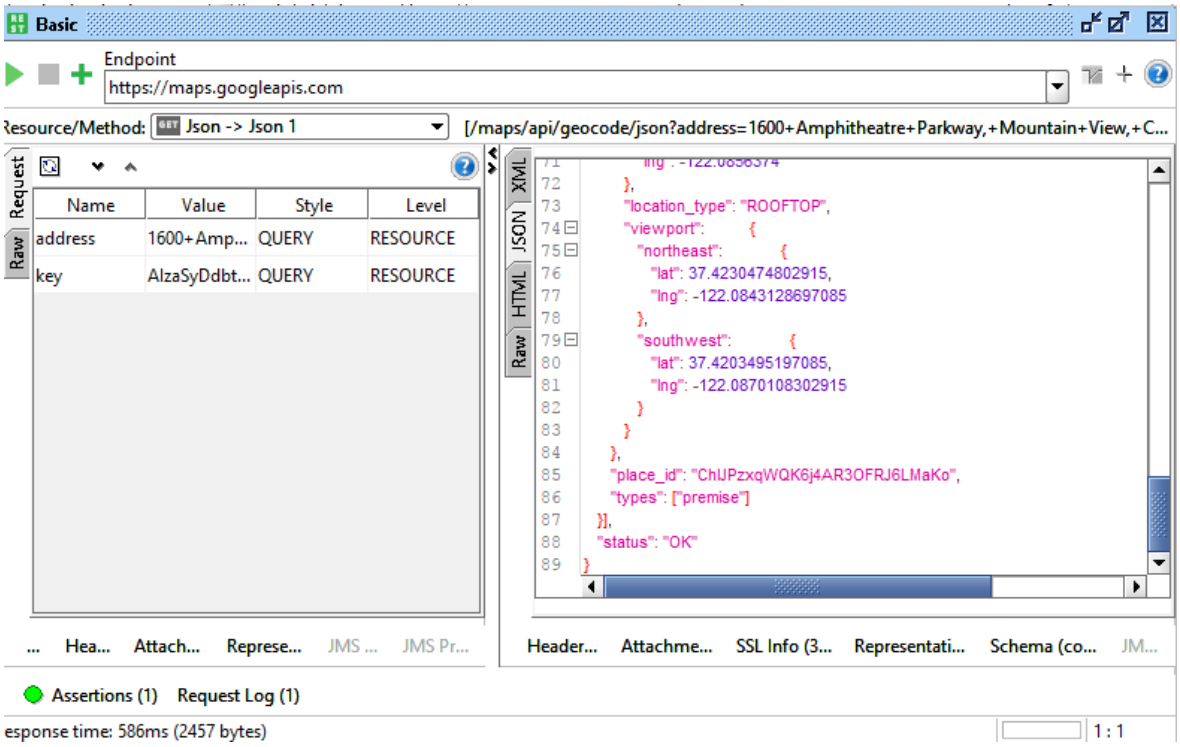

Figure 3 JSON response for basic query defined above

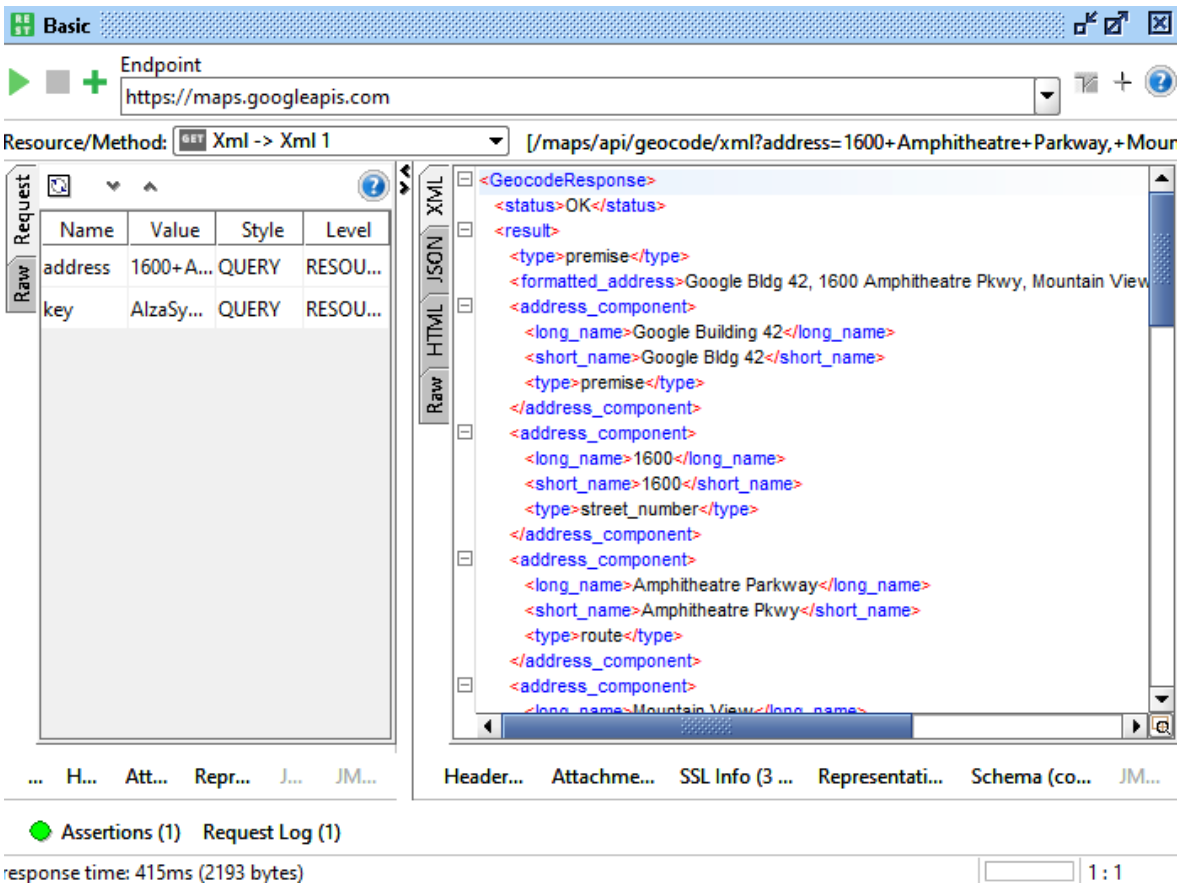

Figure 4 XML response for basic query defined up

Previously displayed images show JSON and xml responses in the SoapUI environment. Although the way of viewing is different, the responses have the same meaning. In the other test cases JSON responses are used (subjective choice).

1.3. JSON3steps test case

Three test steps with queries:

1. https://maps.googleapis.com/maps/api/geocode/json?address=Vrelo Bosne&key=YOUR_API_KEY – VreloBosne step
2. https://maps.googleapis.com/maps/api/geocode/json?address=Envera Sehovica&key=YOUR_API_KEY –Envera Sehovica step
