Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

CPSC 471: Database Management Systems
Library Management System

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

Table Of Contents

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

1. Abstract

The efficient storage and management of information is one of the most vital yet commonly underestimated aspects of technological advancement. Record-keeping can be traced back as far as 5000 years ago to the ancient civilization of Sumer, and as far as historical findings suggest; the practice hasn't geared down in any way since. As technology advanced, so too have the systems used to keep track of large sets of information. However, due to limited resources or devastating conflicts, these advancements have been significantly held back in certain regions. While the majority of the world's population is raised alongside advanced electronic systems and devices, some are left behind; falling further and further behind the technological curve. However, with increased efforts toward developing straightforward and easily implemented electronic systems, this gap in knowledge can very easily be bridged.

2. Introduction

Library Management Systems have evolved significantly over the years with the growth of the Internet. These systems have also become increasingly difficult to use because of the breadth of functionality that they offer to their users. This kind of rich functionality is a benefit for countries that have a small digital divide (a larger part of the population familiar with technology, as second nature), however, there are still countries where it is fairly difficult to find staff who can use these systems for a lower cost of salary. These are the issues we had in mind when developing our program; we intended to build an interface that is lighter and a much simpler option for people who may not know much about electronic systems to use with ease. With our system, we wanted to make a program that can be used to obtain, store and process large sets of information about all books that a library possesses (because even small libraries may have thousands of books). Furthermore, we wanted a system capable of holding and navigating through not only information about all books, but also the necessary information for people who borrow them. Most countries where technology is more of a privilege have libraries still using card systems to keep track of the books they hold, and how many they have lent; thus our aim is to improve their library systems with as much ease and affordability as possible.

Keeping these problems in mind, we have created a database that solves the problems described above. Our database is useful in helping libraries migrate from the traditional card system of keeping track of borrowers and books that they have borrowed to a centralised, electronic system that stores all this information. Alongside this, our system shows users the books available for them to borrow from the library and gives them the ability to reserve study rooms if there are any available. Among the members of the library who can borrow books, we also have administrators who manage all users of the system (that have lower system privileges than themselves) as well as librarians who keep track of books borrowed by members. The functionalities each of the users has to their access are described in more detail in this report.

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

3. Functionalities available to different users of the system

3.1 Administrators

These users have access to the back end of the database, but may also use the front end of the system to carry out their duties in a library. As an Administrator, they can search information about users as well as update their personal information should there be any changes. The Administrators are responsible for adding users to the system, where they can choose whether the new user is a member of the library or librarian of the same. Depending on the type of user created, they have a specific set of tasks that they can carry out within the system. Administrators of the system can also manage users of the system, so if a user is no longer a stakeholder of the library, they can easily remove these users from the system.

3.2 Librarians

Being the users who will manage the library on a day to day basis, librarians have a more comprehensive set of tasks they can do to manage books in the library more conveniently. These tasks include things like managing inventory which is used to keep track of all the items that the library owns (as well as where these items are located in the library). Librarians are responsible for adding items to the inventory as well as updating information about the existing items in the library. If the item being added to the inventory is a book, the librarian can use the item id they defined when they were adding an item to inventory and assign it to a book that the library holds or has recently acquired. This way the librarian can keep track of how many books there are in the library and where exactly they are located.

Alongside management of the inventory, the librarian can use this system to search for books that the library owns and whether or not these books are lendable to members. The librarian may, at the click of a button, access the full list of books that the system holds alongside the item ids of these books to track their locations. Additionally; using the ISBN of the book they may also see how many copies of this book are available.

Librarians can also search for a member in order to obtain their contact information if they have to (for any reason). The librarian can look at the number of books that are borrowed by a member that have not been returned yet to make sure that the member can only borrow new books after having returned the old ones. When lending books, the librarian can also specify how long the member is allowed to borrow a book for so that they can keep track of whether the book was returned on time or otherwise issue them a fine as a result. Incase the member needs an extension, the librarian can extend their duration of having the book borrowed as well as use this functionality to add previous records of books borrowed incase the library has recently migrated from using a card system to ours (or incase both are simultaneously used together).

The librarians can add information about publishers of books they have in stock. In fact, they can use these details about the publisher (linked to the book when the book is being added to the inventory) to keep complete information about a book incase more copies are needed.

3.3 Member

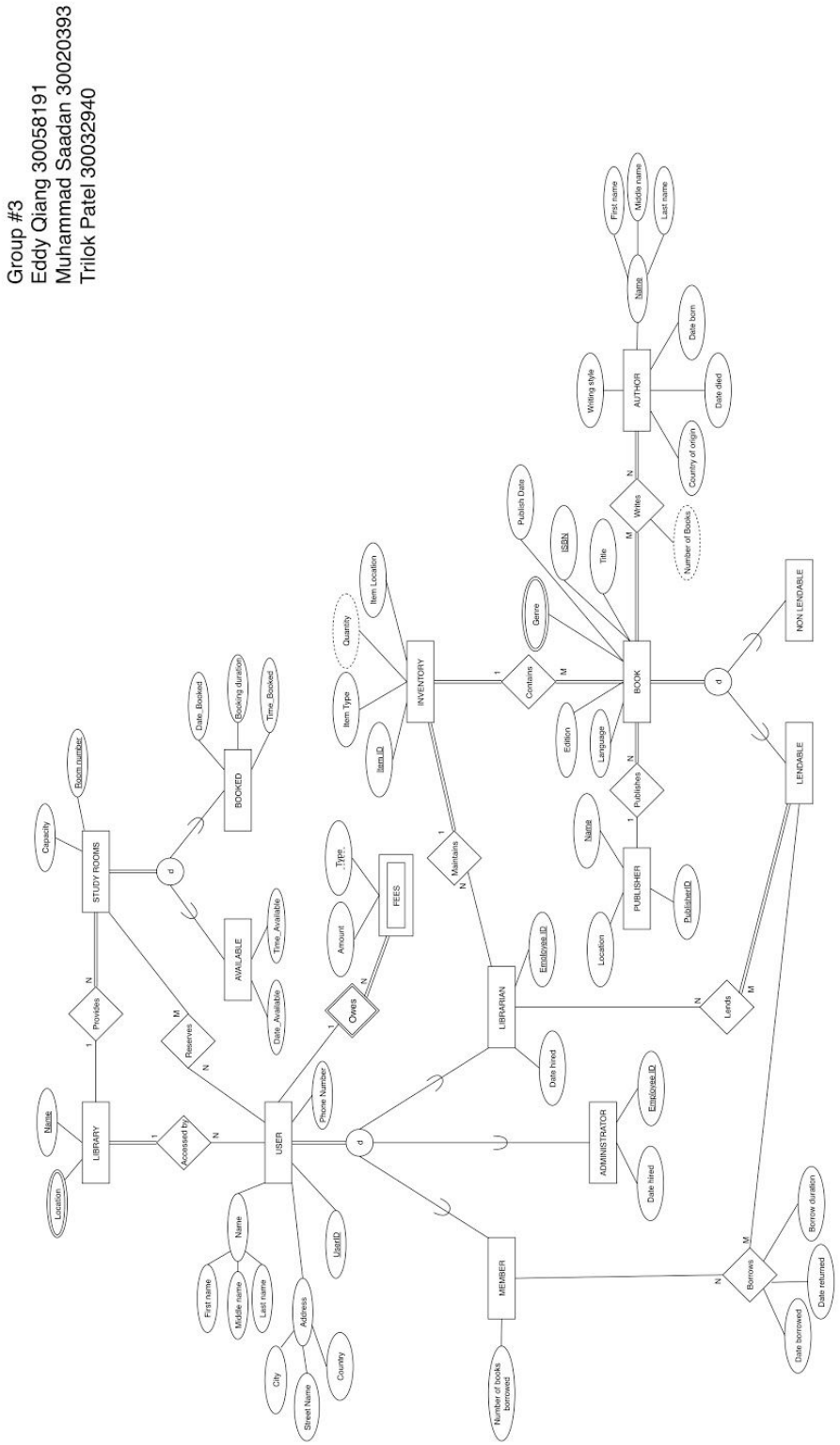
As a member, one can search for books that are available to borrow from the library based on their ISBN. Incase they don't know the ISBN of the book that they want to borrow, they can also view the whole list of books that are available to borrow and obtain the ISBN of the book that they intend to borrow as well as see whether or not this book is available to be lent.

Alongside borrowing books, incase there are any study rooms that are available to use, the members of the system can use it to book these rooms as well as view their reservations incase they forget when or where their reservation was placed for.

All the users can only access this system through their user id and the password that has been issued to them. It is through the Administrator of the system that these users (the members and librarians) can obtain these credentials and otherwise can not use the system.

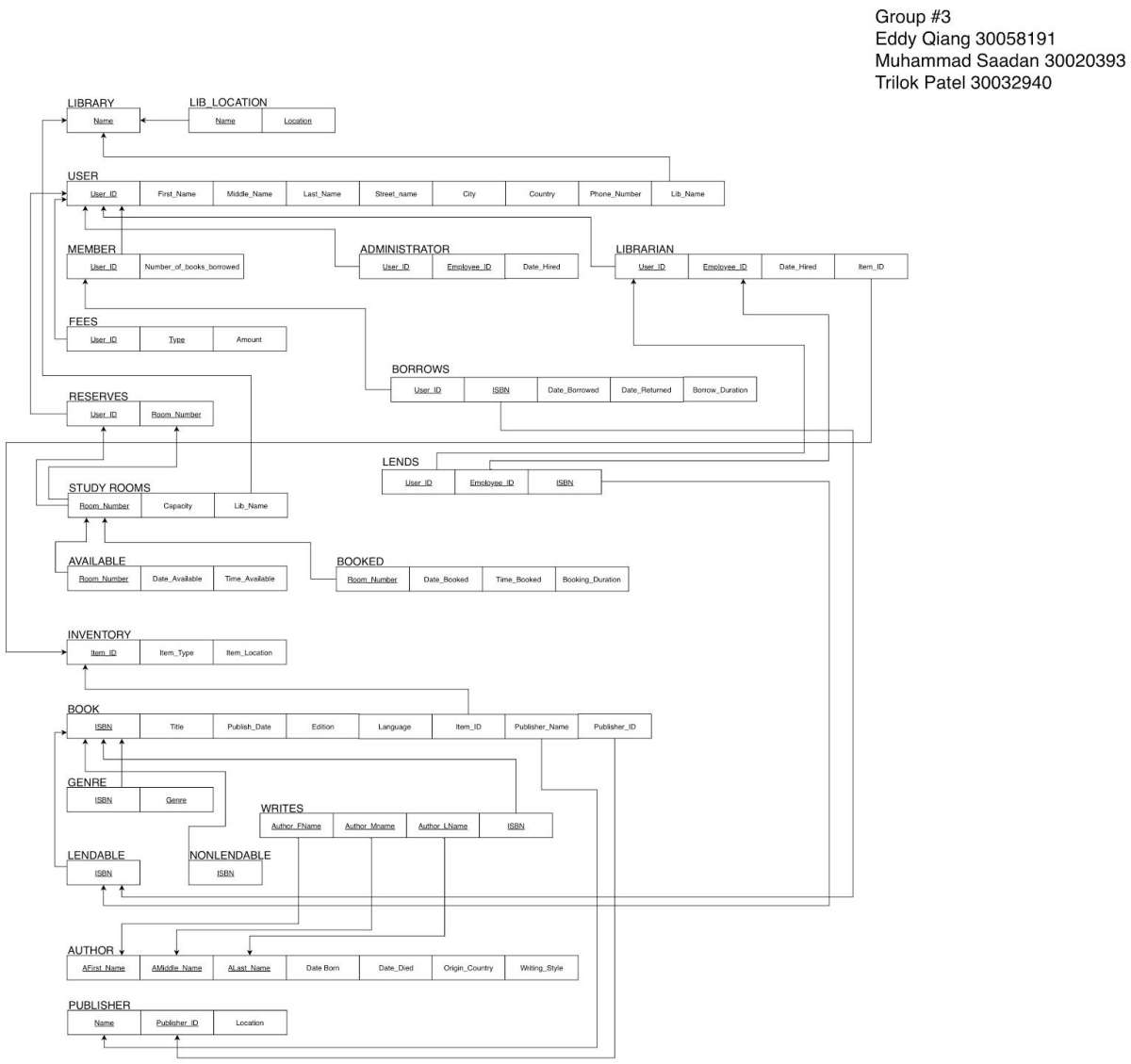
To see how these different users can access the functionalities, please see the User Manual in this report that describes in detail how to use the system in order to carry out specific tasks.

4. EERD of the Library Management System



Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

5. Relational Model of the Library Management System

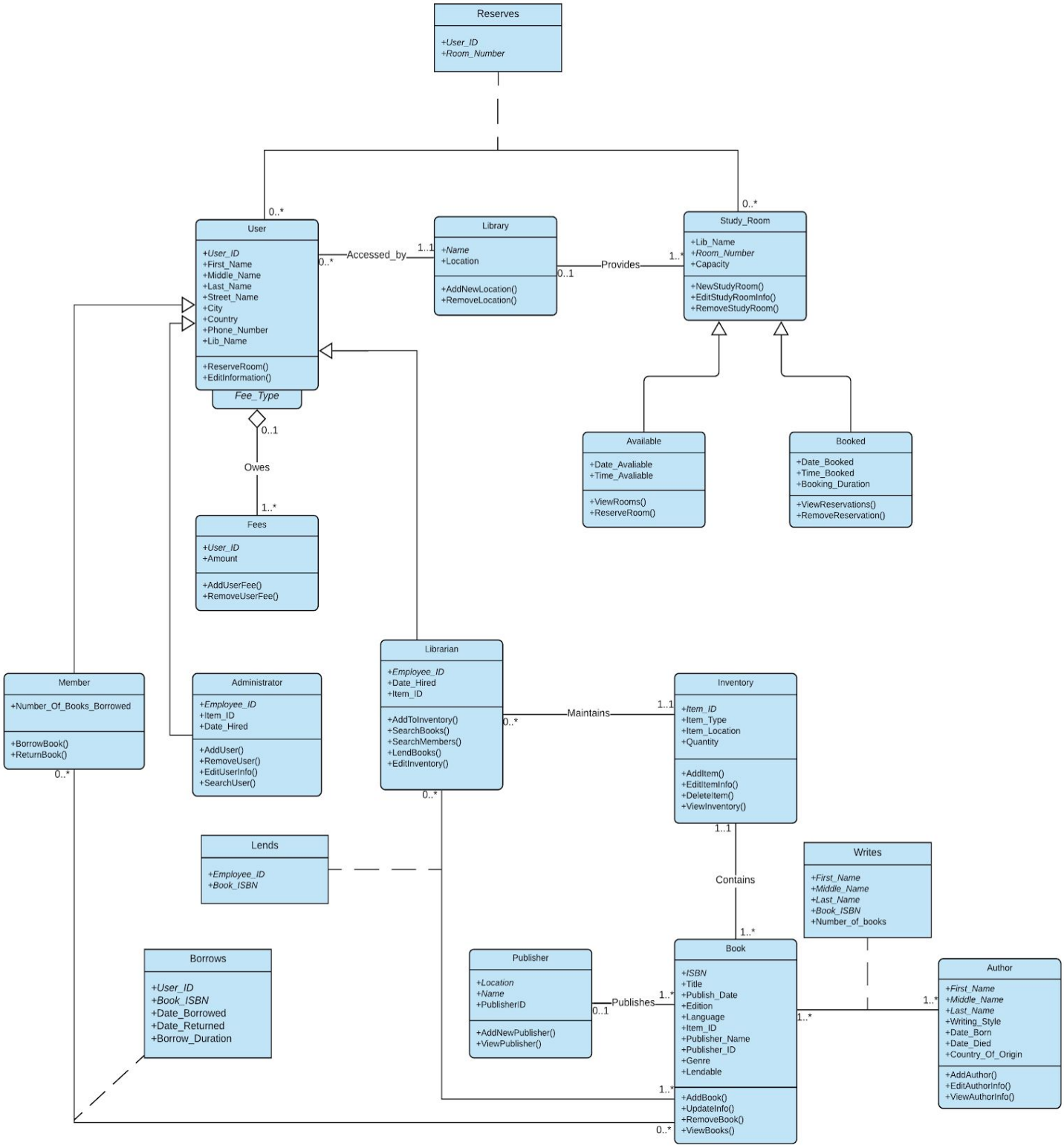


6. Implementation of the Library Management System

This system was implemented using mySQL and PHP. The following sections of the report highlight important details about the implementation of the system as well as the SQL queries that were used when implementing the database of this system

