# A Project Report
# On
# Library Management Program.


Submitted By

Name: Shrinibas Mahanta

Class: XII Sc.

Roll No.: _____

Session: 2022-2023

Under The Guidance Of

Mr K. Jagannath Reddy

(Computer Science Teacher)

Department of Computer Science

Rabindra Vidya Niketan, Keonjhar

Odisha

# INDEX


| SN | Details | Page No. |
|---|---|---|
| 1. | Acknowledgment | 1 |
| 2. | Certificate from supervisor | 2 |
| 3. | System Requirements | 3 |
| 4. | Technology Stack Used | 4 |
| 5. | List of Database and Tables used | 5 |
| 6. | Source Code | 6 to 11 |
| 7. | Output | 12to17 |
| 8. | Objective of the Project | 18 |
| 9. | Bibliography / References | 19 |

# Acknowledgment

I would like to Give my special thanks of gratitude to my Computer Science Teacher ,“MR K. Jagannath Reddy for his able Guidance and support during the completion of this Project

I would like to extend my gratitude to my Group members Without whose Contributions and Co-operation this project would not have been possible .

I would also like to thank my Parents and my friends for their Support and valuable suggestions regarding this Project.

At last I would thank each and everyone who has been helpful to me during the completion of this project

Student Sign.

# Certificate

This is To certify That The Project Work

Titled:

Library Management Program.

Is developed by

Shrinibas Mahanta

Prachurya Chandra jena

and

Sagnik Behera

in a group of 3 students.

This project work is carried out to fulfill the partial requirement of CBSE AISSCE Practical Exam 2023 for the subject Computer Science (083). This is a Original Work of Ours

Student Sign.

Teacher I/C Sign.

# System Requirements

## Hardware used:

- Operating System : Fedora Linux 37 x86_64
- Processor : Intel® Core™ i3-1115G4 Processor
- Ram : 8GB
- Graphics Processor: Intel® UHD Graphics G4
- Storage: 256 GB S.S.D

## Software used:

- Linux OS
- Code Editor: VS Code
- Python : 3.11.0

# Technology Stack Used

## Front-End: -

Front-End refers to the interface which the user uses to communicate with the System underlying Programs and Databases. I have Used **Python 3.11.0** to develop the Front-End of the Project, Which is a Command Line Interface(CLI). I've used **VS Code** as my Code editor for writing the Program.

## Back-End: -

Back-End refers to the database underlying in the system which is working with the front-end To store data. I have used MariaDB database for my back-end in this project.

Both The software used for developing this project are specified by C.B.S.E and are free and open source.

# List of Database and Tables Used

Database: pathsala

Table(s): books and borrower

Structure of Table:books

```
MariaDB [pathsala]> desc books;
+--------------------+-------------+------+-----+---------+-------+
| Field              | Type        | Null | Key | Default | Extra |
+--------------------+-------------+------+-----+---------+-------+
| SN                 | int(5)      | NO   | PRI | NULL    |       |
| Book_Name          | varchar(30) | YES  |     | NULL    |       |
| Quantity_Available | int(10)     | YES  |     | NULL    |       |
| Price_Per_Day      | int(10)     | YES  |     | NULL    |       |
+--------------------+-------------+------+-----+---------+-------+
```

Structure of Table:borrower

```
MariaDB [pathsala]> desc borrower;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| SN             | int(5)      | YES  |     | NULL    |       |
| borrowers_name | varchar(40) | YES  |     | NULL    |       |
| book_lent      | varchar(20) | YES  |     | NULL    |       |
| date           | date        | YES  |     | NULL    |       |
| contact_no     | varchar(15) | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
```

