|  |  |
| --- | --- |
|  | «Audio Cataloger» Project |
| Test Plan  **Project Documentation** |
| **Background** | Estimations, schedule, strategy, and metrics are needed to organize the testing process efficiently. |
| **Purpose** | To organize the testing process effective and efficient during  the whole project period. |
| **Scope** | Testing process description, metrics, schedule, resources. |
| **Audience** | Management staff, QA team, project team. |
| **File** | 02 03 - Test Plan.docx |

Contents

1. [Project scope and main goals 3](#_bookmark0)
2. [Requirements to be tested 3](#_bookmark1)
3. [Requirements NOT to be tested 3](#_bookmark2)
4. [Test strategy and approach 3](#_bookmark3)
   1. [General approach 3](#_bookmark4)
   2. [Functional testing levels 3](#_bookmark5)
5. [Criteria 4](#_bookmark6)
6. [Resources 4](#_bookmark7)
7. [Schedule 4](#_bookmark8)
8. [Roles and responsibilities 4](#_bookmark9)
9. [Risk evaluation 4](#_bookmark10)
10. [Documentation 5](#_bookmark11)
11. [Metrics 5](#_bookmark12)

# Project scope and main goals

Correct automated conversion of text documents in different source encodings to one destination encoding with performance significantly higher than human performance during the same actions.

A console tool with functionality of creating a list of all audio files in users’ possession with duplications cross-reference and corrupted audio files.

* The tool must create files duplications cross-reference.
* The tool should use smart comparison algorithms for better performance.
* The tool should support mp3, flac, wav, ogg, wma file formats.
* The resulting list should be viewable via web-browser (HTML output support).
* The resulting list should be editable in spreadsheet software (CSV output support).
* The tool reliability should be higher than competitors.

# Requirements to be tested

See referenced sections in “File Converter Requirements.docx”:

* UR-1.\*: smoke test.
* UR-2.\*: smoke test, critical path test.
* UR-3.\*: critical path test.
* BR-1.\*: smoke test.
* BR-2.\*: critical path test.
* BR-3.\*: smoke test.
* BR-4.\*: critical path test.
* QA-1.\*: critical path test.
* QA-2.\*: critical path, extended test.
* QA-3.\*: smoke test.
* L-4: smoke test.
* DS-\*: smoke test, critical path test.

# Requirements NOT to be tested

See referenced sections in “File Converter Requirements.docx”:

* SC-1: the application is a console one by design.
* SC-2, L-1, L-2: the application is developed with proper PHP version.
* L-3: no implementation required.

# Test strategy and approach

## General approach

The application is to be configured once by an experienced specialist and later used by end users, for whom only one operation is available – placing the file into the input directory. Therefore, issues of usability, security, etc. not explored during testing.

The application is to be configured once by an experienced specialist and later used by end users, for whom only one operation is available – selecting the directories for catalog creation.

## Functional testing levels

* Smoke test: automated with batch files under Windows and Linux.
* Critical path test: executed manually.
* Extended test: not executed as the probability of defects detection on this level is negligibly small.
* Extended test: automated with batch files containing a large number or randomized data sets under Windows and Linux.

Due to the team cross-functionality, a significant contribution to quality improvement can be expected from the code review combined with manual testing using the white box method. Unit-testing will be applied partly due to time limitations: QA-1, QA-2, QA-3.

# Criteria

* Acceptance criteria: 100% success of test cases on smoke test level, 90% success of test cases on critical path test level and 70% success of test cases on extended level (see “[Test cases success percentage](#_bookmark13)” metric) if 100% of critical and major bugs are fixed (see “[Overall defects fixed percentage](#_bookmark14)” metric). Final requirements coverage by tests (see “[Requirements](#Requirements) [coverage by tests](02_03_-_Test_Plan_Sample%20—%20result.docx)” metric) should be at least 80%.
* Testing start criteria: new build.
* Testing pause criteria: critical path test must begin only after 100% success of test- cases on the smoke test (see “[Test cases success percentage](#_bookmark13)”); test process may be paused is with at least 25% test-cases executed there is at least 50% failure rate (see “[Stop-factor](#_bookmark16)” metric).
* Testing resumption criteria: more than 50% of bugs found during the previous iteration are fixed (see “[Ongoing defects fixed percentage](#_bookmark15)” metric).
* Testing finish criteria: more than 80% planned for the current iteration test cases are executed (see “[Test-cases execution percentage](#_bookmark17)”).

# Resources

* Software: eight virtual machines (two with Windows 10 Ent x64, two with Linux Ubuntu 18 LTS x64), four IntelliJ IDEA licenses (latest version available).
* Hardware: four standard workstations (16GB RAM, i7 4.5GHz).
* Personnel:
  + One senior developer with testing experience (100% workload during all project time). Roles: team lead, senior developer.
  + One senior(middle) developer with testing experience (100% workload during all project time). Role: middle developer.
  + Two tester with Java knowledge (100% workload during all project time). Role: testers.
* Time: 15 days (88 work hours).
* Finances: according to the approved budget.

# Schedule

* 01.05 - 03.05 – requirements testing and finalizing.
* 04.05 – 05.05, 08.05 – test-cases and scripts for automated testing creation.
* 09.05 - 10.05 – main testing stage (test-cases execution, defect reports creation).
* 11.05 – testing finalization, reporting.
* 12.05, 15.05 – test-cases and scripts updates, extended test scripts creation.
* 16.05 – 17.05 – additional testing stage (reverse tests, extended tests, defect reports creation).
* 18.05 – testing finalization, reporting.
* 19.05 – reserved day for unexpected issues.

# Roles and responsibilities

* Senior developer: participation in requirements testing and code review.
* Senior(middle) developer: participation in requirements testing and code review, scripts creation.
* Testers: documentation creation, test-cases execution, participation in code-review.

# Risk evaluation

* Personnel (low probability): if any team member is inaccessible, we can contact the representatives of the “Cataloger” project to get a temporary replacement (the commitment from the “Cataloger” PM John Smith was received).
* Time (high probability): the customer has indicated a deadline of 16.05, therefore time is a critical resource. It is recommended to do our best to complete the project by 14.05 so that one day (15.05) remains available for any unexpected issues.
* Other risks: no other specific risks have been identified.

# Documentation

* Requirements. Responsible person – tester 1 and 2, deadline – 03.05.
* Test cases and defect reports. Responsible – tester 1 and 2, creation period – 04.05-10.05.
* Scripts for automated testing docs. Responsible – senior(middle) developer, creation period – 04.05 – 08.05.
* Test result report. Responsible person – tester 1 and 2, deadline – 11.05.
* Test cases updates and defect reports. Responsible – tester 1 and 2, creation period – 12.05-17.05.
* Scripts for automated testing updates and extended test docs. Responsible – senior(middle) developer, creation period – 12.05 – 15.05.
* Test result report. Responsible person – tester 1 and 2, deadline – 18.05.

# Metrics

* Test cases success percentage:

𝑇 𝑆𝑃 = 𝑇 𝑆𝑢𝑐𝑐𝑒𝑠𝑠 / 𝑇 𝑇𝑜𝑡𝑎𝑙 ∙ 100%, where

𝑇 𝑆𝑃 – percentage of successfully passed test cases,

𝑇 𝑆𝑢𝑐𝑐𝑒𝑠𝑠 – quantity of successfully passed test cases,

𝑇 𝑇𝑜𝑡𝑎𝑙 – total quantity of executed test cases.

Minimally acceptable borders:

* Beginning project phase: 10%.
* Main project phase: 40%.
* Final project phase: 80%.
* Overall defects fixed percentage:

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑇𝑃 = 𝐷𝐿𝑒𝑣𝑒𝑙 𝐶𝑙𝑜𝑠𝑒𝑑 / 𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑜𝑢𝑛𝑑 ∙ 100%, where

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑇𝑃 – overall defects fixation percentage by 𝐿𝑒𝑣𝑒𝑙 during all project lifetime,

𝐷𝐿𝑒𝑣𝑒𝑙 𝐶𝑙𝑜𝑠𝑒𝑑 – quantity of defects of 𝐿𝑒𝑣𝑒𝑙 fixed during all project lifetime,

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑜𝑢𝑛𝑑 – quantity of defects of 𝐿𝑒𝑣𝑒𝑙 found during all project lifetime.

Minimally acceptable borders:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Defect severity | | | |
| Minor | Medium | Major | Critical |
| Project phase | Beginning | 10% | 40% | 50% | 80% |
| Main | 15% | 50% | 75% | 90% |
| Final | 20% | 60% | 100% | 100% |

* Ongoing defects fixed percentage:

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝐶𝑃 = 𝐷𝐿𝑒𝑣𝑒𝑙 𝐶𝑙𝑜𝑠𝑒𝑑 / 𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑜𝑢𝑛𝑑 ∙ 100%, where

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝐶𝑃 – defects fixation percentage by 𝐿𝑒𝑣𝑒𝑙 (defects found in the previous build and fixed in the current build),

𝐷𝐿𝑒𝑣𝑒𝑙 𝐶𝑙𝑜𝑠𝑒𝑑 – quantity of defects of 𝐿𝑒𝑣𝑒𝑙 fixed in the current build,

𝐷𝐿𝑒𝑣𝑒𝑙 𝐹𝑜𝑢𝑛𝑑 – quantity of defects of 𝐿𝑒𝑣𝑒𝑙 found in the previous build.

Minimally acceptable borders:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Defect severity | | | |
| Minor | Medium | Major | Critical |
| Project phase | Beginning | 60% | 60% | 60% | 60% |
| Main | 65% | 70% | 85% | 90% |
| Final | 70% | 80% | 95% | 100% |

* Stop-factor:

𝑆 = { 𝑌𝑒𝑠, 𝑇 𝐸 ≥ 25% && 𝑇 𝑆𝑃 < 50%

𝑁𝑜, 𝑇 𝐸 < 25% || 𝑇 𝑆𝑃 ≥ 50% } , where

𝑆 – decision to pause the testing process,

𝑇 𝐸 – current tests executed value,

𝑇 𝑆𝑃 – current tests successfully passed value.

* Test-cases execution percentage:

𝑇 𝐸 = 𝑇 𝐸𝑥𝑒𝑐𝑢𝑡𝑒𝑑 / 𝑇𝑃𝑙𝑎𝑛𝑛𝑒𝑑 ∙ 100%, where

𝑇 𝐸 – test-cases execution percentage,

𝑇 𝐸𝑥𝑒𝑐𝑢𝑡𝑒𝑑 – quantity of executed test-cases,

𝑇 𝑃𝑙𝑎𝑛𝑛𝑒𝑑 – quantity of planned (to execution) test-cases.

Levels (borders):

* + Minimal: 80%.
  + Desired: 95%-100%.
* Requirements coverage by tests:

𝑅 𝐶 = 𝑅 𝐶𝑜𝑣𝑒𝑟𝑒𝑑 / 𝑅 𝑇𝑜𝑡𝑎𝑙 ∙ 100%, where

𝑅 𝐶 – requirements coverage by tests (percentage),

𝑅 𝐶𝑜𝑣𝑒𝑟𝑒𝑑 – quantity of requirements covered with test-cases,

𝑅 𝑇𝑜𝑡𝑎𝑙 – overall quantity of requirements.

Minimally acceptable borders:

* Beginning project phase: 40%.
* Main project phase: 60%.
* Final project phase: 80% (90%+ recommended).