CSC3600 ICT Professional Project

File Metadata Harvester and Searcher

Project Plan

Version No: 3

Document ID: 1.3

Aug 08, 2019

**Team Name**: *USQ Learning Emporium*

**Team Members:**

Jesse Hare W0070949

James MacKeown U1093772

Ryan Sharp U1090072

Richard Dobson U1011744

Vincent Roberts U1046454

Supervisor: Stijn Dekeyser

Project Plan Version: 2

Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Change** | **Date** |
| 1 | Jesse: Created Doc Template | 2/08/2019 |
| 2 | Jesse: Added Content to Introduction | 6/08/2019 |
| 3 | Added content to the Specification | 9/08/2019 |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[1 Introduction 3](#_Toc16246382)

[The Project 3](#_Toc16246383)

[Motivation 3](#_Toc16246384)

[Ultimate Goal 3](#_Toc16246385)

[General plan 3](#_Toc16246386)

[2 Project Specification 4](#_Toc16246387)

[Project details 4](#_Toc16246388)

[Component 1: File Metadata Harvester 4](#_Toc16246389)

[Component 2: File Metadata Searcher 4](#_Toc16246390)

[Problems (Scope, Limitations, Assumptions) 5](#_Toc16246391)

[Assumption 6](#_Toc16246392)

[Objectives 6](#_Toc16246393)

[Expected Outcome and deliverables 6](#_Toc16246394)

[3 Project Design 7](#_Toc16246395)

[4 Work Break-down Structure and Task Scheduling 7](#_Toc16246396)

[5 Time and Cost Estimation 7](#_Toc16246397)

[6 Project Schedule 7](#_Toc16246398)

[7 Risk Management Plan 7](#_Toc16246399)

[8 Code of Conduct 8](#_Toc16246400)

[9 Conclusions 9](#_Toc16246401)

[References 9](#_Toc16246402)

[Appendix 9](#_Toc16246403)

[Contribution Statement 9](#_Toc16246404)

# 1 Introduction

## The Project

This project consists of two parts.

1. Design and Develop a CLI script that takes a directory path as input and returns (on Standard Output) CSV-formatted text having one line per file in the directory, and listing all metadata attributes and values extracted from that file.
2. Develop a GUI that reads the output of the CLI script (CSV file) and display a spreadsheet-like grid, that can be interacted with. Examples of interactions include sorting by specified attributes, searching for file information on multiple criteria and double clicking a file opens it.

The project is to be implemented in either the Python, C or C++ languages, and the desired platform that the components of the project are to run on is Linux, although depending on time constraints we may expand support to other OSes. (Such as Windows or MacOs).

## Motivation

The motivation behind the project is the ‘client’ requires an efficient tool for searching for files and their metadata. We hope to provide a tool that is easier to use and more useful for a user than the usual file manager applications offered with Linux distributions.

## Ultimate Goal

The ultimate goal is to design, develop and implement a quality product that satisfies all functional and non-functional requirements within the allotted time frame.

## General plan

1. Meet with the client to determine all requirements needed for the product.
2. Construct a design for the project
3. Commence development of the project
4. On completion, undergo testing and optimisation
5. Deliver to the client

# 2 Project Specification

## Project details

The file metadata harvester and searcher requested by the client consists of two parts, the metadata harvester, and the searcher. The harvester program searches the operating system and creates the data that is then used as standard input into the metadata searcher.

### Component 1: File Metadata Harvester

The Metadata harvester is a script that is to be executed via the Command Line Interface. It takes as input a directory path, and returns via standard output CSV-formatted text, with one line per file in the directory, listing all metadata attributed and values available for each file.

The Harvester Script should have an option to search recursively – that is the user should have the choice of only searching the folder specified in the file path input, or also all subdirectories below the specified directory as well. An example of this tool being invoked via the CLI is below:



The sample above is an example of a CLI call to the harvester tool to recursively harvest all metadata in the Downloads folder, and output to a csv formatted file.

