LIBRARY MANAGEMENT SYSTEM

Session: 2021 – 2025

Submitted by:

Mehak Aftab 2021-CS-92

Mahnoor Ijaz 2021-CS-95

Amna Imran 2021-CS-96

Aqsa Mahmood 2021-CS-110

Supervised by:

Mr. Nazeef Ul Haq

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Acknowledgments

We are grateful to Allah Almighty that He provided us with the strength and power to complete this project in time. We are thankful to our course teacher Sir Nazeef Ul Haq for guiding us and it is due to his supervision that we are able to complete the project on time.
We would also like to thank him for his theory lectures which helped us analyse and understand the working of the database.
Moreover, we as a team were able cope the problems which were encountered during the implementation of this project.

Contents

Acknowledgments **i**

List of Figures **iv**

List of Tables **v**

Abstract **vi**

1 Introduction 1
- 1.1 Description 1
- 1.2 Purpose of the project 3
- 1.3 Features and Functionalities 3
- 1.4 Project Actors and Stakeholders 4
  - 1.4.1 Employees 4
  - 1.4.2 Student 4
  - 1.4.3 Administrator 4
- 1.5 System Requirements 4
- 1.6 ER Diagram 5
  - 1.6.1 One to One Relationship 5
  - 1.6.2 One to Many Relationship 5
  - 1.6.3 Many to Many Relationship 6
  - 1.6.4 Weak Entities 6
  - 1.6.5 Strong Relationship 6
  - 1.6.6 Multivalued Attributes 6
  - 1.6.7 Complex Attributes 6
  - 1.6.8 Computed Attributes 6
- 1.7 User Interface Details 7
- 1.8 Database Design 8

2 Use Cases 9
- 2.1 Use Case 1(Login) 9
- 2.2 Use Case 2(Employee) 10
- 2.3 Use Case 3(Ranks) 11
- 2.4 Use Case 4(Mark Attendance) 12
- 2.5 Use Case 5(Author) 13
- 2.6 Use Case 6(Books) 14
- 2.7 Use Case 7(Genre) 15

2.8 Use Case 8(Floors) 16
2.9 Use Case 9(Student) 17
2.10 Use Case 10(Issuance and Return) 18
2.11 Use Case 11(Returned Books) 18
3 Transactions 19
4 Indexes 19
4.1 Queries 19
5 Views 19
5.1 Queries 19
6 Stored Procedures 21
6.1 Queries 21
7 Triggers 23
7.1 Query 23
8 Limitations 25
9 Future Work 25
10 Conclusion 25

List of Figures

1 ER diagram of project 5
2 Database Design 8
3 Log In Screen 9
4 Employees Info Screen 10
5 Employees Ranks Screen 11
6 Employees Attendance Screen 12
7 Authors Screen 13
8 Books Screen 14
9 Genre Screen 15
10 Floor Screen 16
11 Student Screen 17
12 Issuances Screen 18
13 Returned Books Screen 18

List of Tables

1 System Requirements 4
2 User Interface Details 7
3 Used to describe use-case 1 of login 9
4 Used to describe use-case 2 of Employee 10
5 Used to describe use-case 3 of Ranks 11
6 Used to describe use-case 4 of Mark Attendance 12
7 Used to describe use-case 5 of Author 13
8 Used to describe use-case 6 of Books 14
9 Used to describe use-case 7 of Genre 15
10 Used to describe use-case 8 of Floors 16
11 Used to describe use-case 9 of Student 17
12 Used to describe use-case 10 of Issuance and Return 18
13 Used to describe use-case 11 of Returned Books 18

Abstract

In this project a 'Library operation system' is essential for the effective functioning of a library, as it helps librarians to keep track of various realities similar as scholars, books, authors, workers, and deals. The system should be suitable to induce reports on book revolution, overdue books, and outstanding deals.

It's also important for the system to keep track of hand attendance and rank, as well as the different bottoms, sections, and shelves where books are located. Overall, the library operation system is an necessary tool that helps librarians to manage the day- to- day operations of a library effectively.

1 Introduction

Library management system is a software application that helps librarians perform vital tasks such as borrowers, returning, tracking books, lending and transactions on a day-to-day basis. In this modern age of technology, having a robust and comprehensive library management system is critical for smooth functioning of a library. In this context, this essay discusses the various entities that are considered in a library management system and their significance in ensuring efficient library operations.

1.1 Description

A requisite in library services nowadays is a library management system. This software application helps librarians perform vital tasks such as tracking books, borrowers, lending and returning, and transactions on a day-to-day basis. In order to execute successful library management, the system needs to consider multiple entities including students, issuance, return, transactions, book details such as author and genre classification.

Another important entity that aids in efficient library operations is employee attendance including rank. By storing comprehensive data on students including their full names, places of residence, contact information and other relevant particulars; the library management system can ensure efficient organization. This approach also necessitates tracking all book-related activity per student including issued copies and accruing fines or fees. Paralleling this feature is prompt identification of issuance dates matched simultaneously by an expectation date for every book taken out.

To ensure efficient management of library resources and to cater to the needs of users, it is imperative for a library system to be able to generate reports that provide information on currently issued books and their recipients. Additionally, the system should also keep users informed on whether any books are overdue. Considerably, an indispensable feature that entails proper book circulation is return, which is equally important as issuance.

Return is the opposite of issuance, and is equally important in a library management system. The system needs to keep track of all the books that are returned by students, including the date of return and any fines or fees that may have been incurred. The system should also be able to generate reports on which books have been returned, and which are still outstanding.

Transactions are the backbone of a library management system. The system needs

to keep track of all the transactions that occur within the library, including the issuance and return of books, as well as any fines or fees that are collected. The system should be able to generate reports on all transactions, including those that are outstanding or overdue.

Books are, of course, the primary entity in a library management system. The system needs to keep track of all the books that are available in the library, including their title, author, publisher, publication date, and other relevant details. The system should also be able to generate reports on which books are currently available, as well as any that are out of stock or have been lost.

Authors and genres are also important entities in a library management system. The system needs to keep track of all the authors and their associated works that are available in the library, as well as the different genres that books can be classified under. This information can be used to help users find books that are relevant to their interests, as well as to help librarians keep track of which books are popular and which are not.

Employees are another important entity in a library management system. The system needs to keep track of all the employees who work in the library, including their name, position, and other relevant details. Additionally, the system needs to keep track of the attendance of employees, as well as any absences or tardiness.

Floor is another important entity in a library management system. The system needs to keep track of the different floors in the library, as well as the different sections and shelves where books are located. This information can be used to help users find books that are relevant to their interests, as well as to help librarians keep track of which books are popular and which are not.

