PROJECT REPORT ON

PYTHON

**ASANSOL ENGINEERING COLLEGE**

Prepared by:

**MD MINHAJ KHAN** (10800116089)

Branch: CSE

Session : 2016 - 2020

LIBRARY MANAGEMENT SYSTEM

IN

PYTHON

**Prepared for:**

CSE Department

Asansol Engineering College

Under the guidance of:

Mrs. Deepa Ashok Kumar

**Prepared by:**

Md Minhaj Khan

10800116089

CSE 4th Semester

**Table of Contents**

Acknowledgement ……..…………………………………………………………………………………………………………………….

Abstract ……………...…………………………………………………………………………………………………………………………….

Requirement Specification …………………………………….…………………………………………………………………………...

Database Design ..………………………………………………………………………………………...…………………………………….

Screenshots ………………………………………………………………………………………………………………………………………..

Code …………………..……………………………………………………………………………………………………………………………….

Future Scope of Improvements .………………………………………………………………………………………………………….

References …………………………………………………………………………………….………………………………………………….

**Acknowledgement**

I take this opportunity to express my profound gratitude and deep regards to my faculty for her exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by her time to time shall carry me a long way in the journey of life on which I am about to embark.

I am obliged to my colleagues for their valuable input and support provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

Md Minhaj Khan

CSE (10800116089)

**Abstract**

A management system that stores and manages the inventory of a library.

* The user can add books to the remove connected to the source code and likewise remove a particular book if inventory runs out.
* The user can also control the members database of the library and add or remove the former respectively as required.
* Also it can be used to issue books for specific time periods and display the books issued to each user.

The primary project goal is to deliver a user friendly experience for computerized management of a basic inventory that has multiple users access to. It is easy to use and understand.

**Requirement Specification**

Domain Description

We used python idle in our project and sqlite for database at the backend.

Problem Definition

The basic idea was to solve the problem of storing up the data in an organized way each specific to a certain user. It easily stores all the inputs of the inventory to each user issuing after it is processed.

Functional Requirements

With the most recent developments and changes in the business environment, data and information must be effectively gathered, managed and stored. Thus, an easier and user friendly manner of handling information is necessary.This user-friendly software will provide the facility of managing the inventory of a library or a similar system.

**Database Design**

**Table structure:**

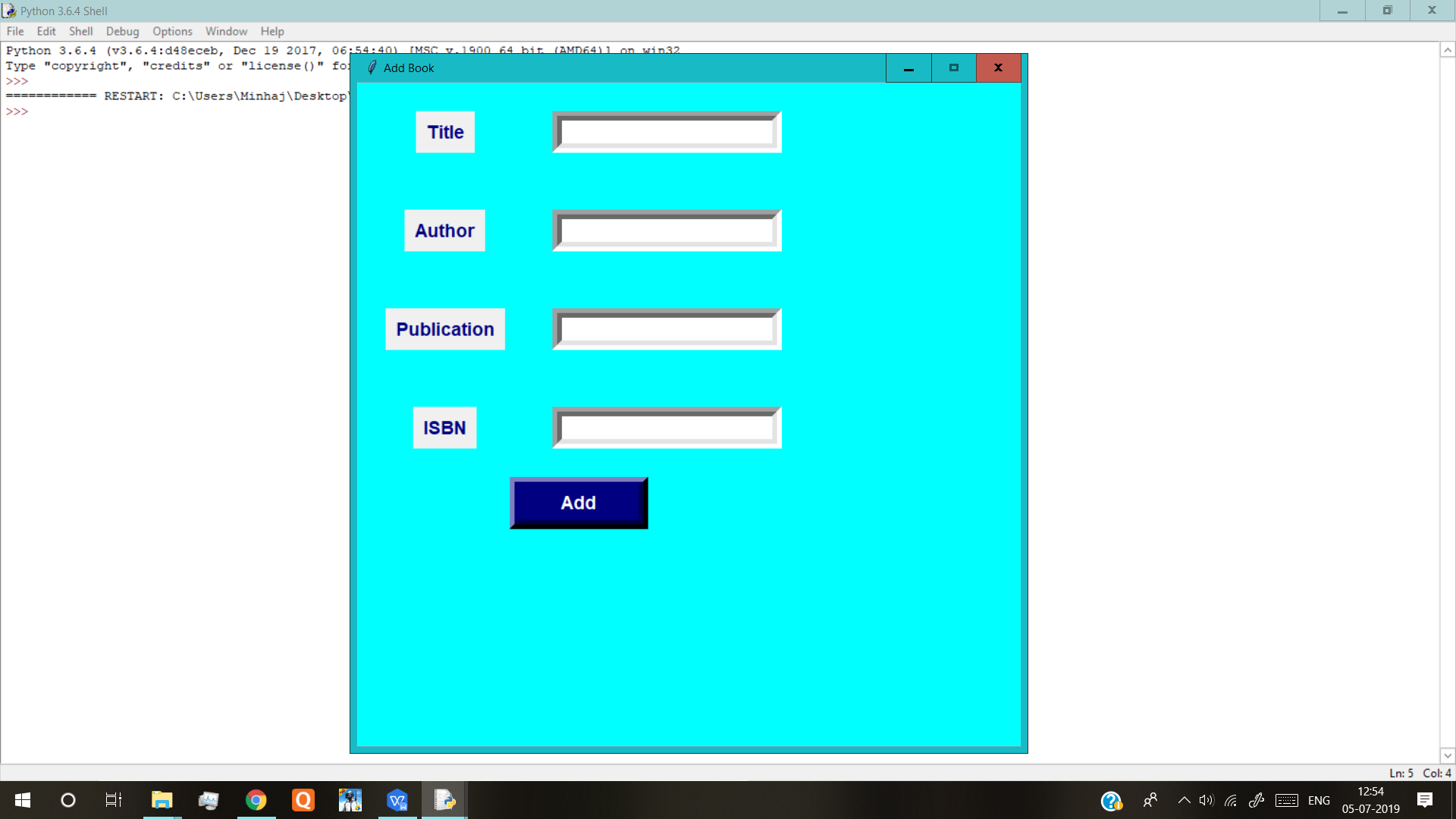
|  |  |
| --- | --- |
| **Field Name** | **Data Type** |
| **title** | **String** |
| **author** | **String** |
| **isbn** | **Integer** |
| **User\_id** | **String** |
| **name** | **String** |
| **alloc\_book\_isbn** | **Integer** |
| **issue\_date** | **Integer** |
| **return\_date** | **Integer** |

**SCREENSHOTS**

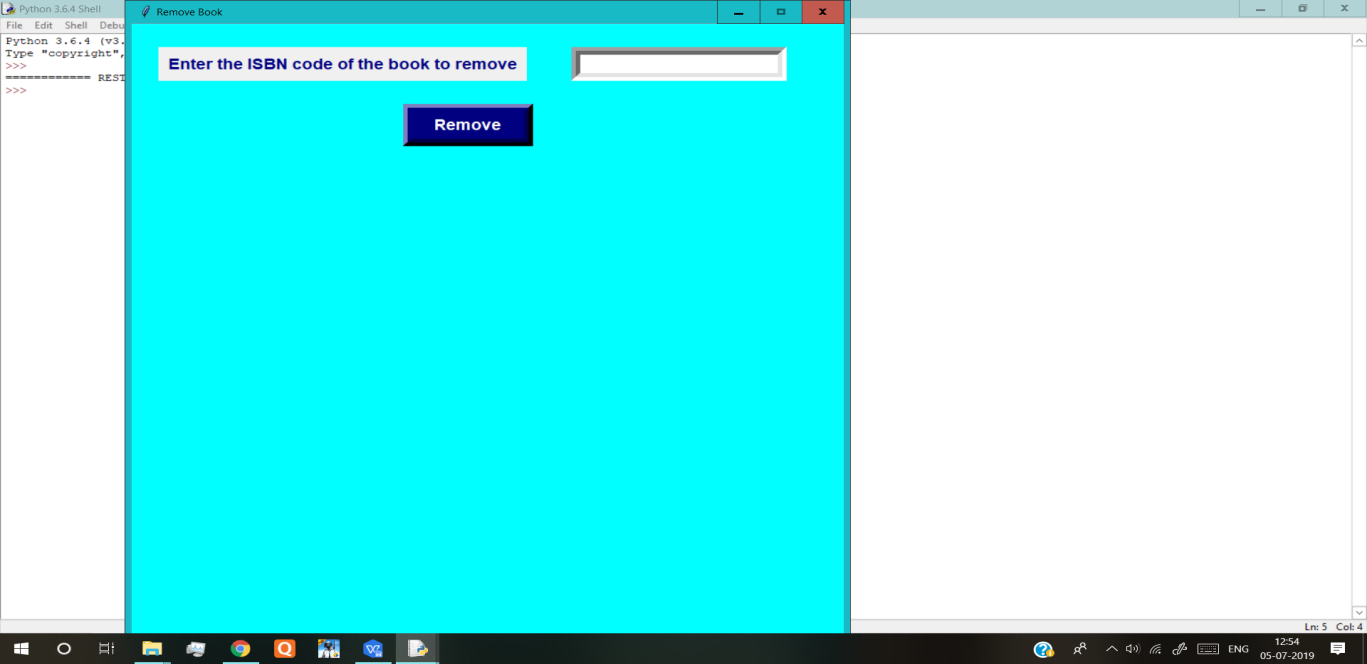
**Home Screen:**

****

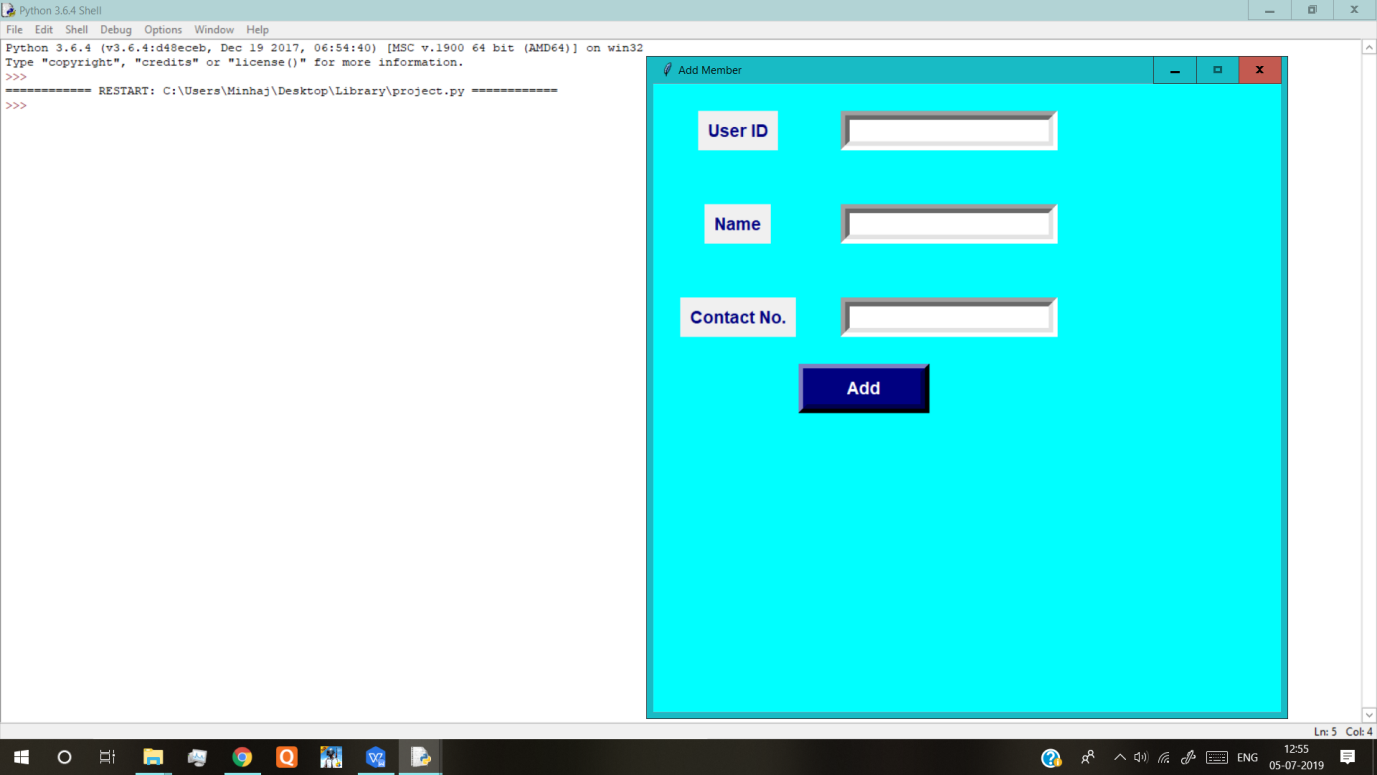
**Add Book:**

****

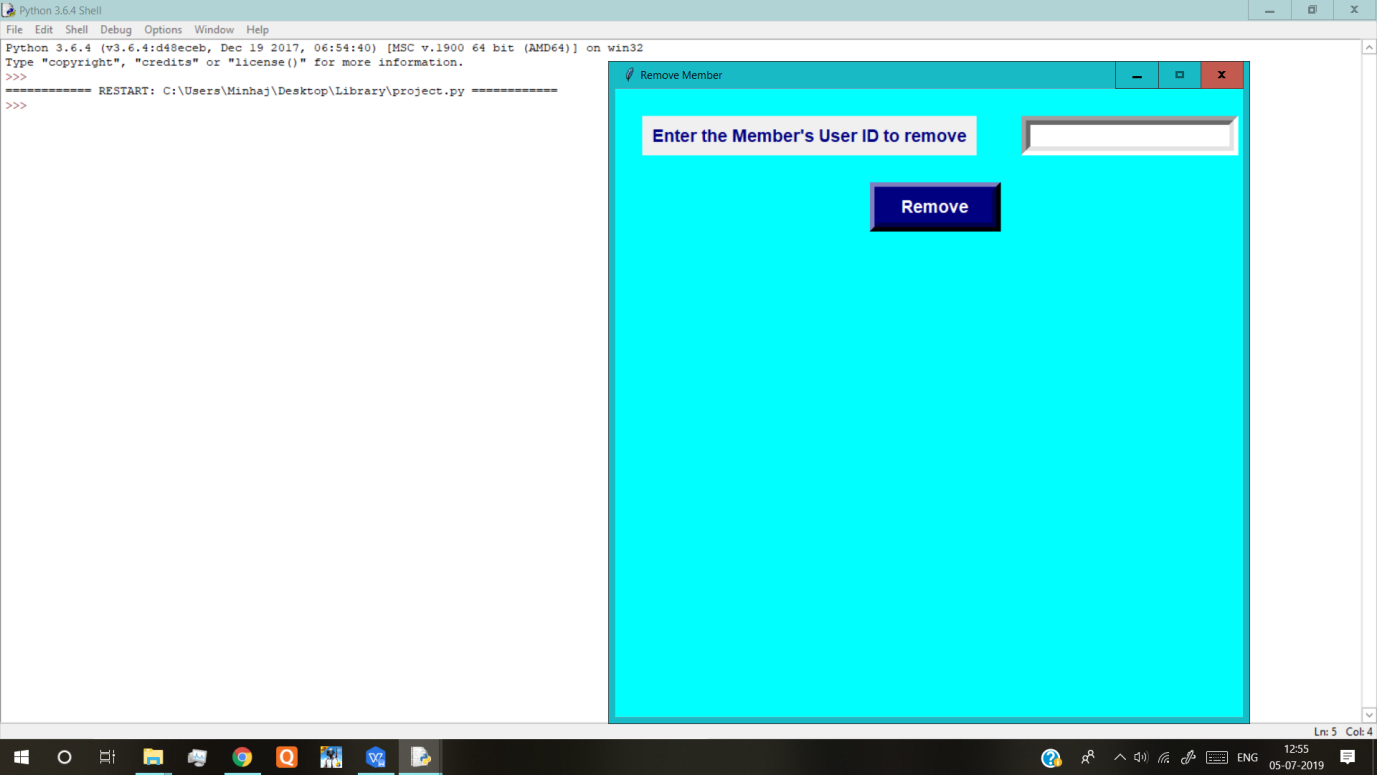
**Remove Book:**

****

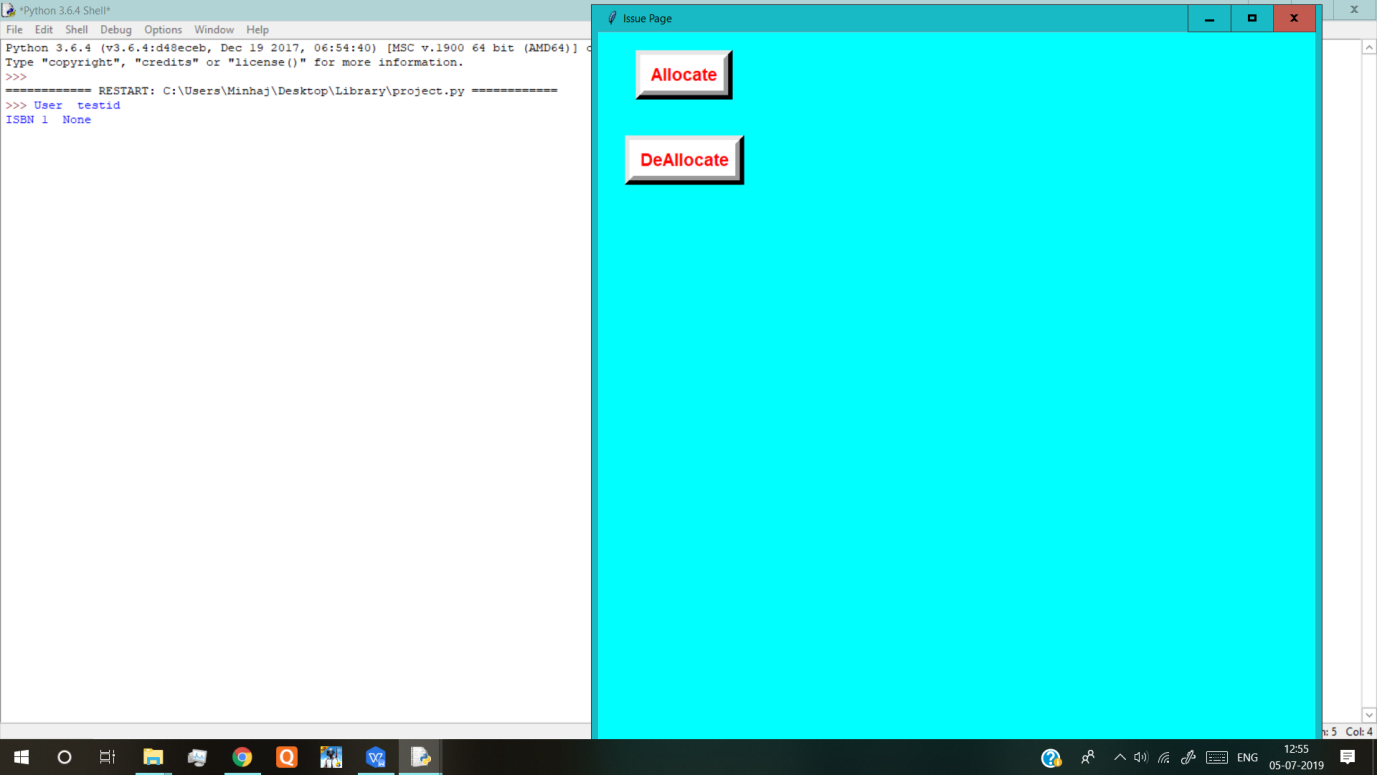
**Add Member:**

****

**Remove Member:**

****

**Issue Page:**

****

**Code:**

import sqlite3

from tkinter import \*

import tkinter

db=sqlite3.connect("Library.db")

home=tkinter.Tk()

home.title("Home")

home.geometry("700x700")

home.resizable(0,0)

home.configure(background='Green')

txt14=None

checkremisbn=""

def add\_book():

global addbook

global txt0

global txt1

global txt2

global txt3

global txt4

addbook=tkinter.Tk()

addbook.title("Add Book")

addbook.geometry("700x700")

addbook.resizable(0,0)

addbook.configure(background='Cyan')

#-----------------Title----------------------------

lb2=tkinter.Label(addbook,text="Title",font=('Arial',14,'bold'),bd="10",fg="navy")

lb2.grid(row=0,column=0,padx=30,pady=30)

txt0=tkinter.Entry(addbook,font=('Arial',14,'bold'),bd="10",fg="Navy")

txt0.grid(row=0,column=1,padx=20,pady=30)

#----------------Author----------------------------

lb3=tkinter.Label(addbook,text="Author",font=('Arial',14,'bold'),bd="10",fg="navy")

lb3.grid(row=1,column=0,padx=30,pady=30)

txt1=tkinter.Entry(addbook,font=('Arial',14,'bold'),bd="10",fg="navy")

txt1.grid(row=1,column=1,padx=20,pady=30)

#-------------------Publication----------------------

lb4=tkinter.Label(addbook,text="Publication",font=('Arial',14,'bold'),bd="10",fg="navy")

lb4.grid(row=2,column=0,padx=30,pady=30)

txt2=tkinter.Entry(addbook,font=('Arial',14,'bold'),bd="10",fg="navy")

txt2.grid(row=2,column=1,padx=20,pady=30)

#------------------ISBN----------------------------

lb5=tkinter.Label(addbook,text="ISBN",font=('Arial',14,'bold'),bd="10",fg="navy")

lb5.grid(row=3,column=0,padx=30,pady=30)

txt3=tkinter.Entry(addbook,font=('Arial',14,'bold'),bd="10",fg="navy")

txt3.grid(row=3,column=1,padx=20,pady=30)

#-------------Add Button---------------------

btn6=tkinter.Button(addbook,text="Add",font=('Arial',14,'bold'),width=10,bd="10",bg="navy",fg="white",command=add\_book\_to\_db)

btn6.grid(row=4,column=0,columnspan=2)

def add\_book\_to\_db():

global addbook

global txt0

global txt1

global txt2

global txt3

global txt4

global isbn

conn=sqlite3.connect("Library.db")

conn.execute('''create table if not exists book(title text not null,author text not null,publication text not null,isbn integer not null)''')

title=txt0.get()

author=txt1.get()

public=txt2.get()

isbn=txt3.get()

row=[(title,author,public,isbn)]

conn.executemany("insert into book(title,author,publication,isbn) values(?,?,?,?)",row)

conn.commit()

if conn.total\_changes==1:

lbladded=tkinter.Label(addbook,text="Book Successfully Added",font=('Arial',14,'bold'),bd="10",fg="Light blue")

lbladded.grid(row=5,column=1,columnspan=2)

rows=conn.execute("select \* from book")

for row in rows:

print("Title ",row[0])

print("Author ",row[1])

print("Publication ",row[2])

print("ISBN",row[3])

def add\_member():

global addmem

global txt4

global txt5

global txt6

addmem=tkinter.Tk()

addmem.title("Add Member")

addmem.geometry("700x700")

addmem.resizable(0,0)

addmem.configure(background='Cyan')

#-----------------User ID----------------------------

lb6=tkinter.Label(addmem,text="User ID",font=('Arial',14,'bold'),bd="10",fg="navy")

lb6.grid(row=0,column=0,padx=30,pady=30)

txt4=tkinter.Entry(addmem,font=('Arial',14,'bold'),bd="10",fg="Navy")

txt4.grid(row=0,column=1,padx=20,pady=30)

#----------------Name----------------------lb7=tkinter.Label(addmem,text="Name",font=('Arial',14,'bold'),bd="10",fg="navy")

lb7.grid(row=1,column=0,padx=30,pady=30)

txt5=tkinter.Entry(addmem,font=('Arial',14,'bold'),bd="10",fg="navy")

txt5.grid(row=1,column=1,padx=20,pady=30)

#-------------Contact No.-------------------------

lb8=tkinter.Label(addmem,text="Contact No.",font=('Arial',14,'bold'),bd="10",fg="navy")

lb8.grid(row=2,column=0,padx=30,pady=30)

txt6=tkinter.Entry(addmem,font=('Arial',14,'bold'),bd="10",fg="navy")

txt6.grid(row=2,column=1,padx=20,pady=30)

#-------------Add Button---------------------

btn7=tkinter.Button(addmem,text="Add",font=('Arial',14,'bold'),width=10,bd="10",bg="navy",fg="white",command=add\_member\_to\_db)

btn7.grid(row=3,column=0,columnspan=2)

def add\_member\_to\_db():

global addmem

global txt4

global txt5

global txt6

global user\_id

conn=sqlite3.connect("Library.db")

conn.execute('''create table if not exists member(user\_id text primary key,name text not null,contact integer not null)''')

conn.execute('''create table if not exists bookissue(user\_id text primary key,issue\_id text)''')

user\_id=txt4.get()

name=txt5.get()

cont=txt6.get()

row=[(user\_id,name,cont)]

row1=[(user\_id,None)]

conn.executemany("insert into member(user\_id,name,contact) values(?,?,?)",row)

conn.commit()

if conn.total\_changes==1:

lbladded1=tkinter.Label(addmem,text="Member Successfully Added",font=('Arial',14,'bold'),bd="10",fg="Light blue")

lbladded1.grid(row=4,column=1,columnspan=2)

conn.executemany("insert into bookissue(user\_id,issue\_id) values(?,?)",row1)

conn.commit()

def remove\_book():

global removebook

global checkremisbn

global txt14

conn=sqlite3.connect("Library.db")

removebook=tkinter.Tk()

removebook.title("Remove Book")

removebook.geometry("800x800")

removebook.resizable(0,0)

removebook.configure(background='Cyan')

lb6=tkinter.Label(removebook,text="Enter the ISBN code of the book to remove",font=('Arial',14,'bold'),bd="10",fg="navy")

lb6.grid(row=0,column=0,padx=30,pady=30)

txt14=tkinter.Entry(removebook,font=('Arial',14,'bold'),bd="10",fg="Navy")

txt14.grid(row=0,column=1,padx=20,pady=30)

btn8=tkinter.Button(removebook,text="Remove",font=('Arial',14,'bold'),width=10,bd="10",bg="navy",fg="white",command=remove\_book\_from\_db)

btn8.grid(row=3,column=0,columnspan=2)

def remove\_book\_from\_db():

conn=sqlite3.connect("Library.db")

global removebook

global checkremisbn

global txt14

count=0

checkremisbn=txt14.get()

print(checkremisbn)

bookisbn=conn.execute("select \* from book where isbn='"+str(checkremisbn)+"'")

for row in bookisbn:

count+=1

if count==0:

lb7=tkinter.Label(removebook,text="Book ISBN doesn't exist",font=('Arial',10),bd=10,fg="Navy")

lb7.grid(row=1,column=0,padx=30,pady=20,columnspan=2)

else:

conn.execute("delete from book where isbn='"+str(checkremisbn)+"'")

lb8=tkinter.Label(removebook,text="Book Removed",font=('Arial',10,'bold'),bd=10,fg="Navy")

lb8.grid(row=1,column=0,padx=30,pady=20,columnspan=2)

conn.commit()

def remove\_member():

conn=sqlite3.connect("Library.db")

global removemem

global userid

global rem\_member\_id

removemem=tkinter.Tk()

removemem.title("Remove Member")

removemem.geometry("700x700")

removemem.resizable(0,0)

removemem.configure(background='Cyan')

lb6=tkinter.Label(removemem,text="Enter the Member's User ID to remove",font=('Arial',14,'bold'),bd="10",fg="navy")

lb6.grid(row=0,column=0,padx=30,pady=30)

txt4=tkinter.Entry(removemem,font=('Arial',14,'bold'),bd="10",fg="Navy")

txt4.grid(row=0,column=1,padx=20,pady=30)

rem\_member\_id=txt4.get()

btn8=tkinter.Button(removemem,text="Remove",font=('Arial',14,'bold'),width=10,bd="10",bg="navy",fg="white",command=remove\_mem\_from\_db)

btn8.grid(row=3,column=0,columnspan=2)

def remove\_mem\_from\_db():

conn=sqlite3.connect("Library.db")

global removemem

global userid

global txt4

global rem\_member\_id

userid=conn.execute("select \* from member where user\_id='"+str(rem\_member\_id)+"'")

if userid==None:

lb7=tkinter.Label(removemem,text="Member doesn't exist",font=('Arial',10),bd=10,fg="Navy")

lb7.grid(row=1,column=0,padx=30,pady=20,columnspan=2)

else:

conn.execute("delete from member where user\_id='"+str(rem\_member\_id)+"'")

lb8=tkinter.Label(removemem,text="Book Removed",font=('Arial',10,'bold'),bd=10,fg="Navy")

lb8.grid(row=1,column=0,padx=30,pady=20,columnspan=2)

def issueing():

global issue

issue=tkinter.Tk()

issue.title("Issue Page")

issue.geometry("800x800")

issue.configure(background='Cyan')

btn8=tkinter.Button(issue,text="Allocate",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=allocate)

btn8.grid(row=1,column=1,padx=30,pady=20,columnspan=2)

btn9=tkinter.Button(issue,text="DeAllocate",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=deallocate)

btn9.grid(row=2,column=1,padx=30,pady=20,columnspan=2)

def allocate():

global issue

global alloc\_member\_id

global alloc\_book\_isbn

global issue\_date

global return\_date

global txt7

global txt8

global txt9

global txt10

conn=sqlite3.connect("Library.db")

lb9=tkinter.Label(issue,text="Enter Book ISBN to Allocate",font=('Arial',10),bd=10,fg="Navy")

lb9.grid(row=4,column=0,padx=30,pady=20,columnspan=1)

txt7=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt7.grid(row=4, column=1,padx=30,pady=20,columnspan=2)

lb10=tkinter.Label(issue,text="Enter Member Receiving the book",font=('Arial',10),bd=10,fg="Navy")

lb10.grid(row=5,column=0,padx=30,pady=20,columnspan=1)

txt8=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt8.grid(row=5, column=1,padx=30,pady=20,columnspan=2)

lb11=tkinter.Label(issue,text="Enter Issue Date in yymmdd format",font=('Arial',10),bd=10,fg="Navy")

lb11.grid(row=6,column=0,padx=30,pady=20,columnspan=1)

txt9=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt9.grid(row=6, column=1,padx=30,pady=20,columnspan=2)

lb12=tkinter.Label(issue,text="Enter Return Date in yymmdd format",font=('Arial',10),bd=10,fg="Navy")

lb12.grid(row=7,column=0,padx=30,pady=20,columnspan=1)

txt10=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt10.grid(row=7, column=1,padx=30,pady=20,columnspan=2)

btn10=tkinter.Button(issue,text="Confirm",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=allocate\_to\_db)

btn10.grid(row=8,column=1,padx=30,pady=20,columnspan=2)

def allocate\_to\_db():

global issue

global alloc\_member\_id

global alloc\_book\_isbn

global issue\_date

global return\_date

global txt7

global txt8

global txt9

global txt10

alloc\_book\_isbn=txt7.get()

alloc\_member\_id=txt8.get()

issue\_date=txt9.get()

return\_date=txt10.get()

count1=0

count2=0

count3=0

count4=0

count5=0

conn=sqlite3.connect("Library.db")

checkisbn=conn.execute("select \* from book where isbn='"+(alloc\_book\_isbn)+"'")

for row in checkisbn:

count1+=1

if count1==0:

lb13=tkinter.Label(issue,text="Book ISBN doesn't exist",font=('Arial',10),bd=10,fg="Navy")

lb13.grid(row=8,column=0,padx=30,pady=20,columnspan=1)

else:

checkmem=conn.execute("select \* from member where user\_id='"+str(alloc\_member\_id)+"'")

for row in checkmem:

count2+=1

if count2==0:

lb14=tkinter.Label(issue,text="Member ID doesn't exist",font=('Arial',10),bd=10,fg="Navy")

lb14.grid(row=9,column=0,padx=30,pady=20,columnspan=1)

else:

check1=conn.execute("select issue\_id from bookissue where user\_id='"+str(alloc\_member\_id)+"'")

for row in check1:

count3+=1

if count3!=0:

issue\_id=issue\_date+return\_date

rows1=[(issue\_id)]

conn.execute("update bookissue set issue\_id ='"+str(issue\_id)+"' where user\_id='"+str(alloc\_member\_id)+"'")

conn.commit()

lb15=tkinter.Label(issue,text="Book Issued Sucessfully",font=('Arial',14,'bold'),bd="10",fg="Red")

lb15.grid(row=10,column=0,padx=30,pady=20,columnspan=2)

else:

lb15=tkinter.Label(issue,text="User Cannot Issue more Books",font=('Arial',14,'bold'),bd="10",fg="Red")

lb15.grid(row=10,column=0,padx=30,pady=20,columnspan=2)

def deallocate():

global issue

global txt11

global txt12

conn=sqlite3.connect("Library.db")

lbl9=tkinter.Label(issue,text="Enter the Issue ID Provided",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White")

lbl9.grid(row=4,column=0,padx=30,pady=20,columnspan=2)

txt11=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt11.grid(row=4, column=2,padx=80,pady=20,columnspan=2)

lb10=tkinter.Label(issue,text="Enter Days Overdue",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White")

lb10.grid(row=5,column=0,padx=30,pady=20,columnspan=2)

txt12=tkinter.Entry(issue,font=('Arial',10),bd="10",fg="Navy")

txt12.grid(row=5, column=2,padx=80,pady=20,columnspan=2)

btn10=tkinter.Button(issue,text="Confirm",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=deallocate\_to\_db)

btn10.grid(row=6,column=1,padx=30,pady=20,columnspan=2)

def deallocate\_to\_db():

global txt11

global txt12

count6=0

conn=sqlite3.connect("Library.db")

dealloc=txt11.get()

due=txt12.get()

fine=5\*int(due)

checkissue\_id=conn.execute("select \* from bookissue where issue\_id='"+(dealloc)+"'")

for row in checkissue\_id:

count6+=1

if count6!=0:

conn.execute("update bookissue set issue\_id='"+str(None)+"'")

conn.commit()

lb10=tkinter.Label(issue,text="Book DeAllocated",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White")

lb10.grid(row=7,column=1,padx=30,pady=20,columnspan=2)

lb11=tkinter.Label(issue,text="Fine to Be Paid "+(fine)+"'",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White")

lb11.grid(row=8,column=1,padx=30,pady=20,columnspan=2)

def disp():

conn=sqlite3.connect("Library.db")

rows=conn.execute("select \* from bookissue")

for row in rows:

print("User ",row[0])

print("ISBN 1 ",row[1])

lb0=tkinter.Label(home,text="Welcome",font=('Arial',35,'bold'),bd="10",fg="Red",bg="light blue")

lb0.grid(row=0,column=0,padx=80,pady=30,columnspan=4)

lb1=tkinter.Label(home,text="Library Management",font=('Arial',30,'bold'),bd="10",fg="Red",bg="light blue")

lb1.grid(row=1,column=0,padx=80,pady=30,columnspan=4)

btn1=tkinter.Button(home,text="Add a Book",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=add\_book)

btn1.grid(row=2,column=0,padx=80,pady=30,columnspan=2)

btn2=tkinter.Button(home,text="Remove Books",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=remove\_book)

btn2.grid(row=2,column=2,padx=50,pady=20,columnspan=2)

btn3=tkinter.Button(home,text="Add a Member",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=add\_member)

btn3.grid(row=3,column=0,padx=80,pady=30,columnspan=2)

btn4=tkinter.Button(home,text="Remove Members",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=remove\_member)

btn4.grid(row=3,column=2,padx=50,pady=20,columnspan=2)

btn5=tkinter.Button(home,text="Issueing",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=issueing)

btn5.grid(row=4,column=1,padx=80,pady=30,columnspan=2)

btntst=tkinter.Button(home,text="Display",font=('Arial',14,'bold'),bd="10",fg="Red",bg="White",command=disp)

btntst.grid(row=5,column=1,padx=80,pady=30,columnspan=2)

#=================END=======================

**Future Scope of Improvements:**

* The project is a humble attempt to create a computerized user friendly Library management system that would ease the method of storing and organizing a library.
* Presently, the number of books and users that can be added is limited. It can be further enhanced to create a wider catalogue of books for which the inventory management can be implemented.
* This can greatly save time and effort involved in the otherwise manual process of organizing.

**References**

* https://stackoverflow.com
* https://docs.python.org/