An abbreviated example of what the contents of the csv file generated would look like is as follows:

*/home/Bob/Downloads/readme.txt,size:400,createdate:2018-07- 27,owner:Bob*

*/home/Bob/Downloads/vids/sample.avi,size:123000,length:1'32",dimensions:240x120*

*/home/Bob/Downloads/songs/jeremy.avi,size:321000,length:3'32",title:Jeremy,artist:Pearl Jam,rating:4*

*/home/Bob/Downloads/pics/20180727102501.jpg,size:89000,dimensions:640x240,camera:nikon,tags:wedding*

The sample output above is based on harvesting file metadata in a directory and its subdirectories, with 4 files in total found.

### Component 2: File Metadata Searcher

The metadata searcher will make use of a GUI and will enable the user to view the metadata for each file in the csv file, which will be organised in a spreadsheet-like grid. There will be an ability to sort and filter the data by attribute values, and a search function which can take multiple criteria. The spreadsheet will have a degree of interactivity, in that clicking on a record for a file will open it.

### Problems (Scope, Limitations, Assumptions)

#### Metadata Harvester

Scope:

* Allow execution from the CLI, searching a specified directory (and subdirectories) for all files, outputting their metadata into CSV formatted text through standard output
* There will be no GUI, will only be accessible via the CLI
* Must be usable on Linux at the least

Problems:

* Need to account for varying file types with different sets of metadata
* Files of the same type may differ in the metadata available for them
* Output must be in a standard format to ensure it is easily used by the Searcher program
* Should certain types of files be ignored?
* Should users only be able to harvest data for files of which they are the owners of? How will this be handled?

Limitations:

* Must be invoked via CLI
* Must only use open-source libraries
* Standard output must be CSV formatted to allow use of output by searcher program
* Must be implemented in either C, C++ or Python

#### Metadata Searcher

Scope:

* GUI program that reads the output of the Harvester script
* Displays the CSV file read in a spreadsheet-like grid
* Will allow following operations:
  + Opening of CSV files from harvester
  + Searching opened CSV files from harvester
  + Display CSV files from harvester
  + Opening files that appear in harvested data
  + Sorting records in harvested CSV data

Problems:

* The harvested data used by the Searcher will have some inconsistencies, such as empty metadata fields, which will need to be accounted for
* How will the files and metadata displayed be organised by default?
* What will the order be for metadata attributes displayed?
* If files are sorted by a field how will others be sorted that do not have a value for that field?
* Will displayed files and metadata be organised by file type in separate sections? Or will they be displayed all together?
* Will the program be invoked via CLI or a Launcher?

Limitations:

* Must make use of a GUI
* Must only use open-source libraries
* Must be implemented in either C, C++ or Python

### Assumption

* Users should know how to use the CLI
* Users should have access to a system that is not headless in order to make use of the Searcher program and it’s GUI
* User will not use the programs for nefarious purposes and will only use them to harvest and search metadata for files that they have permission to access.

### Objectives

* Client satisfaction with all needed functionality implemented on time
* High degree of usability for the searcher and harvest programs
* Program is secure (may involve encoding of harvested data and maybe even requiring special privileges to execute the searcher program)

### Expected Outcome and deliverables

The expected outcome of this project is to provide both components with the full functionality desired by the client.

The specific deliverables will be:

1. The file metadata harvester program
2. The file metadata searcher program
3. Supporting documentation for both components

# 3 Project Design

Notes:

\_ Design specification depends on the type of projects

\_ For example, for a software development project design

{ Functional requirements

{ Non-functional requirements

{ High level (architecture) design

\_ For information system projects

{ Methodology design

{ Data collection strategy

{ Data analysis strategy

\_ The speci\_cation should depend on the agreement within the team and

supervisor

# 4 Work Break-down Structure and Task Scheduling

Notes:

\_ Deliverables and tasks should have reference back to the objectives speci\_ed

in \Project Specification"

{ Break the project plan into phases based on objectives, then

{ Break each phase into tasks, then

{ Break each task into sub-tasks if necessary

{ No need to go to further details

\_ Present the schedule of tasks and deliverables in the table or \_gure form

\_ Present the break-down structure in the table form.

