**Performance Testing: Assignment**

**1. What is Performance Testing?**

Software Performance testing is type of testing perform to determine the performance of system to major the measure, validate or verify quality attributes of the system like responsiveness, Speed, Scalability, Stability under variety of load conditions. The system is tested under a mixture of load conditions and check the time required responding by the system under varying workloads.

**2. Types of Performance Testing?**

* **Load testing** - checks the application's ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.
* **Stress testing** - involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify breaking point of an application.
* **Endurance testin**g - is done to make sure the software can handle the expected load over a long period of time.
* **Spike testing** - tests the software's reaction to sudden large spikes in the load generated by users.
* **Volume testing** - Under Volume Testing large no. of. Data is populated in database and the overall software system's behavior is monitored. The objective is to check software application's performance under varying database volumes.
* **Scalability testing** - The objective of scalability testing is to determine the software application's effectiveness in "scaling up" to support an increase in user load. It helps plan capacity addition to your software system.

**3. What is Jmeter and Advantages of Jmeter over other performance testing tools.**

JMeter is an software that can be used to execute performance testing, load testing and functional testing of your web applications. JMeter can also simulate a heavy load on a server by creating tons of virtual concurrent users to web server.

### **Advantages of using JMeter**

* **Open Source:** JMeter is an open source software. This means that it can be downloaded free of cost. It is also a 100% pure Java application. The developer can use its source code, can modify and customize it as per their requirement. They can also contribute their code to make a better JMeter.
* **Ease of Use:** The user can install and use JMeter easily. Just download it from internet, install and run. As a pure Java desktop application, it comes ready to use with default settings. It does not require you to have any specific skills or domain knowledge to use it.
* **Platform independent:** JMeter is developed in Java, which is the most popular programming language in the world. Therefore, it can run in any OS be it Window, Linux or Mac.
* **Robust Reporting:** JMeter can generate the effective reporting. The test result can be visualized by using Graph, Chart, and Tree View. JMeter supports different formats for reporting like text, XML, HTML and JSON.
* **Ultimate Testing:** With JMeter, user can do any kind of testing they want. Load Test, Stress Test, Functional Test, Distributed Test, all in one tool.
* **Flexibility:** You can customize JMeter as per your requirement and apply the automation testing to JMeter. You can save the effort of executing test cases manually.
* **Multi Protocol Support:** JMeter supports several protocols like HTTP, FTP, SOAP, JDBC, JMS, and LDAP. It can also be used for testing the performance of your database.

**4. What is Thread Group in Jmeter?**

A Thread Group is a set of threads executing the same scenario. It is the base element for every JMeter test plan.There are multiple thread groups available which can be configured to simulate how the users interact with the application, how the load is maintained and over what period of time.

## **Thread Group**

* **Number Of Threads:** It represents the total number of virtual users performing the test script execution.
* **Ramp-Up Period (in seconds):** It tells JMeter how long to take to reach the full number of threads. For example, if you have 100 users with a ramp-up period of 50 seconds, JMeter will take 50 seconds to get all 100 threads running, adding 2 threads per second.
* **Loop Count:** It is the number of executions for the script. For example, if the loop count is 2 and number of threads is 100 then the script will run 200 times. If the loop count is set “forever” then new threads will keep starting until the tests are stopped.
* **Delay Thread Creation Until Needed:** If this option is checked, the ramp-up delay and startup delay are performed before the thread data is created. If not checked, all the data required for the threads is created before starting the execution of a test.
* **Scheduler:** This schedules the tests. You can set custom duration and startup delay to create the threads in this section.

**5. Write down and explain any 5 listeners present in Jmeter?**

Listeners are the JMeter component that displays test results. All Listeners can be found by clicking Add->Listeners.

**Following are the List of listeners**

# **View Results Tree:**

The **View Results Tree** listener displays samples that the JMeter samplers generate, and the assertion results that are related to these samples. This listener displays the samples in the order they are generated by the JMeter script ,and provides parameters and data for each of them.

**Summary Report:**

* **Label**: It is the name/URL for the specific HTTP(s) Request. If you have selected “Include group name in label?” option then the name of the Thread Group is applied as the prefix to each label.
* **#Samples**: This indicates the number of virtual users per request.
* Average: It is the average time taken by all the samples to execute specific label. In our case, average time for Label 1 is 942 milliseconds & total average time is 584 milliseconds.
* **Min:** The shortest time taken by a sample for specific label. If we look at Min value for Label 1 then, out of 20 samples shortest response time one of the sample had was 584 milliseconds.
* **Max:** The longest time taken by a sample for specific label. If we look at Max value for Label 1 then, out of 20 samples longest response time one of the sample had was 2867 milliseconds.
* **Std. Dev.**: This shows the set of exceptional cases which were deviating from the average value of sample response time. The lesser this value more consistent the data. Standard deviation should be less than or equal to half of the average time for a label.
* **Error%**: Percentage of Failed requests per Label.
* **Throughput:** Throughput is the number of request that are processed per time unit(seconds, minutes, hours) by the server. This time is calculated from the start of first sample to the end of the last sample. Larger throughput is better.
* **KB/Sec:** This indicates the amount of data downloaded from server during the performance test execution. In short, it is the Throughput measured in Kilobytes per second.

**Aggregate Report**

The Aggregate Report listener shows the aggregated and statistical data for each sample of the script.

* **Label** - The label of the sample. If "Include group name in label?" is selected, then the name of the thread group is added as a prefix. This allows identical labels from different thread groups to be collated separately if required.
* **# Samples** - The number of samples with the same label
* **Average** - The average time of a set of results
* **Median** - The median is the time in the middle of a set of results. 50% of the samples took no more than this time; the remainder took at least as long.
* **90% Line** - 90% of the samples took no more than this time. The remaining samples took at least as long as this. (90th percentile)
* **95% Line** - 95% of the samples took no more than this time. The remaining samples took at least as long as this. (95th percentile)
* **99% Line** - 99% of the samples took no more than this time. The remaining samples took at least as long as this. (99th percentile)
* **Min** - The shortest time for the samples with the same label
* **Max** - The longest time for the samples with the same label
* **Error %** - Percent of requests with errors
* **Throughput** - the Throughput is measured in requests per second/minute/hour. The time unit is chosen so that the displayed rate is at least 1.0. When the throughput is saved to a CSV file, it is expressed in requests/second, i.e. 30.0 requests/minute is saved as 0.5.
* **Kb/sec** - The throughput measured in Kilobytes per second Average - The average time of a set of results

**6. Explain How do you recorded/made your script in your language.**

Installation of Jmeter in ubuntu system

* From <https://jmeter.apache.org/> download the software.
* After downloading extract that file and save it in home directory.
* From /apache-jmeter-5.0/bin directory run the jmeter.sh shell file in cmd.

Running Script in jmeter

* In jmeter, right click on **test plan** select add then select **Threads (users)**  in threads select **thread group** .
* Given name to the thread group. Ex : login
* Right click **thread group(login)** select add then select **Logic controller** in that select **recording controller** .
* Right click **thread group(login)** select add then select **Listener** in that select **View Result tree.**
* right click on **test plan** select add then select **Non test Elements**  in that select  **HTTP(s) test script recorder** .
* In **HTTP(s) test script recorder** we record the script.
* In  **HTTP(s) test script recorder** we can add request filtering for skip the unwanted files.
* Click on start in **HTTP(s) test script recorder.**
* In Browser(firefox) we need to set proxy for recording a script.
* For setting proxy in firefox , click on preference -> network proxy ->settings -> click manual proxy configuration
* Set http proxy as **localhost** and port **8888**
* For adding certificate in browser ,click on preference -> certificates -> view certificates ->servers .
* In **servers** click on **add exception** and add web page url that we want to test.(keep jmeter run for adding certificates in browser.)
* In **Authorities** click on **import** and add certificate of jmeter ( /apache-jmeter-5.0/bin/ApacheJMeterTemporaryRootCA.crt)
* Load the web page we want to test.
* Enter the username and password and click on sign up.
* After signup click on logout.
* Stop the jmeter script.
* All the recorded script will display in **recording controller.**
* From **view result tree** listener we can run the script and display the output.

**7. Why do we need to do performance testing?**

Performance Testing is done to provide stakeholders with information about their application regarding speed, stability and scalability. More importantly, Performance Testing uncovers what needs to be improved before the product goes to market. Without Performance Testing, software is likely to suffer from issues such as: running slow while several users use it simultaneously, inconsistencies across different operating systems and poor usability. Performance testing will determine whether or not their software meets speed, scalability and stability requirements under expected workloads.