#---------------------------------------------------------------------------

# Test Script Description

#---------------------------------------------------------------------------

Test tool used – JMeter v5.4.3

Test script name - AssurityConsulting\_PerfAssignment.jmx

Directory Structure:

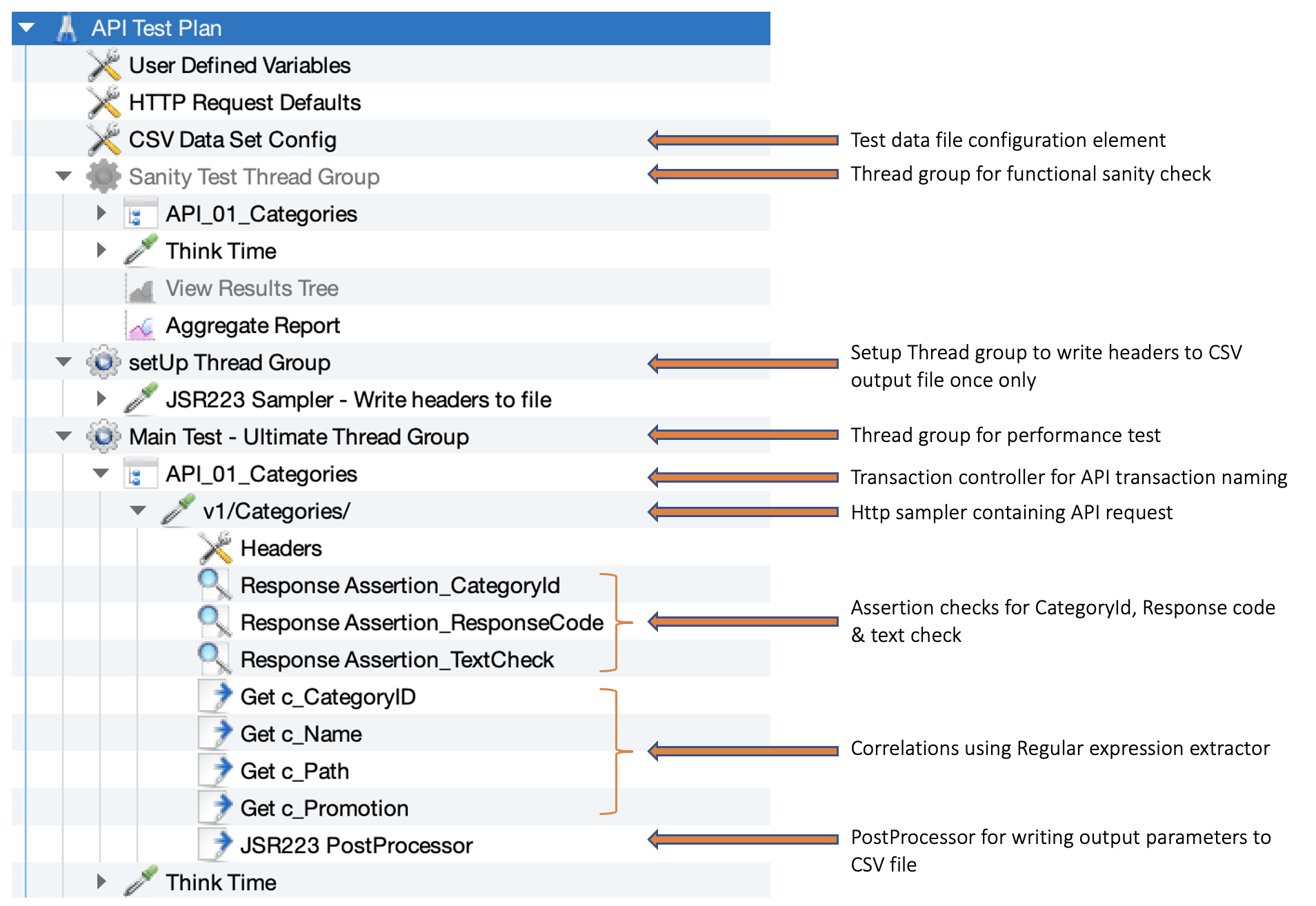
Chart

Description automatically generated

#---------------------------------------------------------------------------

Overview of Test Script

#---------------------------------------------------------------------------



#---------------------------------------------------------------------------

Test Script Elements, and explanation on why it is used

#---------------------------------------------------------------------------

---------------------------------------------------------------------------

User Defined Variables – total 4 variables defined globally

Graphical user interface, text, application, email

Description automatically generated

---------------------------------------------------------------------------

HTTP Request Default – Used to send all requests in the test script to the same domain or server. To add a new request, we just need to specify the path, method type and parameters to be sent with the request, no need to add protocol, Server name/ip and port.

