**Business requirements and application design**

Whizzy Media Organiser

Logo

Description automatically generated

Martyn Bell

Contents

[Introduction 2](#_Toc88820386)

[Background information 2](#_Toc88820387)

[Conceptual Model 3](#_Toc88820388)

[Use Case Model 4](#_Toc88820389)

[Business Requirements 5](#_Toc88820390)

[Assumptions 5](#_Toc88820391)

[Initial application design 6](#_Toc88820392)

[Manage file screen 6](#_Toc88820393)

[Manage Categories Screen 6](#_Toc88820394)

[Manage Playlists Screen 7](#_Toc88820395)

[Application file data format 7](#_Toc88820396)

# Introduction

Text

Description automatically generated

You work for Whizzy Software, a software house specialising in serving the needs of clients in the media industry, such as TV and radio companies, music streaming services, etc. One of their clients’ needs a stand-alone component that supports the organisation of media files on a device such as a desktop computer, laptop, tablet or smartphone. Your manager would like you to design, build, and test an initial version of this component – the “Media Organiser”.

### Background information

* Whizzy Software develops software components for clients in media industries: music, video, radio, TV etc.
* This project, for one of their clients, is to build a component intended to form part of larger systems which will need to be ported to other platforms.
* The component will be used as a fully-functional prototype that the client can use to test requirements with their own customers.
* The component may run on any execution environment of your choice, on a desktop computer or a mobile computing device.
* The execution environment should support a local filing system that may contain streamable mediafiles (AAC, MP3, WAV, MP4, AVI, etc).
* The component should launch as a stand-alone application in that environment.
* The component does not need to access the network.
* The component should be designed and built to production standards.
* You are advised to get a basic version of the component working first, before adding richer features. Examples of richer features could include:
  + A choice of ways to browse the contents of the organiser.
  + The ability to sort Playlists according to various user-specified criteria.
  + Showing thumbnails of images in the user interface.
  + The ability to change the scope and search again during a session

### Conceptual Model

Diagram

Description automatically generatedThe UML model below shows the main concepts required by the application and their relationships. Data held in the memory of the organiser, and data loaded and saved by the organiser, should correspond to this conceptual model.

* Each MediaFile has a name which is a valid filename (such as “Moonlight Sonata”), a file path (such as “D:\Data\Music\”), a file type (such as “MP3”), and a comment.
* Each MediaFile may be associated with zero or more Categories, each of which is named by a string (such as “Classical”).
* Each MediaFile may be related to zero or one Image, where an Image has a filename and a file path.
* Each MediaFile may appear ordered in any number of Playlists.

### Use Case Model

The UML Use Case model below shows the interactions that a user may have with the media

organiser.

Diagram

Description automatically generated

# Business Requirements

A list of requirements has been generated with a corresponding ID so I can reference during the testing phase of this project.

|  |  |
| --- | --- |
| Requirement ID | Requirement |
| **1.0** | **Specifying scope.** |
| 1.1 | User will be able to specify a directory where the application will search for files. |
| 1.2 | User will be able to specify the file types within the search window. |
| 1.3 | Using the specified location and filetypes the user will be able to import the selected files into a predefined structure within the application. |
| **2.0** | **Load and save internal state of application.** |
| 2.1 | User will be able to save the internal state of the application such as media data, playlist data, category data to a text file. |
| 2.2 | User will be able to load the internal state of the application from a previously save file. |
| 2.3 | The format of the save/load file should be in a format that can be easily used by the application. |
| **3.0** | **Specify Categories.** |
| 3.1 | User will be able to create, delete and rename a category. |
| 3.2 | User will be able to associate categories with media files. |
| **4.0** | **Manage media files.** |
| 4.1 | User can select a media file. |
| 4.2 | User may add, edit, or delete a comment about the selected media file. |
| 4.3 | User may add or change an Image related to the selected media file. |
| 4.4 | User may choose or change the Categories associated with the selected media file. |
| **5.0** | **Make playlists** |
| 5.1 | A Playlist refers to a sequence of Media Files chosen by the user |
| 5.2 | User may create and delete Playlists, change their names, and reorganise their content. |

### Assumptions

* The user will provide media files to the application
* The user will provide category information to the application
* The user will provide playlist information to the application
* The user will want to be able to change the search scopes during runtime.

# Initial application design

The design for this application wants to be clean and not overcrowded for a good user experience.

### Manage file screen

This screen will have a search bar to filter the table as per the entry.

The buttons will be for adding and removing files, when a file is clicked the stored information can be seen and edited within the items on the right-hand side

Diagram, table

Description automatically generated

### Manage Categories Screen

This screen will have a search bar to filter the table as per the entry.

The buttons will be used to add, edit and delete table entries

Table

Description automatically generated

### Manage Playlists Screen

This screen will have a search bar to filter the table as per the entry.

The buttons at the top will be used to add, edit, delete playlists.

The buttons at the side will be used to manage the files and order within the selected playlist.

Table

Description automatically generated

## Application file data format

The application will save serialized data to a text file, this will make the files smaller in comparison to a full data dump.

The file will contain data like the following however will be unreadable by humans.

[ {

Mediafiles [ fileIndex: {File name,

File path,

File type,

File comment,

File categories[],

File image path

},

],

Playlists[ PlaylistName:[(orderNum,

Mediafile

)],

Categories[ Categoryname]

}]