COMPUTER PROJECT

TOPIC : LIBRARY MANAGEMENT SYSTEM

SUBMITTED BY : NOEL M ABY

BHAVAN'S VIDYA MANDIR CHITHALI

SENIOR SECONDARY SCHOOL AFFLIATED TO CBSE, DELHI. NO. 930445

(MANAGED BY BHARTIYA VIDYA BHAVAN, PALAKKAD KENDRA)

LET NOBLE THOUGHTS COME TO US FROM EVERY SIDE

CERTIFICATE

NAME OF THE STUDENT : _____

CLASS : _____

SUBJECT : _____

ACADEMIC YEAR : _____

Certified that this is a bonafide record of Project /Practical work of

Master/Kumari _____ *with*

Register Number _____ *in the Academic Year* _____.

Principal : _____ *Teacher in charge :*_____

*External Examiner :*_____ *Internal Examiner:*_____

ACKNOWLEDGMENT

I would like to express my special thanks of gratitude to my teacher Mrs.Sujatha as well as our principal Dr. Subadra Muralidharan who gave me the golden opportunity to do this wonderful project on the topic Library management system, which also helped me in doing a lot of research and I came to know about so many new things I am really thankful to them.

Secondly I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

INDEX

SNO	CONTENT	PAGE NO
1	INTRODUCTION	5
2	PROGRAM CODE	8
3	STRUCTURE OF SQL TABLES	38
4	OUTPUT SCREEN	39
5	BIBLIOGRAPHY	43

INTRODUCTION

Learning to read is about listening and understanding as well as working out what is printed on the page. Through hearing stories, children are exposed to a wide range of words. This helps them build their own vocabulary and improve their understanding when they listen, which is vital as they start to read.

Libraries are important cornerstones of a healthy community. Libraries give people the opportunity to find jobs, explore medical research, experience new ideas, get lost in wonderful stories, while at the same time providing a sense of place for gathering.

School libraries provide another space for children to learn. They can offer a quiet area for students to study, and encourage students to read.. School

libraries provide a safe haven for all students to think, create, share, and grow.

Library Management System:

By using library management system, the operation of borrowing and managing inventories is paperless.This system will store all the books and members information that consist book-id,book name,author name and genres to the system database.
Only admin have the access to view or edit data from the system databases.The admin will handle the admin functions such as add new stock and delete stock.User need to enter correct id and password before they can access to this functions such as delete stock and delete reports.

Objectives of Library Management System:

- **It will monitor and manage all library operations efficiently.**

- **Give an opportunity to librarians to reduce mistakes that always happen during manual method.**

- **To store properly the library items in order to maintain their security.**

- **To enter and preserve details of the various issues and keep a track on their returns**

