KHULNA UNIVERSITY

Course Title: Database Systems Project/Fieldwork

Course No: CSE 3102

Library Management System Project Final Report

Submitted by:

S M Musfikur Rahman (SID: 190224)
Masud Karim Omi (SID: 200220)
Utsa Debnath (SID : 200242)
3rd year, 1st term
Computer Science and Engineering Discipline, Khulna University

Submitted to:

Dr. Md. Anisur Rahman
Professor
Computer Science and Engineering Discipline, Khulna University

Submission Date : 11-04-2023

LIBRARY MANAGEMENT SYSTEM

Table of Contents

1. INTRODUCTION

This chapter gives an overview of the aim, objectives, background and operation environment of the system.

1.0 OVERVIEW

Performing library management manually is quite difficult, as is maintaining its records in order. These are the issues that the Library Management System should address and answer.
The Library Management System (LMS) is a program that helps librarians manage a library by managing the daily book transaction and managing the member, books record more efficiently. It can improve the management of the book property in the library.

This library management system is mainly used by librarians and library admin. Normal librarians can manage the member maintenance module, book maintenance module and also the most important module in a library which is the book transaction module. Besides that, the library management system also allows users to manage the publisher as well as the lost book module. On the other hand, another type of user which is admin-level staff can handle the staff module and view the report module.

The methodology I used to develop this system is the waterfall model. Thus, the report's chapter included system planning, requirement analysis, system design, programming, system testing, and evaluation of the project.

This study was undertaken to design an advanced database for a library management system for the Seminar Library CSE discipline at Khulna University. Library Management System was a system that was designed to keep records of how librarians and book bankers handle and process data and ease the problem of the receiving and issuing of books to students. The management system involved all those activities designed to control the life cycle of collection and retrieval of records quickly.

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

1. Online book issue
2. Request the librarian for providing new books
3. Student login page where student can and books issued by him/her and date of return.
4. Students can check the availability of books.
5. The Library Management System's major goal is to keep track of information such as books and borrowers' information. It manages the data, such as Book borrowers' basic info, as well as the librarians.
6. Solve the problems that arose from the current system.
7. Database interface that was appropriately used for records management in the library.

8. The system is developed to cope with the current issues and problems of the library. The system can add users, and validate the user and is also bug-free.
9. After the computerized system is implemented less human force will be required to maintain the library thus reducing the overall cost.
10. Save time that librarians can search records by using a few clicks of the mouse and a few search keywords thus saving them valuable time.
11. The proposed system will help increase the efficiency and effectiveness of the department's services to the institution and the customers.

1.2 BACKGROUND OF THE PROJECT

Library Management System is an application which refers to library systems which are generally small or medium in size. It is used by a librarian to manage the library using a computerized system where he/she can record various transactions like an issue of books, the return of books, the addition of new books, the addition of new students etc. Books and student maintenance modules are also included in this system which would keep track of the students using the library and also a detailed description of the books a library contains. With this computerized system, there will be no loss of book records or member records which generally happens when a non-computerized system is used. In addition, the report module is also included in Library Management System. If the user's position is admin, the user can generate different kinds of reports like lists of students registered, lists of books, and issue and return reports. All these modules can help librarian to manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

1.3 MOTIVATION FOR THE PROJECT

The number of users is changing day by day. To cope with this, the KU library was automating some of the services to hasten the collection and retrieval of records. This action was meeting problems like the insufficient staff who were overworked, the classification of books in the book banks with different methods which was cumbersome, inadequate reserve section for storing books, and the problem of mismanagement of records since the system was manual affected the level of efficiency of the library. These problems provided the focus for the study and necessitate an investigation to arrest the situation before it became worse. Therefore, the researcher aimed at designing and developing a Library Management System that was used to handle the library services efficiently.

2 SYSTEM ANALYSIS

In this section, we will discuss and analyze the developing process of the Library Management System and compare existing and proposed systems. Besides that, existing vs proposed provides a view of how the proposed system will be more efficient than the existing one.

2.1 PRODUCT DESCRIPTION

Library Management System is a computerized system which helps the user(librarian) to manage the library's daily activity in electronic format. It reduces the risk of paperwork such as file loss, file damage and time consumption. It can help user to manage transaction or record more effectively and timesaving.

2.2 EXISTING VS PROPOSED SYSTEM

1. Existing system does not have any facility for student login whereas the proposed system will have a facility for student login.
2. Existing system does not have any facility of an online notice board where the description of workshops happening in our university as well as nearby Universities is being provided.
3. Existing system does not have any facility to generate student reports as well as book issue reports whereas the proposed system provides librarians with a tool to generate reports.
4. Existing system does not have any facility for book requests and suggestions whereas in the proposed system after logging in to their accounts student can request books as well as provide suggestions to improve the library.
5. Existing system does not have any fine system for losing a book but the proposed system has the feature of a fine system.

2.3 DESCRIPTION:

The List of Modules that the Library Management System should have:

REGISTER NEW STUDENT

Description of feature:

This feature can be performed by all students to register a new user to create an account.

Functional requirements:

1. System must be able to verify the information.
2. System must be able to delete information if the information is wrong.

REGISTER NEW BOOK

Description of feature:

This feature allows the addition of new books to the library.

Functional requirements:

1. System must be able to verify information
2. System must be able to enter the number of copies into the table.
3. System must be able to not allow two books having the same book id

SEARCH BOOK

Description of feature:

This feature is found in the book maintenance part. We can search books based on book id, book name, publication or by author name.

Functional requirements:

1. System must be able to search the database based on a select search type.
2. System must be able to filter books based on the keyword entered.
3. System must be able to show the filtered book in the table view.

ISSUE BOOKS AND RETURN BOOKS

Description of feature:

This feature allows us to issue and return books and also view reports of the book issued.

Functional requirements:

1. System must be able to enter issue information in the database.
2. System must be able to update the number of books.
3. System must be able to search if the book is available or not before issuing books
4. System should be able to enter issue and return date information.

LOST BOOK

Description of feature:

We can search books based on book id, book name, publication, author name or lost id which book are lost.

Functional requirements:

1. System must be able to search the database based on a select search type.
2. System must be able to filter books based on the keyword entered.
3. System must be able to show the filtered book in the table view.

2.4 SOFTWARE AND TOOLS

SOFTWARE

- Operating system: Windows 11 Home
- Database: MySQL
- Development tools: Microsoft Visual Studio Code
- Programming language- HTML, CSS, Bootstrap, PHP

Here HTML, CSS, Bootstrap, and PHP will be used to design the front-end site and User Interface. PHP Web Programming Language will be used here to access the database. Here PHP Language will mediator between our System web page and Database. To execute Database Queries SQL will be used. MySQL Workbench is a Database Management Software to manage data.

HARDWARE

- Processor: Intel(R) Core(TM) i5-9500K CPU @ 3.90GHz
- Ram: 8 GB

2.5 SAMPLE QUERIES:

In our system there will be 2 types of Queries:

Select Query:

A select query retrieves data from a database. A select query is used to search for any information from Database. We can Fetch and Display data from Corresponding as well as Related Data Tables from the Database. This query will not change any data.

Action Query:

An action query asks for additional operations on data, such as Inserting, Updating, Deleting, Altering or any other forms of data manipulation. Action query will be used to Change and Manipulate data in Database. This Query can influence data.

SOME EXAMPLES OF SAMPLE QUERIES IN OUR SYSTEM:

1. Retrieve all books from the database:

```
SELECT * FROM books;
```

2. Retrieve all books published after 2010:

```
SELECT * FROM books WHERE publication_year > 2010;
```

3. Retrieve all books with the title "Software Engineering":

```
SELECT * FROM books WHERE title = 'Software Engineering;
```

4. Add a new book to the database:

```
INSERT INTO books (book_id, author, publication_year, publisher, book_name, term,
edition, category);
VALUES (5051, 'Roger Pressman', 1991, ‘Mc Graw Hil Education’, 'Software
Engineering',’1st’,’10th’,’ CSE’);
```

Updating Books:

5. UPDATE books SET publication_year = 1999 WHERE title = ‘Analog Communication';

3. SYSTEM DESIGN

3.1 SCHEMA DESIGN

```
CREATE TABLE Student (
    student_id INT PRIMARY KEY,
    student_name VARCHAR(50),
    student_email VARCHAR(50),
   year INT,
   term INT
);

CREATE TABLE Librarian (
    librarian_id INT PRIMARY KEY,
    librarian_name VARCHAR(50),
    librarian_email VARCHAR(50)
);

CREATE TABLE Book (
    book_id INT PRIMARY KEY,
    author VARCHAR(50),
    publication_year INT,
    publisher VARCHAR(50),
    book_name VARCHAR(100),
    term VARCHAR(50),
    edition VARCHAR(50),
    category VARCHAR(50)
);
```

```
CREATE TABLE BorrowedBooks (
    borrow_id INT PRIMARY KEY,
    student_id INT,
    book_id INT,
    borrow_date DATE,
    due_date DATE,
    FOREIGN KEY (student_id) REFERENCES Student(student_id),
    FOREIGN KEY (book_id) REFERENCES Book(book_id)
);

CREATE TABLE Return_Books (
    return_id INT PRIMARY KEY,
    borrow_id INT,
    return_date DATE,
    fine INT,
    FOREIGN KEY (borrow_id) REFERENCES BorrowedBooks(borrow_id)
);

CREATE TABLE Lost_Books (
    lost_id INT PRIMARY KEY,
    borrow_id INT,
    lost_date DATE,
    fine INT,
    FOREIGN KEY (borrow_id) REFERENCES BorrowedBooks(borrow_id)
);

CREATE TABLE Report (
    report_id INT PRIMARY KEY,
    student_id INT,
    librarian_id INT,
    book_id INT,
    borrow_date DATE,
    due_date DATE,
    return_date DATE,
    fine INT,
    FOREIGN KEY (student_id) REFERENCES Student(student_id),
    FOREIGN KEY (librarian_id) REFERENCES Librarian(librarian_id),
    FOREIGN KEY (book_id) REFERENCES Book(book_id)
);
```

3.2. NORMALIZATION

STUDENT TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

LIBRARIAN TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

BOOKS TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

BORROWEDBOOKS TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

RETURNBOOK TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

LOSTBOOKS TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- All non-key attributes are fully functionally dependent on the primary key. So, the table is in 2NF.

