Library Software: CST2550

Jakub Sabat M00904365

January 2024

**Contents:**

1. Introduction
2. Software Design
3. Software Testing
4. Implementation
5. Conclusion

**1. Introduction**

This report outlines the development process of a Library Management System, designed to align with a provided UML diagram. The system interfaces with the user through the console, allowing them to perform library tasks such as: adding a user, issuing a book, returning a book, listing borrowed books and calculating fines.

A screenshot of a computer

Description automatically generated**2. Software Design**

My software was based on the provided UML diagram, I renamed the variables for ease of use - other than that, the classes were strictly kept the same as shown on the UML. I added a struct called Date as this was a datatype present in the provided UML. User input and other helper methods were coded in utils.cpp. My interpretation of the UML suggested that I needed some global variables or a class to hold the members and books, I chose to go for the global variable inside of utils approach as it was simpler.

**3. Software Testing**

To test my software, I used manual and automatic testing, during development I used manual testing using breakpoints and user input from myself. Such testing using breakpoints makes it easy to see where errors may form as it a direct view into the stack.

I applied automatic testing by using catch2 as a single header include, my test cases were written after my main code was complete. Some modifications to the main code had to be made, for example the passing pointers to std::cout and std::cin so that I can include simulated user input and output in my testing.

**4. Implementation**

During the implementation phase I started by prototyping the class headers based on the provided UML. After I had the class headers setup, I began programming the methods. As I was coding in Visual Studio, I didn’t need a make file to begin with, once I was happy with my solution, I started writing test cases to test the methods.

**5. Conclusion**

This project culminated in the successful development of a Library Management System guided by the provided UML diagram. In terms of limitations, the biggest one was having to work around the pre-defined classes, as I couldn’t implement the Library Management System in the way that I envisioned it working. In other projects, I would prefer to use class based encapsulation instead of global variables, but I wanted to stay safe with sticking to the UML. Another future improvement could also be to write tests first and program around them to save time on manual testing.