

Software Quality Assurance

15CSSE045

Final Report

Washington Post Test Report

Submitted by

|  |  |
| --- | --- |
| **Group Members** | **ID** |
| Lobna Hesham | 120431 |
| Mohab AbuGabal | 117276 |
| Ramy Mamdouh | 118371 |
| Mina Essam | 119791 |

Supervised by

Pro.Dr. Osman Ibrahim

And

Ms. Ghadeer Mobasher

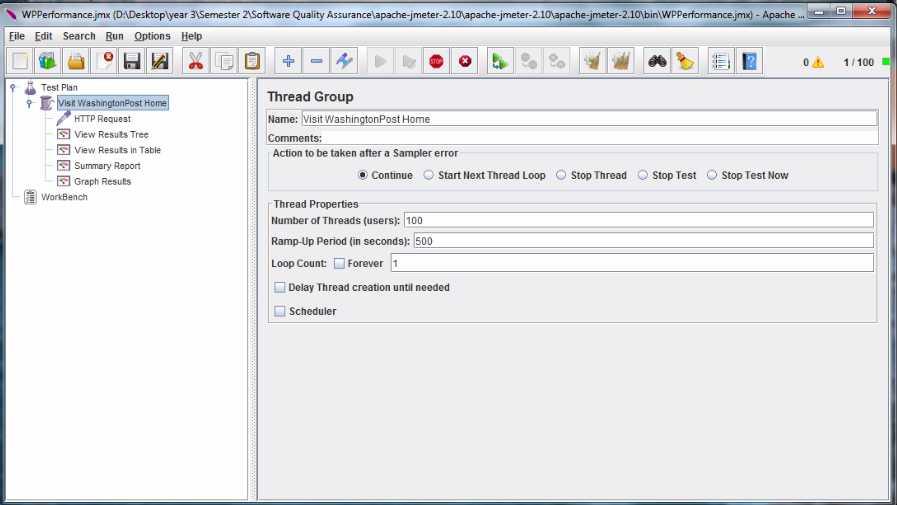
**Performance Testing**

A performance test was made for Washington post’s website. Jmeter, the graphical server performance testing software was used in order to test the website’s behavior.

**Thread Group**

The picture below shows that the number of threads entered are (100). Which means the number of users sending requests to the website are (100). Since the value of loop count is (1) it means that each user will send only one request (100\*1 = 100 requests) one request/ user. The value was assigned to the ramp-up period here is 500 seconds, and this number is used to tell the program the ***total*** amount of seconds taken by all threads. Inserting (500) into the ramp-up period does not mean the delay between each thread "request here" is 500 seconds, but it has a rule to calculate the delay (ramp-up period/ Number of Threads = 500/100= 5 seconds between each thread).

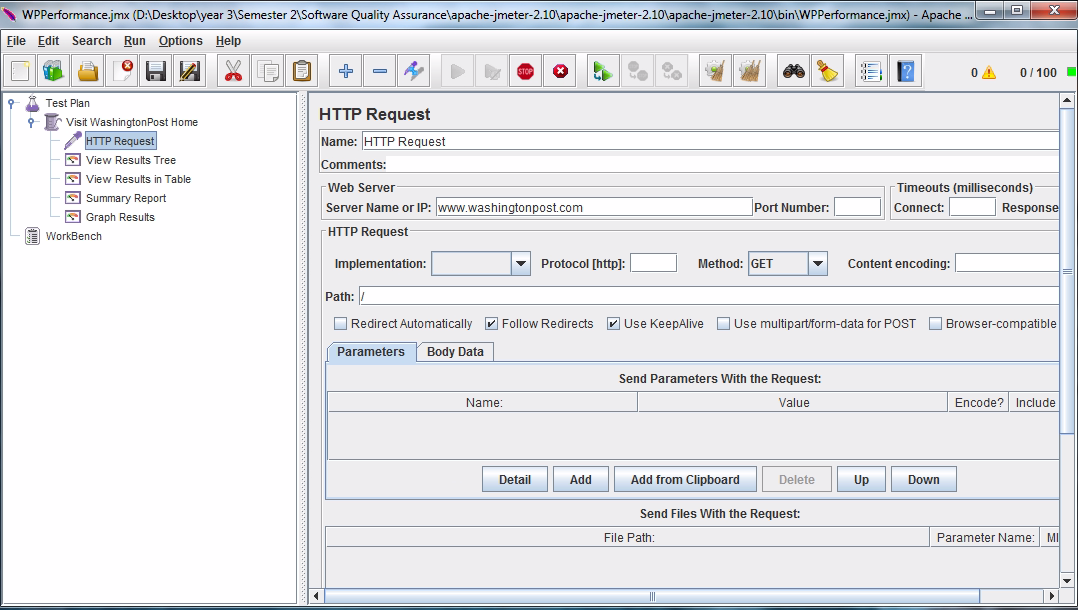
**Screenshot**



**HTTP Request**

* Name field takes the name of HTTP request, and it is assigned here as HTTP Request.
* "Server Name or IP" takes the IP address or URL of the website needs to be tested.
* "Path" represents the path of other page in the website to test, but here the home page is the one will be tested so we put "/" which means the root of the website "home page of the website".