Graphical user interface, text, application

Description automatically generated

---------------------------------------------------------------------------

CSV Data Set Config – use to give input test data file to test script

Test data file name - CategoryID.txt

Parameter name - p\_CategoryID

Graphical user interface, application

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

---------------------------------------------------------------------------

Thread Group –

Total users – 5 users (half the count of Category Ids available in the test data file)

Ramp-up rate – 1 user per second

Steady states – 60 seconds

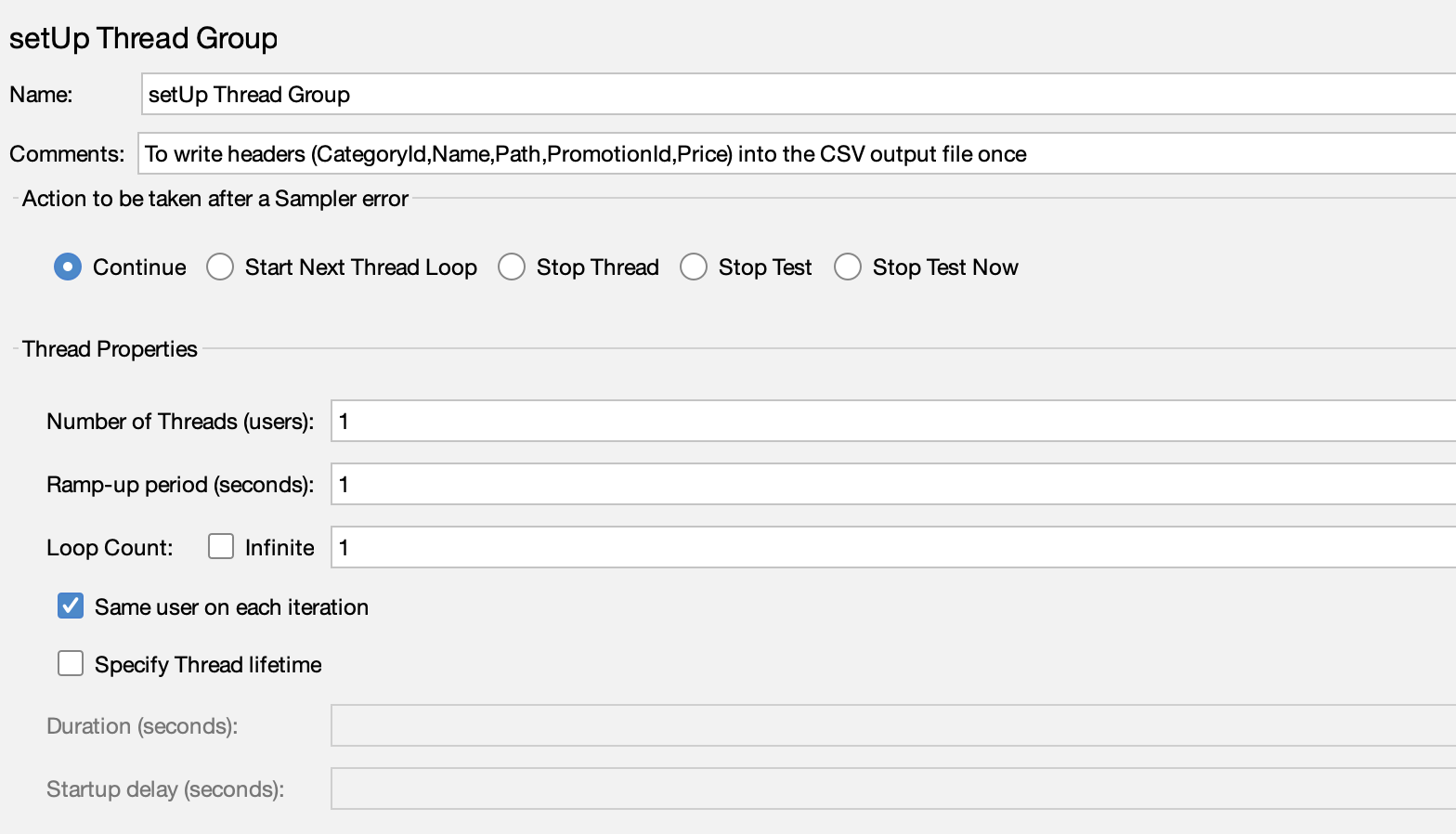
Ramp-down rate – 1 user per second

Application, table

Description automatically generated

---------------------------------------------------------------------------

Setup Thread Group – to write headers into CSV file once at the start of the test and executed with single user for single iteration.



---------------------------------------------------------------------------

Transaction Controller – for API transaction naming

Transaction name – API\_01\_Caterories

Graphical user interface, text, application, email

Description automatically generated

---------------------------------------------------------------------------

HTTP Sampler – to generate API request

Method – Get

Category ID replaced with parameter ${p\_CategoryID}

Protocol & Server name – defined in the HTTP Request Default

Graphical user interface, text, application

Description automatically generated

---------------------------------------------------------------------------

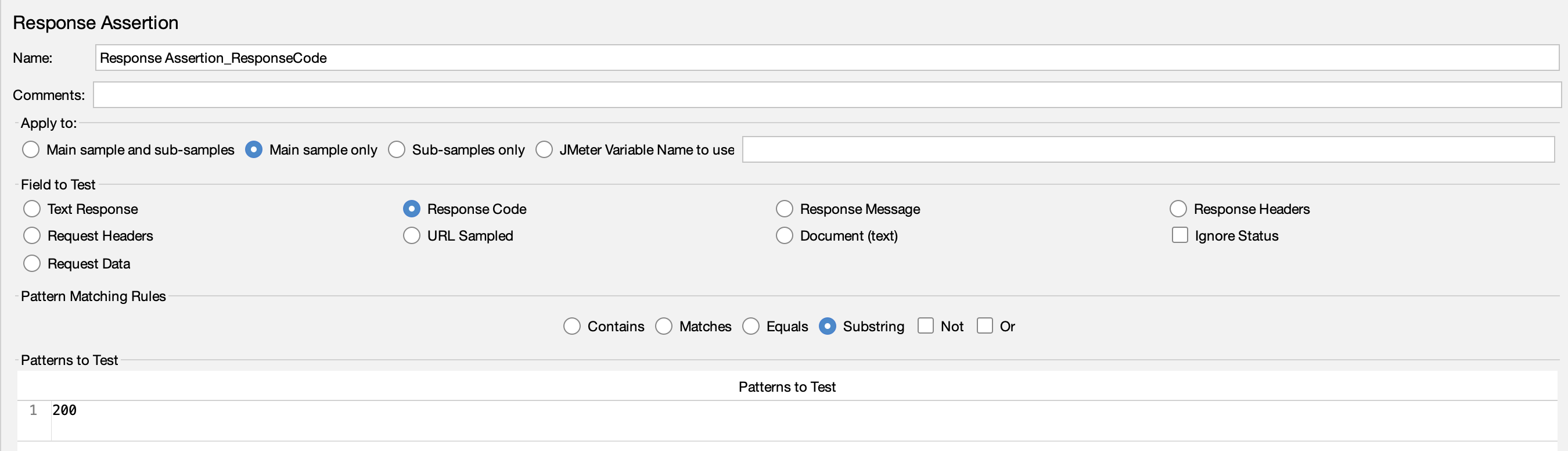
Response Assertions –

Parameter check – Category Id

Graphical user interface, text, application, email

Description automatically generated

Response code – 200



Text check - "CanRelist": true

Graphical user interface, text, application, email

Description automatically generated

---------------------------------------------------------------------------

Regular Expression Extractor – to correlate and capture the data from response.

Total 4 Regular Expression Extractors are applied to the correlation values of capture Category ID, Name, Path and Promotion

Graphical user interface, text, application, email

Description automatically generated

---------------------------------------------------------------------------

JSR223PostProcessor – used for customized coding in Groovy language,

1. To capture Promotion Ids and respective price from Promotion JSON collection.
2. Captured data written to file.

Diagram

Description automatically generated

---------------------------------------------------------------------------

Flow Control Action – used to provide a pause between the execution of two elements

