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

**Testreport**

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

# Table of contents

Table of contents 2

List of Figures 3

List of Tables 3

List of Abbreviations 4

Version History 5

Preface 6

1 Introduction 7

1.1 Goal of this document 7

1.1 Test Phases 7

2 Test case Generation 8

2.1 Test base 8

3 Test summary 10

3.1 Testing Scope 10

3.2 Unit/Module Test 11

3.3 System Test 12

3.4 Usability Test 12

3.5 Performance Test 13

3.6 User Acceptance Test 13

4 Test results 14

4.1 Unit/Module Test 14

4.2 System Test 16

4.3 Usability Test 21

4.4 Performance Test 24

4.5 User Acceptance Test 26

5 Bug report 27

5.1 List of open bugs 27

Appendix 29

# List of Figures

[Figure 1 Test plan organigram 9](#_Toc454226313)

[Figure 2: Overall code coverrage 16](#_Toc454226314)

# List of Tables

[Table 1: Performed tests associated to a specific version 7](#_Toc454226299)

[Table 2: Recommended system requirements 10](#_Toc454226300)

[Table 3: Distribution of unit tests on modules 11](#_Toc454226301)

[Table 4: Distribution of test cases on system views 12](#_Toc454226302)

[Table 5: Functions tested with respect to usability 12](#_Toc454226303)

[Table 6: System Test Cases 17](#_Toc454226304)

[Table 7: Usability Scenario - Create article and format content 21](#_Toc454226305)

[Table 8: Usability Scenario - Search article 22](#_Toc454226306)

[Table 9: Usability Scenario - Uploading a file 22](#_Toc454226307)

[Table 10: Usability Scenario - Delete File 23](#_Toc454226308)

[Table 11: Usability Scenario - Alert author 23](#_Toc454226309)

[Table 12: Usability Scenario - Delete an article 24](#_Toc454226310)

[Table 13: Acceptance test summary 26](#_Toc454226311)

[Table 14: Classification of found bugs 27](#_Toc454226312)

# List of Abbreviations

# Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| 0.1 | 13.06.2016 | Initial |
| 0.2 | 17.06.2016 | Added open bugs list, bugs in total |
| 1.0 | 20.06.2016 | Fixed layout, typos, added open bug |

# Preface

This document describes and documents all actions performed by the testing team, related to the Metis knowledgebase project. Chapter one gives a brief introduction about the goal of this document and describes which types of tests have been performed during the single phases of the project. Chapter two describes the test case generation process. The third chapter summarizes which kind of tests were performed and how they were performed. The test results are presented in chapter 4. The last chapter lists bugs which are not fixed, and informs the reader about their criticality.

# Introduction

This chapter gives an overview, about the goals of this document and maps to every release version which types of tests have been executed.

## Goal of this document

The Goal of this document is the documentation of all performed tests which were part of the test process,

## Test Phases

The test phases are defined by the release dates, which are part of the project plan. There are three phases, each of this phases ends with a release of one version. The following table shows a mapping between the single phases and their related version number.

Table 1: Performed tests associated to a specific version

|  |  |  |  |
| --- | --- | --- | --- |
| **Start** | **End** | **Version** | **Performed tests** |
| 09.05.2016 | 06.06.2016 | 0.1 | Unit-tests, system tests |
| 06.06.2016 | 13.06.2016 | 0.2 | system tests, unit tests, integration tests, usability tests, performance tests, regression tests |
| 13.06.2016 | 24.06.2016 | 1.0 | system tests, unit tests, integration tests, performance tests, regression tests |

# Test case Generation

The listed test cases derive from the test plan which has been created by the quality management. Section 2.1 describes the structure of the test plan and informs the reader about the process of test case generation.

