Load and Performance

Test Plan

|  |  |
| --- | --- |
| Key | Value |
| Application Name | <ACME-Management> |
| Application Version | v.6.0.0.6 - Current Version, used for Baseline Test  v.7.0.0.7 - Version to be performance tested |
| Scope | Measure response times and stability |
| Test Lead | R.Scheiwiller |
| Tickets | <https://your-jira.your-company.com/browse/LUP-8008>  <https://your-jira.your-company.com/browse/LUP-9009> |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Author | Date | Version | Comment |
| R.Scheiwiller | 01.01.1970 | v42.5 | Initial Version |
|  |  |  |  |

Document Review

|  |  |  |
| --- | --- | --- |
| Reviewer | Date Reviewed | Comment |
| Jane Doe | 01.01.1970 | Initial Version |
|  |  |  |

**Table of Contents**

[1 Introduction 3](#_Toc18854604)

[2 Systems under Performance Test 3](#_Toc18854605)

[2.1 General Information 3](#_Toc18854606)

[2.2 Application Architecture 4](#_Toc18854607)

[2.3 System Achitecture 4](#_Toc18854608)

[3 Requirements & Design 5](#_Toc18854609)

[3.1 Testing Scope 5](#_Toc18854610)

[3.1.1 In Scope 5](#_Toc18854611)

[3.1.2 Out of Scope 5](#_Toc18854612)

[3.2 Testing Targets 5](#_Toc18854613)

[3.3 Test Environment 6](#_Toc18854614)

[3.3.1 Hardware Information 6](#_Toc18854615)

[3.4 Use Cases 7](#_Toc18854616)

[3.5 Test Scenarios 7](#_Toc18854617)

[3.5.1 Average Load 7](#_Toc18854618)

[3.5.2 Peak Load 8](#_Toc18854619)

[3.5.3 Baseline Test 9](#_Toc18854620)

[3.5.4 Stability Test 9](#_Toc18854621)

[3.5.5 Stepped Scalability Test 10](#_Toc18854622)

[3.5.6 Page Request Analysis 10](#_Toc18854623)

[3.6 Data 11](#_Toc18854624)

[3.6.1 Test Database Setup 11](#_Toc18854625)

[3.6.2 Test Users 11](#_Toc18854626)

[3.6.3 Test Data 11](#_Toc18854627)

[4 Used Tools 12](#_Toc18854628)

[5 Roles & Contacts 13](#_Toc18854629)

[6 Responsibilities 14](#_Toc18854630)

[7 Risks 15](#_Toc18854631)

[8 Appendix 16](#_Toc18854632)

[8.1 Description of Load Test Scenarios 16](#_Toc18854633)

# Introduction

This document assembles all the important information needed to carry out performance testing. It is used as the main source of information for the performance engineers, as well as to inform all the involved parties about the used performance testing approach.

The goal of the test plan is to get to a mutual agreement between all the parties regarding the load testing approach, before the actual testing takes place. Therefore the document should be reviewed at least by the project lead.

# Systems under Performance Test

## General Information

|  |  |
| --- | --- |
| Key | Value |
| Application Purpose | <General description of what the application is used for.> |
| Application URL | <URL> |
| Technology Stack | <Java>  <Springboot>  <Hibernate>  <…> |
| Code Repository | <Link to Repo or N/A > |
| Known Issues | <Memory Leaks>  <Known slow use cases>  <Application outages>  <…> |

## Application Architecture

<Add Diagram or link to architecture document>

**Example Application Architecture**

Frontend Layer

Web Service API's

3rd Party Services

Backend Services

Backend Layer

APIs

Database Access

Internet

## System Achitecture

<Add Diagram or link to architecture document>

**Example System Architecture**

Frontend Server

App Instance A

App Instance B

Load Balancer

3rd Party Services

Backend Server

Instance X

Instance Z

DB

# Requirements & Design

## Testing Scope

### In Scope

The following components of the system are in scope of the testing and of the analysis:

|  |  |  |
| --- | --- | --- |
|  | Component | Comment |
|  | Frontend Layer | Application “ACME” |
|  | Backend Layer | Application “Foo” and “Bar” |
|  | Database |  |
|  | Calls to Service Providers |  |
|  | 3rd Party Services | Calls to “www.giveresponse.com/api/” |
|  | Other |  |

### Out of Scope

Out of scope are all other applications and components not mentioned above.