3. https://maps.googleapis.com/maps/api/geocode/json?address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&key=YOUR_API_KEY –basic step

Assertion for all steps is: Contains “status”=”OK”, or simple “OK”.

Right click on Test Steps - > Show TestCase Editor

Left click on Runs this testcase & Using option Test On Demand it is possible to get result that is shown on picture below.

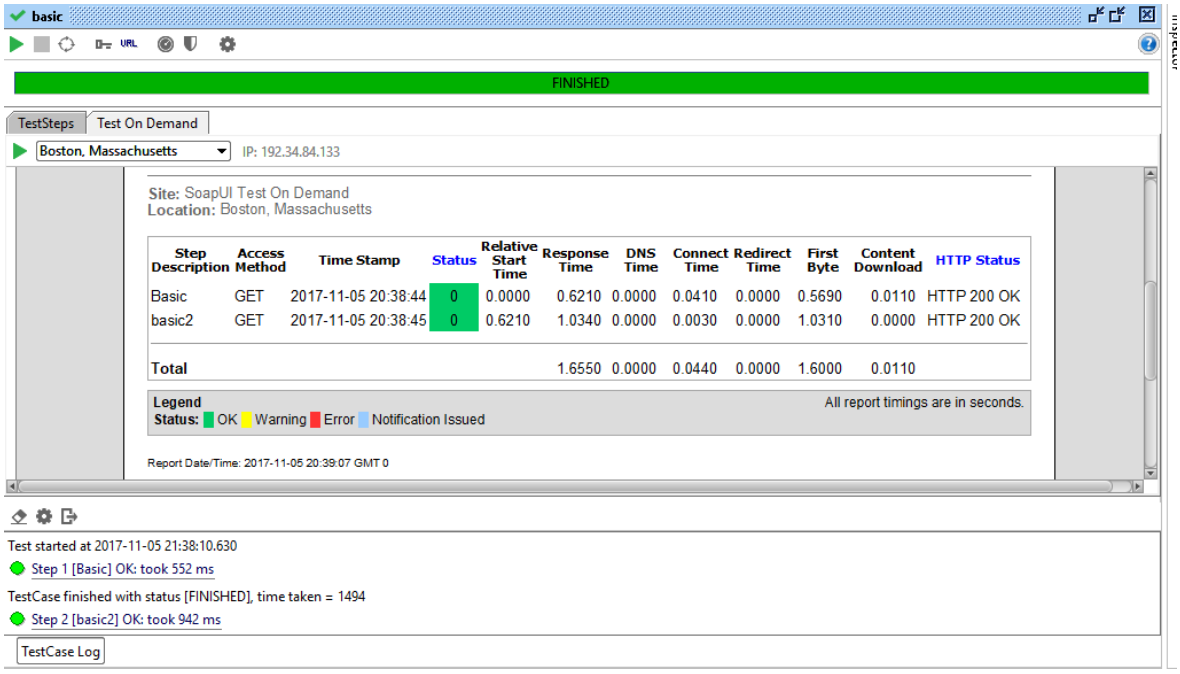

Figure 5 JSON3steps test case, Test On Demand

It is also possible to run option “New Load Tests”, for more information and detailed analyst of results, as it’s shown on picture below.

