Library Management Application

Submitted by:

AP21110011567

AP21110011635

AP21110011576

AP21110011583

Problem Statement:

Software is to be developed for automating a public library. The system should be standalone in nature. It should be designed with a focus on security and should have a break through user interface to make it easy for the people working on it. The following functionalities are required:

Issue of books:

- A members should be able to issue books.
- Each member can issue only a single book.
- The software takes the current system date as the date of issue and
- calculates the date of return.
- The due date for the return of the book is stamped on the book.

Return of books:

- Any person can return the issued books {if they have the member code).
- The system displays the member's details on whose name the book is issued as well as the date of issue and return of the book
- The information is saved and the corresponding updating takes place in the database.

Objective:

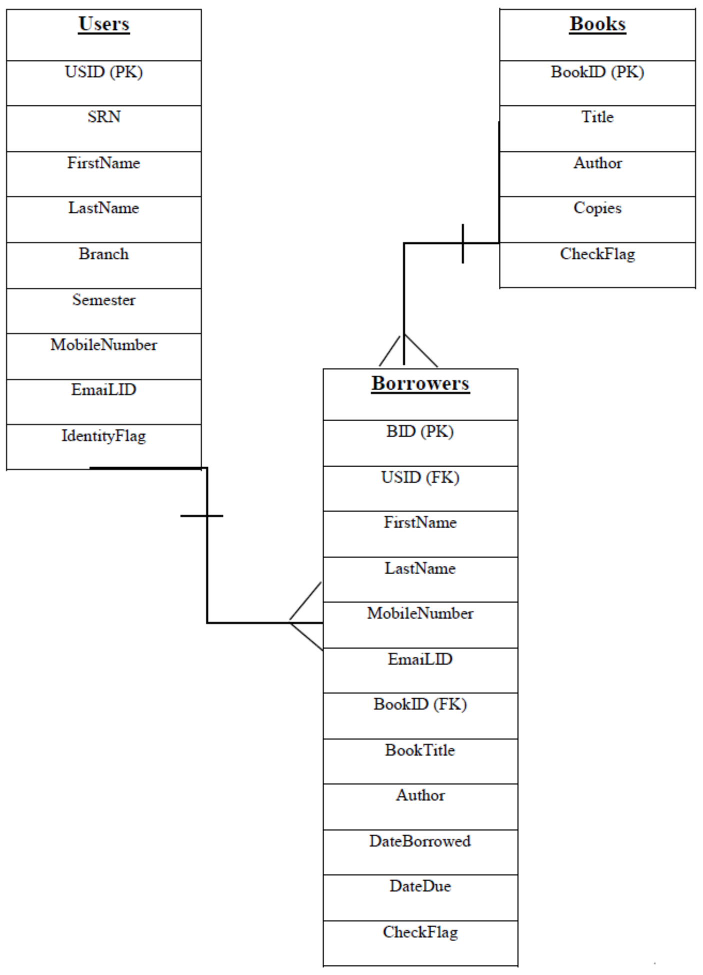
The primary goal is to Effectively Access, Manage, Track, and Perform Daily Library Operations with Ease.The application keeps track of all the books readily available and also the books those have been issued to various lenders for the time period. Readers usually tend to forget the date to return their library books, so this application alerts them & even calculates fine depending on the expiry date. Thus, this digital library management system provides enhanced library functionalities for the modern world with alerts, reminders, notifications and related intelligence

Abstract:

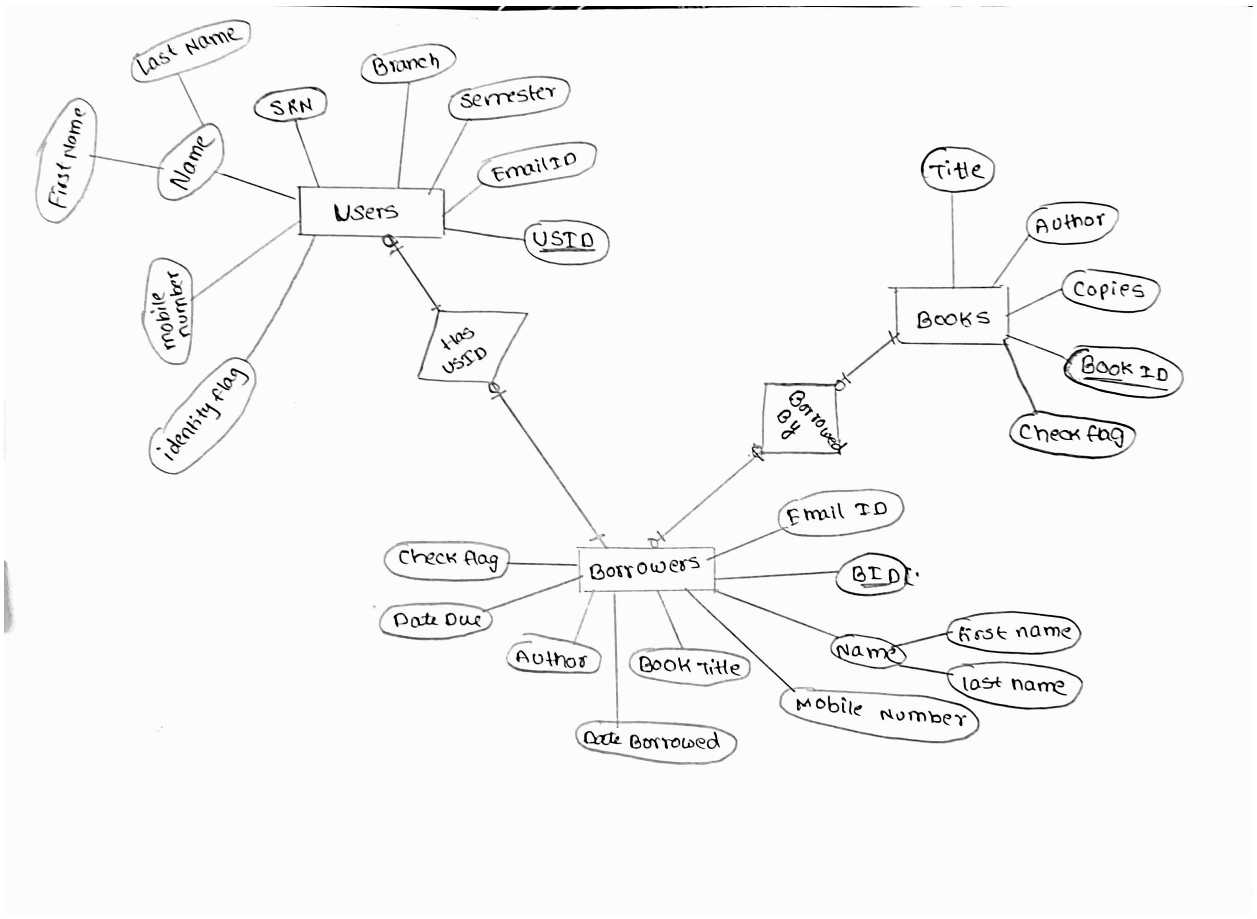
A library management system is an effort to Digitize Library Operations in an Educational Institution; and this is a web application designed to manage all the functions of a library. It helps librarian to maintain the database of new books, books borrowed by members along with their due dates, maintain libra1Y records, track the number of books in the libra1Y, how many books are issued, or how many books have been returned or renewed or late fine charges, etc Librarian can even search for books, issue / reissue books, and manage all the data efficiently and orderly. The purpose of a libra1Y management system is to provide instant and accurate data wrt any type of book, borrowers, users and related and thus digitizing the Educational World

System Design:

ER Diagram:



Schema Diagram and tables :



Users	Borrowers	Books
USID(PK)	BID(PK)	BookID(PK)
SRN	USID(FK)	Title
FirstName	FirstName	Author
LastName	LastName	Copies
Branch	MobileNumber	CheckFlag
Semester	EmaiLID	
MobileNumber	BookID(FK)	
EmaiLID	BookTitle	
IdentityFlag	Author	
	DateBorrowed	
	DateDue	
	CheckFlag	

Implementation:

Frontend:

Python Tkinter

```python
from tkinter import *
from tkinter import ttk
import tkinter.ttk
from functools import partial
```

Backend:

```python
import mysql.connector
```

Code:

