

PYTHON MINI PROJECT LIBRARY MANAGEMENT SYSTEM

GROUP MEMBERS-
RISHIKESH KADAM (30)
SMRUTI KSHIRSAGAR (34)

➤ PROBLEM STATEMENT

Create a library management system which has the following functions-

- Issue a book
- Return a book
- Pay fine
- Re-issue a book
- Search a book

Put reasonable constraints over number of books issued, amount of fine pending, fine to be paid for each day exceeding permissible limit, etc.

➤ FUNCTIONALITIES

- ☐ The library system allows maximum 3 books to be issued to one user at a time.
- ☐ The user has to return the book within 7 days. From 8th day onwards, the user will be fined Rs.10 per day.
- ☐ If a user has pending fine of Rs.100 or more, he will not get more books.
- ☐ Every book has limited number of copies, so it should be checked if the copies are available before issuing the book.
- ☐ The fine should be updated once a day. Even if the programs runs a number of times or doesn't run at all, the fine should be correctly calculated for all the users.
- ☐ A user may re-issue a book on 8th day. The day count will be set to 0 on re-issuing.
- ☐ A book can be searched by entering the title. Book ID, number of available copies and author information will be returned.

➤ DATABASE

The database contains 5 tables.

Their schema is-

sqlite> .schema **BOOKS**

CREATE TABLE BOOKS (bookID integer primary key, title varchar(20), author varchar(20), copies integer);

```
sqlite> .schema USERINFO
```

```
CREATE TABLE USERINFO (userID integer primary key, userName varchar(20),  
booksBorrowed integer, fine integer);
```

```
sqlite> .schema USERBOOKS1
```

```
CREATE TABLE USERBOOKS1 (userID integer primary key, book1 integer, book2 integer,  
book3 integer);
```

```
sqlite> .schema USERBOOKS2
```

```
CREATE TABLE USERBOOKS2 (userID integer primary key, days1 integer, days2 integer,  
days3 integer);
```

```
sqlite> .schema LASTUPDATE
```

```
CREATE TABLE LASTUPDATE (pointer integer, year integer, month integer, day integer);
```

➤ **FEATURES**

1. ISSUE A BOOK

This feature has 4 functions dedicated to it-

- **borrowBK1()**

This function creates a window to enter user ID.

- **borrowBK2()**

This function calls the checkUserAvail(uid) function with the “uid” taken from borrowBK1().

- **checkUserAvail(uid)**

This function takes one parameter- uid i.e. user ID. It checks whether the user has already borrowed three books (return 2) and if the user has pending fine of Rs.100 or more (return 3). Accordingly, it returns the values.

- **borrowBK3()**

According to the value returned by checkUserAvail(uid), this prints the error message or proceeds towards issuing the book. It takes the input of book ID and checks if the copies are available. If yes, then the book is issued. The book ID is put in the database for the corresponding user ID and the count for that book starts.

2. RETURN A BOOK

This feature has 2 functions dedicated to it-

- **returnBK1()**

This function takes the input of user ID and book ID.

- **returnBK2()**

This function checks if the entered book is actually borrowed by the user. If yes, then it deletes the book ID from corresponding user's database entry. It also adds 1 to the available copies of that book.

3. RE-ISSUE A BOOK

This feature has 2 functions dedicated to it-

- **reIssue1()**

This function takes the input of user ID and book ID.

- **reIssue2()**

This function checks if the entered book is actually borrowed by the user. If yes, then it sets the entry of number of days for which that book has been borrowed to 0.

4. SEARCH A BOOK

This feature also has 2 functions dedicated to it-

- **bkSearch1()**

This function takes the input of title of the book.

- **bkSearch2()**

This function searches for the entered title in the database. If the book is found, it displays the book ID, author and number of copies available of that book.

5. FINE CALCULATION AND PAYMENT

This is the most important feature of the project.

The fine is automatically calculated for each day for all the users for all the books borrowed by them. For this purpose, the date of the updation of fine is stored in the database. If the stored date and current date are different, then the fine is updated.

This has 2 functions dedicated to it-

- **updateDate()**

This function stores current date in the database.

- **updateFine(diff)**

This function takes the difference (in number of days) between date stored in database and current date as argument. It checks if a user has borrowed a book for more than 7 days. If yes, then fine is calculated and updated in database.

For fine payment, there are 4 functions-

- **pfWin()**

This function takes user ID as input.

- **checkPayFine()**

This function calls retrieveFine(uid) function to get the pending fine for corresponding user and displays it.

- **retrieveFine(uid)**

This function retrieves fine of corresponding user from the database and returns it.

- **finePaid()**

This function is called once the fine is paid in cash. This function sets the pending fine to Rs.0 for the user. It also sets the book-days for all the books borrowed by that user to 0.

➤ UPDATE DATE AND FINE CALCULATION FUNCTIONS

def updateFine (diff) :

global conn

with conn :

cur = conn.cursor()

cur.execute("""SELECT userID from USERINFO;""")

rn = cur.fetchall()

maxID = 0

for i in rn :

maxID = max(i)

for i in range(1,maxID+1) :

cur.execute("""SELECT booksBorrowed FROM USERINFO

```

WHERE userID=?;''',(i,))
rn = cur.fetchall()
for j in rn :
    bb = j[0]
if bb == 1 :
    cur.execute("""UPDATE USERBOOKS2
SET days1=days1+?
WHERE userID=?;''',(diff,i,))
cur.execute("""SELECT days1 FROM USERBOOKS2
WHERE userID=?;''',(i,))
d = cur.fetchall()
for k in d :
    for l in k :
        if l > 7 :
            fnd = l - 7
            cur.execute("""UPDATE USERINFO
SET fine = ?
WHERE userID = ?;''',(fnd*10 , i,))
if bb == 2 :
    cur.execute("""UPDATE USERBOOKS2
SET days1=days1+? ,
days2=days2+?
WHERE userID=?;''',(diff,diff,i,))
fnd=0
cur.execute("""SELECT days1, days2 FROM USERBOOKS2
WHERE userID=?;''',(i,))
d = cur.fetchall()
for k in d :
    for l in k :
        if l > 7 :
            fnd = fnd + (l-7)
    cur.execute("""UPDATE USERINFO
SET fine = ?
WHERE userID = ?;''',(fnd*10 , i,))
if bb == 3 :
    cur.execute("""UPDATE USERBOOKS2

```

```

SET days1=days1+?,
days2=days2+?,
days3=days3+?
WHERE userID=?;''''',(diff,diff,diff,i,))
fnd=0
cur.execute('''''SELECT days1, days2 FROM USERBOOKS2
WHERE userID=?;''''',(i,))
d = cur.fetchall()
for k in d :
    for l in k :
        if l > 7 :
            fnd = fnd + (l-7)
cur.execute('''''UPDATE USERINFO
SET fine = ?
WHERE userID = ?;''''',(fnd*10 , i,))

```

def updateDate(y1, m1, d1) :

```

global conn
with conn :
    cur = conn.cursor()
    cur.execute('''''UPDATE LASTUPDATE
SET year = ? ,
month = ? ,
day = ?
WHERE pointer = ?;''''',(y1,m1,d1,1,))

```

➤ TECHNOLOGIES USED

• SQLITE3

This is a relational database management system. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

SQLite3 is used in this project for all the database related functions, i.e. storing the data, retrieving the data and searching for the data.

SQLite3 provides support for-

- ☐ Python
- ☐ PHP
- ☐ Java
- ☐ Node.js

• TKINTER

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI.

The name Tkinter comes from Tk interface.

It provides various widgets like-

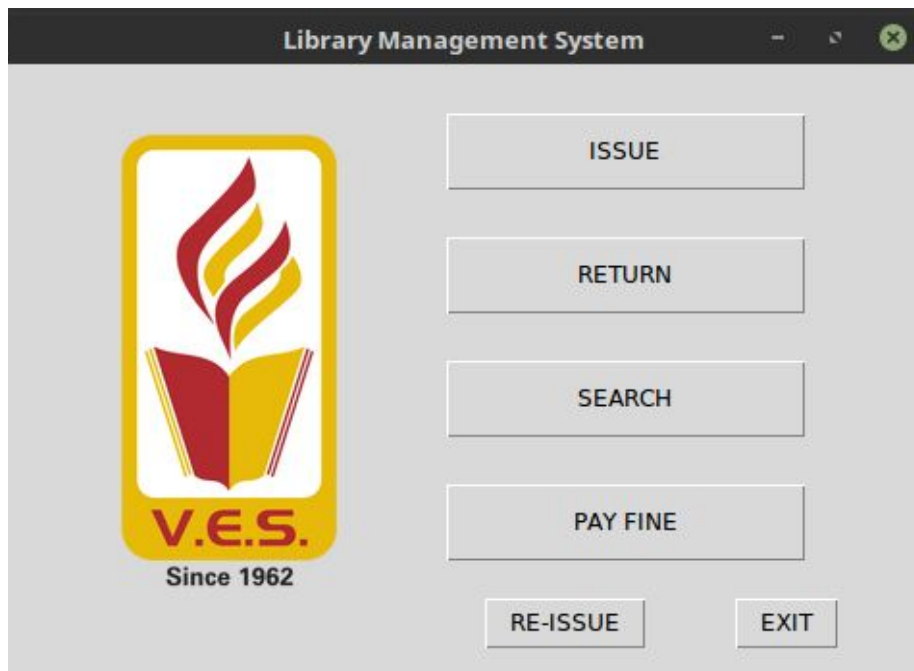
- ☐ Label
- ☐ Entry
- ☐ Button
- ☐ Frame
- ☐ CheckButton
- ☐ Menu
- ☐ MenuButton

• MESSAGEBOX

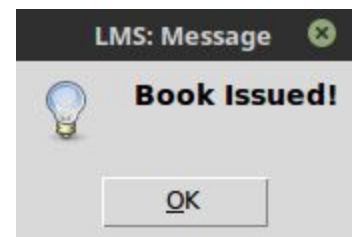
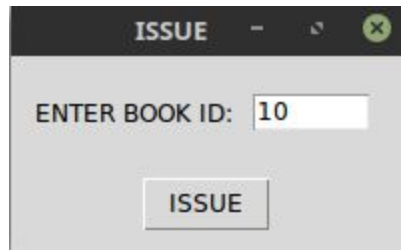
It is a built in package in tkinter. messagebox provides the functionality to display small message window without much hassle. It has three types of message boxes-

- ☐ showinfo ('title' , 'message')
- ☐ showerror ('title' , 'message')
- ☐ showwarning ('title' , 'message')

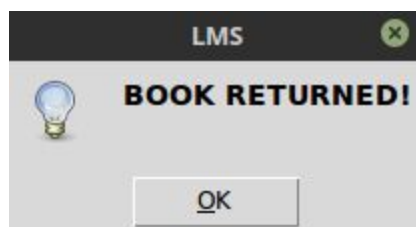
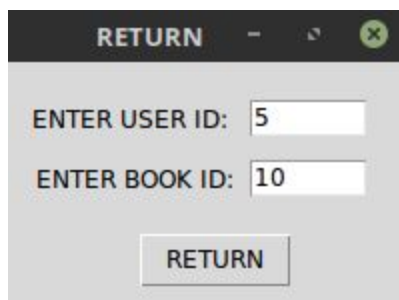
➤ HOME



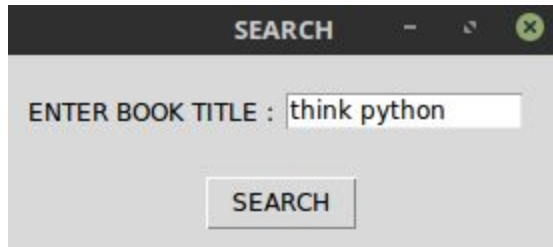
➤ ISSUE A BOOK



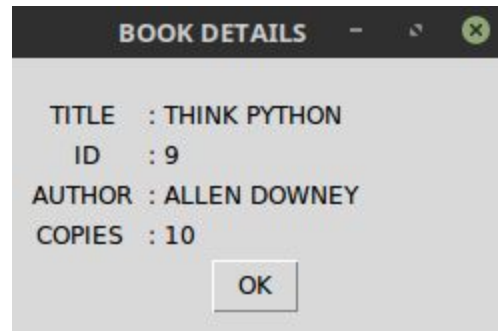
➤ RETURN A BOOK



➤ SEARCH A BOOK




A window titled "SEARCH" with a dark header bar containing a close button. The main area has a label "ENTER BOOK TITLE :" followed by a text input field containing "think python". Below the input field is a button labeled "SEARCH".



A window titled "BOOK DETAILS" with a dark header bar containing minimize, maximize, and close buttons. The main area displays the following information:
TITLE : THINK PYTHON
ID : 9
AUTHOR : ALLEN DOWNEY
COPIES : 10
At the bottom is an "OK" button.

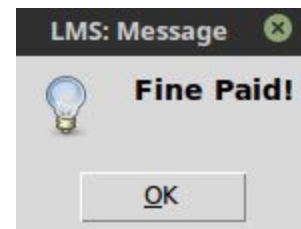
➤ PAY FINE



A window titled "PAY FINE" with a dark header bar containing a close button. The main area has a label "ENTER USER ID:" followed by a text input field containing "2". Below the input field is a button labeled "ENTER".

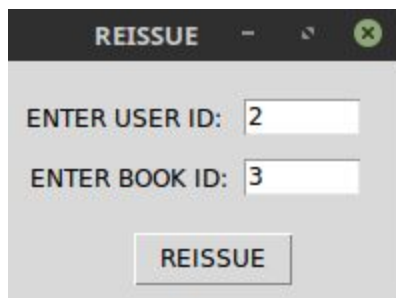


A window titled "PAY FINE" with a dark header bar containing minimize, maximize, and close buttons. The main area displays the text "Fine To Be Paid: Rs.100". Below the text is a button labeled "PAY".

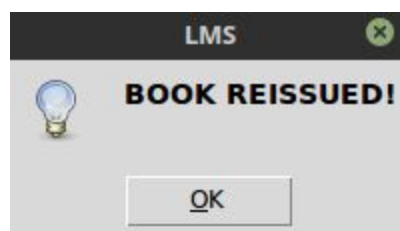


A window titled "LMS: Message" with a dark header bar containing a close button. The main area features a lightbulb icon, the text "Fine Paid!", and an "OK" button.

➤ RE-ISSUE A BOOK



A window titled "REISSUE" with a dark header bar containing a close button. The main area has two labels: "ENTER USER ID:" with a text input field containing "2", and "ENTER BOOK ID:" with a text input field containing "3". Below the input fields is a button labeled "REISSUE".



A window titled "LMS" with a dark header bar containing a close button. The main area features a lightbulb icon, the text "BOOK REISSUED!", and an "OK" button.