Making IT
good for society

BCS Digital Industries Apprenticeship

Software Development Technician Project A – Media Organiser

Version 1.2
November 2018

Change History

Any changes made to the project shall be clearly documented with a change history log. This shall include the latest version number, date of the amendment and changes made. The purpose is to identify quickly what changes have been made.

Version Number and Date	Changes Made
Version 1.0 May 2018	Document created
Version 1.1 July 2018	Submission email address amended
Version 1.2 November 2018	Declaration template removed. To be supplied in a separate document.

Project Overview and Objectives

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".

You will need to:

1. **Review all the key information and create a design for the media organiser;**
2. **Construct the media organiser in accordance with the design;**
3. **Test that the media organiser meets its requirements;**
4. **Document what you built.**

Project Outputs and Deliverables

Once completed, to demonstrate completion of the tasks you will be asked to provide a series of outputs, that should be submitted together with the synoptic project declaration.

Deliverable	Output	Evidence
Design	Create documentation to describe what the media organiser will do and how it will work. This is likely to include: • Any assumptions made about the requirements or changes made to the requirements; • Sketches of the user interface; • Brief explanations describing what each element of the user interface does; • A specification of the format of data saved and loaded by the component.	Word or PDF document or similar.
Construction	Write a program that implements the media organiser. • Your program should be logically structured; • It should follow good coding practices.	Files containing program code.
Test	Create and execute a set of tests that demonstrate that the program meets its requirements. • The tests may be manual or automated; • The tests may be written before, after, or at the same time as the program code; • For each test, you should document its expected outcome and the actual result.	Any suitable format e.g. textual documents, spreadsheets, program code.
Document	Document the results of your work. • Discuss any limitations of your design and/or implementation; • Propose future improvements; • Create a user guide.	Word or PDF document or similar. A video might be suitable for a user guide.

Project Information and Equipment

To complete this project, you will need to review all the information in the bullet list below. This can be found in the Appendix and will enable you to deliver the key outputs and deliverables for this project as detailed in the table above.

- Background information.
- Conceptual model.
- Use cases.

In addition, you will be provided with access to a virtual platform or alternatively if a virtual platform is not available, your training provider and or employer will provide you with all resources required to complete your project including:

- computer equipment with access to the Internet;
- an appropriate software development environment;
- suitable document preparation software.

Competencies

Below is a list of competencies and knowledge standards covered by this project.

This project covers the following technical competencies:

- Logic: writes simple code for discrete software components following an appropriate logical approach to agreed standards (whether for web, mobile or desktop applications).
- Data: makes simple connections between code and defined data sources as specified.
- Test: functionally tests that the deliverables for that component have been met or not.
- Analysis: follows basic analysis models such as use cases and process maps.
- Quality: follows organisational and industry good coding practices (including those for naming, commenting etc.).
- Communication: clearly articulates the role and function of software components to a variety of stakeholders (including end users, supervisors etc.).
- User Interface: develops user interfaces as appropriate to the organisations development standards and the type of component being developed.

Appendix – Background Information & Business Requirements

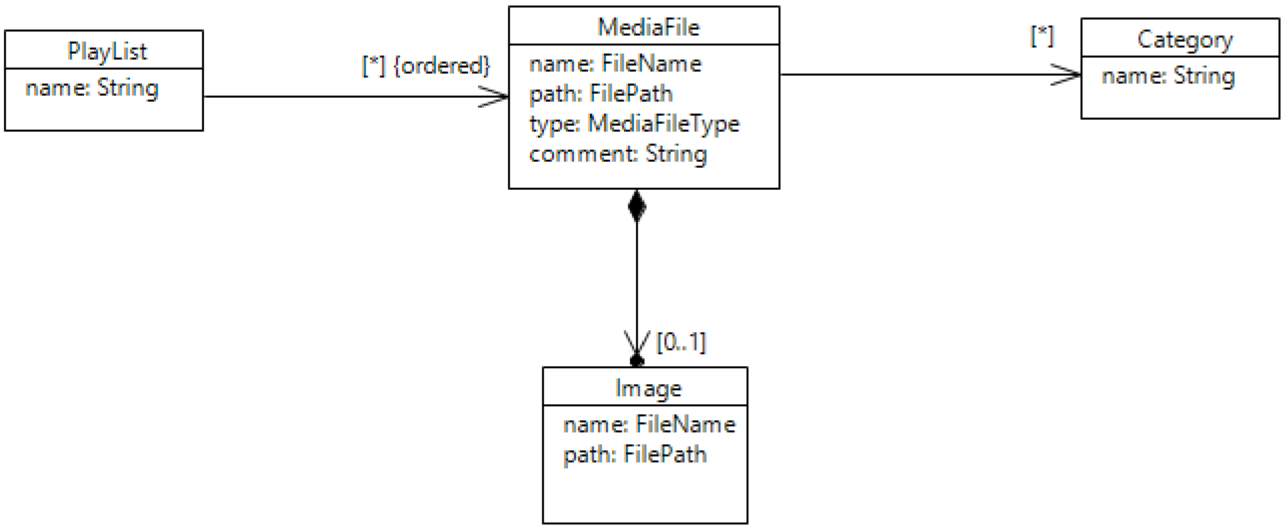
The following additional information has been provided to help you with the completion of the project.

