**Upgrade - Performance Coding Challenge**

**Summary**

Included here, is the completed Load Test Project for running the microservice included in the project provided for this challenge..

Points to note

This project is completed using Maven with jmeter dependencies set in pom file. Refer to Performance Score Card for the SLA”s I have defined and the measured values from the tests based on which I have provided my recommendations.

**Performance ScoreCard:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Concurrent Threads** | **Endpoints** | **Target SLA’s** | | **Measured** | | **Target/5mins** | **Status** |
| **Avg** | **99th Percentile** | **Avg** | **99th Percentile** |
| 100 | /user/v1/create | 1 secs | 2secs | 11.01secs | 29.20secs | 1000 | **Fail** |
| /user/v1/find/<valid-uuid> | 1secs | 2secs | 11.10secs | 31.12secs | 1000 | **Fail** |

**Instructions for running this test:**

1. JAVA 11 should be installed and path variables should be set.
2. Extract the project from zip file and Import the project it in IDE
3. Install Docker
4. Go to terminal and run docker-compose up –build
5. Make sure the “perf-code-challenge” container is created in docker wit its apps/images
6. Go to directory and set following inputs to run this load test incase you plan to run with less load “\src\test\jmeter\loadtest.properties  
   THREAD\_COUNT=100  
   RAMP\_TIME=120  
   DURATION=300
7. Open another terminal and run mvn clean verify

**Reports & Logs:**

1. Go to directory “target/jmeter/reports” and open the ‘Index.html’ file for the HTML report of the test.
2. Log file will be available here “target/jmeter/results”.
3. Go to Grafana dashboard to analyse the health of the app. [Perf-test-challenge-dashboard - Grafana](http://localhost:3000/d/3Akj3rk4k/perf-test-challenge-dashboard?orgId=1&refresh=5s)

**Test Observations & Recommendations:**

1. With 100 Concurrent threads there is a big performance degradation with both Avg and 99th Percentile response times way above defined SLA of 2secs
2. All Tests were ran for 5 mins interval. Tests were ran on 08/06 between 2PM to 3PM and 10PM to 12AM.
3. Did Tests with reduced load to assess the performance. Below is the snapshot of performance under varying load conditions as given below

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Endpoint** | **# Successful Requests** | | | | | **Avg Response Time(ms)** | | | | |
| **50 Threads** | **75 Threads** | **100 Threads** | **150 Threads** | **200 Threads** | **50 Threads** | **75 Threads** | **100 Threads** | **150 Threads** | **200 Threads** |
| **Create User** | 909 | 965 | 1021 | 1124 | 1014 | 5544 | 8294 | 11097 | 15664 | 23958 |
| **Retrieve User** | 887 | 926 | 982 | 1048 | 859 | 6143 | 9168 | 11993 | 16745 | 26400 |
| Throughput | 1796 | 1891 | 2003 | 2172 | 1873 |  |  |  |  |  |

1. Throughput is decreasing over load and due to following reasons
   * All the apps are being hosted on same docker with 1gb memory . 938MB of 1GB is being used during the test.
   * CPU Count is 1 and adding more CPU will help boost performance.
   * Max 10 JDBC connections are allowed and we reached the max during all our tests.
   * Request did not timeout even after 100secs in the queue. Having a timeout set will help improve user experience by atleast showing a meaning message
   * Response Time trend shows throughput drop due to spikes and it could be a possibility of request queuing due to connection reaching threshold limit.
   * **Chart, line chart

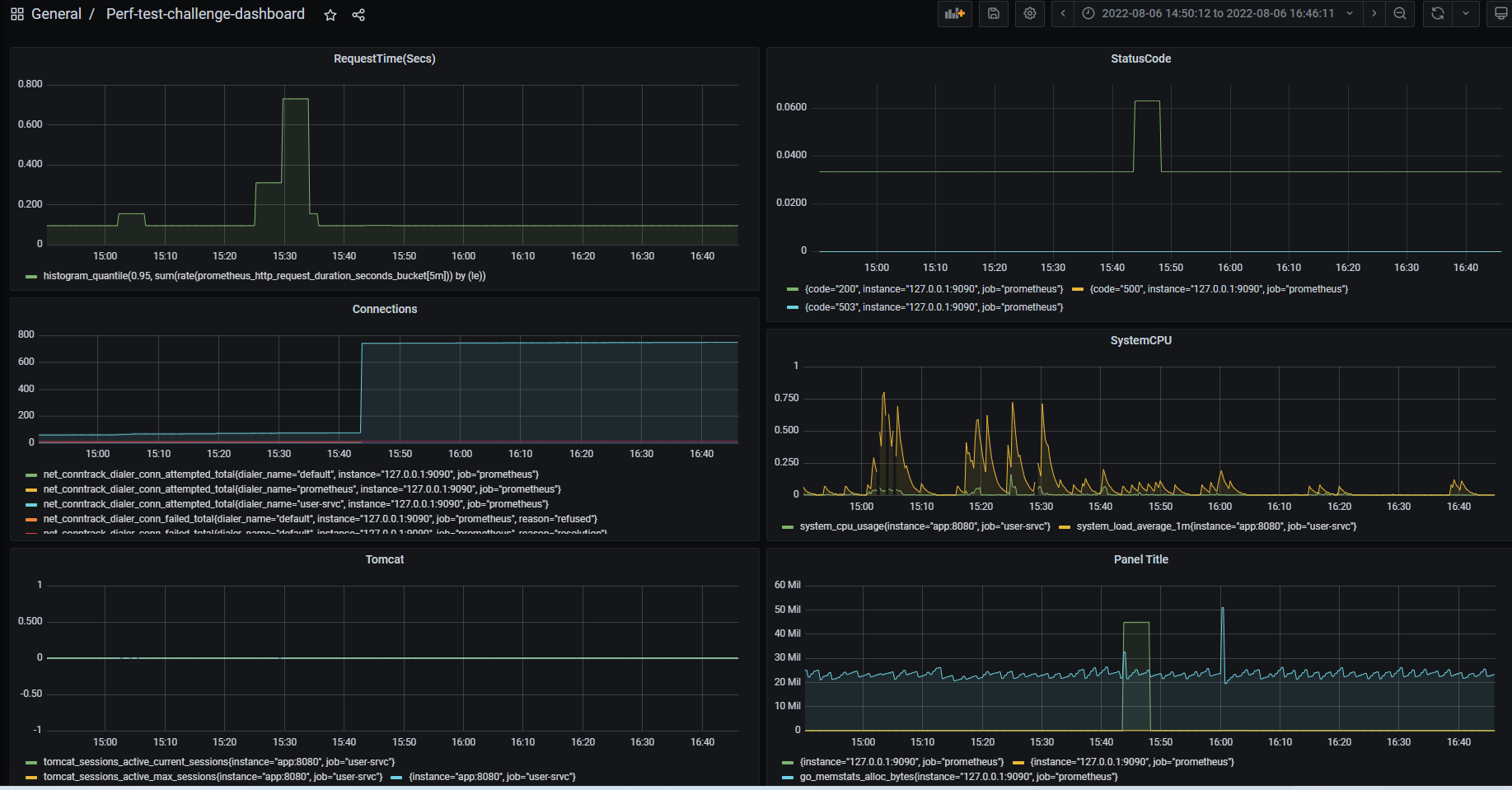
     Description automatically generated**

**Chart

Description automatically generated**

* + Thread state reached max as shown in panel below **Chart

    Description automatically generated**



Postgress Data Source could not be added due to this error. This made me skip DB monitoring.  
Graphical user interface, text, application

Description automatically generated