|  |  |
| --- | --- |
|  |  |

**Testreport**

**Authors**Rimac Valdez Quispe  
Robert Kettler  
Daniel Weidle

# Inhaltsverzeichnis

Inhaltsverzeichnis 2

Abbildungsverzeichnis 3

Tabellenverzeichnis 3

Abkürzungsverzeichnis 4

Änderungsverzeichnis 5

Vorwort 6

1 Einleitung 7

1.1 Ziel der Testphasendokumentation 7

1.1 Testphasen 7

2 Testfallgenerierung 8

2.1 Testbasis 8

3 Testfälle 10

3.1 Version 0.1 10

3.2 Version 0.2 10

3.3 Version 1.0 10

# Abbildungsverzeichnis

Abbildung 1: Verbinden eines Textteils mit einer Formatvorlage 9

# Tabellenverzeichnis

Tabelle 1: Unterschiede zwischen Abbildungen und Tabellen 10

# Abkürzungsverzeichnis

# Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Stand** | **Änderungsbeschreibung** |
| 1.0 | 13.06.2016 | Erstellung des Dokuments. |

# Preface

Dieses Dokument dient zur Dokumentation der in den Testphasen durchgeführten Aktivitäten. Die Testphasen orientieren sich an den von der Projektleitung festgelegten Release Zeiträumen. Des Weiteren dient dieses Dokument der Nachvollziehbarkeit, der Testfallgeneration.

# Introduction

Dieses Kapitel gibt einen Überblick, über das Ziel dieses Dokumentes und ordnet den Release Versionen den in den Testphasen durchgeführten Tests zu.

## Ziel der Testphasendokumentation

Das Ziel diese Dokumentation ist, die durchgeführten Tests innerhalb des Testprozesses zu dokumentieren.

## Testphasen

Die Testphasen orientieren sich an den von der Projektplanung vorgegebenen Release Terminen. Es handelt sich dabei um drei Phasen. Jede dieser Phasen endet mit dem Release einer Version. Die nachfolgende Tabelle zeigt eine Zuordnung der Phasen zu den korrespondierenden Versionen, sowie den durchgeführten Tests.

Tabelle 1: Phasenzeiträume und Release Versionen

|  |  |  |  |
| --- | --- | --- | --- |
| **Beginn** | **Ende** | **Version** | **Durchgeführte Tests** |
| |  |  | | --- | --- | | 09.05.2016 | 06/Jun/16 | | 06.06.2016 | 0.1 | ? |
| 06.06.2016 | 13.06.2016 | 0.2 | ? |
| 13.06.2016 | 24.06.2016 | 0.3 | ? |

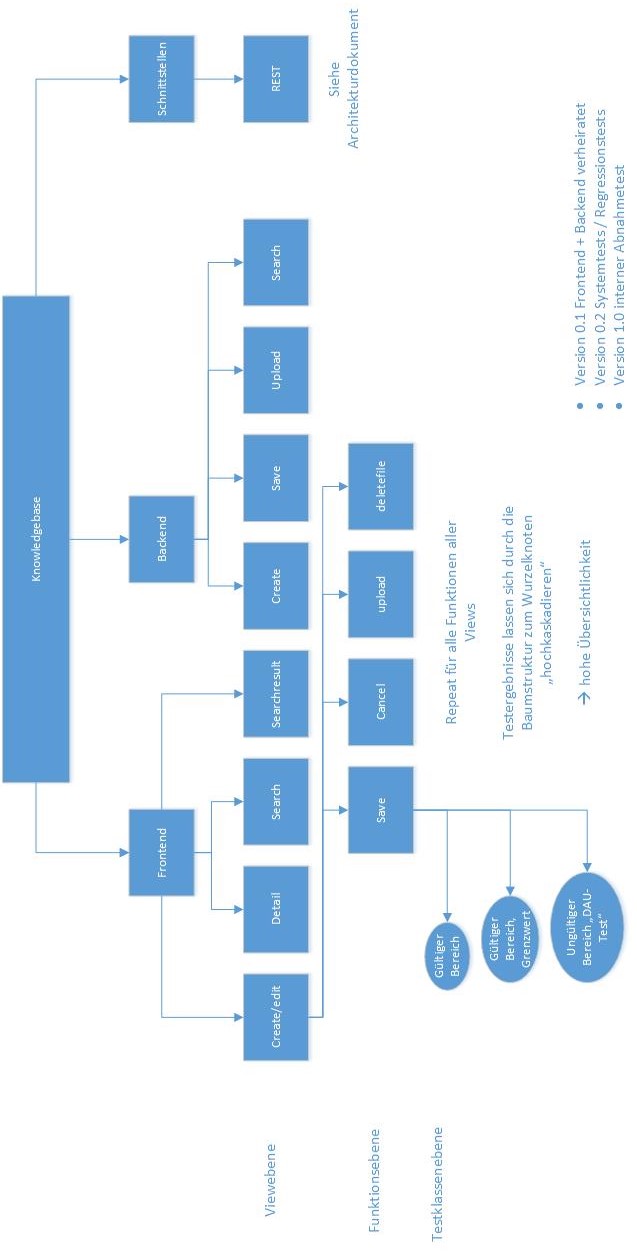
# Testfallgenerierung

Die in diesem Dokument aufgeführten Testfälle leiten sich aus dem, vom Qualitätsmanagment, erstellten Testplan ab. In Abschnitt 2.1 wird auf den Aufbau des Testplans eingegangen, um dem Leser ein Verständnis für die Testfallgenerierung zu vermitteln.