THIRD NORMAL FORM (3NF):

- The table already is in 2NF
- All non-key attributes are not transitively dependent on the primary key.
- So, the table is in 3NF.

REPORT TABLE

FIRST NORMAL FORM (1NF) :

- The table has a primary key that uniquely identifies each record.
- All attribute values are atomic, meaning they cannot be further divided. So, the table is in 1NF.

SECOND NORMAL FORM (2NF)

- The table already is in 1NF
- The partial dependencies are -
- borrow_id -> Fine
- borrow_id -> borrow_date, due_date, and return_date

Create table :

Fine(borrow_id , fine)

Borrow(borrow_id, borrow_date, due_date, and return_date); So, now it is in 2NF.

THIRD NORMAL FORM (3NF):

- The table is in 2NF
- Now, after making it 2NF, All non-key attributes have no transitive dependencies. So, the table is in 3NF.

AFTER THE NORMALIZATION PROCESS, THE NEW SCHEMA WITH THE MODIFIED TABLES WOULD BE:

Student (student_id, student_name , student_email ,year INT, term INT);

Librarian (librarian_id, librarian_name , librarian_email);

Book (book_id, author, publication_year, publisher, book_name, term, edition, category_id,);

BorrowedBooks (borrow_id, student_id, book_id);

Fine (borrow_id,fine);

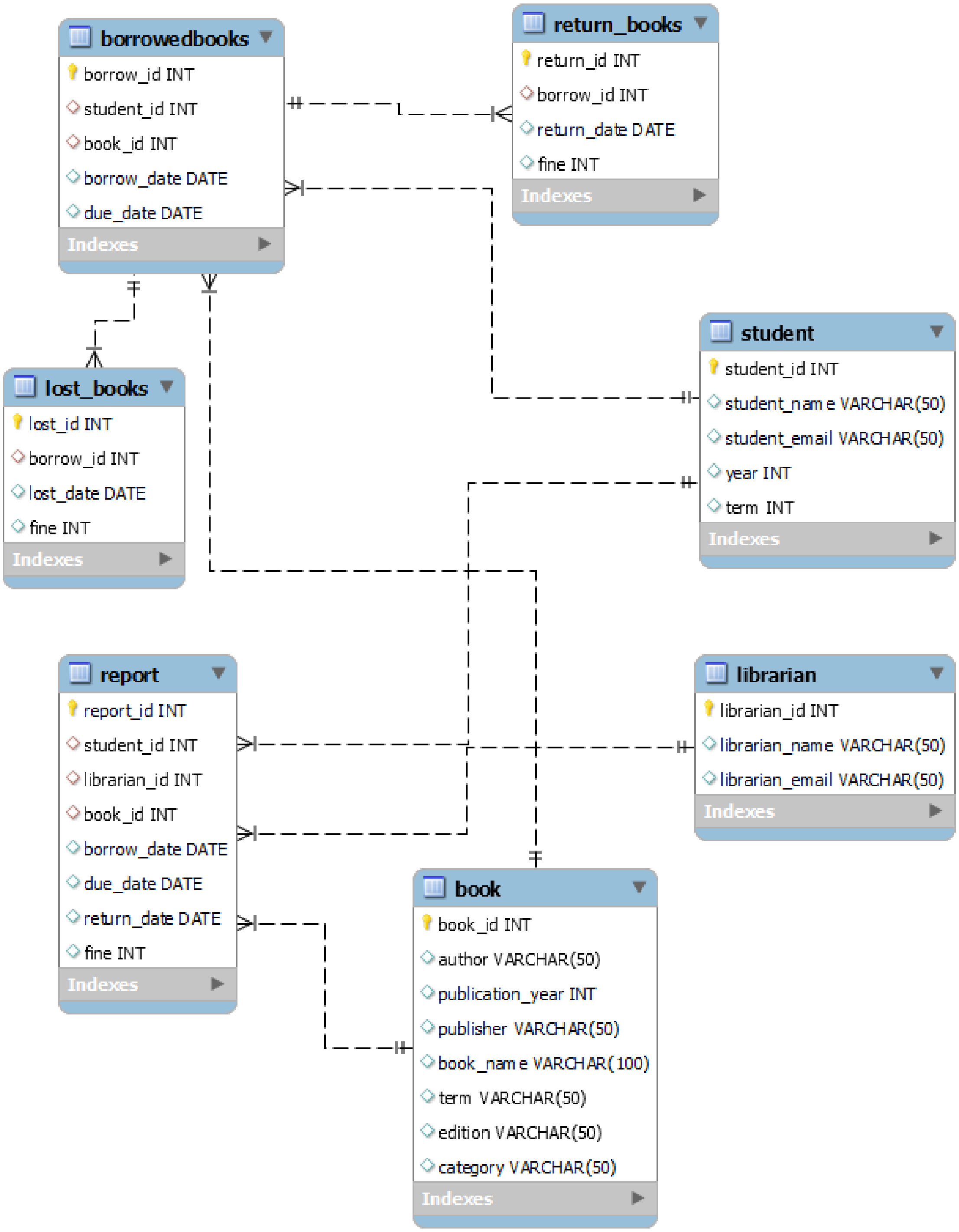
Borrow (borrow_id, borrow_date, due_date, return_date);

Return_Books (return_id, borrow_id, return_date);

Lost_Books (lost_id, borrow_id, lost_date);

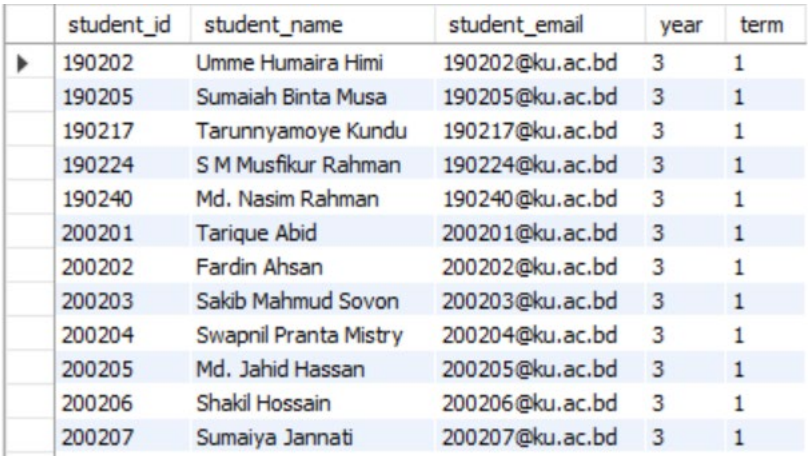
Report (report_id, student_id, librarian_id, book_id, borrow_id);

3.2. SCHEMA DIAGRAM



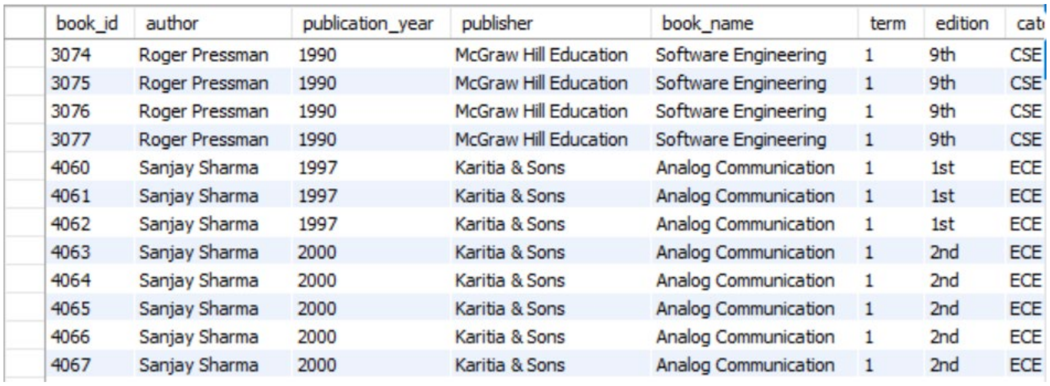
3.3. DATABASE TABLE

STUDENT TABLE:



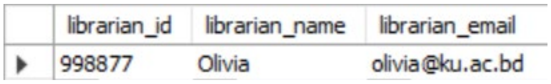
student_id	student_name	student_email	year	term
190202	Umme Humaira Himi	190202@ku.ac.bd	3	1
190205	Sumaiah Binta Musa	190205@ku.ac.bd	3	1
190217	Tarunnyamoye Kundu	190217@ku.ac.bd	3	1
190224	S M Musfikur Rahman	190224@ku.ac.bd	3	1
190240	Md. Nasim Rahman	190240@ku.ac.bd	3	1
200201	Tarique Abid	200201@ku.ac.bd	3	1
200202	Fardin Ahsan	200202@ku.ac.bd	3	1
200203	Sakib Mahmud Sovon	200203@ku.ac.bd	3	1
200204	Swapnil Pranta Mistry	200204@ku.ac.bd	3	1
200205	Md. Jahid Hassan	200205@ku.ac.bd	3	1
200206	Shakil Hossain	200206@ku.ac.bd	3	1
200207	Sumaiya Jannati	200207@ku.ac.bd	3	1

BOOKS TABLE:



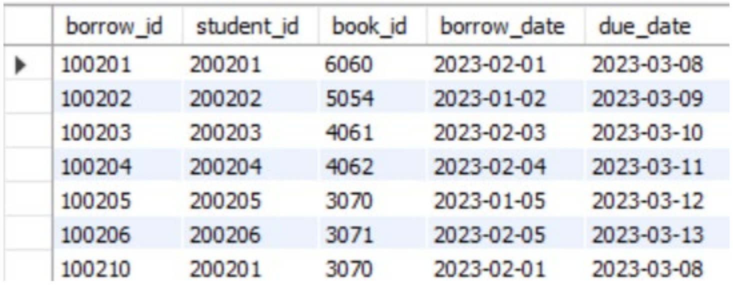
book_id	author	publication_year	publisher	book_name	term	edition	cat
3074	Roger Pressman	1990	McGraw Hill Education	Software Engineering	1	9th	CSE
3075	Roger Pressman	1990	McGraw Hill Education	Software Engineering	1	9th	CSE
3076	Roger Pressman	1990	McGraw Hill Education	Software Engineering	1	9th	CSE
3077	Roger Pressman	1990	McGraw Hill Education	Software Engineering	1	9th	CSE
4060	Sanjay Sharma	1997	Karitia & Sons	Analog Communication	1	1st	ECE
4061	Sanjay Sharma	1997	Karitia & Sons	Analog Communication	1	1st	ECE
4062	Sanjay Sharma	1997	Karitia & Sons	Analog Communication	1	1st	ECE
4063	Sanjay Sharma	2000	Karitia & Sons	Analog Communication	1	2nd	ECE
4064	Sanjay Sharma	2000	Karitia & Sons	Analog Communication	1	2nd	ECE
4065	Sanjay Sharma	2000	Karitia & Sons	Analog Communication	1	2nd	ECE
4066	Sanjay Sharma	2000	Karitia & Sons	Analog Communication	1	2nd	ECE
4067	Sanjay Sharma	2000	Karitia & Sons	Analog Communication	1	2nd	ECE