main.py

```python
import time
from tkinter import *
from tkinter import ttk
import tkinter.ttk
from functools import partial

from PIL import ImageTk, Image
import mysql.connector
from tkinter import messagebox
from ReturnBook import *
from Auth import *
from fines import *
from ViewBorrowers import *
import Manage_Window
from Manage_Window import *
import searchfunc

import datetime

import mysql.connector

mypass = "root"
mydatabase="db"

con = mysql.connector.connect(host="localhost",user="root",password="root",database="db")
cur = con.cursor(buffered=True)


def myfunc(root1):
   root1.destroy()

   global root,Entry1b,Entry2,Entry2b,Entry3,Entry3b,Entry4

   global member_var, srn_var, first_var, last_var, bookid_var, booktitle_var, author_var, dateborrowed_var, datedue_var, search_var
   root=Tk()
   root.title('Library Management System')
   root.geometry('1600x800')

   member_var=StringVar()
   srn_var=StringVar()
   first_var=StringVar()
   last_var = StringVar()
   bookid_var = StringVar()
   booktitle_var = StringVar()
   author_var = StringVar()
   dateborrowed_var = StringVar()
   datedue_var = StringVar()

   search_var=StringVar()
```

```
l1=Label(text='LIBRARY MANAGEMENT SYSTEM',bg='#ff6e40',fg='black',borderwidth=10,relief=RIDGE,font=('Times New Roman',50,'bold',),padx=2

frame=Frame(root,bg='#ff6e40').place(x=0,y=102,width=1275,height=570)

# &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&& FRONT END &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

# ========================================================================DataFrameLeft===============================================

DataFrameLeft=LabelFrame(frame,bg='#ff6e40',text='Library Member Information',fg='midnightblue',font=('algerian',15,'bold'),borderwidth=

lbl1=Label(frame,text='Member Type',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl1.place(relx=0.03,y=130)
comMember=tkinter.ttk.Combobox(DataFrameLeft,font=('times new roman',18,'bold'),width=13,state='readonly',textvariable=member_var)  #add
comMember['value']=('Student','Lecturer')   #values in the drop down box
comMember.place(relx=0.18, rely=0.19,relwidth=0.35)

lbl1b=Label(DataFrameLeft,text='Book ID',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl1b.place(relx=0.03,rely=0.45)
Entry1b=Entry(bg='white',width=30,font=('times new roman',18,'bold'),textvariable=bookid_var).place(relx=0.18,height=30.5,relwidth=0.35,

lbl2=Label(DataFrameLeft,text='SRN No',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl2.place(relx=0.03,rely=0.25)
Entry2=Entry(bg='white',width=20,font=('times new roman',18,'bold'),textvariable=srn_var).place(relx=0.18,rely=0.25,height=30.5,relwidth

def autoUser():
   userid=srn_var.get()
   fetchuser = "SELECT * FROM users WHERE SRN = " + str(userid)
   cur.execute(fetchuser)
   info = []
   for i in cur:
      info.append(i)
   first_var.set(info[0][2])
   last_var.set(info[0][3])



b1 = Button(root, bg='chocolate', fg='black', borderwidth=5, relief=RAISED, text='Fetch',
            font=('Times new roman', 15, 'bold'), command=autoUser).place(relx=0.49,rely=0.25,height=30.5,relwidth=0.08)

lbl2b=Label(DataFrameLeft,text='Book Title',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl2b.place(relx=0.02,rely=0.53)
Entry2b=Entry(bg='white',width=30,font=('times new roman',18,'bold'),textvariable=booktitle_var).place(relx=0.18,height=30.5,relwidth=0.

lbl3=Label(DataFrameLeft,text='First Name',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl3.place(relx=0.03,rely=0.31)
Entry3=Entry(bg='white',width=20,font=('times new roman',18,'bold'),textvariable=first_var).place(relx=0.18,rely=0.31,height=30.5,relwid

lbl3b=Label(DataFrameLeft,text='Author',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl3b.place(relx=0.03,rely=0.62)
Entry3b=Entry(bg='white',width=30,font=('times new roman',18,'bold'),textvariable=author_var).place(relx=0.18,height=30.5,relwidth=0.35,

lbl4=Label(DataFrameLeft,text='Last Name',bg='#ff6e40',fg='black',font=('Times new roman',18,'bold'))
lbl4.place(relx=0.03,rely=0.37)
Entry4=Entry(bg='white',width=20,font=('times new roman',18,'bold'),textvariable=last_var).place(relx=0.18,rely=0.37,height=30.5,relwidt

# ==================================================================DataFrameRight====================================================

DataFrameRight=LabelFrame(frame,bg='#ff6e40',fg='midnightblue',text='Book Details',font=('algerian',15,'bold'),borderwidth='10',relief=R

# Taking Books from the database into a list :
global b
b = []
author = []

s = "SELECT title FROM books"
cur.execute(s)

for x in cur:
   b.append(x)

s = "SELECT author FROM books"
cur.execute(s)

for x in cur:
   author.append(x)

def select(event=""):
   info=[]
   value=str(a.get(a.curselection()))
   x=value[2:-3]
   s = "SELECT bid, title, author FROM books WHERE title = '%s'" %x
   cur.execute(s)
   for i in cur:
      info.append(i)
      break
   bookid_var.set(info[0][0])
   booktitle_var.set(info[0][1])
   author_var.set(info[0][2])

a=Listbox(DataFrameRight,font=('times new roman',12,'bold'),width=59,height=17)
a.bind("<<ListboxSelect>>",select)
a.place(x=765,y=180)

lbl8b = Label(DataFrameRight, text='Search', bg='#ff6e40', fg='black', font=('Times new roman', 18, 'bold'))
lbl8b.place(x=765, y=130)
Entry8b = Entry(bg='white', width=30, font=('times new roman', 18, 'bold'),textvariable=search_var)
Entry8b.place(x=850, y=130, height=35, relwidth=0.24)

def searching(b):
   searchfunc.search("books",1)
   con2 = mysql.connector.connect(host="localhost", user="root", password="root", database="db")
   cur2= con2.cursor(buffered=True)
   cur2.execute("SELECT title from books WHERE flag=1")
   b1=[]
   for i in cur2:
      b1.append(i)
   if(len(b1)==0)and(len(searchfunc.searchquery)==0):
      a.delete(0, END)
      for item in b:
         a.insert(END, item)
   else:
      b=[]
      b=b1
      a.delete(0,END)
      for item in b:
         a.insert(END, item)
   scrlbar = Scrollbar(DataFrameRight)
   scrlbar.place(x=1245, y=180, height=345)

   a.config(yscrollcommand=scrlbar.set)
   scrlbar.config(command=a.yview())

   scrlbar.config(command=a.yview)

b1 = Button(root, bg='chocolate', fg='black', borderwidth=5, relief=RAISED, text='Go',
            font=('Times new roman',15, 'bold'), command=partial(searching,b)).place(x=1170, y=130, height=35,relwidth=0.07)

for item in b:
   a.insert(END,item)


scrlbar=Scrollbar(DataFrameRight)
scrlbar.place(x=1245,y=180,height=345)

a.config(yscrollcommand=scrlbar.set)
scrlbar.config(command=a.yview())

scrlbar.config(command=a.yview)

def autoRet():
   l=[]
   l.append(srn_var.get())
   l.append(first_var.get())
   l.append(last_var.get())
   extractBid = "SELECT Bookid, BookTitle, Author FROM borrowers WHERE SRN = '%s'" %l[0]
   cur.execute(extractBid)
   info = []
   for i in cur:
      info.append(i)
   bookid_var.set(info[0][0])
   booktitle_var.set(info[0][1])
   author_var.set(info[0][2])

   returnn(l)
   def Exit():
       root.destroy()
       ManageUsers()

   def Reset():
       member_var.set(""),
       srn_var.set(""),
       first_var.set(""),
       last_var.set(""),
       booktitle_var.set(""),
       author_var.set(""),
       bookid_var.set(""),


   # ====================================Buttons==========================================

   frameBtn=Frame(root,bg='powder blue',borderwidth=10,relief=RIDGE,padx=20).place(x=0,y=550,width=1275,height=95)

   b1=Button(frameBtn,bg='chocolate',fg='black',borderwidth=5,relief=RAISED,text='Issue',font=('Times new roman',35,'bold'),command=issuebo
   b2=Button(frameBtn,bg='chocolate',fg='black',borderwidth=5,relief=RAISED,text='Return',font=('Times new roman',35,'bold'),command=autoRe
   b3=Button(frameBtn,bg='chocolate',fg='black',borderwidth=5,relief=RAISED,text='Display Borrowers',font=('Times new roman',35,'bold'),comm
   b4=Button(frameBtn,bg='chocolate',fg='black',borderwidth=5,relief=RAISED,text='Reset',font=('Times new roman',35,'bold'),command=Reset).
   b5=Button(frameBtn,bg='chocolate',fg='black',borderwidth=5,relief=RAISED,text='Exit',font=('Times new roman',35,'bold'),command=partial(


   # ==================================================Info Bar==================================

def issuebook():

   input_bookid = str(bookid_var.get())
   title = str(booktitle_var.get())
   author = str(author_var.get())
   input_userid = int(srn_var.get())
   name = str(first_var.get())
   last_name = str(last_var.get())

   '''con = mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
   cur = con.cursor()'''

   root2 = Tk()
   root2.title("Library")
   root2.geometry("600x500")
   valid = False
   type = "student"
   flag = 0

   cur.execute(
       "SELECT bid FROM books")
   for k in cur:
       a = k[0]
       if (a == input_bookid): # checks if the book id inputed is valid by cross checking with the existing book ids in the book database
           valid = True

   cur.execute("select * from borrowers where SRN=%s", (input_userid,))
   #If the person has some book already : He first has to return that : He will not be allowed to borrow any other book :
   l = []

   for k in cur:
       l.append(k)

   if(len(l)!=0):
       messagebox.showwarning("ERROR","Return the Book which you already have")
       return

   else:

       fetchuser="SELECT SRN,name,Last_name,Branch,semester,mobile_no,email_id FROM users WHERE SRN = "+str(input_userid)
       cur.execute(fetchuser)

       l=[]
       for i in cur:
           l.append(i)

       borrow_date = datetime.date.today()
       duedate = borrow_date + datetime.timedelta(days=14)
       duedate = str(duedate)
       borrow_date = str(borrow_date)
       val = (input_userid,name,last_name,l[0][-2],l[0][-1],input_bookid,title,author,borrow_date,duedate)
       cur.execute(
           "INSERT INTO borrowers (SRN,FirstName,LastName,Mobile,Email,Bookid,BookTitle,Author,DateBorrowed,datedue) values(%s,%s,%s,%s,%s,%s
           val)
       # inserts details of the borrowed book with user details into issued books table
       con.commit()

       fetchbooks = "SELECT copies FROM books WHERE title = '" + title + "'"
       cur.execute(fetchbooks)

       for i in cur:
           cpy = int(i[0][0])
       cpy -= 1
       addSql = "UPDATE books \nSET copies = '%s' \nWHERE title = '%s';" %(str(cpy),str(title))
       cur.execute(addSql)
       con.commit()
       # basic gui code for a dialog box----A summary Box must be displayed too :
       #print("success, book has been issued")

       Canvas1 = Canvas(root2)

       Canvas1.config(bg="#006B38")
       Canvas1.pack(expand=True, fill=BOTH)

       headingFrame1 = Frame(root2, bg="#FFBB00", bd=5)
       headingFrame1.place(relx=0.25, rely=0.1, relwidth=0.5, relheight=0.13)

       headingLabel = Label(headingFrame1, text="Issue Summary", bg='black', fg='white', font=('Courier', 15))
       headingLabel.place(relx=0, rely=0, relwidth=1, relheight=1)

       labelFrame = Frame(root2, bg='black')
       labelFrame.place(relx=0.1, rely=0.3, relwidth=0.8, relheight=0.5)

       game_scroll = Scrollbar(labelFrame)
       game_scroll.pack(side=RIGHT, fill=Y)

       my_game = ttk.Treeview(labelFrame, yscrollcommand=game_scroll.set)

       my_game.pack(fill=BOTH, expand=True)

       game_scroll.config(command=my_game.yview)

       my_game['columns'] = (
           'sl_no','SRN', 'FirstName', 'LastName', 'Mobile', 'Email', 'Bookid', 'BookTitle', 'Author', 'DateBorrowed', 'datedue')

       # format our column
       my_game.column("#0", width=0, stretch=NO)
       my_game.column("sl_no", anchor=CENTER, width=20)
       my_game.column("SRN", anchor=CENTER, width=30)
       my_game.column("FirstName", anchor=CENTER, width=80)
       my_game.column("LastName", anchor=CENTER, width=80)
       my_game.column("Mobile", anchor=CENTER, width=80)
       my_game.column("Email", anchor=CENTER, width=160)
       my_game.column("Bookid", anchor=CENTER, width=20)
       my_game.column("BookTitle", anchor=CENTER, width=120)
       my_game.column("Author", anchor=CENTER, width=80)
       my_game.column("DateBorrowed", anchor=CENTER, width=80)
       my_game.column("datedue", anchor=CENTER, width=80)

       # Create Headings
       my_game.heading("#0", text="", anchor=CENTER)
       my_game.heading("sl_no", text="Sl no", anchor=CENTER)
       my_game.heading("SRN", text="SRN", anchor=CENTER)
       my_game.heading("FirstName", text="FirstName", anchor=CENTER)
       my_game.heading("LastName", text="LastName", anchor=CENTER)
       my_game.heading("Mobile", text="Mobile No", anchor=CENTER)
       my_game.heading("Email", text="Email Id", anchor=CENTER)
       my_game.heading("Bookid", text="BookId", anchor=CENTER)
       my_game.heading("BookTitle", text="BookTitle", anchor=CENTER)
       my_game.heading("Author", text="Author", anchor=CENTER)
       my_game.heading("DateBorrowed", text="DateBorrowed", anchor=CENTER)
       my_game.heading("datedue", text="DateDue", anchor=CENTER)
       c = 1
       y = 0.3
       l=[]
       cur.execute("SELECT SRN,FirstName,LastName,Mobile,Email,Bookid,BookTitle,Author,DateBorrowed,datedue FROM borrowers WHERE SRN = "+str

       m = 0
       for k in cur:
          m += 1
          my_game.insert(parent='', index='end', iid=m, text='', values=(m,) + k)
       my_game.pack()

       quitBtn = Button(root2, text="Quit", bg='#f7f1e3', fg='black', command=root2.destroy)
       quitBtn.place(relx=0.53, rely=0.9, relwidth=0.18, relheight=0.08)

       root2.mainloop()
       return
    #For autoEmails : A window List of all PPl who havent submitted the books on time; -Later

    #Returned Successfully in return screen : + lil error handling : like no user while borrowing : -Imp
    #Add a dynamic alert/Notif in dashboard(eg : "This book" due date is on "this date")
```

main_auth.py

```
from functools import partial
from tkinter import *
from PIL import ImageTk,Image
from Auth import *
from tkinter import messagebox

# declare the window
root = Tk()
root.title("Login Page")
root.minsize(width=400, height=400)
root.geometry("1600x800")

Canvas1 = Canvas(root)

Canvas1.config(bg="#ff6e40")
Canvas1.pack(expand=True, fill=BOTH)

headingFrame1 = Frame(root,bg="#FFBB00",bd=5)
headingFrame1.place(relx=0.17,rely=0.1,relwidth=0.7,relheight=0.16)

headingLabel = Label(headingFrame1, text="WELCOME TO MY LIBRARY", bg='black', fg='white', font=('TIMES NEW ROMAN',40,'bold'))
headingLabel.place(relx=0,rely=0, relwidth=1, relheight=1)

btn0 = Button(root, text="Login", bg='black', fg='white',font=('baskerville old face',25,'bold'),borderwidth=3,relief=RAISED, command=parti
btn0.place(relx=0.20, rely=0.5, relwidth=0.20, relheight=0.1)

btn1 = Button(root, text="Quit", bg='black', fg='white',font=('baskerville old face',25,'bold'),borderwidth=3,relief=RAISED, command=root.d
btn1.place(relx=0.60, rely=0.5, relwidth=0.20, relheight=0.1)

root.mainloop()
```

add_book.py

```
from tkinter import *
from PIL import ImageTk, Image
from tkinter import messagebox
import mysql.connector
from functools import partial
import Manage_Window


def bookRegister():
    # bid = 1,2,3....
    title = bookInfo2.get()
    author = bookInfo3.get()
    #To check if a book already exists in the database :
    check="SELECT bid, title, author FROM books WHERE title = '%s' and author = '%s' " %(title,author)
    cur.execute(check)
    l=[]
    for i in cur:
        l.append(i)
    if(len(l)!=0):
        messagebox.showerror("Error!!", "Book Already Exists")
        return
    else:

        insertBooks = "insert into " + bookTable +" (title,author,copies) values('" + title + "','" + author + "',5)"
        try:
            cur.execute(insertBooks)
            con.commit()
            messagebox.showinfo('Success', "Book added successfully")
        except:
            messagebox.showinfo("Error", "Can't add data into Database")

def addBook(root1):
    root1.destroy()

    global bookInfo1, bookInfo2, bookInfo3, bookInfo4, Canvas1, con, cur, bookTable, root

    root = Tk()
    root.title("Library")
    root.minsize(width=400, height=400)
    root.geometry("1600x800")

    # Add your own database name and password here to reflect in the code
    mypass = "root"
    mydatabase = "db"

    con = mysql.connector.connect(host="localhost", user="root", password=mypass, database=mydatabase)
    cur = con.cursor(buffered=True)

    # Enter Table Names here
    bookTable = "books"
    Canvas1 = Canvas(root)

    Canvas1.config(bg="#ff6e40")
    Canvas1.pack(expand=True, fill=BOTH)

    headingFrame1 = Frame(root, bg="#FFBB00", bd=5)
    headingFrame1.place(relx=0.2, rely=0.05, relwidth=0.7, relheight=0.13)

    headingLabel = Label(headingFrame1, text="Add Books", bg='black', fg='white', font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel.place(relx=0, rely=0, relwidth=1, relheight=1)

    labelFrame = Frame(root, bg='black')
    labelFrame.place(relx=0.05, rely=0.2, relwidth=0.9, relheight=0.7)

    # Title
    lb2 = Label(labelFrame, text="Title ", bg='black', fg='white',font=('courier', 23))
    lb2.place(relx=0.07, rely=0.2)

    bookInfo2 = Entry(labelFrame,font=('times new roman', 18, 'bold'))
    bookInfo2.place(relx=0.2, rely=0.21, relwidth=0.7, relheight=0.08)

    # Book Author
    lb3 = Label(labelFrame, text="Author ", bg='black', fg='white',font=('courier', 23))
    lb3.place(relx=0.07, rely=0.34)

    bookInfo3 = Entry(labelFrame,font=('times new roman', 18, 'bold'))
    bookInfo3.place(relx=0.2, rely=0.35, relwidth=0.7, relheight=0.08)

    # Submit Button
    SubmitBtn = Button(root, text="Submit", bg='#f7f1e3', fg='black',font=('times new roman',20), command=bookRegister)
    SubmitBtn.place(relx=0.3, rely=0.7, relwidth=0.18, relheight=0.08)

    quitBtn = Button(root, text="Quit", bg='#f7f1e3', fg='black', font=('times new roman',20),command=partial(Manage_Window.ManageBooks,roo
    quitBtn.place(relx=0.5, rely=0.7, relwidth=0.18, relheight=0.08)

    root.mainloop()
```

Auth.py

```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
import tkinter.ttk
import mysql.connector
from Manage_Window import *
#from Manage_Users import *


# look into auth fns
```

```
def auth():
    global user
    # global cur
    # con = mysql.connector.connect(host="localhost", user="root", password="root", database="test76", port=3306)
    # cur = con.cursor()

    user = info1.get()
    passs = info2.get()

    Librarian = "Surya"
    password = "surya525"

    if (user != Librarian) or (passs != password):
        messagebox.showerror("ERROR!","Incorrect Username/Password")
        return
        #UserId doesnt exist :
    else:
        ManageWindow(root)

    Submit.destroy()
    labelFrame.destroy()
    lb1.destroy()
    info1.destroy()
    info2.destroy()
    quitBtn.destroy()

def gui_auth(root1):
    root1.destroy()

    global Submit, labelFrame, lb1, info1, info2, quitBtn, root, Canvas1, status

    # declare the window
    root = Tk()
    root.title("Library")
    root.minsize(width=400, height=400)
    root.geometry("1600x800")

    Canvas1 = Canvas(root)

    Canvas1.config(bg="#ff6e40")
    Canvas1.pack(expand=True, fill=BOTH)

    headingFrame1 = Frame(root, bg="#FFBB00", bd=5)
    headingFrame1.place(relx=0.25, rely=0.1, relwidth=0.5, relheight=0.13)

    headingLabel = Label(headingFrame1, text="My Library", bg='black', fg='white', font=('times new roman', 50,'bold'))
    headingLabel.place(relx=0, rely=0, relwidth=1, relheight=1)

    labelFrame = Frame(root, bg='black')
    labelFrame.place(relx=0.1, rely=0.3, relwidth=0.8, relheight=0.5)

    # Username
    lb1 = Label(labelFrame, text="Username", bg='black', fg='white',font=('courier',27))
    lb1.place(relx=0.07, rely=0.2)

    info1 = Entry(labelFrame,font=('times new roman',15,'bold'))
    info1.place(relx=0.3, rely=0.2, relwidth=0.62,relheight=0.12)

    # Password
    lb2 = Label(labelFrame, text="Password", bg='black', fg='white',font=('courier',27))
    lb2.place(relx=0.07, rely=0.4)

    info2 = Entry(labelFrame,font=('times new roman',15,'bold'),show='*')
    info2.place(relx=0.3, rely=0.4, relwidth=0.62,relheight=0.12)

    # Submit Button
    Submit = Button(root, text="Submit", bg='#d1ccc0', fg='black',font=('times new roman',20),command=auth)
    Submit.place(relx=0.28, rely=0.65, relwidth=0.18, relheight=0.08)

    #Quit Button
    quitBtn = Button(root, text="Quit", bg='#d1ccc0', fg='black', command=root.destroy,font=('times new roman',20))
    quitBtn.place(relx=0.53, rely=0.65, relwidth=0.18, relheight=0.08)

    root.mainloop()
```

deletebook.py

```
import tkinter
from tkinter import *
from PIL import ImageTk,Image
from tkinter import messagebox
import mysql.connector
from functools import partial
from tkinter import ttk
import Manage_Window

# Add your own database name and password here to reflect in the code
mypass = "root"
mydatabase="db"

con = mysql.connector.connect(host="localhost",user="root",password="root",database="db")
cur = con.cursor()

# Enter Table Names here
bookTable = "books" #Book Table

def select(e):

    title = bookInfo1.get()

    s=""
    for i in title:
        if(i.isalnum())or (i.isspace()):
            s+=i
    title=s
    for i in range(len(b)):
        if(b[i][0]==title):
            break
    authors=[]
    authors.append(author[i])
    bookInfo2.config(value=authors)

    au=author[i]

    DeleteBtn = Button(root, text="Delete", bg='#f7f1e3', fg='black',font=('times new roman', 20), command=partial(deleteBook,title))
    DeleteBtn.place(relx=0.28, rely=0.83, relwidth=0.18, relheight=0.08)

    quitBtn = Button(root, text="Quit", bg='#f7f1e3', fg='black',font=('times new roman', 20), command=partial(Manage_Window.ManageBooks,ro
    quitBtn.place(relx=0.53, rely=0.83, relwidth=0.18, relheight=0.08)

    root.mainloop()

def delete(root1):
    root1.destroy()

    global bookInfo1,bookInfo2,bookInfo3,bookInfo4,Canvas1,con,cur,bookTable,root,title
    global b,author

    root = Tk()
    root.title("Library")
    root.minsize(width=400,height=400)
    root.geometry("1600x800")


    Canvas1 = Canvas(root)

    Canvas1.config(bg="#006B38")
    Canvas1.pack(expand=True,fill=BOTH)

    headingFrame1 = Frame(root,bg="#FFBB00",bd=5)
    headingFrame1.place(relx=0.25,rely=0.1,relwidth=0.5,relheight=0.13)

    headingLabel = Label(headingFrame1, text="Delete Book", bg='black', fg='white', font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel.place(relx=0,rely=0, relwidth=1, relheight=1)

    labelFrame = Frame(root,bg='black')
    labelFrame.place(relx=0.1,rely=0.3,relwidth=0.8,relheight=0.5)

    b = []
    author = []
    s = "SELECT title FROM books"
    cur.execute(s)

    for x in cur:
        b.append(x)

    s = "SELECT author FROM books"
    cur.execute(s)

    for x in cur:
        author.append(x)

    # creating DropBox
    lb2 = Label(labelFrame, text="Book Title : ", bg='black', fg='white',font=('courier', 15))
    lb2.place(relx=0.07, rely=0.2)

    bookInfo1 = ttk.Combobox(labelFrame,value=b,font=('times new roman',18,'bold'),state='readonly')
    #bookinfo1.current(0)
    bookInfo1.pack()
    bookInfo1.place(relx=0.2, rely=0.21, relwidth=0.7, relheight=0.08)

    #DropBox 2 :
    lb3 = Label(labelFrame, text="Author", bg='black', fg='white',font=('courier', 15))
    lb3.place(relx=0.07, rely=0.4)

    bookInfo2 = ttk.Combobox(labelFrame, value=author,font=('times new roman',18,'bold'),state='readonly')
    # bookinfo2.current(0)
    bookInfo2.pack()
    bookInfo2.place(relx=0.2, rely=0.41, relwidth=0.7, relheight=0.08)

    bookInfo1.bind("<<ComboboxSelected>>", select)

    root.mainloop()

def deleteBook(title):
    deleteSql = "delete from " + bookTable + " where title = '" + title + "'"
    print(deleteSql)
    try:
        cur.execute(deleteSql)
        con.commit()
        messagebox.showinfo('Success', "Book Record Deleted Successfully")
    except:
        messagebox.showinfo("Please enter the correct title of the book")

    bookInfo1.delete(0, END)
#Some Problem here : colon isnt acccpeted as a valid symbol
#What if a person wants to delete a book from entry evnthough another person has borrowed it :
```

fines.py

```
#14days = borrow period
#10 days post borrow : 1rs/day = fine
#Next 15 days : 1rs/day + 10rs/day + fine(compounded)
#If days b/w borrow and returned >39 : Fine+=2000 (book=lost : we will implement this later)


from tkinter import *
from tkinter import messagebox
import datetime

def fines(Bor,Ret):
    l1=Bor.split('-')
    dateB=datetime.date(int(l1[0]),int(l1[1]),int(l1[2]))
    dateR=datetime.date.today()

    days=dateR-dateB
    days=str(days)
    if(days=="0:00:00"):
        return 0
    days=str(days[0:days.index(" ")])
    days=int(days)

    fine=0
    Lost=0
    if(days>14):
        fine+=1*(days-14)
        if(days>24):
            fine+=10*(days-14-10)
            if(days>39):
                Lost=1

    if(Lost==1):
        fine+=2000 #2k Extra Fine : Fr the book + inconvenience caused :
        #If this fine is not paid within the next 15 days : Library Membership Revoked :
    print(fine)
    #For now no lost books

    return fine

#The fine will still be present : Maybe if the person doesnt pay it : he will not get the Hall Ticket fr Exams : LATER IF TIME :
```

FirstAdd.py

```
# Reading Books and authors into b and author From a file : BOOKS & AUTHORS.txt

import mysql.connector

mypass = "root"
mydatabase="db"

con = mysql.connector.connect(host="localhost",user="root",password="root",database="db")
cur = con.cursor()

bookTable="books"
def ret():
    f = open("BOOKS & AUTHORS.txt", "r")
    z = []
    b = []
    for i in f:
        z = f.readlines()

    s1 = ""
    s2 = ""
    author = []
    for i in z:
        s1 = i[3:(i.index('-'))]
        s2 = i[(i.index('>') + 1):(len(i))]
        b.append(s1)
        author.append(s2)

    print(b)
    print(author)

    for i in range(len(b)):
        insertBooks = "insert into " + bookTable + " (title,author,copies) values('" + b[i] + "','" + author[i] + "',5)"
        cur.execute(insertBooks)
        con.commit()
#ret()
```

