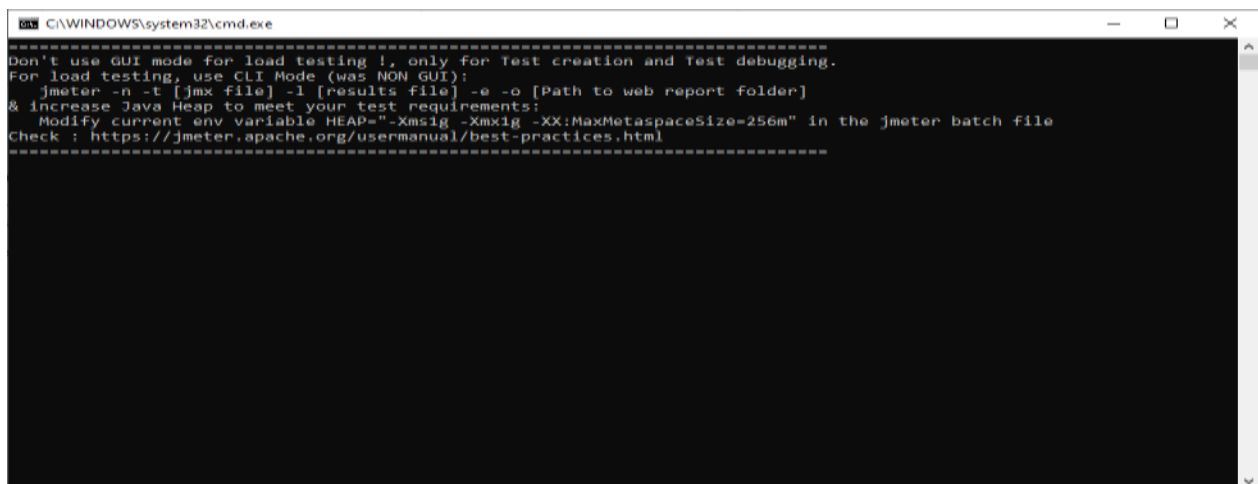


What is JMeter?

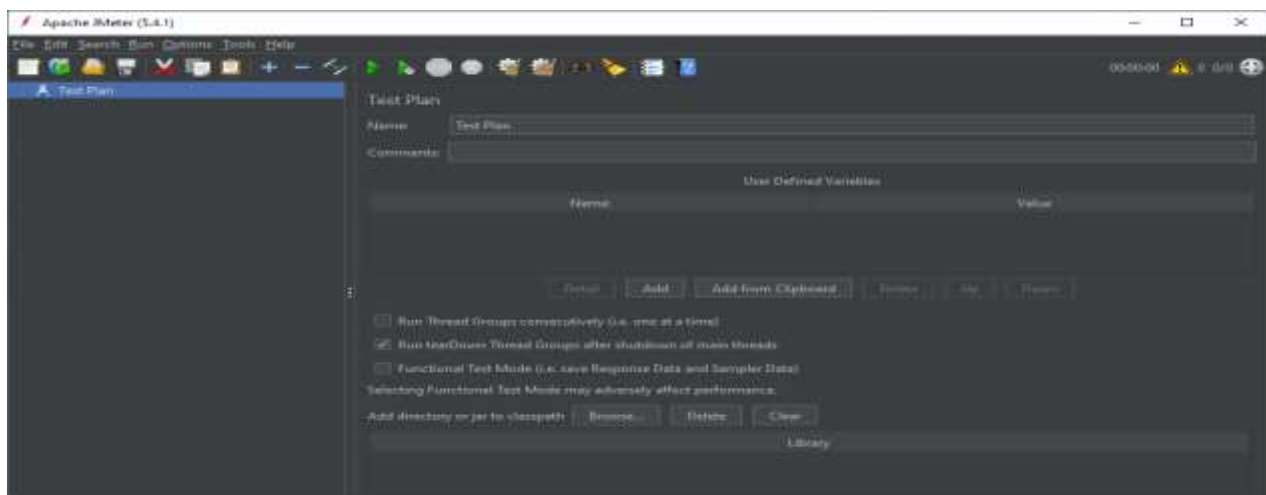
- Performance test application
- Open source and free
- Build using java
- Used for testing web, SOAP/REST Web services / FTP/ Database/ Mail
- **SOAP** stands for Simple Object Access Protocol. **SOAP** only works with XML formats
- **REST** stands for Representational State Transfer. **REST** work with plain text, XML, HTML and JSON.

How to install JMeter?

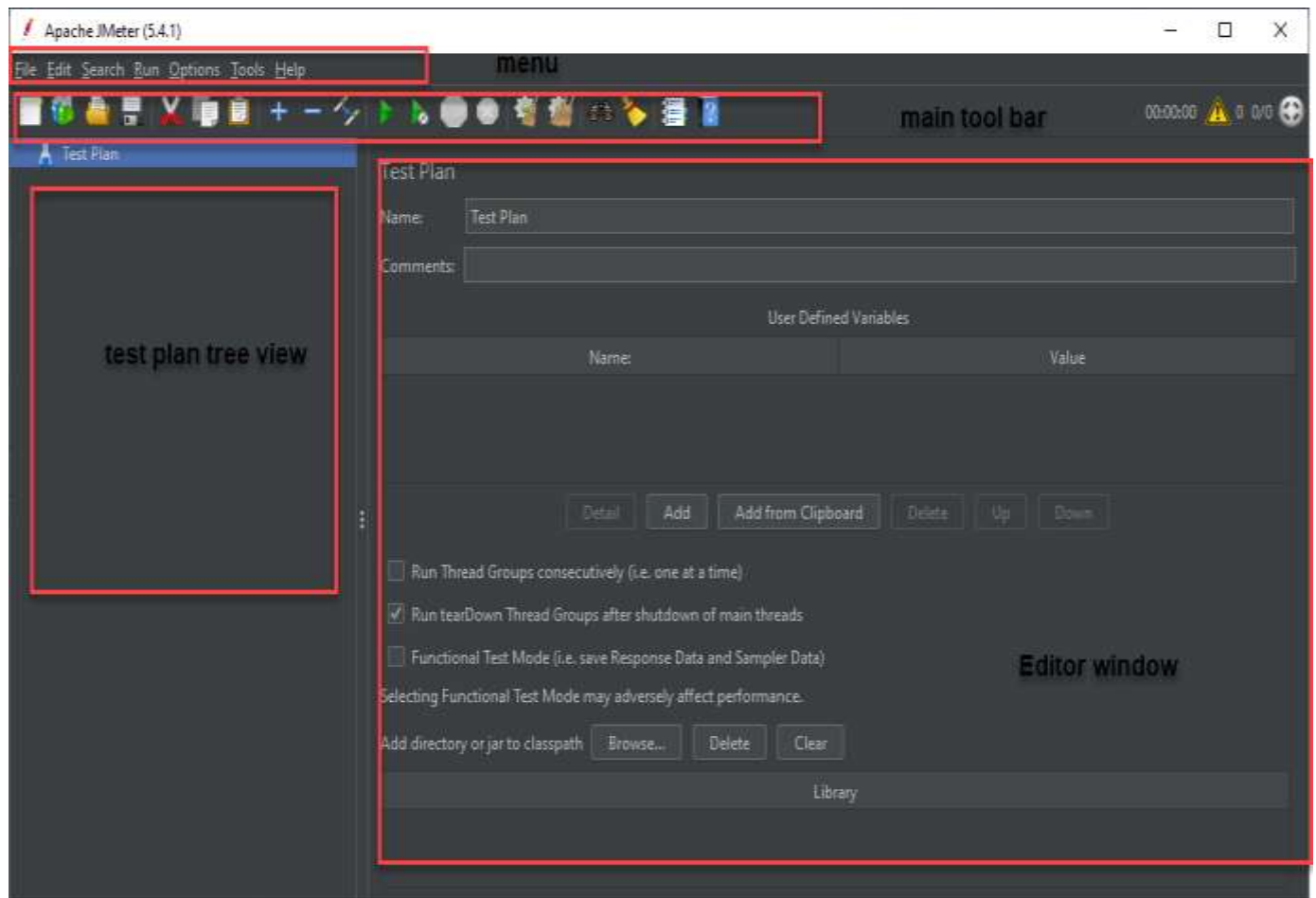
- 1) Ensure Java is installed on system. Version Greater than 8.
- 2) Download JMeter from this link https://jmeter.apache.org/download_jmeter.cgi
(Download from Binaries)
- 3) Unzip and keep JMeter folder at any location. (Right click and select Extract all)
- 4) Start JMeter. Windows – **jmeter/bin – jmeter.bat**



```
C:\WINDOWS\system32\cmd.exe
=====
Don't use GUI mode for load testing !, only for Test creation and Test debugging.
For load testing, use CLI Mode (was NON GUI):
jmeter -n -t [jmx file] -l [results file] -e -o [Path to web report folder]
& increase Java Heap to meet your test requirements:
  Modify current env variable HEAP="-Xms1g -Xmx1g -XX:MaxMetaspaceSize=256m" in the jmeter batch file
Check : https://jmeter.apache.org/usermanual/best-practices.html
=====
```



GUI Overview:

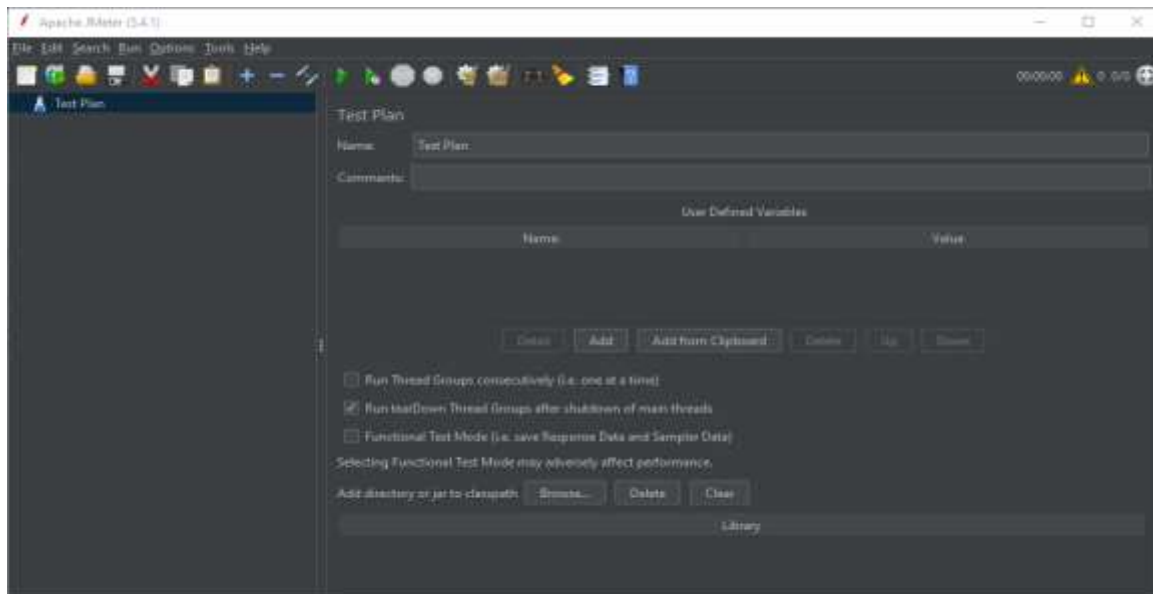


Test Plan >> Threads >> Sampler >> Request >> Listeners /
Assertions

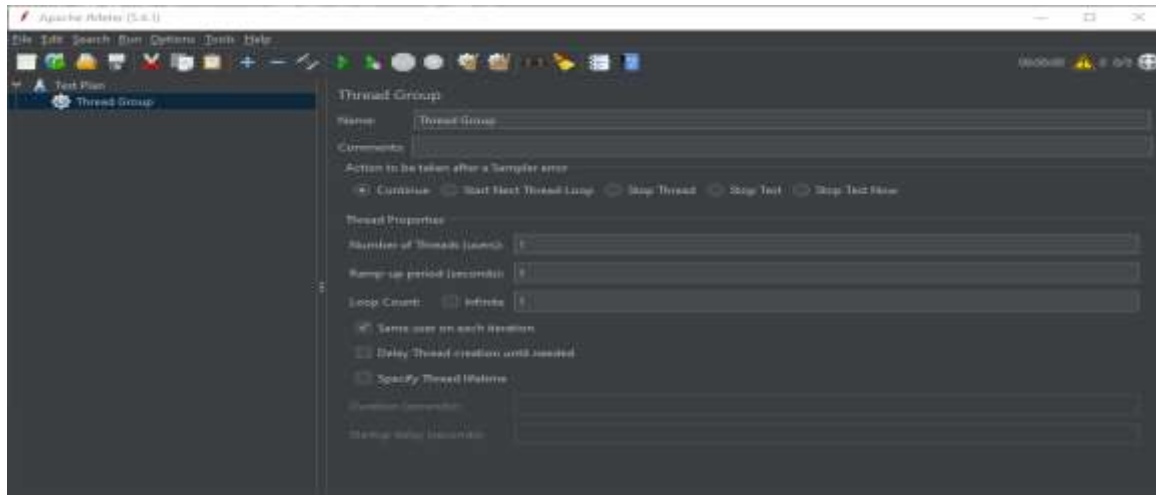
Create First JMeter Test:

Steps

- Start JMeter
- Create a Test Plan
- Create a thread group (user)
- Add a sampler (Http)
- Add Listener
- Run the Test
- Go to menu > Click on **File > New**



- For Creating thread group right click on **Test Plan** > Add > Thread Users > Thread Group

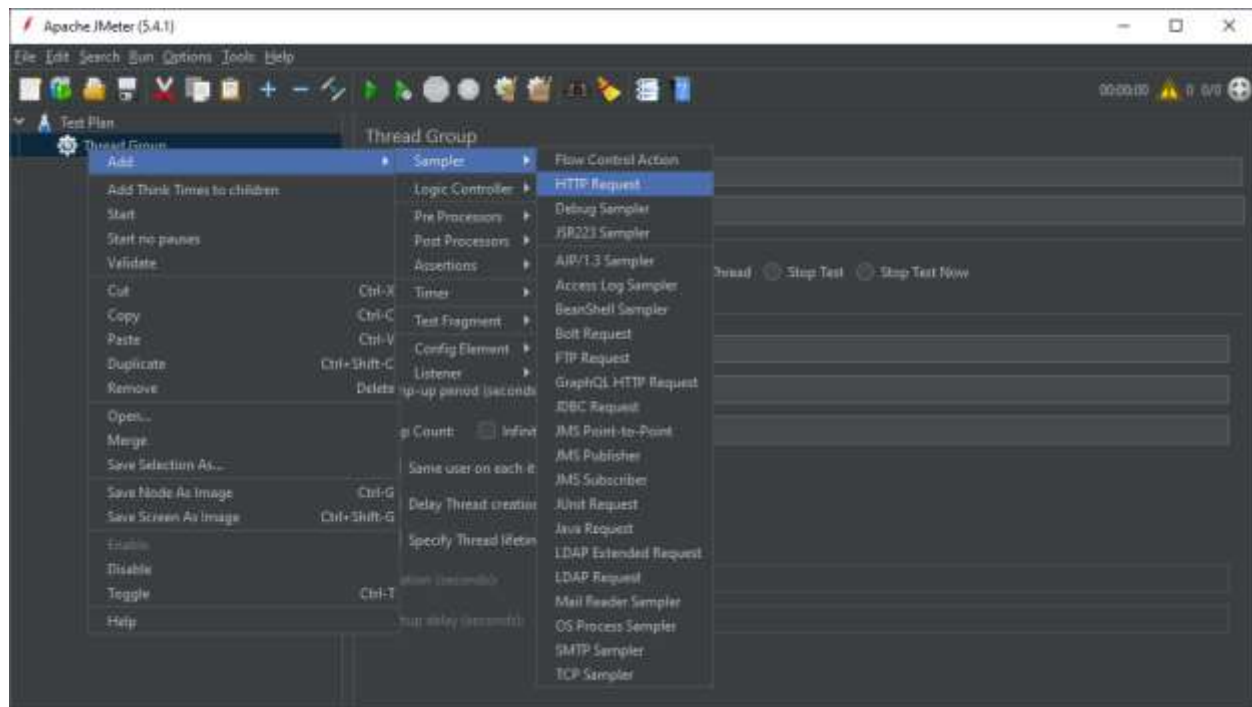


Number of threads: How many users you want to use for this test

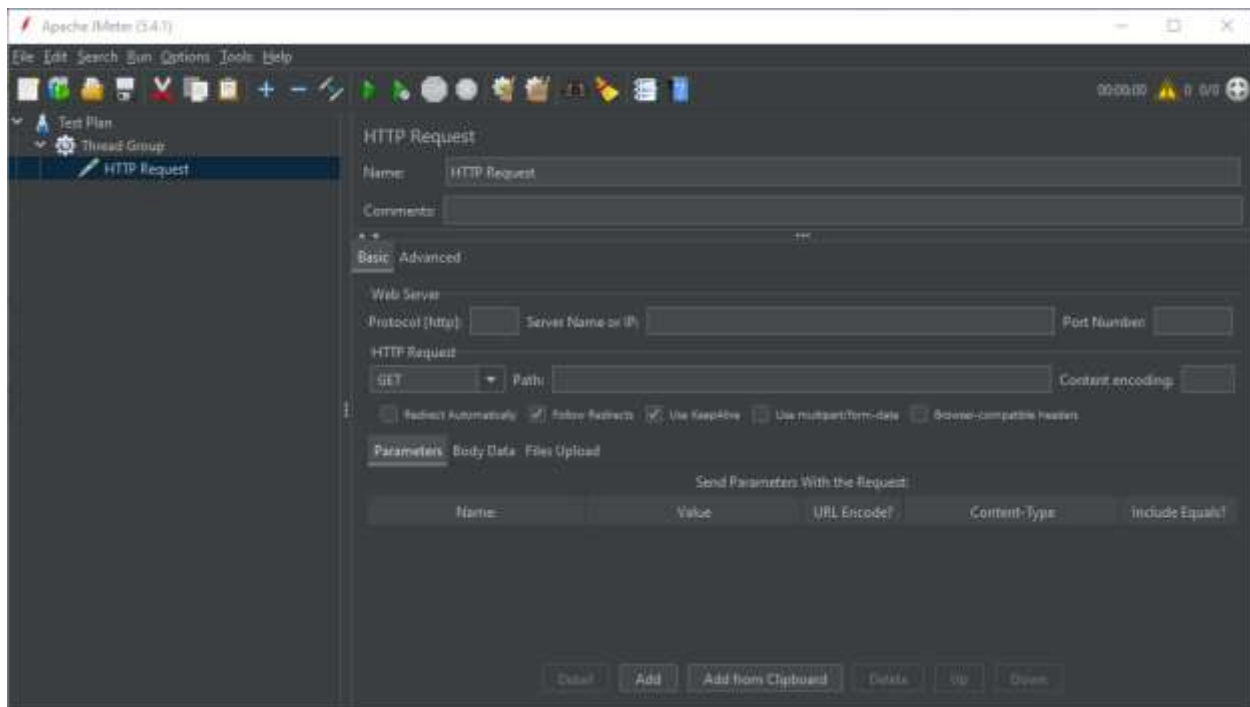
Ramp up Period: If ramp up period is one second that means, all the 10 users will start testing instantly without having any ramp up time. But if ramp up time is selected 20 seconds, that means 10 users will be ramped up in 20 seconds. That means, 1 user will be added after every 2 seconds and after 20 seconds there will be 10 users. The test will be run with a full load of 10 users.

Loop Count: Means how many iterations we need to run this test. Or how many times we want to run this test. If we select **infinite** the test will go on until we forcefully stop it.

- Add sampler: Right click on thread group > add > sampler > Http Request



From the various type of samplers, we are using http type sampler and the test plan will be prepared for **Basic**.

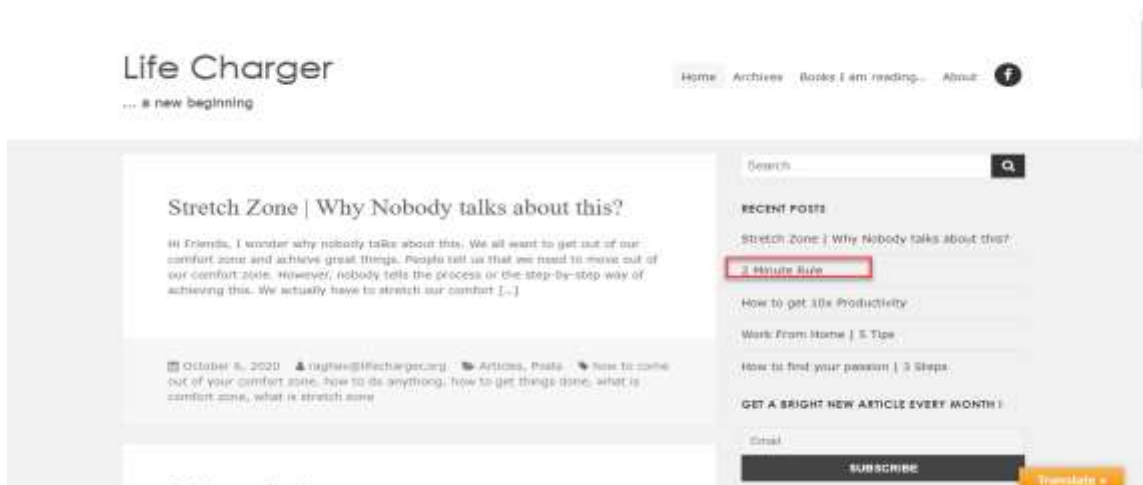


Server Name or IP: put the URL we want to test. For example, for the URL <https://lifecharger.org/> we are keeping **lifecharger.org** for server name or IP.

Protocol: For protocol we are keeping the https

HTTP Request: Let's put it GET, unless we are doing or using any API testing.

Path: The path we will be using of the web page. For example, <https://lifecharger.org/2-minute-rule/> . /2-minute-rule/ is the path here.



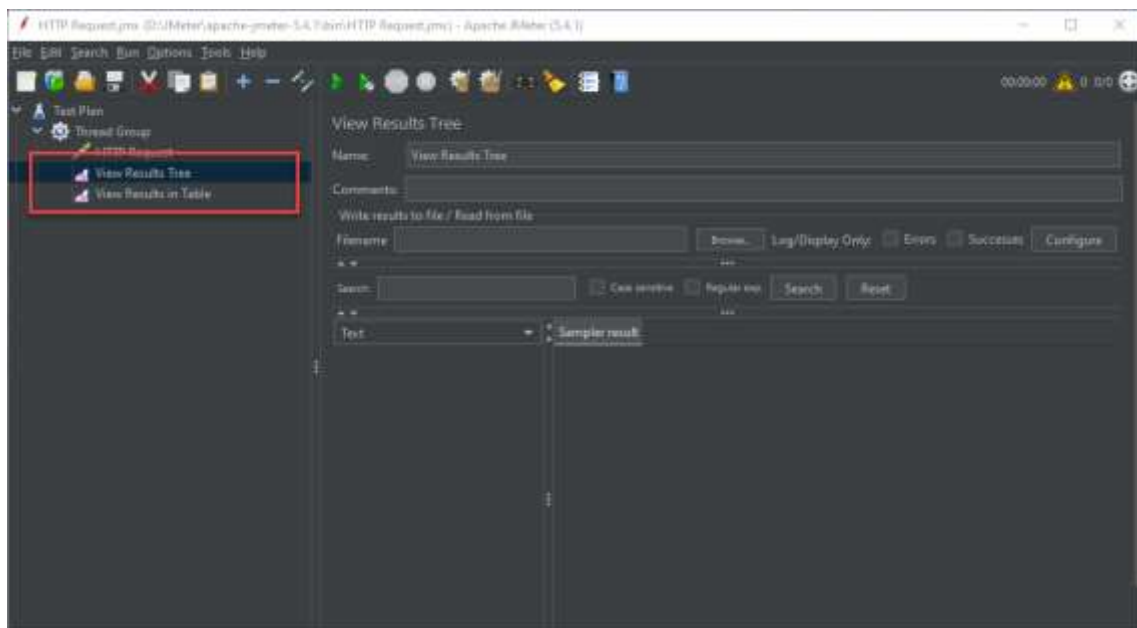
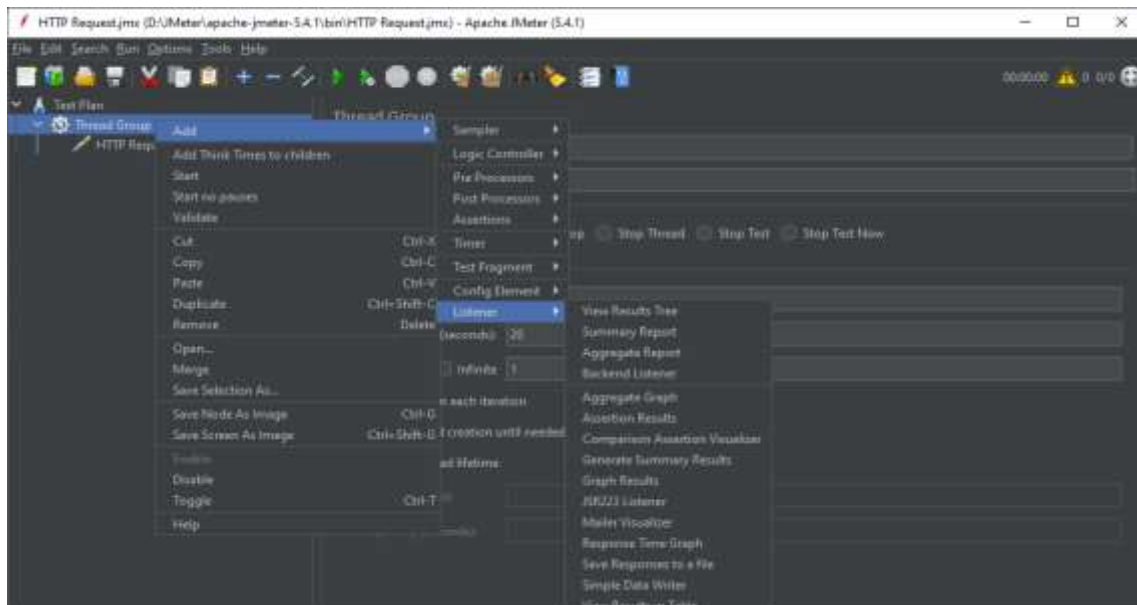
Parameters, Body Data, File Upload can be added if we want to do any kind of API testing.