## Testbasis

Das in Abbildung 1 dargestellte Organigramm zeigt den Aufbau des Testplans. Auf Basis dieses Planes lassen sich drei Testbereiche identifizieren. Es handelt sich dabei um das Frontend, das Backend, sowie die Restschnittstellen zwischen dem Backendserver und der Searchengine. Neben der logischen Aufteilung spezifiziert der Testplan drei aufeinander aufbauende Ebenen. Die Viewebene entspricht den vom Benutzer aufrufbaren Seiten. Jede einzelne dieser Seiten bietet dem Benutzer eine bestimmte Anzahl an Funktionalitäten. Diese werden in der Funktionsebene dargestellt. Anschließend werden für sämtliche Funktionen Testfälle mithilfe von Äquivalenzklassenbildung generiert. Definiert sind die Testklassen, „Gültiger Bereich“, „Gültiger Bereich – Grenzwert“, sowie „Ungültiger Bereich.“. Auf Basis dieses Plans werden von den Testern Testfälle generiert. Eine Auflistung sämtlicher durchgeführter Tests findet sich in Kapitel 3.

Abbildung 1: Organigramm des Testplans



# Test summary

All tests are performed on the recommended hardware specifications described below. The results are based on local communication – network traffic is excluded.

Table 1: Recommended system requirements

|  |  |
| --- | --- |
|  | **Recommended requirements** |
| **Operating system** | Windows 7, Linux or MacOS |
| **CPU** | Core 2 Duo at 2,6 GHz with VT-X |
| **Memory** | 2 GB RAM |
| **Hard drive** | At least <<XX>> GB free hard disk space |
| **Network** | 1 Gbit/s |
| **Internet** | <<PRODUCTNAME>> requires internet access to download the Docker images. After the build is done, internet connection isn’t required anymore. |

## Testing Scope

The following lists shows the tested and not tested aspects of <<NAME>> software.

In scope

Functional Testing for the following modules are in scope of testing

* Database Connector
* File System Connector
* Search Engine Connector

Integration Testing are performed to proof the interaction of the application with the services

* Open search server
* Nginx webserver
* Mongodb
* ClientUI

Usability tests….<<RIMAC>>

Regression Testing

Out of scope

* Security and Penetration tests was not done for this application

## Unit/Module Test

These tests aim on the key functionalities of the product. According the “Qualitätshandbuch” they cover functional and nonfunctional Requirements. Like described in the Architecture Documentation, the software is split in modules. According to this, the tests are also split in these modules. Table 1 gives a distribution overview of the tests to the modules.

Table 2: Distribution of tests on modules

|  |  |
| --- | --- |
| **Module** | **# Tests** |
| Database Connector |  |
| File System Connector |  |
| Search Engine Connector |  |
| Service API |  |

Unit and Module Tests were performed automatically and continuously during the development of the software on every build by the build server. The test environment mirrors the production environment, which was specified in the requirements. In discharge of this, the integration of services like the open search server was tested early in the development.

## Usability Test

<<RIMAC>>

## Performance Test

The performance of <<PRODUKTNAME>> were performed on every release with the Apache Benchmark tool ApacheBench. According to the specification in the requirements document <<PRODUCTNAME>> is able to handle up to 5000 concurrent requests without losing performance - this is 100x more than specified.

## User Acceptance Test

<<RIMAC/MARCEL>>

# Test results

## Unit/Module Test

Tester: R. Kettler; D. Weidle  
Release Version: 1.0

<< REPORT REPORT>>

## System Test

28

<< RIMAC FRONTEND TEST>>

## Usability Test

<<RIMAC/MARCEL>>

## Performance Test

Performance Tests are performed on every release since Version 0.2 to make sure the major functionalities are working fine in defined conditions. Detailed reports can be found on appendix.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#ID** | **Description** | **Tester** | **Version** | **Pass/Fail** |
| P-01 | Handle 50 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 0.2 | Pass |
| P-02 | Handle 100 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 0.2 | Pass |
| P-03 | Handle 200 search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 0.2 | Pass |
| P-04 | Handle 500 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 0.2 | Pass |
| P-05 | Handle 1000 concurrent request and respond them within 2 seconds. | R. Kettler | 0.2 | Pass |
| P-06 | Handle 50 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 1.0 | Pass |
| P-07 | Handle 100 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 1.0 | Pass |
| P-08 | Handle 200 concurrent request and respond them within 2 seconds. | R. Kettler | 1.0 | Pass |
| P-09 | Handle 500 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 1.0 | Pass |
| P-10 | Handle 1000 concurrent search- and 3 create requests and respond them within 2 seconds. | R. Kettler | 1.0 | Pass |

## User Acceptance Test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #ID | Description | Tester | Version | Pass/Fail |
| UA-01 |  |  |  |  |
| UA-02 |  |  |  |  |

# Metrics

## Bug report

|  |  |  |  |
| --- | --- | --- | --- |
| **Test cases planned** | **Test cases executed** | **Passed** | **Failed** |
|  |  |  | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Low** | **Middle** | **High** | **Highest** |
| Closed | 1 |  |  |  |
| Open |  |  |  |  |

## List of open bugs

JIRA EXPORT KOMMT FREITAG 9:00