## Test base

Figure 1 shows an organigram, which describes the structure of the test plan. According to this plan, there are three different test sections. Namely the frontend, the backend and the rest interface between the backend server and the search engine. Beside this logical separation, the test plan also specifies three levels of detail, which are built on each other. The view level represents the building blocks which are used to create any page, which can be opened by the user. Every single view provides a certain amount of functionalities. These functions are described in the underlying function level. The last level describes for every function, in the function view, test cases which are part of a specific equivalence class. There are three different classes. The classes are „Valid Area “, “Valid Area – Boundary value” and “Invalid Area”. Based on this plan, all tests are generated by the testers themselves. A listing of all executed test cases can be found in chapter three.

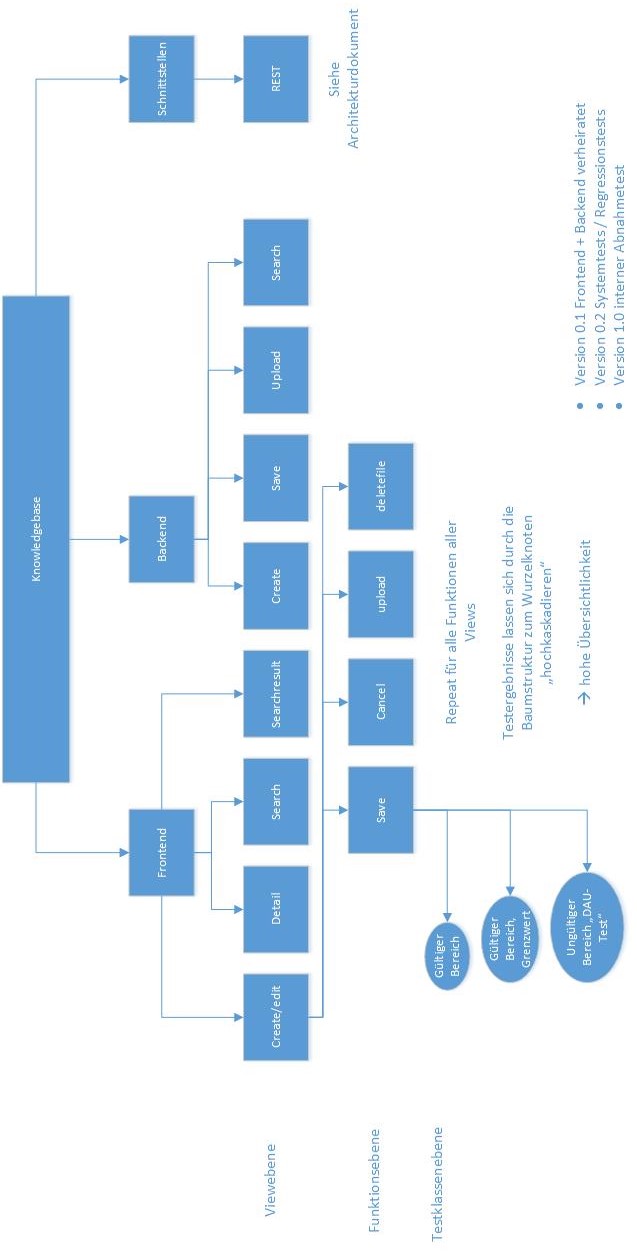


Figure 1 Test plan organigram

# 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 2: 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 5 GB free hard disk space |
| **Network** | 1 Gbit/s |
| **Internet** | Metis requires internet access to download the Docker images. After the build is done, internet connection isn’t required anymore. |

## Testing Scope

The following list shows the tested and not tested aspects of Metis software.

In scope

Functional testing for the following modules are in scope of testing

* Database Connector
* File System Connector
* Search Engine Connector
* Service API and Handlers

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

* Open search server
* Nginx webserver
* MongoDB
* ClientUI

Usability tests

* Six scenarios performed by five persons who did not have contact with Metis before. Goal of this tests was measuring the average time needed by the test persons and also getting feedback about the design decisions.

