A Project Report
on
Library Management System

Computer Science (083)

2023-24

<u>Submitted By:</u>

ARADHYA JHA

Class: XII th D

Roll No: 10

Under the guidance of:

Mrs. Nidhi Shinde

Rosary Senior Secondary School

Radio Colony, Kingsway Camp

Delhi - 110009

CONTENTS

CERTIFICATE

This is to certify that the project titled, "**Library Management System**" is a piece of work done by **Aradhya Jha** of class XIIth– D, in partial fulfillment of CBSE'^{S} AISSCE 2023-24 and has been carried out under my supervision and guidance. This report or an identical report on this topic has not been submitted for any other examination and does not form a part of any other course undergone by the candidate.

....

Signature of Teacher/Guide

Name: Mrs. Nidhi Shinde

Designation: PGT (Computer Science)

Place: Rosary Sr. Sec. School, Delhi

Date: --/--/2023-24

ACKNOWLEDGEMENT

I would like to extend my gratitude to the Principal Rev. Fr. Savariraj and Head MistressSr. Carmen for providing me with all the facility that was required.

I would also like to express my gratitude to my guide Mrs. Nidhi Shinde for her able guidance and support in completing my project.

Last but not the least I would like to thank my parents and friends who helped me a lot.

…………………………

Signature of Student

Name: Aradhya Jha

Class & Section: XIITH-D

Place: Rosary Sr. Sec. School, Delhi

Date: --/--/2023-24

INTRODUCTION

The Library Management System project is designed to streamline and automate the operations of a library. Utilizing the MySQL database for data storage and retrieval, the system provides a comprehensive set of functionalities for both book management and borrower transactions.

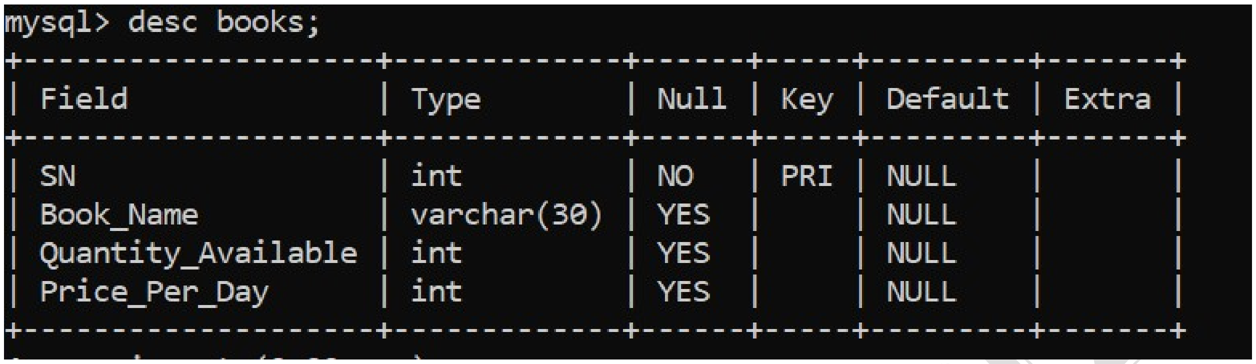
Upon connecting to the localhost, the system initializes the database "pathsala" and creates two essential tables: "books" and "BORROWER." The "books" table stores information about available books, including a unique serial number (SN), book name, quantity available, and price per day. The "BORROWER" table records details of borrowers, such as their name, contact number, and the book they have borrowed.

The system offers various operations, including viewing details of available books, checking specific book information, lending books to borrowers, adding new books to the inventory, updating data, and viewing borrower details. Additionally, it calculates fines for borrowers based on the number of days a book is kept.

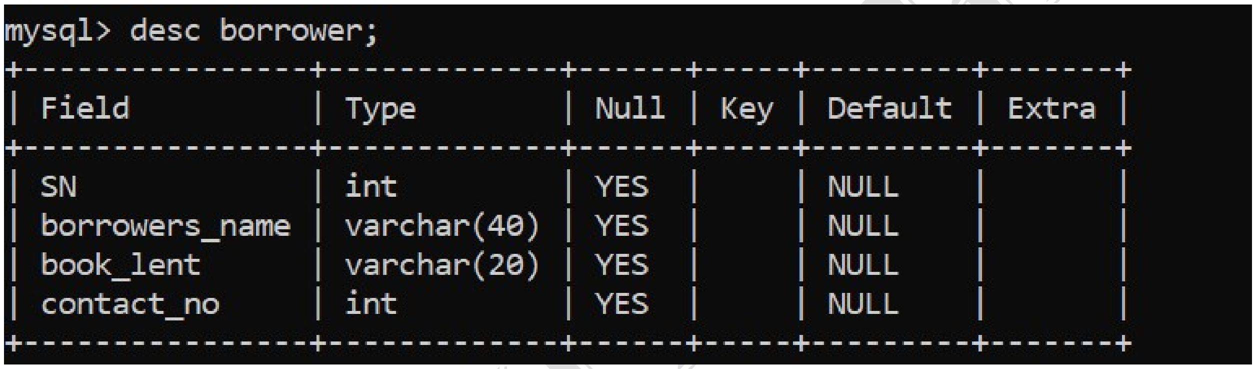
While the project provides fundamental library management functionalities, it has limitations such as the absence of robust user authentication, a lack of modularization, and limited error handling. These aspects could be enhanced for improved security, maintainability, and user experience.

FRONT-END: Python
BACK END: MySQL

TABLE STRUCTURE

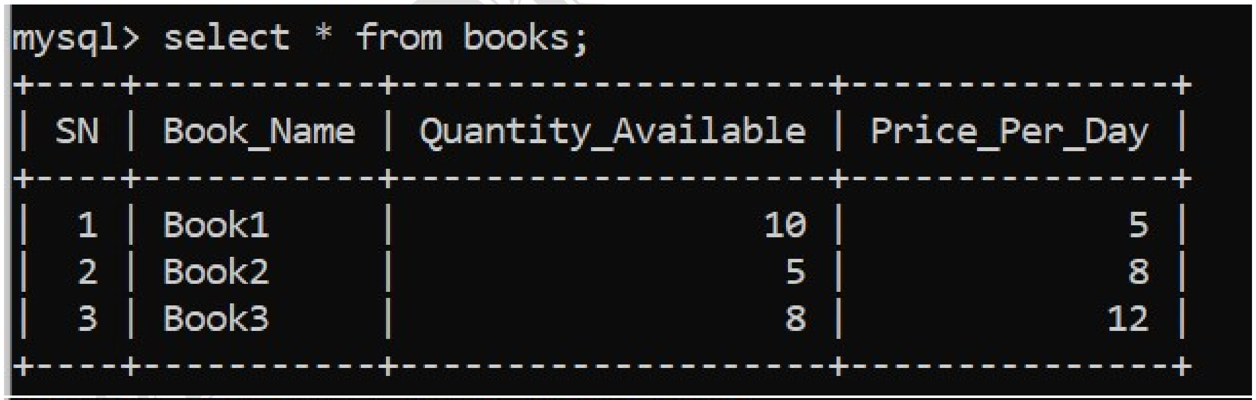


```
mysql> desc books;
+-------------------+-------------+------+-----+---------+-------+
| Field             | Type        | Null | Key | Default | Extra |
+-------------------+-------------+------+-----+---------+-------+
| SN                | int         | NO   | PRI | NULL    |       |
| Book_Name         | varchar(30) | YES  |     | NULL    |       |
| Quantity_Available | int        | YES  |     | NULL    |       |
| Price_Per_Day     | int         | YES  |     | NULL    |       |
+-------------------+-------------+------+-----+---------+-------+
```

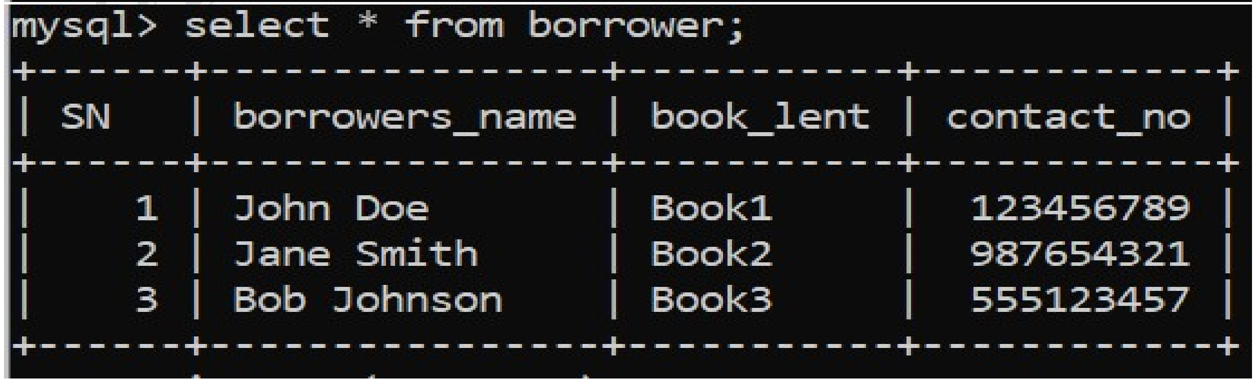


```
mysql> desc borrower;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| SN             | int         | YES  |     | NULL    |       |
| borrowers_name | varchar(40) | YES  |     | NULL    |       |
| book_lent      | varchar(20) | YES  |     | NULL    |       |
| contact_no     | int         | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
```

TABLE DATA



```
mysql> select * from books;
+----+-----------+--------------------+---------------+
| SN | Book_Name | Quantity_Available | Price_Per_Day |
+----+-----------+--------------------+---------------+
|  1 | Book1     |                 10 |             5 |
|  2 | Book2     |                  5 |             8 |
|  3 | Book3     |                  8 |            12 |
+----+-----------+--------------------+---------------+
```



```
mysql> select * from borrower;
+------+----------------+-----------+------------+
| SN   | borrowers_name | book_lent | contact_no |
+------+----------------+-----------+------------+
|    1 | John Doe       | Book1     |  123456789 |
|    2 | Jane Smith     | Book2     |  987654321 |
|    3 | Bob Johnson    | Book3     |  555123457 |
+------+----------------+-----------+------------+
```