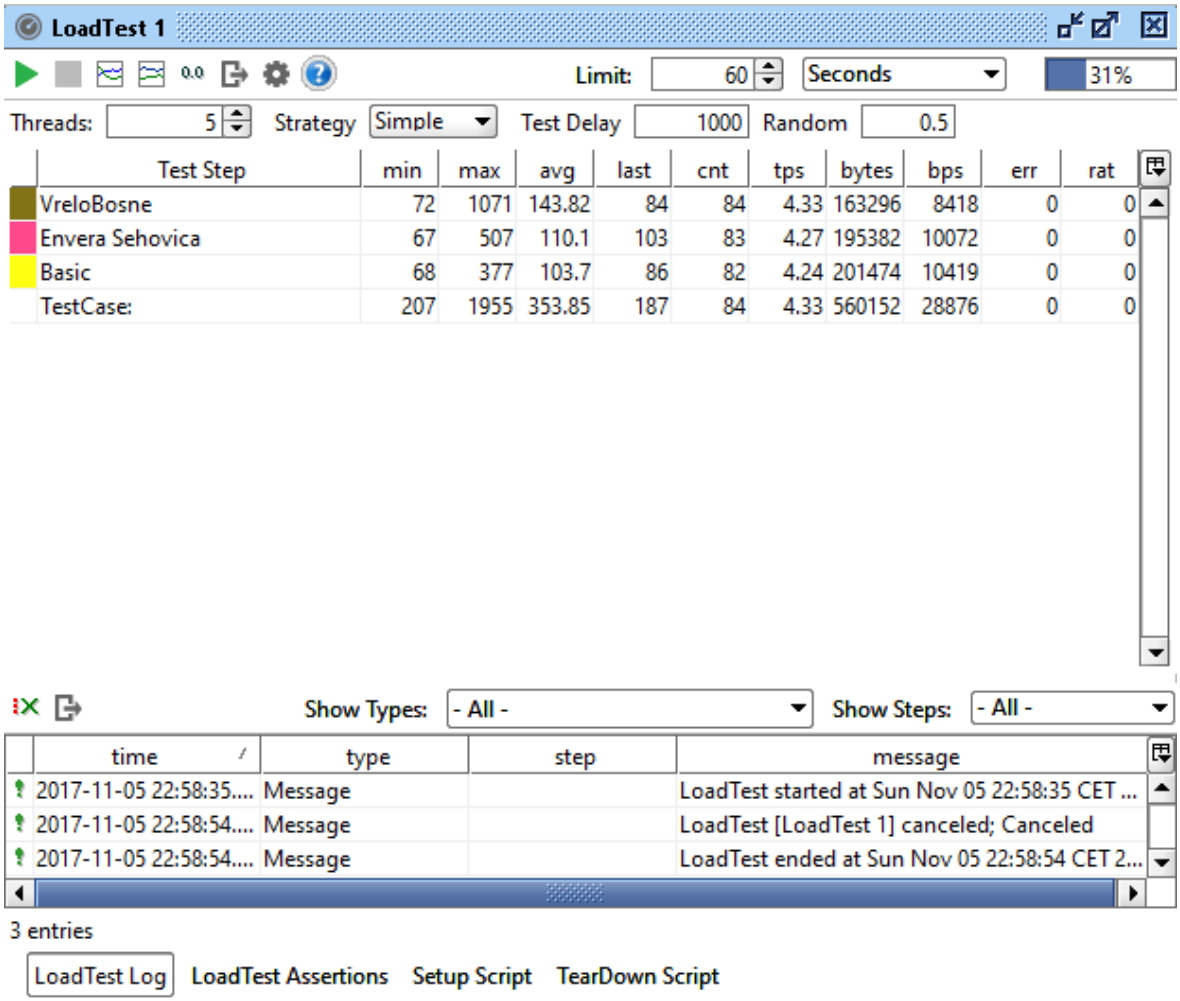

Figure 6 Load Test for JSON3steps test case

1.4. StatusRequest test case

Test steps in StatusRequest test case, match these queries:

https://maps.googleapis.com/maps/api/geocode/json?address=Prelijepa&key=YOUR_API_KEY – “Status”:”ZERO_RESULTS”

https://maps.googleapis.com/maps/api/geocode/json?key=YOUR_API_KEY – “Status”:”INVALID_REQUEST”

https://maps.googleapis.com/maps/api/geocode/json?address=VreloBosne&key=InvalidKEY - “Status”:”REQUEST_DENIED”

These were three of six possible status, that are:

1. "OK" indicates that no errors occurred; the address was successfully parsed and at least one geocode was returned.
2. "ZERO_RESULTS" indicates that the geocode was successful but returned no results. This may occur if the geocoder was passed a non-existent address.
3. "OVER_QUERY_LIMIT" indicates that you are over your quota.
4. "REQUEST_DENIED" indicates that your request was denied.
5. "INVALID_REQUEST" generally indicates that the query (address, components or latlng) is missing.
6. "UNKNOWN_ERROR" indicates that the request could not be processed due to a server error. The request may succeed if you try again.

For given queries and each Test Step, there is result shown on picture below. There are used Contains assertion with status value that are expected in response.

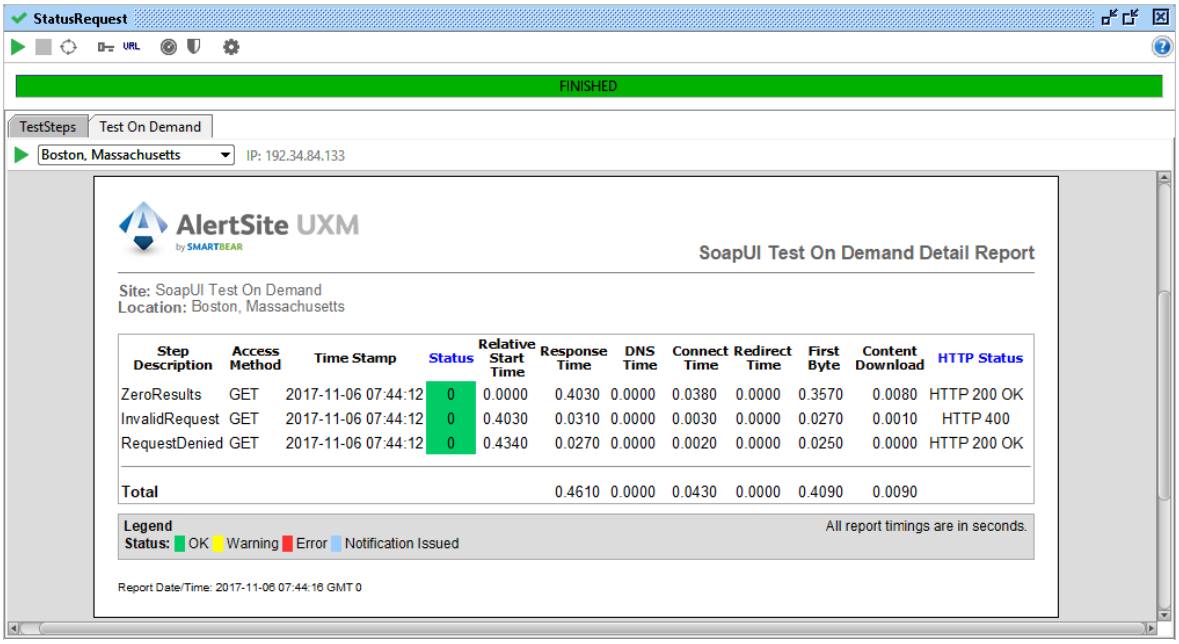

Figure 7 Test On Demand for StatusRequest steps

Picture with Load Test is shown below:

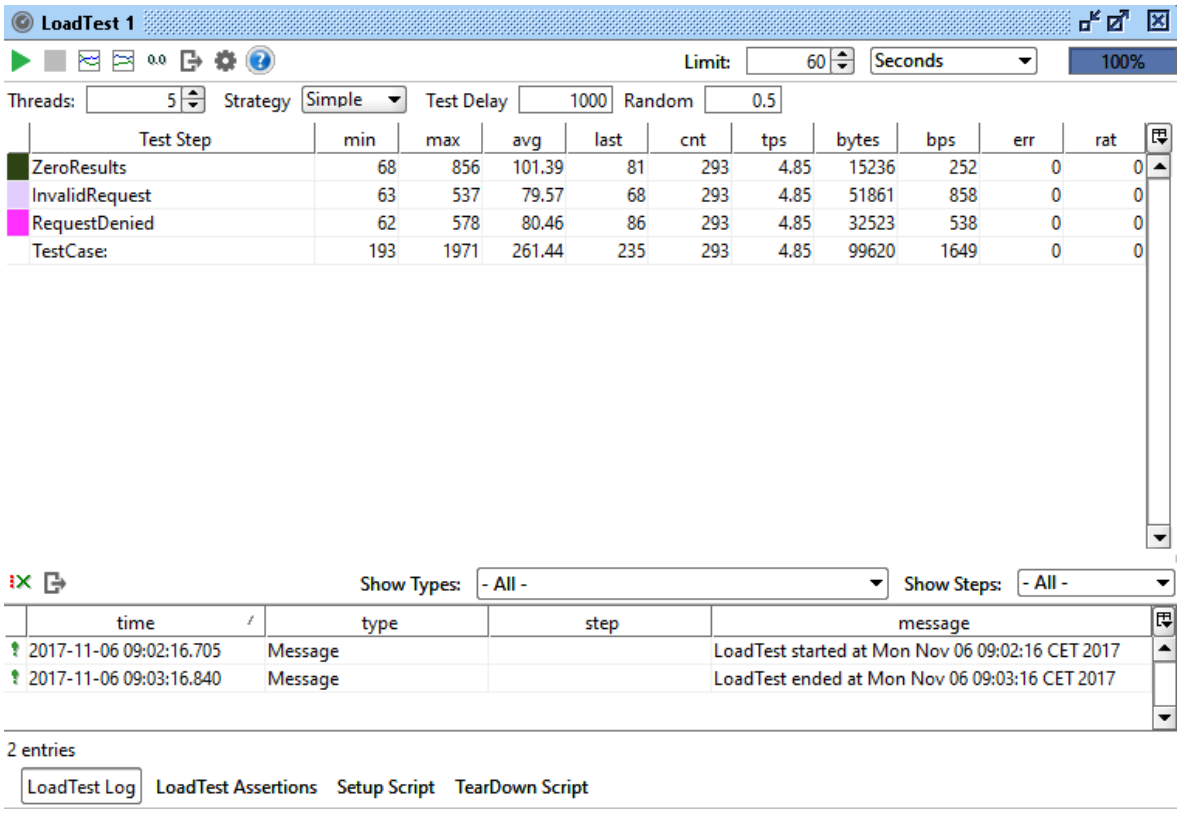

Figure 8 Load Test for each Status Test Step, duration 60 seconds

Load Test shows us that there are no errors during 60 seconds of testing. It's also possible to show results on graphic, as it's shown on picture below.

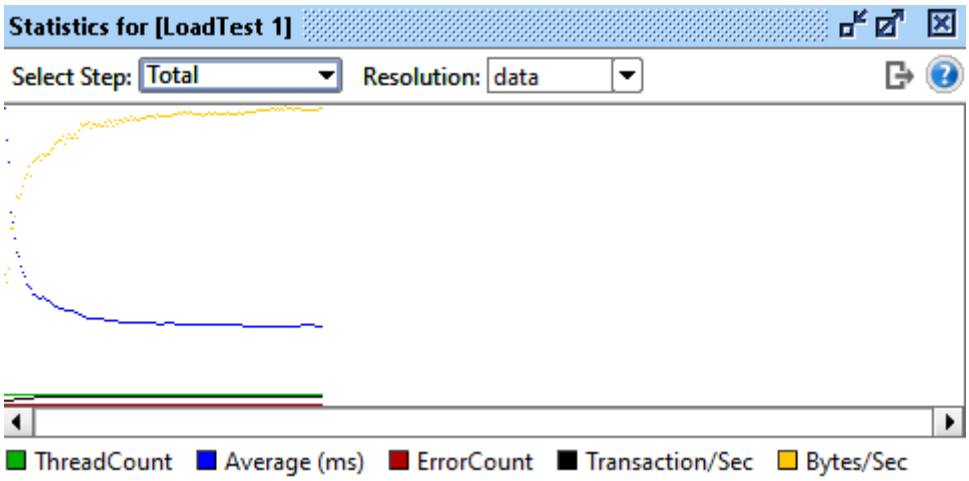

Figure 9 Graphic that shows statistic for Load Test

1.5. allParam test case

In this query there are 6 parameters that are added:

Name	Value
address	1600 Amphitheatre Pkwy
components	administrative_area:TX|country:US
key	YOUR_API_key
language	en
region	US
postal_code	94043

Picture with response for given request is shown below:

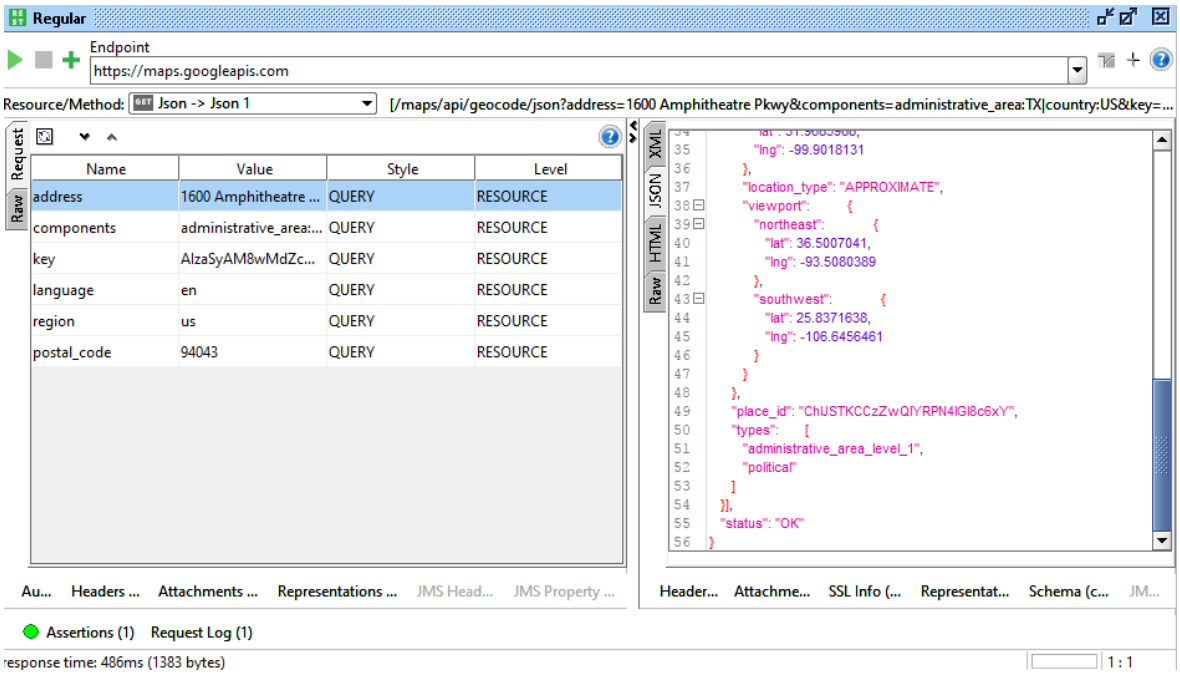

Figure 10 allParam test case and response

2. Smoke Test

Smoke Testing, also known as “Build Verification Testing”, is a type of software testing that comprises of a non-exhaustive set of tests that aim at ensuring that the most important functions work. The results of this testing is used to decide if a build is stable enough to proceed with further testing.

The term ‘smoke testing’, it is said, came to software testing from a similar type of hardware testing, in which the device passed the test if it did not catch fire (or smoked) the first time it was turned on. [2]

For all test cases written in FirstHeading, there is one test case that includes most of parameters. The test case with most parameters is named “allParam test case”. It’s enough to run this test and check in any error appeared. If there is no error, service is stable enough to proceed with further testing.

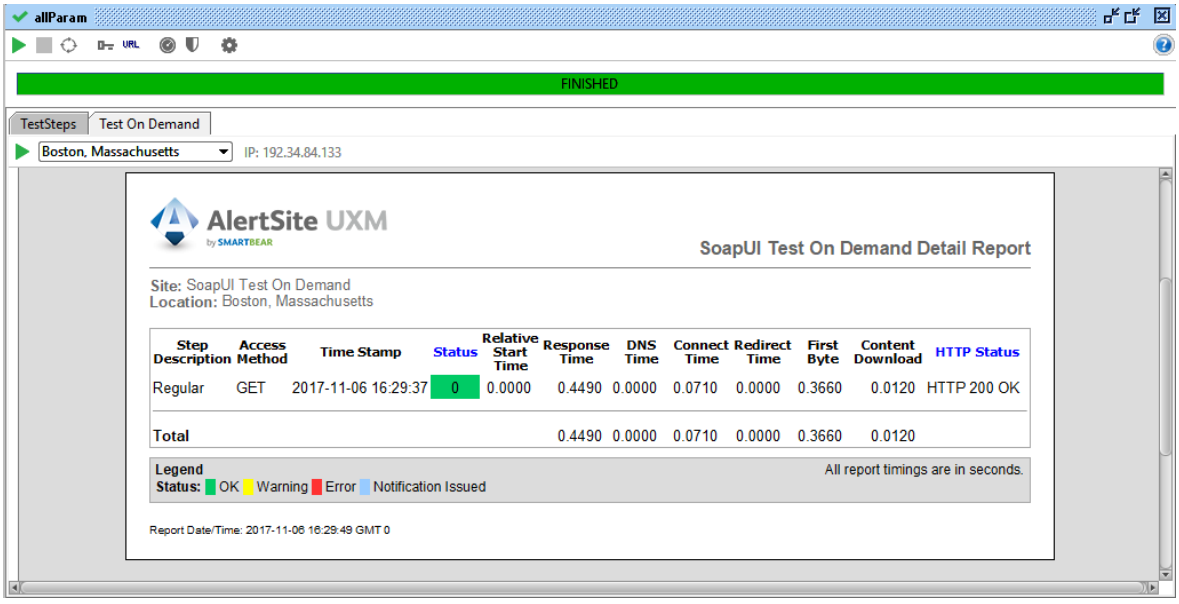

Figure 11 Test On Demand for „allParam test case“

On Figure 11 there is Test On Demand that shows allParam test case with status „OK“.

