**Name: Anas Saad  
Matric number: 2510059  
Module code: AC21008**

compile command: g++ lms.cpp

**Program Structure:**

It’s split into a Book class which manages Book objects, a User class which manages User objects, and a Library class which manages the whole library also in relation to the Book and User classes. There are some helper functions that don’t belong to any class with are (displayMenu, processMenu, and testScenarios).

**Book Class:**\* The Book class has attributes: title, author, ISBN, and availability.\* It has a default constructor and a parametrised constructor. **\*** It also has getter and setter functions to access private attributes or to update them.

**User Class:**\* The User class has attributes: ID, name, and borrowedBooks.\* It has a default constructor and a parametrised constructor.\* It also has getter and setter functions to access private attributes or to update them. **\*** It has functions like borrowBook, returnBook, and displayBorrowedBooks that manage the user’s books.

**Library Class:**  
\* This class is the main class in terms of managing the system.  
\* it contains a list of books and a list of users to keep track of all books and users.  
\* It includes the functionality to add and remove books, to search and display books, also the ability to add and remove users.  
\* It has also allows for saving the state of the library or loading in it from a files.

**Helper functions:  
\*** displayMenu: displays the system’s menu.  
\* processMenu: deals with the user’s choice and uses functions from the Library class.  
\* testScenarios: executes some scenarios to test the system.

**Main Function:  
\*** The main function creates a Library object and uses the processMenu() function to start the system. **\*** it also has the testScenarios() function to test some scenarios.

**Class Relationships:**\* The Library class deals with the Book class and the User, as it manages the whole system. **\*** The User class contains a vector of Book objects to represent the books borrowed by a user.

**Functionality:**

**Expected functionality:  
\*** Add books  
\* Remove books  
\* Search for books  
\* Borrow and return books  
\* Add users  
\* Remove users  
\* File I/O  
\* Error handling  
\* Managing different items  
\* Item hierarchy  
\* Flexible user class  
\* Use of templates  
\* Late fee calculation  
\* Advanced search capabilities  
\* Item reservation system

**Implemented functionality:  
\*** Add books  
\* Remove books  
\* Search for books  
\* Borrow and return books  
\* Add users  
\* Remove users  
\* File I/O  
\* Error handling

**Design Decisions:**

**Data Structure:**   
Vectors: The programme stores user profiles and book collections dynamically using vectors. Vectors are user-friendly and flexible.

**File I/O:**  
Files for Inventory and Users: Data persistence is accomplished by keeping track of books and users in different files (inventory.txt and users.txt). This guarantees that the state of the library may be loaded and preserved across sessions.

**User-Book Interaction:**  
Division of Responsibilities: The Library class oversees user-book interactions. The Library class's borrowing and returning procedures update the user's list of borrowed books as well as the availability of the book.

**Interface User:**  
Menu-Driven: The application uses menus to communicate with users. This offers an easy-to-use interface by giving users a selection of alternatives for various actions.