

Performance Testing TCF Documentation

February 2022

Contents

[Why Performance testing? 3](#_Toc96522186)

[Performance test during a sprint for a stable project 3](#_Toc96522187)

[Tools used to do Performance testing. 4](#_Toc96522188)

[Types of performance testing 5](#_Toc96522189)

[Process followed while doing performance testing in project 5](#_Toc96522190)

[Analysis of results 7](#_Toc96522191)

[Post Run activities 7](#_Toc96522192)

[Archiving the run data. 8](#_Toc96522193)

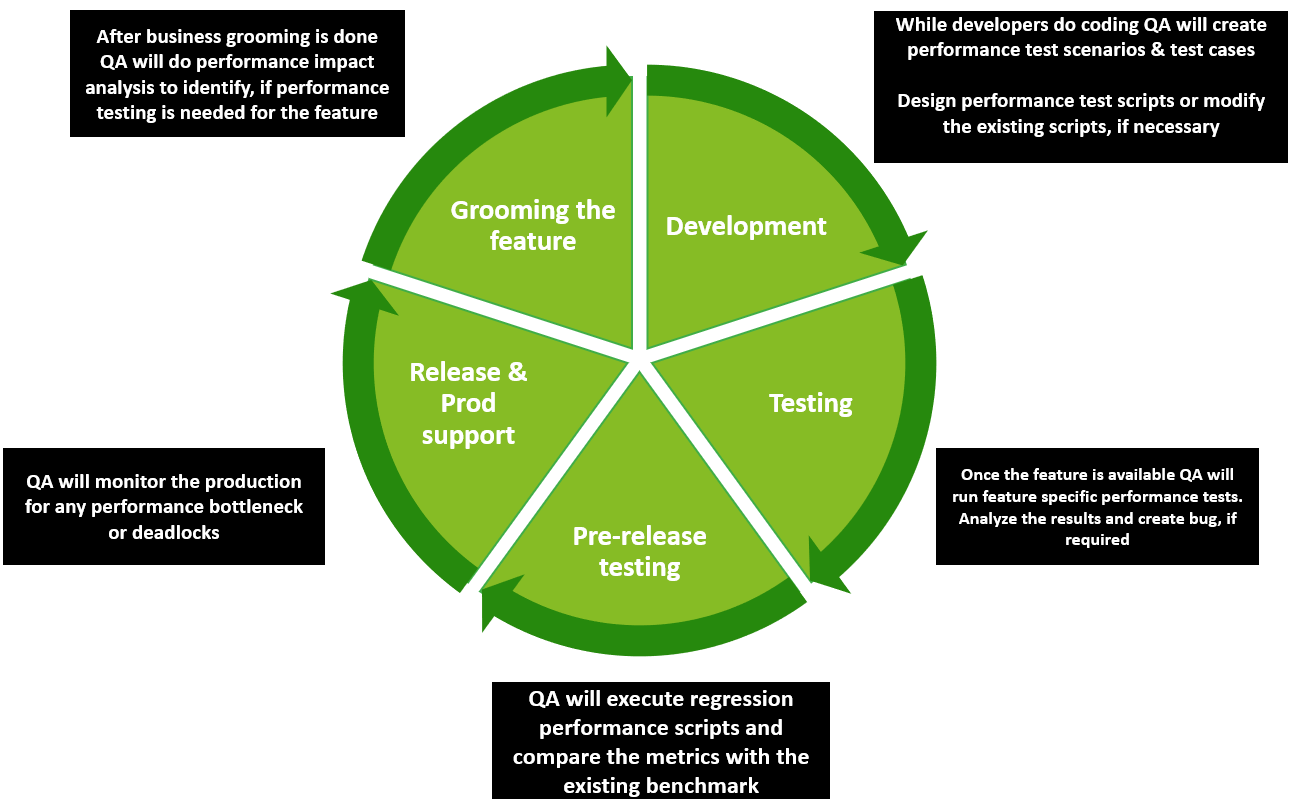
# Why Performance testing?

Performance Testing is a software testing process used for testing the following

* Speed
* Response time
* Stability
* Reliability
* Scalability
* Resource usage

of a software application under workload. The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application before releasing it.

# Performance test during a sprint for a stable project



# Tools used to do Performance testing.

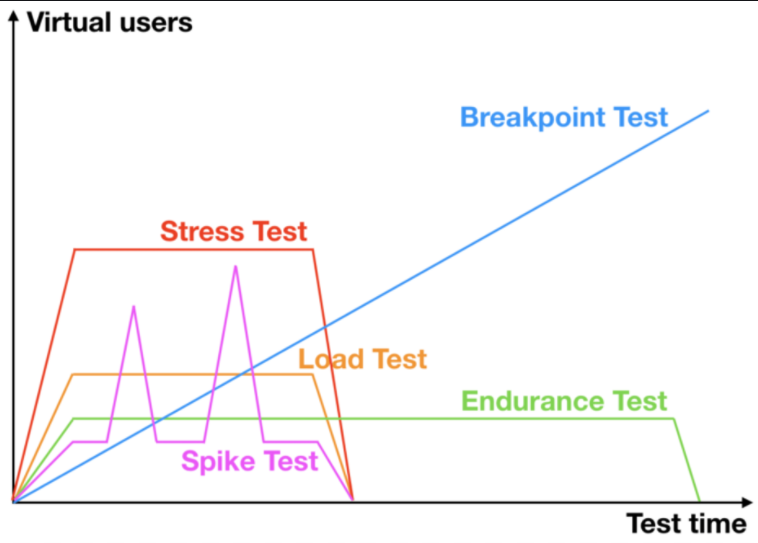
Listed are few performance testing tools, which have been used widely in industry.

|  |  |
| --- | --- |
| **Tool name** | **Advantages** |
| JMeter | 1. Open-source tool, so no subscription charges 2. Has a very good crowd support for troubleshooting 3. Highly portable and support all java-based apps 4. Protocol supported  * HTTP * HTTPS * SAP GUI Web * WebSocket * Java-based protocol * Google Web Toolkit * Oracle forms |
| Load Ninja | 1. Open source 2. Script less load test creation & playback. 3. It is hosted on cloud 4. Protocol supported  * HTTP * HTTPS * XML * SOAP * Java-based protocols * FTP |
| Load runner | 1. Has inbuild reporting functionality & provides detailed metrics 2. Accurate detection of system, end user, and code-level bottlenecks 3. It is paid tool, so has a dedicated client support. 4. It also supports lesser-known protocols like Tibco, SAP, TruClient etc. |

# Types of performance testing

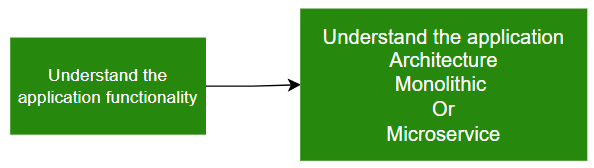


* **Load testing** – It checks the application’s ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.
* **Stress testing** – It involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify the breaking point of an application.
* **Endurance testing** – It is performed to make sure the software can handle the expected load over a long period of time.
* **Spike testing** – It 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 a 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

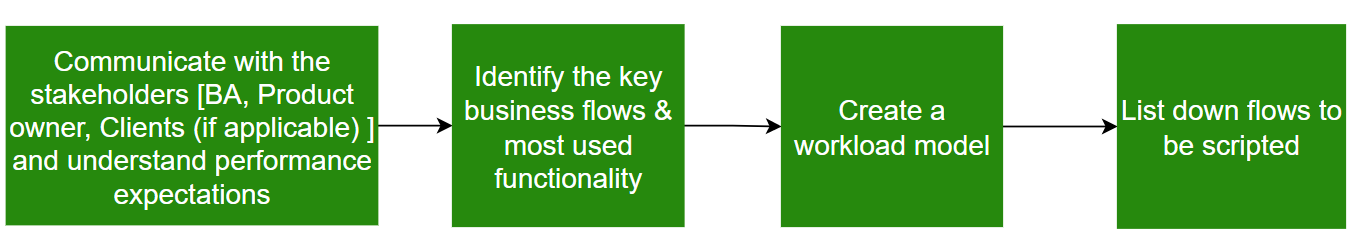


# Process followed while doing performance testing in project

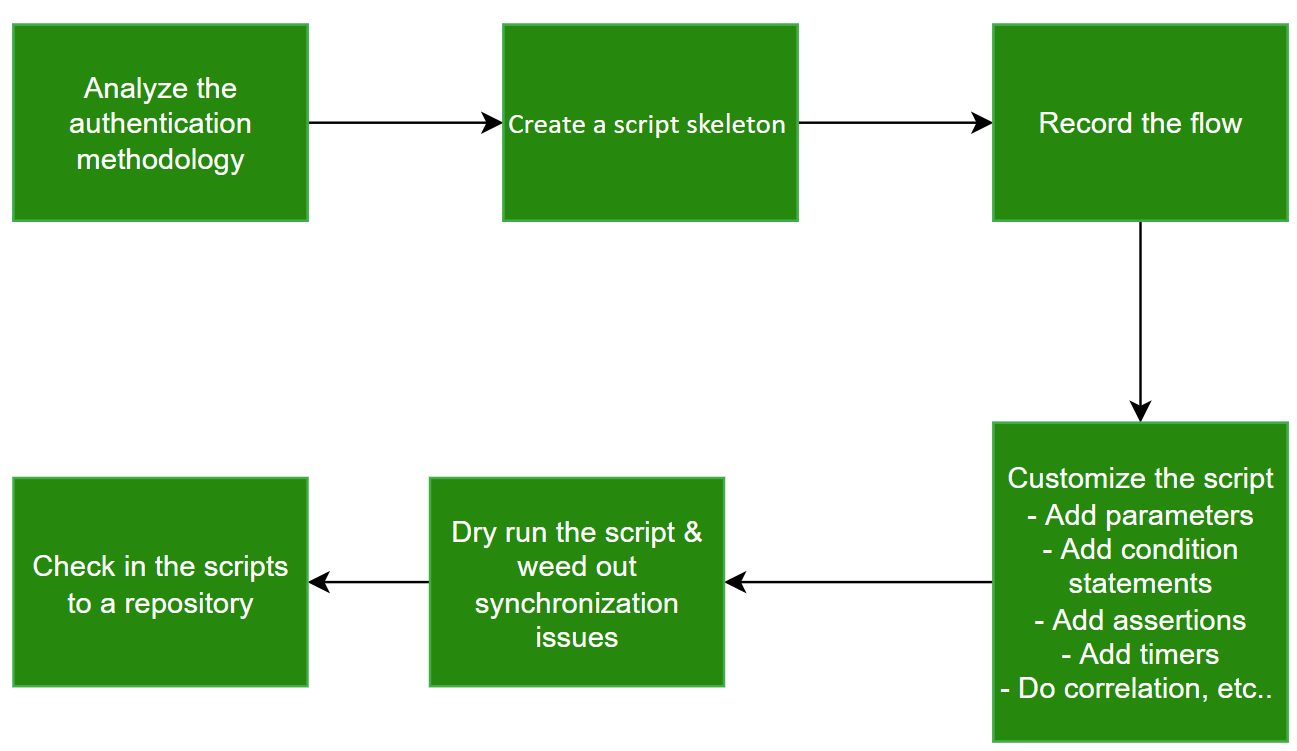
**Step 1**: Get Initial understanding of the application



**Step 2**: Gathering performance requirement



**Step 3**: Designing Script



**Step 4**: Run performance test

One should run multiple rounds of performance tests with different set of configurations specified in the workflow model to get the most accurate results.

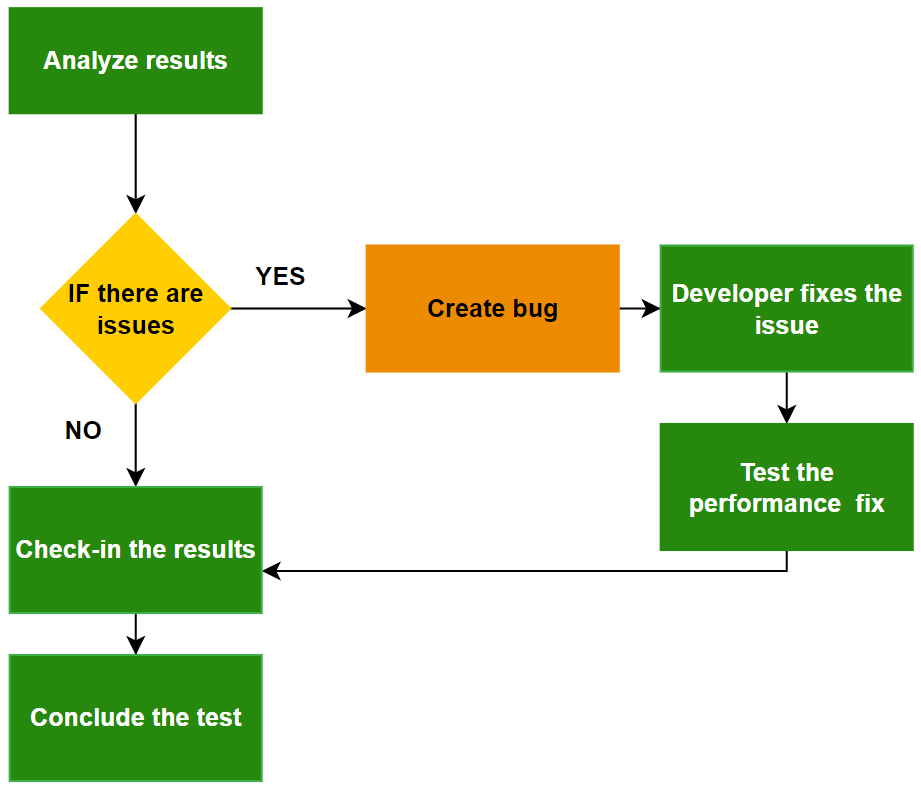
# Analysis of results

There are numerous metrics which can be generated from testing tools. However, following are the most fundamentals ones

1. Metrics from testing tool:
   1. Label **-** Name/URL for the specific Protocol (HTTP(s)) Request
   2. Average response time - Average time taken by all the samples to execute specific label
   3. 90 percentile response time - 90% of the samples took no more than this time. The remaining samples took at least as long as this. (90th percentile)
   4. Latency – Time taken to receive the first byte after request is sent.
   5. Throughput - The amount of bandwidth used during the test.
2. Metrics from other sources:
   1. CPU utilization – Time the CPU needs to process requests
   2. Memory utilization - Memory it needed to process the requests

Apart from above metrics there are lot of other metrics also which can be used depending on test scenario requirements.

# Post Run activities



# Archiving the run data.

It is recommended to save the test results every time we run the valid scenario as it helps to find the overall health of system over a period. As when there are multiple rounds being run, the amount of data created will be huge and managing them can become challenging. One way to tackle this situation is

1. To give unique ID (like simple serial no.’s) to every run
2. For corresponding serial no. write the scenario in excel sheet.

So now if any team member needs to check results of previous run then

1. They can refer to excel sheet for scenario and get unique ID no.
2. Search for the data associated with the unique ID no.