Think time is set to 29.77 secs, and its value is calculated using Littles Law,

No. of Users =(Response Time + Think Time) \* Transactions per Second

As per our requirement we have,

No. of Users = 5

Response time for single transaction (avg.) = 0.23 secs

Transaction per Second = 0.16

So, Think Time = (No. of Users / TPS) - Response Time = ~29.77

Graphical user interface, application

Description automatically generated

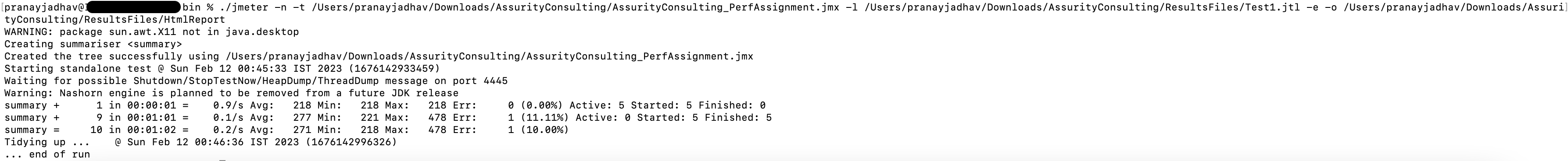
#---------------------------------------------------------------------------

# Steps to execute test plan in Non-GUI mode

#---------------------------------------------------------------------------

Execute below command from terminal (MacOS)/ powershell (Windows)

*./jmeter -n -t /{script directory path}/AssurityConsulting/AssurityConsulting\_PerfAssignment.jmx -l /{script directory path}/AssurityConsulting/ResultsFiles/Test1.jtl -e -o /{script directory path}/AssurityConsulting/ResultsFiles/HtmlReport*



#---------------------------------------------------------------------------

# JMeter test raw result file

#---------------------------------------------------------------------------

Result file location and name - */{script directory path}/AssurityConsulting/ResultsFiles/Test1.jtl*

To open raw result file follow below steps,

1. Open JMeter 5.4.3 UI
2. Open test script ‘*AssurityConsulting\_PerfAssignment.jmx’*
3. Navigate to Sanity Test Thread Group -> Aggregate Results Listener.
4. Locate Tes1.jtl file to view the test statistics.

#---------------------------------------------------------------------------

# CSV Output file

#---------------------------------------------------------------------------

Text

Description automatically generated

#---------------------------------------------------------------------------

# Performance Test Report

#---------------------------------------------------------------------------

# **Purpose and Scope**

The Purpose of this performance testing assessment is to provide the business and technology stakeholders, enough information in terms of API’s performance, scalability & availability under the production-like scenario. Performance testing activity is to assess the performance of Category API.

## **Objective**

The objective behind the performance testing of Category API is to assess the application performance under the production like scenario and measure it against the pre-defined sets of non-functional requirement criteria’s. The high-level requirements include,

|  |  |
| --- | --- |
| **NFR #** | **Description** |
| NFR-01 | Test should support 5 Vusers (Threads) i.e. half the count of Category IDs shared in Test Data |
| NFR-02 | The test should ramp up at one VUser (Thread) per second |
| NFR-03 | Test should achieve 10 API calls in total for the 1-minute Steady State duration |
| NFR-04 | 90 percent of the times the API is expected to perform within 500 ms |

## **Executive Summary**

|  |  |
| --- | --- |
| **Performance Evaluation Outcome** | **Overall Status** |
| * Performance evaluation failed to meet NFR-04. API response time @90 percentile, breached the response time limit of 500 ms. * 3 out of 4 non-functional requirements successfully met the goal. |  |

Text

Description automatically generated

NFR wise details,

|  |  |  |  |
| --- | --- | --- | --- |
| **NFR #** | **Description** | **Outcome** | **Status** |
| NFR-01 | Test should support Vusers (Threads) half the count of Category IDs shared in Test Data | Test was designed to support 5 Vusers (Threads), refer Test Graph - snapshot1 section for more details |  |
| NFR-02 | The test should ramp up at one VUser (Thread) per second | Test was designed to ramp up 1 Vuser (thread) per second, refer Test Graph - snapshot1 section for more details |  |
| NFR-03 | Test should achieve 10 API calls in total for the 1-minute Steady State duration | Test has achieved 10 API calls in a steady state duration of 1 minute, refer Test Analysis section for more details |  |
| NFR-04 | 90 percent of the times the API is expected to perform within 500 ms | 90 percentile response time for the API is greater than 500ms, refer response time table for more details |  |

Text

Description automatically generated with medium confidence

## **Test Analysis**

**Response time details,**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **API Name** | **Transactions** | **Response Times in milliseconds** | | | | |
| **Min.** | **Avg.** | **90%** | **Max** | **% Error** |
| API\_01\_Categories | 10 | 222 | 434.20 | 867.70 | 878 | 10% |

* There is one transaction failed due to text check assertion failure for one of the Category ID.

**Resource utilizations**, (based on assumption that technology stack contains single app instance and a database)

|  |  |  |
| --- | --- | --- |
| **Bandwidth Utilization** | | Avg. - 0.61 Mbps |
| **CPU Utilization** | **App1** | less than 10% |
| **DB** | 20 to 30% |
| **Memory Utilization** | **App1** | 3% |
| **DB** | 18% |

**Test Graphs/Snapshots:**

Snapshot 1 – Test Plan

Graphical user interface, chart

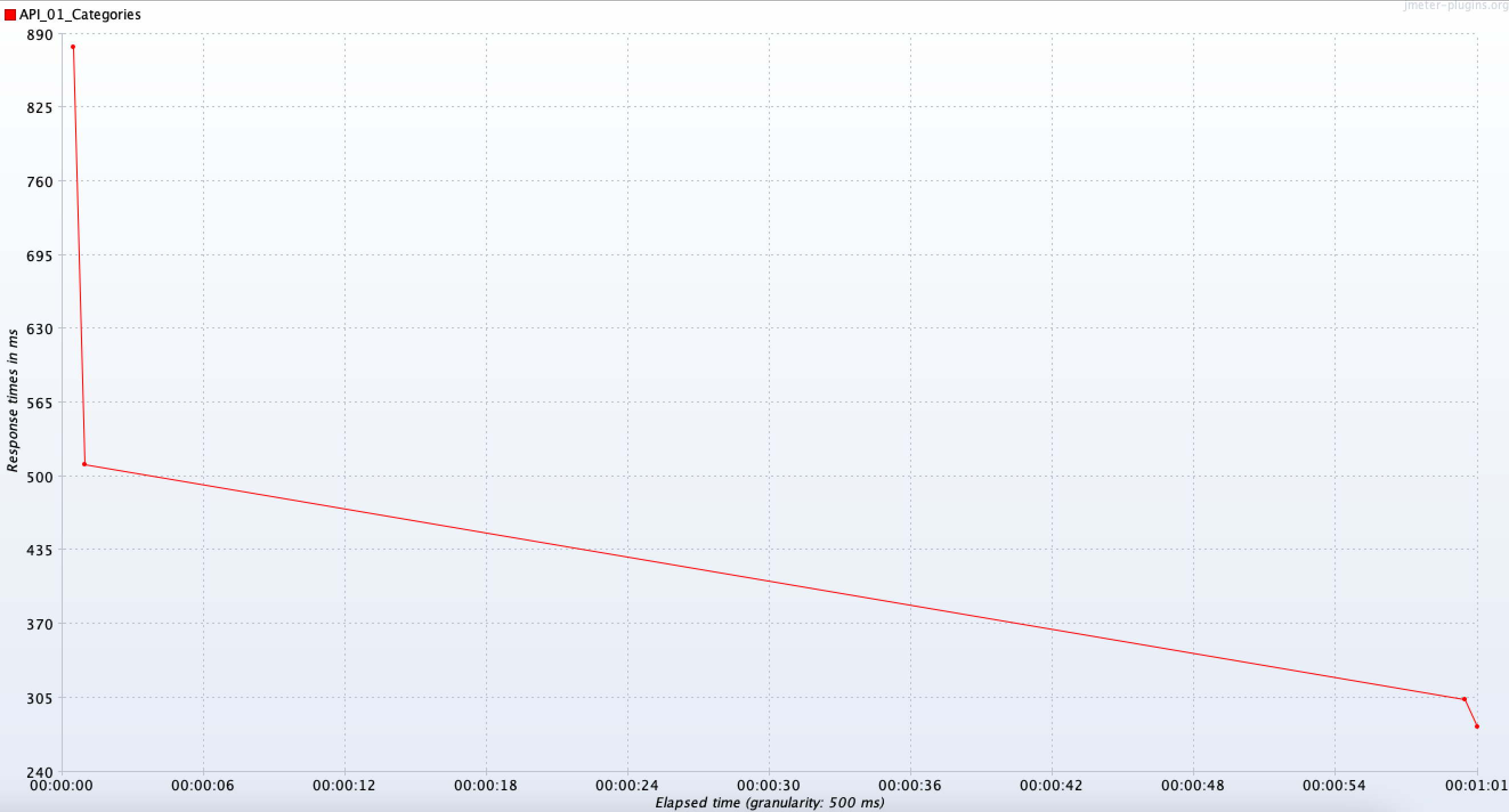
Description automatically generated with medium confidence

Snapshot 2 – Active threads during test execution

Graphical user interface, application

Description automatically generated

Snapshot 3 – Response times over time



Snapshot 4 – Response time percentile

Chart, line chart

Description automatically generated