Regression tests

* The regression tests on backend code were performed automatically on every build, to be sure that changes have no malfunction on other features.
* Regression tests regarding the frontend were performed on every release.

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 to 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 3 gives a distribution overview of the tests to the modules.

Each Test case was built out of the underlying requirements specification. This approach ensures that the expected behavior of a functions is tested by choosing input values based on equivalence classes.

Table 3: Distribution of unit tests on modules

|  |  |
| --- | --- |
| **Module** | **# Tests** |
| Database Connector | 4 |
| File System Connector | 23 |
| Search Engine Connector | 14 |
| Service API | 27 |

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.

## System Test

System tests have been performed automated and manually. For Frontend test automation, the selenium java library has been used. All test cases were generated as explained in section 2.1. The following table shows the distribution of test cases on views.

Table 4: Distribution of test cases on system views

|  |  |
| --- | --- |
| **View** | **# Tests** |
| Create/Edit | 14 |
| Detail | 7 |
| Search | 4 |
| Search Results | 3 |

## Usability Test

The usability test contains six different scenarios. In order to gain information about the usability of the product these scenarios were executed by five different persons, which have not used the software before. The following table lists which functions were tested in the usability tests.

Table 5: Functions tested with respect to usability

|  |  |
| --- | --- |
| **Test number** | **Functionality** |
| 1 | Create article and format content |
| 2 | Search article |
| 3 | Upload a file |
| 4 | Delete a file |
| 5 | Alert an author |
| 6 | Delete an article |

## Performance Test

The performance of Metis was tested on every release since version 0.2 with the Apache Benchmark tool ApacheBench. According to the specification in the requirements document, Metis is able to handle up to 1000 concurrent requests without losing significant performance - this is 50x more than specified in the requirements documentation.

## User Acceptance Test

Acceptance test has been performed on Friday 17th of June on Version 1.0. This test includes all scenarios described in section 3.4.

# Test results

This chapter gives a brief overview about all test results. Detailed information about single test cases, such as step by step test case description can be found in test report documents, referenced in the appendix section.

## Unit/Module Test

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

All tests are executed using the Mocha test runner for node.js. The following test report is generated by Mocha’s spec reporter. The following list gives an overview over all executed tests, passing and failing. Previous results can be found at Jenkins build server.

GET

✓ request with q=test (1150ms)

✓ request with q=asdfg

✓ request with q=

With Article:

/api/articles?q=

✓ request with q=test (48ms)

✓ request with q=

✓ request with unknown q (140ms)

✓ search in article (49ms)

✓ search in title (44ms)

✓ search author

✓ search author maxlength

✓ search author with spaces

✓ search author with keyword (66ms)

✓ search author with spaces with keyword

✓ search in file (74ms)

✓ search filename

✓ search filename with extension

/api/articles?ids=

✓ request with ids=ArticleIds[0]

✓ request with ids=[ArticleIds] (78ms)

✓ request with ids=

✓ request with ids=1,2,3

✓ request with valid invalid ids mixed (43ms)

/api/articles/:ArticleId

✓ request the articles just after save (80ms)

/api/articles/:ArticleId?old

✓ request the article old after save

POST

/api/articles/

✓ request with all valid information

With Article:

/api/articles/:articleId/documents

✓ upload json testfile

/api/articles/:articleId/documents

1) upload testfile 10 times to test replacement

PUT

With Article:

/api/articles/:ArticleId

✓ put with valid data

✓ put without data

✓ put with empty json

✓ put with string as data

✓ put with invalid json as data

email

✓ invalid

author

✓ to long

title

✓ to long

DELETE

With Article:

/api/articles

✓ delete existing article without files

/api/articles

✓ delete existing article with files

/api/articles

✓ delete not existing article

/:articleId/documents/:filename

✓ delete file from existing article

✓ delete article.html

✓ delete server.js

/:articleId/documents/:filename

✓ delete file from not existing article

Database

✓ findAllPermArticleIds

✓ deleteTemporaryArticlesOlderThan

✓ should fail for values equal 0

✓ should fail for values less 0

Filesystem

✓ deleteEmptyArticles (2002ms)