## Testing Targets

The following targets are in scope of the load and performance testing

|  |  |  |  |
| --- | --- | --- | --- |
|  | Metric | Target(s) | Comment |
|  | Response Time | 90 percentile <= 3 sec | For all transactions except reporting functionality |
|  | Subsystem Response Time | 90 percentile <= 1 sec | Target for System <A> and <B> |
|  | CPU Usage | Average <= 75% |  |
|  | Server Memory Usage | Average <= 75% |  |
|  | Errors | Rate <= 5% | Error Rate below 5% for the entire test |
|  | Stability | 8 hours | Application is stable for 8 hours of average load |

## Test Environment

The environment used for performance testing must have the same configuration as the productive environment regarding hardware, system and application settings. Without the same configuration (e.g. different memory or CPU capacity) the performance test results cannot be used to make a valid assumption about the application behavior in production.

### Hardware Information

The frontend and backend are running on the following servers with CentOS as the operating system:

|  |  |
| --- | --- |
| Processor | |
| Processor type | Intel Xeon Gold 5118 |
| Number of processors | 2 x |
| Processor clock frequency | 2.30 GHz |
| Number of threads | 24 |
| Number of processor cores*i* | 12 |
| Number of processors socket | 2 x |
| Max. turbo clock rate | 3.20 GHz |
| Max. TDP*i* | 105 W |
| L3 Cache*i* | 16.50 MB |
| Processor family | Xeon |

|  |  |
| --- | --- |
| Memory | |
| Memory type | DDR4-RAM |
| Memory Slots max | 24 |
| Storage capacity 1 (RAM) | 16 GB |
| RAM total | 32 GB |
| Max. RAM | 3000 GB |
| RAM speed | 2666 MHz |

|  |  |
| --- | --- |
| Network | |
| Ethernet | 6 x |
| RJ45/LAN port*i* | 1 x |
| Max. port speed*i* | 25000 Mbit/s |

## Use Cases

The list of use cases can be found in the use case documentation:

* <https://www.your-company.com/files/docs/1970-01-01_ACME_UseCaseDescription.docx>

## Test Scenarios

Test scenarios describe the load that will be produced during a test. A load test scenario consists of one or multiple use cases, number of users and the load that each of the use cases should produce.

The target is to create scenarios which will test against the requirements of the test. You can find a description of common load test scenarios in the section "8.1 Description of Load Test Scenarios".

Explanation of terms in this chapter:

* **Users:** The number of users simulated and executing this script. (Might be done with less user accounts)
* **Exec/h:** The number of script executions per hour(total executions of all users).
* **Start Offset:** The offset of the script start from the actual test start.
* **Ramp Up Users:** Number of users started in parallel on each ramp up interval.

### Average Load

**Scenario Details**

|  |  |
| --- | --- |
| Key | Value |
| Description | Average business day simulation with common use cases. |
| Test Duration | 1 hour |
| Total Users | 190 Users |
| Total Executions | 2500 Executions/h |

**Load Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use Case | Users | Exec/h | Start Offset | RampUp Users |
|  | ACME\_01\_BrowseProducts | 100 | 1500 | 00:00:00 | 2 |
|  | ACME\_02\_ManageCart | 20 | 300 | 00:00:00 | 2 |
|  | ACME\_03\_ApplyGiftCode | 5 | 100 | 00:00:00 | 2 |
|  | ACME\_04\_Checkout | 70 | 700 | 00:10:00 | 1 |
|  | ACME\_05\_PrintReport | 15 | 300 | 00:00:00 | 1 |