LIBRARIAN TABLE:



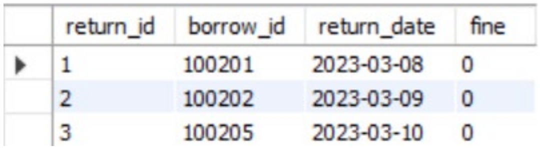
librarian_id	librarian_name	librarian_email
998877	Olivia	olivia@ku.ac.bd

BORROWED BOOKS TABLE:



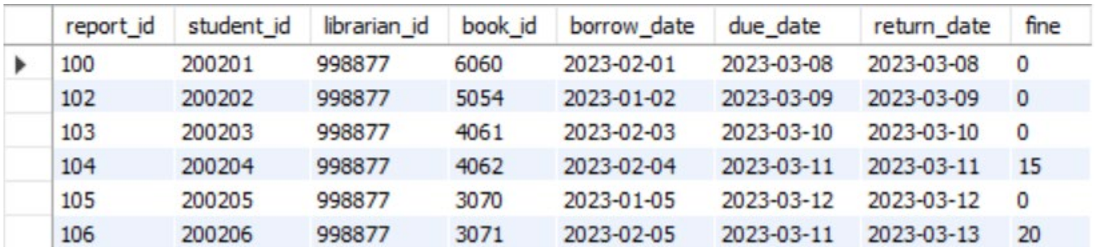
borrow_id	student_id	book_id	borrow_date	due_date
100201	200201	6060	2023-02-01	2023-03-08
100202	200202	5054	2023-01-02	2023-03-09
100203	200203	4061	2023-02-03	2023-03-10
100204	200204	4062	2023-02-04	2023-03-11
100205	200205	3070	2023-01-05	2023-03-12
100206	200206	3071	2023-02-05	2023-03-13
100210	200201	3070	2023-02-01	2023-03-08

RETURN BOOK TABLE:



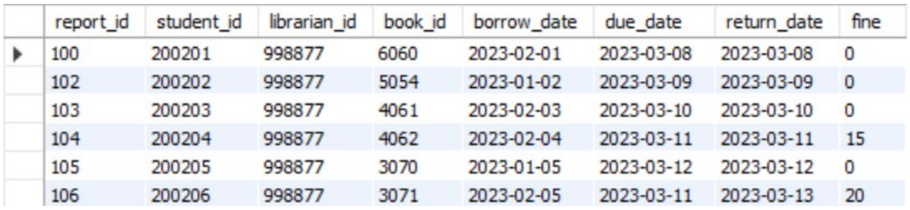
return_id	borrow_id	return_date	fine
1	100201	2023-03-08	0
2	100202	2023-03-09	0
3	100205	2023-03-10	0

REPORT TABLE:



report_id	student_id	librarian_id	book_id	borrow_date	due_date	return_date	fine
100	200201	998877	6060	2023-02-01	2023-03-08	2023-03-08	0
102	200202	998877	5054	2023-01-02	2023-03-09	2023-03-09	0
103	200203	998877	4061	2023-02-03	2023-03-10	2023-03-10	0
104	200204	998877	4062	2023-02-04	2023-03-11	2023-03-11	15
105	200205	998877	3070	2023-01-05	2023-03-12	2023-03-12	0
106	200206	998877	3071	2023-02-05	2023-03-11	2023-03-13	20

LOST BOOK TABLE:



report_id	student_id	librarian_id	book_id	borrow_date	due_date	return_date	fine
100	200201	998877	6060	2023-02-01	2023-03-08	2023-03-08	0
102	200202	998877	5054	2023-01-02	2023-03-09	2023-03-09	0
103	200203	998877	4061	2023-02-03	2023-03-10	2023-03-10	0
104	200204	998877	4062	2023-02-04	2023-03-11	2023-03-11	15
105	200205	998877	3070	2023-01-05	2023-03-12	2023-03-12	0
106	200206	998877	3071	2023-02-05	2023-03-11	2023-03-13	20