In this Heading it’s concluded that allParam test case could be Smoke Test.

3. Positive and Negative test

For **positive and negative forward geocoding** it will be used allParam test case. First positive test step is already tested. Negative test step will include query with wrong parameters and same assertion as positive test step. The result of these step tests is below.

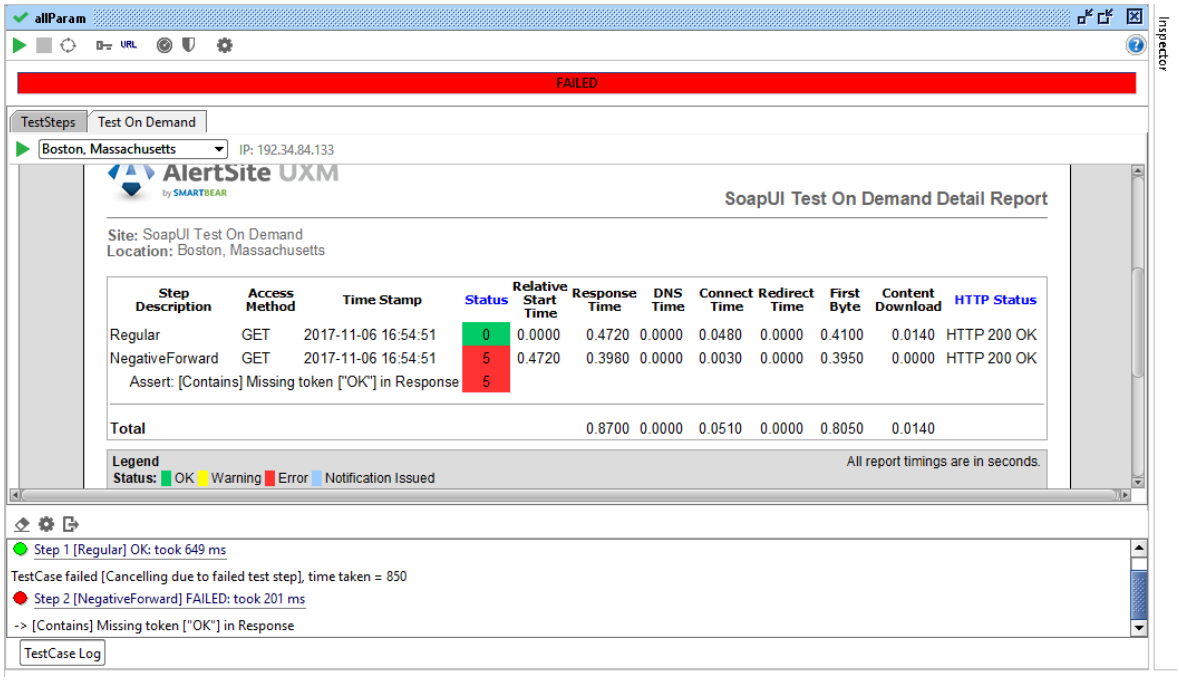

Figure 12Negative and positive forward geocoding test

Table 1Query with wrong parameter values

Name	Value
address	Prelijepa
components	ba
key	YOUR_API_key
language	sa
region	ba
postal_code	213412

3.1. Positive and negative reverse geocoding

The term geocoding generally refers to translating a human-readable address into a location on a map. The process of doing the opposite, translating a location on the map into a human-readable address, is known as reverse geocoding.

Required parameters in a reverse geocoding request:

- latlng — The latitude and longitude values specifying the location for which you wish to obtain the closest, human-readable address.
- key — Your application's API key. This key identifies your application for purposes of quota management. [1]

Steps for creating REST project are shown in FirstHeading, only diference are used parameters.

Positive reverse geocoding:

Parameters and query for positive reverse geocoding are shown below.

https://maps.googleapis.com/maps/api/geocode/json?latlng=40.714224,-73.961452&key=YOUR_API_KEY

Name	Value
latlng	40.714224,-73.961452
key	YOUR_API_KEY

Negative reverse geocoding:

Parameters and query for negative reverse geocoding are shown below.

https://maps.googleapis.com/maps/api/geocode/json?latlng=100024,-7323423.961452&key=YOUR_API_KEY

Name	Value
latlng	100024,-7323423.961452
key	YOUR_API_KEY

Results for positive and negative reverse geocoding request is shown on Figure 13.