### Peak Load

**Scenario Details**

|  |  |
| --- | --- |
| Key | Value |
| Description | Peak business day simulation(3 times average load) with common and uncommon use cases. |
| Test Duration | 1 hour |
| Total Users | 630 Users |
| Total Executions | 8700 Executions/h |

**Load Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use Case | Users | Exec/h | Start Offset | RampUp Users |
|  | ACME\_01\_BrowseProducts | 300 | 4500 | 00:00:00 | 2 |
|  | ACME\_02\_ManageCart | 60 | 900 | 00:00:00 | 2 |
|  | ACME\_03\_ApplyGiftCode | 15 | 300 | 00:00:00 | 2 |
|  | ACME\_04\_Checkout | 210 | 2100 | 00:10:00 | 1 |
|  | ACME\_05\_PrintReport | 45 | 900 | 00:00:00 | 1 |

### Baseline Test

**Scenario Details**

|  |  |
| --- | --- |
| Key | Value |
| Description | Minimal load including all use cases to check maximum reachable performance. |
| Test Duration | 20 minutes |
| Total Users | 5 Users |
| Total Executions | 50 Executions/h |

**Load Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use Case | Users | Exec/h | Start Offset | RampUp Users |
|  | ACME\_01\_BrowseProducts | 1 | 10 | 00:00:00 | 2 |
|  | ACME\_02\_ManageCart | 1 | 10 | 00:00:00 | 2 |
|  | ACME\_03\_ApplyGiftCode | 1 | 10 | 00:00:00 | 2 |
|  | ACME\_04\_Checkout | 1 | 10 | 00:10:00 | 1 |
|  | ACME\_05\_PrintReport | 1 | 10 | 00:00:00 | 1 |

### Stability Test

**Scenario Details**

|  |  |
| --- | --- |
| Key | Value |
| Description | All use cases with average load running for a full business day. |
| Test Duration | 8 hours |
| Total Users | 210 Users |
| Total Executions | 2900 Executions/h |

**Load Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use Case | Users | Exec/h | Start Offset | RampUp Users |
|  | ACME\_01\_BrowseProducts | 100 | 1500 | 00:00:00 | 2 |
|  | ACME\_02\_ManageCart | 20 | 300 | 00:00:00 | 2 |
|  | ACME\_03\_ApplyGiftCode | 5 | 100 | 00:00:00 | 2 |
|  | ACME\_04\_Checkout | 70 | 700 | 00:10:00 | 1 |
|  | ACME\_05\_PrintReport | 15 | 300 | 00:00:00 | 1 |

### Stepped Scalability Test

**Scenario Details**

|  |  |
| --- | --- |
| Key | Value |
| Description | Increase the load every hour by the average load. |
| Test Duration | 1 hour for every load level |
| Total Users | N/A |
| Total Executions | N/A |

**Load Configuration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use Case | Users | Exec/h | Start Offset | RampUp Users |
|  | ACME\_01\_BrowseProducts | 100 | 1500 | 00:00:00 | 2 |
|  | ACME\_02\_ManageCart | 20 | 300 | 00:00:00 | 2 |
|  | ACME\_03\_ApplyGiftCode | 5 | 100 | 00:00:00 | 2 |
|  | ACME\_04\_Checkout | 70 | 700 | 00:10:00 | 1 |
|  | ACME\_05\_PrintReport | 15 | 300 | 00:00:00 | 1 |

### Page Request Analysis

This test scenario is used for analysis of single page requests, with the intention of finding potential performance improvements on the page using tools like Google Insights or Page Analyzer.

The following use cases should be analyzed:

* Loading a Product page with 100 products: <https://acme-test.your-company.com/app/productlist>
* Open the dashboard: <https://acme-test.your-company.com/app/dashboard>

## Data

### Test Database Setup

The amount and complexity of the data is a main factor on how fast it will be processed by your application. Therefore the quality of test data is crucial for simulating productive like load. The database should contain data in the same volume and complexity as in the productive environment. For an existing application, the best way is to clone the database from the productive environment to the environment used for testing.

The following approach will be used:

Clone database from existing production system

Existing, artificial data in system under test

Synthetic data generated for performance tests

Other:

### Test Users

The amount of data a user can access highly depends on the rights he has in the application. Therefore the test users need similar application rights as the productive users to execute valid performance tests. The access rights have to be provided by the project team.

The following users will be used for executing the tests:

* <Add list of users or link>

### Test Data

The following test data will be delivered by the project and used for testing:

|  |  |  |
| --- | --- | --- |
| Data | Amount | Comment |
| SearchTerms | 30 | Various search terms to filter the product list |
| GiftCodes | 1000 | Gift codes used for the use case ACME\_03\_ApplyGiftCode |
| CreditCard | 5 | Test credit card values for checkout. |

# Used Tools

The following tools will be used for performance testing, monitoring and analysis:

|  |  |
| --- | --- |
| Name | Purpose |
| Silk Performer | Used for creating the test scripts and running the load tests. |
| Dynatrace | Monitoring tool to collect various metrics and details about application behavior during the load test. |
| Splunk | Log Management tool used for analyzing the log files of the application. |
| Page Analyzer | Open Source web application used to analyze single page requests using .har-Files. <https://github.com/xresch/PENG_Toolbox/tree/master/Java/PageAnalyzer_Workspace> |
| CSV Comparator | Tool used to compare result files in .csv-Format for faster analysis: <https://sourceforge.net/projects/csvcomparator/> |
| JIRA | Performance issues will be tracked with tickets in the JIRA. |

# Roles & Contacts

The following table contains the people working on the performance testing team and are the primary contacts in the project. The person holding a role is responsible for the deliverables assigned to the specific role in the section "6 Responsibilities".

|  |  |  |
| --- | --- | --- |
| Role | Name | Contact Info |
| Application Owner | Jane Doe | <email> <phone> <comments> |
| Project Lead |  |  |
| Test Manager |  |  |
| Lead Performance Engineer |  |  |
| Performance Engineer |  |  |
| Architect |  |  |
| Developer |  |  |
| […] |  |  |

# Responsibilities

In the following table you find the deliverables over the performance testing cycle and which role is responsible for each deliverable. The responsible person can as well delegate the task to some other member of his team.

|  |  |  |
| --- | --- | --- |
| Deliverable | Responsible | Task |
| Questionnaire | Project Lead | The Project Lead receives the questionnaire from the Performance Engineer and is responsible to collect as many information asked by the question as possible. Sends the filled out questionnaire back to the Performance Engineer. |
| Test Plan | Performance Engineer | Collects all the information required for the test plan from the project members. |
| Test Plan Review | Project Lead | Review the whole document and make a decision if the performance testing can be executed as described. |
| Effort Estimation | Lead Performance Engineer | Estimate the effort needed for executing the performance testing as stated in the test plan. |
| Test Environment | Test Manager | Provide access to the environment used for testing and give updates on the availability of the environment. |
| Test Slots | Test Manager | Coordinate with the Performance Engineer the test slots that can be used for executing performance tests. |
| Test Data | Test Manager | Delivers the test data in an appropriate format to the Performance Engineer. |
| Test Users | Test Manager | Delivers the test users with login credentials to the Performance Engineer. |
| Application Rights | Test Manager | Organizes the setup of the application rights for the test users. |
| Use Case Description | Test Manager | Delivers the use case description to the Performance Engineer. |
| Test Scripts | Performance Engineer | Create the test scripts according to the use case description. |
| Test Setup | Performance Engineer | Setup the test scenarios and the testing infrastructure needed for the test execution. |
| Test Execution | Performance Engineer | Execute and analyze the test. |
| Tickets for Issues | Performance Engineer | Create tickets for the found performance issues, including tuning recommendation if possible. |
| Test Report | Performance Engineer | Write the test report and delivers it to the project team. |

# Risks

The following is a list of potential risks that can impact carrying out the performance testing described in this test plan. The criticality of the occurrence of a situation is for most projects the same, while the probability can vary a lot depending on the system under test.