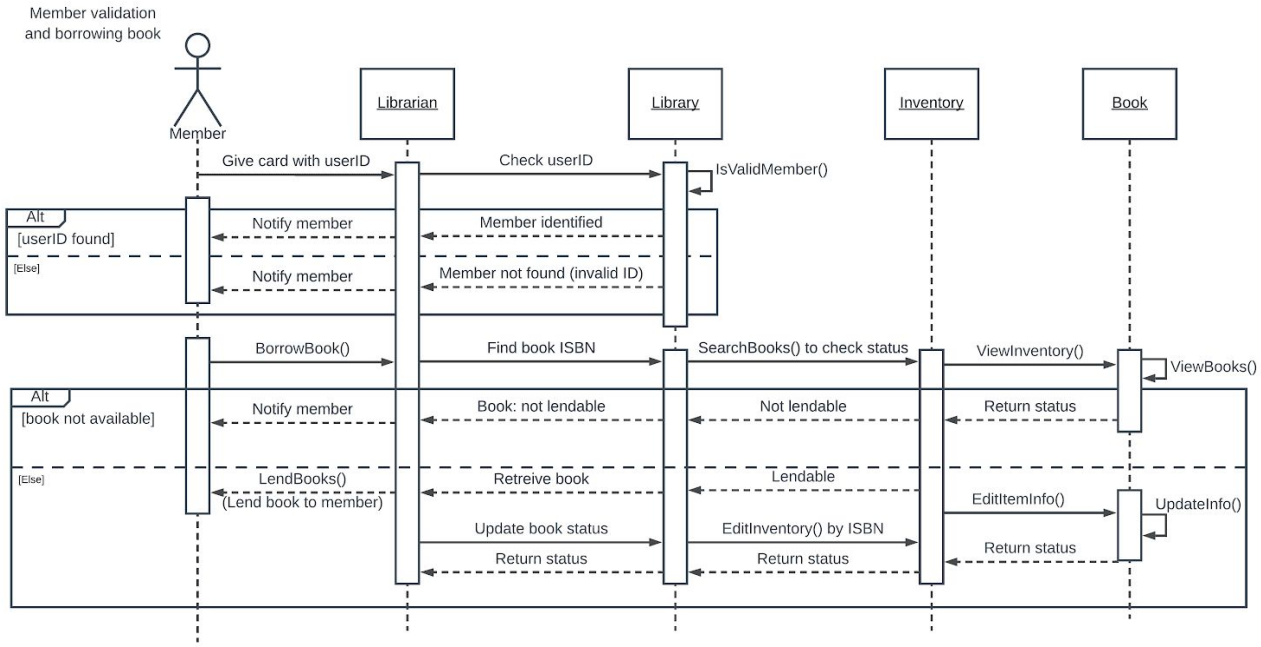
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

7. UML Diagram

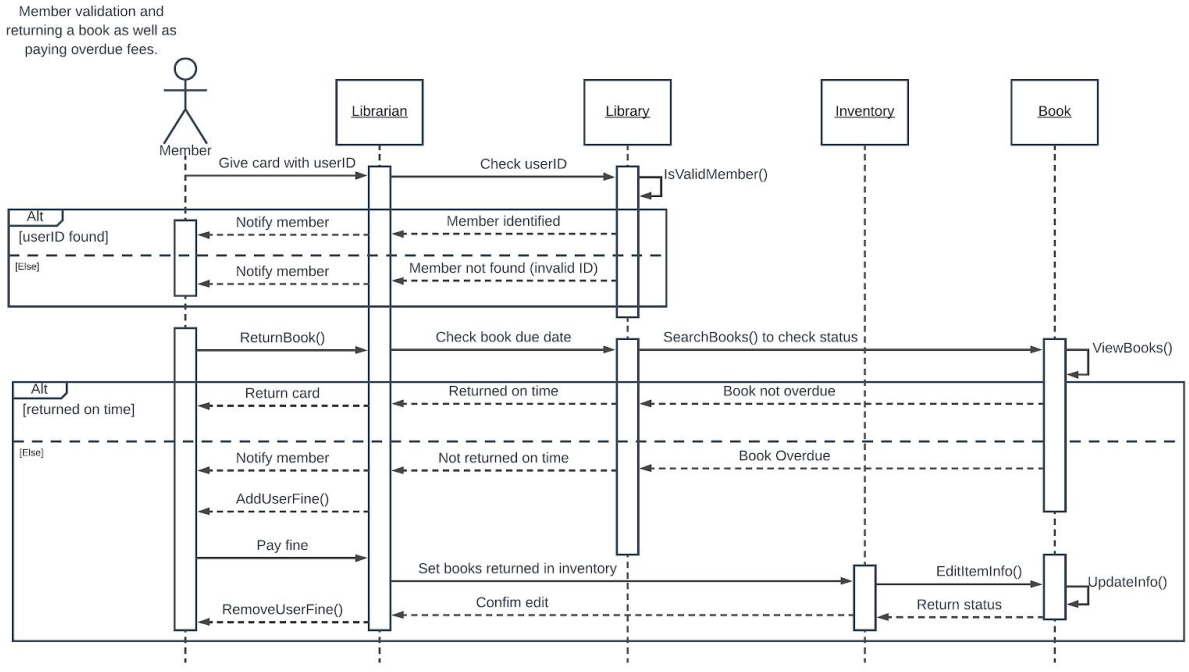


8. Sequence Diagrams

8.1

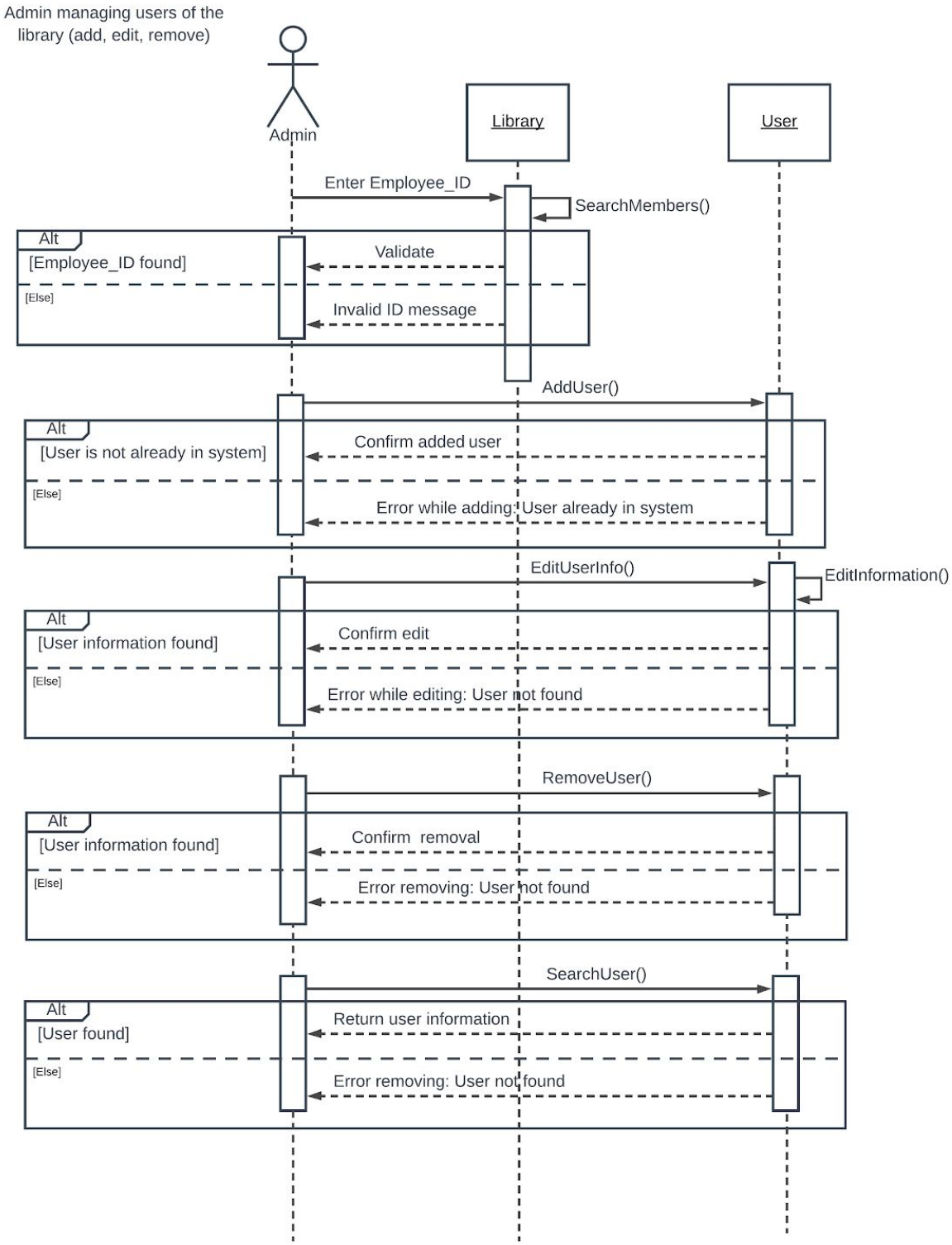


8.2



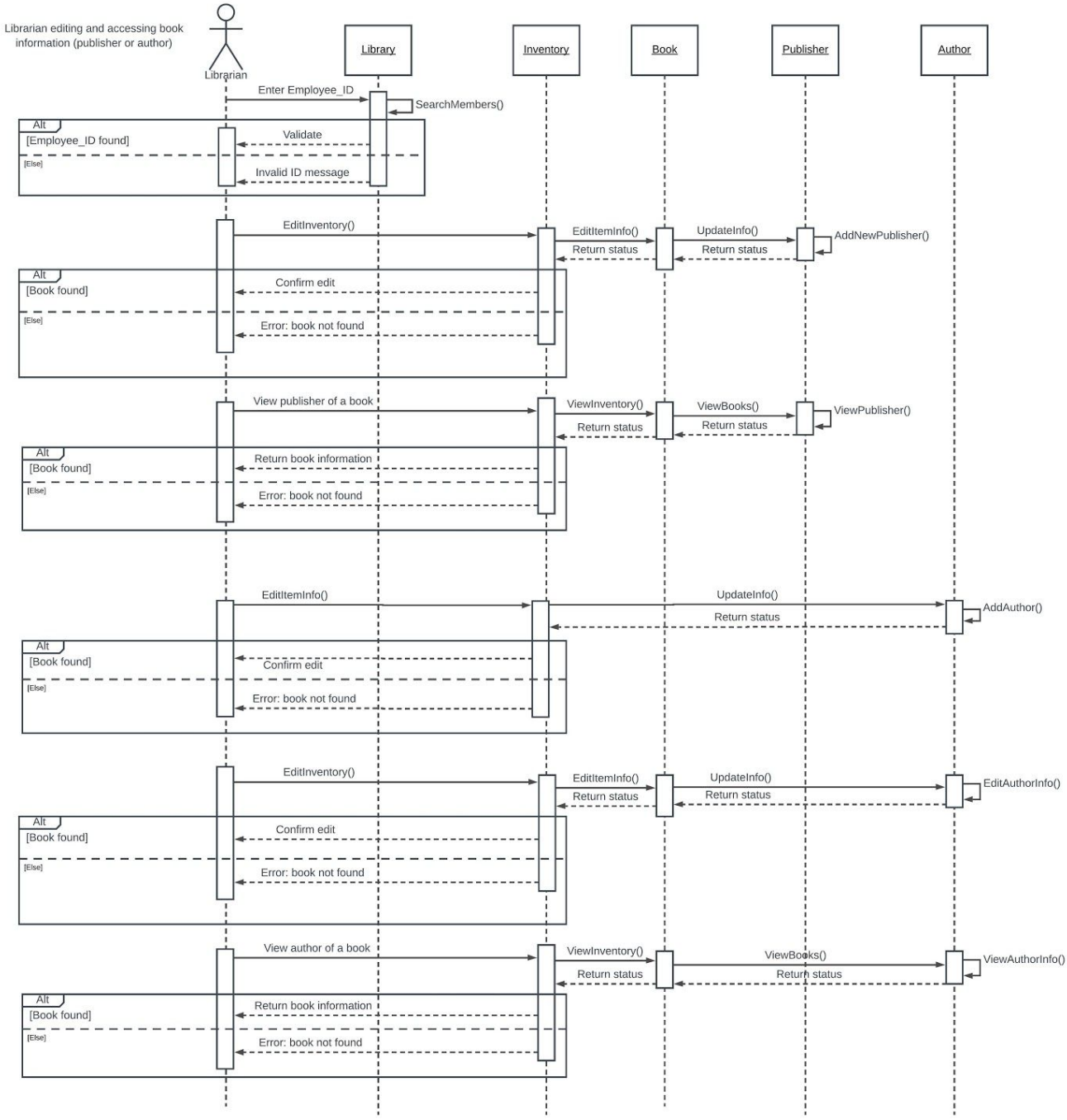
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

8.3



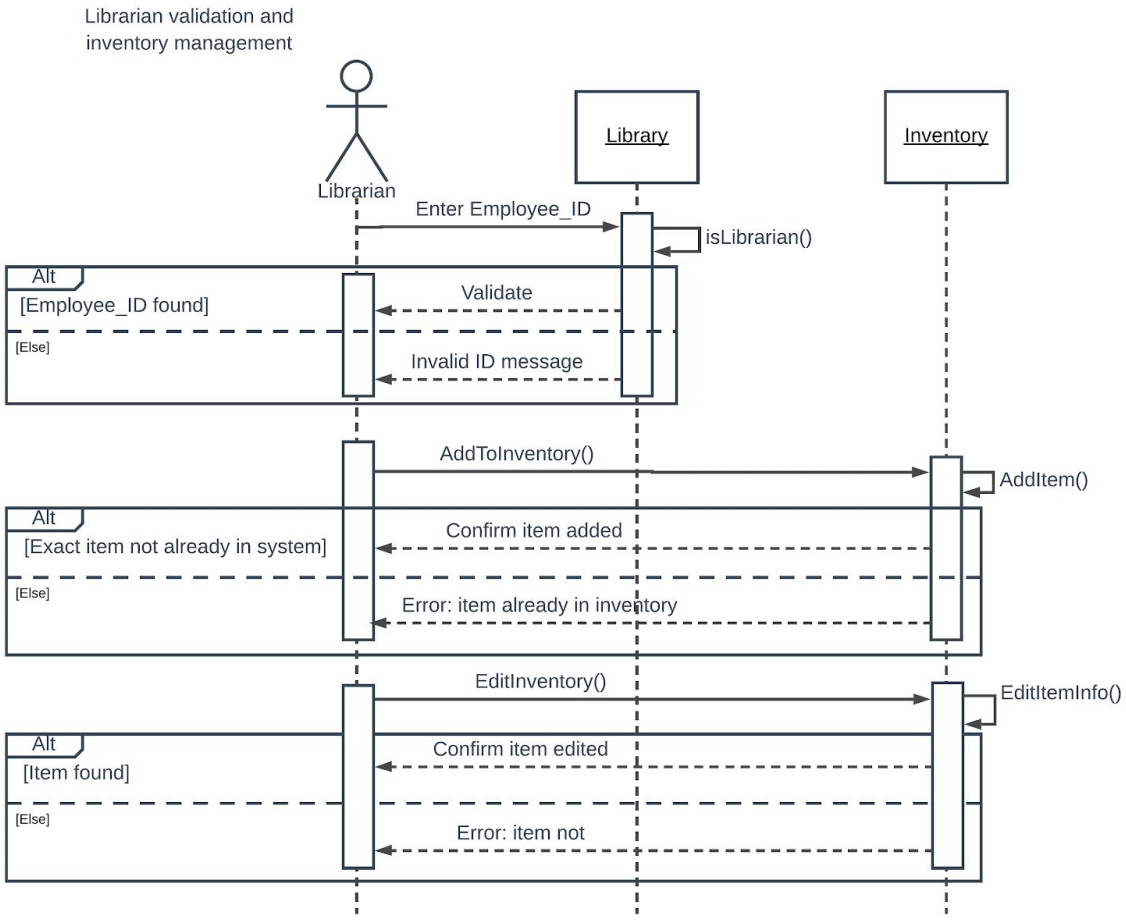
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

8.4



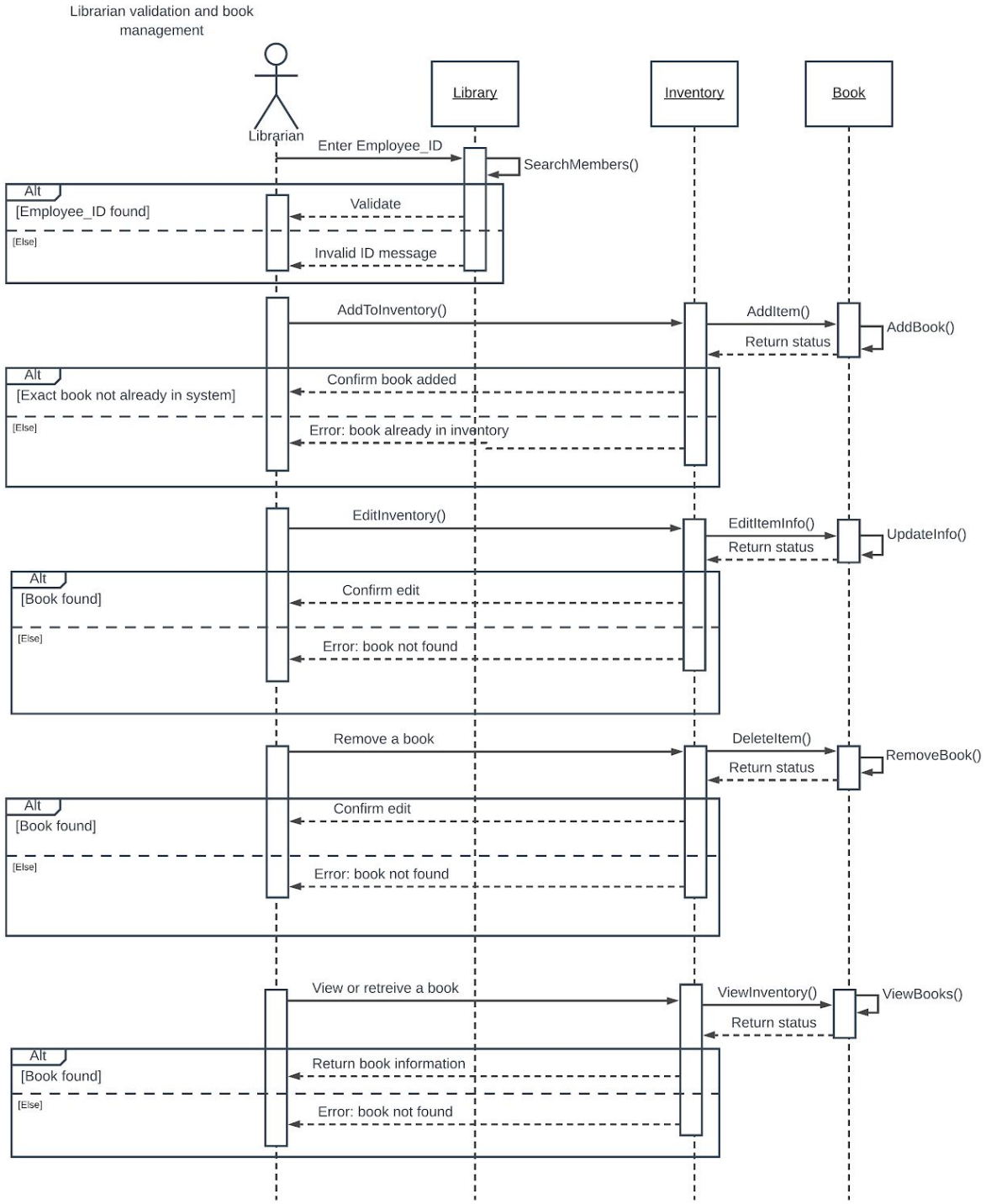
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

8.5



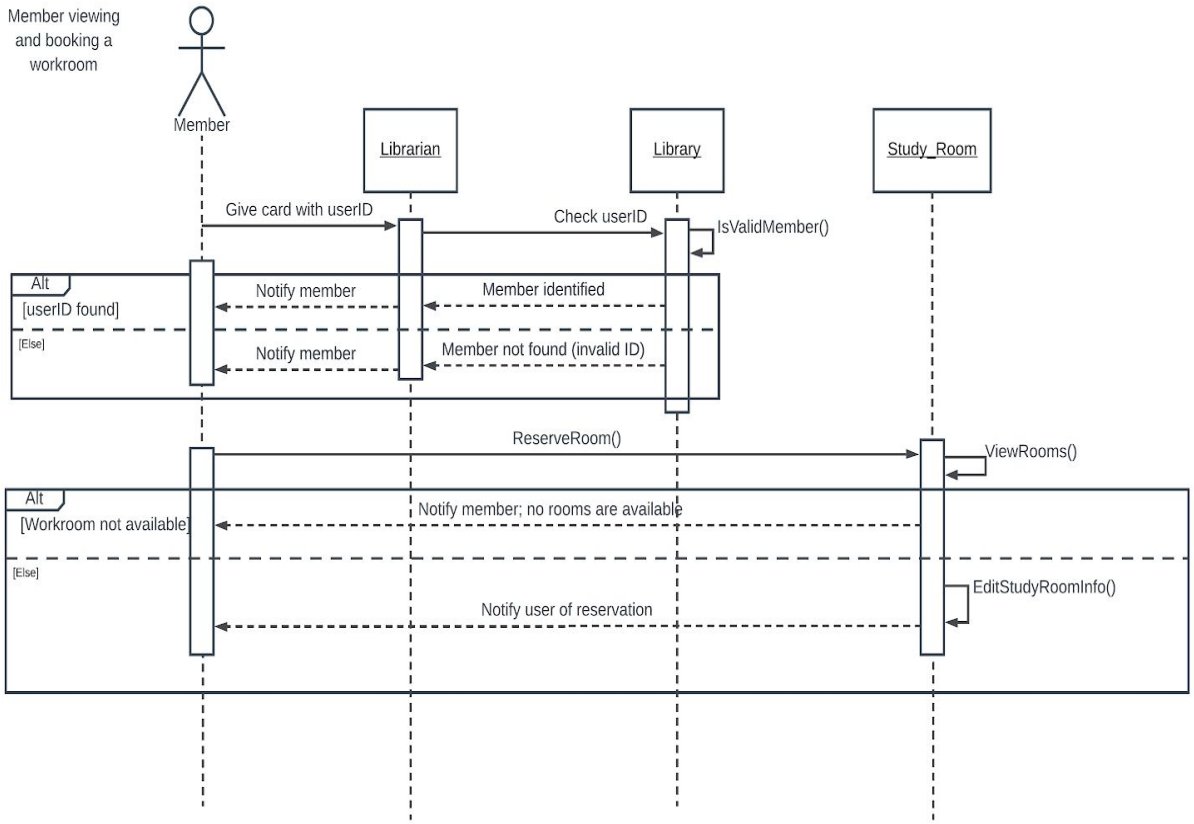
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

8.6



Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

8.7



9. Queries for all Classes

The queries shown below modify the attributes for their respective classes in relation to the classes shown in the class diagram

9.1 Library

Function: AddNewLocation()
Inputs: @Name, @Location
Outputs: None
Query: INSERT INTO Library VALUES (@Name, @Location);

Function: RemoveLocation()
Inputs: @Name, @Location
Outputs: None
Query: DELETE FROM Library
WHERE Name = @Name AND Location = @Location
ON DELETE CASCADE ;

9.2 Study_Room

Function: NewStudyRoom()
Inputs: @Lib_Name, @Room_Number, @Capacity
Outputs: None
Query: INSERT INTO Study_Room VALUES (@Lib_Name, @Room_Number,
@Capacity);

Function: EditStudyRoomInfo()
Inputs: @Lib_Name, @Room_Number, @Capacity
Outputs: None
Query: UPDATE Study_Room
SET “attribute”
WHERE Lib_Name = @Lib_Name AND Room_Number = @Room_Number
AND Capacity = @Capacity;

Function: RemoveStudyRoom()
Inputs: @Lib_Name, @Room_Number, @Capacity
Outputs: None
Query: DELETE FROM Study_Room
WHERE Lib_Name = @Lib_Name AND Room_Number = @Room_Number
AND Capacity = @Capacity
ON DELETE CASCADE;

9.2.1 Available

Function: ViewRooms()
Inputs: @Date_Available, @Time_Available
Outputs: Room_Number, Capacity, Date_Available, Time_Available
Query: SELECT sr.Room_Number, sr.Capacity, a.Date_Available,
a.Time_Available
FROM Study_Room AS sr, Available AS a
WHERE a.Date_Available = @Date_Available AND a.Time_Available =
@Time_Available;

Function: ReserveRoom()
Inputs: @Date_Available, @Time_Available
Outputs: None
Query: DELETE FROM Available
WHERE Date_Available = @Date_Available AND Time_Available =
@Time_Available;
ON DELETE CASCADE;

9.2.2 Booked

Function: ViewReservations()
Inputs: None
Outputs: Room_Number, Capacity, Date_Available, Time_Available
Query: SELECT sr. Room_number, sr.Capacity, a.Date_Booked,
a.Time_Booked, a.Booking_Duration
FROM Study_Room AS sr, Booked AS b

Function: RemoveReservation()
Inputs: @Room_Number, @Date_Booked, @Time_Booked
Outputs: None
Query: DELETE FROM Booked
WHERE Room_Number = @Room_Number AND Date_Booked =
@Date_Booked AND Time_Booked = @Time_Booked;

9.3 User

Function: ReserveRoom()
Inputs: @Date_Available, @Time_Available
Outputs: None
Query: DELETE FROM Available
WHERE Date_Available = @Date_Available AND Time_Available = @Time_Available;
ON DELETE CASCADE;

Function: EditInformation()
Inputs: @User_ID, @First_Name, @Middle_Name, @Last_Name, @Street_Name, @City, @Country, @Phone_Number, @Lib_Name
Outputs: None
Query: UPDATE User
SET “attribute”
WHERE User_ID = @User_ID AND First_Name = @First_Name AND Middle_Name = @Middle_Name AND Last_Name = @Last_Name AND Street_Name = @Street_Name AND City = @City AND Country = @Country AND Phone_Number = @Phone_Number AND Lib_Name = @Lib_Name;

9.3.1 Member

Function: BorrowBook()
Inputs: @User_ID, @Book_ISBN, @Date_Borrowed, @Date_Returned, @Borrow_Duration
Outputs: None
Query: INSERT INTO Borrows VALUES (@User_ID, @Book_ISBN, @Date_Borrowed, @Date_Returned, @Borrow_Duration);

Function: ReturnBook()
Inputs: @User_ID, @Book_ISBN, @Date_Borrowed, @Date_Returned, @Borrow_Duration
Outputs: None
Query: DELETE FROM Borrows
WHERE User_ID = @User_ID AND Book_ISBN = @Book_ISBN AND Date_Borrowed = @Date_Borrowed AND Date_Returned = @Date_Returned AND Borrow_Duration = @Borrow_Duration;

9.3.2 Administrator

Function: AddUser()
Inputs: @User_ID, @First_Name, @Middle_Name, @Last_Name,
@Street_Name, @City, @Country, @Phone_Number, @Lib_Name
Outputs: None
Query: INSERT INTO User VALUES (@User_ID, @First_Name,
@Middle_Name, @Last_Name, @Street_Name, @City,
@Country, @Phone_Number, @Lib_Name);

Function: RemoveUser()
Inputs: @User_ID, @First_Name, @Middle_Name, @Last_Name,
@Street_Name, @City, @Country, @Phone_Number, @Lib_Name
Outputs: None
Query: DELETE FROM User
WHERE User_ID = @User_ID AND First_Name = @First_Name AND
Middle_Name = @Middle_Name AND Last_Name =
@Last_Name AND Street_Name = @Street_Name AND City =
@City AND Country = @Country AND Phone_Number =
@Phone_Number AND Lib_Name = @Lib_Name
ON DELETE CASCADE;

Function: EditUserInfo()
Inputs: @User_ID, @First_Name, @Middle_Name, @Last_Name,
@Street_Name, @City, @Country, @Phone_Number, @Lib_Name
Outputs: None
Query: UPDATE User
SET “attribute”
WHERE User_ID = @User_ID AND First_Name = @First_Name AND
Middle_Name = @Middle_Name AND Last_Name =
@Last_Name AND Street_Name = @Street_Name AND City =
@City AND Country = @Country AND Phone_Number =
@Phone_Number AND Lib_Name = @Lib_Name;

Function: SearchUser()
Inputs: @User_ID, @First_Name
Outputs: User_ID, First_Name, Middle_Name, Last_Name,
Street_Name, City, Country, Phone_Number, Lib_Name
Query: SELECT *
FROM User
WHERE User_ID = @User_ID AND First_Name = @First_Name;

9.3.3 Librarian

Function: AddToInventory()
Inputs: @Item_ID, @Item_Type, @Item_Location
Outputs: None
Query: INSERT INTO Inventory VALUES (@Item_ID, @Item_Type,
@Item_Location);

Function: SearchBooks()
Inputs: @ISBN
Outputs: Item_ID, Item_Location, Quantity, ISBN, Title, Edition, Language
Query: SELECT i.Item_ID, i.Item_Location, COUNT(ISBN) AS Quantity, b.ISBN,
b.Title, b.Edition, b.Language
FROM Inventory AS i, Book as b
WHERE b.ISBN = @ISBN AND b.Item_ID = i.Item_ID
GROUP BY (b.Title, b.Edition)
HAVING Quantity > 0;

Function: SearchMember()
Inputs: @User_ID, @First_Name
Outputs: User_ID, First_Name, Last_Name, Phone_Number, Lib_Name,
Number_Of_Books_Borrowed
Query: SELECT User_ID, First_Name, Last_Name, Phone_Number, Lib_Name,
Number_of_Books_Borrowed
FROM Member
WHERE User_ID = @User_ID AND First_Name = @First_Name;

Function: LendBooks()
Inputs: @User_ID, @Book_ISBN, @Date_Borrowed, @Date_Returned,
@Borrow_Duration
Outputs: None
Query: INSERT INTO Borrows VALUES (@User_ID, @Book_ISBN,
@Date_Borrowed, @Date_Returned, @Borrow_Duration);

Function: EditInventory()
Inputs: @Item_ID,@Item_Type, @Item_Location
Outputs: None
Query: UPDATE Inventory,
SET “attribute”
WHERE Item_ID = @Item_ID AND Item_Type = @Item_Type AND
Item_Location = @Item_Location;

9.4 Fees

Function: AddUserFee()
Inputs: @User_ID, @Fee_Type, @Amount
Outputs: None
Query: INSERT INTO Fees VALUES (@User_ID, @Fee_Type, @Amount);

Function: RemoveUserFee()
Inputs: @User_ID, @Fee_Type, @Amount
Outputs: None
Query: DELETE FROM Fees
WHERE User_ID = @User AND Fee_Type = @Fee_Type AND Amount =
@Amount;

9.5 Inventory

Function: AddItem()
Inputs: @Item_ID, @Item_Type, @Item_Location
Outputs: None
Query: INSERT INTO Inventory VALUES (@Item_ID, @Item_Type,
@Item_Location);

Function: EditItemInfo()
Inputs: @Item_ID,@Item_Type, @Item_Location
Outputs: None
Query: UPDATE Inventory,
SET “attribute”
WHERE Item_ID = @Item_ID AND Item_Type = @Item_Type AND
Item_Location = @Item_Location;

Function: DeleteItem()
Inputs: @Item_ID,@Item_Type, @Item_Location
Outputs: None
Query: DELETE FROM Inventory
WHERE Item_ID = @Item_ID AND Item_Type = @Item_Type AND
@Item_Location
ON DELETE CASCADE;

Function: ViewInventory()
Inputs: None
Outputs: Item_Type, Item_Location, Quantity
Query: SELECT Item_Type, Item_Location, COUNT(Item_Type) AS Quantity
FROM Inventory
GROUP BY Item_Type

9.6 Publisher

Function: AddNewPublisher()
Inputs: @PublisherID, @Name, @Location
Outputs: None
Query: INSERT INTO Publisher VALUES (@PublisherID, @Name, @Location);

Function: ViewPublisher()
Inputs: None
Outputs: @PublisherID, @Name, @Location
Query: SELECT *
FROM Publisher;

9.7 Author

Function: AddAuthor()
Inputs: @First_Name, @Middle_Name, @Last_Name, @Writing_Style, @Date_Born, @Date_Died, @Country_Of_Origin
Outputs: None
Query: INSERT INTO Author VALUES (@First_Name, @Middle_Name, @Last_Name, @Writing_Style, @Date_Born, @Date_Died, @Country_Of_Origin);

Function: EditAuthorInfo()
Inputs: @First_Name, @Middle_Name, @Last_Name, @Writing_Style, @Date_Born, @Date_Died, @Country_Of_Origin
Outputs: None.
Query: UPDATE Author
SET “attribute”
WHERE First_Name = @First_Name AND Middle_Name = @Middle_Name AND Last_Name = @Last_Name AND Writing_Style = @Writing_Style AND Date_Born = @Date_Born AND Date_Died = @Date_Died AND Country_Of_Origin = @Country_Of_Origin;

Function: ViewAuthorInfo()
Inputs: @First_Name, @Last_Name
Outputs: First_Name, Middle_Name, Last_Name, Writing_Style, Date_Born, Date_Died, Country_Of_Origin
Query: SELECT *
FROM Author
WHERE First_Name = @First_Name AND Last_Name = @Last_Name;

9.8 Book

Function: AddBook()
Inputs: @ISBN, @Title, @Publish_Date, Edition, @Language, @Item_ID, @Publisher_ID, @Publisher_Name, @Genre
Outputs: None
Query: INSERT INTO Book VALUES (@ISBN, @Title, @Publish_Date, Edition, @Language, @Item_ID, @Publisher_ID, @Publisher_Name, @Genre);

Function: UpdateInfo()
Inputs: @ISBN, @Title, @Publish_Date, @Edition, @Language, @Item_ID, @Publisher_ID, @Publisher_Name, @Genre
Outputs: None
Query: UPDATE Book
SET “attribute”
WHERE ISBN = @ISBN AND Title = @Title AND Publish_Date = @Publish_Date AND Edition = @Edition AND Language = @Language AND Item_ID = @Item_ID AND Publisher_ID = @Publisher_ID AND Publisher_Name = @Publisher_Name AND Genre = @Genre;

Function: ViewBook()
Inputs: @ISBN, @Title, @Edition
Outputs: ISBN, Title, Publish_Date, Edition, Language, Item_ID, Publisher_ID, Publisher_Name, Genre
Query: SELECT *
FROM Book AS b
WHERE b.ISBN = @ISBN AND b.Title = @Title AND b.Edition = @Edition;

Function: RemoveBook()
Inputs: @ISBN, @Title, @Publish_Date, @Edition, @Language, @Item_ID,
@Publisher_ID, @Publisher_Name, @Genre
Outputs: None
Query: DELETE FROM Book
WHERE ISBN = @ISBN AND Title = @Title AND Publish_Date =
@Publish_Date AND Edition = @Edition AND Language = @Language
AND Item_ID = @Item_ID AND Publisher_ID = @Publisher_ID AND
Publisher_Name = @Publisher_Name AND Genre = @Genre
ON DELETE CASCADE;

10. Appendix

The Appendices below show the instances of the value that are currently held in all the different tables of the system where some of the instances are directly connected to other tables because of their foreign key constraints. Some of the table contain information that should be used when adding information to tables for example to add new books, publisher information must be filled and since this information is linked through foreign keys, it is added to the publisher table first before adding an instance of a book

4.1 Appendix A: Administrators Instances

(UserID, EmployeeID, Date_Hired, Password)
(4, 1, '2019-04-08', 'pass4'),
(10, 6, '2019-04-11', 'pass10'),
(11, 5, '2019-03-20', 'pass11'),
(12, 4, '2017-07-11', 'pass12');

4.2 Appendix B: Author Instances

(First_Name, Middle_Name, Last_Name, Date_Born, Date_Died, Origin_Country, Writing_Style)
('Bruce', 'S', 'Davie', '1955-03-07', NULL, 'Canada', 'Informational'),
('Doug', 'A', 'Lowe', '1980-05-25', NULL, 'Australia', 'Informational'),
('Larry', 'L', 'Peterson', '1958-05-08', NULL, 'United States', 'Informational');

4.3 Appendix C: Book Instances

(ISBN, Item_ID, Title, Edition, Language, Publisher_Name, Publisher_ID, Publish_Date, Lendable)
('1111111111111', '2', 'Harry Potter and The Deathly Demos', 2, 'English', 'Pearson', 1, '2019-03-04', 1),
('978012385091', '4', 'Computer Networks: A System\'s Approach', 8, 'English', 'Emerald Group Publishing', 3, '1997-09-15', 1),
('9780470179154', '4', 'Networking for Dummies', 4, 'English', 'Macmillan Learning', 4, '2019-04-16', 1),

('9999999999999', '1', 'Fundamentals of Database Systems', 7, 'English', 'Pearson', 1, '2019-04-01', 1);

4.4 Appendix D: Booked Study Room Instances
(Room_Number, Date_Booked, Time_Booked, Booking_Duration)
('101', '2019-04-12', '06:14:00', 1),
('105', '2019-04-21', '14:16:00', 1),
('202', '2019-04-27', '18:15:00', 1);
4.5 Appendix E: Borrows Instances
(UserID, ISBN, Date_Borrowed, Date_Returned, Borrow_Duration)
(1, '9999999999999', '2019-04-12', '0000-00-00', 2),
(2, '9999999999999', '2019-04-12', '2019-04-15', 4),
(6, '1111111111111', '2019-04-12', '0000-00-00', 2);

4.6 Appendix F: Inventory Instances
(Item_ID, Item_Type, Item_Location)
('1', 'Book', 'main'),
('2', 'Book', 'SecondFloor'),
('3', 'Book', 'ThirdFloor'),
('4', 'Book', 'SecondFloor');

4.7 Appendix G: Librarian Instances
(UserID, EmployeeID, Date_Hired, Password)
(3, 1, '2019-05-12', 'pass'),
(9, 2, '2019-01-14', 'pass9'),
(13, 3, '2018-06-04', 'pass1');

4.8 Appendix H: Library Instances
(Name, Location)
('Doucette', 'University of Calgary'),
('Gallagher', 'University of Calgary'),
('TFDL', 'University of Calgary'),
('TITL', 'University of Calgary');

4.9 Appendix I: Member Instances
(UserID, Num_Of_Books_Borrowed, Password)
(1, 0, 'pass1'),
(2, 0, 'pass2'),
(6, 0, 'pass6');

4.10 Appendix J: Publisher Instances
(Name, ID, Location)
('Cengage Learning', 2, 'Boston, United States'),

('Emerald Group Publishing', 3, 'Bingly, United Kingdom'),
('Macmillan Learning', 4, 'London, United Kingdom'),
('McGraw-Hill Education', 5, 'New York, United States'),
('Pearson', 1, 'Canada'),
('Routledge Taylor & Francis', 6, 'New York, United States'),
('Wiley', 7, 'San Francisco, United States'),
('Wolters Kluwer', 8, 'South Holland, Netherlands');

4.11 Appendix K: Study Room Instances

(UserID, Room_Number)
(1, '101'),
(5, '202'),
(15, '105');

4.12 Appendix L: User Instances

(Room_Number, Capacity, Lib_Name)
('100', 5, 'TFDL'),
('101', 5, 'TFDL'),
('102', 5, 'TFDL'),
('103', 5, 'TFDL'),
('105', 5, 'TFDL'),
('106', 5, 'TFDL'),
('107', 6, 'TFDL'),
('108', 6, 'TFDL'),
('109', 6, 'TFDL'),
('110', 6, 'TFDL'),
('201', 8, 'TITL'),
('202', 8, 'TITL'),
('203', 8, 'TITL'),
('204', 8, 'TITL'),
('205', 8, 'TITL'),
('301', 4, 'Gallagher'),
('302', 4, 'Gallagher'),
('303', 4, 'Gallagher'),
('304', 4, 'Gallagher'),
('305', 4, 'Gallagher'),
('401', 12, 'Doucette'),
('402', 12, 'Doucette'),
('403', 6, 'Doucette'),
('404', 4, 'Doucette'),
('405', 10, 'Doucette');

4.13 Appendix M: Writes Instances

(UserID, First_Name, Middle_Name, Last_Name, Street_Name, City, Country, Phone_Number)
(1, 'Dan', 'M', 'Trump', '135', 'Calgary', 'Canada', '4829119'),
(2, 'Edward', 'M', 'Macadamia', '1234', 'Calgary', 'Canada', '4829119'),
(3, 'Narshbla', 'Da', 'librar', '135', 'Calgary', 'Canada', '4829119'),
(4, 'Kashfia', 'S', 'Sailunaz', 'demo street', 'Calgary', 'Canada', '1234567890'),
(5, 'Pavol', 'T', 'Federl', 'Rue Sherbrooke', 'Montreal', 'Canada', '4035236666'),
(6, 'Eddy', 'X', 'Qiang', 'SaddleTowne', 'calgary', 'canada', '4031234567'),
(7, 'Karim', 'K', 'Beyk', 'Argyle Street', 'Halifax', 'Canada', '5876365555'),
(8, 'Rogers', 'G', 'Goodman', 'Aspen Meadows', 'Calgary', 'Canada', '5874123456'),
(9, 'Billy', 'D', 'Richman', 'Abalone Street', 'Calgary', 'Canada', '9856321047'),
(10, 'Trilok', 'K', 'Patel', 'Cedarwood', 'Calgary', 'Canada', '5874632588'),
(11, 'Muhammad', 'G', 'Saadan', 'Citadel Pass', 'Calgary', 'Canada', '5874123690'),
(12, 'Trevor', 'H', 'Boyd', 'Deerside', 'Calgary', 'Canada', '1587412201'),
(13, 'Shuji', 'J', 'Chen', 'Copperstone', 'Calgary', 'Canada', '5878222077'),
(14, 'Kim', 'D', 'Jong', 'Coral Shores', 'Calgary', 'Canada', '4032382901'),
(15, 'Jose', 'O', 'Mourinho', 'Parkridge', 'Calgary', 'Canada', '4032016325');

11. References

Elmasri, Ramez, and Shamkant B. Navathe. *Fundamentals of Database Systems, 7th Ed*. Pearson Education, 2017.

Tutorial Slides of CPSC 471 for format of queries

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

Library Management System User Manual

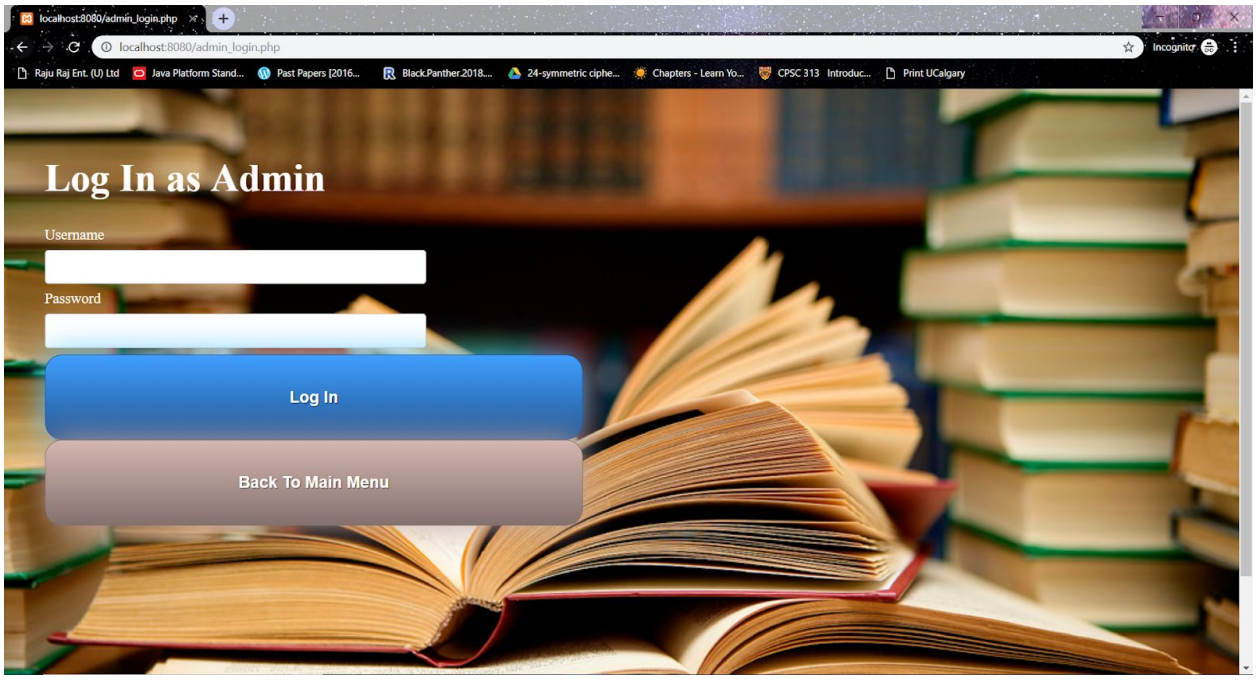
Table Of Contents:

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

1.0 - Main Menu

To log into the system, select the button depending on the type of user you are.

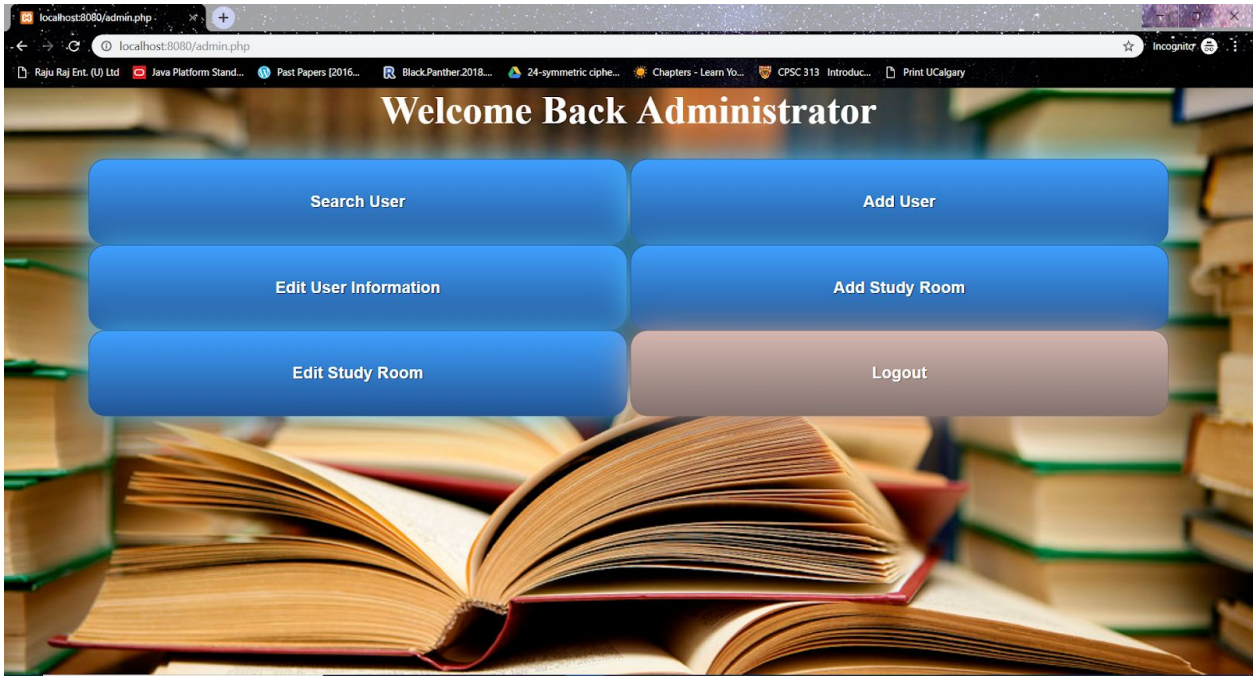
2.0 Admin Login Menu



Enter admin login credentials and click ‘Login’ to proceed.