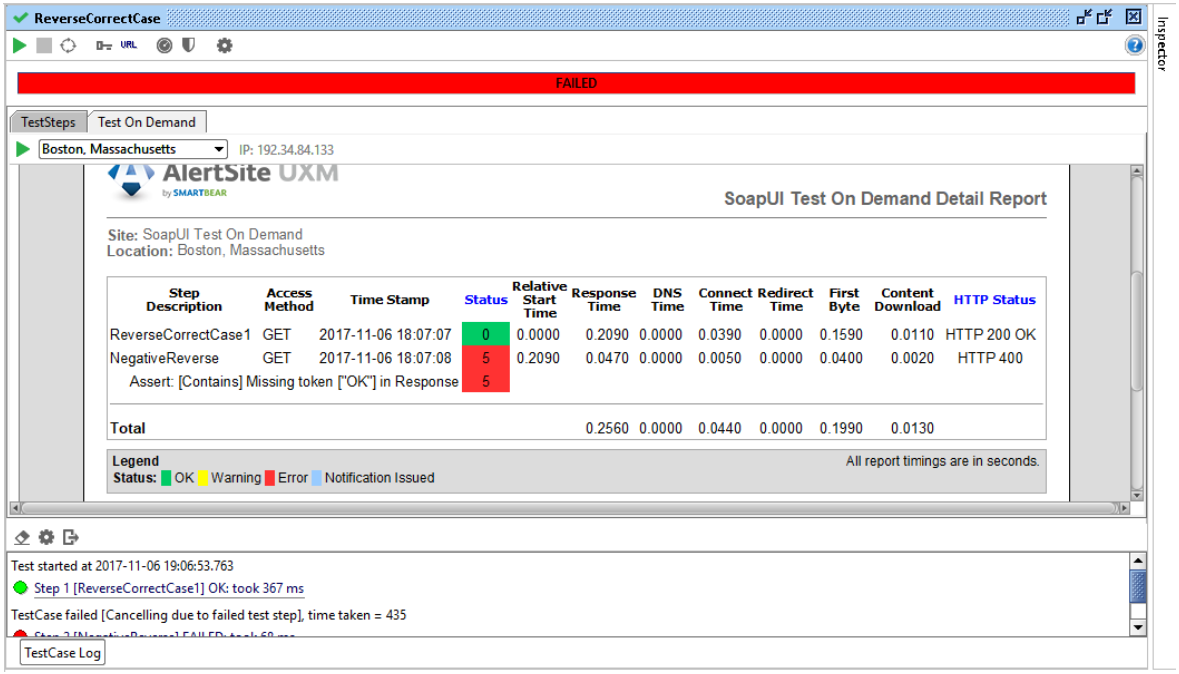

Figure 13 Positive and negative test steps for reverse geocoding

Assertion for both steps was the same (Contains “Status”:”OK”).

4. Smoke test automation

Right click on allParam test case, and launch TestRunner to create report.

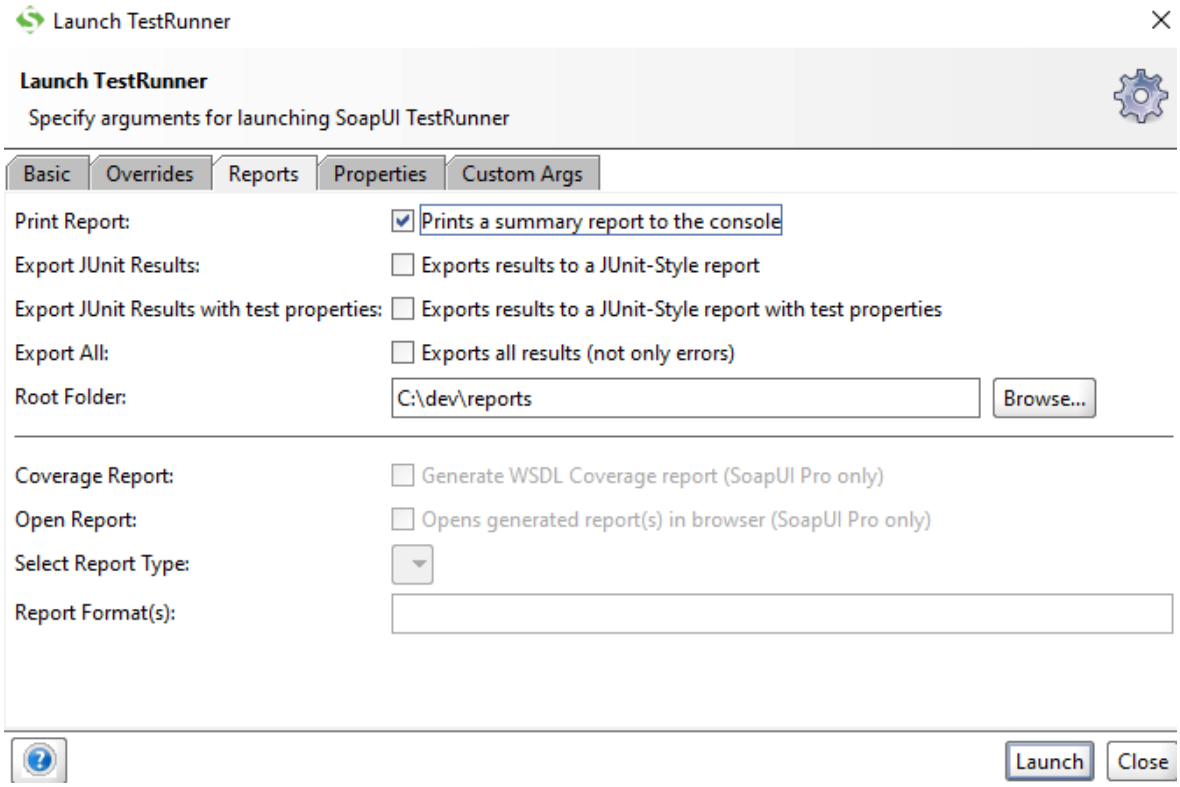

Figure 14 Launching TestRunner

TestRunner is configured as it is shown on Figure 14 and Figure 15.

In this creating report of allParam that represents Smoke Test is automated.

