

|  |
| --- |
| PERFORMANCE TEST PLAN |
| RightFind Enterprise Performance Testing |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 0.1 | Initial Version | Denys Hurt [dhurt@copyright.com](http://dhurt@copyright.com) | 8-July-2021 |  |  |

**CONTENTS**

[1 INTRODUCTION 4](#_Toc76665331)

[1.1DEFINITIONS, ACRONYMS, ABBREVIATIONS 4](#_Toc76665332)

[2 PERFORMANCE REQUIREMENTS 4](#_Toc76665333)

[3 RISKS 5](#_Toc76665334)

[4 TIMELINES 5](#_Toc76665335)

[5 LOAD MODEL 6](#_Toc76665336)

[5.1USER FLOWS 6](#_Toc76665337)

[6 TEST TYPES 8](#_Toc76665338)

[6.1SERVER-SIDE TESTS 8](#_Toc76665339)

[6.1.1 BASELINE 8](#_Toc76665340)

[6.1.2 CAPACITY (RAMP UP) 8](#_Toc76665341)

[6.1.3 FIXED LOAD (BENCHMARK) 9](#_Toc76665342)

# 1 INTRODUCTION

This document details the NFR’s, load model and the user flow for the RightFind Enterprise application. This will serve as an input for creating the required performance scripts.

## DEFINITIONS, ACRONYMS, ABBREVIATIONS

|  |  |
| --- | --- |
| Acronym | Definition |
| vUsers | Virtual Users |

# PERFORMANCE REQUIREMENTS

**Main purpose of the application:**

RightFind® Enterprise provides faster discovery and insights with immediate access to scientific literature and data and strengthens copyright-compliant collaboration

**Testing objectives:**

1. TEST environment, <https://www.test.rightfind.com/>:
   1. App Version: RightFind v8.4.1.016(AWS TestG1)
   2. Goal: Measure overall application performance. Ensure that there is no performance degradation.
2. Response time SLA: TBD
3. Throughput SLA: TBD
4. Error rate SLA: TBD
5. Test users: user, admin

**System under test architecture (diagram):**

<https://com-copyright-rms-pdev.s3.amazonaws.com/moleshchuk/diagram/gravy.html>

**Monitoring:**

1. TBD

**Jenkins integration:** <https://qatestcontrol.copyright.com/>

# 

# 3 RISKS

|  |  |  |
| --- | --- | --- |
| No | Risk | Impact |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |

# 4 TIMELINES

Table 4.1.1 Timelines

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Start Date | End Date | Dependencies |
| Preliminary Test Plan | 7/8/2021 |  |  |
| Test Scripting & Validation |  |  |  |
| Jenkins Integration |  |  | Infrastructure |
| Test Execution Baseline |  |  | Working scripts |
| Test Execution Capacity |  |  |  |
| Test Execution Fixed Load |  |  |  |
| Report and Analysis |  |  | End of Test Execution |
| Test Completion |  |  | Analysis & Reporting |

# 5 LOAD MODEL

The load model is an instrument simulating use of application in real-world. It is an integrated part of the test execution process.

Peak number of concurrent users: 400-500

Average number of concurrent users: 150-200

Time applicable time to run the tests:

1. Baseline test: Any time
2. Capacity test: TBD

## USER FLOWS

A list of all the scenarios that would be used in the system was identified. Please find workflow diagram in figure 5.1.2

Please find a list of scenarios to be covered below:

Table 5.1.1: Performance scenarios

|  |  |
| --- | --- |
| Scenarios | Probability |
| Search | |
| Regular Search (no Semantic) | 100% |
| Regular Search + Full Text | 100% |
| Semantic Search + Barchart Aggregation | 100% |
| A-Z Catalog | 100% |
| Opening libraries with big number of documents | |
| With Semantic capabilities | 100% |
| Without Semantic capabilities | 100% |
| Search in user libraries | |
| With Semantic capabilities | 100% |
| Without Semantic capabilities | 100% |
| Other | |
| Multi-user work | 100% |

Diagram

Description automatically generated

Figure 5.1.2 Workflow diagram: Regular Search (no Semantic)

# TEST TYPES

## SERVER-SIDE TESTS

## BASELINE

**Purpose**:

To check that system is ready for performance testing and scripts are OK

**Priority**: Highest

Duration: 1h

Ramp-up: 1 user every 10 secs

Think Time: from 10 sec to 15 sec

vUsers: up to 10

The pipe-clean test is a preparatory task that serves to validate each performance test script in the performance test environment. The test is normally executed for a single use case as a single virtual user for a set period or for a set number of iterations. This execution should ideally be carried out without any other activity on the system to provide a best-case measurement. You can then use the metrics obtained as a baseline to determine the amount of performance degradation that occurs in response to increasing numbers of users and to determine the server and network footprint for each scripted use case.

### CAPACITY (RAMP UP)

**Purpose**:

To determine how many users and/or transactions a given system will support and still meet performance goals, to find out server capacity, stability under incremental load and scalability of the system. Also, capacity testing results are key points to create performance tests of other types, for example fixed load or stress.

**Priority**: Highest

Duration: 1h

Ramp-up: 1 user every 10 secs

Think Time: from 10 sec to 15 sec

vUsers: up to 500

System capacity - the highest level of load it can take and handle without:

* Significant response times increase
* Stability decreases

A capacity test complements load testing by determining server’s ultimate failure point. You perform capacity testing in conjunction with capacity planning, which you use to plan for future growth, such as an increased user base or increased volume of data. For example, to accommodate future loads, you need to know how many additional resources (such as processor capacity, memory usage, disk capacity, or network bandwidth) are necessary to support future usage levels. Capacity testing helps to identify a scaling strategy to determine whether system should scale up or scale down.

### FIXED LOAD (BENCHMARK)

**Purpose**:

To get response times statistics under different levels of load and compare them against target/previous release (build, sprint, etc.) results

**Priority**: High

Duration: 1 hour

Ramp-up: 1 user every 10 secs

Think Time: from 10 sec to 15 sec

vUsers: 150-200

Fixed load testing is conducted to verify that your application can meet your desired performance objectives; these performance objectives are often specified in a service level agreement (SLA).

A load test enables you to measure response times, throughput rates, and resource-utilization levels, to verify application behavior under normal and peak load conditions.