manage_users.py

```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
import tkinter.ttk
import mysql.connector
from functools import partial
import Manage_Window
import searchfunc
search=None


def ManageUsers(root1):
    root1.destroy()

    global root
    root = Tk()
    root.title("Manage Users")
    root.minsize(width=400, height=400)
    root.geometry("1600x800")

    Canvas2 = Canvas(root)

    Canvas2.config(bg="#ff6e40")
    Canvas2.pack(expand=True, fill=BOTH)

    headingFrame2 = Frame(root, bg="#FFBB00", bd=5)
    headingFrame2.place(relx=0.25, rely=0.1, relwidth=0.5, relheight=0.13)

    headingLabel2 = Label(headingFrame2, text="Manage Users", bg='black', fg='white',
                          font=('times new roman', 30, 'bold'))
    headingLabel2.place(relx=0, rely=0, relwidth=1, relheight=1)

    labelFrame2 = Frame(root, bg='black')
    labelFrame2.place(relx=0.1, rely=0.3, relwidth=0.8, relheight=0.5)

    # Add Users
    Add = Button(labelFrame2, text="Add New User", bg='black', fg='white', font=('baskerville old face', 25),
                 command=partial(AddUserGUI, root))
    Add.place(relx=0, rely=0, relwidth=1, relheight=1 / 3)

    '''# Delete Users
    Delete = Button(labelFrame2, text='Delete User', bg='black', fg='white', font=('baskerville old face', 25))
    Delete.place(relx=0, rely=1 / 4, relwidth=1, relheight=1 / 4)'''

    # Display Current Users  # Need to display from database  # must be done by Sowmesh and Suvan
    Display = Button(labelFrame2, text="Display Current Users", bg='black', fg='white', font=('baskerville old face', 25),command=partial(v
    Display.place(relx=0, rely=1/3, relwidth=1, relheight=1 / 3)

    quit = Button(labelFrame2, text='Quit', bg='black', fg='white', font=('baskerville old face', 25), command=partial(Manage_Window.Manage
    quit.place(relx=0, rely=2 / 3, relwidth=1, relheight=1 / 3)

    root.mainloop()

def viewusers(root1,val):
    root1.destroy()

    global search

    con = mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
    cur = con.cursor(buffered=True)
    cur4=con.cursor(buffered=True)

    if(val==2):

       cur.execute("SELECT SRN,name,Last_name,Branch,semester,mobile_no,email_id FROM users WHERE flag=1")
    if(val==1):
        cur.execute("SELECT SRN,name,Last_name,Branch,semester,mobile_no,email_id FROM users")


    ws  = Tk()
    ws.title('PythonGuides')

    ws['bg'] = '#ff6e40'

    width= ws.winfo_screenwidth()
    height= ws.winfo_screenheight()
    #setting tkinter window size
    ws.geometry("%dx%d+0+0" % (1600, 800))

    headingFrame5 = Frame(ws, bg="#FFBB00", bd=5)
    headingFrame5.place(relx=0.15, rely=0.05, relwidth=0.7, relheight=0.13)

    headingLabel5 = Label(headingFrame5, text="View User", bg='black', fg='white',
                         font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel5.place(relx=0, rely=0, relwidth=1, relheight=1)


    game_frame = Frame(ws,height=800,width=800,bg="black")
    game_frame.place(relx=0.03, rely=0.3, relwidth=0.95, relheight=0.55)

    #scrollbar
    game_scroll = Scrollbar(game_frame)
    game_scroll.pack(side=RIGHT, fill=Y)

    my_game = ttk.Treeview(game_frame,yscrollcommand=game_scroll.set)
    my_game.pack(fill=BOTH,expand=True)

    game_scroll.config(command=my_game.yview)

    #define our column

    my_game['columns'] = ('sl_no', 'Srn', 'name', 'last_name', 'branch','Semester','phone_number','emailid')

    # format our column
    my_game.column("#0", width=0,  stretch=NO)
    my_game.column("sl_no",anchor=CENTER, width=20)
    my_game.column("Srn",anchor=CENTER,width=160)
    my_game.column("name",anchor=CENTER,width=80)
    my_game.column("last_name",anchor=CENTER,width=80)
    my_game.column("branch",anchor=CENTER,width=80)
    my_game.column("Semester",anchor=CENTER, width=80)
    my_game.column("phone_number",anchor=CENTER, width=80)
    my_game.column("emailid",anchor=CENTER, width=80)

    #Create Headings
    my_game.heading("#0",text="",anchor=CENTER)
    my_game.heading("sl_no",text="Sl no",anchor=CENTER)
    my_game.heading("Srn",text="srn",anchor=CENTER)
    my_game.heading("name",text="name",anchor=CENTER)
    my_game.heading("last_name",text="last name",anchor=CENTER)
    my_game.heading("branch",text="branch",anchor=CENTER)
    my_game.heading("Semester",text="Semester",anchor=CENTER)
    my_game.heading("phone_number",text="Phone Number",anchor=CENTER)
    my_game.heading("emailid",text="Email ID",anchor=CENTER)
    #button command=searchfunc

    global searchtype,searchentry

    lbl1 = Label(ws, text='Search Type', bg='#ff6e40', fg='white', font=('Times new roman', 15, 'bold'))
    lbl1.place(x=2, rely=0.19, relwidth=0.4, relheight=0.03)
    searchtype = tkinter.ttk.Combobox(ws, font=('times new roman', 18, 'bold'), width=13, state='readonly')  # adding drop down box
    searchtype['value'] = ('SRN', 'name','Last_name','Branch','semester')  # values in the drop down box
    searchtype.place(relx=0.06, rely=0.23, relwidth=0.18, relheight=0.05)

    def searching():
        searchfunc.search("users",2)
        con.commit()
        viewusers(ws,2)

    searchentry= Entry(ws, font=('times new roman', 18, 'bold'))
    searchentry.place(relx=0.25, rely=0.23, relwidth=0.55, relheight=0.05)
    searchbutton= Button(ws, text="search", bg='#d1ccc0', fg='black',font=('times new roman',20),command=searching)
    searchbutton.place(relx=0.82, rely=0.23, relwidth=0.14, relheight=0.05)
    quitbutton= Button(ws, text="Quit", bg='#d1ccc0', fg='black',font=('times new roman',20),command=partial(ManageUsers,ws))
    quitbutton.place(relx=0.36, rely=0.87, relwidth=0.18, relheight=0.05)



    m=0
    for k in cur:
        m+=1
        my_game.insert(parent='',index='end',iid=m,text='',values=(m,)+k)



def adduser():
    srn=SRN.get()
    emailid=email.get()
    mobile_no2=mobile_no.get()
    a=info5.get()
    b=info5b.get()
    c=comMemberc.get()
    d=comMemberd.get()
    con = mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
    cur = con.cursor()
    if emailid!=""  and mobile_no2.isdigit() and a!="" and b!="" and c!="" and d.isdigit():
     try:

       cur.execute("INSERT INTO users (srn,name,Last_name,Branch,semester,mobile_no,email_id) values(%s,%s,%s,%s,%s,%s,%s)",(srn,a,b,c,d,mo
       con.commit()
       root.destroy()
       except:
          messagebox.showinfo('Error', 'Values entered invalid, Please enter them again:')
          AddUserGUI(root)
    else:
          messagebox.showinfo('Error', 'Values entered invalid, Please enter them again:')
          AddUserGUI(root)






def AddUserGUI(root1):
    #root1.destory()
    #print("enter")
    global info5,info5b,comMemberc,comMemberd,email,mobile_no,SRN,root

    '''root = Tk()
    root.title("Add User")
    root.minsize(width=400, height=400)
    root.geometry("1600x800")
    print("enter 2")

    Canvas5 = Canvas(root)

    Canvas5.config(bg="#ff6e40")
    Canvas5.pack(expand=True, fill=BOTH)
    print("enter 3")

    headingFrame5 = Frame(root, bg="#FFBB00", bd=5)
    headingFrame5.place(relx=0.2, rely=0.05, relwidth=0.7, relheight=0.13)

    headingLabel5 = Label(headingFrame5, text="Add User", bg='black', fg='white',
                         font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel5.place(relx=0, rely=0, relwidth=1, relheight=1)

    labelFrame5 = Frame(root, bg='black')
    labelFrame5.place(relx=0.05, rely=0.2, relwidth=0.9, relheight=0.75)

    # labelFrame5b = Frame(root, bg='black')
    # labelFrame5b.place(relx=0.05, rely=0.25, relwidth=0.8, relheight=0.5)


    # Name
    lb5 = Label(labelFrame5, text="Name", bg='black', fg='white', font=('courier', 23))
    lb5.place(relx=0.04, rely=0.07)

    info5 = Entry(labelFrame5,font=('times new roman',15,'bold'))
    info5.place(relx=0.2, rely=0.07, relwidth=0.7,relheight=0.08)

    lb5b = Label(labelFrame5, text="Last name", bg='black', fg='white', font=('courier', 23))
    lb5b.place(relx=0.04, rely=0.2)

    info5b = Entry(labelFrame5, font=('times new roman', 15, 'bold'))
    info5b.place(relx=0.2, rely=0.21, relwidth=0.7, relheight=0.08)

    lb6b = Label(labelFrame5, text="SRN", bg='black', fg='white', font=('courier', 23))
    lb6b.place(relx=0.04, rely=0.34)

    SRN = Entry(labelFrame5, font=('times new roman', 15, 'bold'))
    SRN.place(relx=0.2, rely=0.35, relwidth=0.7, relheight=0.08)

    lb5c = Label(labelFrame5, text="Branch ", bg='black', fg='white', font=('courier', 23))
    lb5c.place(relx=0.04, rely=0.48)

    comMemberc = tkinter.ttk.Combobox(labelFrame5, font=('times new roman', 15, 'bold'), width=18,
                                 state='readonly')  # adding drop down box
    comMemberc['value'] = ('CSE', 'ECE', 'EEE','Mechanical Engineering','Biotechnology')  # values in the drop down box
    comMemberc.place(relx=0.2, rely=0.49,relwidth=0.7,relheight=0.08)

    lb5d = Label(labelFrame5, text="Semester ", bg='black', fg='white', font=('courier', 23))
    lb5d.place(relx=0.04, rely=0.62)

    comMemberd = tkinter.ttk.Combobox(labelFrame5, font=('times new roman', 15, 'bold'), width=18,
                                  state='readonly')  # adding drop down box
    comMemberd['value'] = (1,2,3,4,5,6,7,8)  # values in the drop down box
    comMemberd.place(relx=0.2, rely=0.63, relwidth=0.7, relheight=0.08)
```

```
lb7 = Label(labelFrame5, text="mobile No", bg='black', fg='white', font=('courier', 23))
lb7.place(relx=0.04, rely=0.75)

mobile_no= Entry(labelFrame5, font=('times new roman', 15, 'bold'))
mobile_no.place(relx=0.2, rely=0.76, relwidth=0.7, relheight=0.08)

lb8 = Label(labelFrame5, text="email id", bg='black', fg='white', font=('courier', 23))
lb8.place(relx=0.04, rely=0.87)

email = Entry(labelFrame5, font=('times new roman', 15, 'bold'))
email.place(relx=0.2, rely=0.88, relwidth=0.7, relheight=0.08)



'''
'''def Submit_5():
     #UserAdd()  # need to define UserAdd fn which is the fn to add the user into database... must be done by sowmesh and suvan
     messagebox.showinfo('Success','User Added Successfully') # need to add fn to show error as well


#Submit Button
Submit5 = Button(labelFrame5, text="Submit", bg='#d1ccc0', fg='black',font=('times new roman',20),command=adduser)
Submit5.place(relx=0.3, rely=0.97, relwidth=0.18, relheight=0.08)'''
root = Tk()
root.title("Add User")
root.minsize(width=400, height=400)
root.geometry("1600x800")

Canvas5 = Canvas(root)

Canvas5.config(bg="#ff6e40")
Canvas5.pack(expand=True, fill=BOTH)

headingFrame5 = Frame(root, bg="#FFBB00", bd=5)
headingFrame5.place(relx=0.2, rely=0.05, relwidth=0.7, relheight=0.13)

headingLabel5 = Label(headingFrame5, text="Add User", bg='black', fg='white',
                     font=('TIMES NEW ROMAN', 30, 'bold'))
headingLabel5.place(relx=0, rely=0, relwidth=1, relheight=1)

labelFrame5 = Frame(root, bg='black')
labelFrame5.place(relx=0.05, rely=0.2, relwidth=0.9, relheight=0.75)

# labelFrame5b = Frame(root, bg='black')
# labelFrame5b.place(relx=0.05, rely=0.25, relwidth=0.8, relheight=0.5)


# Name
lb5 = Label(labelFrame5, text="First Name", bg='black', fg='white', font=('courier', 23))
lb5.place(relx=0.04, rely=0.045)

info5 = Entry(labelFrame5,font=('times new roman',18,'bold'))
info5.place(relx=0.23, rely=0.045, relwidth=0.7,relheight=0.08)

lb5b = Label(labelFrame5, text="Last Name", bg='black', fg='white', font=('courier', 23))
lb5b.place(relx=0.04, rely=0.15)

info5b = Entry(labelFrame5, font=('times new roman', 18, 'bold'))
info5b.place(relx=0.23, rely=0.15, relwidth=0.7, relheight=0.08)

lb6b = Label(labelFrame5, text="SRN", bg='black', fg='white', font=('courier', 23))
lb6b.place(relx=0.04, rely=0.255)

SRN = Entry(labelFrame5, font=('times new roman', 18, 'bold'))
SRN.place(relx=0.23, rely=0.255, relwidth=0.7, relheight=0.08)

lb5c = Label(labelFrame5, text="Branch ", bg='black', fg='white', font=('courier', 23))
lb5c.place(relx=0.04, rely=0.36)

comMemberc = tkinter.ttk.Combobox(labelFrame5, font=('times new roman', 18, 'bold'), width=18,
                                 state='readonly')  # adding drop down box
comMemberc['value'] = ('CSE', 'ECE', 'EEE','Mechanical Engineering','Biotechnology')  # values in the drop down box
comMemberc.place(relx=0.23, rely=0.36,relwidth=0.7,relheight=0.08)

lb5d = Label(labelFrame5, text="Semester ", bg='black', fg='white', font=('courier', 23))
lb5d.place(relx=0.04, rely=0.465)
```

```
    comMemberd = tkinter.ttk.Combobox(labelFrame5, font=('times new roman', 18, 'bold'), width=18,
                                      state='readonly')  # adding drop down box
    comMemberd['value'] = ( 1, 2, 3, 4, 5, 6, 7, 8)  # values in the drop down box
    comMemberd.place(relx=0.23, rely=0.465, relwidth=0.7, relheight=0.08)

    lb7 = Label(labelFrame5, text="Mobile No", bg='black', fg='white', font=('courier', 23))
    lb7.place(relx=0.04, rely=0.57)

    mobile_no= Entry(labelFrame5, font=('times new roman', 18, 'bold'))
    mobile_no.place(relx=0.23, rely=0.57, relwidth=0.7, relheight=0.08)

    lb8 = Label(labelFrame5, text="Email ID", bg='black', fg='white', font=('courier', 23))
    lb8.place(relx=0.04, rely=0.675)

    email = Entry(labelFrame5, font=('times new roman', 18, 'bold'))
    email.place(relx=0.23, rely=0.675, relwidth=0.7, relheight=0.08)

    Submit5 = Button(labelFrame5, text="Submit", bg='#d1ccc0', fg='black',font=('times new roman',20),command=adduser)
    Submit5.place(relx=0.4, rely=0.8, relwidth=0.18, relheight=0.13)

    Quit = Button(labelFrame5, text="Quit", bg='#d1ccc0', fg='black', font=('times new roman', 20),
                  command=root.destroy)
    Quit.place(relx=0.7, rely=0.8, relwidth=0.18, relheight=0.13)
    #print("leaving")

#Horizontal ScrollBar : ViewUsers :
#Search optn available for all "fields"

#Some Problem with AddBook root1.destory()
```

manage_window.py

```
from tkinter import *
from tkinter import ttk
import tkinter.ttk
from Manage_Users import *
from tkinter import messagebox
from main import myfunc
from AddBook import *
from DeleteBook import *
from ViewBooks import *
from functools import partial
import Auth
import webbrowser

global root


def ManageWindow(root1):
    root1.destroy()

    root = Tk()
    root.title("Dashboard")
    root.minsize(width=400, height=400)
    root.geometry("1600x800")

    Canvas1 = Canvas(root)

    Canvas1.config(bg="#ff6e40")
    Canvas1.pack(expand=True, fill=BOTH)

    headingFrame1 = Frame(root, bg="#FFBB00", bd=5)
    headingFrame1.place(relx=0.25, rely=0.1, relwidth=0.5, relheight=0.13)

    headingLabel = Label(headingFrame1, text="DASHBOARD", bg='black', fg='white', font=('baskerville old face', 50,'bold'))
    headingLabel.place(relx=0, rely=0, relwidth=1, relheight=1)

    labelFrame = Frame(root, bg='black')
    labelFrame.place(relx=0.1, rely=0.3, relwidth=0.8, relheight=0.5)

    # Manage Users
    Users = Button(labelFrame, text="Manage Users", bg='black', fg='white', font=('baskerville old face', 25),command=partial(ManageUsers,r
    Users.place(relx=0, rely=0, relwidth=1, relheight=1 / 3)

    # Manage Books
    Books = Button(labelFrame, text="Manage Books", bg='black', fg='white', font=('baskerville old face', 25),command=partial(ManageBooks,r
    Books.place(relx=0, rely=1/3, relwidth=1, relheight=1 / 3)

    #Manage Borrowers
    Borrowers=Button(labelFrame, text='Manage Borrowers', bg='black', fg='white', font=('baskerville old face', 25),command=partial(myfunc,
    Borrowers.place(relx=0,rely=2/3,relwidth=1,relheight=1/3)

    adminframe = Frame(root, bg='black')
    adminframe.place(relx=0.78, rely=0.008, relwidth=0.2, relheight=0.05)
    adminlabel = Label(adminframe, text='User: '+str(Auth.user), font=('times new roman', 17, 'bold'), fg='white', bg='black')
    adminlabel.place(relx=0, rely=0, relheight=1, relwidth=1)

    logoutbtn = Button(root, bg='black', fg='white', text='LOG OUT', font=('baskerville old face', 20, 'bold'),
                       borderwidth=6, relief=RAISED,command=partial(Auth.gui_auth,root))
    logoutbtn.place(relx=0.78, rely=0.063, relheight=0.07, relwidth=0.15)

    def openhtml():
        #convert into pdf
        pdf="LMS_Project_Report_v0.2.pdf"
        webbrowser.open_new_tab("LMS_Project_Report_v0.2.pdf")
    helpbtn = Button(root, bg='black', fg='white', text='Help', font=('times new roman', 20, 'bold'), borderwidth=6,
                     relief=RAISED,command=openhtml)
    helpbtn.place(relx=0.05, rely=0.008, relwidth=0.15, relheight=0.07)




def ManageBooks(root1):
        root1.destroy()

        root = Tk()
        root.title("Manage Books")
        root.minsize(width=400, height=400)
        root.geometry("1600x800")

        Canvas3 = Canvas(root)

        Canvas3.config(bg="#ff6e40")
        Canvas3.pack(expand=True, fill=BOTH)

        headingFrame3 = Frame(root, bg="#FFBB00", bd=5)
        headingFrame3.place(relx=0.25, rely=0.1, relwidth=0.5, relheight=0.13)

        headingLabel3 = Label(headingFrame3, text="Manage Books", bg='black', fg='white',
                              font=('times new roman', 30, 'bold'))
        headingLabel3.place(relx=0, rely=0, relwidth=1, relheight=1)

        labelFrame3 = Frame(root, bg='black')
        labelFrame3.place(relx=0.1, rely=0.3, relwidth=0.8, relheight=0.5)

        # Add Book
        AddB = Button(labelFrame3, text="Add Books", bg='black', fg='white', font=('baskerville old face', 25),command=partial(addBook,root
        AddB.place(relx=0, rely=0, relwidth=1, relheight=1 / 3)

        # Delete Book
        DeleteB = Button(labelFrame3, text="Delete Books", bg='black', fg='white', font=('baskerville old face', 25),command=partial(delete
        DeleteB.place(relx=0, rely=1 / 4, relwidth=1, relheight=1 / 3)

        # View Books
        ViewB = Button(labelFrame3, text="View Books", bg='black', fg='white', font=('baskerville old face', 25),command=partial(View,root,
        ViewB.place(relx=0, rely=2 / 4, relwidth=1, relheight=1 / 3)

        #Quit Button :
        quitBtn = Button(labelFrame3, text="Quit", bg='black', fg='white', font=('baskerville old face', 25), command=partial(ManageWindow,
        quitBtn.place(relx=0, rely=3/4, relwidth=1, relheight=1/3)
```

return_book.py:

```
from tkinter import *
from PIL import ImageTk,Image
from tkinter import messagebox, ttk
import mysql.connector
from fines import *
import datetime
```

```
# Add your own database name and password here to reflect in the code
mypass = "root"
mydatabase="db"

con = mysql.connector.connect(host="localhost",user="root",password="root",database="db")
cur = con.cursor(buffered=True)

# Enter Table Names here
bookTable = "borrowers"
bookTable2="books"

global info,srn
info=[]

def returnn(l):
    srn=l[0]
    first=l[1]
    last=l[2]

    #global SubmitBtn,labelFrame,lb1,bookInfo1,quitBtn,root,Canvas1,status

    extractBid = "SELECT SRN, FirstName, LastName, Bookid, BookTitle, Author, DateBorrowed, datedue FROM "+bookTable+" WHERE SRN = '"+str(s
    deleteSql = "DELETE FROM borrowers WHERE SRN = "+str(srn)
    cpy=0
    fine=0
    cur.execute(extractBid)
    for i in cur:
        info.append(i)

    if(len(info)==0):
        messagebox.showwarning("ERROR!!","No Book Borrowed!")
        return

    fetchbooks = "SELECT copies FROM books WHERE title = '" + info[0][4] + "'"
    cur.execute(fetchbooks)

    for i in cur:
        cpy = int(i[0][0])
    cpy += 1
    addSql = "UPDATE books \nSET copies = '%s' \nWHERE title = '%s';" % (str(cpy), str(info[0][4]))
    cur.execute(addSql)
    con.commit()
    fine = fines(info[0][6], info[0][7])
    #Now ill have to show a window of the ReturnBook Summary

    cur.execute(deleteSql)
    con.commit()
    info[0]=list(info[0])
    info[0].append(fine)
    info[0]=tuple(info[0])

    returnBook()
    return

def returnBook():
    global quitBtn,Canvas1,con,cur,root,labelFrame

    root = Tk()
    root.title("Library")
    root.minsize(width=400,height=400)
    root.geometry("600x500")


    Canvas1 = Canvas(root)

    Canvas1.config(bg="#006B38")
    Canvas1.pack(expand=True,fill=BOTH)


    headingFrame1 = Frame(root,bg="#FFBB00",bd=5)
    headingFrame1.place(relx=0.25,rely=0.1,relwidth=0.5,relheight=0.13)

    headingLabel = Label(headingFrame1, text="Return Summary", bg='black', fg='white', font=('Courier',15))
    headingLabel.place(relx=0,rely=0, relwidth=1, relheight=1)

    labelFrame = Frame(root,bg='black')
    labelFrame.place(relx=0.1,rely=0.3,relwidth=0.8,relheight=0.5)
    game_scroll = Scrollbar(labelFrame)
    game_scroll.pack(side=RIGHT, fill=Y)

    game_scroll = Scrollbar(labelFrame, orient='horizontal')
    game_scroll.pack(side=BOTTOM, fill=X)

    my_game = ttk.Treeview(labelFrame, yscrollcommand=game_scroll.set, xscrollcommand=game_scroll.set)

    my_game.pack(fill=BOTH, expand=True)

    game_scroll.config(command=my_game.yview)
    game_scroll.config(command=my_game.xview)

    my_game['columns'] = (
    'SRN', 'FirstName', 'LastName', 'Bookid', 'BookTitle', 'Author', 'Fine')

    my_game.column("#0", width=0, stretch=NO)
    my_game.column("SRN", anchor=CENTER, width=30)
    my_game.column("FirstName", anchor=CENTER, width=80)
    my_game.column("LastName", anchor=CENTER, width=80)
    my_game.column("Bookid", anchor=CENTER, width=20)
    my_game.column("BookTitle", anchor=CENTER, width=120)
    my_game.column("Author", anchor=CENTER, width=80)
    my_game.column("Fine", anchor=CENTER, width=50)

    # Create Headings
    my_game.heading("#0", text="", anchor=CENTER)
    my_game.heading("SRN", text="SRN", anchor=CENTER)
    my_game.heading("FirstName", text="FirstName", anchor=CENTER)
    my_game.heading("LastName", text="LastName", anchor=CENTER)
    my_game.heading("Bookid", text="BookID", anchor=CENTER)
    my_game.heading("BookTitle", text="BookTitle", anchor=CENTER)
    my_game.heading("Author", text="Author", anchor=CENTER)
    my_game.heading("Fine", text="Fine", anchor=CENTER)
    '''for i in range(len(info[0])-3):
        print(info[0][i])'''
    c=0
    info[0]=list(info[0])
    info[0].pop(6)
    info[0].pop(6)
    info[0]=tuple(info[0])
    for i in info:
        my_game.insert(parent='', index='end', text='', values=i)
        c+=1
    my_game.pack()

    quitBtn = Button(root,text="Quit",bg='#f7f1e3', fg='black', command=root.destroy)
    quitBtn.place(relx=0.53,rely=0.9, relwidth=0.18,relheight=0.08)

    root.mainloop()
#For Custom Error msg if user cant borrow : cuz he is not registered in the users database :
```

search_book.py

```
import mysql.connector
import main
import Manage_Users
import ViewBooks
import ViewBorrowers
def search(table,val):
    global searchquery
    if(val==1):
        searchquery = main.search_var.get()
        column="title"
    if(val==2):
        column=Manage_Users.searchtype.get()
        searchquery=Manage_Users.searchentry.get()
        if(column=="SRN")or(column=="semester"):
            '''if(searchquery==""):
                return'''
            searchquery=int(searchquery)
    if(val==3):
        column=ViewBooks.searchtype.get()
        searchquery=ViewBooks.searchentry.get()
        '''if (searchquery == ""):
            return'''
    if(val==4):
        column=ViewBorrowers.searchtype.get()
        searchquery = ViewBorrowers.searchentry.get()


    con=mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
    cur=con.cursor(buffered=True)
    cur_update=con.cursor(buffered=True)
    cur_ColumnCheck=con.cursor(buffered=True)

    try:
        cur.execute("SELECT * FROM "+table)
    except:
        print("searchfunc: table does not exist. table names are case sensitive")
        return
    cur.execute("Update "+table+" SET flag=0")
    con.commit()

    if (searchquery == ""):
        return

    try:
        cur.execute("SELECT "+column+" from "+table)
    except:
        print("searchfunc: column in the table does not exist. column names are case sensitive")
        return
    try:
        if type(searchquery)==str:

           cur_ColumnCheck.execute("SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE table_name =%s AND COLUMN_NAME =%s",(table,colum
           for k in cur_ColumnCheck:
               if k[0]!="varchar":
                  print("searchfunc: searchquery data type and column data type are not the same")
                  return
                  # checking if column data type is str. if column data type is not str, program wll exit the function.

           searchquery=searchquery.split()
           for k in cur:
              try:
                  s=k[0].split()
              except:
                  print("searchfunc: None data type found. skipping this iteration")
                  continue
                  # None.split() causes error therefore, handling error by skipping iteration. this may not even happen but just in case if

              for m in s:
                 for l in searchquery:
                    if (m.upper()).startswith(l.upper()):
                       cur_update.execute("UPDATE "+table+" SET flag=1 WHERE "+column+" =%s",(k[0],))
                       con.commit()

        elif type(searchquery)==int:

           cur_ColumnCheck.execute("SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE table_name = %s AND COLUMN_NAME = %s",(table,col
           for k in cur_ColumnCheck:
               if k[0]!="int":
                  print("searchfunc: searchquery data type and column data type are not the same")
                  return
                  # checking if column data type is int. if column data type is not int, program wll exit the function.

           for k in cur:
               s=k[0]
               if searchquery==s:
                   cur_update.execute("UPDATE "+table+" SET flag=1 WHERE "+column+" =%s",(s,))
                   con.commit()

        else:
              print("searchfunc: search query data type not int or string. only str and int supported")
              return
    except:
           print("searchfunc: error when performing search")
           return

#if no book records exists :
```

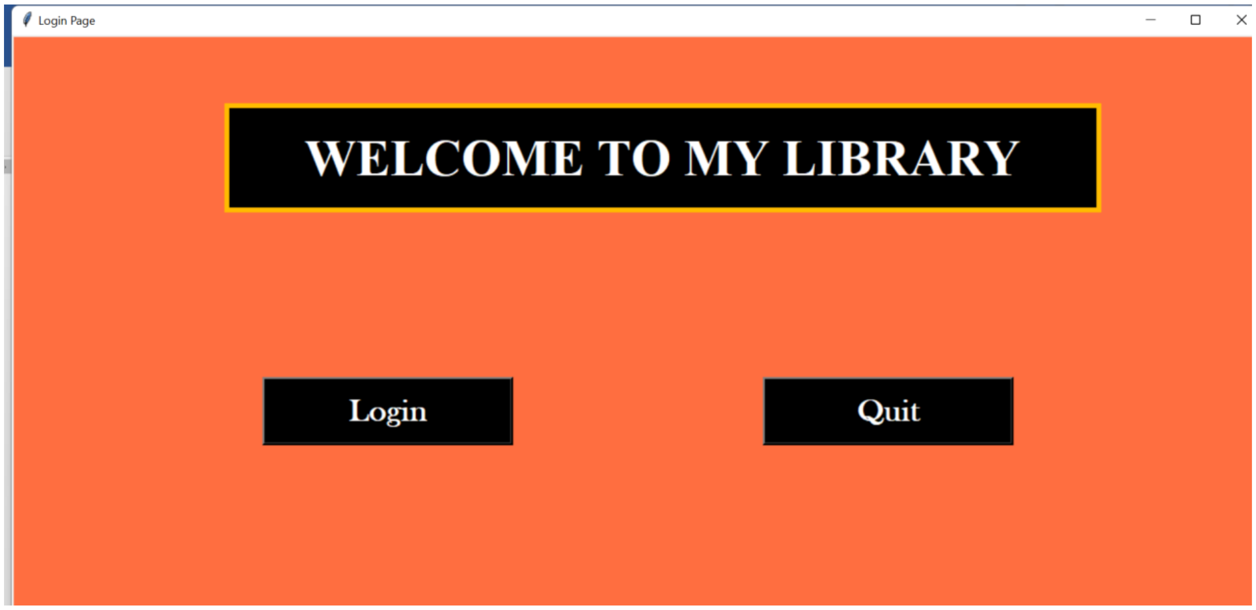
ViewBooks.py

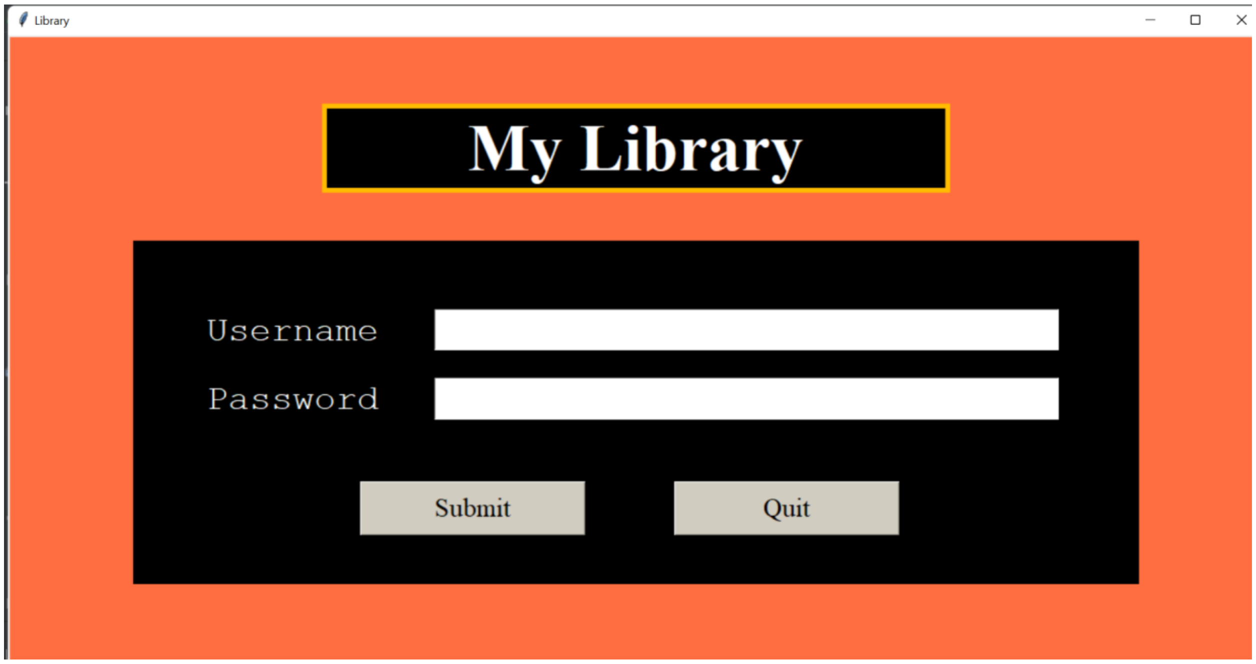
```
from functools import partial
from tkinter import *
from tkinter import ttk
import tkinter.ttk
import Manage_Window

import mysql.connector

import searchfunc

# Enter Table Names here
bookTable = "books"


def View(root1,val):
    root1.destroy()

    global search
    con = mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
    cur = con.cursor(buffered=True)
    cur4 = con.cursor(buffered=True)

    # if search!=None and search!="":
    if (val == 2):
        cur.execute("SELECT bid,title,author FROM books WHERE flag=1")
    if (val == 1):
        cur.execute("SELECT bid,title,author FROM books")

    ws = Tk()
    ws.title('PythonGuides')

    ws['bg'] = '#ff6e40'

    width = ws.winfo_screenwidth()
    height = ws.winfo_screenheight()
    # setting tkinter window size
    ws.geometry("%dx%d+0+0" % (1600, 800))

    headingFrame5 = Frame(ws, bg="#FFBB00", bd=5)
    headingFrame5.place(relx=0.15, rely=0.05, relwidth=0.7, relheight=0.13)

    headingLabel5 = Label(headingFrame5, text="View Books", bg='black', fg='white',
                          font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel5.place(relx=0, rely=0, relwidth=1, relheight=1)

    game_frame = Frame(ws, height=800, width=800, bg="black")
    game_frame.place(relx=0.03, rely=0.3, relwidth=0.95, relheight=0.55)

    # scrollbar
    game_scroll = Scrollbar(game_frame)
    game_scroll.pack(side=RIGHT, fill=Y)

    my_game = ttk.Treeview(game_frame, yscrollcommand=game_scroll.set)

    my_game.pack(fill=BOTH, expand=True)

    game_scroll.config(command=my_game.yview)

    # define our column

    my_game['columns'] = ('bid', 'title', 'author')

    # format our column
    my_game.column("#0", width=0, stretch=NO)
    my_game.column("bid", anchor=CENTER, width=160)
    my_game.column("title", anchor=CENTER, width=160)
    my_game.column("author", anchor=CENTER, width=160)

    # Create Headings
    my_game.heading("#0", text="", anchor=CENTER)
    my_game.heading("bid", text="BID", anchor=CENTER)
    my_game.heading("title", text="Title", anchor=CENTER)
    my_game.heading("author", text="Author", anchor=CENTER)
    # button command=searchfunc
    global searchtype, searchentry

    lbl1 = Label(ws, text='Search Type', bg='#ff6e40', fg='black', font=('Times new roman', 15, 'bold'))
    lbl1.place(x=2, rely=0.19, relwidth=0.4, relheight=0.03)
    searchtype = tkinter.ttk.Combobox(ws, font=('times new roman', 18, 'bold'), width=13,
                                  state='readonly')  # adding drop down box
    searchtype['value'] = ('title','author')  # values in the drop down box
    searchtype.place(relx=0.06, rely=0.23, relwidth=0.18, relheight=0.05)

    def searching():
        searchfunc.search("books", 3)
        con.commit()
        View(ws, 2)

    searchentry = Entry(ws, font=('times new roman', 18, 'bold'))
    searchentry.place(relx=0.25, rely=0.23, relwidth=0.55, relheight=0.05)
    searchbutton = Button(ws, text="search", bg='#d1ccc0', fg='black', font=('times new roman', 20), command=searching)
    searchbutton.place(relx=0.82, rely=0.23, relwidth=0.14, relheight=0.05)
    quitbutton = Button(ws, text="Quit", bg='#d1ccc0', fg='black', font=('times new roman', 20),
                        command=partial(Manage_Window.ManageBooks, ws))
    quitbutton.place(relx=0.36, rely=0.87, relwidth=0.18, relheight=0.05)

    m = 0
    for k in cur:
        my_game.insert(parent='', index='end', text='', values= k)
#Enter column : msgbox if none selected :
```