4. SYSTEM IMPLEMENTATION

4.1. PROGRAMMING

BOOK.HTML //BASIC STRUCTURE WITH STYLING

```
<!DOCTYPE html>
<html>
<head>
    <title>Book Form</title>
    <style>
        label {
            display: block;
            margin-bottom: 10px;
            font-weight: bold;
        }
        input[type=text], input[type=number] {
            width: 100%;
            padding: 12px 20px;
            margin: 8px 0;
            box-sizing: border-box;
            border: 2px solid #ccc;
            border-radius: 4px;
        }
        select {
            width: 100%;
            padding: 12px 20px;
            margin: 8px 0;
            box-sizing: border-box;
            border: 2px solid #ccc;
            border-radius: 4px;
        }
        input[type=submit] {
            background-color: #4CAF50;
            color: white;
            padding: 12px 20px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }
            input[type=submit]:hover {
            background-color: #45a049;
        }
        .container {
            border-radius: 5px;
            background-color: #f2f2f2;
            padding: 20px;
            width: 50%;
            margin: auto;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Book Form</h2>
        <form action="insert_book.php" method="post">
            <label for="book_id">Book ID:</label>
            <input type="number" id="book_id" name="book_id"
required>

            <label for="author">Author:</label>
            <input type="text" id="author" name="author"
required>

            <label for="publication_year">Publication
Year:</label>
            <input type="number" id="publication_year"
name="publication_year" required>

            <label for="publisher">Publisher:</label>
            <input type="text" id="publisher"
```

```
name="publisher" required>

            <label for="book_name">Book Name:</label>
            <input type="text" id="book_name"
name="book_name" required>

            <label for="term">Term:</label>
            <input type="text" id="term" name="term"
required>

            <label for="edition">Edition:</label>
            <input type="text" id="edition" name="edition"
required>

            <label for="category">Category:</label>
            <select id="category" name="category" required>
                <option value="">--Select a category--
</option>
                <option value="Fiction">Fiction</option>
                <option value="Non-Fiction">Non-
Fiction</option>
                <option
value="Biography">Biography</option>
                <option value="Memoir">Memoir</option>
                <option value="Memoir">Academic</option>
            </select>

            <input type="submit" value="Submit">
        </form>
```

In this manner, the other table's HTML with styling file was created.

Insert book.php

```php
<?php
// Check if form is submitted
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Retrieve form data
    $book_id = $_POST['book_id'];
    $author = $_POST['author'];
    $publication_year = $_POST['publication_year'];
    $publisher = $_POST['publisher'];
    $book_name = $_POST['book_name'];
    $term = $_POST['term'];
    $edition = $_POST['edition'];
    $category = $_POST['category'];
    // Connect to MySQL database
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "kucse_library";
    $conn = mysqli_connect($servername, $username, $password,
$dbname);
    if (!$conn) {
        die('Connection failed: ' . mysqli_connect_error());
    }
    // Insert form data into database
    $sql = "INSERT INTO Book (book_id, author, publication_year,
publisher, book_name, term, edition, category)
     VALUES ('$book_id', '$author', '$publication_year',
'$publisher', '$book_name', '$term', '$edition', '$category')";
if (mysqli_query($conn, $sql)) {
        echo "New record created successfully";
    } else {
        echo "Error: " . $sql . "<br>" . mysqli_error($conn);
    }
    mysqli_close($conn);
}
?>
```

borrow_book.php

```php
<?php
// Check if form is submitted
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Get form data
    $borrow_id = $_POST['borrow_id'];
    $student_id = $_POST['student_id'];
    $book_id = $_POST['book_id'];
    $borrow_date = $_POST['borrow_date'];
    $due_date = $_POST['due_date'];
    // Connect to MySQL database
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "kucse_library";
    $conn = mysqli_connect($servername, $username, $password,
$dbname);
    // Check connection
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    // Insert data into table
    $sql = "INSERT INTO BorrowedBooks (borrow_id, student_id,
book_id, borrow_date, due_date)
    VALUES ('$borrow_id', '$student_id', '$book_id',
'$borrow_date', '$due_date')";
    if (mysqli_query($conn, $sql)) {
        echo "New record created successfully";
    } else {
        echo "Error: " . $sql . "<br>" . mysqli_error($conn);
    }
    // Close MySQL connection
    mysqli_close($conn);
}
?>
```

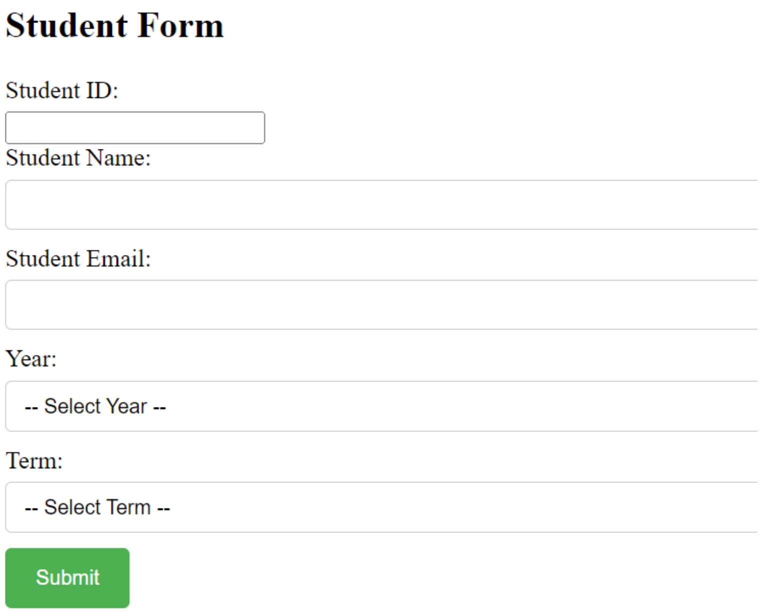
DATABASE_CONNECTION

```
// Connect to MySQL database
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "kucse_library";
    $conn = mysqli_connect($servername, $username, $password,
$dbname);
    // Check connection
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
```

In this manner, the other PHP file was created.