PROGRAM CODE

```
import time
from tkinter import *
from tkinter import messagebox
from tkinter import ttk
from PIL import ImageTk, Image
import pyttsx3


#voice
engine=pyttsx3.init()
voices=engine.getProperty('voices')
engine.setProperty('voice',voices[0].id)
engine.setProperty('rate',165)
engine.runAndWait()
def speak(str):
    engine.say(str)
    engine.runAndWait()

#Erase report details
def erasereport():
    usname = e11.get()
    passw = e22.get()

    if (usname == " " and passw == " "):
        messagebox.showerror("ERROR", "ALL
FIELDS REQUIRED")
    elif (usname == "admin" and passw ==
"12345"):
        import mysql.connector as con
        mydb = con.connect(host="localhost",
user="root", password="1234",
```

```
database="library")
        mycur = mydb.cursor()
        mycur.execute("delete from issuebook")
        mycur.execute("delete from returnbook")
        mydb.commit()
        speak("ALL REPORT DETAILS ERASED")
        messagebox.showinfo("NOTE", "ALL REPORT
DETAILS ERASED !!!",parent=erd)

    else:
        messagebox.showerror("ERROR", "INCORRECT
USERNAME AND PASSWORD \n RE-ENTER USERNAME AND
PASSWORD",parent=erd)


def erasertdetails():
    global e11, e22,erd

    erd = Toplevel()
    erd.title("ADMIN CORNER")
    erd.geometry("300x200")
    erd.resizable(False, False)
    erd.configure(bg="white")
    Label(erd, text="USERNAME").place(x=10,
y=10)
    Label(erd, text="PASSWORD").place(x=10,
y=40)

    e11 = Entry(erd)
    e11.place(x=140, y=10)

    e22 = Entry(erd)
    e22.place(x=140, y=40)
    e22.config(show="*")
```

```
    erabtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\erasebtn.png"))

ease=Button(erd,command=erasereport,image=erabtn
)
    ease.image=erabtn
    ease.place(x=10, y=100)

    erd.mainloop()


#Erase book details
def erasebook():
    usname=e1.get()
    passw=e2.get()

    if(usname==" " and passw==" "):
        messagebox.showerror("ERROR","ALL FIELDS
REQUIRED")
    elif (usname=="admin" and passw=="12345"):
        import mysql.connector as con
        mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
        mycur = mydb.cursor()
        mycur.execute("delete from book")
        mydb.commit()
        speak("ALL BOOK DETAILS ERASED")
        messagebox.showinfo("NOTE","ALL BOOK
DETAILS ERASED !!!",parent=ekd)

    else:
        messagebox.showerror("ERROR","INCORRECT
USERNAME AND PASSWORD \n RE-ENTER USERNAME AND
```

```
PASSWORD",parent=ekd)


#Erase book details
def erasebkdetail():
    global e1,e2,ekd

    ekd = Toplevel()
    ekd.title("ADMIN CORNER")
    ekd.geometry("300x200")
    ekd.resizable(False, False)
    ekd.configure(bg="white")
    Label(ekd,text="USERNAME").place(x=10,y=10)
    Label(ekd,text="PASSWORD").place(x=10,y=40)

    e1=Entry(ekd)
    e1.place(x=140,y=10)

    e2=Entry(ekd)
    e2.place(x=140,y=40)
    e2.config(show="*")

    erabtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\erasebtn.png"))
    ease = Button(ekd, command=erasebook,
image=erabtn)
    ease.image = erabtn
    ease.place(x=10, y=100)
    ekd.mainloop()
```

```
#Delete book
def deletebook():
    bod = int(kide.get())
    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    mycur = mydb.cursor()
    mycur.execute("select * from book where
BOOK_ID='{}'".format(bod))
    dta = mycur.fetchall()
    if dta == []:
        messagebox.showerror("error", "BOOK ID
NOT EXIST \n ENTER VALID BOOK ID", parent=adm)
    else:
        mycur.execute("delete from book where
BOOK_ID='{}'".format(bod))
        mydb.commit()
        speak("BOOK DELETED SUCCESSFULLY")
        messagebox.showinfo("NOTE", "BOOK DELETE
SUCCESSFULLY", parent=adm)



#Report Return
def reportreturn():
    bod = int(kide.get())
    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    mycur = mydb.cursor()
    mycur.execute("select * from book where
```

```
BOOK_ID='{}'".format(bod))
    dta = mycur.fetchall()
    if dta == []:
        messagebox.showerror("error", "BOOK ID
NOT EXIST \n ENTER VALID BOOK ID", parent=adm)
    else:
        mycur.execute("select * from returnbook
where BOOK_ID='{}'".format(bod))
        shwdata = mycur.fetchall()
        if shwdata == []:
            messagebox.showinfo("NOTE", "NO
REPORTS FOUND", parent=adm)
        else:
            box2 = ttk.Treeview(adm)

            box2['columns'] = ("BOOK_ID",
"BOOK_NAME", "STUDENT_NAME", "DATE", "CLASS")

            box2.column("#0", width=0,
stretch=NO)
            box2.column("BOOK_ID", width=150)
            box2.column("BOOK_NAME", width=150)
            box2.column("STUDENT_NAME",
width=150)
            box2.column("DATE", width=150)
            box2.column("CLASS", width=150)

            box2.heading("#0", text=" ",
anchor=W)
            box2.heading("BOOK_ID",
text="BOOK_ID", anchor=W)
            box2.heading("BOOK_NAME",
text="BOOK_NAME", anchor=W)
            box2.heading("STUDENT_NAME",
text="STUDENT_NAME", anchor=W)
```

```
            box2.heading("DATE", text="DATE",
anchor=W)
            box2.heading("CLASS", text="CLASS",
anchor=W)

            count = 0
            for t in shwdata:
                box2.insert('', 0, values=(t[0],
t[1], t[2], t[3], t[4]))
                count = count + 1

            box2.place(x=250, y=450)


#Report issue