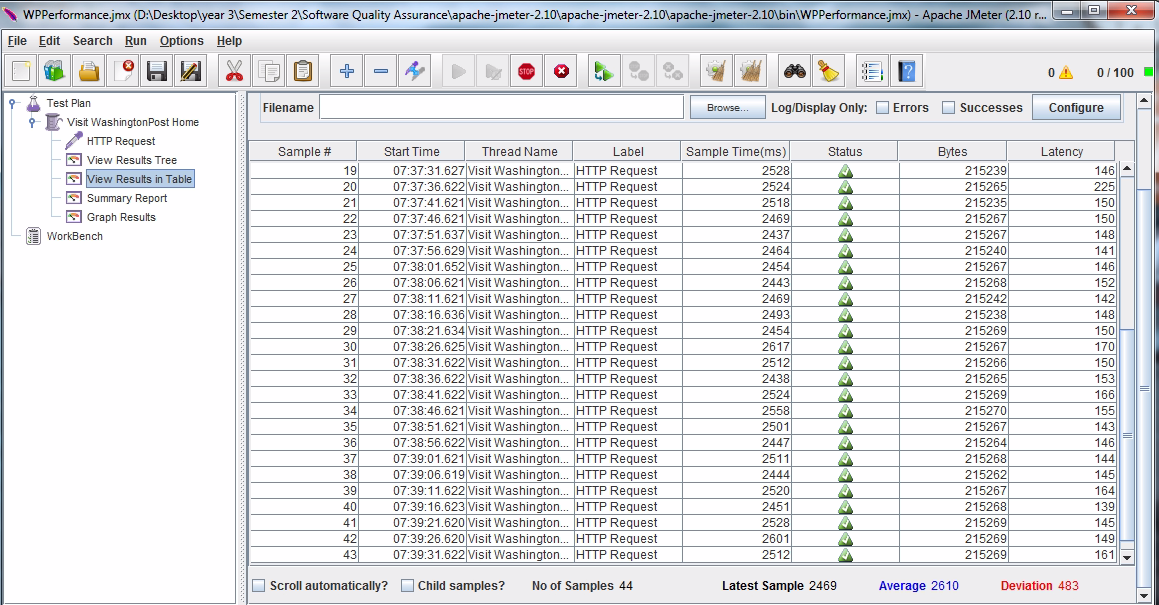
**Screenshot**



**View Results in Table**

* Sample #: Represents the thread number "Request number" was sent to the server and was gotten a response.
* Start Time: The time each thread not loop even if we have assigned more than one loop has been started "sent to the server". Since we set 500 to the ramp-up period and 100 users, the delay between each thread is 5 seconds. That is shown in the picture below the difference between of start time of thread number 20 and thread number 19 was 5 seconds.
* Thread Name: Holds name of threads for thread number 19 will be "Visit Washington Post Home 1 - 19" and for example if the loop count was 3, thread name would appear like "Visit Washington Post Home 1 - 19” three times at a different start time.
* Label: Holds the request type, which is HTTP request.
* Sample Time(ms): The time needed to finish a request. In other words; it is the time in milliseconds that the server take to fully serve the request (response + latency).
* Status: Represents the success or failure of a sent request. Green represents "success", while red represents "failure". Success or failure of getting a response from the server.
* Bytes: Represents the size of the received response(s).
* Latency: The time in milliseconds that elapsed between when JMeter sent the request and when an initial response was received.

**Screenshot**

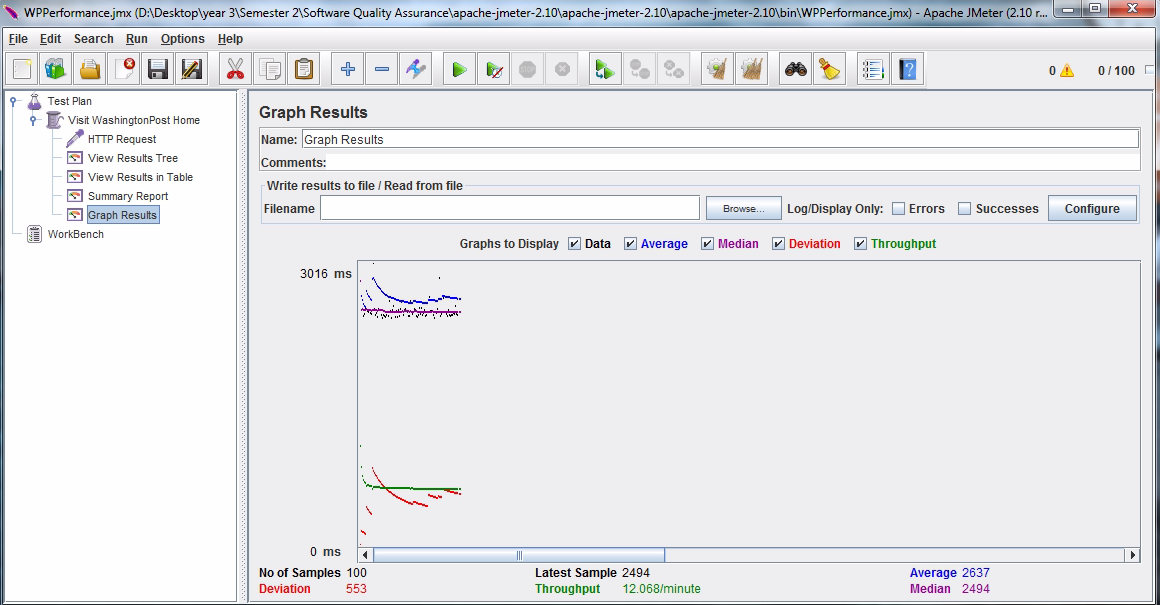


**Graph Results**

* Number of samples: Represents the number of sent and served samples, whether it succeeded or failed. Here in the picture below, all threads were sent and served.
* Latest sample: It is equal to the ***Sample time*** in ***View Results in Table,*** represents time in milliseconds that the server took to fully serve the request.
* Average: Current average of all samples, in other words,it represents the current average time in (ms) that the server took to fully serve the request "the average of response time".
* Deviation: It indicates the deviation from the average (will be illustrated more throughout the report).

Throughput: Represents the ability of the server to handle some load. "number of threads or users Per Minute". In this test with respect to the delay of "5 sec between the threads", server can handle almost 12 users per 1 minute. It is a normal load and number of sent requests is 12 request per minute, which means that all requests were handled by the server. When the number of users set was 100 and the ramp-up period was 100 "1 sec between each request" the number of users can be handled per minute from 20 to 21 users which is considered weak.