After clicking on SAVE icon we can save the test plan, and we can save it at any location.

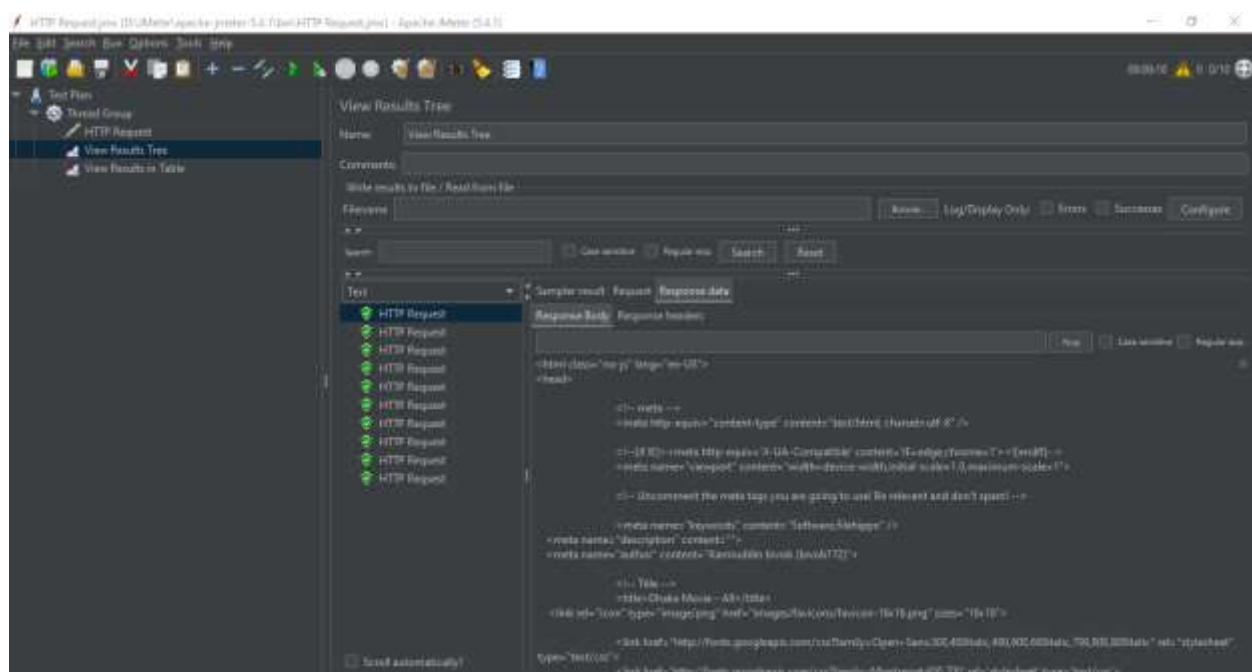
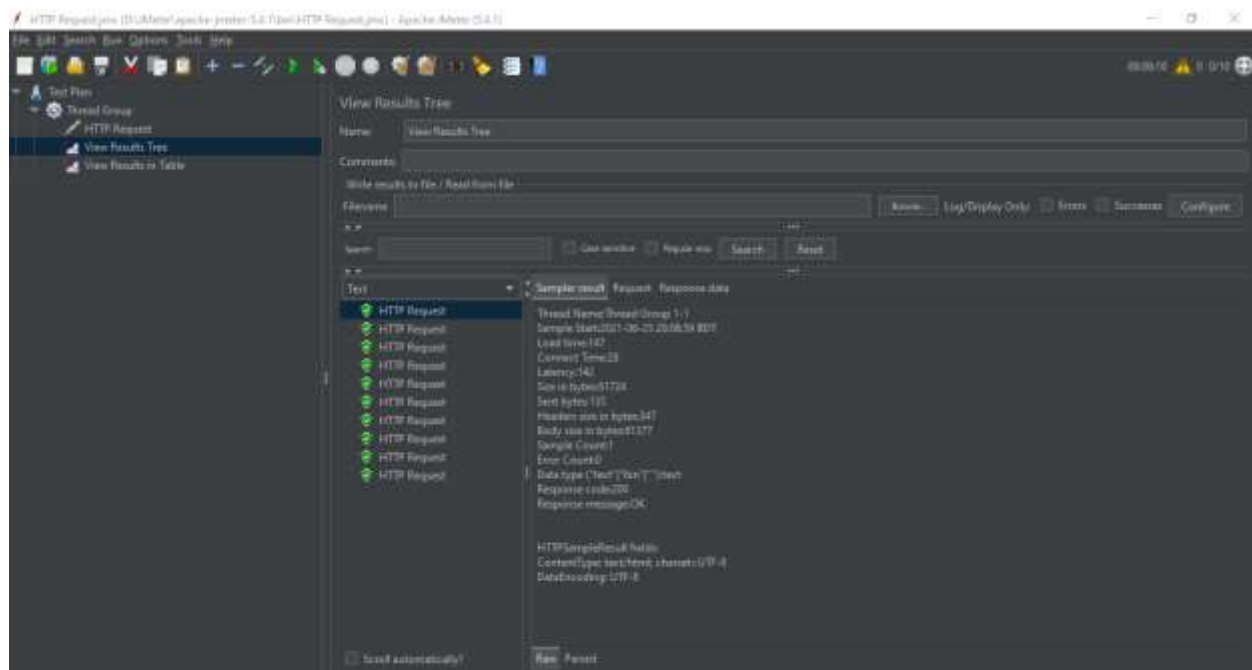
- Add Listeners: For viewing the result we need to do this part. We can add multiple listeners here.

Right click on Thread group > ADD> Listener > View Results Tree

Right click on Thread group > ADD> Listener > View Results in Table



Let's see the output after clicking on the RUN icon.



The screenshot shows the 'View Results in Table' window in Apache JMeter. The table displays the following data:

Sample #	Start Time	Thread Name	Label	Sample Time	Status	Bytes	Sent Bytes	Latency	Connect Time
1	2020-03-09 10:00:00.000	Thread Group 1-1	HTTP Request	147	200	01204	100	146	20
2	2020-03-09 10:00:00.010	Thread Group 1-2	HTTP Request	179	200	01204	100	146	20
3	2020-03-09 10:00:00.020	Thread Group 1-1	HTTP Request	158	200	01204	100	151	22
4	2020-03-09 10:00:00.030	Thread Group 1-2	HTTP Request	175	200	01204	100	149	20
5	2020-03-09 10:00:00.040	Thread Group 1-1	HTTP Request	124	200	01204	100	129	20
6	2020-03-09 10:00:00.050	Thread Group 1-2	HTTP Request	108	200	01204	100	129	20
7	2020-03-09 10:00:00.060	Thread Group 1-1	HTTP Request	103	200	01204	100	127	20
8	2020-03-09 10:00:00.070	Thread Group 1-2	HTTP Request	124	200	01204	100	129	20
9	2020-03-09 10:00:00.080	Thread Group 1-1	HTTP Request	124	200	01204	100	129	20
10	2020-03-09 10:00:00.090	Thread Group 1-2	HTTP Request	124	200	01204	100	127	20

Here From the sample result, we can check our results and status.

From Response Data > Response Body, we can see the html and JavaScript outcome.

From Table view we can get the, Necessary information in tabular format.

From Thread Name 1-1 means first run of first user, 1-2 means first run of second user.

Latency: In the context of **Performance testing:** Network **latency** is an expression of how much time it takes for a packet of data to get from one designated point to another. In some environments, **latency** is measured by sending a packet that is returned to the sender; the round-trip time is considered the **latency**. **Time to get the first byte.**

The screenshot shows the 'View Results in Table' window in Apache JMeter. The table displays the following data:

Sample #	Start Time	Thread Name	Label	Sample Time	Status	Bytes	Sent Bytes	Latency	Connect Time
1	2020-03-09 10:00:00.000	Thread Group 1-1	HTTP Request	127	200	01204	100	146	20
2	2020-03-09 10:00:00.010	Thread Group 1-2	HTTP Request	179	200	01204	100	146	20
3	2020-03-09 10:00:00.020	Thread Group 1-1	HTTP Request	158	200	01204	100	151	22
4	2020-03-09 10:00:00.030	Thread Group 1-2	HTTP Request	175	200	01204	100	149	20
5	2020-03-09 10:00:00.040	Thread Group 1-1	HTTP Request	124	200	01204	100	129	20
6	2020-03-09 10:00:00.050	Thread Group 1-2	HTTP Request	108	200	01204	100	129	20
7	2020-03-09 10:00:00.060	Thread Group 1-1	HTTP Request	103	200	01204	100	127	20
8	2020-03-09 10:00:00.070	Thread Group 1-2	HTTP Request	124	200	01204	100	129	20
9	2020-03-09 10:00:00.080	Thread Group 1-1	HTTP Request	124	200	01204	100	129	20
10	2020-03-09 10:00:00.090	Thread Group 1-2	HTTP Request	124	200	01204	100	127	20

Connect Time : Time took for connecting to the server.

HTTP Request.jmx (D:\Maven\apache-jmeter-5.4.1\bin\HTTP Request.jmx) - Apache JMeter (5.4.1)

File Edit Search Run Options Tools Help

Test Plan
Thread Group
HTTP Request
View Results Tree
View Results in Table

start stop icon enables when we select infinite loop

View Results in Table

time 00:00:18 0 0:00 TOTAL user

Workflows to file / Read from file

Buttons Log/Display Only Errors Successes Configure

	Start Time	Thread Name	Label	Sample...	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	2006:39.369	Thread Group 1-1	HTTP Request	142	✓	61224	135	142	26
2	2007:01.972	Thread Group 1-2	HTTP Request	176	✓	61234	135	169	22
3	2007:03.512	Thread Group 1-3	HTTP Request	159	✓	61234	135	151	22
4	2007:05.970	Thread Group 1-4	HTTP Request	173	✓	61234	135	165	26
5	2007:07.870	Thread Group 1-5	HTTP Request	154	✓	61234	135	115	20
6	2007:09.470	Thread Group 1-6	HTTP Request	134	✓	61234	135	125	25
7	2007:11.369	Thread Group 1-7	HTTP Request	133	✓	61234	135	121	22
8	2007:13.369	Thread Group 1-8	HTTP Request	154	✓	61224	135	114	16
9	2007:15.970	Thread Group 1-9	HTTP Request	154	✓	61234	135	113	26
10	2007:17.871	Thread Group 1-10	HTTP Request	135	✓	61234	135	117	25

☐ Sort automatically? ☐ Only sampled? Run at Samples: 10 Latest Sample: 126 Samples: 10 Download: 0

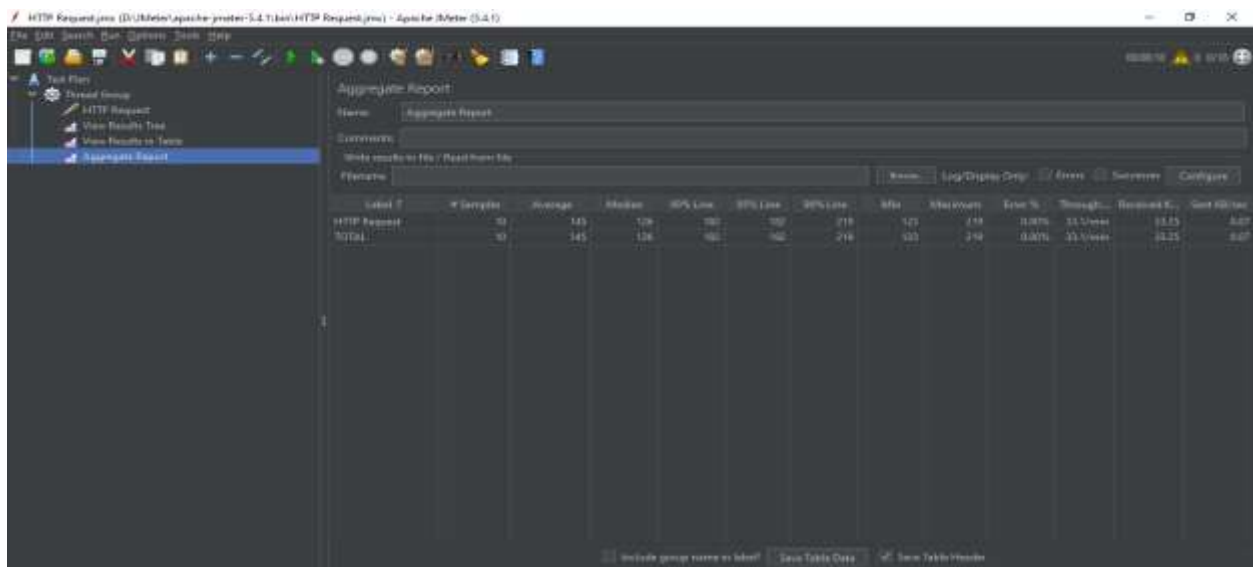
Listeners: (Reporting)

We can create and view the reports of our test. Listener means elements that gather information about the performance test used to view results of the test.

Previously we have seen view results in tree and view results in table. View Result tree is not recommended to use.

Let's see Aggregate Result;

Right click on Thread group > ADD> Listener > Aggregate Results



The screenshot shows the JMeter 'Aggregate Report' window. The 'Name' field is set to 'Aggregate Report'. The 'Summary' section shows 'Write results to file / Read from file' with a file name field. The 'Results' tab is active, displaying a table of performance metrics. The table has columns for Label, # Samples, Average, Median, 90% Line, 95% Line, 99% Line, Min, Max, Min/Max, Error %, Throughput, Received K, and Sent K/Sec. The data rows are for 'HTTP Request' and 'Total'.

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Min/Max	Error %	Throughput	Received K	Sent K/Sec
HTTP Request	10	145	139	160	160	216	121	216	1.18	0.00%	11.5/sec	11.05	8.07
Total	10	145	139	160	160	216	121	216	1.18	0.00%	11.5/sec	11.05	8.07

Average: Average time taken for all the samples

Median: Median is like a middle time; Half of the request took more or less than this time.

90% - 99% Line: 90% - 99% response was within this time

99% and maximum time are more or less the same time.

Min: Minimum time taken by any request in all the samplers.

Max: Maximum time taken by any request in all the samplers.

Error: Error count in percentage

Throughput: Number of requests per minute metrics.

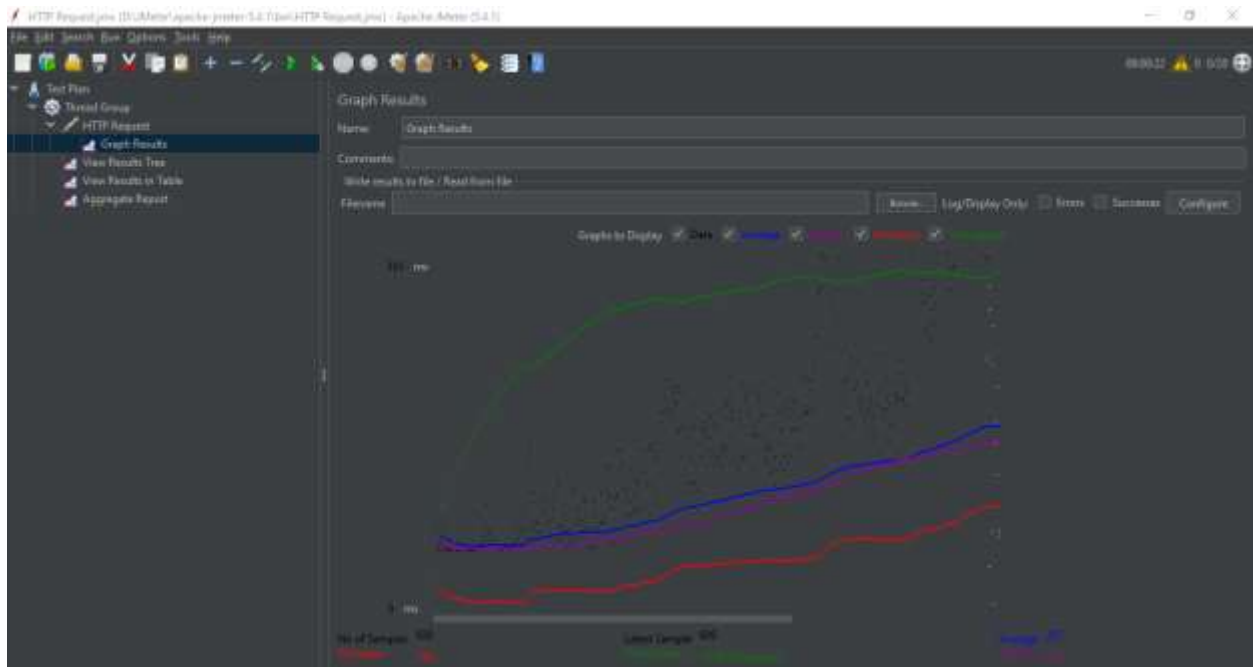
Received and sent: Received and sent data size.

Let's see Graph Result;

Right click on Thread group > ADD> Listener > Graph Results

For getting a better graph view we can set the number of threads 20/30 and select infinite and ramp up period to 30/40.

If you are practicing JMeter you are recommended not to use large number like 200/500/100 for the good for your system.



Let's see Summary Result;

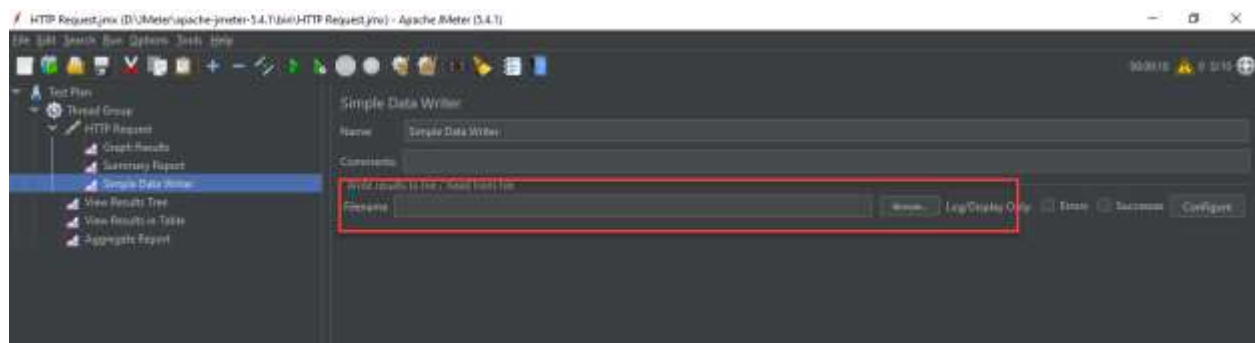
Right click on Thread group > ADD> Listener > Summary Results

Almost same to aggregate report. There is no 90-99% Line row in summary report.

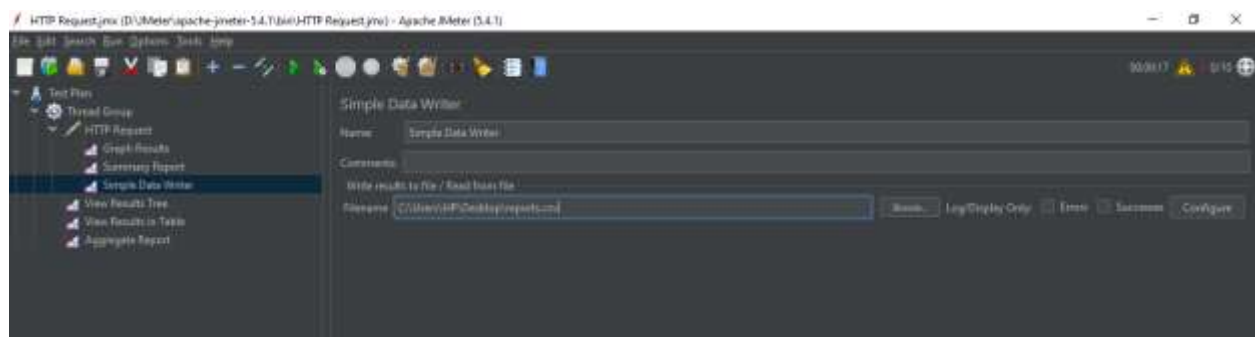
Let's see Simple Data Writer;

Right click on Thread group > ADD> Listener > Simple Data Writer

Here we have an option to send the result we get from JMeter to a file.



Name the report like report.csv after selecting the location where the file will be created.



After that we will find a csv file name report in desktop and if we open that we can see the results.

FileHomeInsertPage LayoutFormulasDataReviewViewHelp

Calibri11AaWrap TextGeneral\$%-./:;[]{}'&"@_!~<=>+*~<=>+*Conditional FormattingTable StylesInsertDeleteFormatCellsAutosumFillSort & Filter & Select

ClipboardFontAlignmentNumberEditing

POSSIBLE DATA LOSS Some features may be lost if you save this workbook in the comma-separated (CSV) format. To preserve these features, save it as an Excel file format.

Don't show againSave As...

A1timeStamp

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	timestamp	elapsed	label	response	response code	thread name	data type	success	failure	message	bytes	sent bytes	group	thread	all threads	URL	latency	idle time	connect		
2	1.62E+12	175	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	167	0	20						
3	1.62E+12	141	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	127	0	21						
4	1.62E+12	146	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	127	0	19						
5	1.62E+12	122	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	112	0	19						
6	1.62E+12	130	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	121	0	14						
7	1.62E+12	121	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	112	0	20						
8	1.62E+12	154	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	152	0	21						
9	1.62E+12	810	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	439	0	90						
10	1.62E+12	456	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	346	0	182						
11	1.62E+12	491	HTTP Req	200 OK	Thread Gr text	TRUE		61705	135	1	1	https://dhu	342	0	100						
12																					
13																					
14																					

Assertions

Assertions = check on the Request / Response

Like-

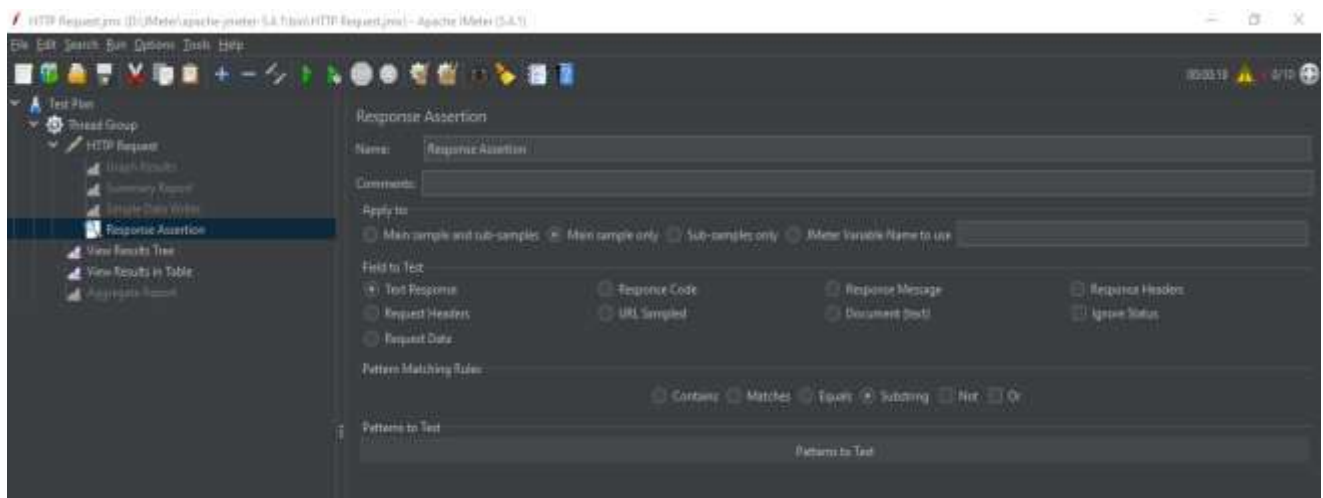
- Response Assertion
- Duration Assertion
- Size Assertion
- HTML Assertion

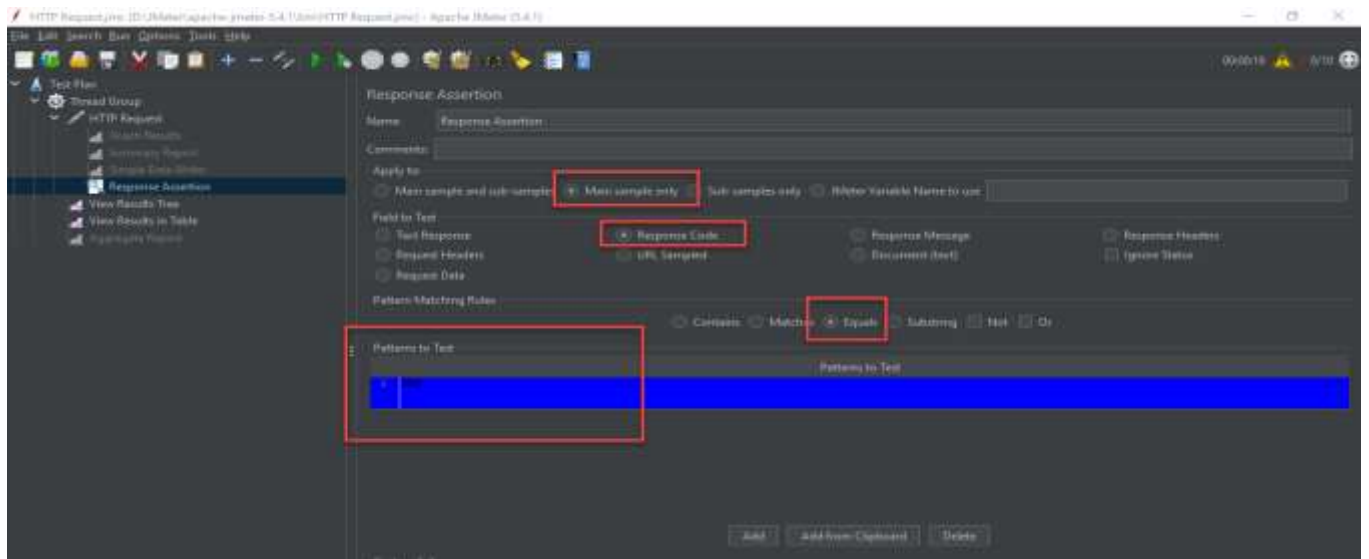
Before jumping into assertion let's disable the listeners without **View Results Tree** and **View Results Table**.