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Description / Impact | Probability | Criticality |
| Environment for testing not ready | Performance tests have to be postponed, reduced time for test execution, scripting might be done in other environment. | HIGH | MEDIUM |
| Missing Access Rights | Scripting cannot be started and tests not executed. | MEDIUM | MEDIUM |
| Data Volume inadequate | Load tests will not deliver results that are close to production. | MEDIUM | HIGH |
| Wrong Hardware Sizing | The hardware used in the test environment is not sized as in production (same amount of CPU, Memory etc…), test results will be invalid. | LOW | VERY HIGH |
| Subsystems unavailable | Unavailable subsystems used by the application can create higher or lower measured response times, depending on if the application is waiting until a timeout occurs or gets immediately interrupted by an exception. | LOW | MEDIUM |
| High Error Rate | Can decrease the validity of the test results tremendously and render them useless. | LOW | HIGH |
| Frequent Changes | Frequent changes on the application UI or interfaces can break the script and cause additional fixing effort. | LOW | MEDIUM |
| Complex Workflows | Increase the effort needed for scripting and decrease the probability of successful execution of the scripts. | LOW | MEDIUM |
| Complex Automation | The technology stack is hard to automate with the chosen tools. Scripting approach might have to be changed and effort increases. | LOW | MEDIUM |
| Available Test Slots | The number of available test slots and the time the environment is available for load testing, impacts how many tests and retests can be executed. | LOW | MEDIUM |

# Appendix

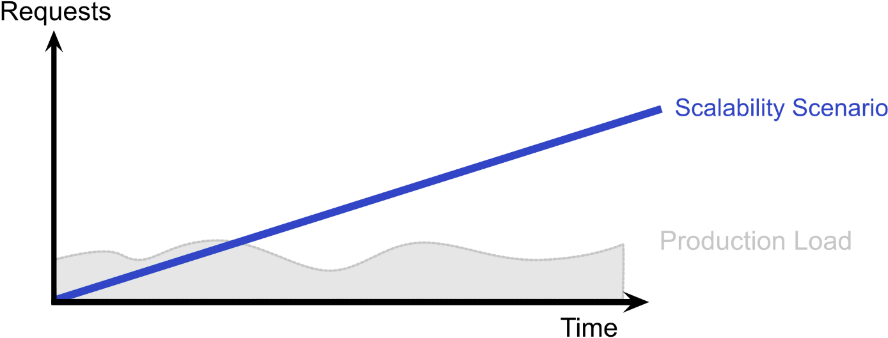
## Description of Load Test Scenarios

In this section the most common load test scenarios are described.

**Baseline Scenario:**  A baseline scenario executes all scripts used in the performance tests with a very low load to determine the best result the application can produce. A baseline can be used if there is no possibility to do a comparison of two tests before and after application changes.

**Average Scenario:** Simulates the average load that is expected to occur in a productive environment. It's the daily business load users will create regularly on your application.

**Peak Scenario:** Simulates a peak situation with higher load than in the average scenario. Such cases can include increased application usage like end of month activities or "Black Friday" events that occur thanks to commercial campaigns.

**Linear Scalability Scenario:** The load is increased linear over time until the system starts to throw errors, gets unresponsive or crashes. This scenario is used to find the limits of a system.  
  


**Stepped Scalability Scenario:** The load is increased over time, but will be hold for a longer time on the same level until the system starts to throw errors, gets unresponsive or crashes. This scenario is used to find the limits of a system.