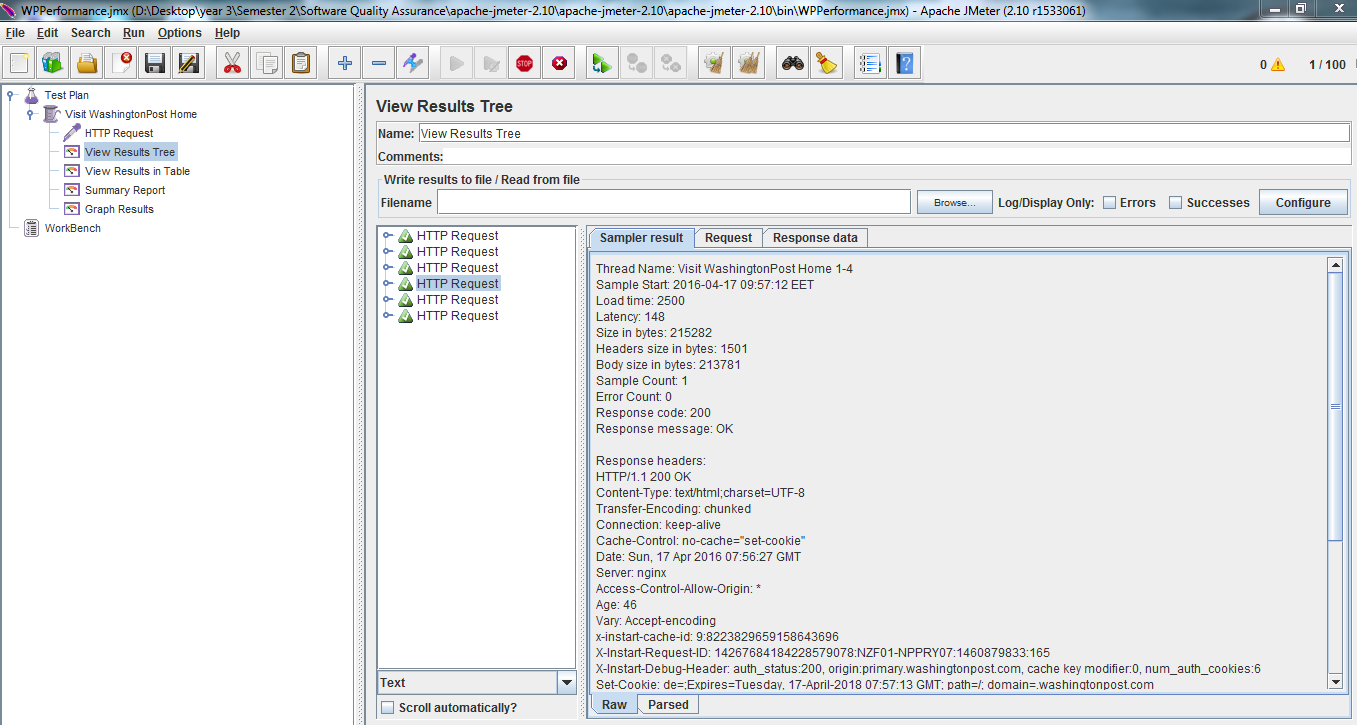
**Screenshot**

****

**View Results Tree**

Result tree shows data of each sent request. Each green HTTP Request shows that a successful request has gotten a response. In addition to the green HTTP request, the successful case appears in: the response code: 200, server message: OK.

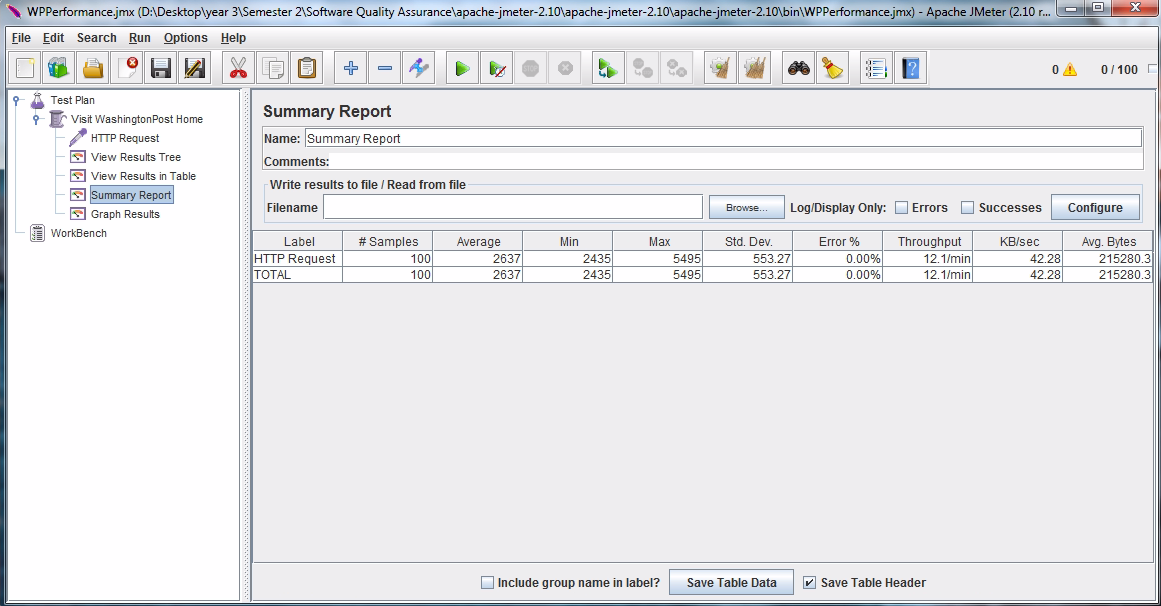
**Screenshot**



**Summary**

* Label: holds the request type and it is here only HTTP request. Similar to label in View results in table.
* Number of Samples: In the first row the number of current Sample. While in the second row it represents the number of total samples served till this moment.
* Average: Current average of all samples, in other words represents the current average time in (ms) that the server took to fully serve the request "the average of response time". Similar to Average in Graph results.
* Min: Represents the current Min time in (ms) that the server took to fully serve the request "the Minimum response time and the fastest response".
* Max: Represents the current Max time in (ms) that the server took to fully serve the request "the Maximum response time and the slowest response".
* standard deviation: The std dev (σ) measures the mean distance of the values to their average (μ). In other words, it gives us a good idea of the dispersion or variability of the measures to their mean value.
* Error %: The percentage of failed requests, or requests with error.
* Throughput: Represents the ability of the server to handle some load. "number of threads or users Per Minute" in the picture below 12 users per one minute.
* KB/sec: The throughput "number of requests per minute" measured in Kilobytes per second.
* Avg. Bytes: average size of the sample response in bytes.

**Screenshot**



**Assertions Performance Test**

All assertions results appear in the "***Assertion Result***".

In this test case the number of threads are 100 and the total delay between all threads are 400. Which makes the "Delay between each thread or user is 4 seconds".

**Duration Assertion**

Duration assertion tests that each response was received within a given amount of time, in our test case it is "3000 ms". Any response takes longer than 5000 milliseconds (specified by the user) is marked as a failed response.

The picture below indicates two cases of response results. First one represents a successful HTTP request, and the second one represents failed HTTP request.

The reason of failure in the second HTTP request is because the duration taken for response exceeded the 5 seconds (5000 milliseconds).

**Screenshot**



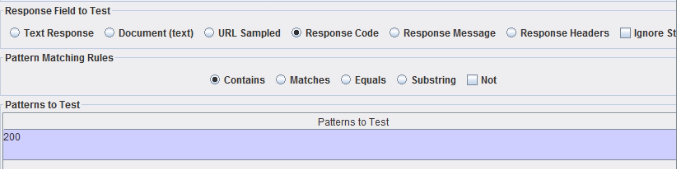
**Response Assertion**

Response Assertion lets the user add pattern strings to be compared against various fields of the response.

Here in this test case, the response will be compared against its response code.

The expected response code is assigned as 200 "Showed in the picture below".

**Screenshot**



As long as the response contains a response code of 200, the HTTP request will pass. If no other error appears such as, connection error, size assertion, duration assertion, etc.

A successful HTTP request does not show any output message except an "HTTP request", in the assertion result, as shown below.

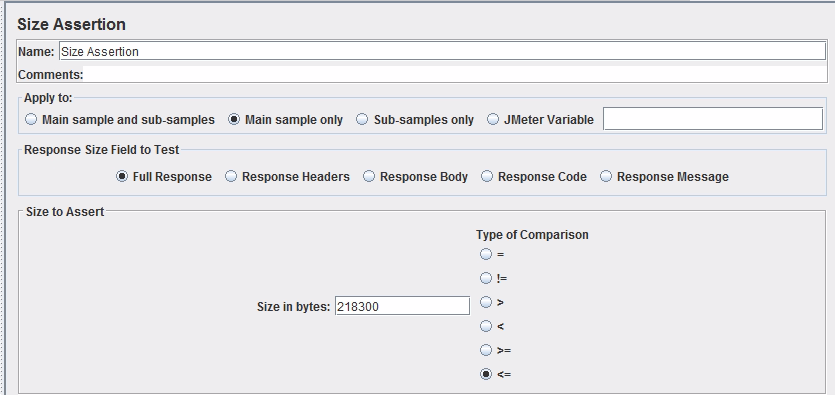
**Screenshot**



**Size Assertion**

Size assertion lets the user specify the response size. It can be specified as equal to, greater than, less than, greater than or equal, less than or equal or not equal to a given number of bytes.

In this test case the size specified as less than or equal to 218300 bytes.



If the received response size was less than or equal to 218300 bytes, no error related to the size assertion will appear. However, if the response size was greater, an error will appear as shown in the picture below.

**Stress Testing**

|  |  |
| --- | --- |
| 1,026,298 visitors/daily | |
| Average/hour | 42763 users/hour |
| Average/minute | 713 users/minute |
| ∴Average/second | 12 users /second |

let’s increase the number of Threads(users) gradually until the server is down. The number of Threads will be multiplied by two in each attempt. When the number of users have reached 60 users/second, some requests have been recorded a failure.

|  |  |
| --- | --- |
| Number of users | 60 |
| Status | 54 out of 60 requests are correctly delivered back |
| Reason(s) for lost requests | * The server is not listening to the port anymore, causing a crash * The internet connection went down either on the user side, server side, or in the middle bridging. * Some firewall, proxy, similar component between you and the server is denying connection |
| Screenshot | 6 requests are lost due to server crash (stress testing) |
| Error name | javax.net.ssl.SSLException: SSL peer shut down incorrectly |
| Difference between received and dropped request  (Red ->dropped  ,green->recieved) | Received requests:  Response code: 200  Response message: OK  Size in bytes: 215322(complete website)  Error Count: 0  Dropped requests :  Response code:  Non HTTP response code:  javax.net.ssl.SSLHandshakeException  Response message:  Non HTTP response message: Remote host closed connection during handshake  Size in bytes:  2771 (incomplete websites only headers and list of errors)  Error Count: 1(error is the dropped request) |
| Error | 12.00% |
|  | |

**Application’s behavior (when pushed beyond normal or peak load conditions)**

* Standard deviation: This shows how many exceptional cases were found, which were deviating from the average value of the receiving time.

In the above case, the value is relatively high, which means that the site is not consistent through high request.