# Source Code

```
import mariadb as sqlctr
import sys
from datetime import datetime
mycon = sqlctr.connect(host='localhost', user='root', password='heaven')
if mycon.connection_id != None:
    print('\n')
    print('Successfully 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),date DATE,contact_no varchar(15))")
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:
            if i[0] == None:
                SN = 1
            else:
                SN = i[0]+1
        print(SN)
        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=str(input('Enter contact no. : '))
        st_insert='insert into borrower values({},"{}","{}","{}","{}")'.format(SN,borrowers_name,book_selected,date,contact)
        print(st_insert)
        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 =="y":
            flag= 'True'
        else:
            flag='False'
        mycon.commit()

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
    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=str(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]
        st2 = 'select date from borrower where borrowers_name="{}" and book_lent="{}"'.format(name,book_name)
        command(st2)
        data1=cursor.fetchall()
        for date in data1:
            date_taken=date[0]
            date_return = str(input('\nEnter returning date for book "{}" (YYYY-MM-DD) , Press ENTER to skip-'.format(book_name)))
```

```
            while date_return!='':
                days_between(str(date_return),str(date_taken))
                price_book(days,i[0])
                print('\nEnter Y : If Rs.{} is paid and book is returned.\nEnter N : If fare is not paid and book is not returned.'.format(t_pric
                dec=str(input('Enter (Y?N) : '))
                if dec.upper()=="Y":
                    st= 'select SN , Quantity_Available from books where Book_Name ="{}"'.format(i[0])
                    command(st)
                    data2=cursor.fetchall()
                    for price in data2:
                        update('books', 'Quantity_Available',price[1]+1,price[0])
                    st_del = 'delete from borrower where borrowers_name="{}" and book_lent="{}"'.format(name,book_name)
                    command(st_del)
                    break
                else:
                    print("\n\nPLEASE PAY THE FARE AND RETURN BOOK AFTER READING.\n\n")
                    break

def insert():
    flag = 'true'
    while flag=='true':
        licol=[]
        li1=[]
        li_val=[]
        command('desc books')
        data=cursor.fetchall()
        for i in data:
            licol.append(i[0])
        command('select max(SN) from books')
        dta=cursor.fetchall()
        for j in dta:
            if j[0] == None:
                li_val.append(1)
            else:
                li_val.append(j[0]+1)
        for k in range(1,4):
            val = str(input('Enter '+licol[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'
```

```
    mycon.commit()
    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.connection_id == None:
        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 par
    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'
            mycon.commit()


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

# Output

## Welcome and Menu:-

```
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-
```

## Choice: 1 (To view details of all Books.):-

```
Enter your choice-1


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

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'CODE', 22, 25)
(2, 'Lord Of The Rings', 30, 26)
(3, 'Grookings Algorithms', 29, 30)
(4, 'COSMOS', 35, 40)
(5, 'Clean Code', 36, 28)
```

## Choice: 2 (To check details of a particular Book.):-

```
Enter your choice-2
enter first name , last name or middle name of a book-CODE

___ALL DATA OF BOOKS HAVING "CODE" IN THEIR NAME FROM BOTH TABLE____

__DATA FROM TABLE BOOKS__

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'CODE', 21, 25)
(5, 'Clean Code', 36, 28)

__DATA FROM TABLE BORROWER__

('SN', 'borrowers_name', 'book_lent', 'contact_no')
(1, 'sm2k4', 'CODE', datetime.date(2022, 12, 22), '8545886955')
```

## Choice: 3 (To Lend a Book.):-

```
Enter your choice-3

___AVAILABLE BOOKS___

('CODE',)
('Lord Of The Rings',)
('Grookings Algorithms',)
('COSMOS',)
('Clean Code',)
1
Enter name of book from above list : CODE
Enter Borrower Name : sm2k4
Enter date (YYYY-MM-DD) : 2022-12-22
Enter contact no. : 8545886955
insert into borrower values(1,"sm2k4","CODE","2022-12-22","8545886955")
Do you want to add more records (y/N) :
```

## Choice: 4 (To Add new Books in the list.):-