4.2. IMPLEMENTATION

STUDENT FORM:



Student Form

Student ID:

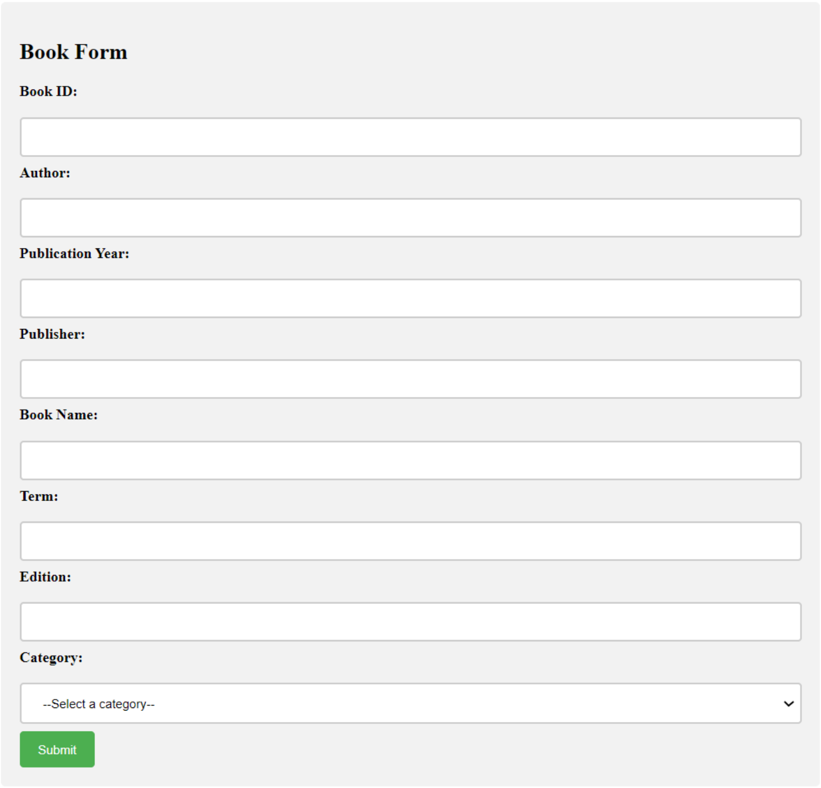
Student Name:

Student Email:

Year:

Term:

BOOK FORM:



Book Form

Book ID:

Author:

Publication Year:

Publisher:

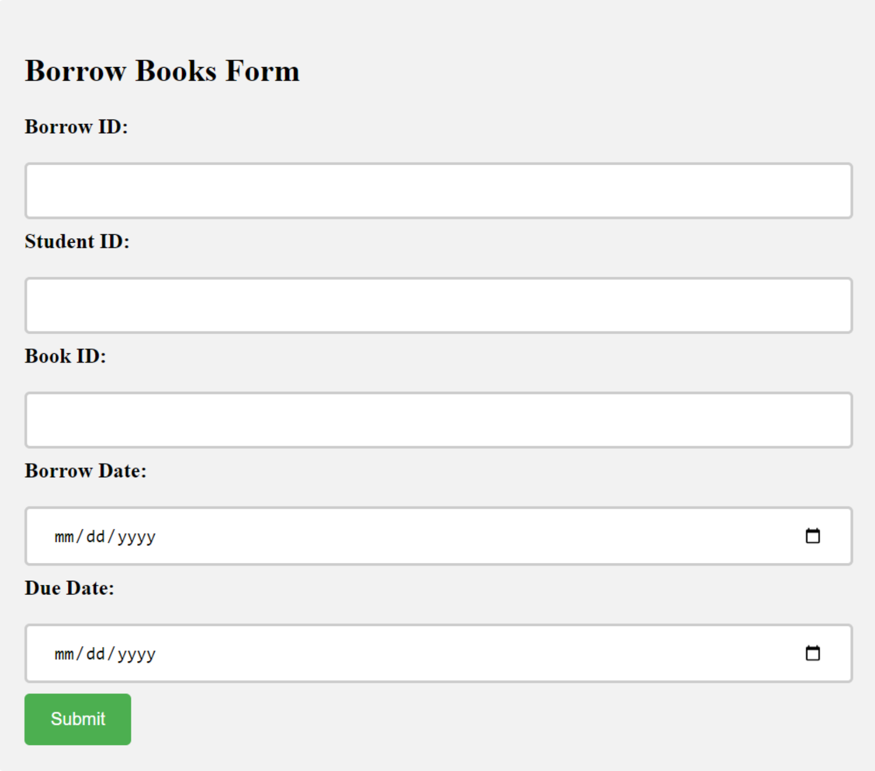
Book Name:

Term:

Edition:

Category:

BORROW BOOK FORM:



Borrow Books Form

Borrow ID:

Student ID:

Book ID:

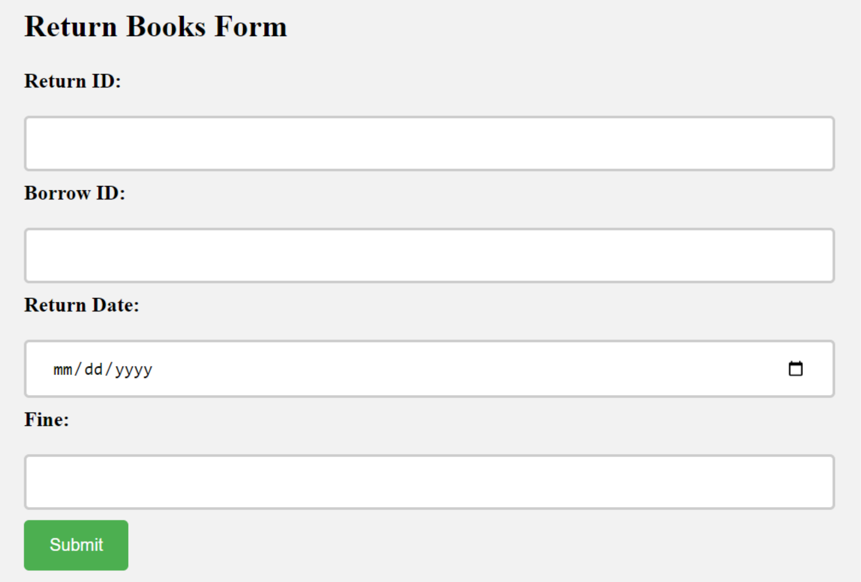
Borrow Date:

mm/dd/yyyy

Due Date:

mm/dd/yyyy

RETURN BOOK FORM:



Return Books Form

Return ID:

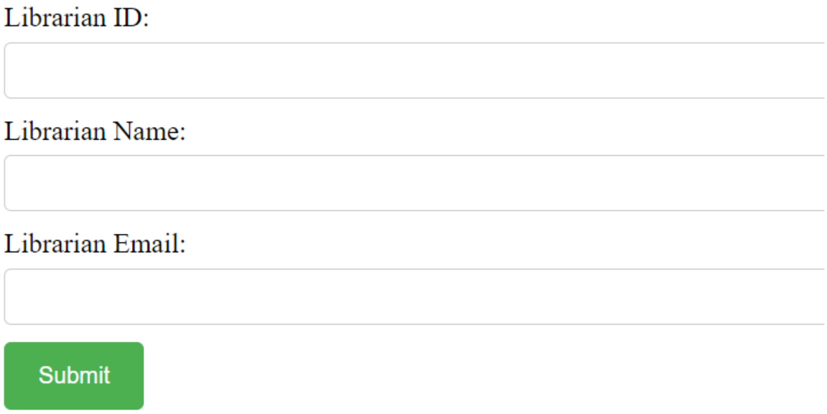
Borrow ID:

Return Date:

mm/dd/yyyy

Fine:

LIBRARIAN:

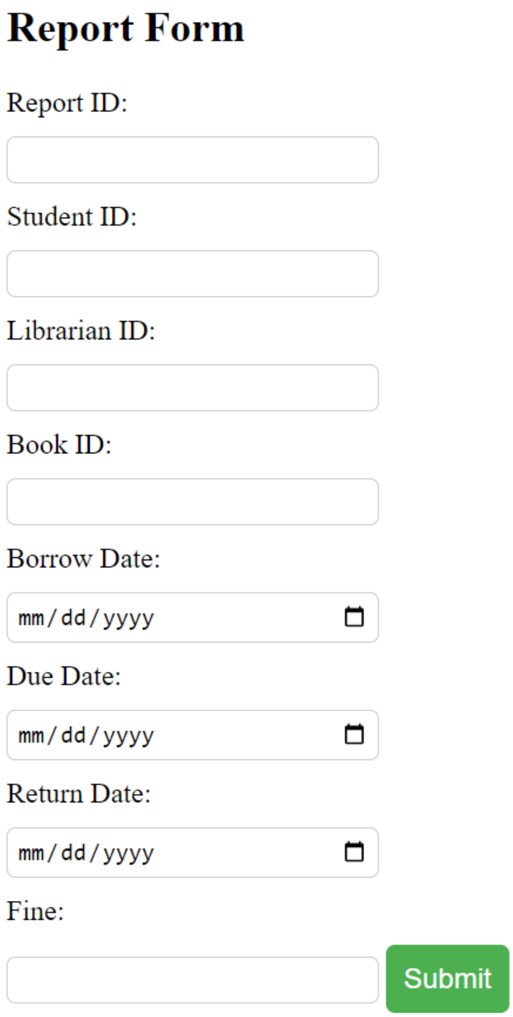


Librarian ID:

Librarian Name:

Librarian Email:

REPORT FORM:



Report Form

Report ID:

Student ID:

Librarian ID:

Book ID:

Borrow Date:

Due Date:

Return Date:

Fine: