Software Requirements Specification

for

Media Player, Release 1.0

Version 1.0 approved

Prepared by Chelsea Davis, Jess Albrecht, and James Felts

Team Ctrl-Alt-Delete

November 19, 2015

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
CD/JA/JF	11/19/15	Initial draft	1.0 draft 1
CD/JA	11/23/15	Changes to initial draft for turn-in	1.0 draft 2

1. Introduction

1.1 Purpose

This SRS describes the software functional and nonfunctional requirements for release 1.0 of Team Ctrl-Alt-Delete's Media Player project. This document is intended to be used by the members of the project team that will implement and verify the correct functioning of the system. Unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

1.2 Intended Audience and Reading Suggestions

This document is intended for developers, testers, and users so they know what features we have planned, have completed, and how to use them. Suggested reading order is simply beginning to end, as the beginning of the document is general and covers high-level information, while the end of the document covers the more specific details of the program.

1.3 Project Scope

Our software is intended for minimalist media consumers who want a basic media playback experience that won't soak up memory but provide more advanced features such as playlists and media searching.

1.4 References

Team Coding Standards - https://docs.google.com/document/d/1FxQWAX4gOGwdqSOjnU6yMg9hnKdlAwxmcmMx0MXdcJ4/edit?ts=56057ee4

2. Overall Description

2.1 Product Perspective

We have been tasked with creating a new media player that finds local media files, organizes them in a database and allows playback from playlist as well as searching and sorting of files by artist and genre.

2.2 Product Features

Media player will scan through the user's standard Windows Music and Video directories, gather & display info from usable media files, and allow the user to play their files. It will also allow the user to search thought the files by artist and genre, and includes various controls such as song looping, volume control, and a time slider to allow advancing through a given song.

2.3 User Classes and Characteristics

Expected users include students, teachers, graders, and average computer media consumer with previous media playing experience. There will be no special privileges or security levels. Our interface is designed to be familiar to the average media player with familiar interfaces such as play, pause buttons, timeline and volume sliders and listing available media to be played which reacts to mouse input.

2.4 Operating Environment

Software will run in Windows 7 or greater, and .NET 4.5.1 or newer.

2.5 Design and Implementation Constraints

Security considerations include preventing Sql injection. For our coding standards please view our attached coding standards PDF (1.4 References).

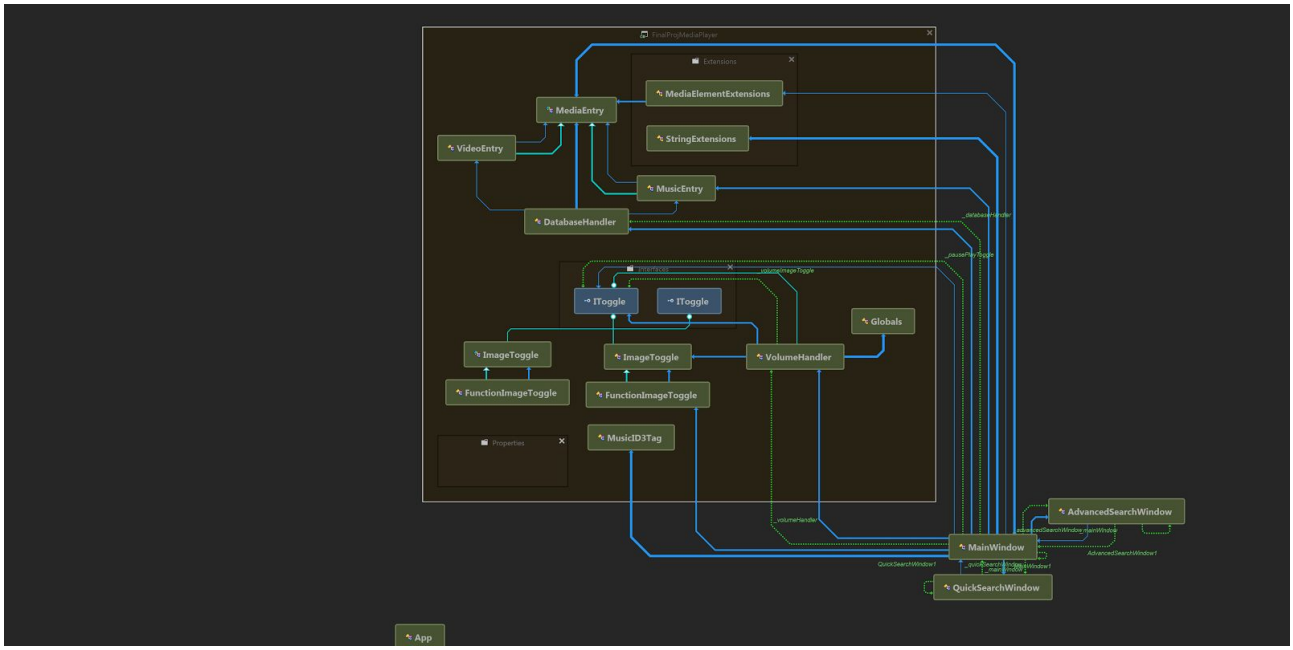
2.6 User Documentation

A ReadMe file will be included with help and contact information as well as brief descriptions of navigation and use.

2.7 Assumptions and Dependencies

This application uses and requires Visual C++ 2013 Update 2 for x64 (required for sqlite, System.data.sqlite dll also included), as well as Windows .NET 4.5.1.

3. System Features



Our system features include:

- Finding media files in windows music and videos folder
- Play, pause, fast forward, rewind to navigate music file
- Volume adjustment and muting
- Playlist reading and executing.

3.1 Finding Media Files

3.1.1 Description and Priority

High priority to find media files in the music and videos folder in order to navigate the media files.

3.1.2 Stimulus/Response Sequences

No user action required other than to have a working windows music and/or video directory

3.1.3 Functional Requirement

REQ-1: User must have a present music and/or video folder
REQ-2: .mp3's and .avi's in the used folder in order to use the media player.

If the player can't find a working music/video directory print an error message but still open the UI.

3.2 Getting Tag Info From Media Files

3.2.1 Description and Priority
Medium priority to gather information from media files (necessary for searching, but not necessarily for displaying or playing media). Goes through file urls in given list and gathers info from tags, including the file's title, artist, genre, and length.

3.2.2 Stimulus/Response Sequences
No user action required other than having files available to gather information from. (Feature will still complete even without files to scan from)

3.2.3 Functional Requirements
REQ-1: Program successfully found directory containing media files & gathered their locations.
REQ-2: Files found are either .mp3 or .avi file

Even if there are no files to gather info from, the UI will still open.

4. External Interface Requirements

4.1 User Interfaces

Main User Interface

4.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

4.3 Software Interfaces

The program will connect to a sqlite database.

4.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

May strain eye sight from over use, eat lots of carrots and take breaks from time to time.

5.3 Security Requirements

Because the program uses a sqlite database, the program will be made resistant to sql injection.

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and

usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: Issues List

< This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.>