```
Enter your choice-4
Enter Book_Name-newbook
Enter Quantity_Available-30
Enter Price_Per_Day-25


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

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'CODE', 21, 25)
(2, 'Lord Of The Rings', 30, 26)
(3, 'Grookings Algorithms', 29, 30)
(4, 'COSMOS', 35, 40)
(5, 'Clean Code', 36, 28)
(6, 'newbook', 30, 25)



DATA INSERTED SUCCESSFULLY

Do u want to insert more data?(Y/N)-
```

## Choice: 5 (To Update data.):-

```
Enter your choice-5


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

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'CODE', 21, 25)
(2, 'Lord Of The Rings', 30, 26)
(3, 'Grookings Algorithms', 29, 30)
(4, 'COSMOS', 35, 40)
(5, 'Clean Code', 36, 28)
(6, 'newbook', 30, 25)

 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, 'CODE')
(2, 'Lord Of The Rings')
(3, 'Grookings Algorithms')
(4, 'COSMOS')
(5, 'Clean Code')
(6, 'newbook')
enter corresponding SN for the data to be changed-6
enter new value for column Book_Name having SN 6-changedname


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

('SN', 'Book_Name', 'Quantity_Available', 'Price_Per_Day')
(1, 'CODE', 21, 25)
(2, 'Lord Of The Rings', 30, 26)
(3, 'Grookings Algorithms', 29, 30)
(4, 'COSMOS', 35, 40)
(5, 'Clean Code', 36, 28)
(6, 'changedname', 30, 25)

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

## Choice: 6 (To view details of Borrowers.):-

```
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-
```

## Choice: 6(1) (To show details of  all Borrowers.):-

```
enter your choice-1


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

('SN', 'borrowers_name', 'book_lent', 'date', 'contact_no')
(1, 'sm2k4', 'CODE', datetime.date(2022, 12, 22), '8545886955')
```

## Choice: 6(2) (To check details of particular Borrower.):-

```
enter your choice-2

enter borrower name-sm2k4
enter borrower contact no.-8545886955

---Details for borrower sm2k4---

('SN', 'borrowers_name', 'book_lent', 'date', 'contact_no')
(1, 'sm2k4', 'CODE', datetime.date(2022, 12, 22), '8545886955')
```

## Choice: 6(3) (To calculate total fine of a Borrower.):-

```
enter your choice-3

Enter borrower name : sm2k4
Enter borrower contact_no : 8545886955

---Details for borrower sm2k4---

('SN', 'borrowers_name', 'book_lent', 'date', 'contact_no')
(1, 'sm2k4', 'CODE', datetime.date(2022, 12, 22), '8545886955')

Enter returning date for book "CODE" (YYYY-MM-DD) , Press ENTER to skip-2022-12-31
No. of days CODE book is kept : 9
Price per day for book CODE is Rs.25
Total fare for book CODE- 225

Enter Y : If Rs.225 is paid and book is returned.
Enter N : If fare is not paid and book is not returned.
Enter (Y?N) : n


PLEASE PAY THE FARE AND RETURN BOOK AFTER READING.
```

## Choice: 7 (To commit all changes and exit.):-

```
Enter your choice-7


connection closed successfully.
```

# Objective of The Project Work

This program is useful for any institution with a library And can make the process of library management easy.

Our program can streamline the whole process of library management And can make the process time efficient. It can help the library to become autonomous as this program does not need external human support to work properly and can be directly used by the customers.

The user Interface is easy enough for regular people to understand and at every part of the program there is pre written guide texts for helping the user to interact with the program.

Debugging the program is very easy as it uses Python and SQL language which are easy to understand,high level and simple programming languages.

# Bibliography / References

The following resources and reference were helpful during the making of this project:-


- Offline classes by the teacher.
- Websites Refereed/Knowledge-base:-
  - geeksforgeeks.org
  - freecodecamp.org
  - sqlcommands.com
- Books Referred:-
  - get programming with python.