\_ Ideally, the WBS and related chart(s), etc. should be generated using

project management tools, e.g., Microsoft Project.

# 5 Time and Cost Estimation

Notes:

\_ Cost of resources (e.g., hardware, software) required for project implemen-

tation

\_ Time estimation is based on the project plan and should have reference

back to the information speci\_ed in \Work Break-down Structure and Task

Scheduling"

\_ Labor Cost

{ Labor cost is based on the number of team members, roles, and the

time required to complete the project

{ Make up the hourly rates based on the current market standard

# 6 Project Schedule

Notes:

\_ GANTT Chart of the project schedule

\_ Speci\_ed based on the agreement within the team and the supervisor

\_ Specifying all milestones (the project lifecycle should be within this semester)

# 7 Risk Management Plan

Notes:

\_ Specify and describe all potential risks

{ risks to the completion of project, e.g.,

\_ Project delay

\_ Loss of team member

\_ Loss of equipment

\_ Loss of data

\_ Loss of necessary services

\_ etc.

{ risks to professionalism and professional ethics, e.g.,

\_ leak of customers' privacy

\_ leak of client's business con\_dentiality

\_ leak of intellectual property like system design

\_ etc.

\_ Specify the risk with Probability (in percentage), Impact to the project

(in percentage), and Status (in High, Neutral, or Low Risk)

\_ Describe the risk management plan to deal with EACH of the speci\_ed

risks

# 8 Code of Conduct

Notes:

\_ Team principles

{ The principles of behaviour, communication, operational processes,

and professional ethics that the team agrees to abide for.

{ Principles should be high-level statements that describe what your

team considers to be the key values, beliefs and norms that contribute

to an e\_ective team environment;

{ Justify each of the principles for why all team members need to follow

it;

\_ Team communication and operational process

{ Operational processes should show how the principles are to be applied

to the daily operations during the project.

{ Provide at least 4-6 examples of communications and operational pro-

cesses that your team has agreed upon.

\_ Professionalism and professional ethics

{ the way that the team deals with customer privacy, business con\_-

dentiality and intellectual properties

{ the way (e.g., attitude) that the team deals with stakeholder, such as

client(s), customers, advisor(s), etc.

{ the way that the team members treat each other;

{ the way that the team deals with data;

{ use of development tools and resources, e.g., licensed vs. pirate soft-

ware, unauthorised resources such as data, images and sample prod-

ucts, etc.

{ provide at least 4-6 examples of professionalism and professional ethics

that your team has agreed upon.

\_ Non-compliance

{ Record your team's agreed de\_nitions of minor non-compliance (in

a way that may adversely a\_ect the project). You should provide

samples to help clarify the de\_nitions.

{ Record your team's agreed de\_nitions of major non-compliance (in a

way that has a major negative impact upon the team's success). You

should provide samples to help clarify the de\_nitions.

{ Separate the de\_nitions to initial and basic rules; minor and major

transgressions

Team name Page 8

Project Plan Version: version number

\_ Dispute resolution and conict management

{ State how your team has agreed to resolve issues like minor breaches

of this Agreement.

{ State how your team has to resolve issues like major breaches of this

Agreement.

# 9 Conclusions

Notes:

\_ Summarize the information presented in the document

\_ Make the conclusions to the document

# References

Notes:

\_ List bibliographies that are mentioned in the document or may support the

information in the document.

# Appendix

# Contribution Statement

The undersigned members of this team agree to abide by this project plan to

ensure the successful completion of the project. The members also agree that the

contribution percentages specified below reflect the true level of contributions

made by each of the members to the works reported in this document.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Student Name | Student ID | Signature | Contribution % | Date |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |