Final Project Report

Martin Barcelon

10250249

12/09/17

ESE-224.01

Final Project Report

The Library Management System is a C++ program that consists of three .CPP files, two .h files, and four .txt files. The main file, main.cpp, starts the program off when the user launches the .exe program. The header files provide the class definitions for the respective .cpp file. The four .txt files provide the storage for the user information and book information. Since the main file is where the program starts, the main file in imports the header files from book.h and user.h as these files provide most of the functions for the program. One should note that by putting class declarations in different files, the program becomes much more modular and easier to manage in design. For example, is something is wrong whenever I add and new book to the library, it is easy to identify where the problem is, since the book.cpp file houses all the functions for books and copies.

Starting with the main menu, the user is prompted to either login as a user or login as a librarian/administrator. Also, here the user can exit the program by entering 3. To accomplish most of the menus' in the program, a simple do-while loop will allow for the user to keep inputting information to the program without ending the program unless specified.

The user.cpp file contained the class implementations mainly for the reader object, the admin object (librarian). In the header file, we can see that user implements the user class, the reader class and the admin class. Each of these class declarations have their accompanying functions. To pass variables that are universally shared between the three classes, such as password and username, the variables are initialized in the user class and the other classes can be created as subclasses by adding ":" in between the declaration and "public user" thereby

inheriting the variables from the user class. Likewise, the book class uses the same method to inherit variables for the copyB class.

To store data, the program uses .txt files like book.txt, user.txt, librarian.txt, and copy.txt. For the program to use these files, the program undergoes a three-step process in this order:

First	Second	Third
Load the file	Change/manipulate the file	Update/save the file

One can see that both the .h files contain functions named load_file() and update_file() which accomplishes this task. By doing this, it allows the program to use the data without having to call the data file every time it is used in a method. It also saves time writing methods to write and access the files in smaller, unnecessary tasks. The init_file() functions as memory wipers, which only can be used by the admin profile, as they reset all the .txtfiles (except libarian.txt) to the original states.

Each class carries a few constructor classes along with them. For example, the book class uses the load_book() method to load data into the object. It also contains accessor methods which pull variables by returning them from the object header file. The default constructor initializes the variables for book class to ISBN = -1, title and author to "NULL", category is -1, and the array copiesQ and reserve to -1 and "NULL" respectively. The category is enumerated and each time the user searches or makes the category for a new book, they are given a chart to use for the numbers. The copyB object by default has its ID to -1, borrower = "NULL", date = -1, resdate = -1, and exdate = -1. The user class initializes its variables to "NULL" for name and password, maxCopies to 10, penalties to 0, boolean teacher value to false, the bag vector (which carries the

copy of the books) to -1 and the reserved vector (which remembers what reserves were made) to -1. Once can see how the variables were stored in the .txt files, which can be shown here:

```
Mart    Barc    1       0       0
-1      -1      -1      -1      -1      -1      -1      -1      -1      -1
-1      -1      -1      -1      -1      -1      -1      -1      -1      -1
NULL    NULL    0       10      0
-1      -1      -1      -1      -1      -1      -1      -1      -1      -1
-1      -1      -1      -1      -1      -1      -1      -1      -1      -1
```

username	password	Teacher?	maxCopies	penalties					
Copy[0]	Copy[1]	…	…	…				…	Copy[9]
Reserved[0]	Reserved[1]	…	…	…				...	Reserved[9]

Lastly, each class has specific important functions to the program. The reader and admin classes contain the functions for menus and login for each class. The menus from the admin class allow the librarian to search the exiting library for specific tags, add new books to the library, delete existing books from the copy class, search for users in the system, add new users into the system into the add_reader() function. Delete users from the system using the delete_user() function, see the admin info and change the password for the administrator profile. Note it doesn't effect, the TA224 profile. The reader class is nearly identical for students and teachers. The important thing here is that the function allows for users borrow books using the borrow_copy() function. This function is important because it allows the user to record that certain copies are taken out. For example, if there was a book that I borrows, the name of its borrower variable would change to my username, signifying I have borrowed it. One can see that changes in the .txt files before and after a borrow. Each book class has a reservation list which can be modified using the reserve_copy() function and the cancel reservation function removes

that reservation from the que in the book class to. To do this, it seeks the username it needs to remove and then shifts each vector one spot towards the left and adds the “NULL” statement towards the end if it finds the username. One can see how the books are reserved and taken by monitoring how the .txt file changes or using the search book function of the program.