«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	AudioCataloger_TestPlan.docx

Contents

1. Project scope and main goals

Development of a tool to:

- Catalog audio files.
- Find duplicates of audio files.
- Find corrupted audio files.

Main goals of the project:

- Provide the Customer with a quick and simple tool to create a list of all audio files in his possession along with duplications cross-reference.
- The resulting list should be viewable via web-browser for quick review, and editable in spreadsheet software for thorough review and processing.
- The tool should not fail (for any reason) during its working process (unlike many competing tools).

2. Requirements to be tested

See referenced sections in “Audio Cataloger Requirements.docx”:

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

3. 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 using Java of proper version.
- QA-1.1: this performance characteristic is at the bottom border of typical operations performance for such applications.
- L-3: no implementation required.

.

4. Test strategy and approach

4.1. 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.

4.2. 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.

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 not be applied due to extreme time limitations.

5. Criteria

- Acceptance criteria: 100% success of test cases on smoke test level and 90% success of test cases on critical path test level (see "Test cases success percentage" metric) if 100% of critical and major bugs are fixed (see "Overall defects fixed percentage" metric). Final requirements coverage by tests (see "Requirements coverage by tests" 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"); test process may be paused is with at least 25% test-cases executed there is at least 50% failure rate (see "Stop-factor" metric).
- Testing resumption criteria: more than 50% of bugs found during the previous iteration are fixed (see "Ongoing defects fixed percentage" metric).
- Testing finish criteria: more than 80% planned for the current iteration test cases are executed (see "Test-cases execution percentage").

6. Resources

- Software: four virtual machines (two with Windows 10 Ent x64, two with Linux Ubuntu 18 LTS x64), standard JRE (Minimal version – 8.0.60).
- Hardware: two standard workstations (8GB RAM, i7 3GHz).
- Personnel:
    - Two senior developers (100% workload during allproject time). Roles: team lead, senior developer.
    - Two testers (100% workload during all project time).Role: tester.
- Time: 15 days (120 work hours).
- Finances: according to the approved budget.

7. Schedule

- 25.05 – requirements testing and finalizing.
- 26.05 – test-cases and scripts for automated testing creation.
- 28.05-06.06 – main testing stage (test-cases execution, defect reports creation).
- 07.06 – testing finalization, reporting.

8. Roles and responsibilities

- Senior developers: participation in requirements testing and code review.
- Testers: documentation creation, test-cases execution, participation in code-review.

9. 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 (low probability): the customer has indicated a deadline of 10.06. It is recommended to do our best to complete the projectby 07.06 so that three days remain available for any unexpected issues. 15 days should be enough to complete the software testing.
- Other risks: no other specific risks have been identified.

10. Documentation

- Requirements. Responsible person – tester, deadline – 25.05.
- Test cases and defect reports. Responsible – tester, creation period – 26.05-06.06.
- Test result report. Responsible person – tester, deadline – 07.06.

11. Metrics

- Test cases success percentage:

  $T^{SP} = \frac{T^{Success}}{T^{Total}} \cdot 100\%$, 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:

  $D^{FTP}_{Level} = \frac{D^{Closed}_{Level}}{D^{Found}_{Level}} \cdot 100\%$, where

  D^{FTP}_{Level}– overall defects fixation percentage by $Level$ during all project lifetime,

  D^{Closed}_{Level} – quantity of defects of $Level$ fixed during all project lifetime,

  D^{Found}_{Level} – 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:

  $D^{FCP}_{Level} = \frac{D^{Closed}_{Level}}{D^{Found}_{Level}} \cdot 100\%$, where

  D^{FCP}_{Level}– defects fixation percentage by $Level$ (defects found in the previous build and fixed in the current build),

  D^{Closed}_{Level} – quantity of defects of $Level$ fixed in the current build,

  D^{Found}_{Level} – 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:

$S = \begin{cases} Yes, T^E \geq 25\% \;\&\&\; T^{SP} < 50\% \\ No, T^E < 25\% \;||\; T^{SP} \geq 50\% \end{cases}$, where

S – decision to pause the testing process,
T^E – current T^E value,
T^{SP} – current T^{SP} value.

- Test-cases execution percentage:

$T^E = \frac{T^{Executed}}{T^{Planned}} \cdot 100\%$, 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:

$R^C = \frac{R^{Covered}}{R^{Total}} \cdot 100\%$, 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).