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

2.1 Admin Menu



To search for a user, click ‘Search User’.
To add a new user, click ‘Add User’.
To edit user, click ‘Edit User Information’.
To add a new study room, click ‘Add Study Room’
To edit a study room information, click ‘Edit Study Room’

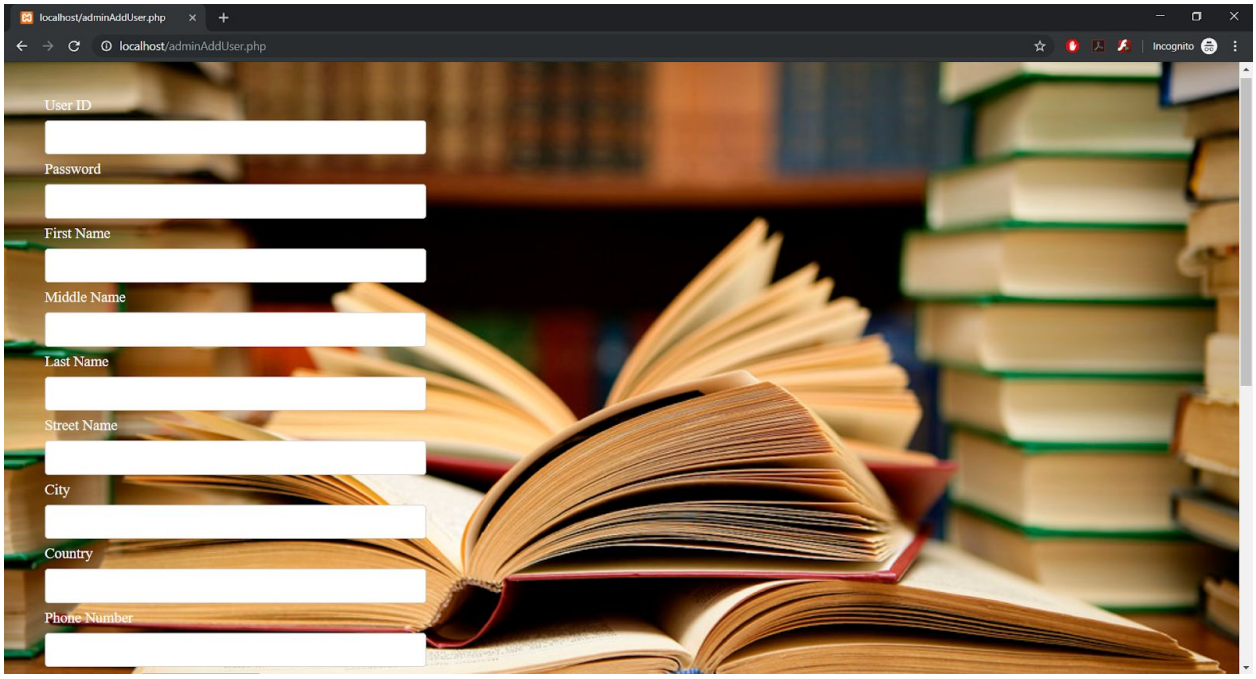
2.2 Search User

Users can be searched by their ID number. Enter the ID and click ‘search’.

This page displays the relevant user information.

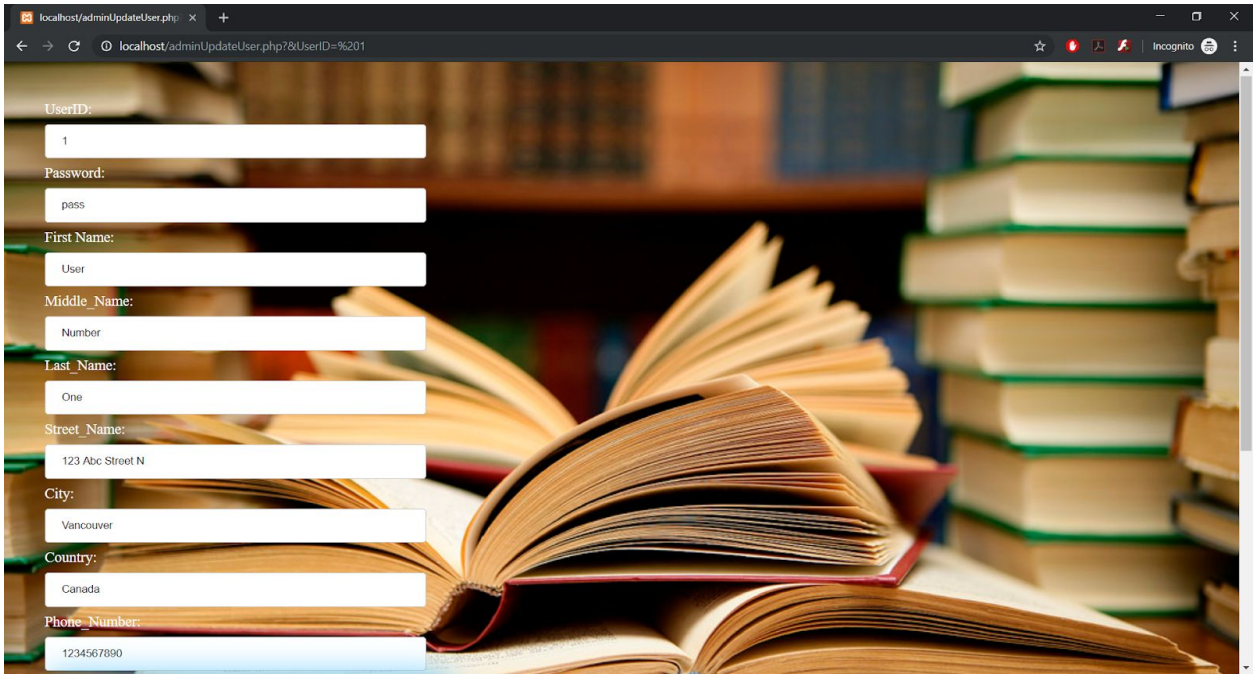
2.3 Add User

Users can be added by entering their information then clicking ‘add’.



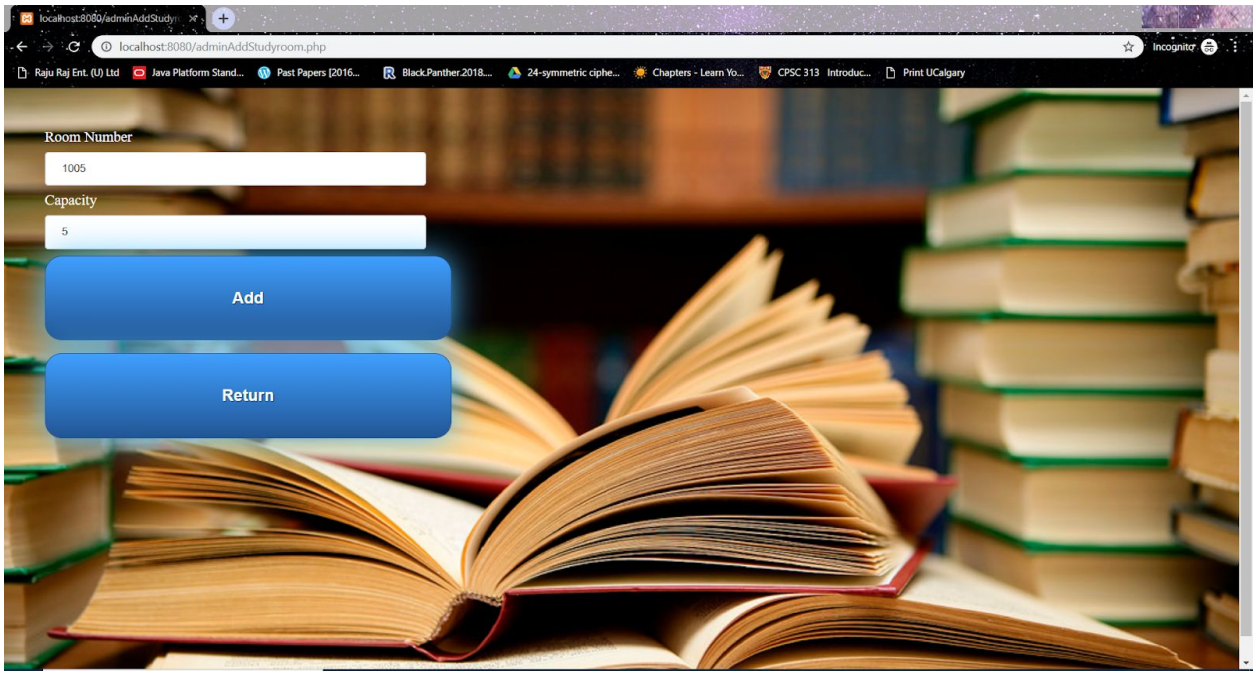
2.4 Edit User

User information can be added by clicking ‘Update’.
User record can be removed by clicking ‘Delete’.



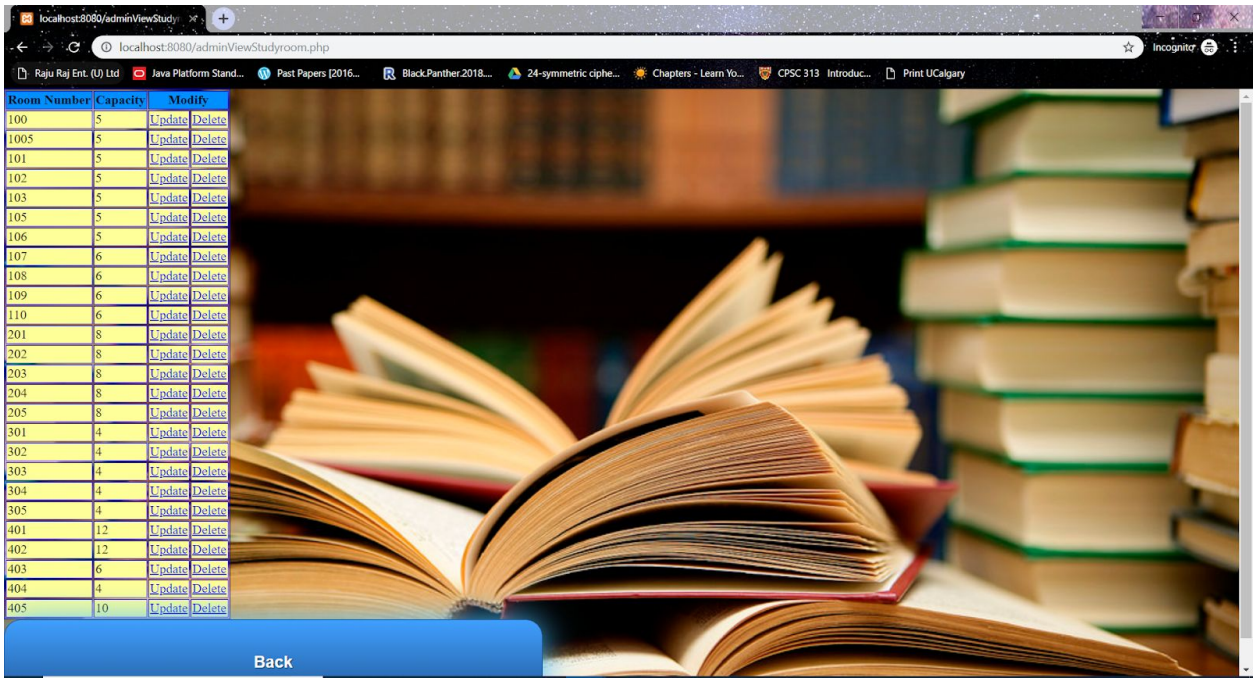
Upon selecting a record you wish to edit and clicking ‘Update’, a new menu will appear allowing you to enter new information. Finally click ‘Update’ to save the changes.

2.5 Add Study Room

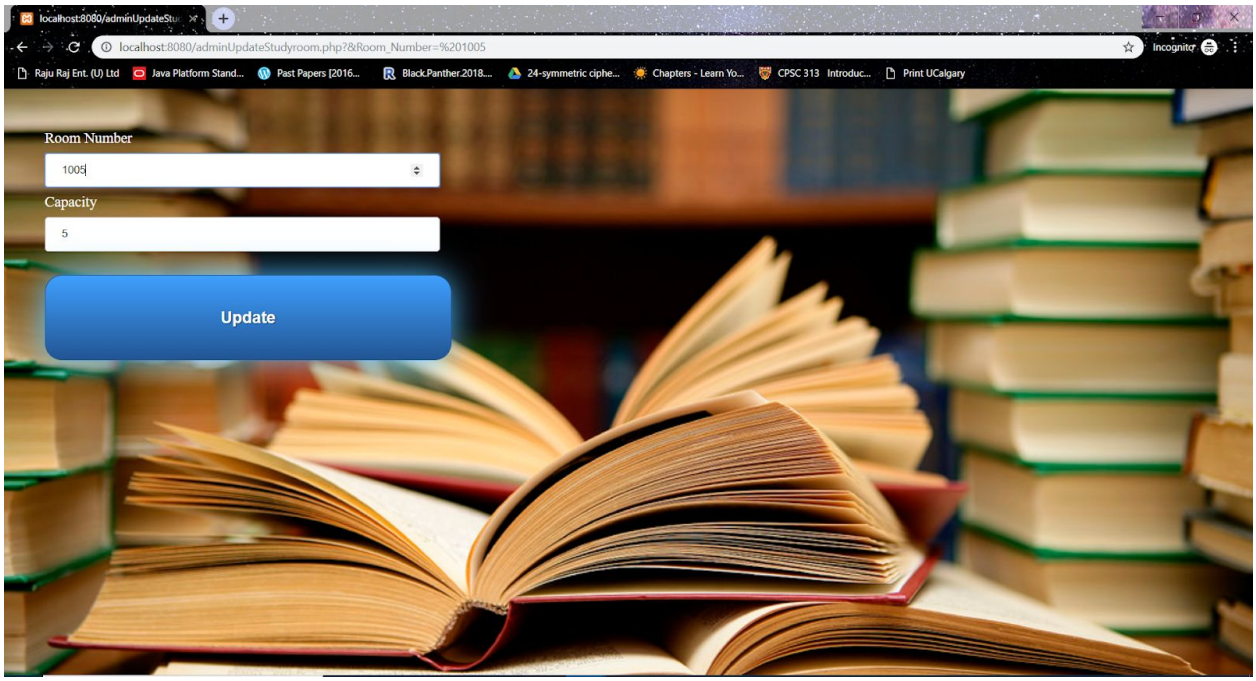


To add a new study room, simply add the information about the room number and click ‘Add’

2.6 Edit Study Room



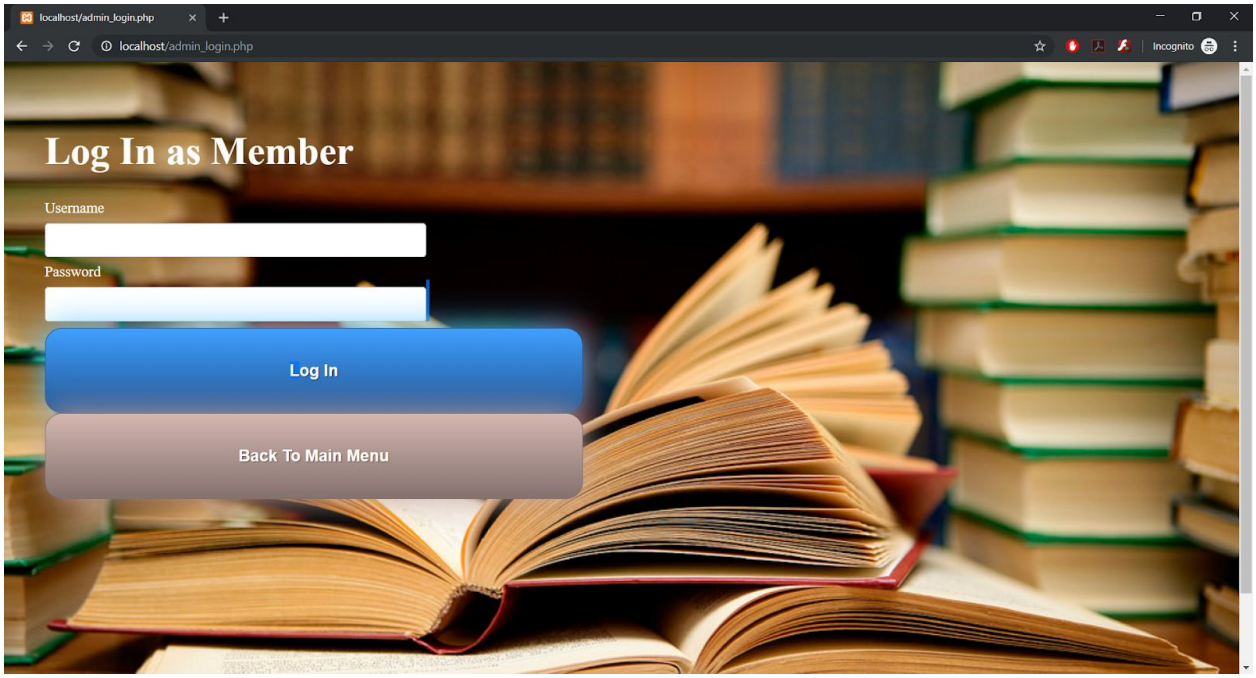
Click on any room in order to either delete it from the database, or click ‘update’ to make any changes to information about a room number



Simply change any information that needs to be updated and then click update to see the changes

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

3.0 Member Login Menu



Enter member login credentials and click ‘Login’ to proceed.