def reportissue():
    bod=int(kide.get())
    import mysql.connector as con

mydb=con.connect(host="localhost",user="root",pa
ssword="1234",database="library")
    mycur=mydb.cursor()
    mycur.execute("select * from book where
BOOK_ID='{}'".format(bod))
    dta = mycur.fetchall()
    if dta == []:
        messagebox.showerror("error", "BOOK ID
NOT EXIST \n ENTER VALID BOOK ID", parent=adm)
    else:
        mycur.execute("select * from issuebook
where BOOK_ID='{}'".format(bod))
        shwdata=mycur.fetchall()
        if shwdata==[]:
            messagebox.showinfo("NOTE", "NO
REPORTS FOUND", parent=adm)
```

```
        else:
            box1 = ttk.Treeview(adm)

            box1['columns'] = ("BOOK_ID",
"BOOK_NAME", "STUDENT_NAME", "DATE", "CLASS")

            box1.column("#0", width=0,
stretch=NO)
            box1.column("BOOK_ID", width=150)
            box1.column("BOOK_NAME", width=150)
            box1.column("STUDENT_NAME",
width=150)
            box1.column("DATE", width=150)
            box1.column("CLASS", width=150)

            box1.heading("#0", text=" ",
anchor=W)
            box1.heading("BOOK_ID",
text="BOOK_ID", anchor=W)
            box1.heading("BOOK_NAME",
text="BOOK_NAME", anchor=W)
            box1.heading("STUDENT_NAME",
text="STUDENT_NAME", anchor=W)
            box1.heading("DATE", text="DATE",
anchor=W)
            box1.heading("CLASS", text="CLASS",
anchor=W)

            count = 0
            for t in shwdata:
                box1.insert('', 0, values=(t[0],
t[1], t[2], t[3], t[4]))
                count = count + 1

            box1.place(x=250, y=450)
```

```
def showadm():
    sec = clicked.get()
    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    mycur = mydb.cursor()
    mycr = mydb.cursor()
    mycur.execute("select * from book where
BOOK_GENRES ='{}' ".format(sec))
    data = mycur.fetchall()

    box = ttk.Treeview(adm)

    box['columns'] = ("BOOK_ID", "BOOK_NAME",
"AUTHOR", "BOOK_GENRES", "STATUS")

    box.column("#0", width=0, stretch=NO)
    box.column("BOOK_ID", width=150)
    box.column("BOOK_NAME", width=150)
    box.column("AUTHOR", width=150)
    box.column("BOOK_GENRES", width=150)
    box.column("STATUS", width=150)

    box.heading("#0", text=" ", anchor=W)
    box.heading("BOOK_ID", text="BOOK_ID",
anchor=W)
    box.heading("BOOK_NAME", text="BOOK_NAME",
anchor=W)
    box.heading("AUTHOR", text="AUTHOR",
anchor=W)
    box.heading("BOOK_GENRES",
text="BOOK_GENRES", anchor=W)
```

```
    box.heading("STATUS", text="STATUS",
anchor=W)

    count = 0
    for t in data:
        box.insert('', 0, values=(t[0], t[1],
t[2], t[3], t[4]))
        count = count + 1

    box.place(x=250, y=450)


#Add book
def ak():
    bi=int(bookide.get())
    bn=bookname.get()
    ar=authore.get()
    gr=genrese.get()
    bs="available"

    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    mycur = mydb.cursor()

    while True:
        flag = False
        mycur.execute("select * from book;")
        for x in mycur:
            if x[0] == bi:
                flag = True
        if flag == False:
            mycur.execute("insert into book
values('{}','{}','{}','{}','{}')".format(bi, bn,
```

```
ar, gr,bs))
            mydb.commit()
            speak("Book successfully added")
            messagebox.showinfo("NOTE","BOOK
SUCCESSFULLY ADDED",parent=adm)
            break
        else:
            messagebox.showerror("NOTE","BOOK-ID
EXIST \nENTER NEW BOOK-ID",parent=adm)
            break



#admin
def admin():
    global
clicked,click,adm,bookide,bookname,authore,genre
se,kide
    lis = []

    adm = Toplevel()
    adm.title("ADMIN CORNER")
    adm.geometry("1200x700")
    adm.resizable(False, False)

    adm.configure(bg="grey")

    adm.bg =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\admwall.png"))
    adm.bg_image = Label(adm,
image=adm.bg).place(x=0, y=0, relwidth=1,
relheight=1)
```

```
    head = Label(adm, text="ADMIN CORNER",
font=("jokerman", 25, "bold"),
fg="orange2",bg="black")
    head.place(x=790, y=10)

    #seperating frame
    seframe = Frame(adm, bg="orange2")
    seframe.place(x=650, y=40, height=360,
width=10)

    #Other frame
    otframe = Frame(adm, bg="white")
    otframe.place(x=686, y=180, height=217,
width=494)

    reportisubtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\isptbtn.png"))
    reisu = Button(otframe, image=reportisubtn,
bd=0, bg="white",command=reportissue)
    reisu.image = reportisubtn
    reisu.place(x=0, y=10)

    reportretbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\reptbtn.png"))
    reret = Button(otframe, image=reportretbtn,
bd=0, bg="white",command=reportreturn)
    reret.image = reportretbtn
    reret.place(x=225, y=10)

    delbtbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\dbbtn.png"))
    delebtn = Button(otframe, image=delbtbtn,
```

```
bd=0, bg="white",command=deletebook)
    delebtn.image = delbtbtn
    delebtn.place(x=0, y=80)

    ebbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\ebdbtn.png"))
    ebokbtn = Button(otframe, image=ebbtn, bd=0,
bg="white",command=erasebkdetail)
    ebokbtn.image = ebbtn
    ebokbtn.place(x=200, y=80)

    eribtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\erdbtn.png"))
