Lucknow International Public School

Project Report

Name	–	Aman Pal
Class	–	12 PCM
Roll No.	–	1
Subject	–	Computer
HOD	–	Mr. Rohit Sir

Acknowledgement

I would like to express my special thanks of gratitude to my teacher Mr. Rohit Sir as well as our Principal Mrs. Mandavi Tripathi Mam who gave me the golden opportunity to create this wonderful Computer Project.

This Project helped me a lot in gaining adequate knowledge about the topic. I have completed this project after proper analysis and research and I came to know about so many new things.

I am really thankful to all my friends, co-partners, and a guide, who have devoted their precious time in completing my project. Secondly I would also like to thank my parents who helped me a lot in finishing this project within the prescribed time.

I am making this project not only for marks but also to increase my knowledge and intellect development

Thanks again to all who helped me.

Certificate

This is to certify that this computer project is a bonafide work of Aman Pal, a student of class 12, who carried out the work under the supervision of his Computer Teacher Rohit Sir.

Signature of Teacher

Preface

The Central Board of Secondary Education has included in its course, a full fledged computer course covering the fundamentals of computer and programming. Exploring the world of computers, and the project is both informative and exciting. The project “School Management System” has been allotted to me. This project work allocated to me is a part of the entire process involved in computerization of the School Management System.

TOPIC OF THE PROJECT

Library Management System

BACKGROUND OF THE PROJECT

INTRODUCTION

In today's world, all the things have become computerized. Generally, the library works on paper work. To issue library card keep book's record and issue books, there is need of a lot paper work. So, this software is useful for doing library's work easy. It registers the book's information such as- book id, name, author, publisher and price. Generally, to issue books, the librarian has to keep record on paper. By using this software, he/she can easily issue book to the student with a few clicks. Hence, this software makes easy work for library's management

OBJECTIVES OF THE PROJECT

- The objective of this project is to let the students apply the programming knowledge into a real- world situation/problem and exposed the students how programming skills helps in developing a good software.
- Write programs utilizing modern software tools.
- Apply object oriented programming principles effectively when developing small to medium sized projects.
- Write effective procedural code to solve small to medium sized problems.
- Students will demonstrate a breadth of knowledge in computer science, as exemplified in the areas of systems, theory and software development.

USE OF TECHNOLOGY

1) What is MySQL?

MySQL is a relational DBMS that can run virtually all platforms, including Linux, Unix and Windows. Popular for web-based applications and online publishing, MySQL is a part of open-source enterprise stack LAMP (Linux, Apache, MySQL, PHP).

MySQL is a freely available open source RDBMS that uses Structured Query Language (SQL). It is down-loadable from site www.mysql.org MySQL is fast, reliable, scalable alternative to many of the commercial RDBMs available today. MySQL provides you with a rich set of features that support a secure environment for storing, maintaining, and accessing data.

MySQL was created and supported by MySQL AB, a company based in Sweden. This company is now a subsidiary of Sun Microsystems, which holds the copyright to most of the codebase. On April 20th, 2009 Oracle Corp., which develops and sells the proprietary Oracle database, announced a deal to acquire Sun Microsystems.

SQL provides many different types of commands used for different purposes. SQL commands can be divided into following categories:

i. Data Definition Language (DDL)
ii. Data Manipulation Language (DML)
iii. Transaction Control Language (TCL)
iv. Session Control Commands v. System Control Commands

2. What is Python?

Python is an interpreter based, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all 20 major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

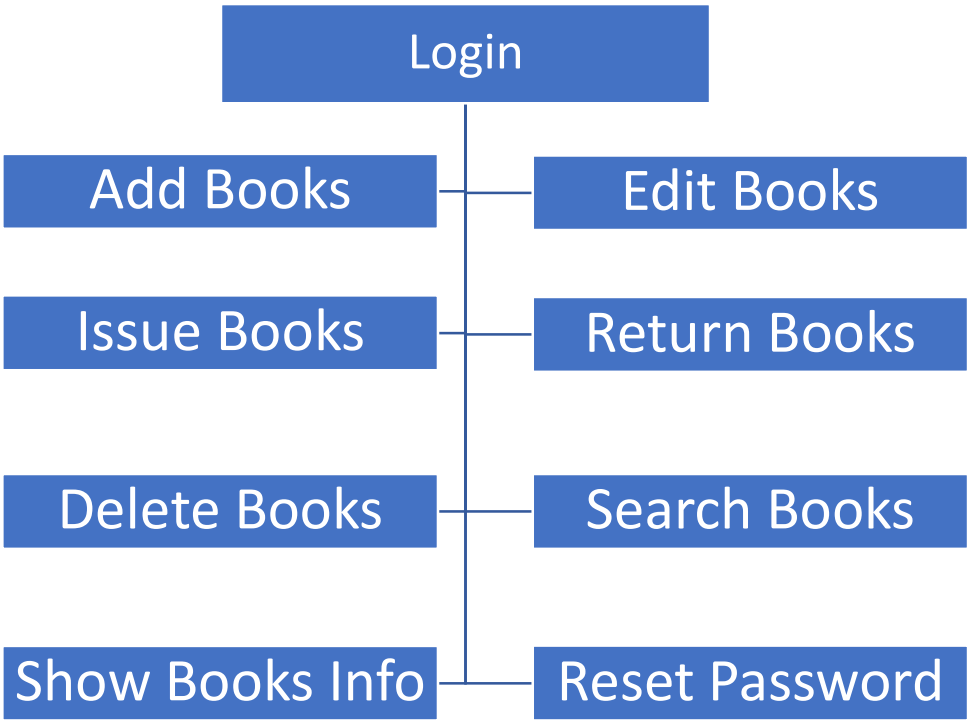
FUNCTIONS AND MODULES

1. **Tkinter:** Tkinter is the most commonly used library for developing GUI (Graphical User Interface) in Python. It is a standard Python interface to the Tk GUI toolkit shipped with Python. As Tk and Tkinter are available on most of the Unix platforms as well as on the Windows system, developing GUI applications with Tkinter becomes the fastest and easiest.
2. **SQLite3:** Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases.
3. **SMTPLib:** SMTPLib creates a Simple Mail Transfer Protocol client session object which is used to send emails to any valid email id on the internet.
4. **DateTime:** This package provides basic functions for display date related values in the program.
5. **Hashlib:** There are many hash functions defined in the HASHLIB library in python. In Python, we can use hashlib.md5() to generate a MD5 hash value from a String.
6. **OS:** The OS module in Python provides functions for interacting with the operating system. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system.

FLOW OF THE PROJECT

1. Our project is based on library management system. When you run main file, you will see login screen. Login to the admin account. You can also reset password of your account.
2. In the dashboard screen there is 7 options i.e.
    1. **Add Books**
    2. **Issue books**
    3. **Edit books**
    4. **Return books**
    5. **Delete books**
    6. **Search books**
    7. **Show books**
3. **Add Books:** in this option you can add book's information such as BOOK ID, NAME, AUTHOR, PUBLISHER and PRICE.
4. **Issue Books:** in this option you can issue book by entering student's id and book id. You can also send emails to the students about the information of the book issued.
5. **Edit books:** in this option you can edit book's information by entering BOOK ID. You can edit BOOK ID, NAME, AUTHOR, PUBLISHER and PRICE.
6. **Return books:** in this option you can return books by entering STUDENT'S ID and Book ID.
7. **Delete Books:** by this option you can delete books by entering Book ID.
8. **Search books:** by this option you can search for a particular book's information such as NAME, AUTHOR, PUBLISHER and availability by entering Book ID.
9. **Show Books:** by this option you can see all the books available in your library and you can also export this information in csv format.

FLOW CHART


Login
Add Books
Edit Books
Issue Books
Return Books
Delete Books
Search Books
Show Books Info
Reset Password

STRUCTURE OF TABLE

UserLogin

Field	Type
UserID	VARCHAR(15)
Password	VARCHAR(64)

Books

Field	Type
BookID	INTEGER
Title	CHAR
Author	CHAR
Publisher	CHAR
Price	CHAR
Issue	CHAR
ID	INTEGER
IssueDate	DATE

Students

Field	Type
ID	INTEGER
Name	CHAR
Class	CHAR
Email	CHAR
Contact	CHAR
NoBook	INTEGER

HARDWARE AND SOFTWARE REQUIREMENTS

1. OPERATING SYSTEM : WINDOWS 7 AND ABOVE
2. PROCESSOR : PENTIUM(ANY) OR AMD
3. MOTHERBOARD : 1.845 OR 915 FOR PENTIUM
4. RAM : 512MB+
5. Hard disk : SATA 40 GB OR ABOVE
6. MONITOR 14.1 or 15 -17 inch
7. Key board and mouse

SOFTWARE REQUIREMENTS:

1. Windows OS
2. Python
3. MySQL

INSTALLATION PROCEDURE

1. There will be two folders namely 'Python Files' and 'EXE files'.
2. The folder 'Python Files' will contain the source code of the software in python language. If you are running the software by the 3rd step mentioned below you have to pre install the following modules :-
    a) Tkinter
    b) Tkcalender
    c) Smtplib
    d) Python-mysql-connector
    e) Datetime
    f) Os
    g) Sys
    h) Hashlib
3. Open the files in any python editors and run it to start and work on the software.
4. The folder 'EXE files' will contain 'main.exe' run the file 'main.exe' to start and work on the software.

SOURCE CODE and OUTPUT

```
import hashlib,random,smtplib,ssl,time
from datetime import date
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from tkinter import *
from tkinter import messagebox, ttk
from database import db,dbstore,dbstudents

root = Tk()
root.title("Library Management System")
root.iconbitmap('Images/library.ico')
root.geometry("900x500+50+100")
root.resizable(0, 0)
class main:
    def login(self):
        self.var1 = self.e1.get()
        self.var2 = self.e2.get()
        cursor = db.cursor(buffered=True)
        cursor.execute("SELECT * FROM UserLogin WHERE UserID='" + self.var1 + "' and Password='" +
hashlib.md5(
            self.var2.encode()).hexdigest() + "'")
        db.commit()
        self.ab = cursor.fetchone()
        if self.ab != None:
            self.under_fm = Frame(root, height=500, width=900, bg='#fff')
            self.under_fm.place(x=0, y=0)
            self.fm2 = Frame(root, bg='#223c83', height=80, width=900)
            self.fm2.place(x=0, y=0)
            self.lbb = Label(self.fm2, bg='#223c83')
            self.lbb.place(x=15, y=5)
            self.ig = PhotoImage(file='Images/library.png')
            self.lbb.config(image=self.ig)
            self.lb3 = Label(self.fm2, text='DASHBOARD', fg='#d39c86', bg='#223c83', font=('Georgia', 30, 'bold'))
            self.lb3.place(x=325, y=17)
            self.name = Label(root, text="Name : ", bg='#fff', fg="black", font=('Georgia', 12, 'bold'))
            self.name.place(x=5, y=83)
            self.name1 = Label(root, text=self.ab[0], fg='black', bg='#fff', font=('Georgia', 12, 'bold'))
            self.name1.place(x=60, y=83)
            self.today = date.today()
            self.dat = Label(root, text='Date : ', bg='#fff', fg='black', font=('Georgia', 12, 'bold'))
            self.dat.place(x=750, y=83)
```

```
            self.dat2 = Label(root, text=self.today, bg='#fff', fg='black', font=('Georgia', 12, 'bold'))
            self.dat2.place(x=800, y=83)
            self.cur()
        else:
            messagebox.showerror('Library System', 'Your ID or Password is invalid!')
    def cur(self):
        self.fm3 = Frame(root, bg='#fff', width=900, height=390)
        self.fm3.place(x=0, y=110)
        def clock():
            h = str(time.strftime("%H"))
            m = str(time.strftime("%M"))
            s = str(time.strftime("%S"))
            if int(h) >= 12 and int(m) >= 0:
                self.lb7_hr.config(text="PM")
            self.lb1_hr.config(text=h)
            self.lb3_hr.config(text=m)
            self.lb5_hr.config(text=s)
            self.lb1_hr.after(200, clock)
        self.lb1_hr = Label(self.fm3, text='12', font=('Georgia', 20, 'bold'), bg='#223c83', fg='white')
        self.lb1_hr.place(x=607, y=0, width=60, height=30)
        self.lb3_hr = Label(self.fm3, text='05', font=('Georgia', 20, 'bold'), bg='#223c83', fg='white')
        self.lb3_hr.place(x=677, y=0, width=60, height=30)
        self.lb5_hr = Label(self.fm3, text='37', font=('Georgia', 20, 'bold'), bg='#223c83', fg='white')
        self.lb5_hr.place(x=747, y=0, width=60, height=30)
        self.lb7_hr = Label(self.fm3, text='AM', font=('Georgia', 17, 'bold'), bg='#223c83', fg='white')
        self.lb7_hr.place(x=817, y=0, width=60, height=30)
        clock()
        self.canvas8 = Canvas(self.fm3, bg='black', width=400, height=300)
        self.canvas8.place(x=475, y=40)
        self.photo9 = PhotoImage(file="Images/home.png")
        self.canvas8.create_image(0, -45, image=self.photo9, anchor=NW)
        self.develop = Label(self.fm3, text='Developed By - Aman Pal', bg='#fff', fg='red', font=('Candara', 12, 'bold'))
        self.develop.place(x=700, y=350)
        self.bt1 = Button(self.fm3, text='  Add Books', fg='#fff', bg='#223c83', font=('Candara', 15, 'bold'),width=170,
height=0, bd=7, relief='flat', command=self.addbook, cursor='hand2',activebackground='black',
activeforeground='#223c83')
        self.bt1.place(x=40, y=40)
        self.logo = PhotoImage(file='Images/add-book.png')
        self.bt1.config(image=self.logo, compound=LEFT)
        self.small_logo = self.logo.subsample(1, 1)
        self.bt1.config(image=self.small_logo)
        self.bt2 = Button(self.fm3, text='  Issue Books', fg='#fff', bg='#223c83', font=('Candara', 15, 'bold'),width=170,
height=0, bd=7, relief='flat', command=self.issuebook, cursor='hand2',activebackground='black',
activeforeground='#223c83')
        self.bt2.place(x=250, y=40)
```

```
        self.log = PhotoImage(file='Images/issue.png')
        self.bt2.config(image=self.log, compound=LEFT)
        self.small_log = self.log.subsample(1, 1)
        self.bt2.config(image=self.small_log)
        self.bt3 = Button(self.fm3, text='  Edit Books', fg='#fff', bg='#223c83', font=('Candara', 15, 'bold'),width=170,
height=0, bd=7, relief='flat', cursor='hand2', command=self.edit,activebackground='black',
activeforeground='#223c83')
        self.bt3.place(x=40, y=120)
        self.logb = PhotoImage(file='Images/edit.png')
        self.bt3.config(image=self.logb, compound=LEFT)
        self.small_logb = self.logb.subsample(1, 1)
        self.bt3.config(image=self.small_logb)
        self.bt4 = Button(self.fm3, text='  Return Books', fg='#fff', bg='#223c83', font=('Candara', 15,
'bold'),width=170, height=0, bd=7, relief='flat', cursor='hand2',
command=self.returnbook,activebackground='black', activeforeground='#223c83')
        self.bt4.place(x=250, y=120)
        self.log4 = PhotoImage(file='Images/return.png')
        self.bt4.config(image=self.log4, compound=LEFT)
        self.small_log4 = self.log4.subsample(1, 1)
        self.bt4.config(image=self.small_log4)
        self.bt5 = Button(self.fm3, text=' Delete Books', fg='#fff', bg='#223c83', font=('Candara', 15,
'bold'),width=170, height=0, bd=7, relief='flat', cursor='hand2', command=self.delete,activebackground='black',
activeforeground='#223c83')
        self.bt5.place(x=40, y=200)
        self.log5 = PhotoImage(file='Images/delete-book.png')
        self.bt5.config(image=self.log5, compound=LEFT)
        self.small_log5 = self.log5.subsample(1, 1)
        self.bt5.config(image=self.small_log5)
        self.bt6 = Button(self.fm3, text=' Show Books', fg='#fff', bg='#223c83', font=('Candara', 15, 'bold'),width=170,
height=0, bd=7, relief='flat', cursor='hand2', command=self.show,activebackground='black',
activeforeground='#223c83')
        self.bt6.place(x=40, y=280)
        self.log6 = PhotoImage(file='Images/show-books.png')
        self.bt6.config(image=self.log6, compound=LEFT)
        self.small_log6 = self.log6.subsample(1, 1)
        self.bt6.config(image=self.small_log6)
        self.bt7 = Button(self.fm3, text='  Search Books', fg='#fff', bg='#223c83', font=('Candara', 15,
'bold'),width=170, height=0, bd=7, relief='flat', cursor='hand2', command=self.search,activebackground='black',
activeforeground='#223c83')
        self.bt7.place(x=250, y=200)
        self.log7 = PhotoImage(file='Images/search.png')
        self.bt7.config(image=self.log7, compound=LEFT)
        self.small_log7 = self.log7.subsample(1, 1)
        self.bt7.config(image=self.small_log7)
```

```
        self.bt8 = Button(self.fm3, text='  Log Out', fg='#fff', bg='#223c83', font=('Candara', 15, 'bold'),
width=170,height=0, bd=7, relief='flat', cursor='hand2', command=self.code,
activebackground='black',activeforeground='#223c83')
        self.bt8.place(x=250, y=280)
        self.log8 = PhotoImage(file='Images/logout.png')
        self.bt8.config(image=self.log8, compound=LEFT)
        self.small_log8 = self.log8.subsample(1, 1)
        self.bt8.config(image=self.small_log8)
    def addbook(self):
        class temp(main):
            def book(self):
                self.fm = Frame(root, bg='#ffe8ec', width=900, height=390)
                self.fm.place(x=0, y=110)
                self.fm1 = Frame(self.fm, bg='#ffe8ec', width=500, height=360, bd=5, relief='flat')
                self.fm1.place(x=200, y=15)
                self.backbt = Button(self.fm, width=60, bg='#ffe8ec', bd=0, relief='flat',
command=self.cur,activeforeground='black', activebackground='#ffe8ec')
                self.backbt.place(x=2, y=7)
                self.log = PhotoImage(file='Images/back.png')
                self.backbt.config(image=self.log, compound=LEFT)
                self.small_log = self.log.subsample(2, 2)
                self.backbt.config(image=self.small_log)
                self.fll = Frame(self.fm1, width=150, height=40, bg='#223c83')
                self.fll.place(x=150, y=15)
                self.ll = Label(self.fll, text='ADD BOOKS', fg='#fff', bg='#223c83', font=('Georgia', 12, 'bold'),width=13)
                self.ll.place(x=0, y=8)
                self.lb = Label(self.fm1, text='ID', fg='black', bg='#ffe8ec', font=('Georgia', 18, 'bold'))
                self.lb.place(x=70, y=90)
                self.lb2 = Label(self.fm1, text='Title', fg='black', bg='#ffe8ec', font=('Georgia', 18, 'bold'))
                self.lb2.place(x=70, y=130)
                self.lb3 = Label(self.fm1, text='Author', fg='black', bg='#ffe8ec', font=('Georgia', 18, 'bold'))
                self.lb3.place(x=70, y=170)
                self.lb4 = Label(self.fm1, text='Publisher', fg='black', bg='#ffe8ec', font=('Georgia', 18, 'bold'))
                self.lb4.place(x=70, y=210)
                self.lb5 = Label(self.fm1, text='Price', fg='black', bg='#ffe8ec', font=('Georgia', 18, 'bold'))
                self.lb5.place(x=70, y=250)
                self.ee1 = Entry(self.fm1, width=25, bd=1, relief='groove', font=('Georgia', 18, 'bold'))
                self.ee1.place(x=200, y=88)
                self.ee2 = Entry(self.fm1, width=25, bd=1, relief='groove', font=('Georgia', 18, 'bold'))
                self.ee2.place(x=200, y=130)
                self.ee3 = Entry(self.fm1, width=25, bd=1, relief='groove', font=('Georgia', 18, 'bold'))
                self.ee3.place(x=200, y=170)
                self.ee4 = Entry(self.fm1, width=25, bd=1, relief='groove', font=('Georgia', 18, 'bold'))
                self.ee4.place(x=200, y=210)
                self.ee5 = Entry(self.fm1, width=25, bd=1, relief='groove', font=('Georgia', 18, 'bold'))
```

```
            self.ee5.place(x=200, y=250)
            self.bt = Button(self.fm1, text='SUBMIT', width=8, fg='white', bg='#223c83',font=('Georgia', 20, 'bold'),
bd=3, relief='flat', command=self.submit1,activebackground='black', activeforeground='#ff6690')
            self.bt.place(x=150, y=300)
        def submit1(self):
            try:
                self.id = self.ee1.get()
                self.ttl = self.ee2.get()
                self.aut = self.ee3.get()
                self.edi = self.ee4.get()
                self.pri = self.ee5.get()
                if (self.id and self.ttl and self.aut and self.edi and self.pri):
                    cursor = dbstore.cursor(buffered=True)
                    cursor.execute("INSERT INTO Books(BookID,Title,Author,Publisher,Price) values(?,?,?,?,?)",
(self.id,   self.ttl, self.aut, self.edi, self.pri))
                    dbstore.commit()
                    messagebox.showinfo("Success", "Book has been added to the library succesfully")
                    self.clear()
                else:
                    messagebox.showerror("Error", "Enter Valid Details")
            except Exception as e:
                messagebox.showerror("Error", "Enter Valid Details")
        def clear(self):
            self.ee1.delete(0, END)
            self.ee2.delete(0, END)
            self.ee3.delete(0, END)
            self.ee4.delete(0, END)
            self.ee5.delete(0, END)
    obj = temp()
    obj.book()
  def issuebook(self):
    class test(main):
        max = 0
        n = 1
        def issue(self):
            self.f = Frame(root, bg='#ffe8ec', width=900, height=390)
            self.f.place(x=0, y=110)
            self.fmi = Canvas(self.f, bg='#ffe8ec', width=900, height=390, bd=0, relief='flat')
            self.fmi.place(x=0, y=0)
            self.fc = Frame(self.fmi, bg='#ffe8ec', width=338, height=230, bd=1, relief='flat')
            self.fc.place(x=70, y=20)
            self.ffbll = Frame(self.fc, bg='#223c83', bd=2, relief='flat', width=310, height=40)
            self.ffbll.place(x=0, y=0)
            self.lc = Label(self.ffbll, text='STUDENT  INFORMATION', bg='#223c83', fg='#fff',font=('Georgia', 16,
'bold'))
```

```
            self.lc.place(x=0, y=6)
            self.lb = Label(self.fc, text='Student ID', bg='#ffe8ec', fg='black', font=('Georgia', 16, 'bold'))
            self.lb.place(x=15, y=90)
            self.em2 = Entry(self.fc, width=30, font=('Georgia', 14, 'bold'))
            self.em2.place(x=150, y=90)
            self.bt = Button(self.fc, text='SUBMIT', width=8, bg='#223c83', fg='#fff', font=('Georgia', 16, 'bold'),bd=5,
relief='flat', command=self.check, activeforeground='#223c83',activebackground='#adefd1')
            self.bt.place(x=15, y=160)
            self.backbt = Button(self.fmi, width=60, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,
relief='flat',command=self.issueback)
            self.backbt.place(x=5, y=5)
            self.log = PhotoImage(file='Images/back.png')
            self.backbt.config(image=self.log, compound=LEFT)
            self.small_log = self.log.subsample(2, 2)
            self.backbt.config(image=self.small_log)
        def check(self):
            self.b = self.em2.get()
            cursor = dbstudents.cursor(buffered=True)
            cursor.execute("SELECT * FROM Students WHERE ID='" + self.b + "'")
            self.var = cursor.fetchone()
            if self.var != None:
                self.fmii = Canvas(self.f, bg='#ffe8ec', width=338, height=90, bd=0, relief='flat')
                self.fmii.place(x=70, y=255)
                self.lb1 = Label(self.fmii, text='Name :', fg='black', bg='#ffe8ec', font=('Georgia', 14, 'bold'))
                self.lb1.place(x=5, y=5)
                self.lb2 = Label(self.fmii, text=self.var[1], fg='black', bg='#ffe8ec',font=('Georgia', 14, 'bold'))
                self.lb2.place(x=70, y=5)
                self.lb3 = Label(self.fmii, text='Class :', fg='black', bg='#ffe8ec', font=('Georgia', 14, 'bold'))
                self.lb3.place(x=5, y=30)
                self.lb4 = Label(self.fmii, text=self.var[2], fg='black', bg='#ffe8ec',font=('Georgia', 14, 'bold'))
                self.lb4.place(x=70, y=30)
                self.lb5 = Label(self.fmii, text='Email :', fg='black', bg='#ffe8ec', font=('Georgia', 14, 'bold'))
                self.lb5.place(x=5, y=55)
                self.lb6 = Label(self.fmii, text=self.var[3], fg='black', bg='#ffe8ec',font=('Georgia', 14, 'bold'))
                self.lb6.place(x=70, y=55)
                self.fr = Frame(self.fmi, bg='#ffe8ec', bd=5, relief='flat', width=338, height=250)
                self.fr.place(x=420, y=20)
                self.ff = Frame(self.fr, bg='#223c83', bd=2, relief='flat', width=160, height=40)
                self.ff.place(x=80, y=0)
                self.lb = Label(self.ff, text='ISSUE BOOK', bg='#223c83', fg='#fff', font=('Georgia', 14, 'bold'))
                self.lb.place(x=13, y=5)
                self.tt = Label(self.fr, text='BOOK ID', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
                self.tt.place(x=30, y=90)
                self.e1 = Entry(self.fr, width=30, font=('Georgia', 14, 'bold'))
                self.e1.place(x=130, y=90)
```

```
            self.bt1 = Button(self.fr, text='SUBMIT', width=8, bg='#223c83', fg='#fff', font=('Georgia', 14, 'bold'),
bd=5, relief='flat', command=self.data, activeforeground='#adefd1', activebackground='#223c83')
            self.bt1.place(x=15, y=160)
        else:
            messagebox.showwarning('Warning', 'This student is not registered !')
            self.em2.delete(0, END)
    def issueback(self):
        try:
            self.boot.destroy()
            self.cur()
        except Exception as e:
            self.cur()
    repeat = 0
    def data(self):
        self.b = self.em2.get()
        cursor = dbstudents.cursor(buffered=True)
        cursor.execute("SELECT * FROM Students WHERE ID='" + self.b + "'")
        self.var = cursor.fetchone()
        self.flag = 0
        print(self.var)
        if (int(self.var[5]) >= 3):
            try:
                self.boot.destroy()
                messagebox.showerror("Unable to process request", "You exceed the limit of Books per student!")
                self.flag = 1
                self.cur()
            except Exception as e:
                messagebox.showerror("Unable to process request", "You exceed the limit of Books per student!")
                self.flag = 1
                self.cur()
        self.vva = self.e1.get()
        cursor = dbstore.cursor(buffered=True)
        cursor.execute("SELECT * FROM Books WHERE BookID='" + self.vva + "'")
        dbstore.commit()
        self.value = cursor.fetchone()
        if self.value != None:
            if (self.flag != 1) and self.value[5] != "Issued":
                self.boot = Tk()
                self.boot.title("Issue Books")
                self.boot.iconbitmap("Images/library.ico")
                self.boot.configure(bg='#ffe8ec')
                self.boot.geometry("370x200+880+30")
                self.boot.resizable(0, 0)
                test.repeat = 1
                self.lb = Label(self.boot, text='Title:', bg='#ffe8ec', fg='black',   font=('Georgia', 12, 'bold'))
```

```
                self.lb.place(x=30, y=30)
                self.lbn = Label(self.boot, text=self.value[1], bg='#ffe8ec', fg='black',    font=('Georgia', 12, 'bold'))
                self.lbn.place(x=120, y=30)
                self.lb = Label(self.boot, text='Author:', bg='#ffe8ec', fg='black',   font=('Georgia', 12, 'bold'))
                self.lb.place(x=30, y=60)
                self.lbn = Label(self.boot, text=self.value[2], bg='#ffe8ec', fg='black',    font=('Georgia', 12, 'bold'))
                self.lbn.place(x=120, y=60)
                self.lb = Label(self.boot, text='Publisher:', bg='#ffe8ec', fg='black',   font=('Georgia', 12, 'bold'))
                self.lb.place(x=30, y=90)
                self.lbn = Label(self.boot, text=self.value[3], bg='#ffe8ec', fg='black',    font=('Georgia', 12, 'bold'))
                self.lbn.place(x=120, y=90)
                self.button1 = Button(self.boot, text='ISSUE', bg='#223c83', fg='#fff', width=10, height=0,
font=('Georgia', 11, 'bold'), activebackground='#223c83',    activeforeground='#adefd1', command=self.issued)
                self.button1.place(x=30, y=150)
                self.btn = Button(self.boot, text='SEND MAIL', bg='#223c83', fg='#fff', width=10,
height=0,font=('Georgia', 11, 'bold'), activebackground='#223c83',activeforeground='#adefd1',
command=self.mail)
                self.btn.place(x=160, y=150)
                self.boot.mainloop()
            else:
                messagebox.showinfo('Library Management System', 'Book Already Issued')
        else:
            messagebox.showerror('Book Not Found', 'No such book exists!')
            self.e1.delete(0, END)
    def yes(self):
        self.n = self.n + 1
        self.bt1 = Button(self.fr, text='SUBMIT', width=8, bg='#adefd1', fg='#fff',font=('Georgia', 12, 'bold'), bd=5,
relief='flat', command=self.data,activeforeground='#adefd1', activebackground='#223c83', state=ACTIVE)
        self.bt1.place(x=15, y=160)
        self.e1.delete(0, END)
        self.max = self.max - 1
    def no(self):
        self.bt1 = Button(self.fr, text='SUBMIT', width=8, bg='#adefd1', fg='#fff',font=('Georgia', 12, 'bold'), bd=5,
relief='flat', command=self.data,activeforeground='#adefd1', activebackground='#223c83', state=DISABLED)
        self.bt1.place(x=15, y=160)
    def issued(self):
        self.ac = self.e1.get()
        self.date = date.today().strftime("%m/%d/%Y")
        cursor = dbstore.cursor(buffered=True)
        cursor.execute(
            "UPDATE Books SET Issue='Issued', ID='" + self.b + "', IssueDate='" + self.date + "' WHERE
BookID='" + self.ac + "'")
        dbstore.commit()
        if self.n <= 3:
            book = dbstudents.cursor(buffered=True)
```

```
            self.IDid1 = self.em2.get()
            book.execute("SELECT * FROM Students WHERE ID='" + self.IDid1 + "'")
            self.issuevar = book.fetchone()
            self.sum = self.issuevar[5] + 1
            book.execute("UPDATE Students SET NoBook='" + str(self.sum) + "' WHERE ID='" + self.b + "' ")
            dbstudents.commit()
        messagebox.showinfo('Library Management System', 'YOUR BOOK HAS BEEN ISSUED')
        self.boot.destroy()
        self.e1.delete(0, END)
    def mail(self):
        self.IDid = self.em2.get()
        cursor = dbstudents.cursor(buffered=True)
        cursor.execute("SELECT * FROM Students WHERE ID='" + self.IDid + "'")
        self.var = cursor.fetchone()
        name = self.var[1]
        reciever = self.var[3]
        self.vva = self.e1.get()
        cursor = dbstore.cursor(buffered=True)
        cursor.execute("SELECT * FROM Books WHERE BookID='" + self.vva + "'")
        dbstore.commit()
        self.value = cursor.fetchone()
        bookid = self.value[0]
        booktitle = self.value[1]
        author = self.value[2]
        publisher = self.value[3]
        msg_body = f"""Hello {name}!\nYour book, {booktitle}, has been Issued\nBook Details\nID:
{bookid}\nBook Title: {booktitle}\nBook Author: {author}\nPublisher: {publisher}\n\nHappy Reading!"""
        sender_email = "com.amanpal@gmail.com"
        receiver_email = reciever
        password = 'euhvogkffgvticqb'
        message = MIMEMultipart("alternative")
        message["Subject"] = "LIPS Library Management System"
        message["From"] = sender_email
        message["To"] = receiver_email
        try:
            message.attach(MIMEText(msg_body, "plain"))
            context = ssl.create_default_context()
            with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
                server.login(sender_email, password)
                server.sendmail(
                    sender_email, receiver_email, message.as_string()
                )
            messagebox.showinfo("Library System", "Send mail Successfully !")
        except Exception as e:
            print(e)
```

```
                messagebox.showinfo("Library System", "An Error Occured While Sending Mail")
        obissue = test()
        obissue.issue()
    def edit(self):
        class editing(main):
            def edbooks(self):
                self.ffm = Frame(root, bg='#ffe8ec', width=900, height=390)
                self.ffm.place(x=0, y=110)
                self.fm1 = Frame(self.ffm, bg='#ffe8ec', width=500, height=270, bd=5, relief='flat')
                self.fm1.place(x=150, y=30)
                self.ed = Frame(self.fm1, bg='#223c83', bd=0, relief='flat', width=230, height=35)
                self.ed.place(x=170, y=0)
                self.lab = Label(self.ed, text='EDIT BOOK DETAILS', bg='#223c83', fg='#fff',font=('Georgia', 14, 'bold'))
                self.lab.place(x=9, y=5)
                self.label3 = Label(self.fm1, text='Book ID', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
                self.label3.place(x=85, y=65)
                self.entry = Entry(self.fm1, width=30, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.entry.place(x=188, y=65)
                self.button7 = Button(self.fm1, text='SEARCH', bg='#223c83', fg='#fff', width=8, font=('Georgia', 14,
'bold'), command=self.searchedit, relief='flat', activebackground='#223c83', activeforeground='#1c1c1b')
                self.button7.place(x=85, y=125)
                self.backbt = Button(self.ffm, width=60, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,
relief='flat',command=self.cur)
                self.backbt.place(x=0, y=0)
                self.log = PhotoImage(file='Images/back.png')
                self.backbt.config(image=self.log, compound=LEFT)
                self.small_log = self.log.subsample(2, 2)
                self.backbt.config(image=self.small_log)
            def searchedit(self):
                self.datas = self.entry.get()
                cursor = dbstore.cursor(buffered=True)
                cursor.execute("SELECT * FROM Books WHERE BookID = '" + self.datas + "'")
                dbstore.commit()
                self.val = cursor.fetchone()
                if self.val != None:
                    self.edcat = Tk()
                    self.edcat.title("Library System")
                    self.edcat.geometry("420x360+600+230")
                    self.edcat.configure(bg='#ffe8ec')
                    self.edcat.iconbitmap("Images/library.ico")
                    self.fc = Frame(self.edcat, bg='#223c83', width=160, height=35)
                    self.fc.place(x=100, y=10)
                    self.lab = Label(self.fc, bg='#223c83', fg='#fff', text='EDIT BOOK', font=('Georgia', 18, 'bold'))
                    self.lab.place(x=3, y=3)
                    self.labid = Label(self.edcat, bg='#ffe8ec', fg='black', text='Book ID:',  font=('Georgia', 14, 'bold'))
```

```
                self.labid.place(x=30, y=60)
                self.labti = Label(self.edcat, bg='#ffe8ec', fg='black', text='Title:',  font=('Georgia', 14, 'bold'))
                self.labti.place(x=30, y=100)
                self.labaut = Label(self.edcat, bg='#ffe8ec', fg='black', text='Author:',   font=('Georgia', 14, 'bold'))
                self.labaut.place(x=30, y=140)
                self.labed = Label(self.edcat, bg='#ffe8ec', fg='black', text='Publisher:',  font=('Georgia', 12, 'bold'))
                self.labed.place(x=30, y=180)
                self.labpr = Label(self.edcat, bg='#ffe8ec', fg='black', text='Price:',  font=('Georgia', 14, 'bold'))
                self.labpr.place(x=30, y=220)
                self.en1 = Entry(self.edcat, width=20, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.en1.place(x=130, y=60)
                self.en2 = Entry(self.edcat, width=20, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.en2.place(x=130, y=100)
                self.en3 = Entry(self.edcat, width=20, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.en3.place(x=130, y=140)
                self.en4 = Entry(self.edcat, width=20, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.en4.place(x=130, y=180)
                self.en5 = Entry(self.edcat, width=20, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.en5.place(x=130, y=220)
                self.butt = Button(self.edcat, text='SUBMIT', bg='#223c83', fg='#fff', width=8,  font=('Georgia', 16,
'bold'), command=self.savedit, relief='flat')
                self.butt.place(x=30, y=273)
                self.en1.insert(0, self.val[0])
                self.en2.insert(0, self.val[1])
                self.en3.insert(0, self.val[2])
                self.en4.insert(0, self.val[3])
                self.en5.insert(0, self.val[4])
                self.edcat.mainloop()
            else:
                messagebox.showerror('Invalid Entry', "This Book doesn't exists!")
                self.entry.delete(0, END)
        def savedit(self):
            self.id = self.en1.get()
            self.ti = self.en2.get()
            self.au = self.en3.get()
            self.ed = self.en4.get()
            self.pi = self.en5.get()
            if (self.id and self.ti and self.au and self.ed and self.pi):
                cursor = dbstore.cursor(buffered=True)
                cursor.execute(
                    "UPDATE Books SET BookID='" + self.id + "', Title='" + self.ti + "',Author='" + self.au +
"',Publisher='" + self.ed + "',Price='" + self.pi + "' WHERE BookID='" + self.datas + "'")
                dbstore.commit()
                messagebox.showinfo('Changes Saved', 'Data has been updated successfully!')
                self.edcat.destroy()
```

```
                self.entry.delete(0, END)
            else:
                messagebox.showerror('Error', 'Enter Valid Details')
                self.entry.delete(0, END)
        obj = editing()
        obj.edbooks()
    def returnbook(self):
        class retu(main):
            def __init__(self):
                self.frame = Frame(root, bd=0, relief='flat', bg='#ffe8ec', width=900, height=390)
                self.frame.place(x=0, y=110)
                self.f1 = Frame(self.frame, bg='#ffe8ec', width=500, height=250, bd=5, relief='flat')
                self.f1.place(x=200, y=15)
                self.ed = Frame(self.f1, bg='#223c83', bd=0, relief='flat', width=200, height=35)
                self.ed.place(x=170, y=0)
                self.lac = Label(self.ed, text='RETURN BOOKS ', bg='#223c83', fg='#fff', font=('Georgia', 16, 'bold'))
                self.lac.place(x=10, y=5)
                self.label8 = Label(self.f1, text='Student ID', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
                self.label8.place(x=85, y=65)
                self.entry4 = Entry(self.f1, width=30, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.entry4.place(x=200, y=65)
                self.label9 = Label(self.f1, text='Book ID', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
                self.label9.place(x=85, y=120)
                self.entry5 = Entry(self.f1, width=30, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
                self.entry5.place(x=200, y=120)
                self.button9 = Button(self.f1, text='RETURN', bg='#223c83', fg='#fff', width=14, height=0,
font=('Georgia', 12, 'bold'), command=self.retbook, activebackground="#000", activeforeground="#223c83")
                self.button9.place(x=85, y=170)
                self.backbt = Button(self.frame, width=60, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,relief='flat',
command=self.cur)
                self.backbt.place(x=0, y=0)
                self.log = PhotoImage(file='Images/back.png')
                self.backbt.config(image=self.log, compound=LEFT)
                self.small_log = self.log.subsample(2, 2)
                self.backbt.config(image=self.small_log)
            def retbook(self):
                self.entry = self.entry4.get()
                self.bookid = self.entry5.get()
                cursor = dbstore.cursor(buffered=True)
                cursor.execute("SELECT * FROM Books WHERE BookID='" + self.bookid + "'")
                dbstore.commit()
                self.booksdata = cursor.fetchone()
                self.issuedate = self.booksdata[7]
                cursor = dbstudents.cursor(buffered=True)
                cursor.execute("SELECT * FROM Students WHERE ID='" + self.entry + "'")
```

```
            dbstudents.commit()
            self.data = cursor.fetchone()
            print(self.data)
            if self.data != None:
                if (int(self.data[5]) >= 1):
                    cursor = dbstudents.cursor(buffered=True)
                    cursor.execute(
                        "UPDATE Students SET NoBook = " + str(self.data[5] - 1) + " WHERE ID='" + self.entry + "'")
                    dbstudents.commit()
                    cursor = dbstore.cursor(buffered=True)
                    cursor.execute(
                        "UPDATE Books SET Issue='', ID='', IssueDate='' WHERE BookID='" + self.bookid + "'")
                    dbstore.commit()
                    messagebox.showinfo("Book Issue Date", f"The Book Was issued on {self.issuedate}")
                    messagebox.showinfo("Success", "Books returned successfully")
                    self.entry4.delete(0, END)
                    self.entry5.delete(0, END)
                else:
                    messagebox.showwarning("No Books Found", "This student does not have any book issued!")
                    self.entry4.delete(0, END)
                    self.entry5.delete(0, END)
            else:
                messagebox.showerror("Invalid Student ID", "This student doesn't exist!")
                self.entry4.delete(0, END)
                self.entry5.delete(0, END)
    object = retu()
def delete(self):
    class dele(main):
        def deletebooks(self):
            self.ff = Frame(root, bg='#ffe8ec', width=900, height=390)
            self.ff.place(x=0, y=110)
            self.f1 = Frame(self.ff, bg='#ffe8ec', width=500, height=200, bd=5, relief='flat')
            self.f1.place(x=200, y=15)
            self.ed = Frame(self.f1, bg='#223c83', bd=0, relief='flat', width=150, height=30)
            self.ed.place(x=150, y=0)
            self.lac = Label(self.ed, text='DELETE BOOKS ', bg='#223c83', fg='#fff', font=('Georgia', 12, 'bold'))
            self.lac.place(x=7, y=3)
            self.label8 = Label(self.f1, text='Book ID', bg='#ffe8ec', fg='black', font=('Georgia', 11, 'bold'))
            self.label8.place(x=85, y=65)
            self.entry4 = Entry(self.f1, width=30, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
            self.entry4.place(x=188, y=65)
            self.button9 = Button(self.f1, text='DELETE', bg='#223c83', fg='#fff', width=8, font=('Georgia', 12, 'bold'),
command=self.deldata, relief='flat', activebackground='black', activeforeground='#223c83')
            self.button9.place(x=85, y=120)
```

```
            self.backbt = Button(self.ff, width=60, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,
relief='flat',command=self.cur)
            self.backbt.place(x=0, y=0)
            self.log = PhotoImage(file='Images/back.png')
            self.backbt.config(image=self.log, compound=LEFT)
            self.small_log = self.log.subsample(2, 2)
            self.backbt.config(image=self.small_log)
        def deldata(self):
            self.a = self.entry4.get()
            cursor = dbstore.cursor(buffered=True)
            cursorv = dbstore.cursor(buffered=True)
            cursorv.execute("SELECT * FROM BOOKS WHERE BookID='" + self.a + "'")
            dbstore.commit()
            self.validation = cursorv.fetchone()
            if (self.validation != None):
                cursor.execute("DELETE FROM Books WHERE BookID='" + self.a + "'")
                dbstore.commit()
                messagebox.showinfo('Succesful', 'The book is successfully removed from the store!')
                self.entry4.delete(0, END)
            else:
                messagebox.showerror('Invalid Operation', 'This book does not exist!')
                self.entry4.delete(0, END)
    occ = dele()
    occ.deletebooks()
  def search(self):
    class demt(main):
        def delmdata(self):
            self.fc = Frame(root, bg='#ffe8ec', width=900, height=390)
            self.fc.place(x=0, y=110)
            self.fc1 = Frame(self.fc, bg='#ffe8ec', width=500, height=200, bd=5, relief='flat')
            self.fc1.place(x=200, y=15)
            self.edm = Frame(self.fc1, bg='#223c83', bd=0, relief='flat', width=150, height=35)
            self.edm.place(x=140, y=0)
            self.lac = Label(self.edm, text='SEARCH BOOKS ', bg='#223c83', fg='#fff', font=('Georgia', 12, 'bold'))
            self.lac.place(x=8, y=5)
            self.label8 = Label(self.fc1, text='Book ID', bg='#ffe8ec', fg='black', font=('Georgia', 11, 'bold'))
            self.label8.place(x=85, y=65)
            self.entryl = Entry(self.fc1, width=30, bd=1, relief='groove', font=('Georgia', 14, 'bold'))
            self.entryl.place(x=188, y=65)
            self.butto = Button(self.fc1, text='SEARCH', bg='#223c83', fg='#fff', width=8,font=('Georgia', 12, 'bold'),
command=self.srch, relief='flat',activebackground='black', activeforeground='#b76e79')
            self.butto.place(x=85, y=120)
            self.backbt = Button(self.fc, width=60, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,
relief='flat',command=self.cur)
            self.backbt.place(x=0, y=0)
```

```
        self.log = PhotoImage(file='Images/back.png')
        self.backbt.config(image=self.log, compound=LEFT)
        self.small_log = self.log.subsample(2, 2)
        self.backbt.config(image=self.small_log)
    def srch(self):
        self.emp = self.entryl.get()
        cursor = dbstore.cursor(buffered=True)
        cursor.execute("SELECT * FROM Books WHERE BookID='" + self.emp + "'")
        dbstore.commit()
        self.srval = cursor.fetchone()
        if self.srval != None:
            self.top = Tk()
            self.top.title("Library System")
            self.top.iconbitmap("Images/library.ico")
            self.top.geometry("450x200+335+250")
            self.top.resizable(0, 0)
            self.top.configure(bg='#ffe8ec')
            if self.srval[5] == "Issued":
                self.frm = Frame(self.top, bg='#223c83', width=150, height=35)
                self.frm.place(x=130, y=10)
                self.mnlb = Label(self.frm, bg='#223c83', fg='#fff', text="Unavailable",font=('Georgia', 14, 'bold'))
                self.mnlb.place(x=9, y=5)
            else:
                self.frm = Frame(self.top, bg='#223c83', width=120, height=35)
                self.frm.place(x=150, y=10)
                self.mnlb = Label(self.frm, bg='#223c83', fg='#fff', text="Available",font=('Georgia', 14, 'bold'))
                self.mnlb.place(x=9, y=5)
            self.lb1 = Label(self.top, text='Title: ', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
            self.lb1.place(x=85, y=70)
            self.lb2 = Label(self.top, text=self.srval[1], bg='#ffe8ec', fg='black',font=('Georgia', 14, 'bold'))
            self.lb2.place(x=195, y=70)
            self.lb3 = Label(self.top, text='Author: ', bg='#ffe8ec', fg='black', font=('Georgia', 14, 'bold'))
            self.lb3.place(x=85, y=110)
            self.lb4 = Label(self.top, text=self.srval[2], bg='#ffe8ec', fg='black',font=('Georgia', 14, 'bold'))
            self.lb4.place(x=195, y=110)
            self.lb5 = Label(self.top, text='Publisher: ', bg='#ffe8ec', fg='black',font=('Georgia', 14, 'bold'))
            self.lb5.place(x=85, y=150)
            self.lb6 = Label(self.top, text=self.srval[3], bg='#ffe8ec', fg='black',font=('Georgia', 14, 'bold'))
            self.lb6.place(x=195, y=150)
            self.entryl.delete(0, END)
        else:
            messagebox.showwarning('Invalid Data', 'This book does not exists!')
            self.entryl.delete(0, END)
object = demt()
object.delmdata()
```

```
    def show(self):
        class test(main):
            def __init__(self):
                self.fc = Frame(root, bg='#ffe8ec', width=900, height=390)
                self.fc.place(x=0, y=110)
                self.popframe = Frame(self.fc, width=220, height=30, bg='#223c83')
                self.popframe.place(x=360, y=0)
                self.lbn = Label(self.popframe, bg='#223c83', text='BOOKS INFORMATION', fg='#fff',font=('Georgia',
12, 'bold'))
                self.lbn.place(x=8, y=4)
                self.button1 = Button(self.fc, text='Print Report', bg='#223c83', fg='#fff', width=10, height=0,
font=('Georgia', 11, 'bold'), activebackground='#223c83', activeforeground='#adefd1', command=self.printrpt)
                self.button1.place(x=750, y=0)
                self.backbt = Button(self.fc, width=30, bg='#ffe8ec', activebackground='#ffe8ec', bd=0,
relief='flat',command=self.cur)
                self.backbt.place(x=0, y=0)
                self.log = PhotoImage(file='Images/back.png')
                self.backbt.config(image=self.log, compound=LEFT)
                self.small_log = self.log.subsample(3, 3)
                self.backbt.config(image=self.small_log)
                self.table_frame = Frame(self.fc, bg='#ffe8ec', bd=1, relief='flat')
                self.table_frame.place(x=0, y=30, width=900, height=360)
                self.scroll_x = Scrollbar(self.table_frame, orient=HORIZONTAL)
                self.scroll_y = Scrollbar(self.table_frame, orient=VERTICAL)
                self.book_table = ttk.Treeview(self.table_frame,columns=("Book ID", "Title", "Author", "Publisher",
"Price"),xscrollcommand=self.scroll_x.set, yscrollcommand=self.scroll_y.set)
                self.scroll_x.pack(side=BOTTOM, fill=X)
                self.scroll_y.pack(side=RIGHT, fill=Y)
                self.scroll_x.config(command=self.book_table.xview)
                self.scroll_y.config(command=self.book_table.yview)
                self.book_table.heading("Book ID", text="Book ID")
                self.book_table.heading("Title", text="Title")
                self.book_table.heading("Author", text="Author")
                self.book_table.heading("Publisher", text="Publisher")
                self.book_table.heading("Price", text="Price")
                self.book_table['show'] = 'headings'
                self.book_table.column("Book ID", width=200)
                self.book_table.column("Title", width=200)
                self.book_table.column("Author", width=200)
                self.book_table.column("Publisher", width=120)
                self.book_table.column("Price", width=110)
                self.book_table.pack(fill=BOTH, expand=1)
                self.fetch_data()
            def printrpt(self):
                import csv
```

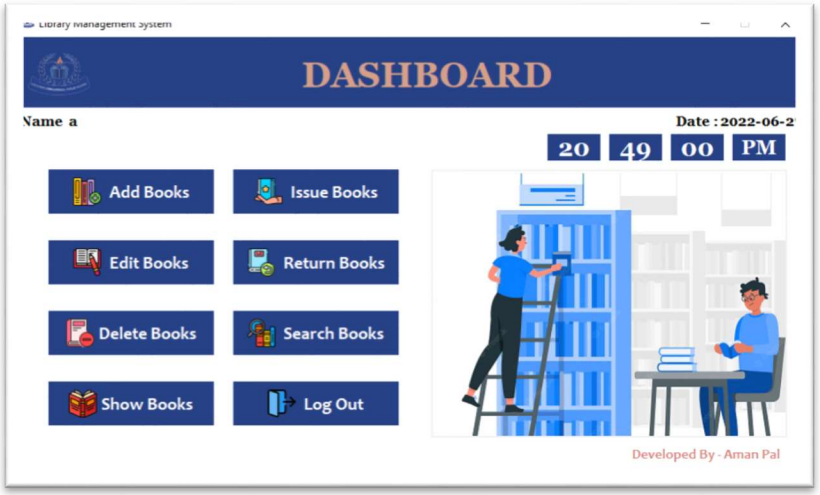
```
            cursor = dbstore.cursor(buffered=True)
            cursor.execute("SELECT * FROM Books")
            self.rows = cursor.fetchall()
            fields = ['BookID', 'Title', 'Author', 'Publisher', 'Price', 'Availability', 'ID', 'Issued By']
            rows = []
            for row in self.rows:
                l = list(row)
                self.stid = l[6]
                if self.stid != '':
                    cursor = dbstudents.cursor(buffered=True)
                    cursor.execute("SELECT * FROM Students WHERE ID='" + str(self.stid) + "'")
                    dbstudents.commit()
                    self.data = cursor.fetchone()
                    if self.data != None:
                        l.append(self.data[1])
                    else:
                        l.append("")
                rows.append(l)
            filename = "report.csv"
            with open(filename, 'w', newline='') as csvfile:
                csvwriter = csv.writer(csvfile)
                csvwriter.writerow(fields)
                csvwriter.writerows(rows)
        def fetch_data(self):
            cursor = dbstore.cursor(buffered=True)
            cursor.execute("SELECT * FROM Books")
            self.rows = cursor.fetchall()
            if len(self.rows) != 0:
                for self.row in self.rows:
                    self.book_table.insert('', END, values=self.row)
            dbstore.commit()
    oc = test()
def mainclear(self):
    self.e1.delete(0, END)
    self.e2.delete(0, END)
def code(self):
    self.fm = Frame(root, height=500, width=900, bg='white')
    self.fm.place(x=0, y=0)
    self.canvas = Canvas(self.fm, height=500, width=900, bg='#d39c86')
    self.canvas.place(x=0, y=0)
    self.photo = PhotoImage(file=r"Images/background.png")
    self.canvas.create_image(0, 0, image=self.photo, anchor=NW)
    self.fm1 = Frame(self.canvas, height=260, width=400, bg='#d39c86', relief='sunken')
    self.fm1.place(x=200, y=120)
    self.b1 = Label(self.fm1, text='User ID', bg="#d39c86", font=('Georgia', 14, 'bold'), fg='#223c83')
```

```
        self.b1.place(x=20, y=42)
        self.e1 = Entry(self.fm1, width=18, font=('Georgia', 14, 'bold'), bd=1, relief='groove')
        self.e1.place(x=130, y=40)
        self.lb2 = Label(self.fm1, text='Password', bg="#d39c86", font=('Georgia', 14, 'bold'), fg='#223c83')
        self.lb2.place(x=20, y=100)
        self.e2 = Entry(self.fm1, width=18, show='*', font=('Georgia', 14, 'bold'), bd=1, relief='groove')
        self.e2.place(x=130, y=100)
        self.btn1 = Button(self.fm1, text='  Login', fg='#fff', bg='#223c83', width=100, font=('Georgia', 11,
'bold'),activebackground='black', activeforeground='white', command=self.login, bd=3, relief='flat',cursor='hand2')
        self.btn1.place(x=65, y=160)
        self.logo = PhotoImage(file=r"Images/login.png")
        self.btn1.config(image=self.logo, compound=LEFT)
        self.small_logo = self.logo.subsample(1, 1)
        self.btn1.config(image=self.small_logo)
        self.btn2 = Button(self.fm1, text='  Clear', fg='#fff', bg='#223c83', width=100, font=('Georgia', 11,
'bold'),activebackground='black', activeforeground='white', bd=3, relief='flat',
cursor='hand2',command=self.mainclear)
        self.btn2.place(x=195, y=160)
        self.log = PhotoImage(file=r"Images/delete.png")
        self.btn2.config(image=self.log, compound=LEFT)
        self.small_log = self.log.subsample(1, 1)
        self.btn2.config(image=self.small_log)
        self.forgot = Label(self.fm1, text='Forgot Password?', fg='#223c83', bg='#d39c86',
activeforeground='black',font=('cursive', 14, 'bold'))
        self.forgot.place(x=100, y=220)
        self.forgot.bind("<Button>", self.mouseClick)
        root.mainloop()
    def mouseClick(self, event):
        self.rog = Tk()
        self.rog.title("Change password")
        self.rog.geometry("550x400+300+210")
        self.rog.iconbitmap("Images/library.ico")
        self.rog.resizable(0, 0)
        self.rog.configure(bg='#d39c86')
        self.framerog = Frame(self.rog, width=180, height=35, bg="#223c83")
        self.framerog.place(x=95, y=15)
        self.label = Label(self.framerog, text="SET NEW PASSWORD", bg='#223c83', fg='white',font=("Calibri", 14,
'bold'))
        self.label.place(x=5, y=4)
        self.user = Label(self.rog, text='User ID', bg='#d39c86', fg='#223c83', font=("Times New Roman", 14, 'bold'))
        self.user.place(x=40, y=95)
        self.ef1 = Entry(self.rog, width=16, font=('Georgia', 14, 'bold'))
        self.ef1.place(x=170, y=95)
```

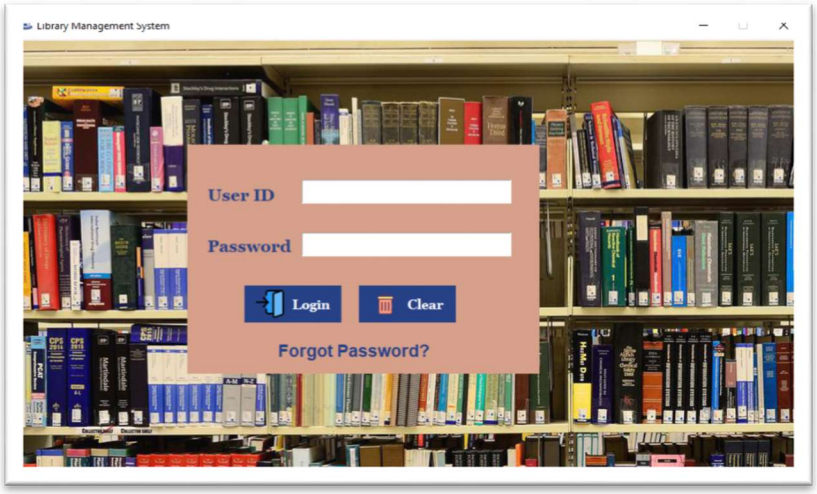
```
        self.sendotpbtn = Button(self.rog, text="Send OTP", fg='#fff', bg='#223c83', width=8,font=('Georgia', 10,
'bold'), activebackground='#223c83', activeforeground='#d39c86',bd=3, relief='flat', cursor='hand2',
command=self.sendotp)
        self.sendotpbtn.place(x=400, y=95)
    def sendotp(self):
        self.a = self.ef1.get()
        cursor = db.cursor(buffered=True)
        cursor.execute("SELECT * FROM UserLogin WHERE UserID='" + self.a + "'")
        db.commit()
        self.data = cursor.fetchone()
        if self.data != None:
            password = 'euhvogkffgvticqb'
            message = MIMEMultipart("alternative")
            message["Subject"] = "LIPS Library Management System"
            message["From"] = "com.amanpal@gmail.com"
            message["To"] = self.data[2]
            self.otp = random.randint(1000, 9999)
            text = f"""Your Password Reset Code is {self.otp}"""
            message.attach(MIMEText(text, "plain"))
            context = ssl.create_default_context()
            with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
                server.login('com.amanpal@gmail.com', password)
                server.sendmail('com.amanpal@gmail.com', self.data[2], message.as_string())
            self.enterotp = Label(self.rog, text='Enter OTP', bg='#d39c86', fg='#223c83',font=("Times New Roman",
14, 'bold'))
            self.enterotp.place(x=40, y=130)
            self.otpent = Entry(self.rog, width=16, font=('Georgia', 14, 'bold'))
            self.otpent.place(x=170, y=130)
            self.sendotpbtn = Button(self.rog, text="Submit", fg='#fff', bg='#223c83', width=8,font=('Georgia', 10,
'bold'), activebackground='#223c83',activeforeground='#d39c86', bd=3, relief='flat',
cursor='hand2',command=self.submitotp)
            self.sendotpbtn.place(x=400, y=130)
        else:
            messagebox.showerror("ERROR", "UserID doesn't exist")
    def submitotp(self):
        self.givenotp = int(self.otpent.get())
        if self.givenotp == self.otp:
            self.user = Label(self.rog, text='New Password', bg='#d39c86', fg='#223c83',font=("Times New Roman",
14, 'bold'))
            self.user.place(x=40, y=165)
            self.user = Label(self.rog, text='Confirm Password', bg='#d39c86', fg='#223c83',font=("Times New
Roman", 14, 'bold'))
            self.user.place(x=40, y=200)
            self.ef1 = Entry(self.rog, width=16, font=('Georgia', 14, 'bold'))
            self.ef1.place(x=170, y=200)
```

```
            self.ef2 = Entry(self.rog, width=16, font=('Georgia', 14, 'bold'))
            self.ef2.place(x=170, y=165)
            self.btn1 = Button(self.rog, text='Submit', fg='#fff', bg='#223c83', width=6, font=('Georgia', 14,
'bold'),activebackground='#223c83', activeforeground='#d39c86', bd=3, relief='flat',cursor='hand2',
command=self.chan_pas)
            self.btn1.place(x=400, y=180)
        else:
            messagebox.showinfo("Error", "OTP is invalid")
    def chan_pas(self):
        self.a = self.ef1.get()
        self.b = self.ef2.get()
        cursor = db.cursor(buffered=True)
        cursor.execute("SELECT * FROM UserLogin WHERE UserID='" + self.a + "'")
        db.commit()
        self.data = cursor.fetchone()
        if self.data != None:
            cursor = conn.cursor(buffered=True)
            cursor.execute("UPDATE UserLogin SET Password='" + hashlib.md5(
                self.b.encode()).hexdigest() + "' WHERE UserID='" + self.a + "'")
            conn.commit()
            messagebox.showinfo("SUCCESSFUL", "Your Password is changed")
            self.rog.destroy()
        else:
            messagebox.showerror("ERROR", "UserID doesn't exist")
            self.rog.destroy()
        self.rog.mainloop()
obj = main()
obj.code()
```

Output



Dashboard



Login

BIBLIOGRAPHY

To develop this project many references were used:

1. COMPUTER SCIENCE TEXTBOOK CLASS 12: SUMITA ARORA
2. https://www.google.com
3. https://www.python.org.in
4. https://www.mysql.org
5. https://www.tutorialaicsip.com
6. https://geeksforgeeks.com/
7. https://pypi.org/