✓ deleteTemporaryArticles

file\_system\_connector

test extract html title content

✓ should pass

✓ should pass with empty title

✓ should pass with empty title immediately closed tag

✓ should pass with nested tags

✓ should pass with malformed tags

✓ should pass with malformed tags

✓ should pass without tags

✓ should pass two opening one closing tag

✓ should pass new line

test extract html body content

✓ should pass

✓ should pass empty body

✓ should pass regular text

✓ should pass missing closing tag

✓ should pass missing opening tag

✓ should pass missing tags

✓ should pass with escaped tags

✓ should pass with newline

test wrap content in html

✓ should pass

✓ should pass both params empty

✓ should pass both params whitespace

✓ should pass both params whitespace

To 1): The test „upload test file 10 times to test replacement “tries to upload a file 10 times concurrently. We rated this bug as low, because this scenario cannot occur as long as our browser client is used (see KNOW-209 in 5.1).

This leads to an overall Statement Coverage of 93,44% as Figure 2 shows below.

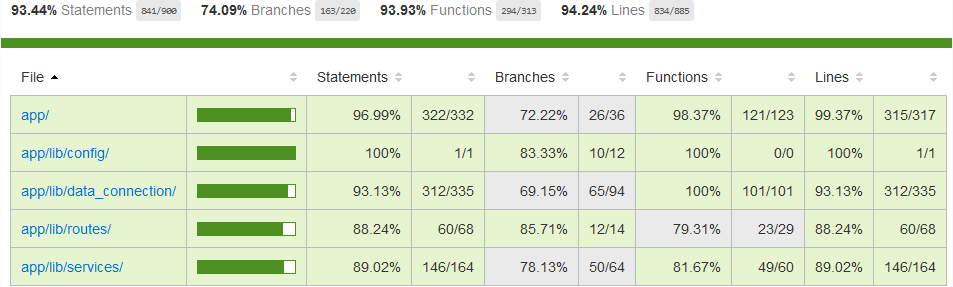


Figure : Overall code coverrage

## System Test

Tester: Rimac Valdez Quispe

All tests below have been performed on a System running Windows 8.1 as operating System. Used Browsers are Chrome and Firefox as specified in the architecture documentation. Some of the test cases failed on Firefox. By manually looking for the reasons, the test author could verify that this has nothing to do with Metis itself. Therefore, behaviorally equivalence between Chrome and Firefox browsers is assumed. The results listed below are the final results and can be viewed as the authoritative source for checking, if a test passed or not.