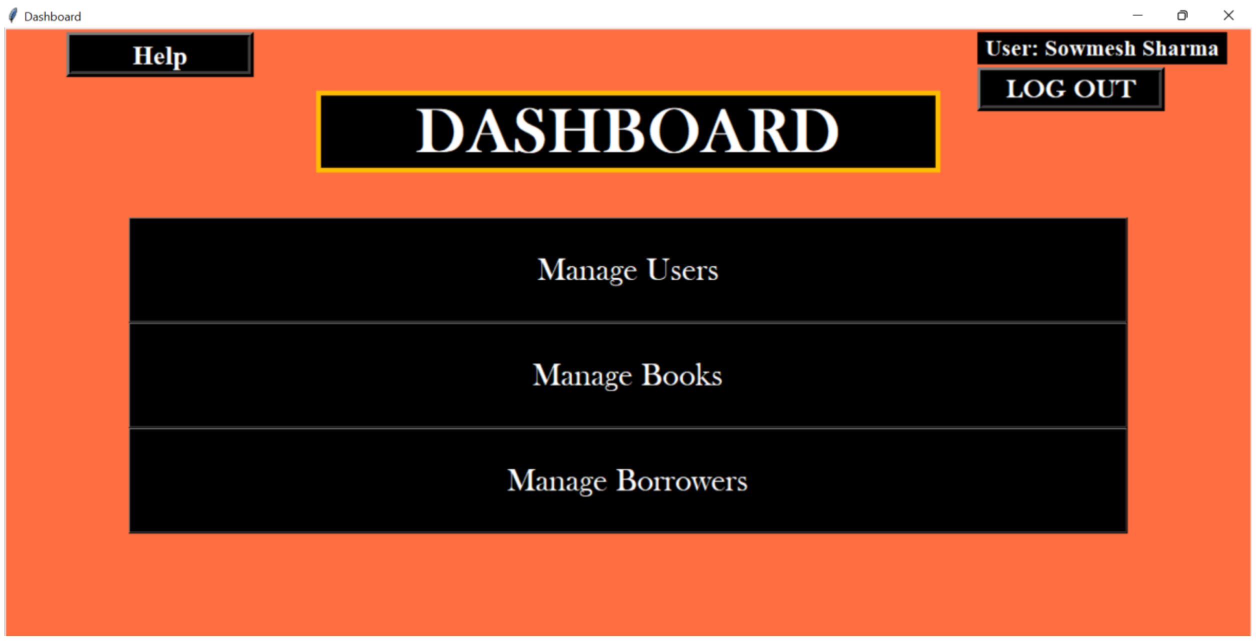
ViewBorrowers.py

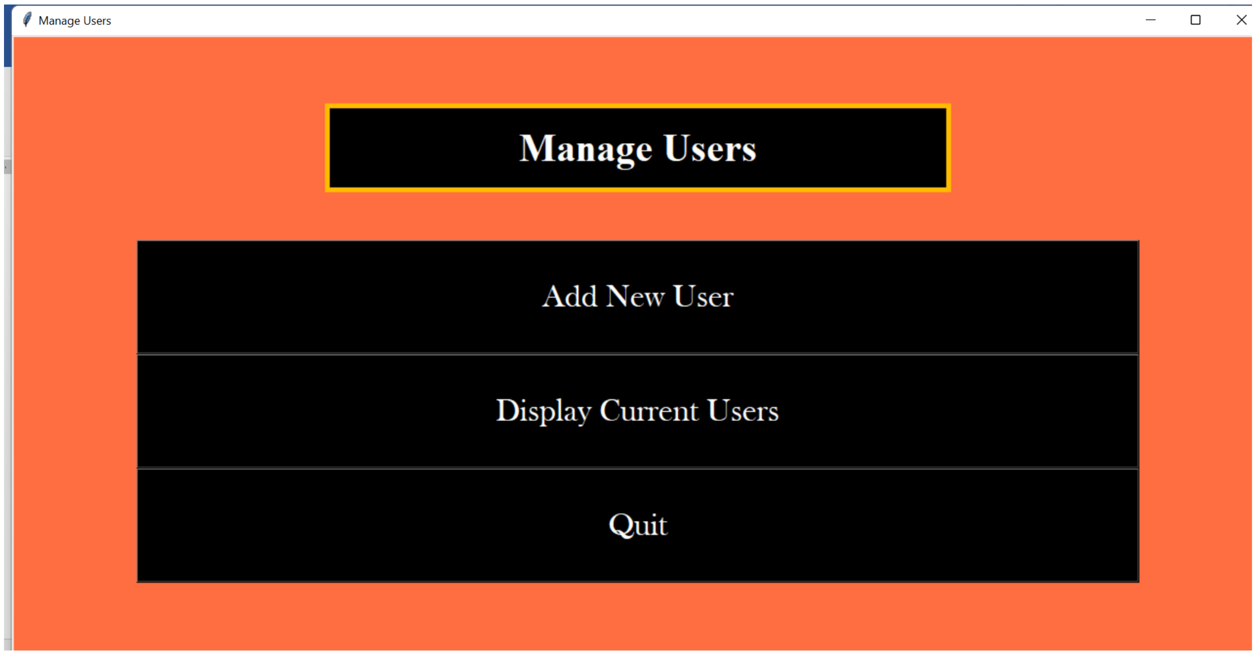
```
from functools import partial
from tkinter import *
from tkinter import ttk
import tkinter.ttk
import main

import mysql.connector

import searchfunc

# Enter Table Names here
bookTable = "borrowers"


def View(root1,val):
    root1.destroy()

    global search
    con = mysql.connector.connect(host="localhost", user="root", password="root", database="db", port=3306)
    cur = con.cursor(buffered=True)
    cur4 = con.cursor(buffered=True)

    if (val == 2):
        cur.execute("SELECT SRN,FirstName,LastName,Email,Bookid,BookTitle,Author,DateBorrowed,datedue FROM borrowers WHERE flag=1")
    if (val == 1):
        cur.execute("SELECT SRN,FirstName,LastName,Email,Bookid,BookTitle,Author,DateBorrowed,datedue FROM borrowers")

    ws = Tk()
    ws.title('PythonGuides')

    ws['bg'] = '#ff6e40'

    width = ws.winfo_screenwidth()
    height = ws.winfo_screenheight()
    # setting tkinter window size
    ws.geometry("%dx%d+0+0" % (1600, 800))

    headingFrame5 = Frame(ws, bg="#FFBB00", bd=5)
    headingFrame5.place(relx=0.15, rely=0.05, relwidth=0.7, relheight=0.13)

    headingLabel5 = Label(headingFrame5, text="View Borrowers", bg='black', fg='white',
                          font=('TIMES NEW ROMAN', 30, 'bold'))
    headingLabel5.place(relx=0, rely=0, relwidth=1, relheight=1)

    game_frame = Frame(ws, height=800, width=800, bg="black")
    game_frame.place(relx=0.03, rely=0.3, relwidth=0.95, relheight=0.55)

    # scrollbar
    game_scroll = Scrollbar(game_frame)
    game_scroll.pack(side=RIGHT, fill=Y)

    game_scroll = Scrollbar(game_frame, orient='horizontal')
    game_scroll.pack(side=BOTTOM, fill=X)

    my_game = ttk.Treeview(game_frame, yscrollcommand=game_scroll.set, xscrollcommand=game_scroll.set)

    my_game.pack(fill=BOTH, expand=True)

    game_scroll.config(command=my_game.yview)
    game_scroll.config(command=my_game.xview)

    # define our column

    my_game['columns'] = ('SRN','FirstName','LastName','Email','Bookid','BookTitle','Author','DateBorrowed','datedue')

    # format our column
    my_game.column("#0", width=0, stretch=NO)
    my_game.column("SRN", anchor=CENTER, width=30)
    my_game.column("FirstName", anchor=CENTER, width=80)
    my_game.column("LastName", anchor=CENTER, width=80)
    my_game.column("Email", anchor=CENTER, width=120)
    my_game.column("Bookid", anchor=CENTER, width=20)
    my_game.column("BookTitle", anchor=CENTER, width=120)
    my_game.column("Author", anchor=CENTER, width=80)
    my_game.column("DateBorrowed", anchor=CENTER, width=80)
    my_game.column("datedue", anchor=CENTER, width=80)

    # Create Headings
    my_game.heading("#0", text="", anchor=CENTER)
    my_game.heading("SRN", text="SRN", anchor=CENTER)
    my_game.heading("FirstName", text="FirstName", anchor=CENTER)
    my_game.heading("LastName", text="LastName", anchor=CENTER)
    my_game.heading("Email", text="Email", anchor=CENTER)
    my_game.heading("Bookid", text="BookID", anchor=CENTER)
    my_game.heading("BookTitle", text="BookTitle", anchor=CENTER)
    my_game.heading("Author", text="Author", anchor=CENTER)
    my_game.heading("DateBorrowed", text="DateBorrowed", anchor=CENTER)
    my_game.heading("datedue", text="datedue", anchor=CENTER)
    # button command=searchfunc

    global searchtype, searchentry

    lbl1 = Label(ws, text='Search Type', bg='#ff6e40', fg='black', font=('Times new roman', 15, 'bold'))
    lbl1.place(x=2, rely=0.19, relwidth=0.4, relheight=0.03)
    searchtype = tkinter.ttk.Combobox(ws, font=('times new roman', 18, 'bold'), width=13,
                                      state='readonly')  # adding drop down box
    searchtype['value'] = ('SRN','FirstName','LastName','BookTitle','Author','DateBorrowed','datedue')  # values in the drop down box
    searchtype.place(relx=0.06, rely=0.23, relwidth=0.18, relheight=0.05)

    def searching():
        searchfunc.search("borrowers", 4)
        con.commit()
        View(ws, 2)

    searchentry = Entry(ws, font=('times new roman', 18, 'bold'))
    searchentry.place(relx=0.25, rely=0.23, relwidth=0.55, relheight=0.05)
    searchbutton = Button(ws, text="Search", bg='#d1ccc0', fg='black', font=('times new roman', 20), command=searching)
    searchbutton.place(relx=0.82, rely=0.23, relwidth=0.14, relheight=0.05)
    quitbutton = Button(ws, text="Quit", bg='#d1ccc0', fg='black', font=('times new roman', 20),
                        command=partial(main.myfunc,ws))
    quitbutton.place(relx=0.36, rely=0.87, relwidth=0.18, relheight=0.05)

    m = 0
    for k in cur:
        my_game.insert(parent='', index='end', text='', values= k)
```

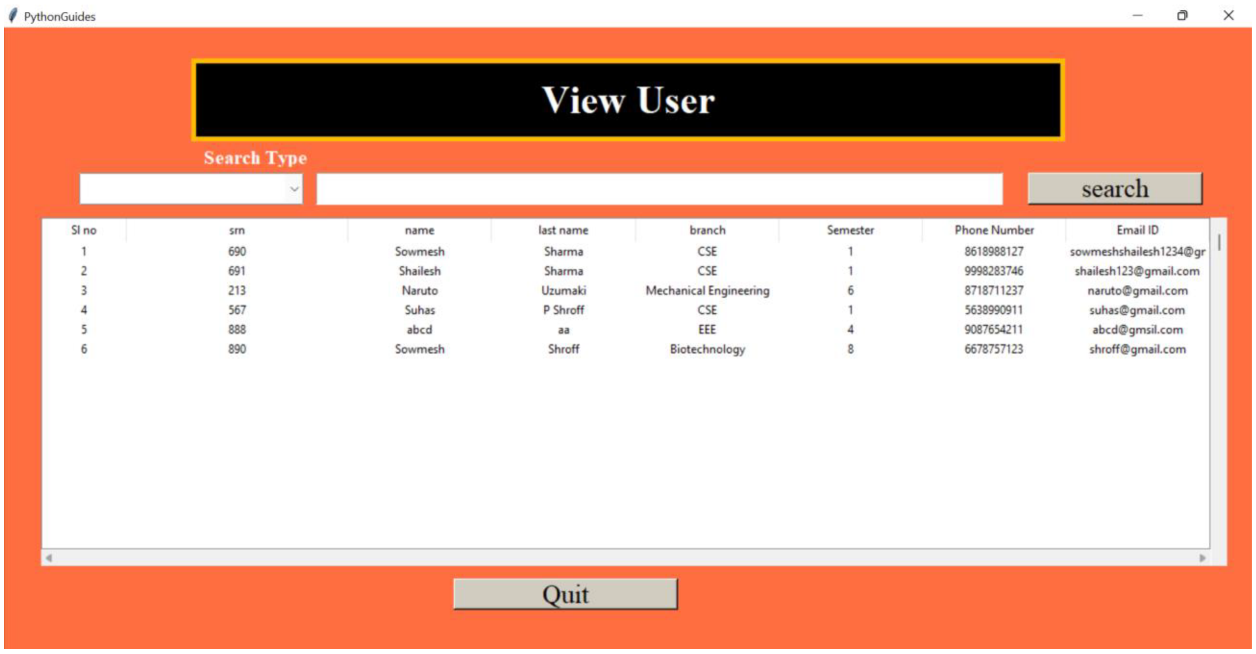
Screenshots:


Login Page
WELCOME TO MY LIBRARY
Login
Quit

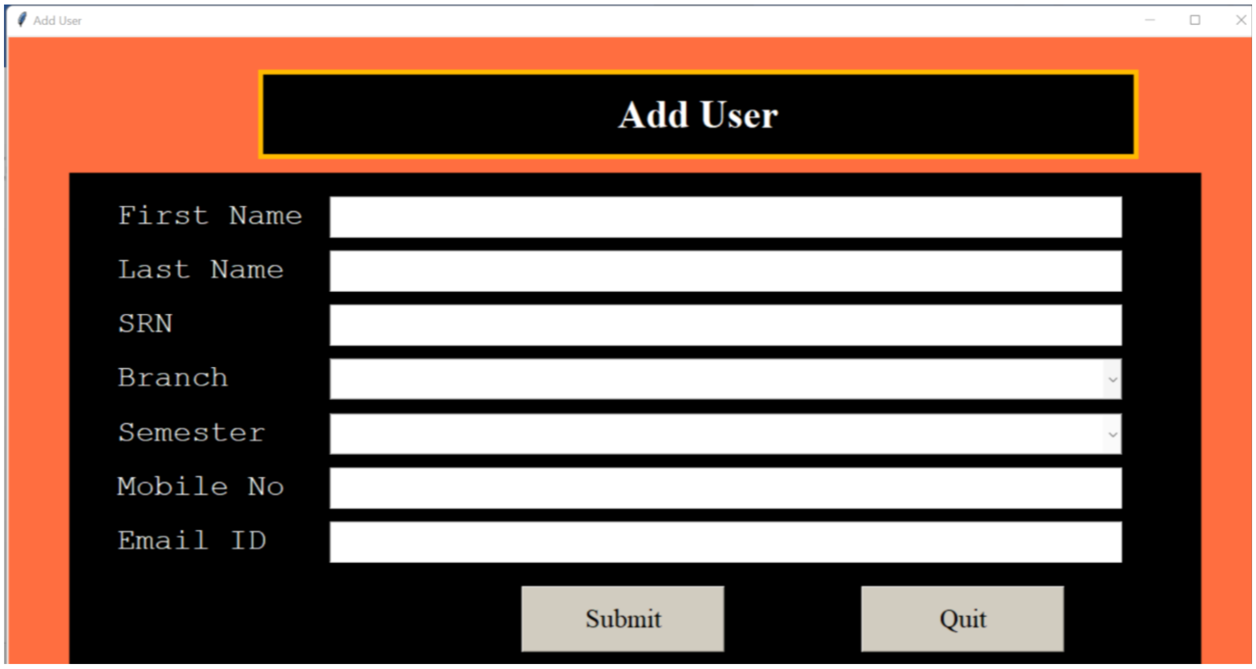

Library
My Library
Username
Password
Submit
Quit

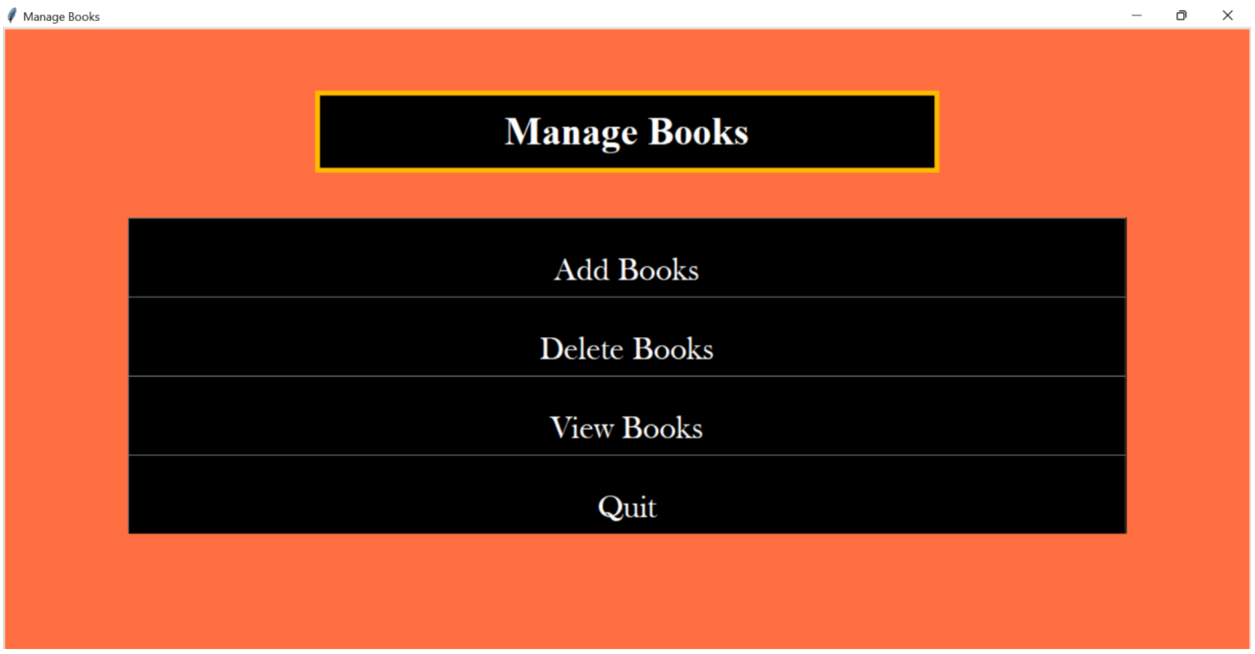

Dashboard
Help
User: Sowmesh Sharma
LOG OUT
DASHBOARD
Manage Users
Manage Books
Manage Borrowers

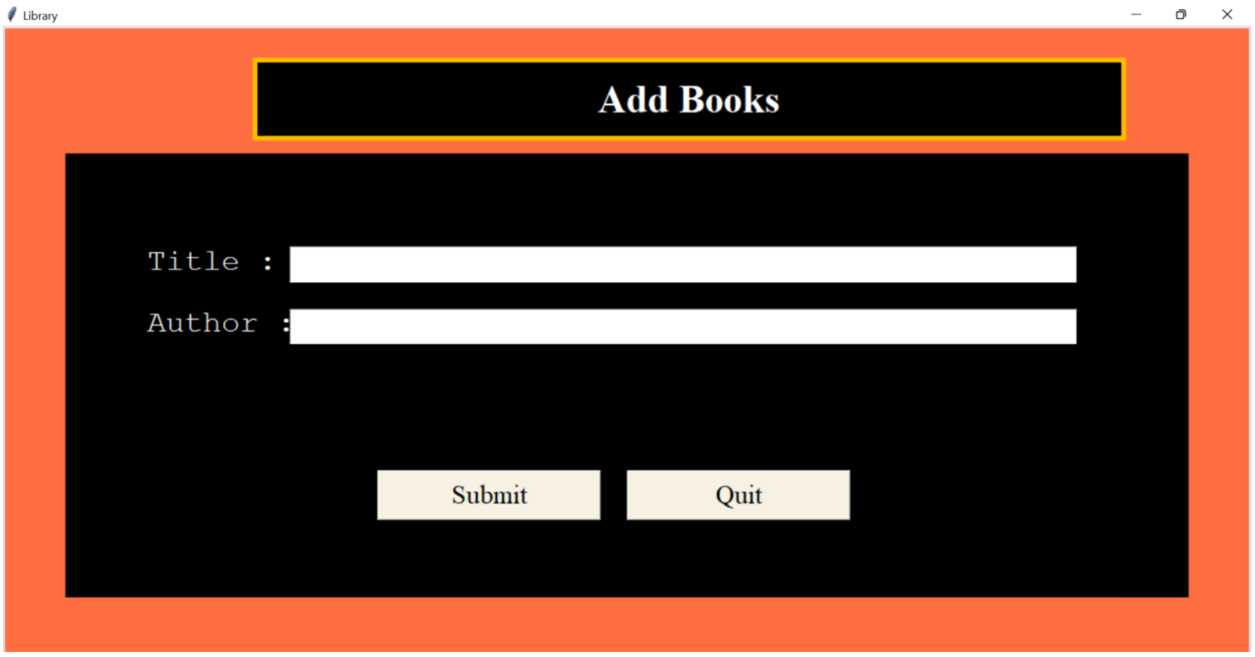

Manage Users
Manage Users
Add New User
Display Current Users
Quit

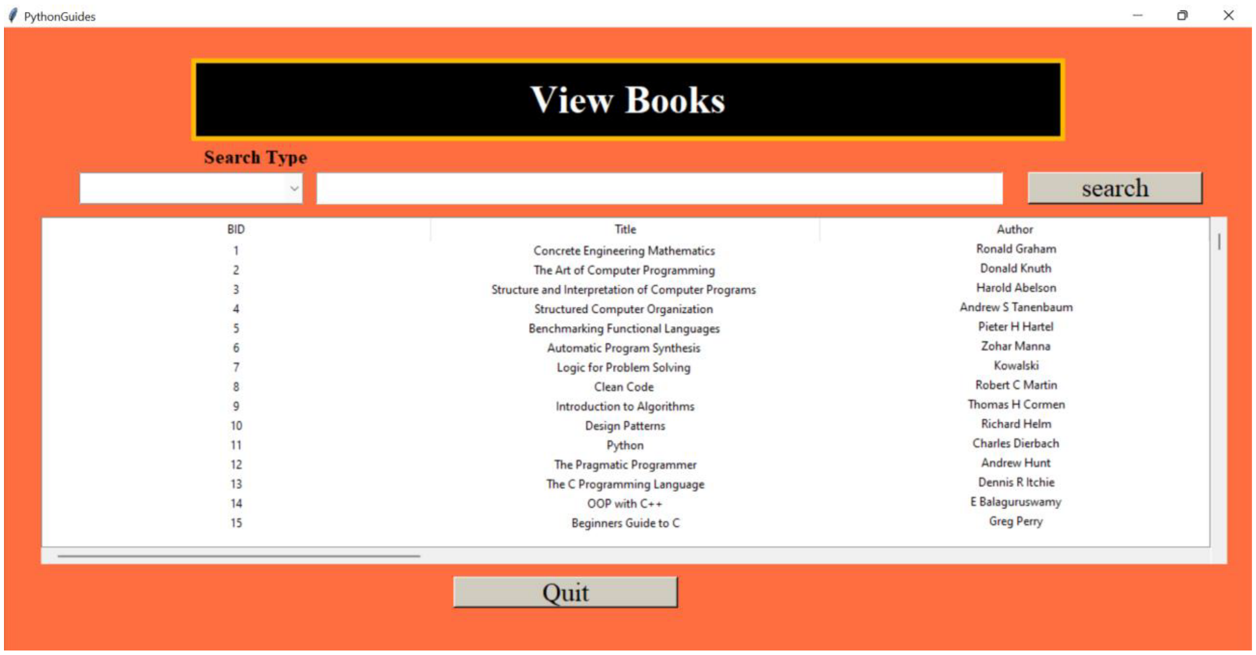


Sl no	srn	name	last name	branch	Semester	Phone Number	Email ID
1	690	Sowmesh	Sharma	CSE	1	8618988127	sowmeshshailesh1234@gr
2	691	Shailesh	Sharma	CSE	1	9998283746	shailesh123@gmail.com
3	213	Naruto	Uzumaki	Mechanical Engineering	6	8718711237	naruto@gmail.com
4	567	Suhas	P Shroff	CSE	1	5638990911	suhas@gmail.com
5	888	abcd	aa	EEE	4	9087654211	abcd@gmsil.com
6	890	Sowmesh	Shroff	Biotechnology	8	6678757123	shroff@gmail.com

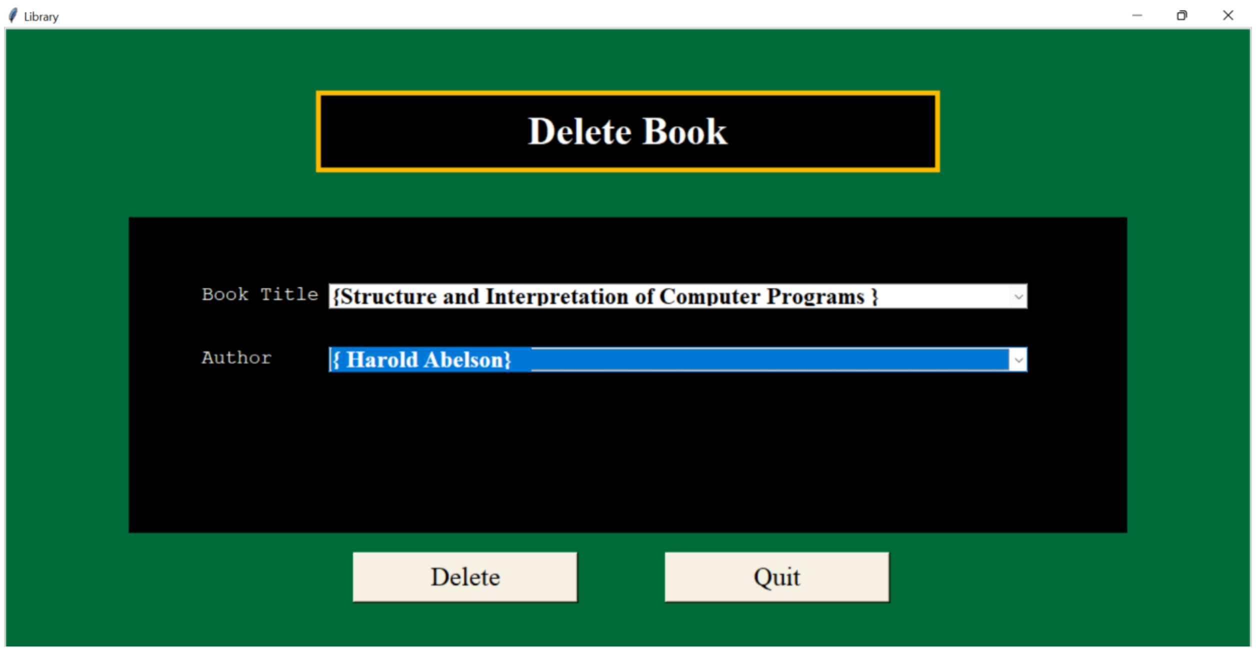



Manage Books
Manage Books
Add Books
Delete Books
View Books
Quit


Library
Add Books
Title :
Author :
Submit
Quit

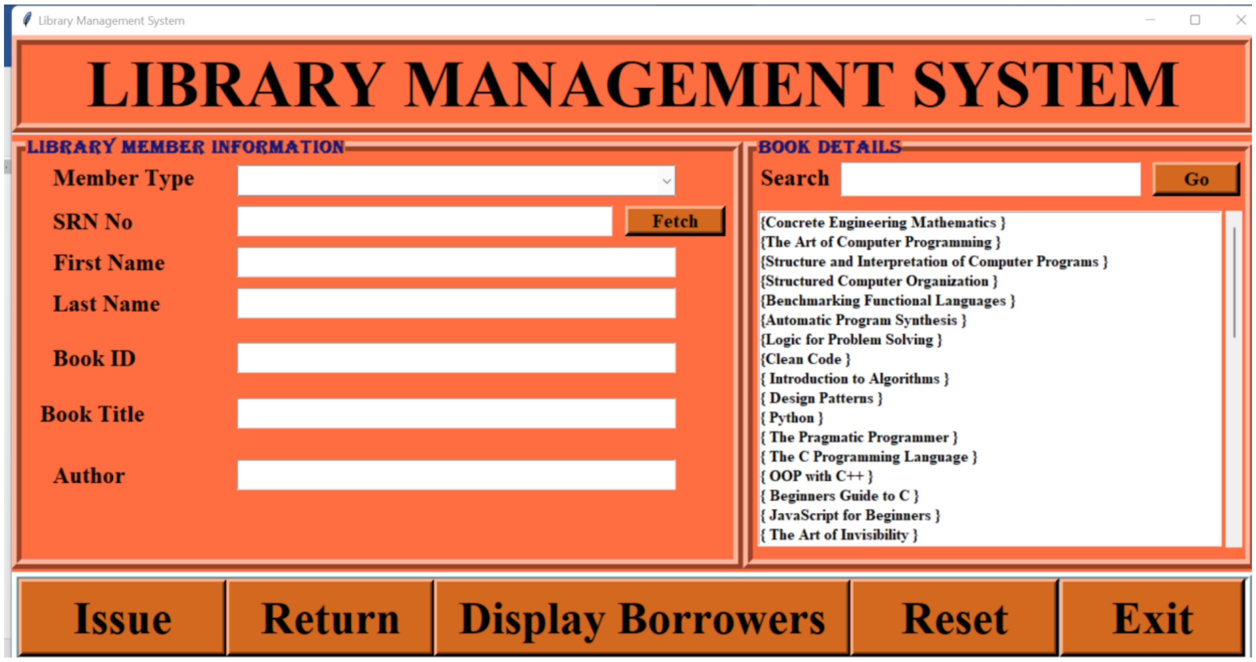


BID	Title	Author
1	Concrete Engineering Mathematics	Ronald Graham
2	The Art of Computer Programming	Donald Knuth
3	Structure and Interpretation of Computer Programs	Harold Abelson
4	Structured Computer Organization	Andrew S Tanenbaum
5	Benchmarking Functional Languages	Pieter H Hartel
6	Automatic Program Synthesis	Zohar Manna
7	Logic for Problem Solving	Kowalski
8	Clean Code	Robert C Martin
9	Introduction to Algorithms	Thomas H Cormen
10	Design Patterns	Richard Helm
11	Python	Charles Dierbach
12	The Pragmatic Programmer	Andrew Hunt
13	The C Programming Language	Dennis R Itchie
14	OOP with C++	E Balaguruswamy
15	Beginners Guide to C	Greg Perry



Book Title {Structure and Interpretation of Computer Programs }

Author { Harold Abelson}



Library Management System

LIBRARY MANAGEMENT SYSTEM

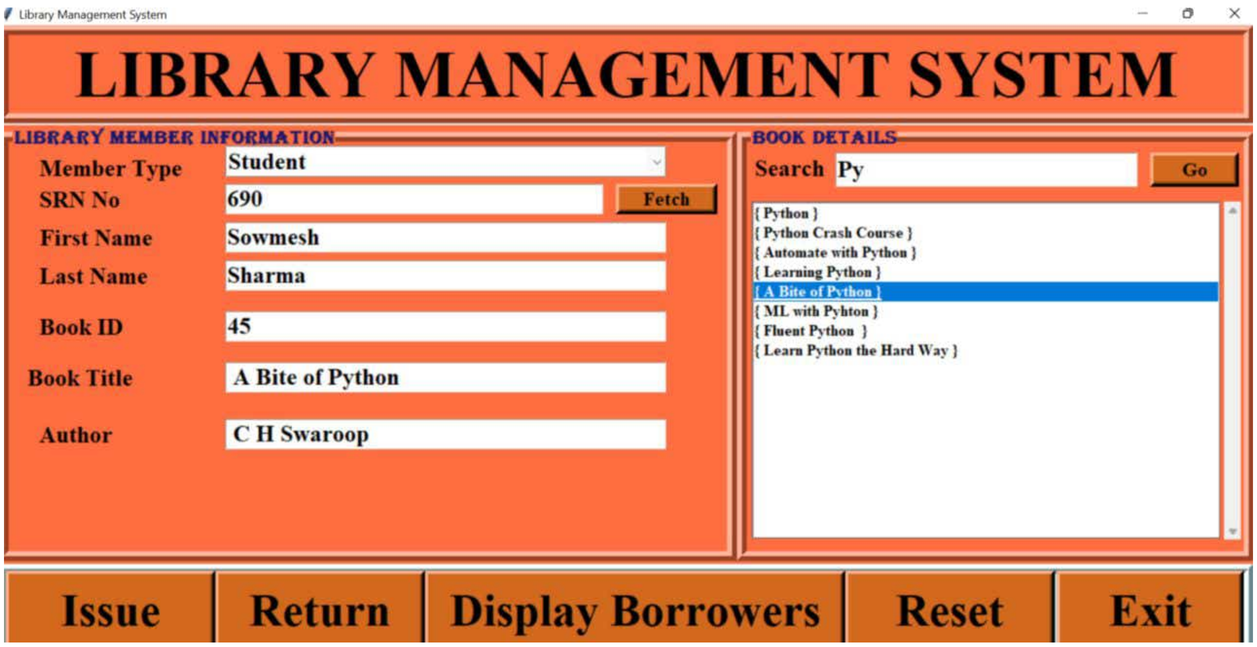
LIBRARY MEMBER INFORMATION

Member Type	
SRN No	
First Name	
Last Name	
Book ID	
Book Title	
Author	

BOOK DETAILS

Search

{Concrete Engineering Mathematics }
{The Art of Computer Programming }
{Structure and Interpretation of Computer Programs }
{Structured Computer Organization }
{Benchmarking Functional Languages }
{Automatic Program Synthesis }
{Logic for Problem Solving }
{Clean Code }
{ Introduction to Algorithms }
{ Design Patterns }
{ Python }
{ The Pragmatic Programmer }
{ The C Programming Language }
{ OOP with C++ }
{ Beginners Guide to C }
{ JavaScript for Beginners }
{ The Art of Invisibility }



Library Management System

LIBRARY MANAGEMENT SYSTEM

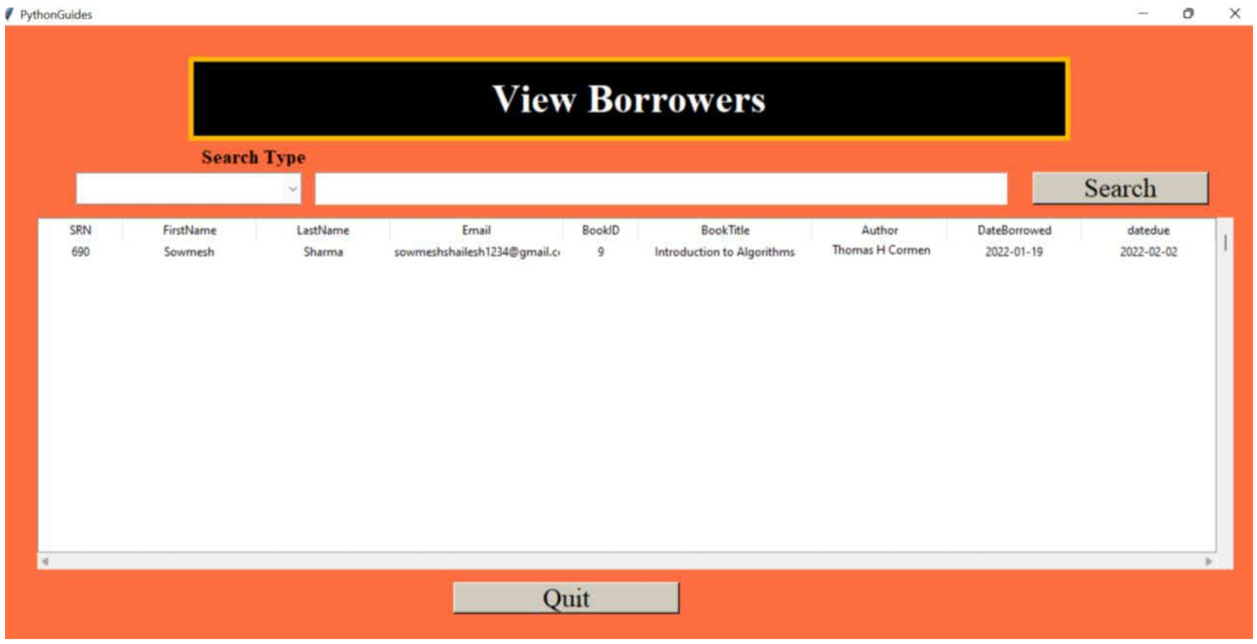
LIBRARY MEMBER INFORMATION

Member Type	Student
SRN No	690
First Name	Sowmesh
Last Name	Sharma
Book ID	45
Book Title	A Bite of Python
Author	C H Swaroop

BOOK DETAILS

Search Py

{ Python }
{ Python Crash Course }
{ Automate with Python }
{ Learning Python }
{ A Bite of Python }
{ ML with Pyhton }
{ Fluent Python }
{ Learn Python the Hard Way }



SRN	FirstName	LastName	Email	BookID	BookTitle	Author	DateBorrowed	datedue
690	Sowmesh	Sharma	sowmeshshailesh1234@gmail.c	9	Introduction to Algorithms	Thomas H Cormen	2022-01-19	2022-02-02

Conclusion:

A library management system is a web application designed to Digitize and Manage all the functions of a library end-to-end; This system can be used in public libraries; It can also be implemented in the library of schools, colleges, institutes, and organizations

Future Scope:

below are the list of Enhancements / Improvisations planned:

- Improve User Experience by making the web application with Responsive Web Page Design
- Enhance the Web Application to support Mobile Screens
- If possible, make it Android or iOS Enabled
- Enhance features to include:
- Better full-fledged Authentication & Authorization for user; and also encrypting / decrypting the Password (For Security)
- Add feature for Students to use this Application (now we have implemented for Librarian only)
- Further Intelligence (like automated reminders, trigger notifications to defaulters)
- Add Reporting features, with customizable Report Templates
- Add Alerts (like SMS) and Notifications (like email)
- Add / Enable feature for Subscriptions; notify students when new Books arrive
- Code provisions for e-Books (reading & management) within the Application
- Bring-in other Test Strategies for Non-Functional Requirements like Security Testing, Performance Testing

References:

- Data Flair (https://data-flair.training/)

1. C. J. Date, A. Kannan and S. Swamynathan, *An Introduction to Database Systems*, Pearson Education, Eighth Edition, 2009.
2. Abraham Silberschatz, Henry F. Korth and S. Sudarshan, *Database System Concepts*, McGraw-Hill Education (Asia), Fifth Edition, 2006.
3. Shio Kumar Singh, *Database Systems Concepts, Designs and Application*, Pearson Education, Second Edition, 2011.
4. Peter Rob and Carlos Coronel, *Database Systems Design, Implementation and Management*, Thomson Learning-Course Technology, Seventh Edition, 2007.
5. Patrick O'Neil and Elizabeth O'Neil, *Database Principles, Programming and Performance*, Harcourt Asia Pte. Ltd., First Edition, 2001.
6. Atul Kahate, *Introduction to Database Management Systems*, Pearson
7.