If we add an assertion to a test case, it will be applicable for all the samplers > Requests.

- We can add the assertion separately to a Request. Right click on request > add > assertion > Response Assertion

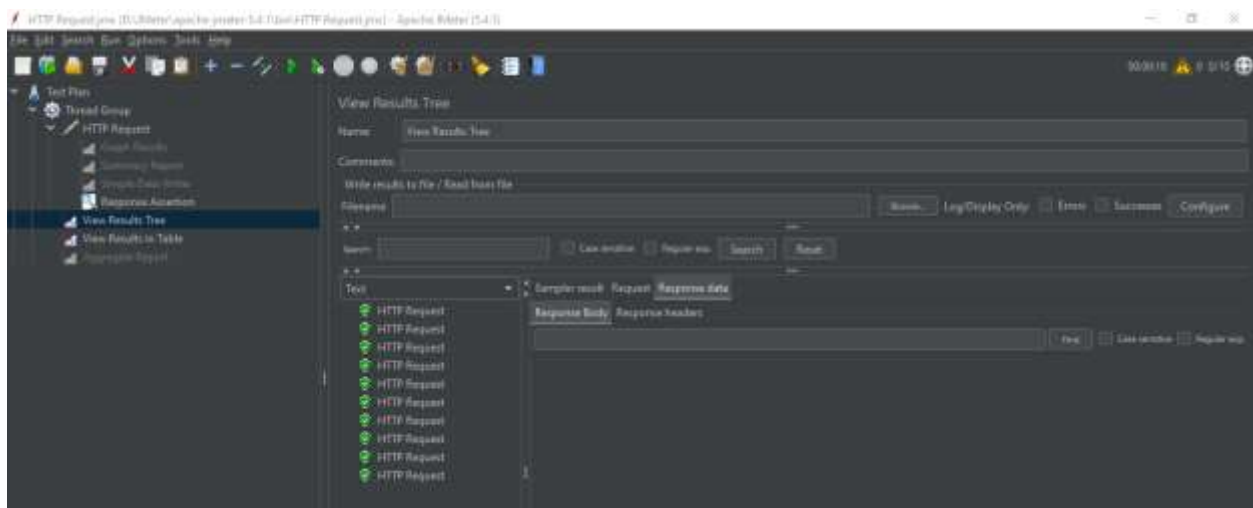
Response Assertion:





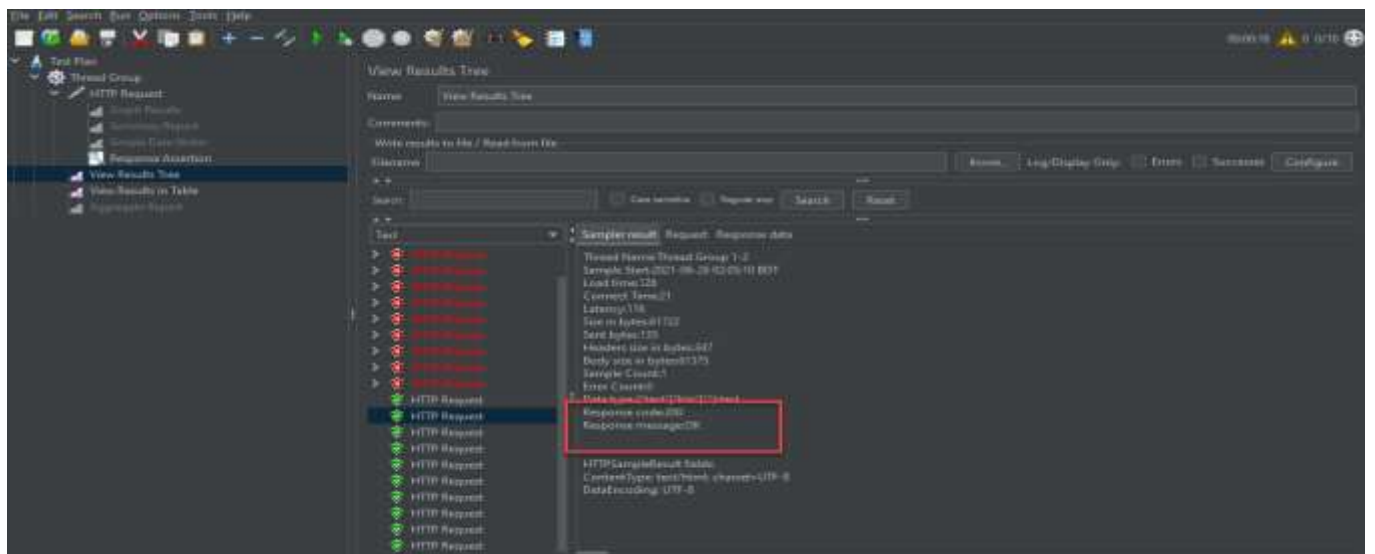
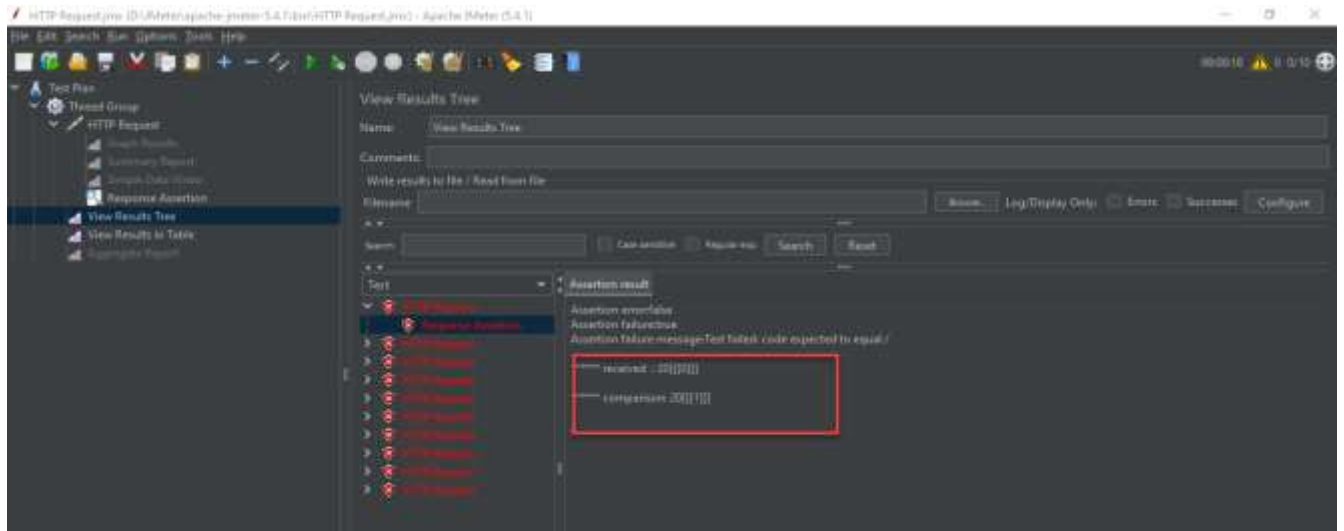
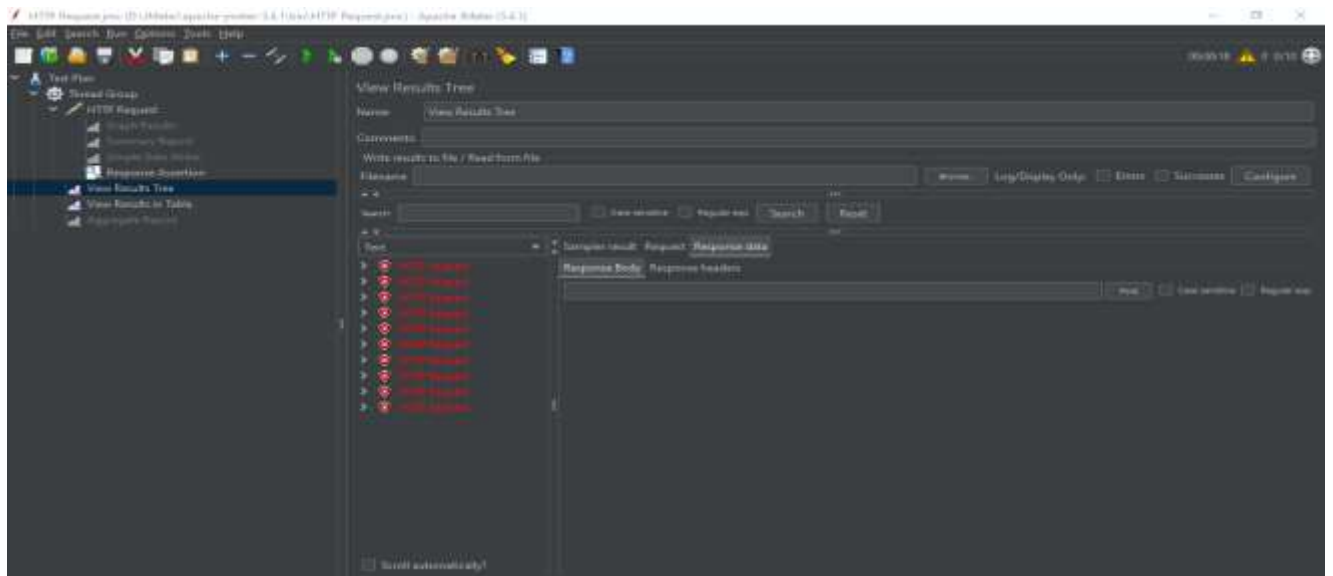
By selecting Response Code, equals and pattern = 200 means, Response code is 200.

Now save and run the test case and see the result in the tree view. Now we can see everything is fine.



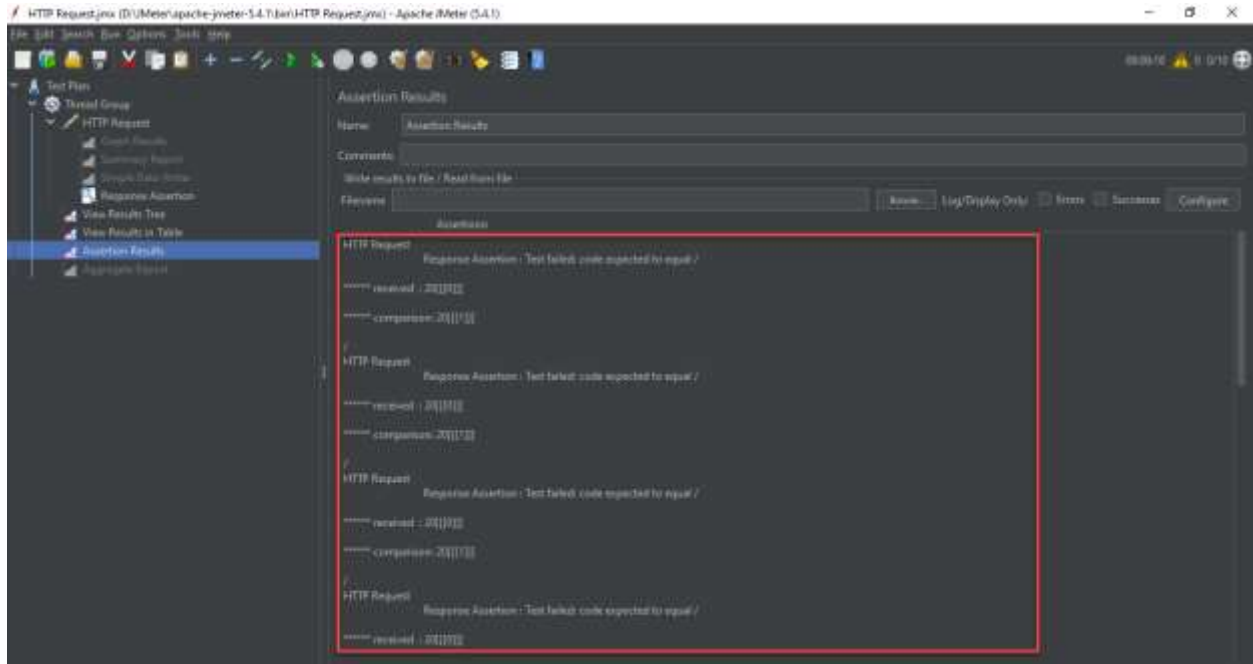
But if we save the response code to 201 it will fail and will show the failed result.

Response code 200 means success. That's why it shows error for other value than 200.

