PERFORMANCE TESTING:

- **Performance Testing is a software testing process used for testing the speed, response time, stability, reliability, scalability, and resource usage of a software application under a particular workload.**
- **The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application.**
- **It is a subset of performance engineering and is also known as *"Perf Testing"*.**

ATTRIBUTES:

Speed – Determines whether the application responds quickly

Scalability – Determines the maximum user load the software application can handle.

Stability – Determines if the application is stable under varying loads.

Reliability – It checks whether the software can perform a fault-free task for a given period of time in a specified environment.

IMPORTANCE OF TESTING:

- **Regular performance tests will help ensure your website performs at its highest level, resulting in better uptime, less maintenance, and greater user interactivity while on site.**

- **This investment, while taking considerable time to build, can have real, long-term benefits for your business, including great user experiences that directly results in return customers.**

WORKFLOW:

1. REQUIREMENT GATHERING:

The performance team interacts with the client for identification and gathering of requirements – technical and business. This includes getting information on the application's architecture, technologies, and database used, intended users, functionality, application usage, test requirement hardware & software requirements, etc.

2. TOOL SELECTION:

Once the key functionality is identified, POC (Proof Of Concept – which is a sort of demonstration of the real-time activity but in a limited sense) is done with the available tools.

The list of available tools depends on the cost of the tool, protocol that application is using, the technologies used to build the application, the number of users we are simulating for the test, etc. During POC, scripts are created for the identified key functionality and executed with 10-15 virtual users.

3.TEST PLAN:

Depending on the information collected in the preceding stages, test planning and designing are conducted.

Test Planning involves information on how the performance test is going to take place – test environment, workload, hardware, etc.

4.TEST DEVELOPMENT:

- **Use cases are created for the functionality identified in the test plan as the scope of PT.**
- **These use cases are shared with the client for their approval. This is to make sure the script will be recorded with the correct steps.**
- **Once approved, script development starts with a recording of the steps in use cases with the performance test tool selected during the POC (Proof of Concepts) and enhanced by performing Correlation (for handling dynamic value), Parameterization (value substitution) and custom functions as per the situation or need. More on these techniques in our video tutorials.**
- **The Scripts are then validated against different users.**
- **Parallel to script creation, the performance team also keeps working on setting up the test environment (Software and hardware).**
- **The performance team will also take care of Metadata (back-end) through scripts if this activity is not taken up by the client.**

5.TEST MODELING:

Performance Load Model is created for the test execution. The main aim of this step is to validate whether the given Performance metrics (provided by clients) are achieved during the test or not. There are different approaches to create a Load model. "Little's Law" is used in most cases.

6.TEST EXECUTION:

The scenario is designed according to the Load Model in Controller or Performance Center but the initial tests are not executed with maximum users that are in the Load model.

Test Execution is done incrementally.
For Example, If the maximum number of users is 100, the scenarios are first run with 10, 25, 50 users and so on, eventually moving on to 100 users.

7. TEST RESULT ANALYSIS:

Test results are the most important deliverable for the performance tester. This is where we can prove the ROI (Return on Investment) and productivity that a performance testing effort can provide.

8. REPORT:

Test results should be simplified so the conclusion is clearer and should not need any derivation. Development Team needs more information on analysis, comparison of results, and details of how the results were obtained.

TYPES:

1. **Load Testing**
2. **Stress Testing**
3. **Soak Testing**
4. **Spike Testing**
5. **Scalability Testing**
6. **Volume Testing**
7. **Connection Testing**
8. **Production Activity Test**

1. LOAD TESTING:

The load testing is used to check the performance of an application by applying some load which is either less than or equal to the desired load is known as load testing.

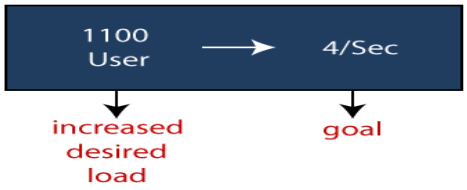
For example:

In the below image, 1000 users are the desired load, which is given by the customer, and 3/second is the goal which we want to achieve while performing a load testing.

2. STRESS TESTING:

The stress testing is testing, which checks the behavior of an application by applying load greater than the desired load.

For example: If we took the above example and increased the desired load 1000 to 1100 users, and the goal is 4/second. While performing the stress testing in this scenario, it will pass because the load is greater (100 up) than the actual desired load.



3. SOAK TESTING:

In this type of testing, we will check the behavior of an application on the environment, which is unsupportive for a long duration of time is known as soak testing.

5.SPIKE TESTING:

Spike testing is a type of performance testing in which an application receives a sudden and extreme increase or decrease in load.

5. SCALABILITY TESTING:

Checking the performance of an application by increasing or decreasing the load in particular scales (no of a user) is known as scalability testing. Upward scalability and downward scalability testing are called scalability testing.

TYPES:

- Upward scalability testing
- Downward scalability testing
- **UPWARD SCALABILITY TESTING:**

  It is testing where we increase the number of users on a particular scale until we get a crash point. We will use upward scalability testing to find the maximum capacity of an application.

- **DOWNWARD SCALABILITY TESTING**

  The downward scalability testing is used when the load testing is not passed, then start decreasing the no. of users in a particular interval until the goal is achieved. So that it is easy to identify the bottleneck (bug).

6. VOLUME TESTING:

Volume testing is testing, which helps us to check the behavior of an application by inserting a massive volume of the load in terms of data is known as volume testing, and here, we will concentrate on the number of data rates than the number of users.

7. CONNECTION TESTING:

All of the application users connect to the system around the same time and then grab a coffee or chat with their colleagues. Even if there are no major activities on the business components of the application, the architecture has to handle this very important peak of user sessions. It should be obvious, then, to ensure that the system won't break every morning. The operation team's lives will be made much easier by validating this point. This test will ramp up all the expected users.

8. PRODUCTION ACTIVITY TEST:

Every application has major business actions. These actions often result in data being updated or inserted into the database. Therefore, there should be a good understanding of the number of records created per day, or per hour. The production activity test ensures that the system can handle the load related to the number of transactions/hour expected in production. This test will load and validate the behavior of the business components within the database.

TOOLS:

- **Apache JMeter**
- **LoadRunner[HP]**
- **LoadNinja**
- **WebLOAD**
- **LoadComplete**
- **NeoLoad**
- **LoadView**