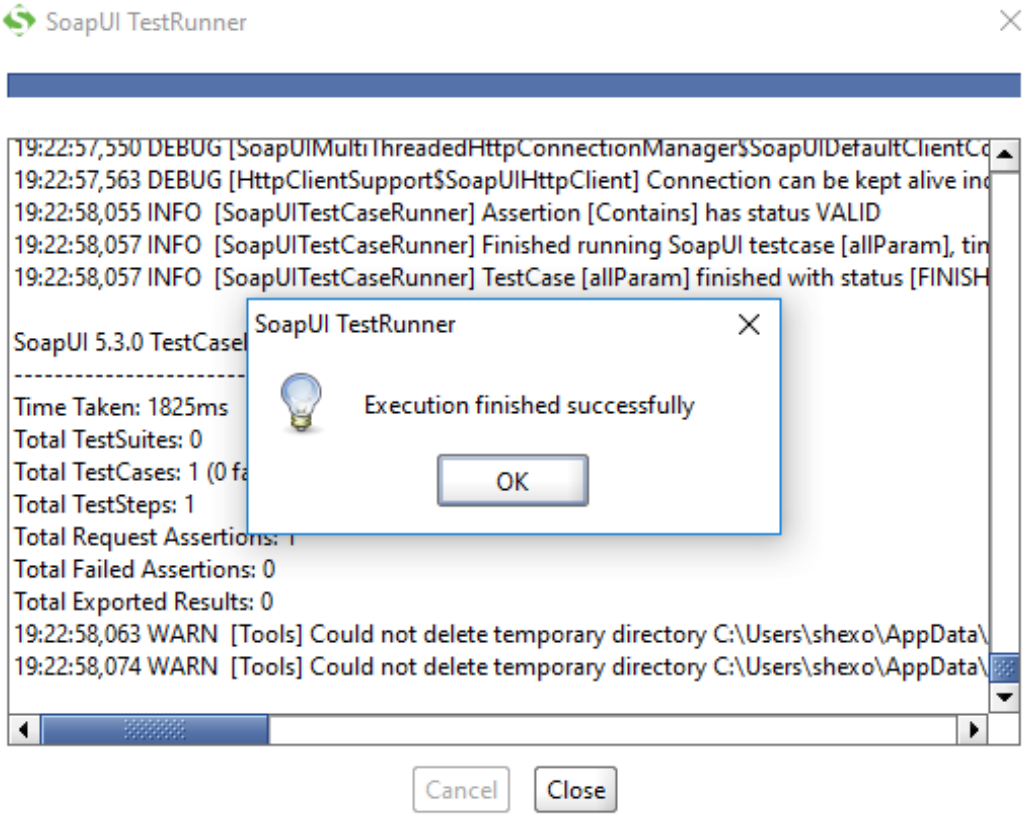

Figure 15 TestRunner execution finished successfully

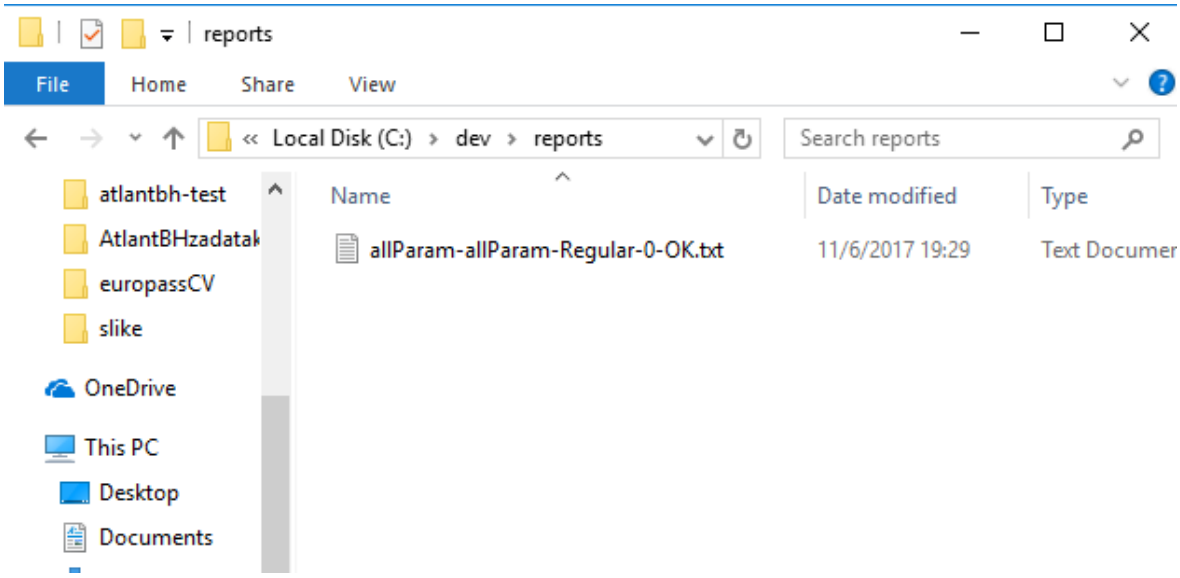

Figure 16 Automatically generated text report

5. Github repository for solution

This project is pushed in https://github.com/merisbihorac/atlantbh-test repository. Below are some steps for using git.

- Install git, make github acc and add new repository

```
# 1. Creating project, beginning initialization
echo "# atlantbh-test" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/merisbihorac/atlantbh-test.git
git push -u origin master


# 2. Updating changes
# Go to local project directory
git add . # adds all changes
git commit -m "commit message"
git push -u origin master # push changes in created repository
```

Conclusion

SoapUI 5.3.0 is good and simple tool for testing geocoding services. By using REST protocol it is possible to reach endpoint (geocoding API service) and get JSON or XML responses. First Heading shows that it’s possible to create multiple test steps, that could be ran automatic. It’s presented what Smoke Test should be, and with using TestRunner creating reports is automated. Git and github are used for pushing this project in given repository. This project could be done in several testing tool and scripting language as JMeter, Java, Ruby, etc. And there is always possibility to upgrade this project to better level, by adding more test cases and by using more testing tools.

Bibliography

[1] [Online].Available:
https://developers.google.com/maps/documentation/geocoding/intro.

[2] [Online]. Available: http://softwaretestingfundamentals.com/smoke-testing/.