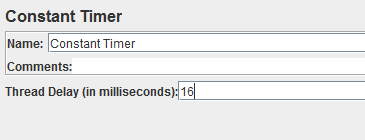
**Not consistent examples**

* Deliver back empty packets
* Deliver incomplete website
* Min, Max and average: In the above case, the value is very high as Washingtonpost.com include a lot of data content in their homepage
* Throughput: 27.6 /min which is considered to be invalid as in the previous test case 60 users were tested and only 6 requests were dropped

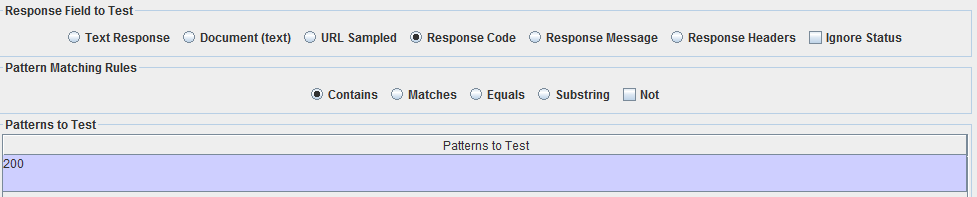
**Conclusion**

* Number of Requests were dropped When the application was pushed beyond limits
* Average of error increases by increasing the number of users/second

**Timer used**: 16 milliseconds between each request in order be consistent for all the calls, 1second/60 users



**Assert**: Make sure that Response code: 200, which means that the request was received and understood and is being processed.



**Load Testing**

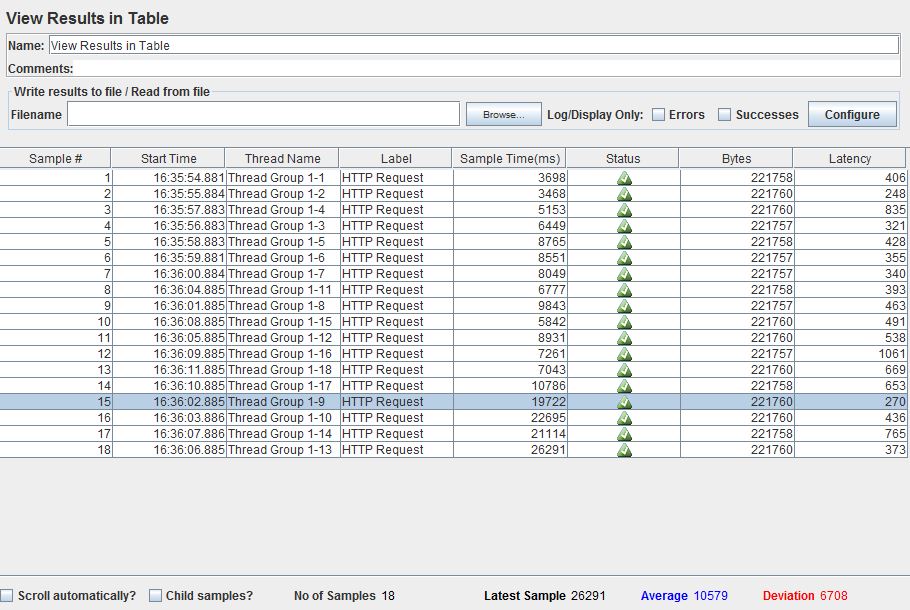
WashingtonPost.com has Daily traffic about (1,527,173 daily users. Which means that the server can handle 18 user request/second. Load Testing is defined as “Load Testing to verify the server behavior under normal and peak conditions”. It was noticed after many runs that the WashingtonPost.com sever can handle 18 users/second. The below report show the steps provided by screen shots of testing with explanation of the results.

**Screenshot**



As mentioned above the number of users is 18 users. As this is the peak of the server, Ramp up period is 18 which means 1 request per second, and loop count is 1 means each user will request only one HTTP request.

**Results Analysis**



Load Testing is focus on ensuring that all requests are sent and received with Ok (status is green)

1. Sample number: represent the number of thread(s) and total number of successful requests. In the website case, the 18 users have requested and received responses with no lost or dropped packets.
2. Start Time: Represents when the thread is initialized, requested and send to the server.
3. Thread Name: Which is the name of the thread( Group Thread).Group index defines the cluster of threads that might be available. While the thread index is the number of users, which is 18 in our case.
4. Sample Time: this variable indicates the response time of the server to client in milliseconds. In sample number 1 the server responds to the request in 3.698 seconds). In load testing it shows how many times the server takes to respond under maximum number of users.
5. Status: Holds the status of the request whether it passes or fail. In the above test it indicates all users who send request is received and the server responded successfully. It is very critical because in load test it should be that all requests send and received successfully.
6. Latest Sample: Holds the last Thread’s Sample time. In the above test case is 26291
7. Average: Is the summation of sample time / total number of samples. In the test case summation of samples = 190438, total number of samples = 18. Thus the average value will be 10579. The average is important in load testing because it experiences the speed of response time in normal and peak testing.
8. Deviation: In JMeter the deviation, it measures the variability of a data set and it is calculated using the following formula

Where

N is number of sample

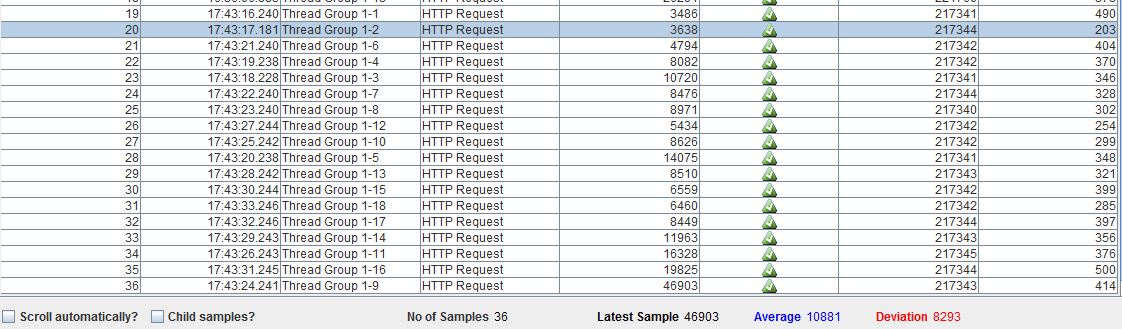
Xi is sample time

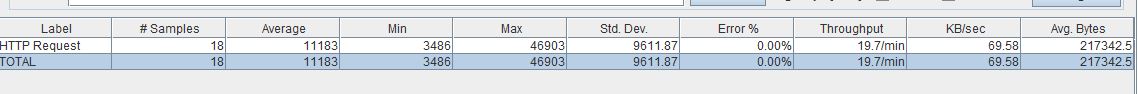
u is average of all sample time

It is important to show the degree of consistency of the server under the maximum

number of request it can handles.

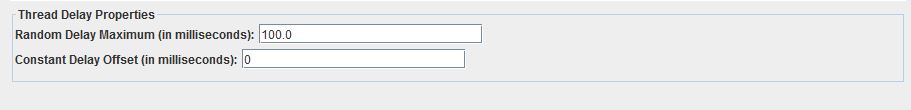
**Summary Report**



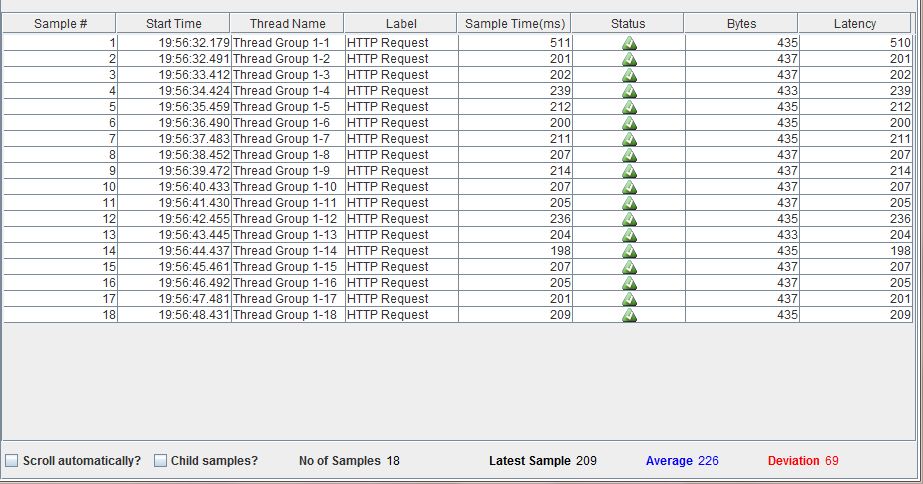


1. Average: 201299/18 = 11183. In load test it is important because it shows if the server is working as expected under maximum number or users or not.
2. Min: In the min time it is 3486ms, in load testing it is very important because it shows if the server will meet the minimum response time under maximum number of users or not.
3. Max: Maximum time (sample Time) taken by the server to response in the given sample = 46903. In load testing it shows if the server will meet the minimum response time under maximum number of users or not.
4. Error Percentage: Indicates percentage of failed tests, in the test case is zero because all tests are passed. This is perfect result in Load test because it ensures that the server is works under the normal and peak conditions.
5. Throughput: number of requests that is handled by the user/min. In load testing it is significance to ensure that the server handles the maximum number of users/min.
6. KB/Sec: throughput measured in Kilobyte/second.

**Results using Random Timer**



In this test case the program will start requesting a maximum of 100 milliseconds. After the previous user, this help in load performance to see the behavior of the server if it can handle the maximum number of requests or not.



**Results using Constant Timer**





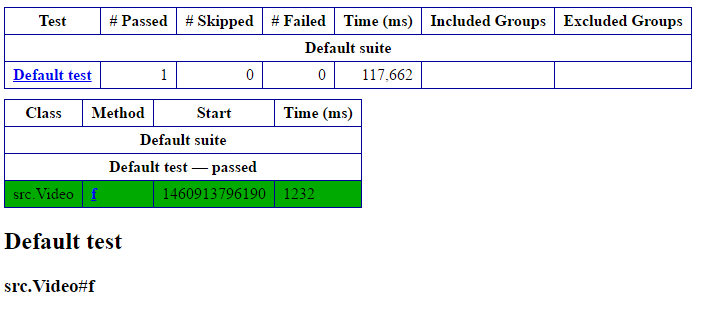
It was observed that the server is better in uniform timer because the value of deviation in compare to the value of deviation in constant is less and less is better and this show the consistency of the server in handling the maximum number of users requesting the server.

**TestNG**

**Test Case 1**

|  |  |
| --- | --- |
| ID test case Name | Video Streaming |
| Objective | * Testing the Websites videos |
| Name of tested module | Video-Washington Post |
| Test scenario | 1. Open <https://www.washingtonpost.com/> 2. Click on section select video or from navigation bar choose video 3. Choose video to display / search for specific video from entering the video name in video search bar then select the video that meets the search keyword |
| Input | In case searching for specific video the input will be the name of the video |
| Expected Output | Streaming the recent video in addition to trying the play, pause and reload video functions |
| Actual Output | 1. Video starts 2. Play video 3. Pause video 4. Then reload the video in order to start from the beginning |
| Pass/Fail | Pass |

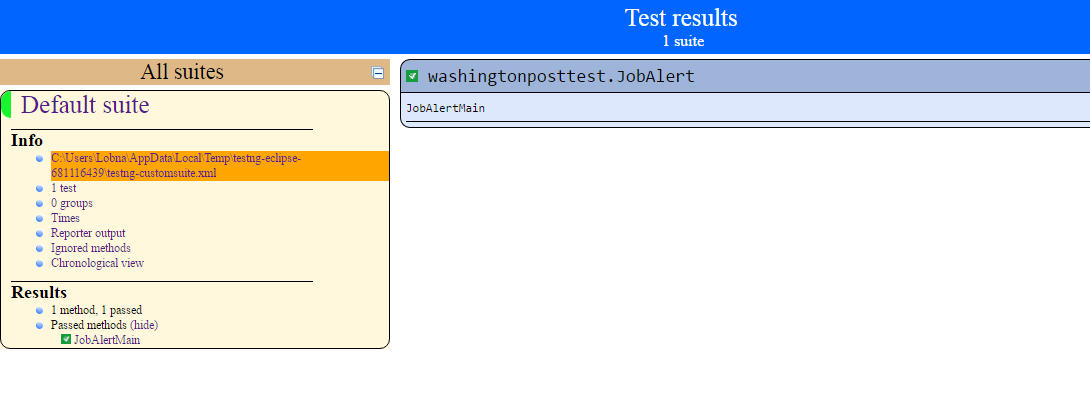
**Result and report**

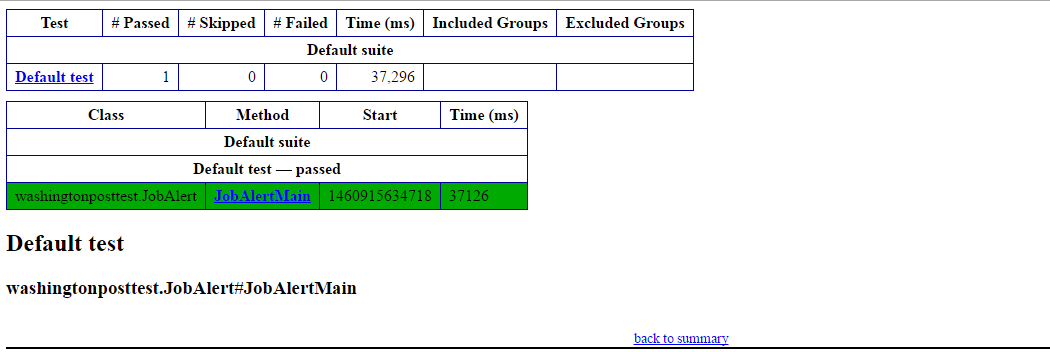


**Test Case 2**

|  |  |
| --- | --- |
| ID test case Name | Job Alert Testing |
| Objective | * The Website email the user when a job is available meeting his/her requirements |
| Name of tested module | Job Alert-Washington post |
| Test scenario | 1. Open <https://www.washingtonpost.com/> 2. Select Jobs from section, select Jobs from Navigation bar 3. The website navigate to <https://jobs.washingtonpost.com/> 4. Select Job Alert 5. The user should fill the form 6. The user press on “Email me jobs like this” button. |
| Input | 1. User E-mail 2. Keywords (marketing, sales, IT) 3. User location 4. Function (accountant, Engineer, Finance) etc. 5. Industry (Education, Automotive) etc. 6. Career level 7. Hours 8. Recruiter type 9. Salary Band |
| Executed Output | Successful Message to show that user successful create job alert |
| Actual Output | Successful message showing the successful of creating a job alert |
| Pass/Fail | Pass |

**Result and report**

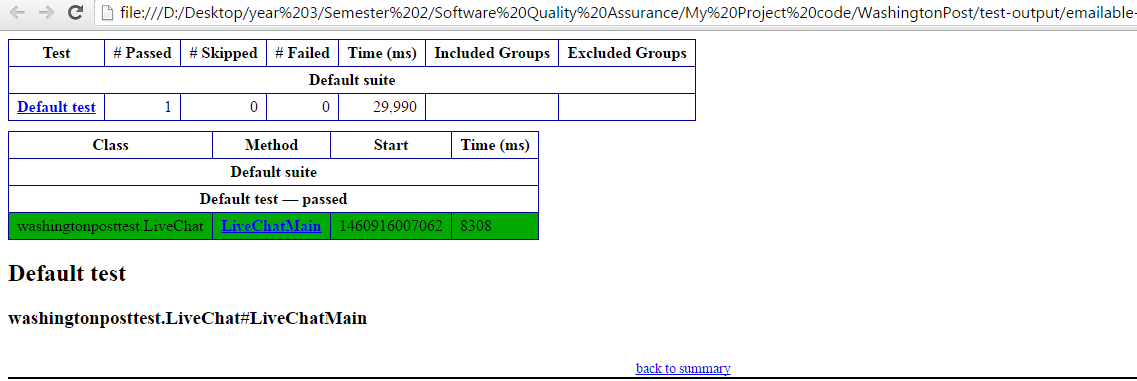
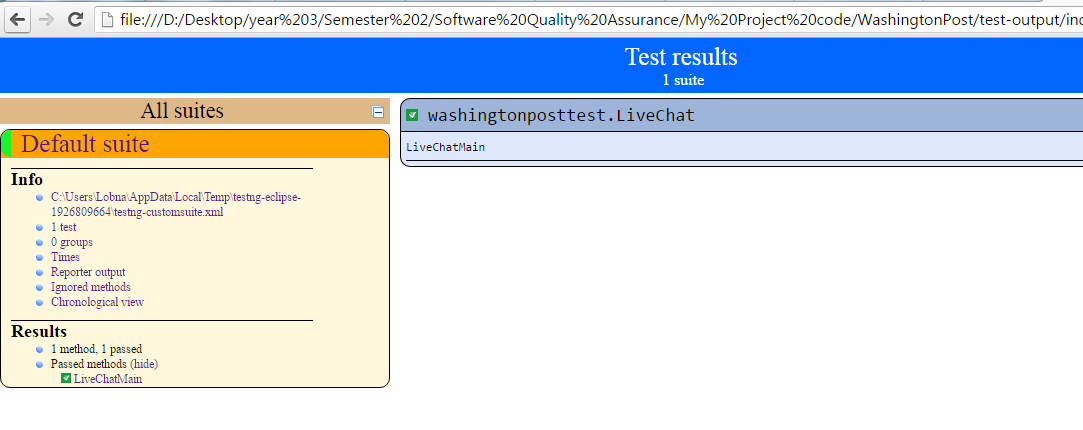