Table 6: System Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#ID** | **Description** | **Comments** | **Version** | **Pass/Fail** |
| ST-1 | [Save] KNOW-121 - Create article |  | 0.1 | Pass |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-2 | [Save] KNOW-123 – Create article with invalid Email |  | 0.1 | Fail |
| 0.2 | Fail |
| 1.0 | Pass |
| ST-3 | [Save] KNOW-122 - Create article without title |  | 0.1 | Fail |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-4 | [Cancel] KNOW-135 - accidently cancel article creation |  | 0.1 | Fail |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-5 | [Cancel] KNOW-134 - cancel article creation |  | 0.1 | Pass |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-6 | [DeleteArticle] KNOW-195 - delete article which has been deleted by other user | No Warning that article has been deleted. But system does not crash. | 0.1 | - |
| 0.2 | Fail |
| 1.0 | Fail |
| ST-7 | [DeleteFile] KNOW-200 - delete file which has been deleted by other user |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-8 | [DeleteArticle] KNOW-194 - delete article |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-9 | [DeleteFile] KNOW-141 - delete accidently file and cancel editing |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-10 | [DeleteFile] KNOW-140 - delete file from article |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-11 | [Save] KNOW-193 - Save article which has been deleted by other user | No warning that article has been deleted by other user. But System does not crash. | 0.1 | - |
| 0.2 | Fail |
| 1.0 | Fail |
| ST-12 | [Upload] KNOW-191 - Uploading 100 MB File | Metis is able to save 100 MB Files, but the test setup was configured to a maximum of 50 MB files. This is a matter of configuration | 0.1 | - |
| 0.2 | (Fail) |
| 1.0 | (Fail) |
| ST-13 | [Upload] KNOW-138 - Uploading 1 MB File |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-14 | [Upload] KNOW-139 - Uploading 50 MB File | Metis is able to save 50 MB Files, but the test setup was configured to a maximum of 50 MB files. This is a matter of configuration | 0.1 | - |
| 0.2 | (Fail) |
| 1.0 | (Fail) |
| ST-15 | [Alert] KNOW-133 - alert author |  | 0.1 | Pass |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-16 | [Alert] KNOW-181 – alert author when author is unknown |  | 0.1 | Fail |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-17 | [DownloadFile] KNOW-199 – download file which has been deleted by other user | User is directed to a 404 Error site. This is specified as expected result. Therefore, this test passes. | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-18 | [DownloadFile] KNOW-198 - download file from article |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-19 | [Edit] KNOW-205 - edit article which has been deleted by other user |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-20 | [PreviousVersion] KNOW-182 - edit article and watch previous version |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-21 | [PreviousVersion] KNOW-183 - edit article without saving and watch previous version |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-23 | [OpenArticle] KNOW-188 - Open article by clicking on item from result list |  | 0.1 |  |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-24 | [OpenArticle] KNOW-189 - Open article which is deleted | No message found that article has been deleted by other user. But System does not crash | 0.1 |  |
| 0.2 | Fail |
| 1.0 | Fail |
| ST-25 | [SearchMeta] KNOW-186 - search for authors |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-26 | [SearchContent] KNOW-184 - search for content in an article |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |
| ST-27 | [SearchContent] KNOW-185 - search for content in document | This test fails because Metis is not able to find content in secured pdf files. See also section 5.2 open bug list KNOW-217 | 0.1 | - |
| 0.2 | (Fail) |
| 1.0 | (Fail) |
| ST-28 | [SearchContent] KNOW-187 - search for data which does not exist |  | 0.1 | - |
| 0.2 | Pass |
| 1.0 | Pass |

## Usability Test

Tester: Rimac Valdez Quispe; Marcel Hoffmann

The following tables describe the specific usability scenarios in detail. The average time needed to complete a scenario is documented as well as single comments made by the test persons.

Table 7: Usability Scenario - Create article and format content

|  |  |
| --- | --- |
| Task: #\_1 Create article and format content |  |
| Goal/ Output: | Create a new article |
| Input: | User creates a new article with the title „ My database error **“.** |
| Assumption: | Start page is open |
| Steps: | * Click „New Article “. * Write a title * Write three bullet points * Click „Save “ |
| Estimated time for experts: | Maximum 5 minutes |
| Instructions for test person: | Imagine you find a solution for common mistake in your database project. You know that other colleagues work with the same technology you are using. Please create an article named “My database error”. Please add three short sentences and format them as bullet points. |
| Comments of test persons | * Font size while writing article content is too small. |
| Average time needed by test persons | 2 minutes and 46 seconds |

Table 8: Usability Scenario - Search article

|  |  |
| --- | --- |
| Task: #\_2 Search article |  |
| Goal/ Output: | Finding an article by using the search function |
| Input: | - |
| Assumption: | User has its earlier created article open. |
| Steps: | * Click on search bar * Write search term * Click on „search “ * Click on article header for opening an article |
| Estimated time for experts: | Maximum 1 minute |
| Instructions for test person: | Please search for your earlier created article |
| Comments of test persons | Summary view of self-created articles would be good. |
| Average time needed by test persons | 35 seconds |

Table 9: Usability Scenario - Uploading a file

|  |  |
| --- | --- |
| Task: #\_3 File upload |  |
| Goal/ Output: | Uploading a file |
| Input: | - |
| Assumption: | Start page is open |
| Steps: | * Click „Edit “. * Click „File Upload “ * Upload Dialog opens * Choose file * Click ok in the dialog * Click „Save “ |
| Estimated time for experts: | Maximum 1 minute |
| Instructions for test person: | After finding your own article, you want to add a file as attachment. |
| Comments of test persons | I thought I wouldn’t need to click on save for making the upload finally |
| Average time needed by test persons | 50 seconds |

Table 10: Usability Scenario - Delete File

|  |  |
| --- | --- |
| Task: #\_4 Delete File |  |
| Goal/ Output: | Delete a file |
| Input: | - |
| Assumption: | User has its earlier created article open, which has a file attachment |
| Steps: | * Click „Edit “. * Click „ red x Icon “ * Click „Save “ |
| Estimated time for experts: | Maximum 30 seconds |
| Instructions for test person: | After noticing, that you don’t need the file, you want to delete the file. |
| Comments of test persons | - Expectation is, that deletion is only valid after clicking on save.  - I thought the delete article button would delete files, because it is grouped below them (2 Persons said this) |
| Average time needed by test persons | 50 seconds |

Table 11: Usability Scenario - Alert author

|  |  |
| --- | --- |
| Task: #\_5 Alert author |  |
| Goal/ Output: | Alert author, that you found an error in his article |
| Input: | - |
| Assumption: | Start page is open. |
| Steps: | * Click on search bar textbox * Write „Alan Turing “ * Click search button * Open the first search result. * Click Alert button |
| Estimated time for experts: | Maximum 2 Minutes |
| Instructions for test person: | Search for „Alan Turing “and inform the author of the article, that you found a mistake in his article. |
| Comments of test persons | I looked for a button “Send Message” |
| Average time needed by test persons | 33 seconds |

Table 12: Usability Scenario - Delete an article

|  |  |
| --- | --- |
| Task: #\_6 Delete article |  |
| Goal/ Output: | Delete an article |
| Input: | - |
| Assumption: | Alan Turing article is open |
| Steps: | * Click on search bar textbox * Write „database “as search term * Click on „Search “ * Open article by clicking on the header in the result view * Click “Edit” * Click “Delete Article” |
| Estimated time for experts: | Maximum 2 Minutes |
| Instructions for test person: | After alerting the author of the Alan Turing article, you decide to delete your earlier created article. |
| Comments of test persons | - |
| Average time needed by test persons | 32 seconds |

## 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

The following table shows the scenarios which were used as acceptance test. These scenarios are equivalent to the scenarios used for usability testing. Detailed description about the steps needed to complete these scenarios therefore can be found in section 4.3

Table : Acceptance test summary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#ID** | **Description** | **Tester** | **Version** | **Pass/Fail** |
| UA-01 | Create article and format content | R. Kettler | 1.0 | Pass |
| UA-02 | Search article | R. Kettler | 1.0 | Pass |
| UA-03 | Upload a file | R. Kettler | 1.0 | Pass |
| UA-04 | Delete a file | R. Kettler | 1.0 | Pass |
| UA-05 | Alert an author | R. Kettler | 1.0 | Pass |
| UA-06 | Delete an article | R. Kettler | 1.0 | Pass |

# Bug report

The following chapter gives an overview of found and open bugs and their criticalities.

As in Table 13 shown, 82 percent of found bugs are fixed. Due the reached coverage described in chapter 4.1, its not expected that high risk, high business value bugs are included.

Table 14: Classification of found bugs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Low** | **Middle** | **High** | **Highest** |
| Closed | 1 | 43 | 2 | 1 |
| Open | 1 | 3 | 0 | 0 |

A full description of the open bugs can be found in the follow chapter.

## List of open bugs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **[KNOW-218] Opening an already deleted article does not show any kind of error message** Created: 14/Jun/16 updated: 14/Jun/16 | | | | |
| **Status:** | | Backlog | | |
| **Project** | | [Knownana](https://danielweidle.de/jira/secure/BrowseProject.jspa?id=10000) | | |
| **Components** | | [Frontend](https://danielweidle.de/jira/issues/?jql=project%3D10000%20AND%20%22component%22%3D10000%20ORDER%20BY%20priority%20ASC) | | |
| **Version** | | 1.0 | | |
| **Type:** | Bug | | **Priority:** | Medium |
| **Author:** | [Rimac Valdez](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=rvaldez) | | **Assigned to:** | [Christoph Brutscher](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=cbrutscher) |
| **Solution:** | Not solved | | **Voices:** | 0 |
| **Description** | The user does not get any kind of feedback. If  he tries to open an already deleted article he is redirected to an empty article | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **[KNOW-217]**[**Secured**](https://danielweidle.de/jira/browse/KNOW-217) **pdf files can not be parsed** Created: 14/Jun/16 updated: 14/Jun/16 | | | | |
| **Status:** | | Backlog | | |
| **Project** | | [Knownana](https://danielweidle.de/jira/secure/BrowseProject.jspa?id=10000) | | |
| **Components** | | [Backend](https://danielweidle.de/jira/issues/?jql=project%3D10000%20AND%20%22component%22%3D10001%20ORDER%20BY%20priority%20ASC) | | |
| **Version** | | 1.0 | | |
| **Type:** | | Bug | | |
| **Author:** | Bug | | **Priority:** | Medium |
| **Solution:** | [Rimac Valdez](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=rvaldez) | | **Assigned to:** | [Alex Schramm](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=aschramm) |
| **Description** | Not solved | | **Voices:** | 0 |
| **Status:** | Secured pdf files cannot be parsed. Content cannot be found | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **[KNOW-211] [Displaying\_Files](https://danielweidle.de/jira/browse/KNOW-211)** Created: 14/Jun/16 updated: 14/Jun/16 | | | |
| **Status:** | Backlog | | |
| **Project** | [Knownana](https://danielweidle.de/jira/secure/BrowseProject.jspa?id=10000) | | |
| **Components** | [Frontend](https://danielweidle.de/jira/issues/?jql=project%3D10000%20AND%20%22component%22%3D10000%20ORDER%20BY%20priority%20ASC) | | |
| **Version** | 1.0 | | |
| **Type:** | Bug | | |
| **Author:** | [Daniel Weidle](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=dweidle) | **Priority:** | Medium |
| **Solution:** | Not solved | **Assigned to:** | [Yannic Soethoff](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=ysoethoff) |
| **Description:** | If an article has an attached file, the articles detail view shows the filename and appends an “,”. . | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **[KNOW-209]**[**Upload**](https://danielweidle.de/jira/browse/KNOW-209) **article 10 times**  Created: 14/Jun/16 updated: 14/Jun/16 | | | |
| **Status:** | Backlog | | |
| **Project** | [Knownana](https://danielweidle.de/jira/secure/BrowseProject.jspa?id=10000) | | |
| **Components** | [Frontend](https://danielweidle.de/jira/issues/?jql=project%3D10000%20AND%20%22component%22%3D10000%20ORDER%20BY%20priority%20ASC) | | |
| **Version** | 0.1 | | |
| **Type:** | Bug | | |
| **Author:** | [Robert](https://danielweidle.de/jira/secure/ViewProfile.jspa?name=dweidle) Kettler | **Priority:** | Low |
| **Solution:** | Not solved | **Assigned to:** |  |
| **Description:** | If a file gets attached parallel 10 times to an article, the upload will fail. | | |

# Appendix

1. Systemtestreport.html
2. Performance test reports