    erepbtn = Button(otframe, image=eribtn,
bd=0, bg="white",command=erasertdetails)
    erepbtn.image = eribtn
    erepbtn.place(x=85, y=150)

    #report frame
    reframe = Frame(adm, bg="white")
    reframe.place(x=735, y=80, height=100,
width=400)
    kid = Label(reframe, text="BOOK-ID",
font=("Goudy old style", 25, "bold"), fg="gray",
bg="white").place(
        x=130, y=10)
    kide = Entry(reframe, font=("times new
roman", 15), bg="lightgray")
    kide.place(x=100, y=50, width=200,
height=28)

    #add book frame
    adbookframe = Frame(adm, bg="white")
```

```
    adbookframe.place(x=25, y=40, height=360,
width=600)

    #Add book headline
    hdbok = Label(adbookframe, text="ADD BOOK",
font=("Castellar", 25, "bold"), fg="black",
bg="orange2")
    hdbok.place(x=200, y=10)

    #Book id
    bookid = Label(adbookframe, text="BOOK-ID",
font=("Goudy old style", 25, "bold"), fg="gray",
bg="white").place(
        x=20, y=60)
    bookide = Entry(adbookframe, font=("times
new roman", 15), bg="lightgray")
    bookide.place(x=20, y=100, width=200,
height=28)

    #Book name
    booknam = Label(adbookframe, text="BOOK-
NAME", font=("Goudy old style", 25, "bold"),
fg="gray", bg="white").place(
        x=20, y=130)
    bookname= Entry(adbookframe, font=("times
new roman", 15), bg="lightgray")
    bookname.place(x=20, y=170, width=200,
height=28)

    #Author
    author = Label(adbookframe, text="AUTHOR",
font=("Goudy old style", 25, "bold"), fg="gray",
bg="white").place(
        x=20, y=200)
    authore = Entry(adbookframe, font=("times
```

```
new roman", 15), bg="lightgray")
    authore.place(x=20, y=240, width=200,
height=28)

    #Genres
    genres = Label(adbookframe, text="BOOK-
GENRES", font=("Goudy old style", 25, "bold"),
fg="gray", bg="white").place(
        x=20, y=270)
    genrese = Entry(adbookframe, font=("times
new roman", 15), bg="lightgray")
    genrese.place(x=20, y=310, width=200,
height=28)

    #Submit btn
    submitbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\submitbtn.png"))
    subtn = Button(adbookframe, image=submitbtn,
bd=0, bg="white",command=ak)
    subtn.image = submitbtn
    subtn.place(x=370, y=120)

    cancebtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\cnlbtn.png"))
    canl = Button(adbookframe, image=cancebtn,
command=adm.destroy, bd=0, bg="white")
    canl.image = cancebtn
    canl.place(x=365, y=200)

    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
```

```
    mycr = mydb.cursor()
    mycr.execute("select distinct(BOOK_GENRES)
from book;")
    for i in mycr:
        lis.append(i[0])
    clicked = StringVar()
    clicked.set("CHOOSE BOOK GENRES")
    drop = OptionMenu(adm, clicked, *lis)
    drop.place(x=50, y=460)
    sbl = Button(adm, text="SHOW BOOK LIST",
bg="lightgrey", font=("verdana", 10),
command=showadm).place(x=50, y=500)
    def clock():
        hour = time.strftime("%I")
        minute = time.strftime("%M")
        second = time.strftime("%S")
        am_pm = time.strftime("%p")
        day = time.strftime("%A")

        mylabel.config(text=hour + ":" + minute
+ ":" + second + " " + am_pm)
        mylabel.after(1000, clock)

    mylabel = Label(adm, text=" ",
font=("Helvetica", 20, "bold"),
fg="deepskyblue", bg="white")
    mylabel.place(x=50, y=550)

    clock()


#returnbook
def returnb():
    import mysql.connector as con
    mydb = con.connect(host="localhost",
```

```
user="root", password="1234",
database="library")
    myr = mydb.cursor()
    mycur = mydb.cursor()
    list8 = []
    list6 = []

    stt = "available"
    flag = False

    bn = int(bokid.get())
    dt = dat.get()
    snam = stname.get()
    clss= click.get()

    mycur.execute("select * from book where
BOOK_ID='{}'".format(bn))
    dta = mycur.fetchall()
    if dta == []:
        messagebox.showerror("error", "BOOK ID
NOT EXIST \n ENTER VALID BOOK ID", parent=win)
    else:
        mycur.execute("select BOOK_ID from book
where STATUS='available'")
        for x in mycur:
            list8.append(x[0])
        if bn in list8:
            messagebox.showerror("error", "BOOK
RETURNED ALREADY \n ENTER ANOTHER BOOK ID")
        else:
            myr.execute("select BOOK_NAME from
book where BOOK_ID='{}' ".format(bn))
            for d in myr:
                list6.append(d[0])
            mycur.execute("insert into
```

```
returnbook values('{}','{}','{}','{}','{}')
".format(bn, list6[0], snam, dt,clss))
            mycur.execute("update book set
STATUS='{}' where BOOK_ID='{}' ".format(stt,
bn))
            speak("BOOK RETURNED
SUCCESSFULLY...HAVE A NICE DAY")
            messagebox.showinfo("NOTE", "BOOK
RETURNED SUCCESSFULLY",parent=win)
            mydb.commit()



#ISSUE_BOOK
def issue():
    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    myr = mydb.cursor()
    mycurs = mydb.cursor()
    mycur = mydb.cursor()
    list1=[]
    list5 = []
    sts = "issued"


    bno = int(bokid.get())
    date = dat.get()
    stuname = stname.get()
    cls =click.get()
    mycur.execute("select * from book where
BOOK_ID='{}'".format(bno))
    dta = mycur.fetchall()
    if dta == []:
```

```
        messagebox.showerror("error", "BOOK ID
NOT EXIST \n ENTER VALID BOOK ID", parent=win)
    else:
        mycur.execute("select BOOK_ID from book
where STATUS='issued'")
        for x in mycur:
            list1.append(x[0])
        if bno in list1:
            messagebox.showerror("error", "BOOK
ALREADY ISSUED \n ENTER ANOTHER BOOK ID",
parent=win)
        else:
            myr.execute("select BOOK_NAME from
book where BOOK_ID='{}' ".format(bno))
            for d in myr:
                list5.append(d[0])
            mycur.execute("insert into issuebook
values('{}','{}','{}','{}','{}') ".format(bno,
list5[0], stuname, date,cls))
            mycur.execute("update book set
STATUS='{}' where BOOK_ID='{}' ".format(sts,
bno))
            mydb.commit()
            speak("BOOK ISSUES SUCCESSFULLY....
HAVE A NICE READING")
            messagebox.showinfo("NOTE", "BOOK
ISSUED SUCCESSFULLY", parent=win)



#Book details table
def show():
    sec = clicked.get()
    import mysql.connector as con
    mydb = con.connect(host="localhost",
```

```
user="root", password="1234",
database="library")
    mycur = mydb.cursor()
    mycr = mydb.cursor()
    mycur.execute("select * from book where
BOOK_GENRES ='{}' ".format(sec))
    data = mycur.fetchall()

    box = ttk.Treeview(win)

    box['columns'] = ("BOOK_ID", "BOOK_NAME",
"AUTHOR", "BOOK_GENRES", "STATUS")

    box.column("#0", width=0, stretch=NO)
    box.column("BOOK_ID", width=150)
    box.column("BOOK_NAME", width=150)
    box.column("AUTHOR", width=150)
    box.column("BOOK_GENRES", width=150)
    box.column("STATUS", width=150)

    box.heading("#0", text=" ", anchor=W)
    box.heading("BOOK_ID", text="BOOK_ID",
anchor=W)
    box.heading("BOOK_NAME", text="BOOK_NAME",
anchor=W)
    box.heading("AUTHOR", text="AUTHOR",
anchor=W)
    box.heading("BOOK_GENRES",
text="BOOK_GENRES", anchor=W)
    box.heading("STATUS", text="STATUS",
anchor=W)

    count = 0
    for t in data:
        box.insert('', 0, values=(t[0], t[1],
```

```
t[2], t[3], t[4]))
            count = count + 1

    box.place(x=235, y=260)

#student corner
def student_corner():
    lis = []
    lis1= [1,2,3,4,5,6,7,8,9,10,11,12]
    global bokid, stname, dat, win,
clicked,click

    import mysql.connector as con
    mydb = con.connect(host="localhost",
user="root", password="1234",
database="library")
    mycr = mydb.cursor()
    mycr.execute("select distinct(BOOK_GENRES)
from book;")
    for i in mycr:
        lis.append(i[0])

    win = Toplevel()

    win.title("STUDENT CORNER")
    win.geometry("1000x500")
    win.resizable(False, False)

    win.configure(bg="grey")

    win.bg =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\stuwall.png"))
    win.bg_image = Label(win,
image=win.bg).place(x=0, y=0, relwidth=1,
```

```
relheight=1)


    #title
    s = Label(win, text="STUDENT CORNER",
font=("jokerman", 20,"bold"), fg="deepskyblue")
    s.place(x=400,y=0)



    # book id
    frm1 = Frame(win, bg="lightgrey")
    frm1.place(relwidth=0.60, relheight=0.07,
x=20, y=50)
    s1 = Label(frm1, text="BOOK ID :",
font=("verdana", 15), fg="black")
    s1.place(relx=0.09, relwidth=0.30,
relheight=1)
    bokid = Entry(frm1, bg="lightgrey",
font=("verdana", 15))
    bokid.place(relx=0.45, relwidth=0.39,
relheight=1)

    # student name
    frm2 = Frame(win, bg="lightgrey")
    frm2.place(relwidth=0.60, relheight=0.07,
x=20, y=100)
    s2 = Label(frm2, text="STUDENT NAME :",
font=("verdana", 15), fg="black")
    s2.place(relx=0.09, relwidth=0.30,
relheight=1)
    stname = Entry(frm2, bg="lightgrey",
font=("verdana", 15))
    stname.place(relx=0.45, relwidth=0.39,
relheight=1)
```

```
    # date
    frm3 = Frame(win, bg="lightgrey")
    frm3.place(relwidth=0.60, relheight=0.07,
x=20, y=145)
    s3 = Label(frm3, text="DATE :",
font=("verdana", 15), fg="black")
    s3.place(relx=0.09, relwidth=0.30,
relheight=1)
    dat = Entry(frm3, bg="lightgrey",
font=("verdana", 15))
    dat.place(relx=0.45, relwidth=0.39,
relheight=1)
    dat.insert(0, "YYYY/MM/DD")

    frm4 = Frame(win, bg="lightgrey")
    frm4.place(relwidth=0.60, relheight=0.07,
x=20, y=189)
    s4 = Label(frm4, text="CLASS:",
font=("verdana", 15), fg="black")
    s4.place(relx=0.09, relwidth=0.30,
relheight=1)
    click = StringVar()
    click.set("CHOOSE CLASS")
    drop = OptionMenu(win, click, *lis1)
    drop.place(x=345, y=190)



issuebtn=ImageTk.PhotoImage(Image.open("C:\\User
s\\Admin\\Desktop\\python
project\\issuebtn.png"))
    isubtn =
Button(win,command=issue,image=issuebtn,bd=0,bg=
"black")
```

```
    isubtn.image=issuebtn
    isubtn.place(x=800, y=75)

    cancelbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\cance.png"))
    canl = Button(win,
image=cancelbtn,command=win.destroy,bd=0,bg="bla
ck")
    canl.image=cancelbtn
    canl.place(x=850, y=125)

    returnbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\returnbtn.png"))
    retbtn = Button(win, image=returnbtn,
bd=0,bg="black",command=returnb)
    retbtn.image = returnbtn
    retbtn.place(x=800, y=176)

    clicked = StringVar()
    clicked.set("CHOOSE BOOK GENRES")

    drop = OptionMenu(win, clicked, *lis)
    drop.place(x=50, y=295)

    sbl = Button(win, text="SHOW BOOK LIST",
bg="lightgrey", font=("verdana", 10),
command=show).place(x=50, y=330)

    def clock():
        hour = time.strftime("%I")
        minute = time.strftime("%M")
        second = time.strftime("%S")
        am_pm = time.strftime("%p")
```

```
        day = time.strftime("%A")

        mylabel.config(text=hour + ":" + minute
+ ":" + second + " " + am_pm)
        mylabel.after(1000, clock)

    mylabel = Label(win, text=" ",
font=("Helvetica", 20, "bold"),
fg="deepskyblue", bg="white")
    mylabel.place(x=10, y=450)

    clock()

    win.mainloop()




#LOGIN_MAIN
class Login:
    def __init__ (self,root):
        self.root = root
        self.root.title("Login System")
        self.root.geometry("626x417+100+50")
        self.root.resizable(False,False)
        #image

self.bg=ImageTk.PhotoImage(Image.open("C:\\Users
\\Admin\\Desktop\\python project\\sara.png"))

self.bg_image=Label(self.root,image=self.bg).pla
ce(x=0,y=0,relwidth=1,relheight=1)


        #login frame
```

```
        Frame_login=Frame(self.root,bg="white")

Frame_login.place(x=25,y=40,height=320,width=400
)

        #TIMEzzz
        def clock():
            hour = time.strftime("%I")
            minute = time.strftime("%M")
            second = time.strftime("%S")
            am_pm=time.strftime("%p")
            day=time.strftime("%A")

            mylabel.config(text=hour + ":" +
minute + ":" + second+" "+am_pm)
            mylabel.after(1000, clock)
            mylabel1.config(text=day)

        def update():
            mylabel.config(text="NEW TEXT")

        mylabel = Label(self.root, text=" ",
font=("Helvetica", 20,"bold"), fg="darkorange",
bg="white")
        mylabel.place(x=466,y=0)

        mylabel1 = Label(self.root, text=" ",
font=("Helvetica", 18,"bold"), fg="darkorange",
bg="white")
        mylabel1.place(x=466,y=35)



        clock()
```

```
title=Label(Frame_login,text="LOGIN",font=("Impa
ct",30,"bold"),fg="orange",bg="white").place(x=5
0,y=10)
        desc=Label(Frame_login, text="LIBRARY
LOGIN ", font=("Goudy old style", 15, "bold"),
fg="orange", bg="white").place(x=50, y=55)

        lab_user = Label(Frame_login,
text="USERNAME", font=("Goudy old style", 25,
"bold"), fg="gray", bg="white").place(x=50,
y=85)
        self.user=Entry(Frame_login,font=("times
new roman",15),bg="lightgray")

self.user.place(x=50,y=120,width=200,height=30)


        lab_user =
Label(Frame_login,text="PASSWORD",font=("Goudy
old style",25,
"bold"),fg="gray",bg="white").place(x=50, y=160)
        self.passw = Entry(Frame_login,
font=("times new roman", 15), bg="lightgray")
        self.passw.place(x=50, y=195, width=200,
height=30)
        self.passw.config(show="*")

        login =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\login.png"))

Login_btn=Button(Frame_login,command=self.login_
function,image=login,bd=0)
        Login_btn.image=login
```

```
        Login_btn.place(x=50,y=240)

        cancel =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\cancel.png"))

exit=Button(Frame_login,command=self.root.quit,i
mage=cancel,bd=0)
        exit.image=cancel
        exit.place(x=190,y=245)

    def login_function(self):
        if self.passw.get()=="" or
self.user.get()=="":
            messagebox.showerror("ERROR","All
Fields Are Required",parent=self.root)
        elif "1234" != self.passw.get() or
self.user.get()!= "admin":

messagebox.showerror("error","Invalid
Username/Password",parent=self.root)
        else:
            messagebox.showinfo("LOGED
IN","WELCOME TO LIBRARY",parent=self.root)
            speak("LOGIN SUCCESSFULL.....
WELCOME TO LIBRAARY")
            top = Toplevel()
            top.title("LIBRARY")
            top.geometry("626x417")
            top.resizable(False, False)
            top.bg =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\sara.png"))
            top.bg_image = Label(top,
image=top.bg).place(x=0, y=0)
```

```
            wt = Label(top, text="LIBRARY
MANAGEMENT", fg="white", bg="black",
font=("jokerman", 13, "bold"), width=100,
                       height=3)
            wt.pack()

            stubtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\studentbtn.png"))
            bt1 =
Button(top,image=stubtn,bd=0,command=student_cor
ner,bg="black")
            bt1.image=stubtn
            bt1.place(x=210, y=100)

            adbtn =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\adminbtn.png"))
            bt2 =
Button(top,image=adbtn,bd=0,command=admin,bg="bl
ack")
            bt2.image=adbtn
            bt2.place(x=225, y=230)

            logout =
ImageTk.PhotoImage(Image.open("C:\\Users\\Admin\
\Desktop\\python project\\logout.jpg"))
            bt3 =
Button(top,command=self.root.quit,image=logout,b
d=0,bg="black")
            bt3.image=logout
            bt3.place(x=7,y=365)

            def clock():
```

```
                hour = time.strftime("%I")
                minute = time.strftime("%M")
                second = time.strftime("%S")
                am_pm = time.strftime("%p")
                day = time.strftime("%A")

                mylabel.config(text=hour + ":" +
minute + ":" + second + " " + am_pm)
                mylabel.after(1000, clock)
            mylabel = Label(top, text=" ",
font=("Helvetica", 20, "bold"), fg="darkorange",
bg="white")
            mylabel.place(x=460, y=366)

            clock()



root = Tk()
obj = Login(root)
root.mainloop()
```

STRUCTURE OF SQL TABLES

- **BOOK DETAILS TABLE:**

Field	Type	Null	Key	Default	Extra
BOOK_ID	int	YES		NULL	
BOOK_NAME	char(50)	YES		NULL	
AUTHOR	char(50)	YES		NULL	
BOOK_GENRES	char(50)	YES		NULL	
STATUS	char(50)	YES		NULL	

- **BOOK ISSUE TABLE:**

Field	Type	Null	Key	Default	Extra
BOOK_ID	int	YES		NULL	
BOOK_NAME	char(50)	YES		NULL	
STUDENT_NAME	char(50)	YES		NULL	
DATE	date	YES		NULL	
CLASS	int	YES		NULL	

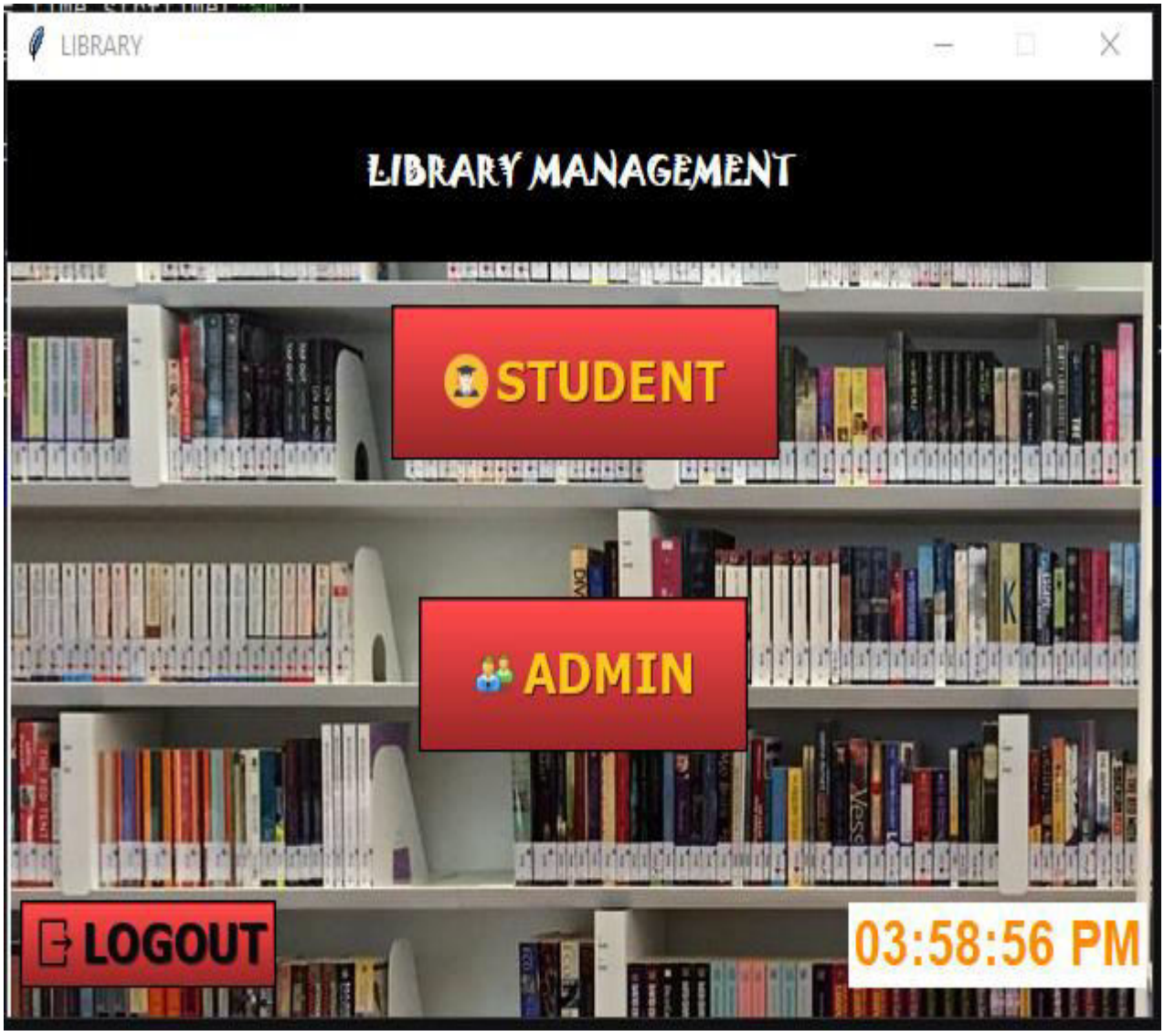
- **BOOK RETURN TABLE:**

Field	Type	Null	Key	Default	Extra
BOOK_ID	int	YES		NULL	
BOOK_NAME	char(50)	YES		NULL	
STUDENT_NAME	char(50)	YES		NULL	
DATE	date	YES		NULL	
CLASS	int	YES		NULL	

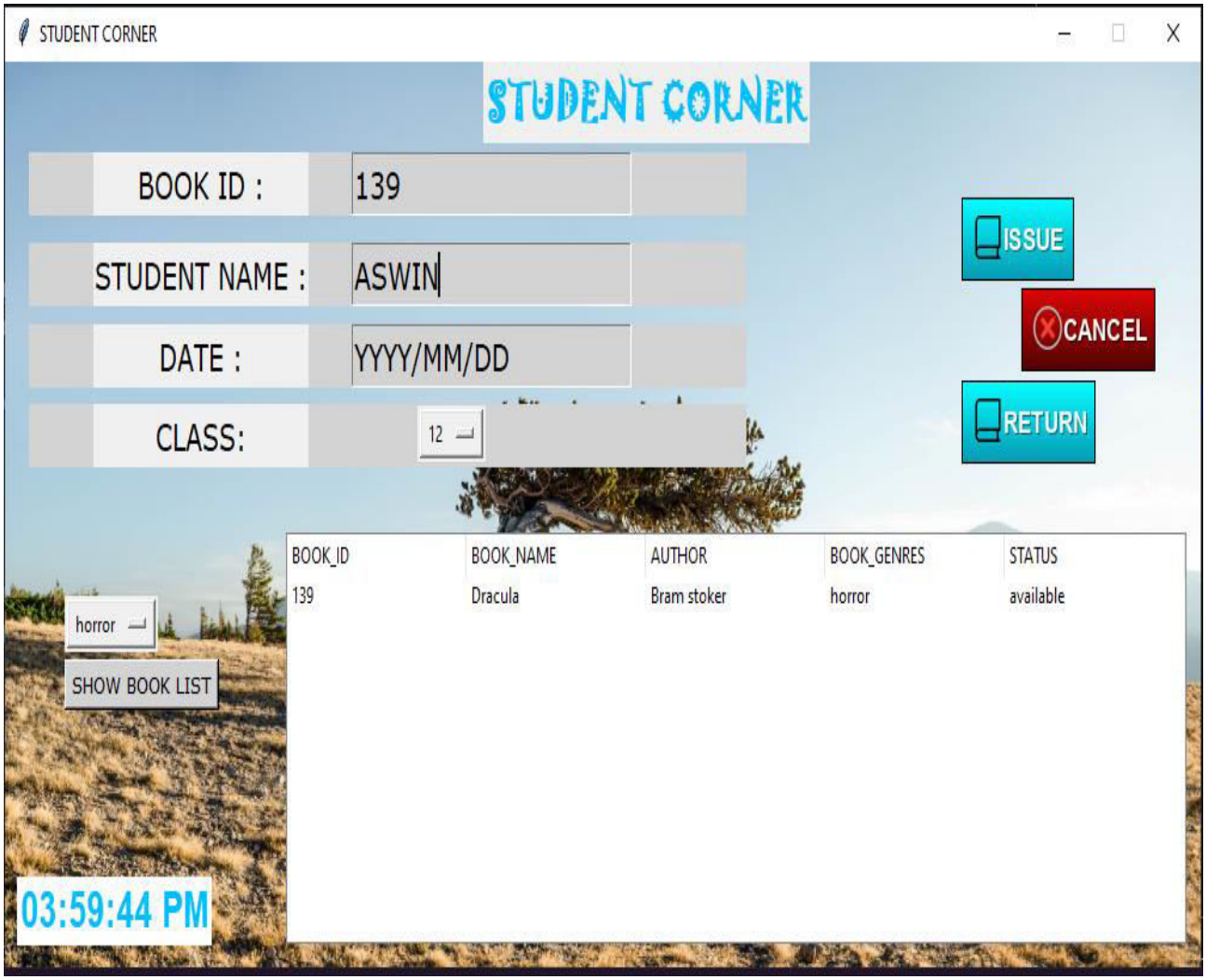
OUTPUT SCREEN

- **SCREEN 1 :**

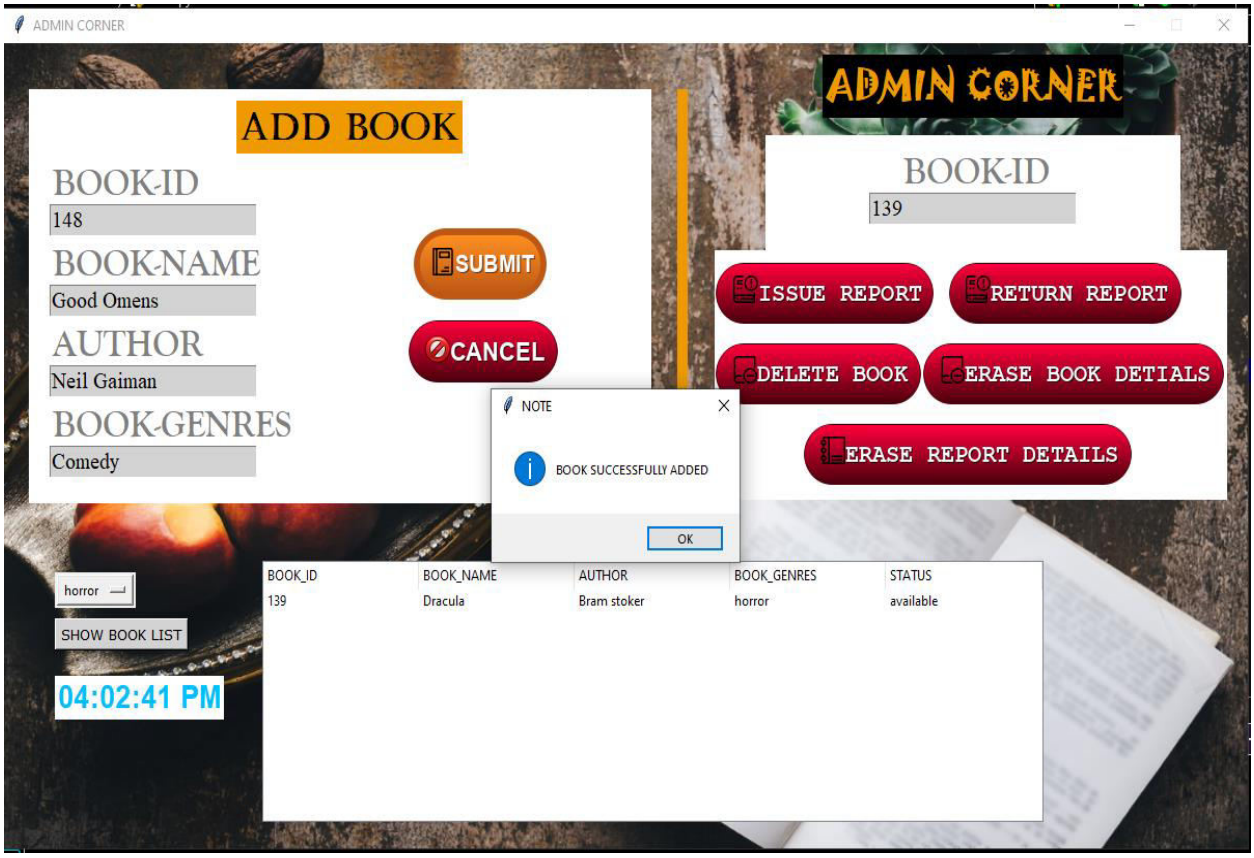
- **SCREEN 2:**


LIBRARY
LIBRARY MANAGEMENT
STUDENT
ADMIN
LOGOUT
03:58:56 PM

- **SCREEN 3:**


STUDENT CORNER
STUDENT CORNER
BOOK ID :
139
STUDENT NAME :
ASWIN
DATE :
YYYY/MM/DD
CLASS:
12
ISSUE
CANCEL
RETURN
BOOK_ID
BOOK_NAME
AUTHOR
BOOK_GENRES
STATUS
139
Dracula
Bram stoker
horror
available
horror
SHOW BOOK LIST
03:59:44 PM

- **SCREEN 4:**


ADMIN CORNER
ADD BOOK
BOOK-ID
148
BOOK-NAME
Good Omens
AUTHOR
Neil Gaiman
BOOK-GENRES
Comedy
SUBMIT
CANCEL
NOTE
BOOK SUCCESSFULLY ADDED
OK
ADMIN CORNER
BOOK-ID
139
ISSUE REPORT
RETURN REPORT
DELETE BOOK
ERASE BOOK DETIALS
ERASE REPORT DETAILS
horror
SHOW BOOK LIST
04:02:41 PM
BOOK_ID
BOOK_NAME
AUTHOR
BOOK_GENRES
STATUS
139
Dracula
Bram stoker
horror
available

BIBLIOGRAPHY

For more references read XII & XI text book of Preeti Arora or visit

https://codemy.com and

https://tutorialspoint.com