**CST2550 Software Engineering Management and Development**

**Coursework 1 – Library System**

**-**

**Student Name: Raisa Silea**

**Student ID: M00854834**

**UML Diagrams**

Before I started developing the code for the library system, I designed the use case diagram and activity diagrams for my system. This has helped me plan my code and visualise how the system would or should work.

A diagram of a diagram

Description automatically generatedUse case diagram:

The diagram shows that the librarian will be the one to mostly interact with the system, as this program is intended for the librarian to use. They would be able to sign in, add a library member and borrow or return books to registered members.

**Code Structure**

I chose to make separate header (.h) files for each code class, and then individual source (.cpp) files for their corresponding implementation of constructors and other functions. This makes my code modular and easy to debug and each part can be easily tested. Additionally, doing it this way made it very easy for me to develop the code as at no point was I confused where things were or where the source of an error was.

All classes follow the provided UML, with no deviation from the system requirements.

I developed the CSVParser to take any csv file that includes title headers as the first line. Therefore, the code will always remove the first line of the CSV file.

The library.cpp file holds the menu of the program, where the user is able to input their choice of task they’d like to carry out. In this file, the inputted file name is parsed using the CSVParser and then stored in an unordered map of key value pairs. This stored the books in the csv file by their number iD.

**Limitations**

One limitation in my design was that I could not include the use of the page count and book type, which were provided in the CSV file. This is because the UML did not specify a setter for that information, and the class constructors also didn’t take that data. Therefore I have not implemented their use in my final program, however, as it can be seen from the below screenshot, I have provided commented out example how those would be taken from the CSV file and could be added to the unordered map as one of the values.

A screenshot of a computer program

Description automatically generated

Another limitation I encountered was proper implementation of user input validation. This is because I did not add it in during the initial development, therefore when attempting to add it in later, it caused errors such as not saving input, or not allowing any other validation to work. (I’m sure it could’ve been solved if I had spent more time fixing it)

A final limitation was the implementation of my makefile. This is because during implementation, the CXXFLAGS = −g −Wall −Wextra −Wpedantic started giving me errors when I was running the make file. I then used this example (262588213843476, n.d.) to create a makefile which automatically takes all files and compiles them into object files. It uses the C++17 standard which seemed to work fine with my code.

**Testing**

I have created Catch2 test cases for the book, librarian, member, person and date .cpp source code files. I have read several online materials to understand the best way to do this, one of the most useful to me being (Bancila, 2018)

For some of the files, such as for “member.cpp”, I have created mock up classes to facilitate my testing. For the “date.cpp”, I have manually created date strings in order to test late days and correct fine calculations.

I have also tested that my program can accept any CSV file, regardless of file name. I tested this by manually renaming a copy of the provided CSV file.

**References**

262588213843476 (n.d.). *Compile all .cpp files into one target under the current directory.* [online] Gist. Available at: https://gist.github.com/wenchy/64db1636845a3da0c4c7 [Accessed 10 Jan. 2024].

Bancila, M. (2018). *Writing C++ unit tests with Catch2*. [online] mariusbancila.ro. Available at: https://mariusbancila.ro/blog/2018/03/29/writing-cpp-unit-tests-with-catch2/ [Accessed 15 Jan. 2024].