Performance Test Plan

## **Contents:**

1. Purpose
2. Entry and Exit Criteria
3. Test Tool
4. Type of testing
5. Results of testing

## **Purpose**

This testing aims to consolidate the knowledge acquired from the lesson on performance testing using a free resource <https://jsonplaceholder.typicode.com>

This includes practical understanding and the creation of a performance test plan.

## **Entry Criteria**

Java Installation:

* Ensure that Java version 7 or higher is installed.

JMeter Installation:

* JMeter must be installed and properly configured.

Stable Internet Connection:

* A stable internet connection is required for consistent testing and accurate results.

Additional Preparation:

* Review the performance testing lesson again to reinforce understanding and ensure all concepts are clear.

## **Exit Criteria**

The performance testing activity will be considered complete when all homework tasks have been executed and saved on GitHub as a file.

## **Test tool**

Test tools used for Load and Stress testing will be:

JMeter

An open-source load testing tool. Predominantly used for volume and performance testing.

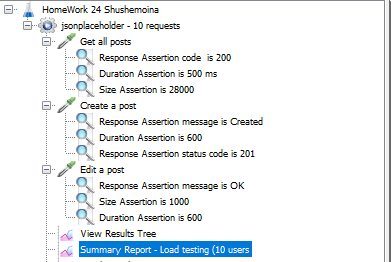
## **Load Testing**

After the baseline metrics are gathered, then the same simulation, which simulates a load profile, is run with an increased number of users to test against the target volumes. The idea of this load test is to test the system against a typical day’s load, simulating the ramp-ups, day’s peaks, and ramp-downs.

## **Stress Testing**

Stress testing aims to find the breaking point of the system, i.e. at what point does the system become unresponsive. If auto-scaling is in place, the stress test will also be a good indicator at which point the system scales and new resources are added. For stress testing, the same simulation used for load testing is used but with a higher than expected load.

## **Results of testing - Load Testing**



### 

### 

### 

### 

### 

### 

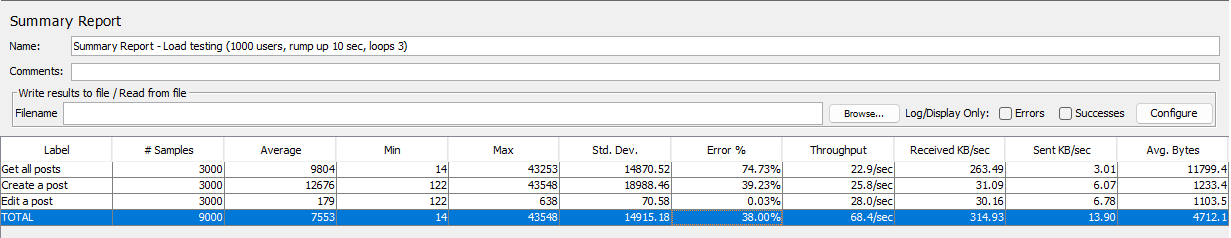
### Overall Performance:

1. The total throughput across all operations is 9.1 requests per second, demonstrating the system's ability to handle concurrent requests efficiently.
2. The average response time (133 ms) is acceptable for web applications.
3. The system maintained a 0% error rate, highlighting its robustness and reliability under load.

## **Conclusion**

The load test results indicate that the system performs well under the tested load conditions. All operations (retrieving, creating, and editing posts) were handled efficiently with low response times and high throughput, without any errors. This suggests that the system is capable of supporting the specified load without performance degradation.

## **Results of testing - Stress testing**



### Overall Performance:

1. The overall throughput for all operations is 68.4 requests per second, indicating moderate capability of the system to handle requests.
2. The average response time (7,553 ms) indicates significant delays in the system.
3. The high error percentage (38.00%) suggests problems with handling requests under high load.

### **Conclusion**

The load testing results show that the system has significant performance issues under heavy load, especially when retrieving and creating posts. The high error percentage and long response times indicate that the system is not handling requests efficiently. Editing posts perform much better but still require optimization. Additional improvements and optimizations are needed to reduce response times and error rates and to enhance the overall performance of the system.