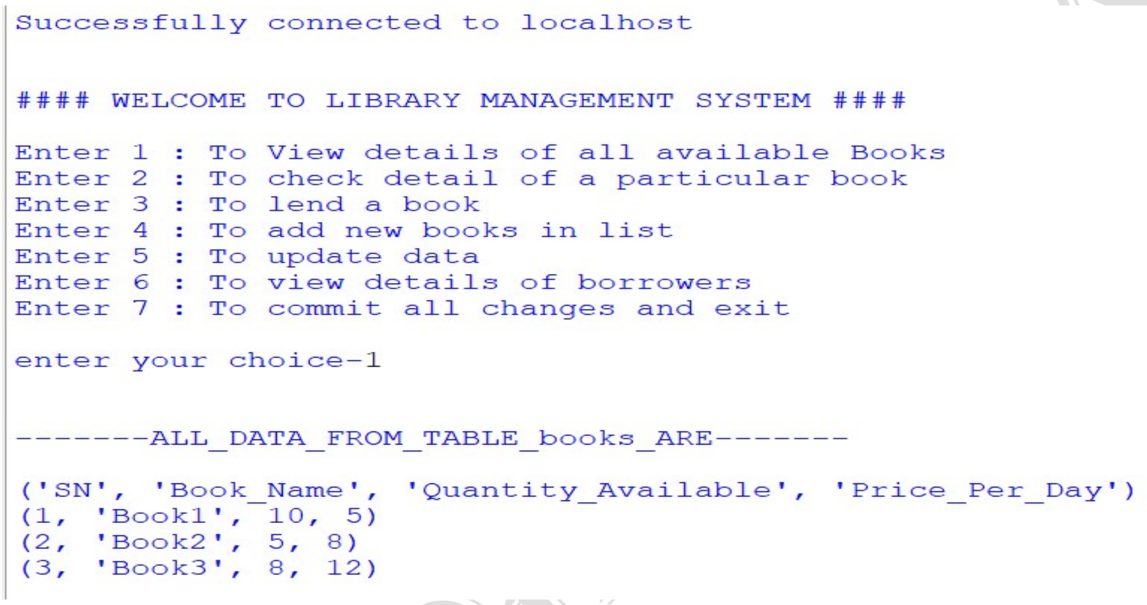
SOURCE CODE

```
import mysql.connector as sqlctr
import sys
from datetime import datetime

mycon = sqlctr.connect(host='localhost', user='root', password='aman jha')

if mycon.is_connected():
    print('\nSuccessfully connected to localhost')
else:
    print('Error while connecting to localhost')

cursor = mycon.cursor()

# Creating database
cursor.execute("create database if not exists pathsala")
cursor.execute("use pathsala")

# Creating the tables we need
cursor.execute("create table if not exists books(SN int(5) primary key, Book_Name varchar(30),
Quantity_Available int(10), Price_Per_Day int(10))")
cursor.execute("create table if not exists BORROWER(SN int(5), borrowers_name varchar(40),
book_lent varchar(20), contact_no int(10))")

def command(st):
    cursor.execute(st)

def fetch():
    data = cursor.fetchall()
    for i in data:
        print(i)

def all_data(tname):
    li = []
    st = 'desc ' + tname
    command(st)
    data = cursor.fetchall()
    for i in data:
        li.append(i[0])
    st = 'select * from ' + tname
    command(st)
    print('\n')
    print('-------ALL_DATA_FROM_TABLE_' + tname + '_ARE-------\n')
    print(tuple(li))
    fetch()

def detail_burrower(name, contact):
    tup = ('SN', 'borrowers_name', 'book_lent', 'date', 'contact_no')
    print('\n---Details for borrower ' + name + '---\n')
    print(tup)
    st = 'select * from borrower where borrowers_name like "{}" and
contact_no={}'.format(name, contact)
    command(st)
    fetch()

def days_between(d1, d2):
    d1 = datetime.strptime(d1, "%Y-%m-%d")
    d2 = datetime.strptime(d2, "%Y-%m-%d")
    global days
    days = abs((d2 - d1).days)
```

```
def price_book(days, book_name):
    st1 = 'select Price_Per_Day from books where Book_Name="{}"'.format(book_name)
    command(st1)
    data = cursor.fetchall()
    for i in data:
        global t_price
        t_price = int(i[0]) * days
    print('No. of days {} book is kept : {}'.format(book_name, days))
    print('Price per day for book {} is Rs.{}'.format(book_name, i[0]))
    print('Total fare for book ' + book_name + '-', t_price)

def lend():
    flag = 'True'
    while flag == 'True':
        print('\n__AVAILABLE BOOKS__\n')
        st0 = 'select Book_Name from books where Quantity_Available>=1'
        command(st0)
        fetch()
        st1 = 'select max(SN) from borrower'
        command(st1)
        data_sn = cursor.fetchall()
        for i in data_sn:
            SN = i[0] + 1
        book_selected = str(input('Enter name of book from above list : '))
        borrowers_name = str(input('Enter Borrower Name : '))
        date = str(input('Enter date (YYYY-MM-DD) : '))
        contact = int(input('Enter contact no. : '))
        st_insert = 'insert into borrower values({}, "{}", "{}", "{}", {})'.format(SN,
borrowers_name, book_selected, date, contact)
        command(st_insert)
        st_quantity = 'select Quantity_Available from books where
Book_Name="{}"'.format(book_selected)
        command(st_quantity)
        data_quantity = cursor.fetchall()
        for quantity in data_quantity:
            qty = quantity[0] - 1
        st_dec = 'update books set Quantity_Available={} where Book_Name="{}"'.format(qty,
book_selected)
        command(st_dec)
        dec = str(input('Do you want to add more records (Y/N) : '))
        if dec.upper() == "Y":
            flag = 'True'
        else:
            flag = 'False'

def borrowers():
    print('\n\n__OPTIONS AVAILABLE__\n\nEnter 1 : To Show detail of all borrowers \nEnter
2 : To check detail of a particular borrower \nEnter 3 : To calculate total fine of a borrower
\nEnter 4 : To go Back \nEnter 5 : To commit all the changes and exit')
    dec = input('enter your choice-')
    if dec == '1':
        all_data('borrower')
    elif dec == '2':
        name = str(input('\nenter borrower name-'))
        contact = str(input('enter borrower contact no.-'))
        detail_burrower(name, contact)
    elif dec == '3':
        tfine()
    elif dec == '4':
        action_list()
    elif dec == '5':
        close()
    borrowers()

def tfine():
    name = str(input('\nEnter borrower name : '))
    contact = input('Enter borrower contact_no : ')
    detail_burrower(name, contact)
    st1 = 'select book_lent from borrower where borrowers_name ="{}" and
contact_no={}'.format(name, contact)
    command(st1)
    data = cursor.fetchall()
    for i in data:
        book_name = i[0]
    li_val = []
    command('desc books')
    data = cursor.fetchall()
    for i in data:
        li_val.append(i[0] + 1)
    for k in range(1, 4):
        val = str(input('Enter ' + i[k] + '-'))
        li_val.append(val)
    li1.append(tuple(li_val))
    values = ', '.join(map(str, li1))
    st1 = "INSERT INTO books VALUES {}".format(values)
    command(st1)
    all_data('books')
    print('\n')
    print("\nDATA INSERTED SUCCESSFULLY\n")
    dec = str(input('Do u want to insert more data?(Y/N)-'))
    if dec.upper() == "Y":
        flag = 'true'
    else:
        flag = 'false'
        action_list()

def update(tname, col1, post_value, pre_value):
    st = str('update %s set %s=%s where SN=%s') % (tname, col1, "'%s'", "'%s'") % (post_value,
pre_value)
    command(st)
    all_data(tname)
    print('\nVALUE UPDATED SUCCESSFULLY')

def close():
    mycon.commit()
    mycon.close()
    if mycon.is_connected():
        print('still connected to localhost')
    else:
        print('\n\nconnection closed successfully.')
        sys.exit()

def action_list():
    print('\n')
    print('#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####\n\nEnter 1 : To
View details of all available Books\nEnter 2 : To check detail of a particular book\nEnter 3 : To
lend a book \nEnter 4 : To add new books in list \nEnter 5 : To update data \nEnter 6 : To view
details of borrowers \nEnter 7 : To commit all changes and exit')
    dec = input('\nenter your choice-')
    if dec == '1':
        all_data('books')
    elif dec == '2':
        tup = ('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
        tup1 = ('SN', 'borrowers_name', 'book_lent', 'contact_no')
        in1 = str(input('enter first name , last name or middle name of a book-'))
        print('\n___ALL DATA OF BOOKS HAVING "{}" IN THEIR NAME FROM BOTH
TABLE__'.format(in1))
        st = str('select * from books where book_name like "{}"'.format('%' + in1 + '%'))
        st1 = str('select * from borrower where book_lent like "{}"'.format('%' + in1 + '%'))
        print('\n_DATA FROM TABLE BOOKS_\n')
        command(st)
        print(tup)
        fetch()
        print('\n_DATA FROM TABLE BORROWER_\n')
        command(st1)
        print(tup1)
        fetch()
        print()
    elif dec == '3':
        lend()
    elif dec == '4':
        insert()
    elif dec == '5':
        flag = 'true'
        while flag == 'true':
            tname = 'books'
            li = []
            st1 = 'desc ' + tname
            command(st1)
            data = cursor.fetchall()
            for i in data:
                li.append(i[0])
            all_data(tname)
            print('\n columns in table ' + tname + ' are')
            print(li)
            col1 = str(input('enter column name for modification from above list-'))
            lipo = ['SN']
            lipo.append(col1)
            print(tuple(lipo))
            st0 = 'select SN , %s from books' % (col1)
            command(st0)
            fetch()
            pre_value = str(input('enter corresponding SN for the data to be changed-'))
            post_value = str(input('enter new value for column %s having SN %s-' % (col1,
pre_value)))
            update(tname, col1, post_value, pre_value)
            dec = str(input('Do you want to change more data?(Y/N)-'))
            if dec == 'y' or dec == 'Y':
                flag = 'true'
            else:
                flag = 'false'

    elif dec == '6':
        borrowers()
    elif dec == '7':
        close()

    action_list()

action_list()
```

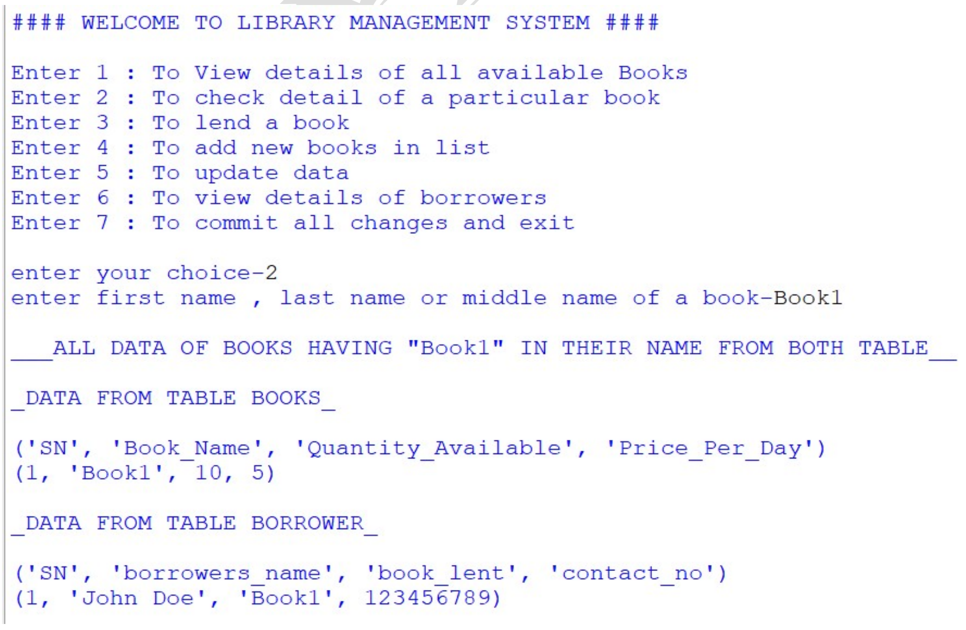
SAMPLE OUTPUT

OPTION – 1:



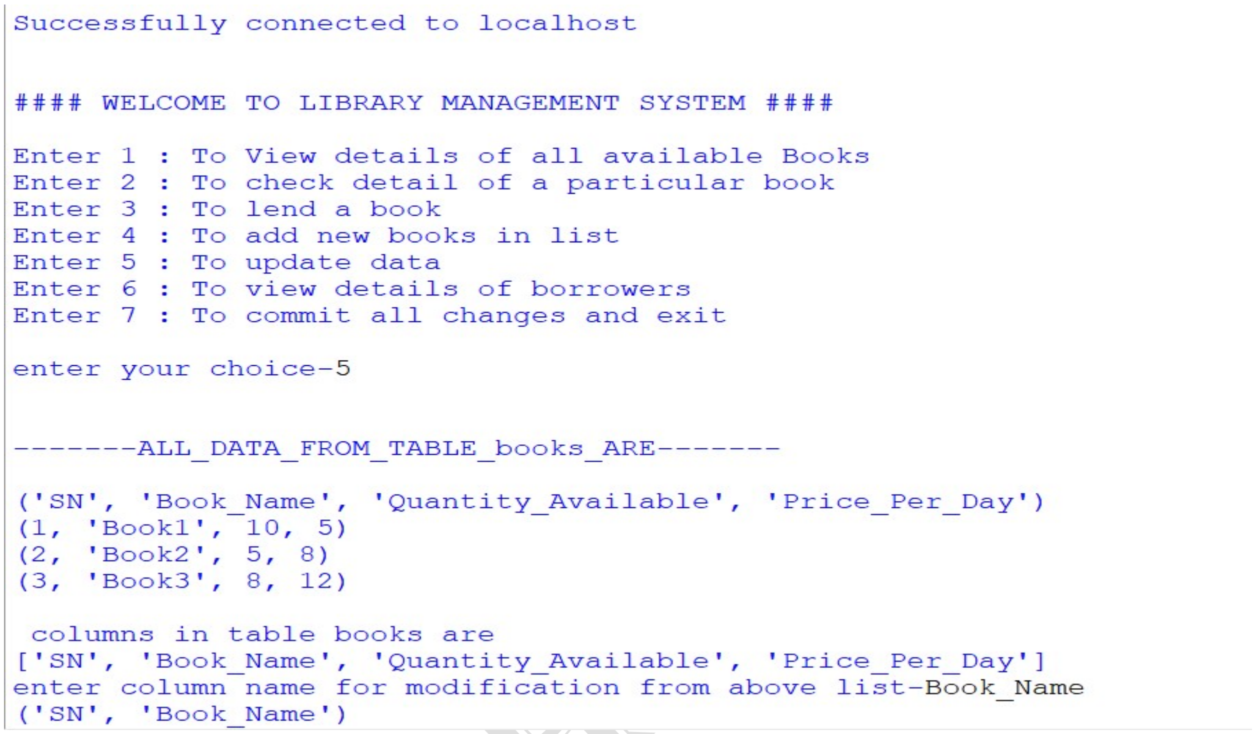
```
Successfully connected to localhost


#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####

Enter 1 : To View details of all available Books
Enter 2 : To check detail of a particular book
Enter 3 : To lend a book
Enter 4 : To add new books in list
Enter 5 : To update data
Enter 6 : To view details of borrowers
Enter 7 : To commit all changes and exit

enter your choice-1


-------ALL_DATA_FROM_TABLE_books_ARE-------

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'Book1', 10, 5)
(2, 'Book2', 5, 8)
(3, 'Book3', 8, 12)
```

OPTION – 2:

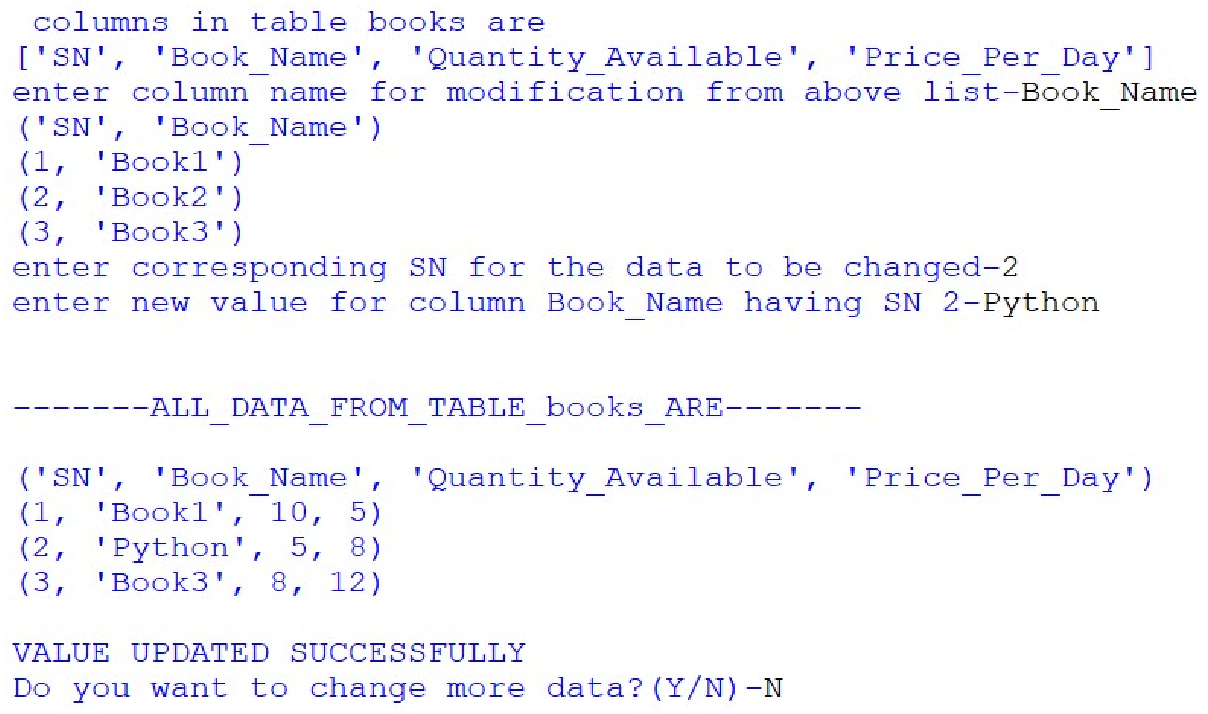


```
#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####

Enter 1 : To View details of all available Books
Enter 2 : To check detail of a particular book
Enter 3 : To lend a book
Enter 4 : To add new books in list
Enter 5 : To update data
Enter 6 : To view details of borrowers
Enter 7 : To commit all changes and exit

enter your choice-2
enter first name , last name or middle name of a book-Book1

___ALL DATA OF BOOKS HAVING "Book1" IN THEIR NAME FROM BOTH TABLE__

_DATA FROM TABLE BOOKS_

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'Book1', 10, 5)

_DATA FROM TABLE BORROWER_

('SN', 'borrowers_name', 'book_lent', 'contact_no')
(1, 'John Doe', 'Book1', 123456789)
```

OUTPUT -3:

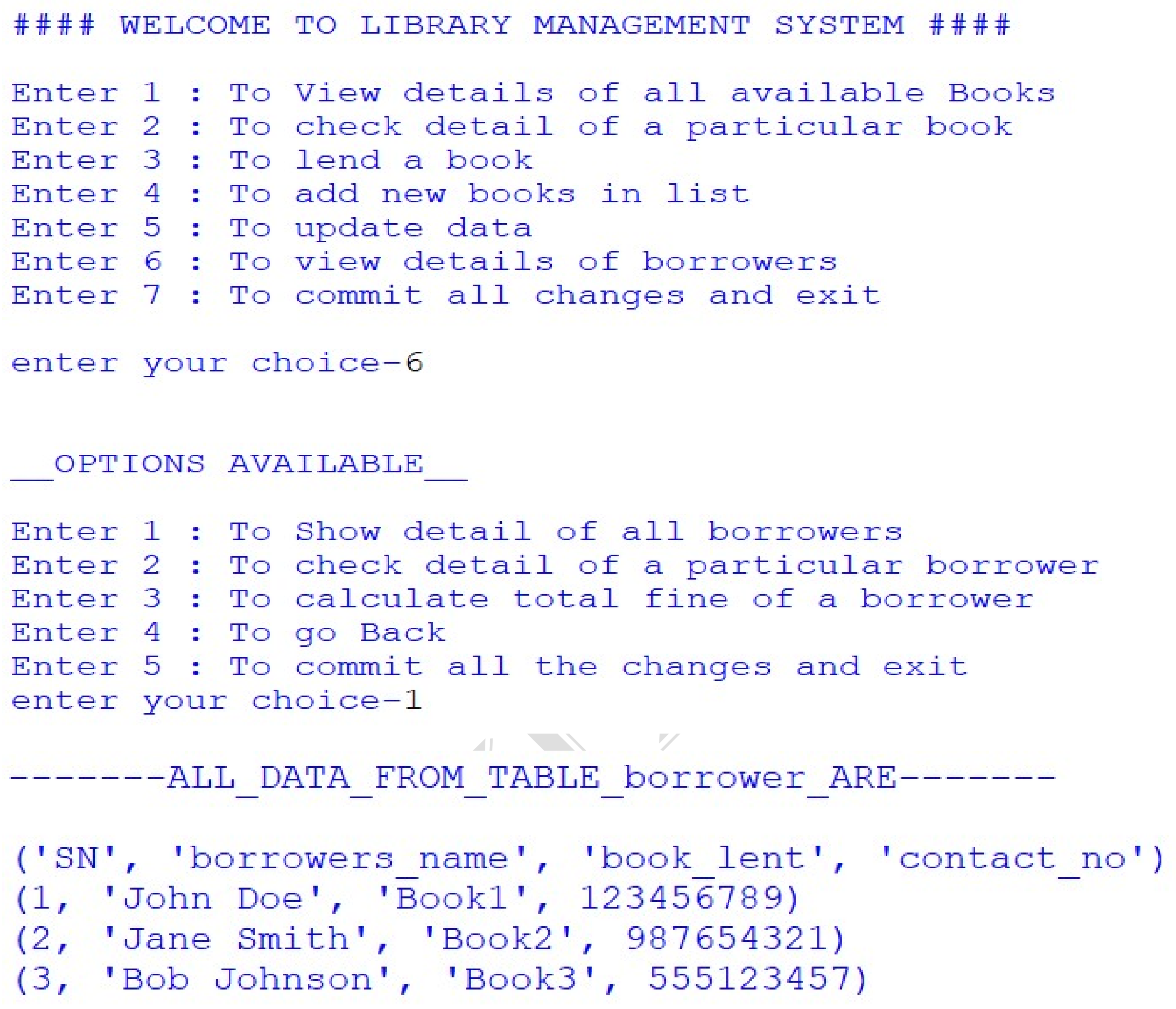


```
Successfully connected to localhost


#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####

Enter 1 : To View details of all available Books
Enter 2 : To check detail of a particular book
Enter 3 : To lend a book
Enter 4 : To add new books in list
Enter 5 : To update data
Enter 6 : To view details of borrowers
Enter 7 : To commit all changes and exit

enter your choice-5


-------ALL_DATA_FROM_TABLE_books_ARE-------

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'Book1', 10, 5)
(2, 'Book2', 5, 8)
(3, 'Book3', 8, 12)

 columns in table books are
['SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day']
enter column name for modification from above list-Book_Name
('SN', 'Book_Name')
```

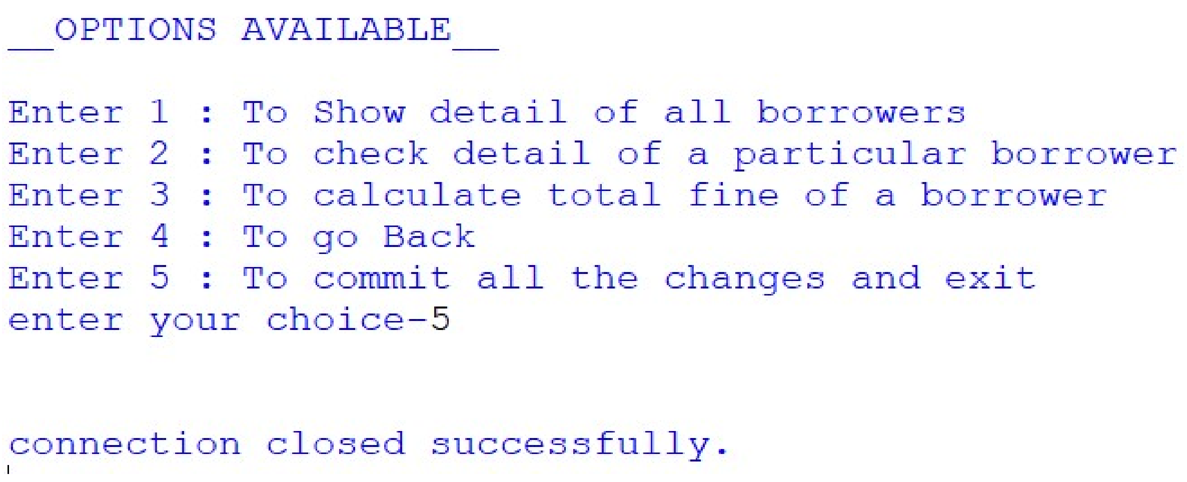


```
 columns in table books are
['SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day']
enter column name for modification from above list-Book_Name
('SN', 'Book_Name')
(1, 'Book1')
(2, 'Book2')
(3, 'Book3')
enter corresponding SN for the data to be changed-2
enter new value for column Book_Name having SN 2-Python


-------ALL_DATA_FROM_TABLE_books_ARE-------

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'Book1', 10, 5)
(2, 'Python', 5, 8)
(3, 'Book3', 8, 12)

VALUE UPDATED SUCCESSFULLY
Do you want to change more data?(Y/N)-N
```

OUTPUT -4:

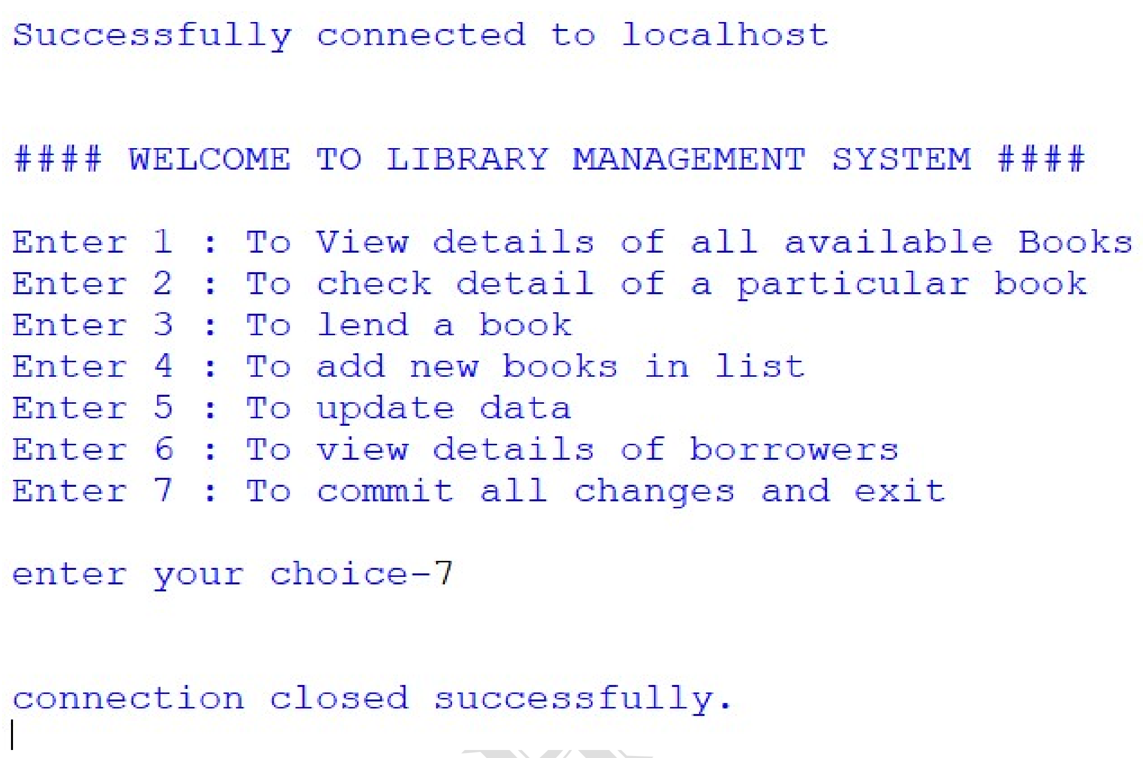


```
#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####

Enter 1 : To View details of all available Books
Enter 2 : To check detail of a particular book
Enter 3 : To lend a book
Enter 4 : To add new books in list
Enter 5 : To update data
Enter 6 : To view details of borrowers
Enter 7 : To commit all changes and exit

enter your choice-6


__OPTIONS AVAILABLE__

Enter 1 : To Show detail of all borrowers
Enter 2 : To check detail of a particular borrower
Enter 3 : To calculate total fine of a borrower
Enter 4 : To go Back
Enter 5 : To commit all the changes and exit
enter your choice-1

-------ALL_DATA_FROM_TABLE_borrower_ARE-------

('SN', 'borrowers_name', 'book_lent', 'contact_no')
(1, 'John Doe', 'Book1', 123456789)
(2, 'Jane Smith', 'Book2', 987654321)
(3, 'Bob Johnson', 'Book3', 555123457)
```



```
__OPTIONS AVAILABLE__

Enter 1 : To Show detail of all borrowers
Enter 2 : To check detail of a particular borrower
Enter 3 : To calculate total fine of a borrower
Enter 4 : To go Back
Enter 5 : To commit all the changes and exit
enter your choice-5


connection closed successfully.
```

OUTPUT – 5:



```
Successfully connected to localhost


#### WELCOME TO LIBRARY MANAGEMENT SYSTEM ####

Enter 1 : To View details of all available Books
Enter 2 : To check detail of a particular book
Enter 3 : To lend a book
Enter 4 : To add new books in list
Enter 5 : To update data
Enter 6 : To view details of borrowers
Enter 7 : To commit all changes and exit

enter your choice-7


connection closed successfully.
```

LIMITATIONS OF THE PROJECT

The given library management system project has a few limitations. Firstly, it lacks proper user authentication and authorization mechanisms, making it vulnerable to unauthorized access and potential misuse. Additionally, the code structure is not modularized, making it challenging to maintain and extend. Error handling is minimal, and there's a lack of input validation, which may result in unexpected behavior or crashes when users input incorrect data. Furthermore, the project lacks data validation and constraints, allowing the possibility of inconsistent or inaccurate data in the database. The user interface is text-based and may not provide a user-friendly experience. Lastly, the code doesn't adhere to best practices for database interactions, such as using parameterized queries, which could expose the system to SQL injection attacks.

CONCLUSION

The Library Management System project presents an efficient solution for organizing and managing library resources. Leveraging MySQL for data storage and Python for system implementation, the project encompasses essential functionalities such as book inventory management, borrower transactions, and fine calculations. The system allows users to view details of available books, check specific book information, lend books to borrowers, and update data seamlessly.

Despite its functionality, the project has certain limitations, including the absence of robust user authentication and a modular structure. These aspects could be improved for enhanced security and system flexibility. Additionally, the system lacks comprehensive error handling mechanisms. Future iterations could benefit from incorporating user authentication features, modularizing the code for better maintainability, and implementing robust error handling to ensure a more reliable and user-friendly experience.

In essence, the Library Management System project serves as a foundational tool for library administration, with the potential for further refinement and expansion to meet evolving needs and standards in library management.

BIBLIOGRAPHY

1. Computer Science (NCERT)
2. Computer Science with Python by Preeti Arora
3. https://www.w3schools.com/python/
4. https://www.geeksforgeeks.org/python-gq/