Finally, employee rank is an important entity in a library management system. The system needs to keep track of the different ranks that employees can attain within the library, as well as any promotions or demotions that occur. This information can be used to help manage employee performance and ensure that the library is staffed with qualified and capable individuals.

In conclusion, a library management system is an essential tool for librarians who need to manage the day-to-day operations of a library.

Word Count: 802

1.2 Purpose of the project

'Library management system' design is to create software application that helps librarians to manage their library resources more easily. This project involves creating a database to store information about books and transactions, designing a user-friendly interface and testing the system to make sure it works exactly. The goal of the project is to make library operations more efficient and give better services to library users.

1.3 Features and Functionalities

1. Student Management
⇒ student details
⇒ Issued Books, Returned Books, Transactions, Fine, Outstanding Books Details

2. Issuance and Return Management
⇒ Employees maintain data of issued books to students with book title, data of issuance and return, and any overdue fines.

3. Transactions Management
⇒ Record of Student Transactions and salary of Employees

4.Book Management
⇒ Book details

5. Author Management
⇒ Author details

6. Genre Management
⇒ Genre details

7. Employee Management
⇒ Employee Details with Ranks
⇒ Employee Attendance

8. Floor Management

⇒ Floor Details with library sections

1.4 Project Actors and Stakeholders

There are basically 3 actors of the system which are described below:

Actors:

1. Employees
2. Student
3. Administrator

1.4.1 Employees

They are staff members who will use the system to manage library resources and services, such as book circulation, cataloging, and acquisitions.

1.4.2 Student

They are the main users who will search for books, borrow and return them, and manage their own information.

1.4.3 Administrator

They are responsible for managing the overall operation of the system, including maintenance, and providing technical support to the users.

1.5 System Requirements

Table 1: System Requirements

Language	C# (3.11.0)
IDEs	Microsoft Visual Studio Community 2019 (16.11.21)
	using System.Windows.Forms;
	using System;
	using System.Collections.Generic;
	using System.ComponentModel;
	using System.Data;
	using System.Drawing;
	using System.Linq;
	using System.Text;
	using System.Threading.Tasks;
	using System.Text.RegularExpressions;
	Guna.UI2
	Pencil Tool
	Latex

1.6 ER Diagram

ER diagram can help provide a roadmap for how the entities and their relationships will be used in the database system. This can help guide the implementation of the database and ensure that it meets the requirements of the stakeholders.

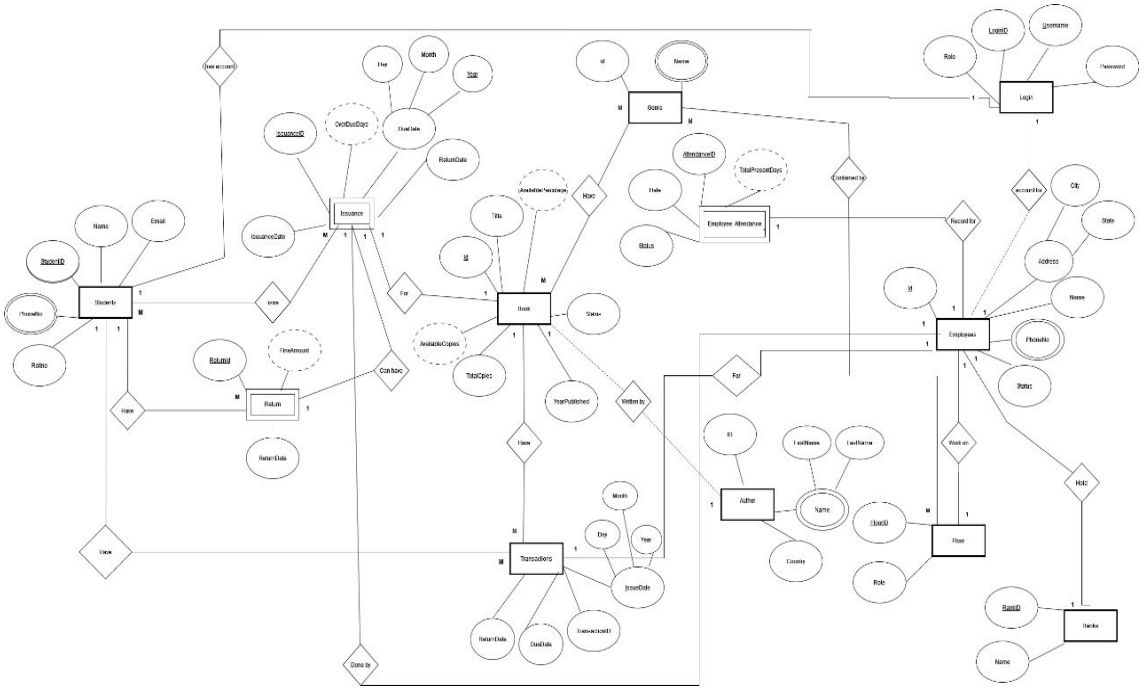


FIGURE 1: ER diagram of project

1.6.1 One to One Relationship

1. Each book is written by only one author: Book table (1) —> (1) Author table
2. Each employee belongs to only one floor: Employees table (1) —> (1) Floors table

1.6.2 One to Many Relationship

1. One student can have many book issuance: Students table (1) —> (*) Issuance table
2. One book can be issued to many students: Books table (1) —> (*) Issuance table
3. One Genre can be assigned to many books: Genre table (1) —> (*) Books table
4. One book can have many Genre Books table (1) —> (*) Categories table
5. One book can have multiple transactions: Books table (1) —> (*) Transactions table
6. One author can have written many books: Authors table (1) —> (*) Books

table
7. One employee can have many transactions: Employees table (1) —> (*) Transactions table
8. One floor can have many employees: Floors table (1) —> (*) Employees table
9. One employee can have many attendance records: Employees table (1) —> (*) Attendance table
10. One employee can issue multiple books: Employees table (1) —> (*) Issuance table

1.6.3 Many to Many Relationship

1. One student can issue many books and one book can be issued to many students: Students table () <—> () Issuance table
2. One book can have many Genre Books table and One Genre can be assigned to many books: Genre table () <—> () Books table

1.6.4 Weak Entities

1. "Issuance" table is a weak entity dependent on both "Book" and "Student" tables
2. "Returns" table is a weak entity dependent on "Issuance" table
3. "Employee Attendance"is a weak entity dependent on "Employee" table

1.6.5 Strong Relationship

1. Each book is written by one author: Books table (1) —> (1) Authors table
2. Each book is written by only one author:Book table (1) —> (1) Author Table

1.6.6 Multivalued Attributes

1. Phone no in "Employee" table
2. Name in "Genre" table
3. Name in "Author"
4. Phone no in "Students"

1.6.7 Complex Attributes

1. Due Date in Issuance Table
2. IssueDate in Transactions Table
3. Name in Author Table
4. Address in Employee Table

1.6.8 Computed Attributes

1. The "Books" table could have a computed attribute "Available Percentage," which would calculate the percentage of available copies out of total copies.

2. Books Table: Available copies (computed by subtracting the number of checked-out books from the total number of copies)
3. Issuance Table: Overdue days (computed by subtracting the due date from the return date, if available)
4. The "Returns" table could have a computed attribute "Fine Amount," which would calculate the amount of fine to be paid by the student for late return of the book.
5. The "Employee Attendance" table could have a computed attribute "Total Present Days," which would calculate the total number of days an employ

1.7 User Interface Details

Table 2: User Interface Details

Interface Id	Text Box	Drop Down	Picture Box	Table	Date Field	Buttons	Gridview	Radio Button	Check Box	Menu	Text Area	Progress Bar
I01	2	0	1	0	0	2	0	3	0	0	6	0
I02	0	0	1	0	0	4	0	0	0	0	1	0
I03	5	2	1	0	0	5	0	0	0	0	6	0
I04	0	0	1	1	0	5	0	0	0	0	16	0
I05	1	0	1	0	0	5	1	0	0	0	4	0
I06	0	0	1	0	0	8	1	0	0	0	1	0
I07	7	3	1	0	0	9	1	0	0	0	8	0
I08	0	0	1	0	0	9	1	0	0	0	16	0
I09	1	0	1	0	0	9	0	0	0	0	4	0
I10	0	0	1	0	0	9	1	0	0	0	3	0
I11	0	0	1	0	0	9	1	0	0	0	3	0
I12	0	0	1	0	0	8	1	0	0	0	3	0
I13	0	0	1	0	0	9	1	0	0	0	3	0
I14	0	0	1	0	0	6	0	0	0	0	1	0
I15	5	0	1	0	0	7	0	0	0	0	7	0
I16	0	0	1	1	0	7	0	0	0	0	15	0
I17	1	0	1	0	0	7	1	0	0	0	4	0
I18	1	1	1	0	0	7	0	0	0	0	3	0
I19	5	0	1	0	0	8	1	0	0	0	7	0
120	0	0	1	0	0	2	0	0	0	0	1	0
I21	2	0	1	0	0	3	1	0	0	0	3	0
I22	0	0	1	0	0	2	0	0	0	0	3	0
I23	0	0	1	0	0	2	1	0	0	0	1	0
I24	0	0	1	0	0	3	1	0	0	0	3	0
I25	0	0	1	0	0	2	0	0	0	0	4	0

1.8 Database Design

The design contain the whole association of all the tables with one another through the primary and foreign key constraints. The key point shows that the other table primary key has been used here.

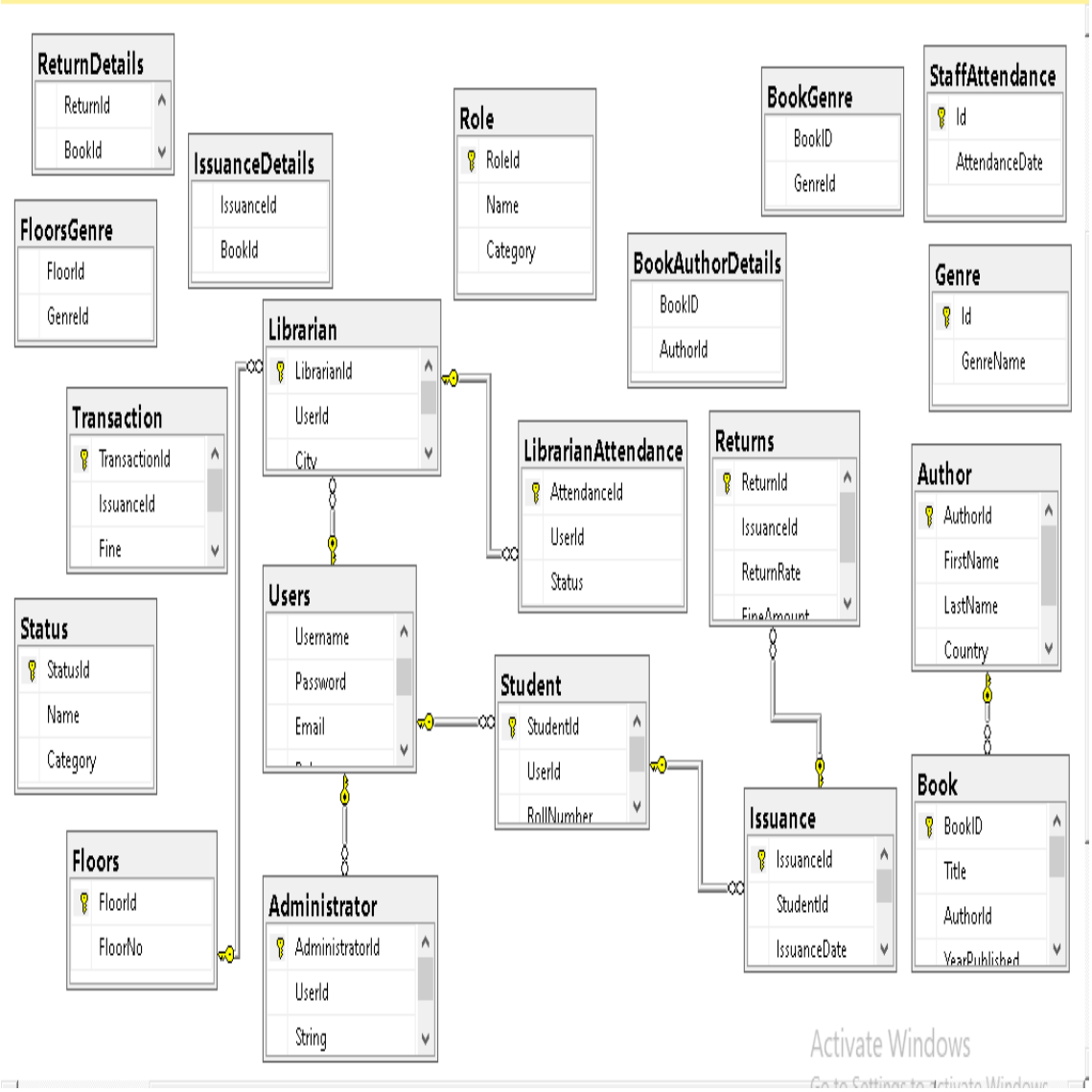


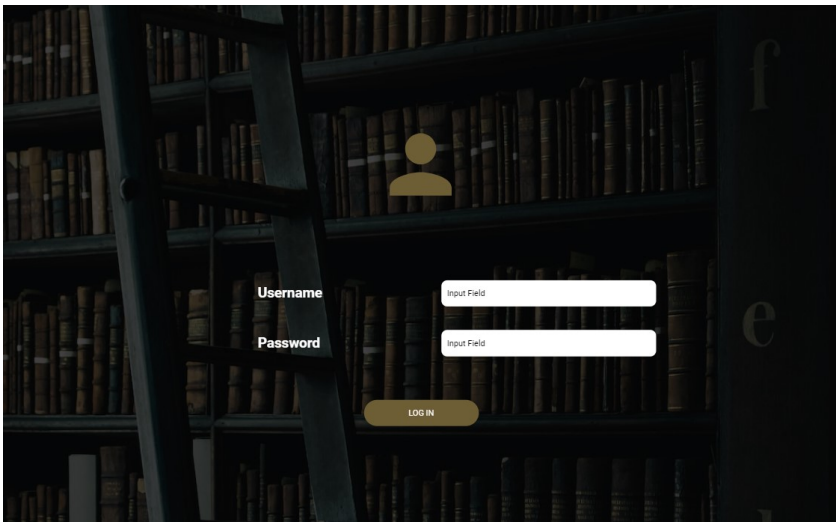
FIGURE 2: Database Design

2 Use Cases

Use cases are basically used to show that this program interface looks like this and how to use that screen and what is basically happening in the project.

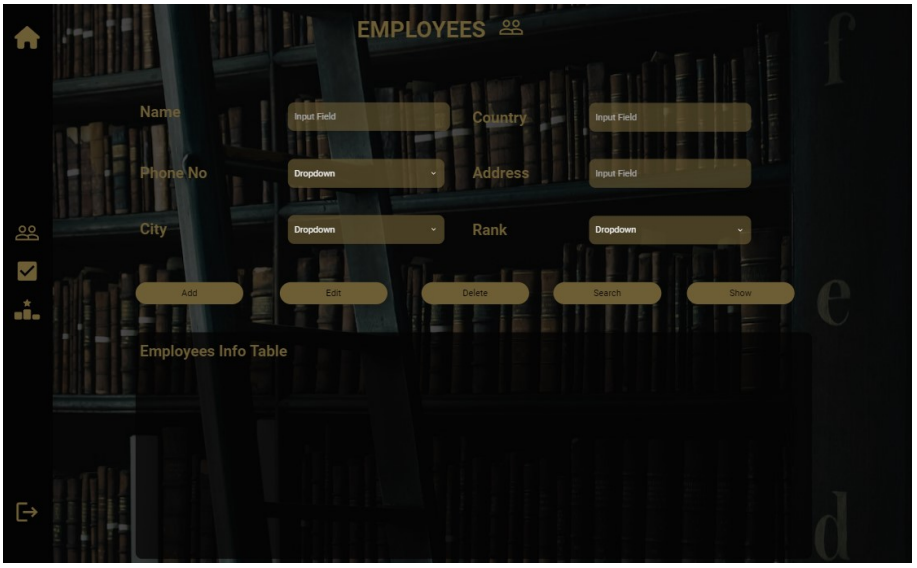
2.1 Use Case 1(Login)

Table 3: Used to describe use-case 1 of login

Use Case ID	U01
Name	Login Process
Actor	Employees/ Students/ Administrator
Description	Allows the Employees/Students/Administrator to login into the system. By entering their unique username and password, they can verify their identity and authenticate their access to the system.
Implemented GUI	 Figure 3: Log In Screen
Validations	1:- Enter unique Username in Input Field. 2:- Enter unique password in other textbox. 3:- Press the Login button to login into the program.

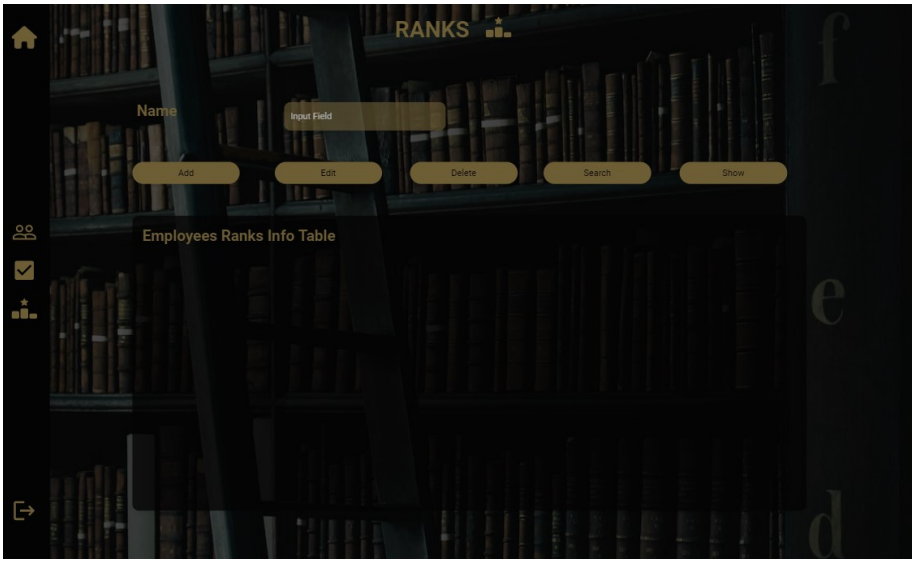
2.2 Use Case 2(Employee)

Table 4: Used to describe use-case 2 of Employee

Use Case ID	U02, U03, U04, U05
Name	Add Employee View Employee Search Employee Edit Employee
Actor	Administrator
Description	It allows administrators to input and update information about library employees. It allows to access and view employee information. It can be useful in case an employee's personal information or job details have changed.
Implemented GUI	 Figure 4: Employees Info Screen
How to use	1:- Fill all Input Fields and select all Dropdowns. 2:- He can edit employee by clicking on edit button as well. 3:- He can view employee by clicking on view button as well. 4:- By clicking on the search button, which will open a new form to enter search criteria such as name and other information. 5:- He can also add new employee by adding information in the side bar in input fields and then click on add button.

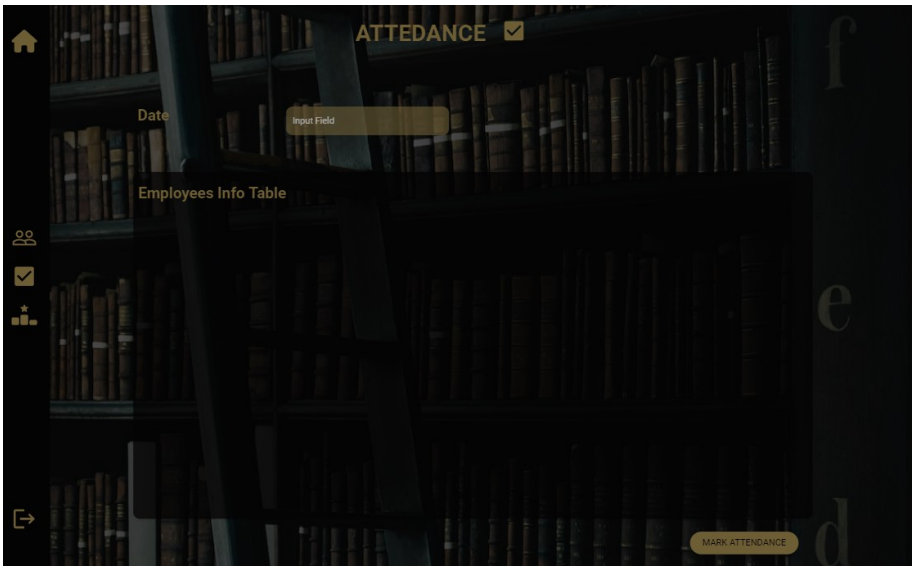
2.3 Use Case 3(Ranks)

Table 5: Used to describe use-case 3 of Ranks

Use Case ID	U06, U07, U08, U09
Name	Add Ranks View Ranks Search Ranks Edit Ranks
Actor	Administrator
Description	The ability to add, view, edit and search employee ranks is an important feature in a library management system, as it allows for the effective management and organization of library staff and resources.
Implemented GUI	 Figure 5: Employees Ranks Screen
How to use	1:- Fill the input fields. 2:- He can view ranks by clicking on view button as well. 3:- He can edit ranks by clicking on edit button as well. 4:- He can also add new ranks by adding information in the side bar in input fields and then click on add button. 5:- By clicking on the search button, which will open a new form to enter search criteria such as name.

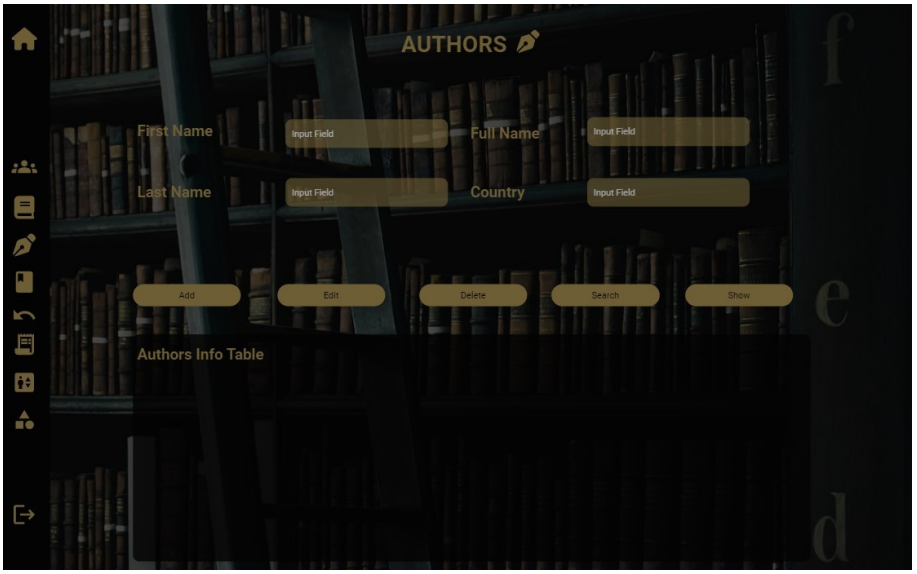
2.4 Use Case 4(Mark Attendance)

Table 6: Used to describe use-case 4 of Mark Attendance

Use Case ID	U10
Name	Mark Attendance
Actor	Administrator
Description	Marking the attendance of registered Employees.
Implemented GUI	 Figure 6: Employees Attendance Screen
How to use	1:- Employees can mark their attendance by entering the date in the date input field. 2:- By clicking the attendance button attendance can mark.

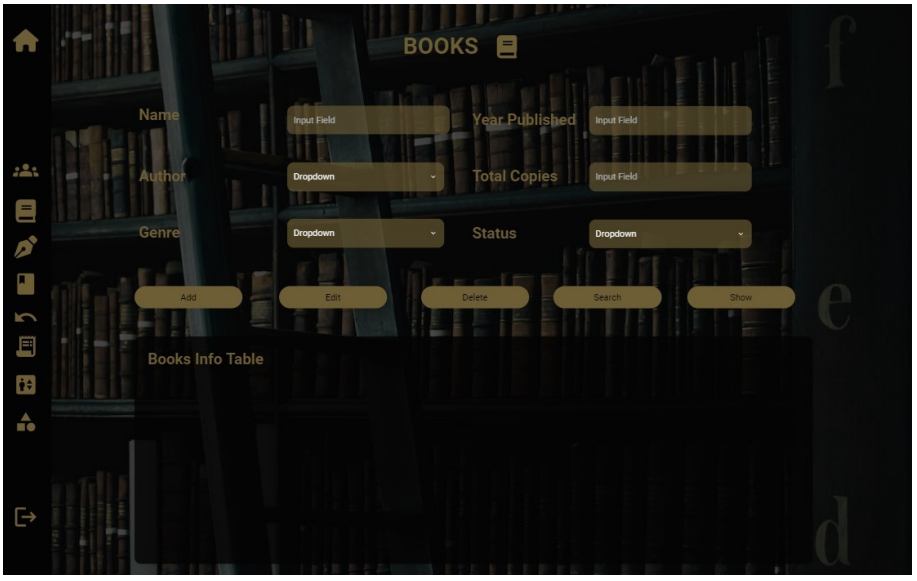
2.5 Use Case 5(Author)

Table 7: Used to describe use-case 5 of Author

Use Case ID	U11, U12, U13, U14
Name	Add Author View Author Search Author Edit Author
Actor	Employee
Description	An employee in a library management system should be able to add, view, search and edit the descriptions of book authors to ensure accurate and up-to-date information for the library's collection.
Implemented GUI	 Figure 7: Authors Screen
How to use	1:- Employee can only fill in those fields with strings. 2:- He can view the Author by click on view button as well. 3:- He can edit the Author by click on edit button as well. 4:- He can also add new Author by adding information in the side bar in input fields and then click on add button. 5:- By clicking on the search button, which will open a new form to enter search criteria such as name.

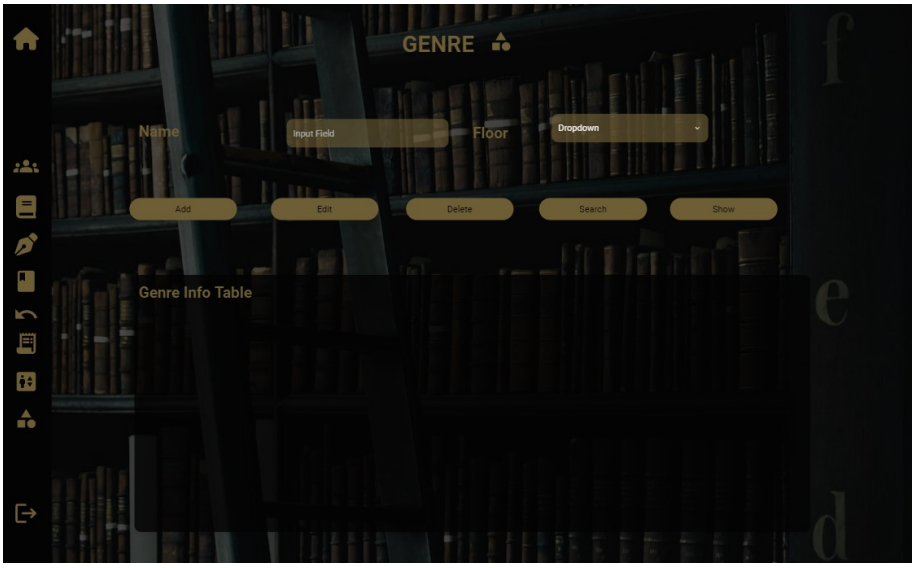
2.6 Use Case 6(Books)

Table 8: Used to describe use-case 6 of Books

Use Case ID	U15, U16, U17, U18
Name	Add Books View Books Search Books Edit Books
Actor	Employee
Description	Employee should be able to view all books that have been added to the library and also have the ability to add new books and update and search any book information, including its name.
Implemented GUI	 Figure 8: Books Screen
How to use	1:- He will be able to fill all input fields. 2:- He will also be able to select all dropdowns. 3:- He can view the books by click on view icon as well. 3:- He can edit any book's information by clicking edit button. 4:- He can also add new books by adding information in the side bar in textboxes and then click on add button. 5:- By clicking on the search button, which will open a new form to enter search criteria such as name.

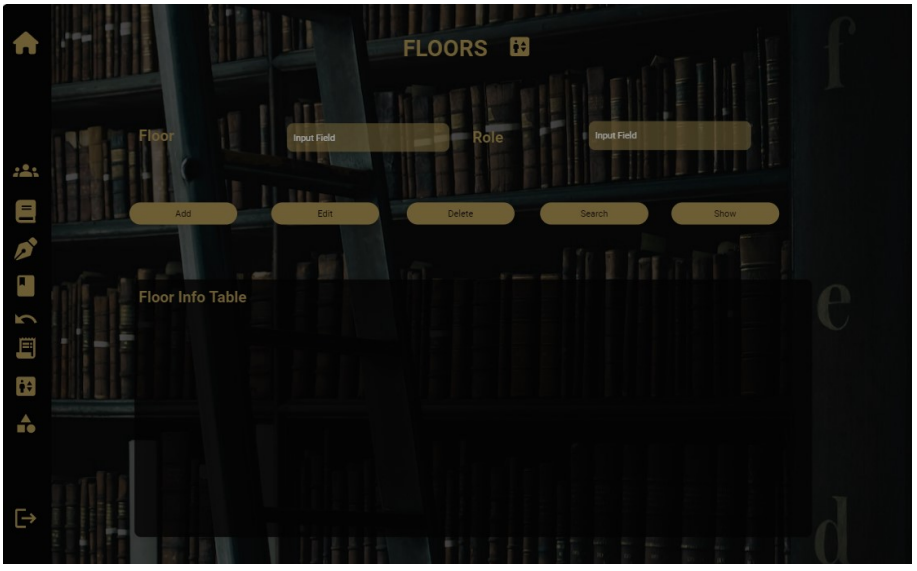
2.7 Use Case 7(Genre)

Table 9: Used to describe use-case 7 of Genre

Use Case ID	U19, U20, U21, U22
Name	Add Genre View Genre Search Genre Edit Genre
Actor	Employee
Description	Employee can add, view and edit genre descriptions in library management system to help categorize and classify books based on their themes and styles. This allows users to easily find books of their interest and helps librarians to keep track of the popularity of different genres.
Implemented GUI	 Figure 9: Genre Screen
How to use	1:- He will be able to fill input fields. 2:- He will also be able to select dropdown. 3:- He can view the genre by click on view icon as well. 4:- He can edit any genre's information by clicking edit button. 5:- He can also add new genre by adding information in the side bar in textboxes and then click on add button.

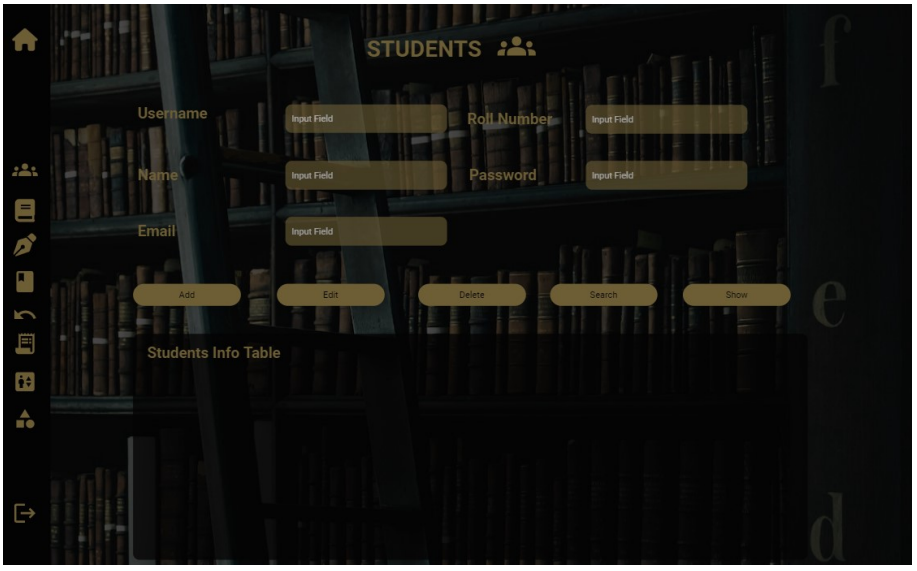
2.8 Use Case 8(Floors)

Table 10: Used to describe use-case 8 of Floors

Use Case ID	U23, U24, U25, U26
Name	Add Floors View Floors Search Floors Edit Floors
Actor	Employee
Description	Employee can add, view, and edit floors in the library management system, which will help in organizing the physical layout of the library. This feature can help to easily locate books in the library based on their floor location. Employee can also assign books to specific floors based on their category and genre.
Implemented GUI	 Figure 10: Floor Screen
How to use	1:- He will be able to fill input fields. 2:- He can view the floors by click on view icon as well. 3:- He can edit any floor by clicking edit button. 4:- He can also add new floor by adding information in the side bar in textboxes and then click on add button.

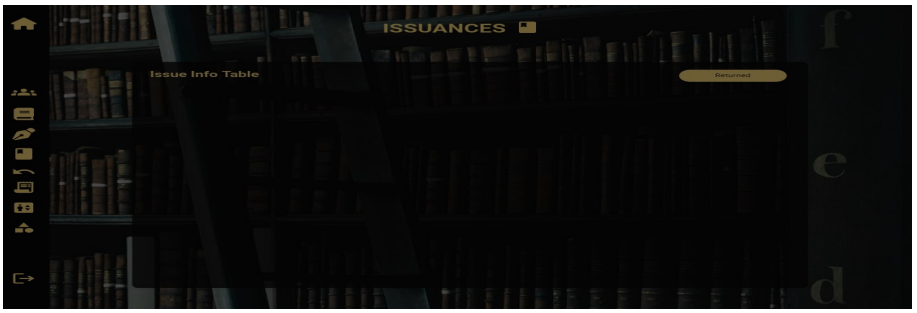
2.9 Use Case 9(Student)

Table 11: Used to describe use-case 9 of Student

Use Case ID	U27, U28, U29, U30
Name	Add Students View Students Search Student Edit Students
Actor	Employee
Description	Employee can add, view, and edit student profiles in the library management system to keep track of their borrowing history and preferences. This feature allows for efficient management of library resources and personalized services for students.
Implemented GUI	 Figure 11: Student Screen
How to use and validations	1:- Username, password and email must be unique. 2:- Roll Number contains only digits and must be unique. 3:- He can edit and view any Student's information by clicking edit or show button. 4:- He can also add new Student by adding information. 5:- By clicking on the search button, which will open a new form to enter search criteria such as name.

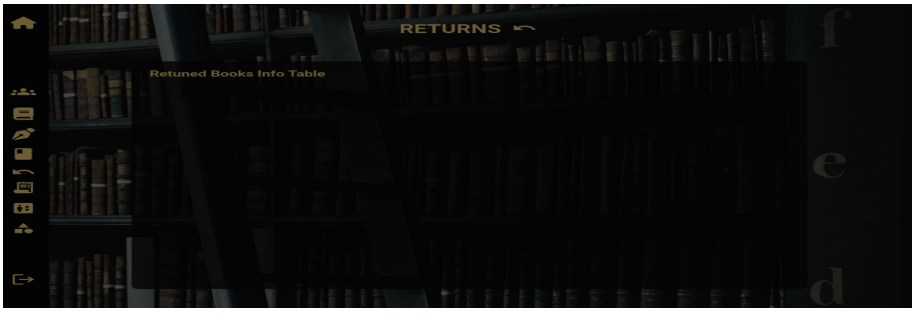
2.10 Use Case 10(Issuance and Return)

Table 12: Used to describe use-case 10 of Issuance and Return

Use Case ID	U31
Name	Issuance and Return
Actor	Employee
Description	Maintaining Issuance and Return of Books.
Implemented GUI	 Figure 12: Issuances Screen

2.11 Use Case 11(Returned Books)

Table 13: Used to describe use-case 11 of Returned Books

Use Case ID	U32
Name	View Returned Books
Actor	Employee
Description	View Returned Books by Students.
Implemented GUI	 Figure 13: Returned Books Screen

3 Transactions

4 Indexes

4.1 Queries

- CREATE INDEX idx_book ON Book(BookID);
- CREATE INDEX idx_author ON Author(AuthorId);
- CREATE INDEX idx_genre ON Genre(Id);
- CREATE INDEX idx_floor ON Floors(FloorId);
- CREATE INDEX idx_student ON Student(StudentId);

5 Views

5.1 Queries

- **CREATE** VIEW [dbo].[AuthorDetails] WITH SCHEMABINDING AS
  SELECT a.FirstName, a.LastName, a.Country,
  COUNT(bad.BookID) AS BookCount
  FROM dbo.[Author] a
  INNER JOIN dbo.[BookAuthorDetails] bad
  ON a.AuthorID = bad.AuthorID
  GROUP BY a.FirstName, a.LastName, a.Country;
- **CREATE** VIEW [dbo].[AvailableBooks] WITH SCHEMABINDING AS
  SELECT b.Title, a.FirstName, a.LastName, g.GenreName
  FROM dbo.[Book] b
  INNER JOIN dbo.[BookAuthorDetails] bad
  ON b.BookID = bad.BookID
  INNER JOIN dbo.[Author] a
  ON bad.AuthorID = a.AuthorID
  INNER JOIN dbo.[BookGenre] bg
  ON b.BookID = bg.BookID
  INNER JOIN dbo.[Genre] g
  ON bg.GenreID = g.ID
  WHERE b.AvailableCopies > 0;

- **CREATE** VIEW [dbo].[BookDetails] WITH SCHEMABINDING AS
  SELECT b.Title, a.FirstName, a.LastName, g.GenreName, b.TotalCopies
  FROM dbo.[Book] b
  INNER JOIN dbo.[BookAuthorDetails] bad
  ON b.BookID = bad.BookID
  INNER JOIN dbo.[Author] a
  ON bad.AuthorID = a.AuthorID
  INNER JOIN dbo.[BookGenre] bg
  ON b.BookID = bg.BookID
  INNER JOIN dbo.[Genre] g
  ON bg.GenreID = g.ID;

- **CREATE** VIEW [dbo].[BookGenres] WITH SCHEMABINDING AS
  SELECT b.Title, g.GenreName
  FROM dbo.[Book] b
  INNER JOIN dbo.[BookGenre] bg
  ON b.BookID = bg.BookID
  INNER JOIN dbo.[Genre] g
  ON bg.GenreID = g.ID;

- **CREATE** VIEW [dbo].[BookIssuanceCount] WITH SCHEMABINDING AS
  SELECT b.Title, COUNT(i.BookID) AS IssuanceCount
  FROM dbo.[Book] b
  INNER JOIN dbo.[IssuanceDetails] i
  ON b.BookID = i.BookID
  GROUP BY b.Title;

- **CREATE** VIEW [dbo].[BookStatus] WITH SCHEMABINDING AS
  SELECT Title, Status
  FROM dbo.[Book];

- **CREATE** VIEW [dbo].[BookYear] WITH SCHEMABINDING AS
  SELECT Title, YearPublished
  FROM dbo.[Book];

- **CREATE** VIEW [dbo].[LibrarianDetails] WITH SCHEMABINDING AS
  SELECT u.Name, l.City, l.Country, f.FloorNo
  FROM dbo.[Librarian] l
  INNER JOIN dbo.[Users] u

ON l.UserID = u.UserID
INNER JOIN dbo.[Floors] f
ON l.FloorID = f.FloorID;

- **CREATE VIEW** [dbo].[RecentBooks] WITH SCHEMABINDING AS
  SELECT b.Title, a.FirstName, a.LastName, g.GenreName
  FROM dbo.[Book] b
  INNER JOIN dbo.[BookAuthorDetails] bad
  ON b.BookID = bad.BookID
  INNER JOIN dbo.[Author] a
  ON bad.AuthorID = a.AuthorID
  INNER JOIN dbo.[BookGenre] bg
  ON b.BookID = bg.BookID
  INNER JOIN dbo.[Genre] g
  ON bg.GenreID = g.ID
  WHERE b.YearPublished >= YEAR(DATEADD(year, -5, GETDATE()));

- **CREATE** VIEW [dbo].[StudentDetails] WITH SCHEMABINDING AS
  SELECT s.StudentID, u.Name, s.RollNumber
  FROM dbo.[Student] s
  INNER JOIN dbo.[Users] u
  ON s.UserID = u.UserID;

6 Stored Procedures

6.1 Queries

- `CREATE PROCEDURE usp_UpdateLibrarian`
  @LibrarianId int,
  @UserId int,
  @City varchar(255),
  @Country varchar(255),
  @Address varchar(255),
  @FloorId int,
  @Status varchar(20)
  AS
  BEGIN
  SET NOCOUNT ON;
  UPDATE Librarian
  SET UserId = @UserId, City = @City, Country = @Country,

Address = @Address, FloorId = @FloorId, Status = @Status
WHERE LibrarianId = @LibrarianId END

- `CREATE PROCEDURE usp_InsertLibrarian`
  @UserId int,
  @City varchar(50),
  @Country varchar(50),
  @Address varchar(100),
  @FloorId int,
  @Status varchar(50)
  AS
  BEGIN
  SET NOCOUNT ON;
  INSERT INTO Librarian (UserId, City, Country, Address, FloorId, Status)
  VALUES (@UserId, @City, @Country, @Address, @FloorId, @Status)
  END

- `CREATE PROCEDURE asp_UpdateBook`
  @BookID INT,
  @Title NVARCHAR(255),
  @YearPublished INT,
  @TotalCopies INT,
  @AvaliableCopies INT,
  @status varchar(20)
  AS
  BEGIN
  SET NOCOUNT ON;
  UPDATE Book
  SET Title = @Title, YearPublished = @YearPublished,
  TotalCopies = @TotalCopies, AvailableCopies = @AvaliableCopies,
  status = @status WHERE BookID = @BookID; END

- `CREATE PROCEDURE usp_InsertBook` @Title NVARCHAR(255),
  @YearPublished INT,
  @TotalCopies INT,
  @AvaliableCopies INT,
  @status varchar(20)
  AS
  BEGIN

SET NOCOUNT ON;
INSERT INTO Book (Title, YearPublished, TotalCopies, AvailableCopies, Status)
VALUES (@Title, @YearPublished, @TotalCopies, @AvaliableCopies, @status);
END

- `CREATE PROCEDURE usp_UpdateGenre`
  @Id INT,
  @GenreName VARCHAR(255)
  AS
  BEGIN
  SET NOCOUNT ON;
  UPDATE Genre
  SET GenreName = @GenreName
  WHERE Id = @Id; END

- `CREATE PROCEDURE usp_InsertGenre`
  @GenreName VARCHAR(255)
  AS
  BEGIN
  SET NOCOUNT ON;
  INSERT INTO Genre (GenreName)
  VALUES (@GenreName) END

7 Triggers

7.1 Query

- `CREATE TRIGGER update_bookavailablecopies`
  ON IssuanceDetails AFTER INSERT AS
  BEGIN
  SET NOCOUNT ON;
  DECLARE @bookid int

  SELECT @bookid=BookId
  FROM IssuanceDetails
  WHERE IssuanceId = (SELECT MAX(IssuanceId)
  FROM IssuanceDetails);

```
DECLARE @availablecopies int
Select @availablecopies=AvailableCopies
from Book
where BookID=@bookid;
if (@availablecopies ! = 0)
UPDATE Book SET AvailableCopies = @availablecopies - 1,
status = 'Available' WHERE BookID = @bookid
ELSE IF(@availablecopies = 0)
UPDATE Book SET AvailableCopies = @availablecopies ,
status = 'UnAvailable' WHERE BookID = @bookid

END;
```

8 Limitations

Following are the limitations of the project:

1. More views could have been added.
2. More Triggers could be inserted.
3. Transactions without delay can be achieved.

9 Future Work

The future work that can be done on the project are enlisted below:

1. Making better Graphical User Interfaces with more features.
2. The project can be linked with the web .

10 Conclusion

A library management system is a crucial resource for librarians to manage book circulation, borrower tracking, and transactions. The system should consider various entities such as books, students, employees, authors, genres, and floors. In the future, integrating cutting-edge technologies such as AI and machine learning can improve the user experience and streamline operations. By continually updating the system, librarians can ensure that their libraries remain pertinent and essential resources for their communities.