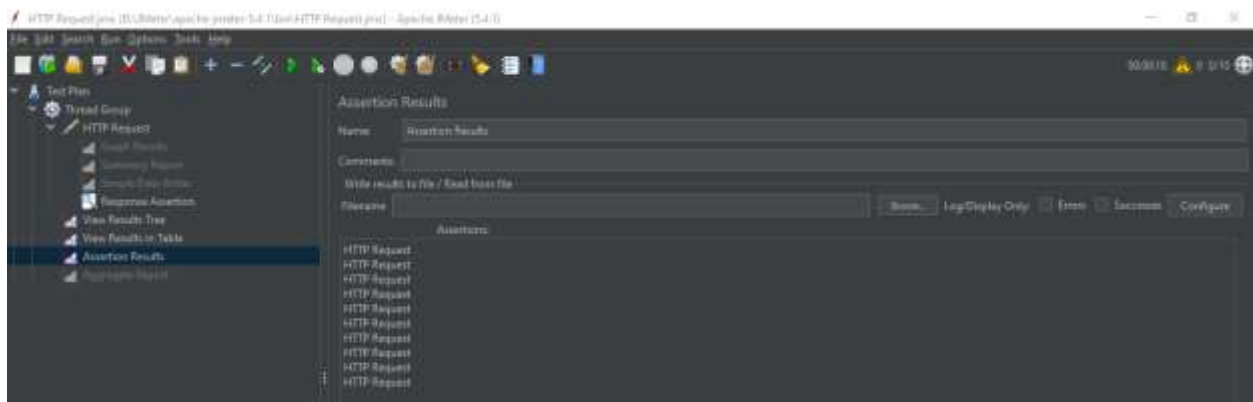


Now

Thread > add > Listener > assertion results; we can use this listener for verifying the results with response code. For 201 we will see failure.



For response code 200 we will see no failure; in assertion result

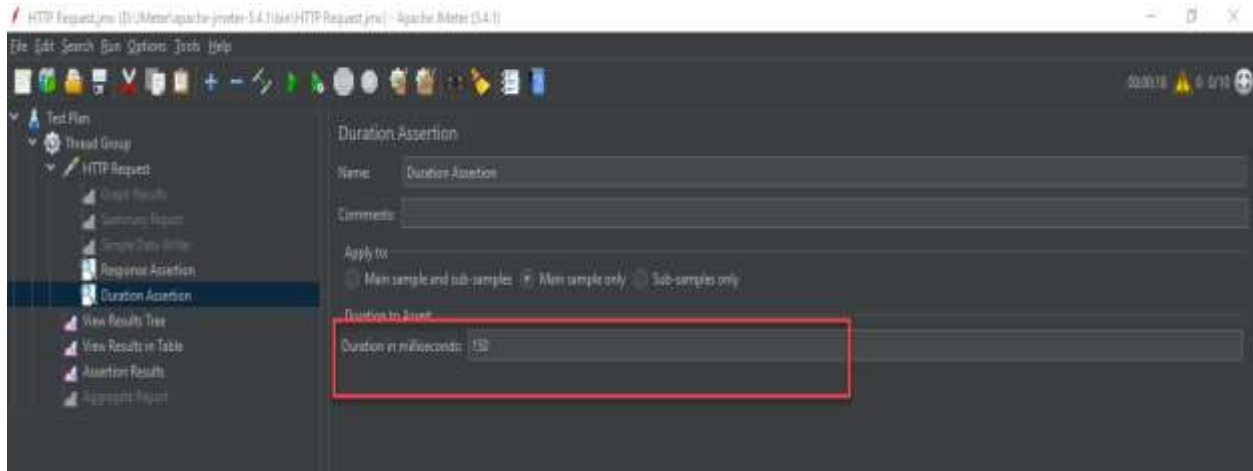


Assertion Result is a type of listener for REQUEST.

Response Assertion is mostly used in JMeter.

Duration Assertion:

- request (Sampler) > add > assertion > Duration Assertion

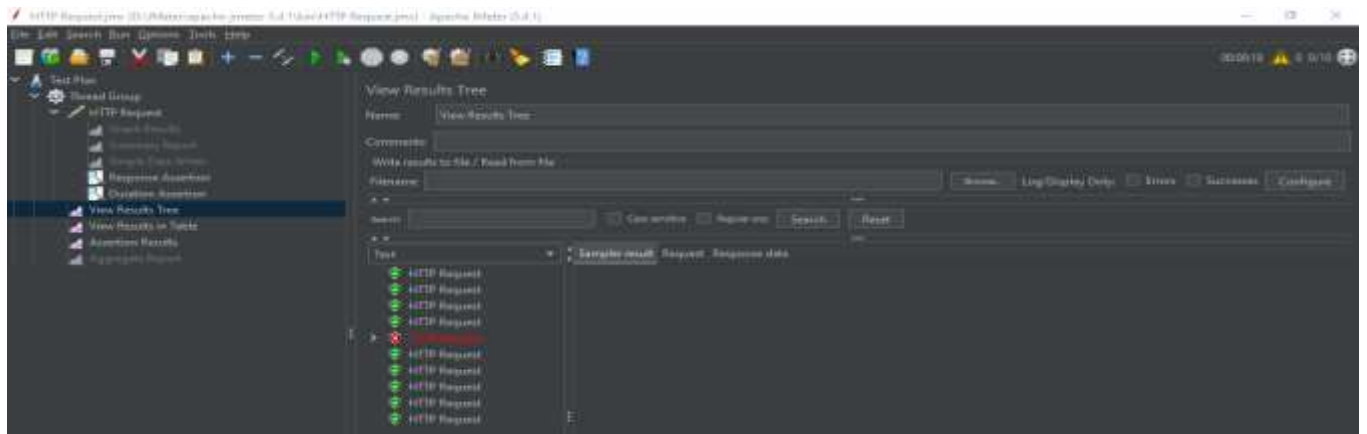


The value of duration assertion is kept in ms. So, from table view we can set the threshold value for duration assertion.

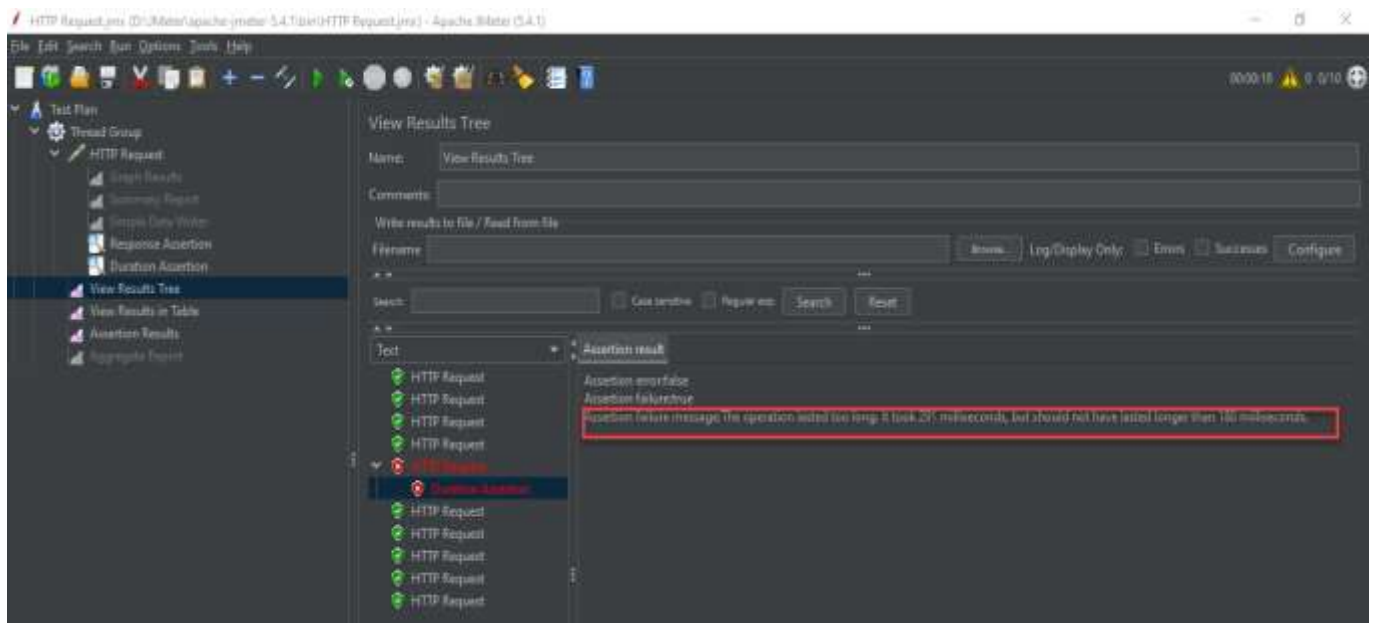
The screenshot shows the 'View Results in Table' window in Apache JMeter. The 'Sample Time' column is highlighted with a red rectangle. The table displays the following data:

Sample #	Start Time	Thread Name	Label	Sample Time	Status	Bytes	Sent Bytes	Latency	Connect Time
1	02:14:33.290	Thread Group 1-1	HTTP Request	239	Success	61752	135	142	24
2	02:14:41.391	Thread Group 1-2	HTTP Request	178	Success	61752	135	140	21
3	02:14:43.390	Thread Group 1-3	HTTP Request	162	Success	61752	135	152	17
4	02:14:45.391	Thread Group 1-4	HTTP Request	179	Success	61752	135	163	20
5	02:14:47.391	Thread Group 1-5	HTTP Request	126	Success	61752	135	113	21
6	02:14:49.392	Thread Group 1-6	HTTP Request	128	Success	61752	135	117	19
7	02:14:51.392	Thread Group 1-7	HTTP Request	224	Success	61750	135	213	22
8	02:14:53.393	Thread Group 1-8	HTTP Request	124	Success	61750	135	114	20
9	02:14:55.393	Thread Group 1-9	HTTP Request	124	Success	61750	135	113	20
10	02:14:57.401	Thread Group 1-10	HTTP Request	175	Success	61750	135	163	19

So, from the test plan we made, we can take the threshold time for 180 ms. As the sample time is between 124 to 239 ms and press the start button. After completing the process, we can check the result from view results tree.



Now we can check the result from the error result from tree view result.

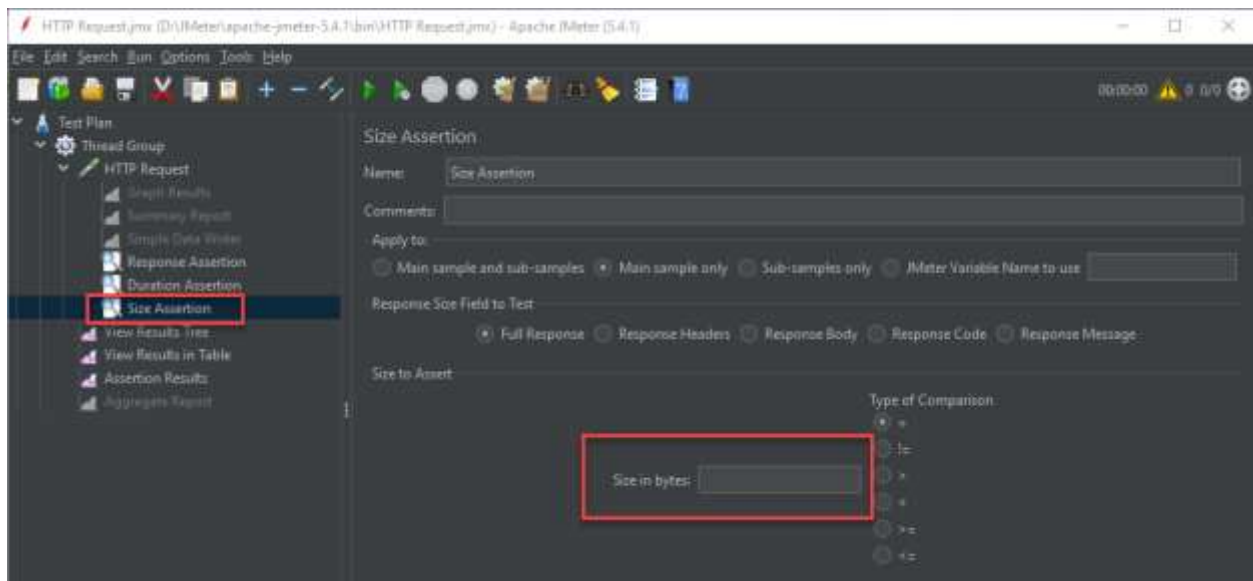


Assertion failure message: The operation lasted too long: It took 291 milliseconds, but should not have lasted longer than 180 milliseconds. Same result is seen from assertion result too.



Size Assertion:

- Let's jump into size assertion; request (Sampler) > add > assertion > Duration Assertion.