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 media files (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

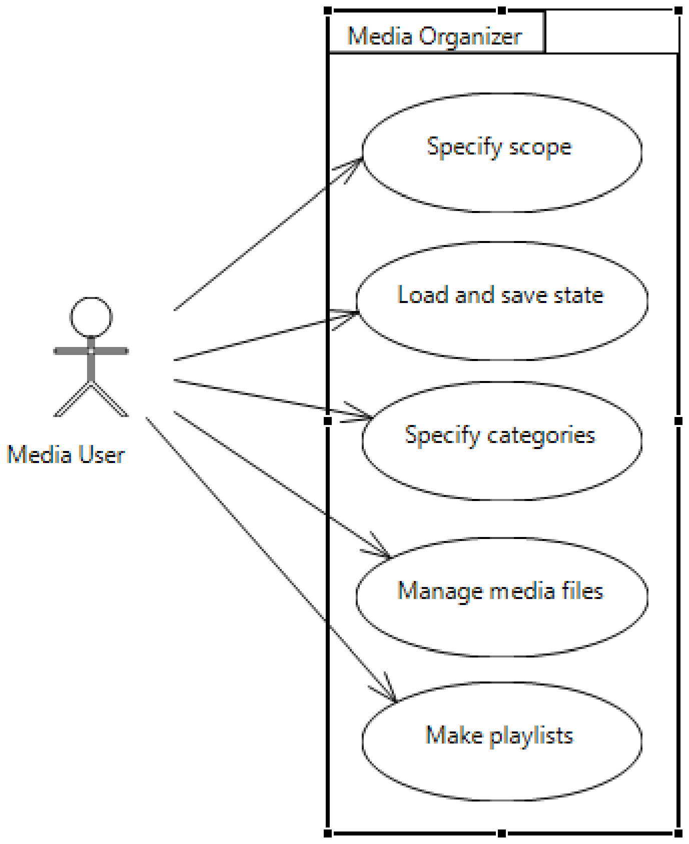
The UML model below shows the main concepts required by the application and their relationships. Data held in the memory of the organiser, and data loaded and saved by the organiser, should correspond to this conceptual model.



- 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.



Specify scope:

- Specify file paths of interest, i.e. the locations in the filing system where the organiser will look for MediaFiles.
- Specify file types of interest, i.e. the types of streamable MediaFiles that the organiser will look for.
- Search for MediaFiles, i.e. within the specified locations, find all the Media Files of the specified types and create corresponding data structures within the organisers internal state.

Load and save state:

- Save the internal state of the organiser (i.e. the set of MediaFile data, Annotations, Image data and PlayLists) in a text file.
- Load the state of the organiser from a previously saved file.
- You will need to design the format of these files so that your program can save and load them easily. You may consider basing it on XML, JSON or any other format of your choice.

Specify categories:

- Allow the user to specify a set of Categories which may be associated with MediaFiles.
- The user may add a Category, delete a Category, or rename an existing Category.

Manage media files:

- The user may select a MediaFile.
- The user may add, edit, or delete a comment about the selected MediaFile.
- The user may add or change an Image related to the selected MediaFile.
- The user may choose or change the Categories associated with the selected MediaFile.

Make playlists:

- A PlayList refers to a sequence of MediaFiles chosen by the user.
- The user may create and delete PlayLists, change their names, and reorganise their content.

On completion, please upload documentation relating to the project deliverables and a completed project declaration (provided separately) to the relevant folder location as specified by your training provider. Alternatively, please send to epateam@bcs.uk