**Test Case 3**

|  |  |
| --- | --- |
| ID test case Name | Live Chat Questions |
| Objective | * To Test posting questions to the favorite writer |
| Name of tested module | Live Chat-Washington Post |
| Test scenario | 1. Open <https://www.washingtonpost.com/> 2. Click on sections 3. Choose Live Chats 4. There is a calendar contains the name of the programs, the user chooses the program 5. The user must sign in and have an active subscription. 6. The website redirect to the page of the chosen program, there is a button “Ask Now” 7. The user enter topic, question 8. User click submit |
| Input | 1. Topic 2. User Question |
| Executed Output | Live Chat between user and writer Ex. (Skype call) |
| Actual Output | Form to submit your questions |
| Pass/Fail | Pass |

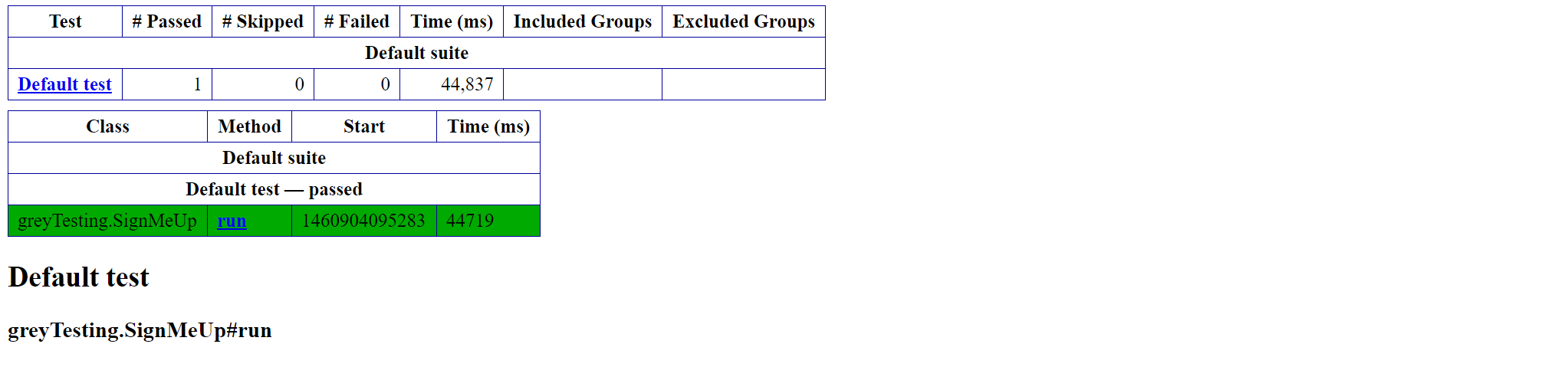
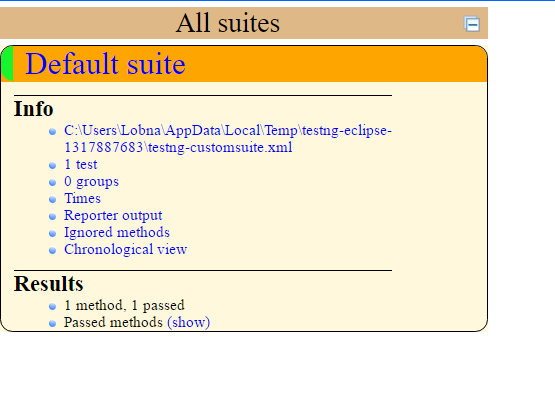
**Result and report**



**Test Case 4**

|  |  |
| --- | --- |
| ID test case Name | Sign up (positive testing) |
| Objective | Testing the process of creating an account. Valid data are required in each text field. |
| Name of tested module | Sign Up |
| Test scenario | 1. open [www.washingtonpost.com](http://www.washingtonpost.com) 2. Click on ‘sign in’ 3. Click on ‘create on today’ 4. Fill the required eleven account information with valid data. 5. Click on ‘Create new account’ 6. Automatically directed to home page |
| Input | 1. Valid email address 2. Password and password confirmation 3. Select the job industry 4. Company Size 5. Location 6. Job title 7. Zip code 8. Responsibilities 9. Year of birth |
| Executed Output | Automatically direct to home page using the new created account. |
| Actual Output | Display the home page, showing the account that is signed in. |
| Pass/Fail | Pass |

**Result and report**



**Junit Testing**



**Difficulties of Testing Phase**

1. Internet connection and the computer processing affects the understanding and analyzing of the results. This may sometimes lead to inaccurate results.
2. In some cases, the request packets may fall/fail because of the internet connection rather than the server itself. This affects even more when it comes to the understanding of the test results.
3. Test results are not fully accurate; even though they are referenced from official web pages. As they do not reflect our own analysis.
4. Testing a bigger number of users(Threads) requires more time. Whenever the number of users increase the processing time is affected
5. When a request is fail/dropped, Jmeter does not specify the reason.

# References

Jha, R. (2014, 4 21). *Jmeter Report Analysis*. Retrieved from Oodles Technology: http://www.oodlestechnologies.com/blogs/Jmeter-Report-Analysis

*jmeter*. (n.d.). Retrieved from Jmeter Apache: http://jmeter.apache.org/usermanual/component\_reference.html

*Ranks*. (2016, February 17). Retrieved from Washington Post : http://rankw.org/site/washingtonpost.com

singh, p. (2014, Novembe 11). *How to analyze jmeter*. Retrieved from Push Pen Drapps : https://pushpendrapps.wordpress.com/2014/11/11/how-to-analyze-the-jmeter-performance-report/

*Use Apache JMeter*. (2014, June 20). Retrieved from Digital Ocean: https://www.digitalocean.com/community/tutorials/how-to-use-apache-jmeter-to-perform-load-testing-on-a-web-server