CIS 227 Assignment 2

Assignment Details

* Use at least one class.
* Create a program to read inventory from a JSON, or CSV file.
* Create a “login” process. – Use clear text passwords and a CSV file to store the information.
* Prompt the user to search the “database” for titles and display the following information:
  + Book Title
  + Book Author
  + Publisher
  + Book Publication Year

Team Roles

Lead Programmer – Jacob Wiles

UX/UI Programmer – Michael Dolan

Functional Programmer – Christopher Rodela

Program – 70

UX/UI – 35

Function - 35

Documentation – 30

Total Possible Points – 100

**Version 0.0.0**

| REVISION HISTORY | | | |
| --- | --- | --- | --- |
| DATE | VERSION | DESCRIPTION | AUTHOR |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# INTRODUCTION

## PURPOSE

Identify and describe scope of product whose technical specifications are being documented and describe desired outcome.

## DOCUMENT CONVENTIONS

Describe any naming or structural conventions employed throughout document and how they benefit reader.

## REFERENCES

List any referenced document names or links.

# DESCRIPTION

## FEATURES

List main features with brief description.

A login

A file reader

A display

## USER OVERVIEW

Define groups and describe user characteristics.

Bookstore workers &/or owners using a Windows PC that need to maintain an inventory for a bookstore.

## ASSUMPTIONS / DEPENDENCIES

Detail all assumed factors (not known facts) that could potentially impact technical specifications set forth. Include external factors. (What other things do users have to have in order to operate this application?)

The user is going to be using a Windows PC.

This is going to be a console-based application, so the user will have to have access to the Windows command prompt. This accessibility requirement is referring both in terms of having the appropriate user permissions within the system as well as having and being able to operate a functioning keyboard or a suitable equivalent that can interact with the command prompt.

(The user will need to have either the RapidCSV or JsonCPP libraries installed onto their machines, depending on which one we use.)

# SYSTEM FEATURES

## SYSTEM FEATURE 1: The Login Process

|  |  |
| --- | --- |
| **DESCRIPTION AND PRIORITY** | The user must enter a username and password in order to log in and utilize the system |
| **STIMULUS / RESPONSE SEQUENCES** | First there is a prompt for the username and then it will check it against the database. If the username is found, then it will prompt for the password. If the username is not found, then it will return the message “Username not found.” |
| **FUNCTIONAL REQUIREMENTS** | (Do we want it to check for username first and then respond with “Username not found”? Or do we want to take both the username and password at the same time and say “The pair is not found”?) We need a user database/file using either .JSON or .CSV file format. (We need to turn this “database” in along with the rest of our other files.) (The above requirements only say to use .CSV for the username “database” instead of giving the option of either .JSON or .CSV. I’ll need to email Prof. Carmon to ask him about that small detail, if it needs to be a .CSV or if it really can be either one.) |

## SYSTEM FEATURE 2: Importing and Reading from .JSON and/or .CSV Files

|  |  |
| --- | --- |
| **DESCRIPTION AND PRIORITY** | (Are we using the .JSON or the .CSV or both?) |
| **STIMULUS / RESPONSE SEQUENCES** | (How do I know that the user wants to import this file? Are we going to do right away on load, or are we going to wait for the user to say. “Import the file? Yes or No?” Or are we going to do it automatically? As in either after the log in process is complete the program will automatically import the file to be ready, or are we going to do it after they have entered in their input into the search query, if either of these then probably the later. Or will we possibly give the user the option of loading in either file, or even both files, to give them the option of which to search through, though this would definitely be more of a bonus if we could do that this week.) |
| **FUNCTIONAL REQUIREMENTS** | The JSON file to be imported, or the CSV file to be imported, (depending on which we use). |

## SYSTEM FEATURE 3: Searching/Querying Through the Book Inventory File/Database and Displaying the Relevant Search Information

|  |  |
| --- | --- |
| **DESCRIPTION AND PRIORITY** | (Currently, the books.json and books.csv only tell us and the users “What books do we currently carry,” and not “What books do we currently have in stock.” Implementing this change might, and probably will, be something that will be required to change later down the line. This isn’t something you need to work on making this week, but it is something worth considering while you try to design this.) |
| **STIMULUS / RESPONSE SEQUENCES** |  |
| **FUNCTIONAL REQUIREMENTS** |  |

# REQUIREMENTS OF EXTERNAL INTERFACE

## USER INTERFACES

Describe product / user interface characteristics, including standards, style guides, constraints, functionality, and sample screens if applicable.

This program is going to be a console application, which will require suitable user permissions to access and interact with the Windows command prompt as well as a functioning keyboard or a suitable equivalent that can interact with the command prompt. A mouse or a mouse equivalent will not be required for the functional operation of this application but having such equipment will allow the user an additional way to be able to close the program prematurely at their own discretion.

(The user will need to have either the RapidCSV or JsonCPP libraries installed onto their machines, depending on which one we use.)

# APPENDICES

## APPENDIX A: GLOSSARY OF TERMS

Define all terms and unique acronyms employed throughout document and specific to project.

## APPENDIX B: ANALYSIS DOCUMENTATION

List file / document names / provided links to all diagrams, models, additional findings pertinent to technical specification development. (Basically, just about any other documentation that we make for this project should be included here. Also include any files/filenames that would be required to compile & run our program that we’ve written ourselves, so our .cpp’s, our .h’s, and any .json and .csv that we’ve made should also be listed here, but do not include any of the files that were given to us as part of the assignment, i.e. the “books.json” and “books.csv” that are the inventory files that we were given.)

## APPENDIX C: ISSUES

List all unresolved issues, TBDs, pending decisions, findings required, conflicts, etc. (REMEMBER: If Professor Carmon runs into a bug in our application and he doesn’t see it listed and documented here, it will count as an automatic 0 points for the total “Functionality” portion for this turn in. So TEST TEST TEST!!! Everybody, test your code and communicate any problems you find! And the Lead needs to test ALL the code and ask if anyone knows of any problems with their work.)

| ISSUES | | |
| --- | --- | --- |
| ID | DESCRIPTION | PARTY RESPONSIBLE |
| 0 | Whenever the program asks the user for input, they can press the enter key while no other input is given to move on to the next line in the command prompt window. This makes it so that the user can keep adding new lines in the command window to offset our menu from its intended position. This does not affect the functionality of the program in any way, but it does affect the possible viewing experience of the user. Recommended solution involves implementing cin.get() to replace the cin << endl;. | The currently next UX/UI Programmer (Jacob W.) |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |