Naresh Nakka

Peformance Engineer

<<Project name>>

Performance Test Plan V 0.1

Contents

[Project Summary 2](#_Toc19088543)

[In-Scope 2](#_Toc19088544)

[Out-Scope 2](#_Toc19088545)

[SLAs 3](#_Toc19088546)

[Business Flows 3](#_Toc19088547)

[Workload Model 3](#_Toc19088548)

[**Load Distribution:** 3](#_Toc19088549)

[**Throughput:** TPH: Transactions Per Hour 3](#_Toc19088550)

[Test Scenarios 4](#_Toc19088551)

[**Capacity Testing:** 4](#_Toc19088552)

[Test To be Performed: 4](#_Toc19088553)

[ Load Test: 4](#_Toc19088554)

[ Capacity Test: 4](#_Toc19088555)

[ Stress Test: 4](#_Toc19088556)

[ Aging Test: 4](#_Toc19088557)

[Monitoring Tools 4](#_Toc19088558)

[Test Deliverables 5](#_Toc19088559)

[Test Tools & Environment 5](#_Toc19088560)

[Risks & Assumption 5](#_Toc19088561)

# Project Summary

The purpose of this document is to outline the performance test plan for <<application name>> application and for their workflows provided below

|  |
| --- |
| Business flow |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| API Name |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

# In-Scope

* Identify the response times for the provided API as per the Workload model.
* Establish baseline for the performance to measure any enhancements made to API to improve the performance.
* Monitoring the Server instances (CPU, Memory & Network, Application Specific metrics if provided monitoring tool) – optional

# Out-Scope

* Performance testing effort outlined in this document will not cover the following
  + Any functional or accuracy testing the software
  + Compatibility testing
  + SLA breach due to external issues

# SLAs

* SLAs (response time) of the all each page and API should not exceed as mentioned below

|  |  |
| --- | --- |
| Content | SLA in Seconds |
| Pages of Functionality – 1 | 4 seconds |
| Pages of Functionality – 2 | 5 Seconds |
| Pages of Reports | 10 Seconds |
|  |  |
|  |  |
|  |  |

# Business Flows

* Below list of Key Business scenarios are identified and provided by the business, will be tested during the performance testing process.

|  |  |
| --- | --- |
| Serial Number | Scenario |
| 1 | Scenario Name 1(e.g., Login) |
| 2 | Scenario Name 2(e.g., Search) |
| 3 | TBD |
| 4 | TBD |
| 5 | TBD |
| 6 | TBD |

# Workload Model

### **Load Distribution:**

|  |  |  |
| --- | --- | --- |
| Scenario/Functionality | Load Distribution Percentage | Example |
| Scenario - 1 | 20% | 20 Users |
| Scenario - 2 | 50% | 50 Users |
| Scenario - 3 | 30% | 30 Users |
| Total | **100%** | **100** |

### **Throughput:** TPH: Transactions Per Hour

|  |  |
| --- | --- |
| Scenario/Functionality | Throughput(TPH) |
| Scenario - 1 | 30 |
| Scenario - 2 | 120 |
| Scenario - 3 | 500 |
| Total | **650** |

# Test Scenarios

### **Capacity Testing:**

To determine the application or server capacity increasing the number of concurrent users in stair-step approach i.e. 1X, 2X, 3X etc. to fail either of below conditions. (X is a minimum number of users to start)

* SLA breach point
* Increased number of Errors
* Server Crash point

# Test To be Performed:

## Load Test:

A load is conducted for a specific number of users with a targeted throughput to be achieve. Here we keep Load and TPH as constant metrics to verify the application performance. An SLA will be applied against each and every page/transaction. This test usually runs for an hour duration.

**Goal**: To identify the application performance for a specific load

## Capacity Test:

Capacity test is conducted, when there is uncertainty to get the required number of peak loads to be applied on the application. We start with a small amount of load and increase it until a certain load where the application SLA breaches or Server resources occupied up to 80%.

**Goal**: To identify the application capacity in term of concurrent users without SLA breach or with server resource limits

## Stress Test:

This test is conducted to see the application maximum capacity in terms concurrent users, before it crashes.

**Goal**: To identify the maximum concurrent user application supports before it crashes.

## Aging Test:

To identify the application stability over a period of time (equals to number of business hours).

Goal: To make sure application does not crash and performance is consistent across the test window.

# Monitoring Tools

* Access to be provided/identified by the application owner

# Test Deliverables

* A detailed test (Word/PDF) report/combined test report will be provided for each test contains below information
  + Response times
    - Number of requests
    - Avg Response times
    - Median
    - 90 Percentile
    - 95 Percentiles
    - Error Percentage
  + Hits/Sec graph to determine the hits received by the app
  + Concurrent users show number of users load tested.
  + Analysis or observations done during the test.
  + Server health metrics i.e., CPU, Memory & Network (If provided access)

# Test Tools & Environment

* JMeter Performance testing tool (an open source tool) will be used as performance testing tool using the distributed testing architecture
* Application owner should provide required Load Generation Instance in AWS cloud as per below configuration, depends on the expected load.
  + A Linux server (Ubuntu) with 2 CPU and 8 GB memory can support up to 400 to 500 Concurrent Users.
  + Concurrent Capacity may change depends page size of application.
* **Prerequisites**: JDK 9 or above

# Risks & Assumption

Performance test will be conducted considering the below assumptions and risks

* Performance testing done on dedicated environment and properly scaled as per the productions size
* No other load coming to application from different sources like dev team, pilot users, functional testers
* Environment downtime impacts the test deliverables.
* Application is functionally stable and no deployments happening during the test.

Approval & Resources

|  |  |  |
| --- | --- | --- |
| Resource Name | Role | Approval Needed |
| John Theophanopoulos |  | Yes |
| Khalifa Alkuwari |  | Yes |