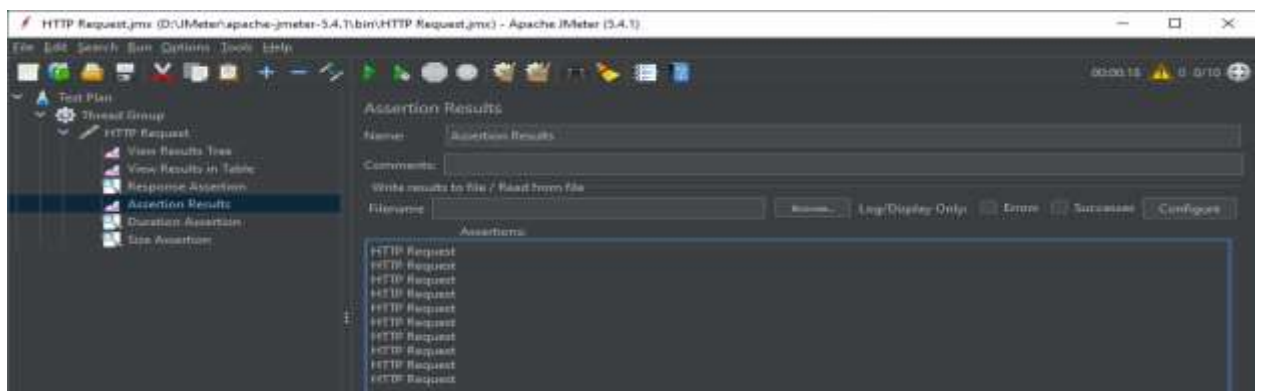
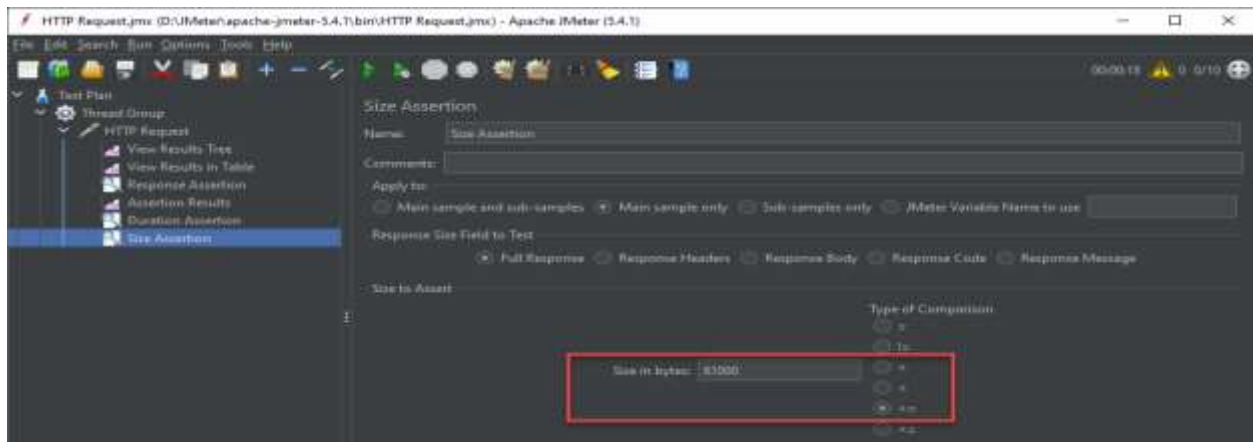
We can size to assert and get the size in bytes from view result in tables (listeners)

The screenshot shows the Apache JMeter 5.4.1 interface with the 'View Results in Table' listener selected. The table displays the results of the HTTP Request sampler. The 'Sample Time' column is highlighted with a red box, and the 'Bytes' column is highlighted with a green box.

Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	02:14:39.390	Thread Group 1-1	HTTP Request	239	✓	61712	135	143	24
2	02:14:41.391	Thread Group 1-2	HTTP Request	178	✓	61712	135	161	21
3	02:14:43.390	Thread Group 1-3	HTTP Request	162	✓	61712	135	152	17
4	02:14:45.391	Thread Group 1-4	HTTP Request	174	✓	61712	135	163	20
5	02:14:47.391	Thread Group 1-5	HTTP Request	126	✓	61712	135	113	21
6	02:14:49.393	Thread Group 1-6	HTTP Request	126	✓	61712	135	117	19
7	02:14:51.393	Thread Group 1-7	HTTP Request	224	✓	61719	135	213	72
8	02:14:53.393	Thread Group 1-8	HTTP Request	124	✓	61719	135	114	20
9	02:14:55.391	Thread Group 1-9	HTTP Request	124	✓	61719	135	111	20
10	02:14:57.401	Thread Group 1-10	HTTP Request	173	✓	61719	135	163	19

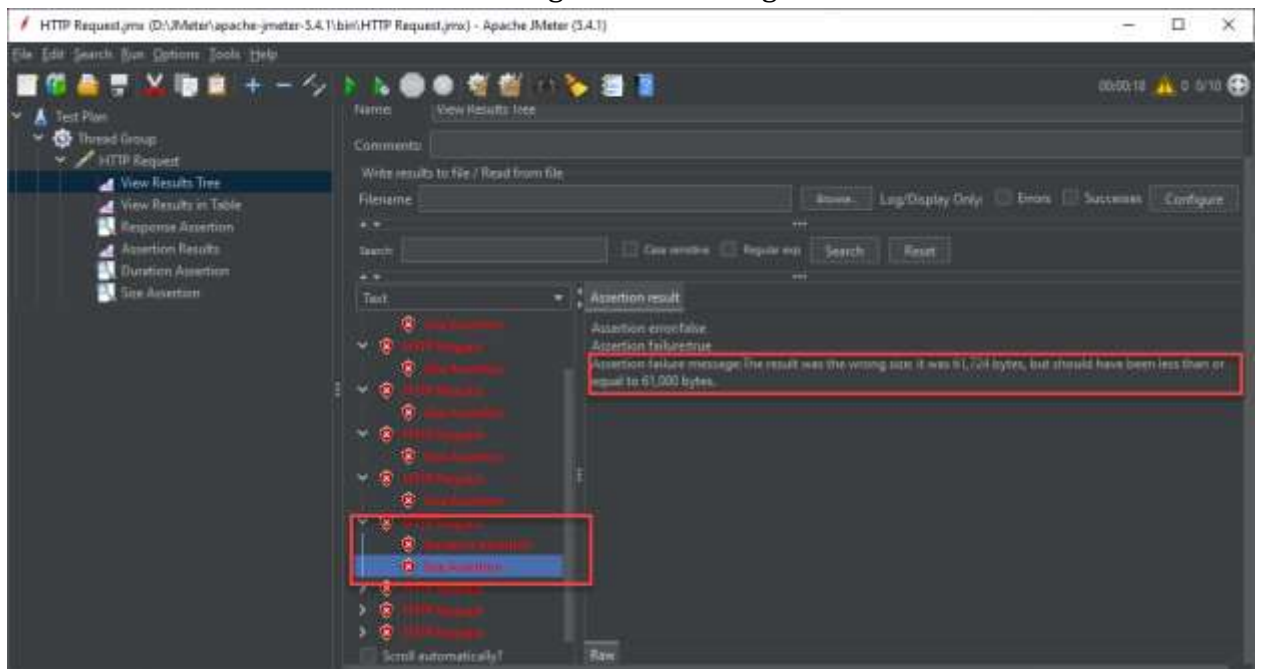
Now we can see our bytes size is around 61,000 and for size assertion we have some conditions like = / != / > / < / >= / <=

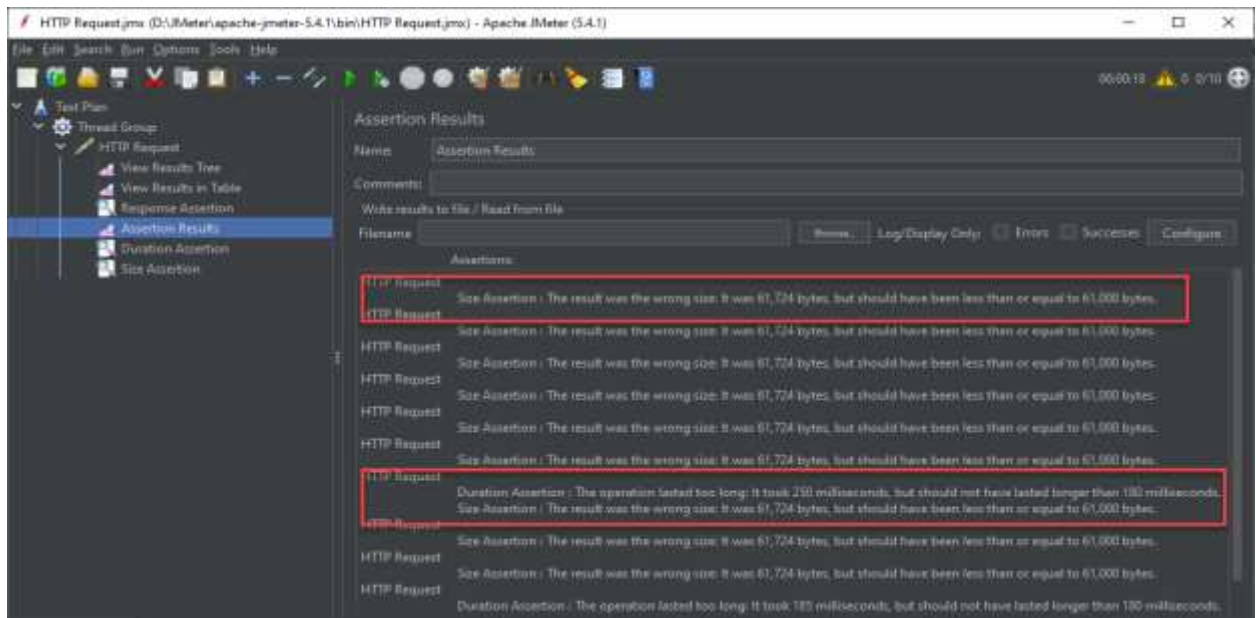
So if we give the input of 61000 in the size in bytes box and use >= condition we can see success in assertion result.



Success in assertion result.

Now if we use 61000 and <= we should get error message from assertion result.





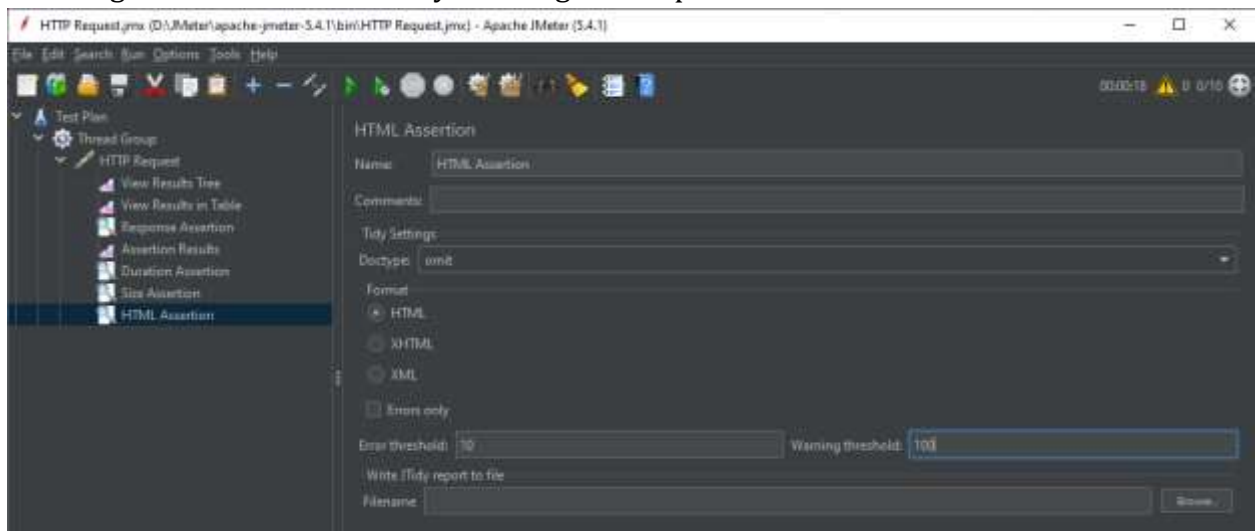
This is how size assertion works.

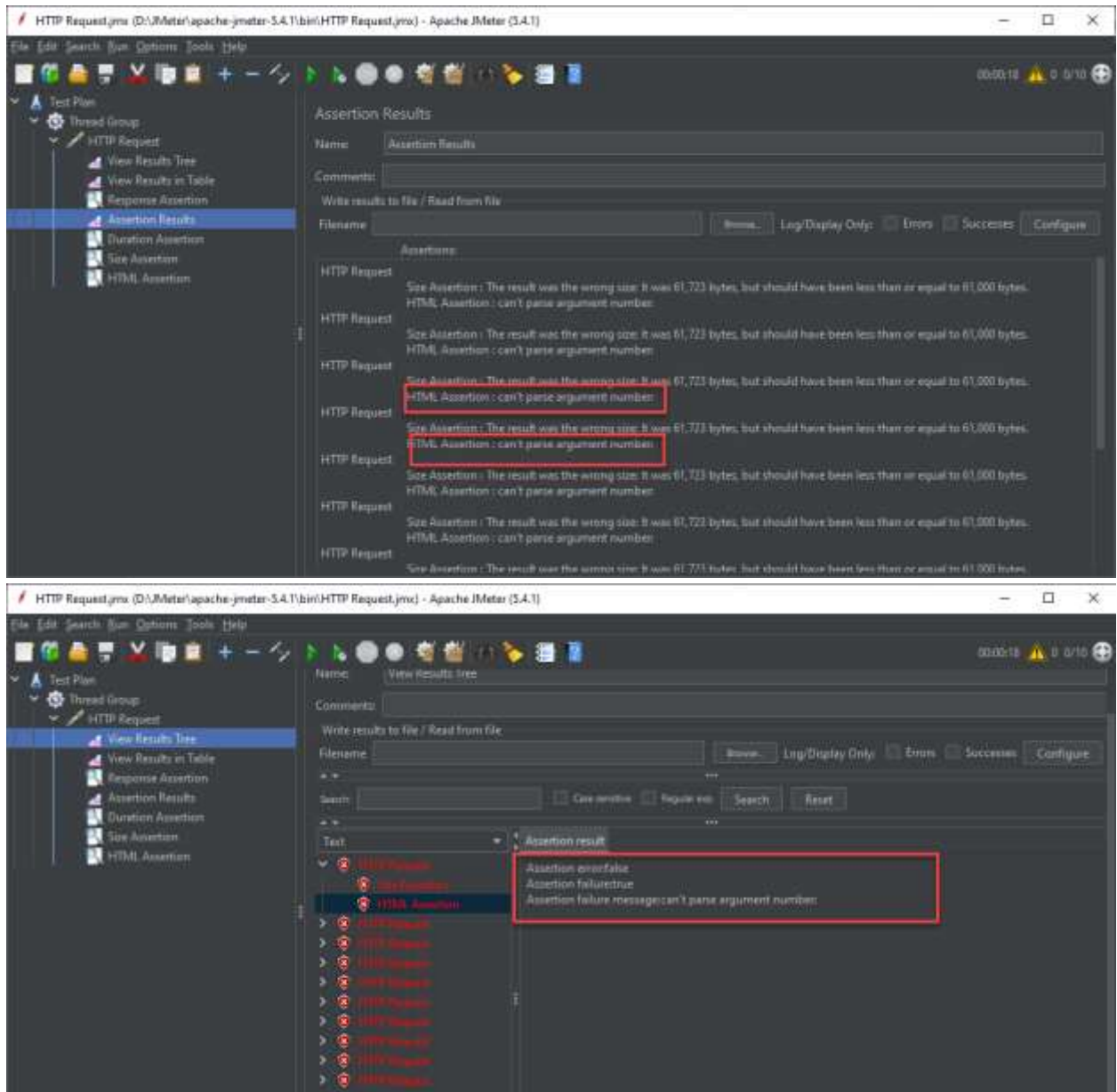
HTML Assertion:

- request (Sampler) > add > assertion > HTML Assertion.

There are two types of thresholds in HTML Assertion;

- 1) Error Threshold – How many errors are expected.
- 2) Warning Threshold – How many warnings are expected





Now let's create a text file in desktop and name it as report.txt. If we browse this file in HTML Assertion then we can find the error and warning list in the text file created in desktop named report.txt

Now let's save, clean and run the test plan again. Let's check the report.txt file in desktop. We can see all the errors and warning here.

Same assertion can be done for XML and XHTML.

```
line 24 column 6 - Warning: unknown attribute "sizes"
line 53 column 9 - Warning: <style> lacks "type" attribute
line 241 column 25 - Warning: <style> lacks "type" attribute
line 269 column 9 - Error: <header> is not recognized!
line 269 column 9 - Warning: discarding unexpected <header>
line 276 column 29 - Warning: trimming empty <div>
line 278 column 1 - Warning: <style> lacks "type" attribute
line 278 column 1 - Warning: <style> isn't allowed in <div> elements
line 296 column 29 - Warning: trimming empty <div>
line 309 column 183 - Warning: Warning: unescaped & or unknown entity "&entries"
line 309 column 194 - Warning: Warning: unescaped & or unknown entity "&sort"|
line 309 column 204 - Warning: Warning: unescaped & or unknown entity "&w"
line 311 column 99 - Warning: Warning: unescaped & or unknown entity "&entries"
line 311 column 110 - Warning: Warning: unescaped & or unknown entity "&Category"
line 311 column 129 - Warning: Warning: unescaped & or unknown entity "&sort"
line 311 column 139 - Warning: Warning: unescaped & or unknown entity "&w"
line 312 column 63 - Warning: Warning: unescaped & or unknown entity "&entries"
line 312 column 74 - Warning: Warning: unescaped & or unknown entity "&Category"
line 312 column 93 - Warning: Warning: unescaped & or unknown entity "&sort"
line 312 column 103 - Warning: Warning: unescaped & or unknown entity "&w"
line 313 column 63 - Warning: Warning: unescaped & or unknown entity "&entries"
line 313 column 74 - Warning: Warning: unescaped & or unknown entity "&Category"
line 313 column 97 - Warning: Warning: unescaped & or unknown entity "&sort"
line 313 column 107 - Warning: Warning: unescaped & or unknown entity "&w"
line 313 column 107 - Warning: Warning: unescaped & or unknown entity "&w"
```

This is basically checking your HTML or the HTML of your response against the standard syntax. If it finds anything wrong against the standard it will show us the result and can log the result in a text file.

If my application wants to interact with other 3rd party application then we need to take care of these warnings and error. That's why we need to perform HTML assertion in that case.

XML JSON and XPATH Assertion:

For API or web services we will need this assertion. We will be watching API testing with JMeter in future.

HTTP Test Script Recorder

What – a element in JMeter that helps to record http request instead of adding your request in a http sampler

Why – we need this kind of recorder so we don't have to add manually every request in our JMeter, we can just browse and do the actions and everything will get recorded. For observing the actions when multiple users are browsing the site. Proxy server allows to record user activity.

When – whenever we want to record our http actions like website or browser, we can use it.

Helps us to record our request from the browser

We can do it in our existing Test Plan or a new Test Plan. I am creating a new test plan.

Right Click on Test Plan > right click > add > Non-Test Elements > HTTP(s) Test script Recorder.

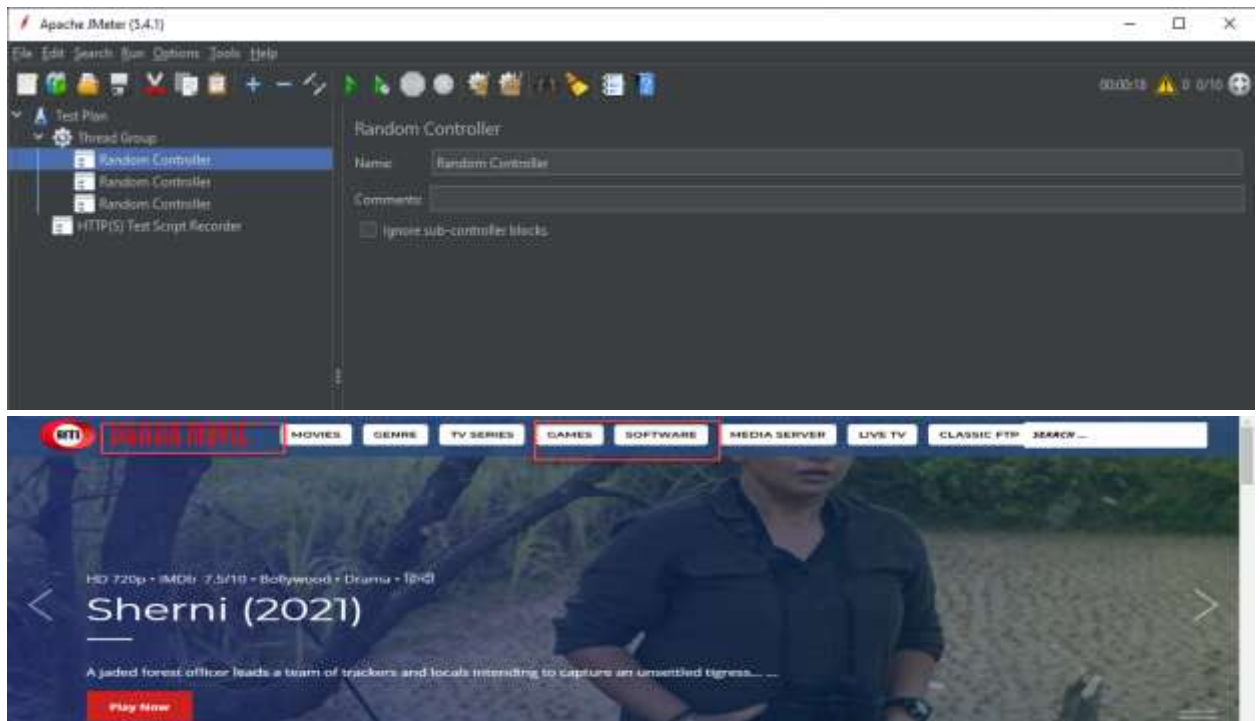
Add a thread group here.

Test plan > add > Threads > Thread Group

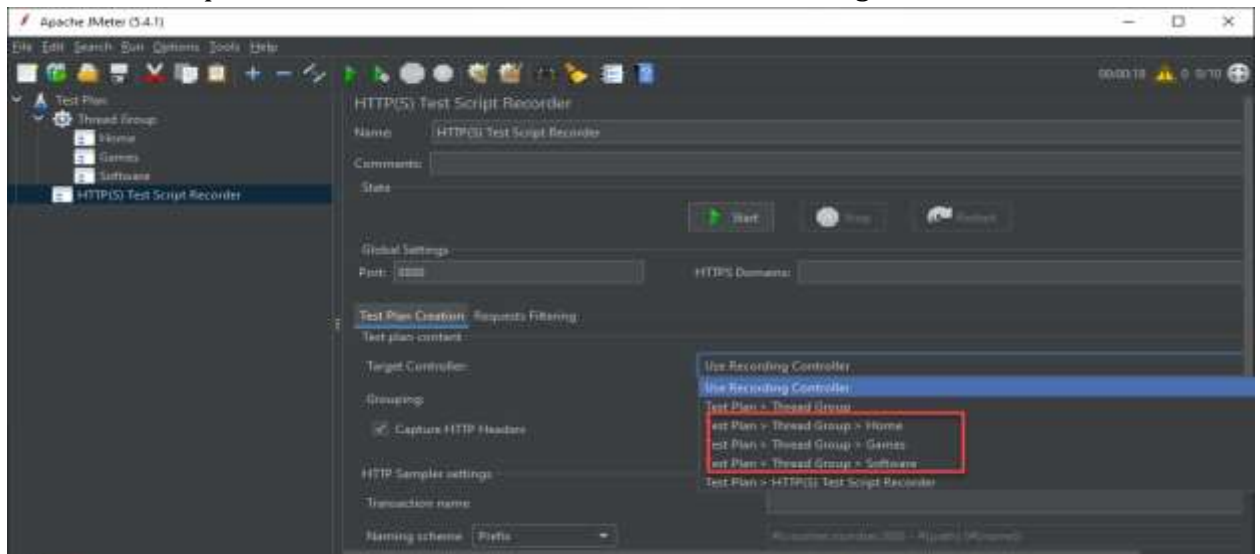
In thread group add a Logic controller

Right click on Thread Group > Add> Logic Controller> Random Controller

If we want to record three pages of a site, like Home, Games, Software in that case we need to add three Recording Controller.

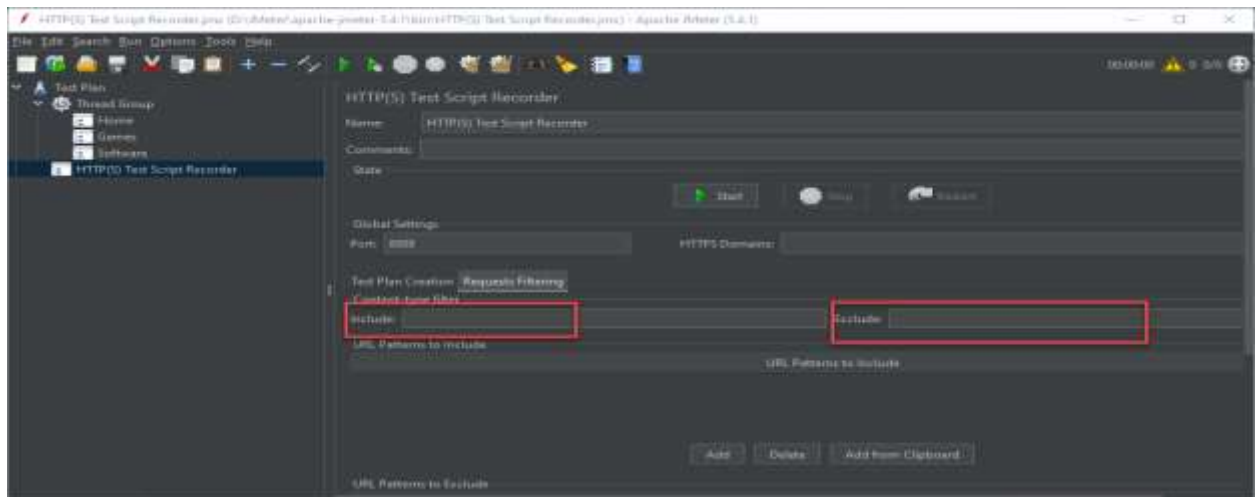


In the test Script Recorder, we can find out the all the recording



We will keep the grouping – Don not group samplers. Just keep it as it is default here. We will keep other settings as it is too.

If we go to request filtering, we can see that, if we want to exclude some URL patterns because when we hit any server there are lot of resources that get exchanged. If we want some patterns not to include in the request, we can add them here. Even if we want to include some patterns, we can also do that here.



Now let's get back to the test plan creation,

Now here in port we can see that the port is set to 8888, and we want that all the traffic from a particular website or a particular http network should only be recorded and not anything else and therefore we set up a port here and, in our browser, we can set a proxy for this particular port.

Let's see that in Firefox browser,

Sample API, =

```
api.openweathermap.org/data/2.5/weather?q=kolkata&APPID=b5227eb415d8925187dc818bdeba0098
```

```
Downloads\apache-jmeter-5.4.1\bin>jmeter -n -t \Users\HP\Downloads\apache-jmeter-5.4.1\bin\loadtest\load.jmx -l \Users\HP\Downloads\apache-jmeter-5.4.1\bin\loadtest\loadtest.csv
```