3.1 Member Menu

To search for a book, click ‘Search Book’.

To reserve a study room, click ‘Reserve Study Room’. To check your reservations, click ‘View Reservations’.

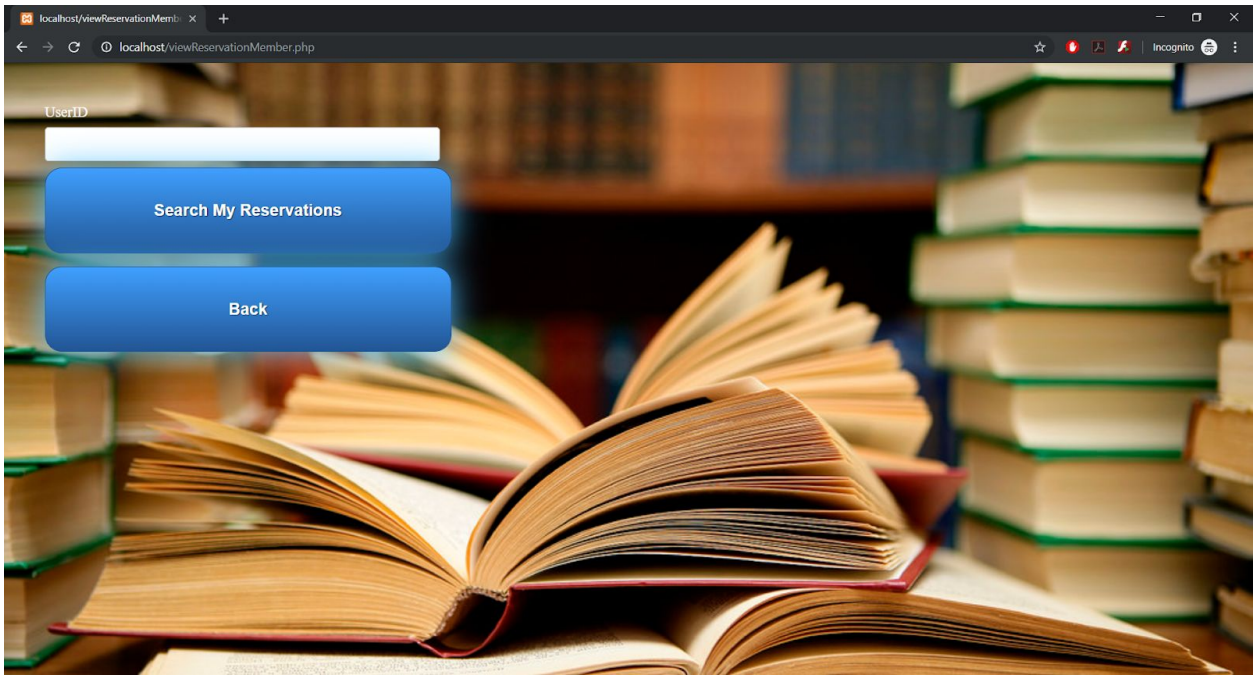
3.2 Search Book

Books can be searched by their ISBN. Enter the number and click ‘search’.

3.3 Reserve Study Room

Study rooms can be reserved by filling out this form and clicking ‘Reserve’.

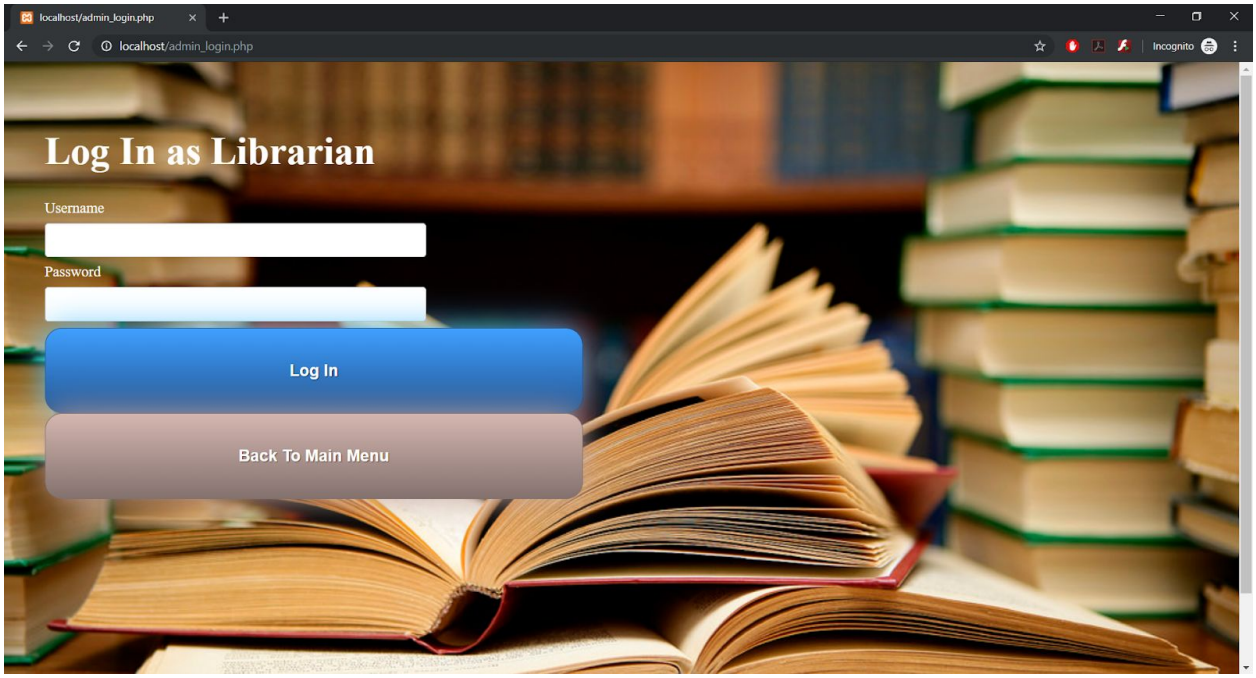
3.4 View Reservations



Room bookings can be viewed by entering your id and clicking ‘Search My Reservations’.

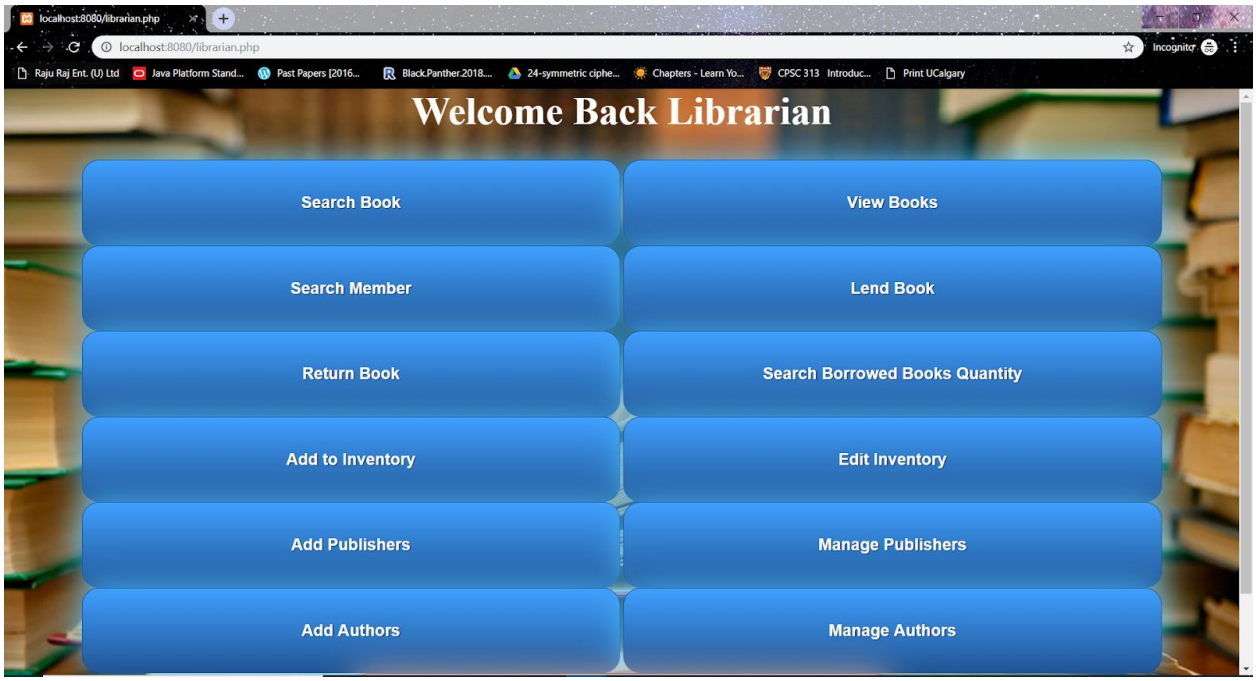
Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel

4.0 Librarian Login Menu



Enter librarian login credentials and click ‘Login’ to proceed.

4.1 Librarian Menu



To search for a book, click ‘Search Book’.
To search for a member, click ‘Search Member’.
To lend out a book to a member, click ‘Lend Book’.
To return a lent out book, click ‘Return Book’.
To see all the books that the library holds, click ‘View books’
To check how many books a member has borrowed, click ‘Search Borrowed Books Quantity’.
To add an item to inventory, click ‘Add to Inventory’.
To edit an item in inventory, click ‘Edit Inventory’.
To add a publisher, click ‘Add Publisher’
To manage information about a publisher, click ‘Manage Publishers’
To add an author, click ‘Add Author’
To manage information about Authors, ‘Manage Authors’
To log out of the system, click ‘logout’

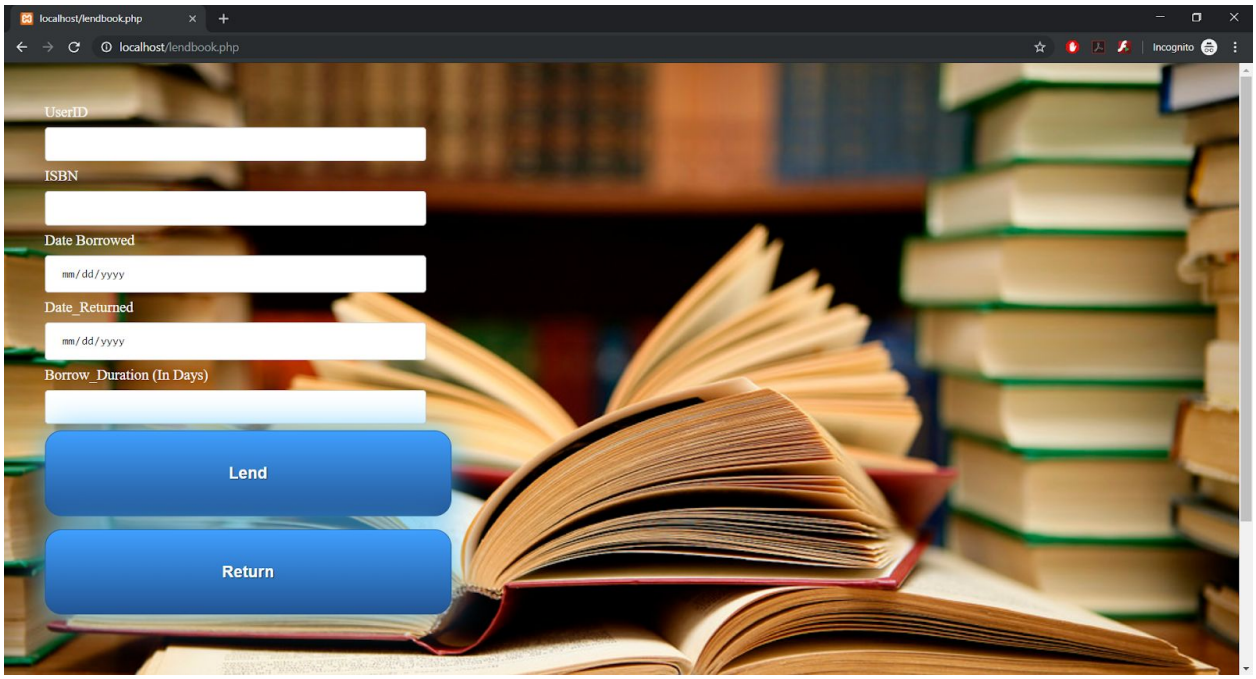
4.2 Search Book

Books can be searched by their ISBN. Enter the number and click ‘search’.
A book information with that ISBN will be shown

4.3 Search Member

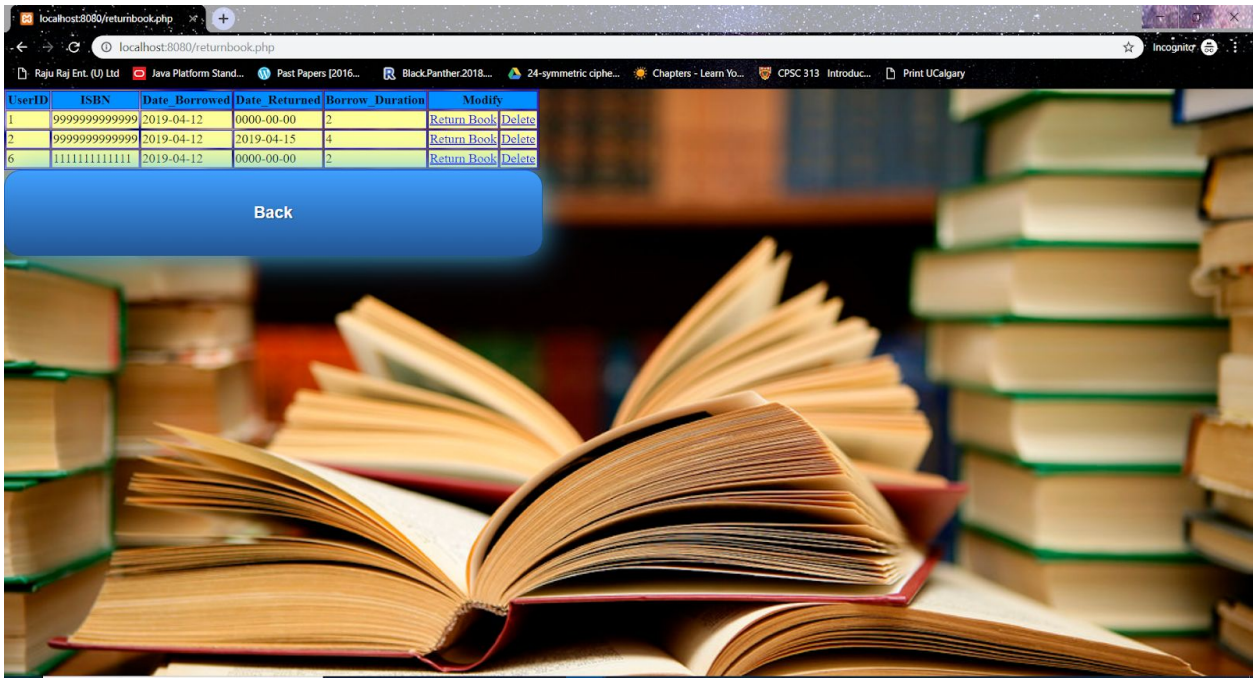
Members can be searched by their ID number. Enter the ID and click ‘search’.

4.4 Lend Book

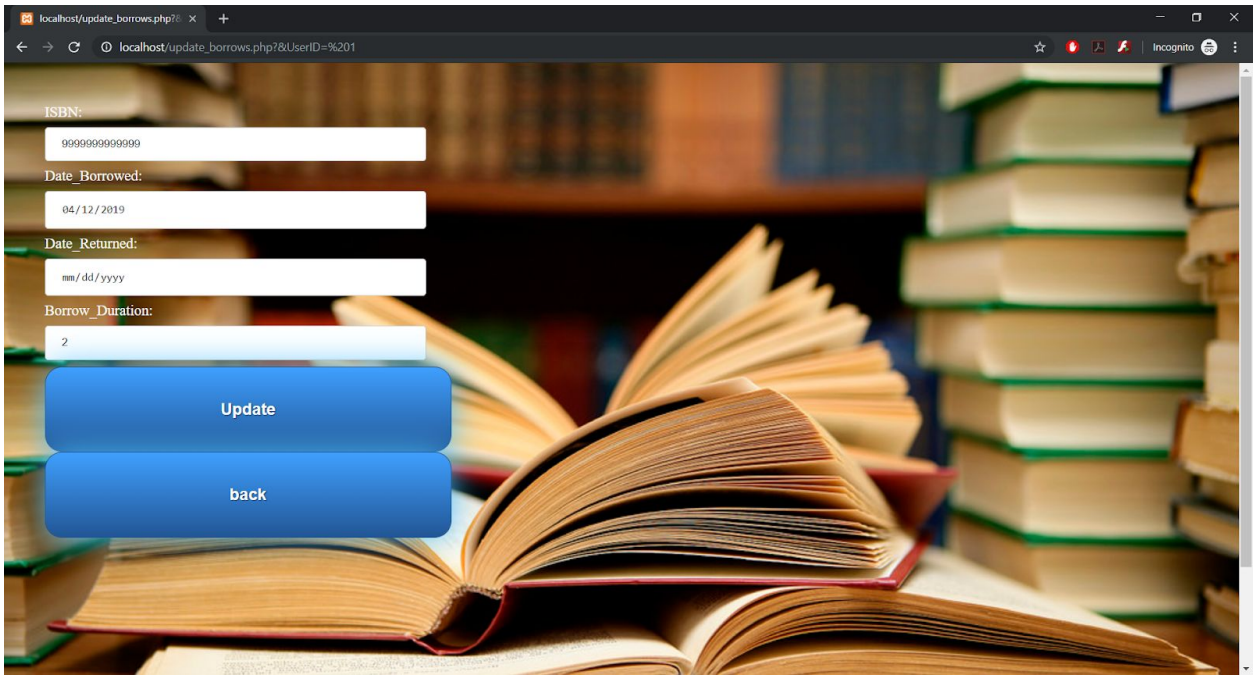


Books can be lent out by completing this form and clicking ‘Lend’.

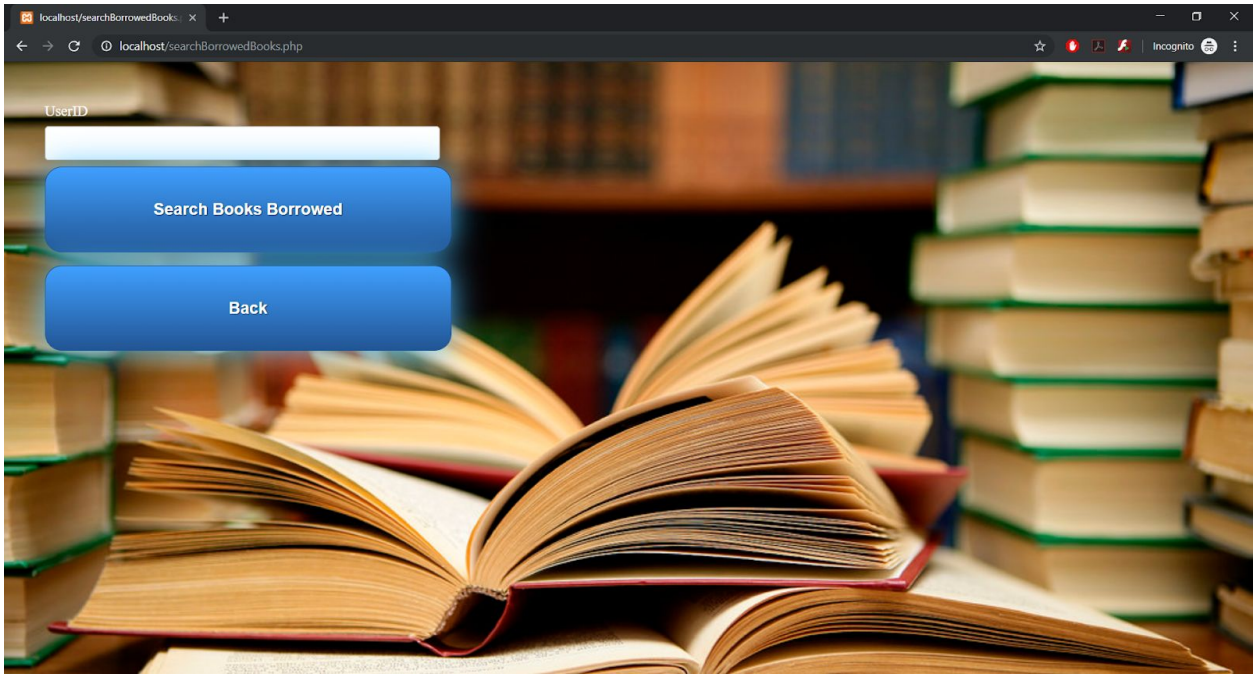
4.5 Return Book



Return a book by clicking ‘Return Book’, then updating the information and clicking ‘update’.

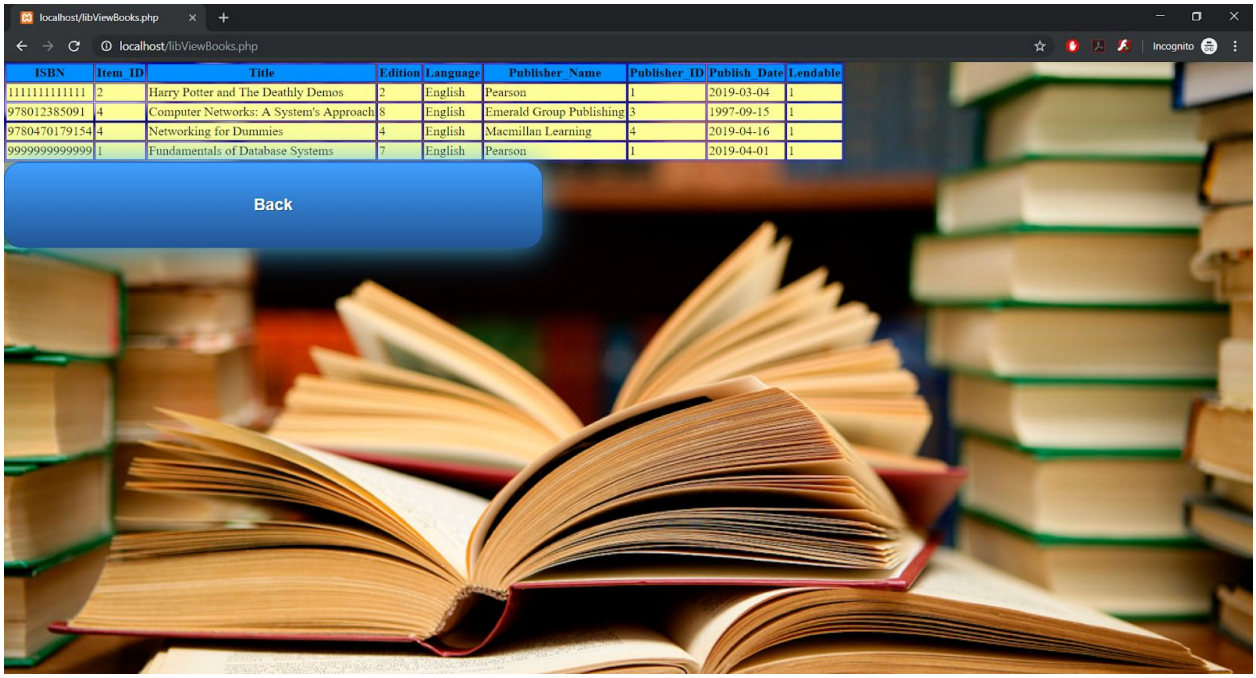


4.6 Search Borrowed Books Quantity



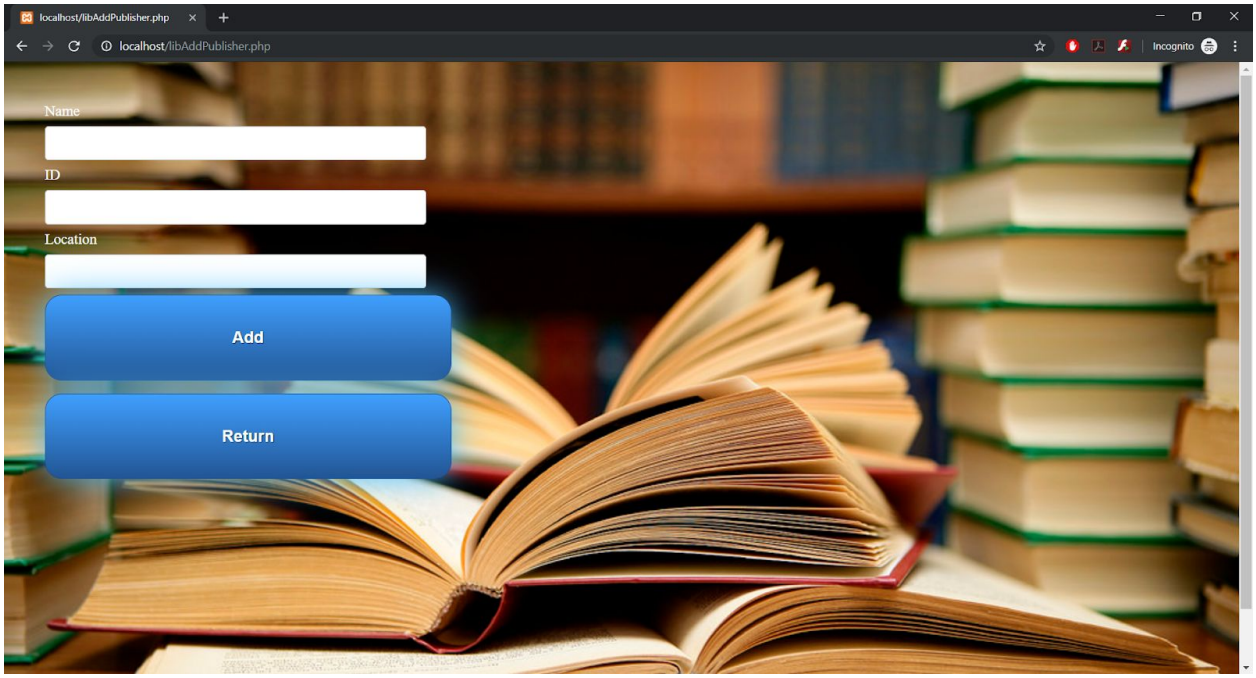
Enter the member's ID and click 'Search Books Borrowed' to see how many books they currently have signed out.

4.7 View Book



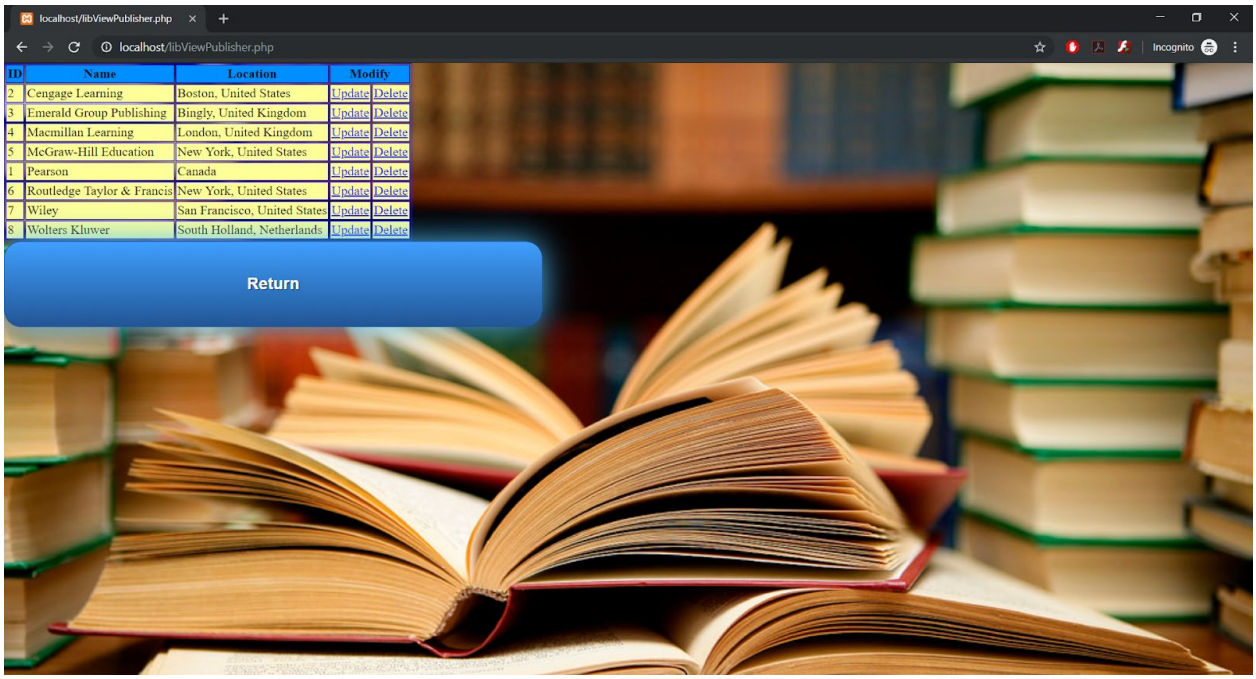
You can look at all the books in this menu.

4.8 Add Publisher

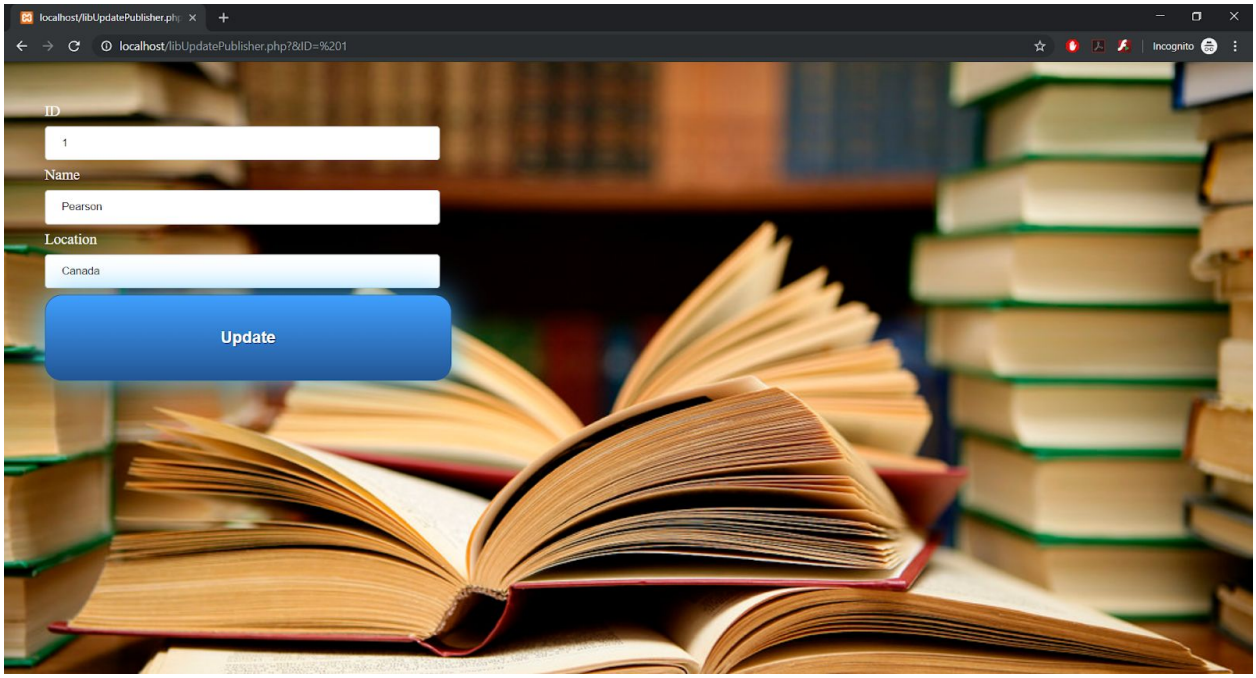


Create a new publisher by entering the publisher's information and click 'Add'.

4.9 Manage Publisher

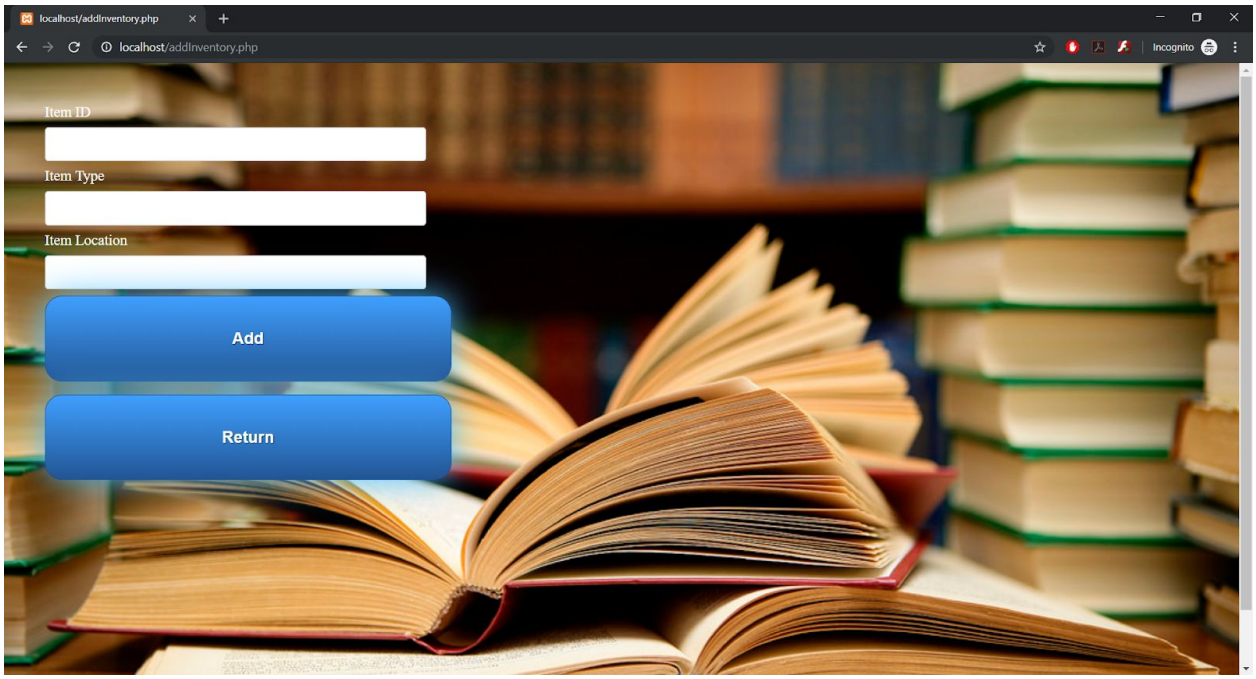


Publisher information can be added by clicking 'Update'.
Publisher record can be removed by clicking 'Delete'.



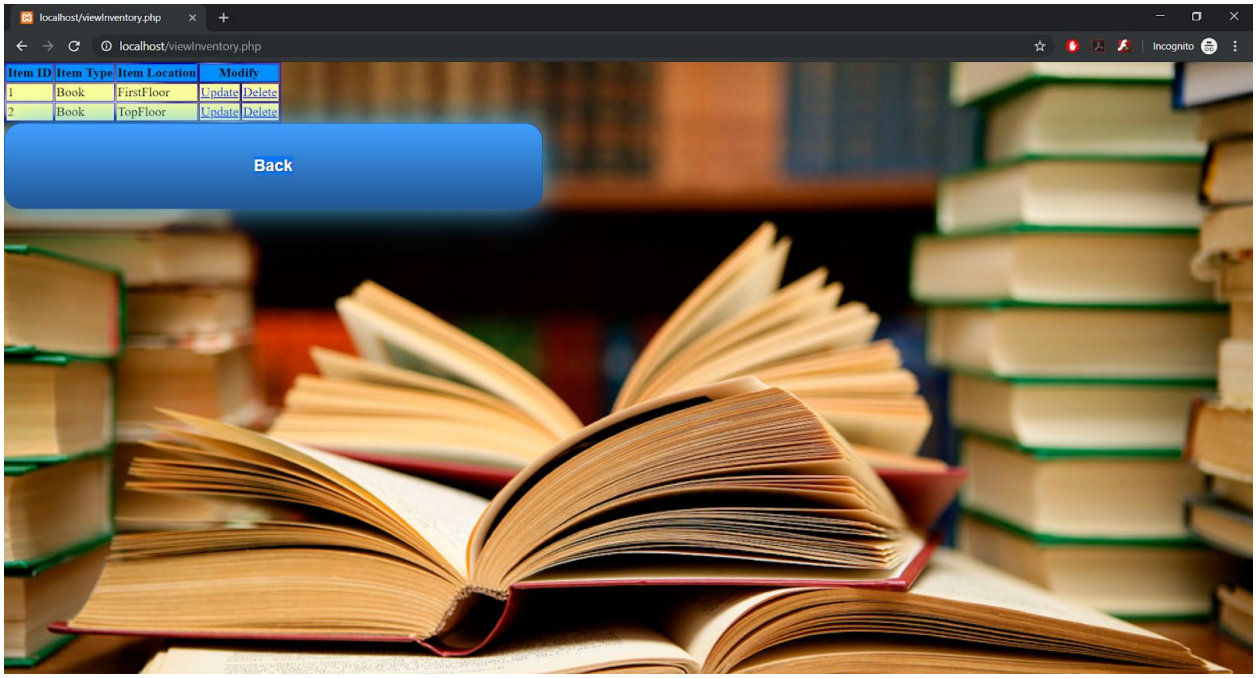
To edit enter the new information and click ‘Update’.

4.10 Add to Inventory

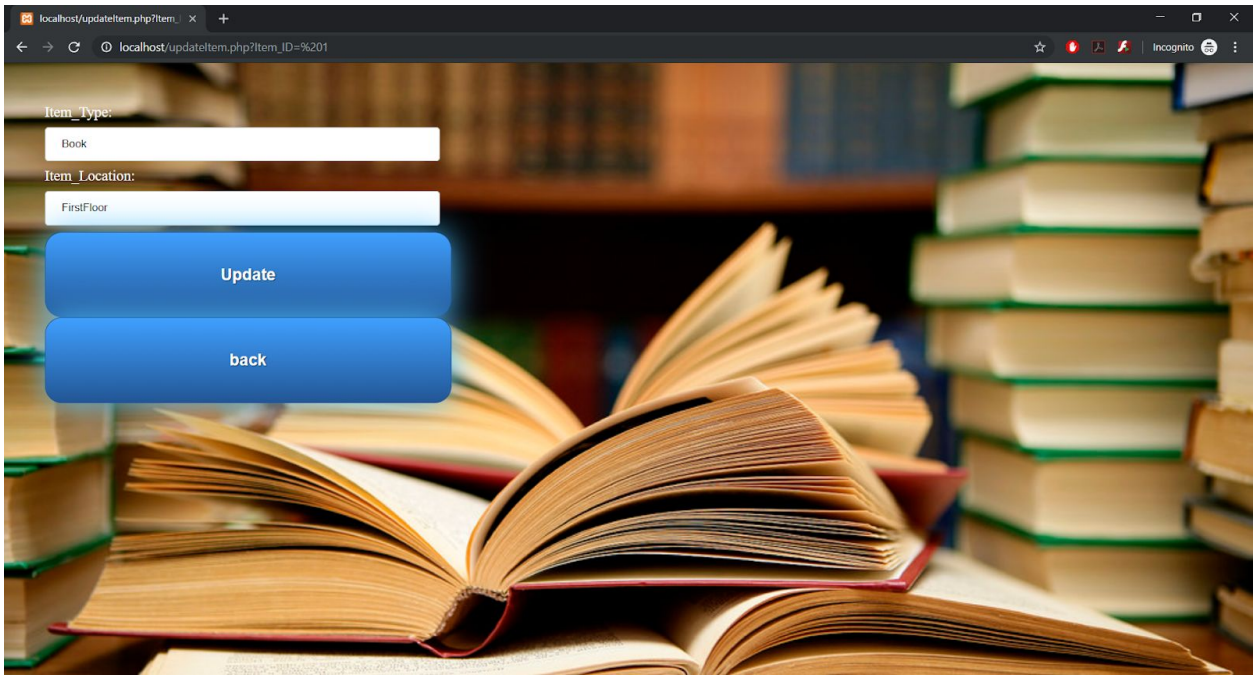


Enter the item’s information and click ‘Add’.

4.11 Edit Inventory



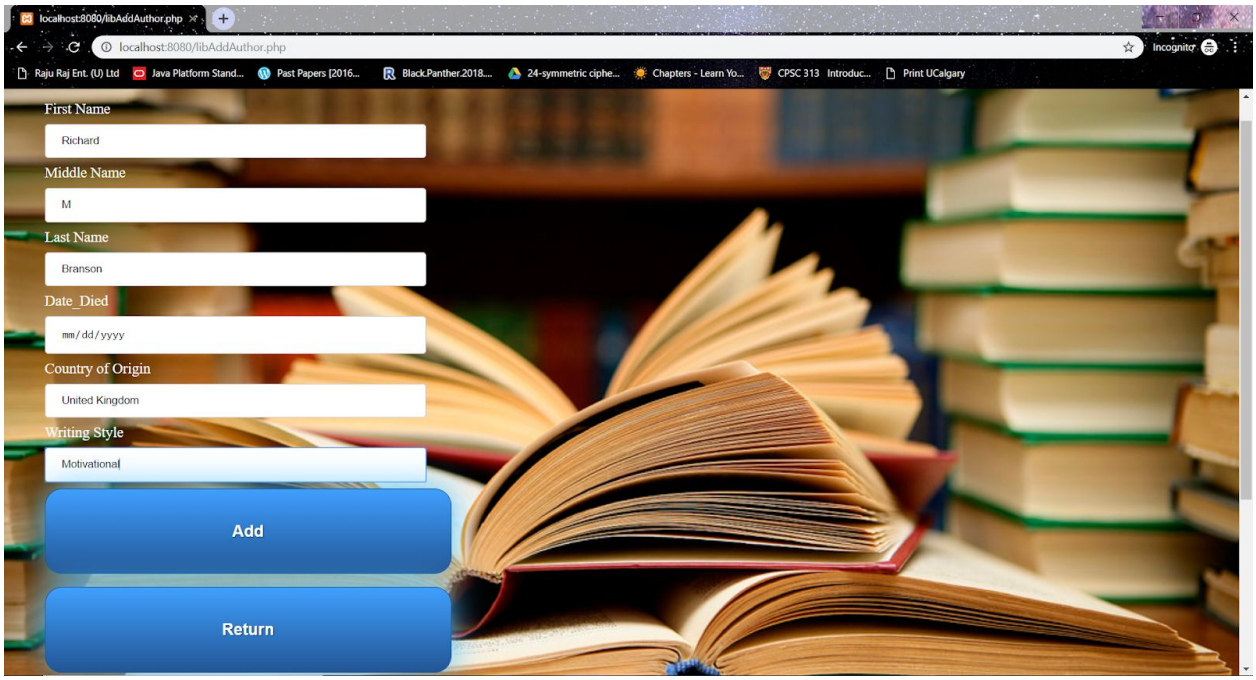
Inventory information can be added by clicking ‘Update’.
Inventory record can be removed by clicking ‘Delete’.



Upon selecting a record you wish to edit and clicking ‘Update’, a new menu will appear allowing you to enter new information. Finally click ‘Update’ to save the changes.

4.13 Add Author

To add a new author to the system, fill out their information and then click ‘Add’



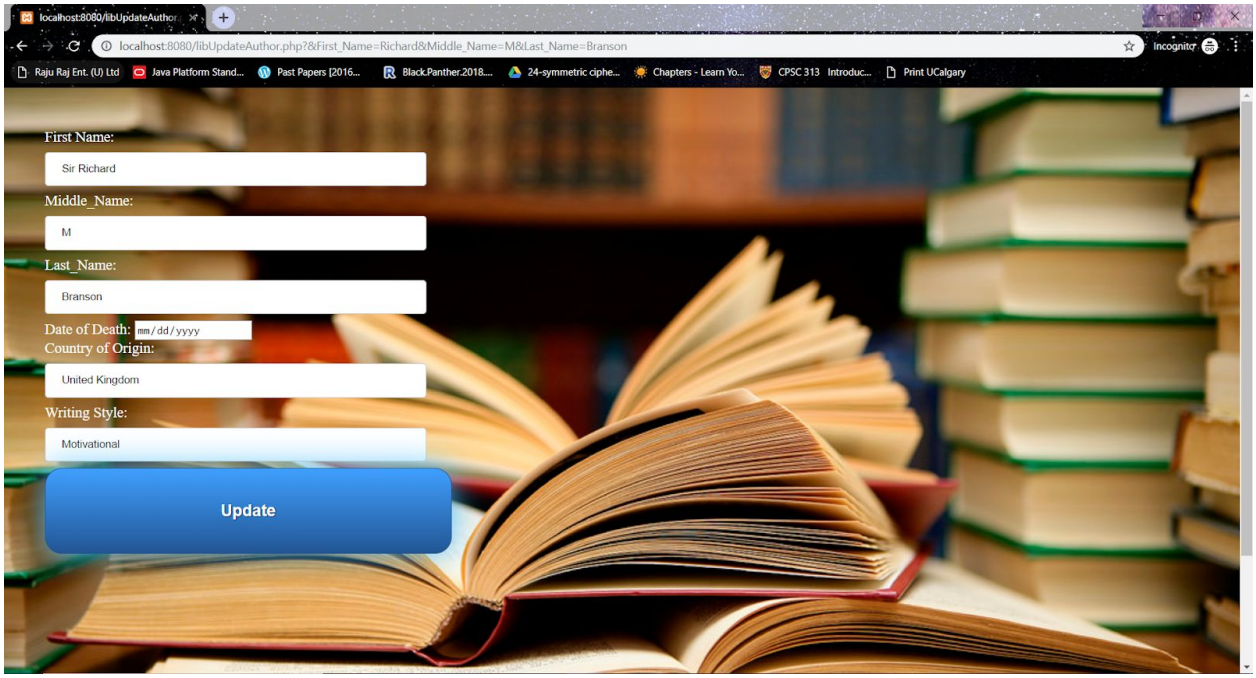
4.14 Manage Author

Incase an Author’s Information is to be removed, Click ‘Delete’

Incase an Author’s information needs to be updated, click ‘Update’

After clicking ‘Update’, any information that needs to be updated about an author such as updating their date of death or spelling mistakes, simply enter the information about this

Group 3: Eddy Qiang, Muhammad Saadan, Trilok Patel



END