**Task-4: Implementation**

AdminPasswordChange.py

from tkinter import \*  
import pymysql  
from pymysql import Error  
from tkinter import messagebox  
class Set(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**"Admin Update New Password"**)  
 self.maxsize(500,300)  
 self.minsize(500,300)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 def verify():  
 if len(q.get()) == 0:  
 messagebox.showinfo(**"Error"**,**"Enter Your Valid User Id"**)  
 elif len(m.get()) < 5 and len(n.get()) < 5 and len(p.get()) < 5:  
 messagebox.showinfo(**"Error"**,**"Enter a Valid Password"**)  
 elif n.get() != p.get():  
 messagebox.showinfo(**"Error"**,**"Passwords are Mismatched.!"**)  
 else:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"Select password from admin where id = %s"**,(q.get()))  
 tValue = self.pointTo.fetchone()  
 if tValue:  
 if str(tValue[0]) == m.get():  
 self.pointTo.execute(**"UPDATE admin SET password = %s WHERE id = %s"**,(n.get(),q.get()))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**"Successful"**,**"Password Updated successfully"**)  
 else:  
 messagebox.showinfo(**"Error"**,**"Old Password Does not Match"**)  
 else:  
 messagebox.showinfo(**"Error"**, **"The User Doesn't Existed"**)  
 except Exception as ex:  
 print(ex)  
 m.set(**""**)  
 n.set(**""**)  
 p.set(**""**)  
 q.set(**""**)  
 userid = Label(self, text=**"User Id"**, font=(**'arial'**, 13, **'bold'**)).place(x=40, y=50)  
 q = StringVar()  
 Uentry= Entry(self, textvariable=q, width=30).place(x=250, y=55)  
 oldpassword = Label(self,text=**"Old Password"**,font=(**'arial'**, 13, **'bold'**)).place(x=40,y=100)  
 m=StringVar()  
 oldpasswordvalue = Entry(self,show=**'\*'**,textvariable=m,width = 30).place(x=250,y=105)  
 newpassword = Label(self,text=**"New Password"**,font=(**'arial'**, 13, **'bold'**)).place(x=40,y=150)  
 n=StringVar()  
 newpasswordvalue = Entry(self,show=**'\*'**,textvariable=n,width = 30).place(x=250,y=155)  
 reenterPassword = Label(self,text=**"Re-Enter password"**,font=(**'arial'**, 13, **'bold'**)).place(x=40,y=200)  
 p=StringVar()  
 reenterPasswordValue = Entry(self,show=**'\*'**,textvariable =p,width = 30).place(x=250,y=205)  
 updateButton=Button(self,text=**"Update"**,width=15,command = verify).place(x=280,y=255)  
Set().mainloop()

BookIssue.py

from tkinter import \*  
from datetime import timedelta, date  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
import sys  
py = sys.executable  
class issue(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**'LIBRARY SYSTEM MANAGEMENT SYSTEM'**)  
 self.maxsize(420, 280)  
  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 p = StringVar()  
 q = StringVar()  
  
 *# verifying input* def isb():  
 if len(p.get()) == 0 or len(q.get()) == 0:  
 messagebox.showinfo(**"Error"**, **"Please Enter both Book Id/Student ID's"**)  
 else:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"Select Availiability from books where Book\_Id = %s"**, (p.get()))  
 tValue = self.pointTo.fetchone()  
 try:  
 if str(tValue[0]) == **'0'**:  
 messagebox.showinfo(**"Oop's"**, **"The Given Book is Already Issued"**)  
 else:  
 self.pointTo.execute(**"Select Fine from students where Student\_Id = %s"**, (q.get()))  
 findfineValue = list(self.pointTo.fetchone())  
 self.pointTo.execute(**"Select Books\_Issued from students where Student\_Id = %s"**, (q.get()))  
 bookissueList = list(self.pointTo.fetchone())  
 if bookissueList[0] < 3:  
 if findfineValue[0] > 100:  
 messagebox.showerror(**'Oops'**, **'Cannot Issue.Please Pay the Fine'**)  
 elif findfineValue [0] == 0:  
 Date\_req = date.today() + timedelta(days=3)  
 todayDate = date.today()  
 print(Date\_req)  
 self.pointTo.execute(**"INSERT INTO issue VALUES (%s,%s,%s,%s)"**,(p.get(), q.get(), todayDate, Date\_req))  
 self.pointTo.execute(**"UPDATE books set Availiability=0 where Book\_Id = %s"**,(p.get()))  
 bookissueList[0] =bookissueList[0] + 1  
  
 self.pointTo.execute(**"Update students set Books\_Issued = %s where Student\_Id = %s"**, (bookissueList[0], q.get()))  
  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Save'**, **'Successfully Issued'**)  
 conf = messagebox.askyesno(**"Confirm"**, **"Do you want to issue another book%s"**)  
 if conf:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'BookIssue.py'**))  
 else:  
 self.destroy()  
 elif findfineValue [0] > 0:  
 Confirm = messagebox.askyesno(**'Confirm'**,**'Are you sure you want to issue.There is a fine'**)  
 if Confirm:  
 Date\_req = date.today() + timedelta(days=3)  
 todayDate = date.today()  
 self.pointTo.execute(**"INSERT INTO issue VALUES (%s,%s,%s,%s)"**, (p.get(), q.get(), todayDate, Date\_req))  
 self.pointTo.execute(**"UPDATE books set Availiability=0 where Book\_Id = %s"**, (p.get()))  
 bookissueList[0] = bookissueList[0] + 1  
 self.pointTo.execute(**"Update students set Books\_Issued = %s where Student\_Id = %s"**,(bookissueList[0], q.get()))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Save'**, **'Successfully Issued'**)  
 conf = messagebox.askyesno(**"Confirm"**, **"Do you want to issue another book%s"**)  
 if conf:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'BookIssue.py'**))  
 else:  
 self.destroy()  
 else:  
 messagebox.showinfo(**'Oops'**, **'Not Issued'**)  
 elif findfineValue[0] > 100:  
 messagebox.showerror(**'Oops'**, **'Cannot Issue.Please Pay the Fine'**)  
 else:  
 messagebox.showerror(**"Can't Issue"**, **"Maximum number of books already issued"**)  
 except TypeError:  
 messagebox.showinfo(**"Oop's"**, **"Either BookID or StudentId Not Available"**)  
 except Exception as ex:  
 print(ex)  
  
 *# label and input box* Label(self, text=**'Issuing New Book'**, font=(**'Arial Black'**, 20)).place(x=85, y=40)  
 Label(self, text=**'Book ID:'**, font=(**'Arial'**, 14), fg=**'black'**).place(x=45, y=100)  
 Entry(self, textvariable=p, width=40).place(x=160, y=106)  
 Label(self, text=**'Student ID:'**, font=(**'Arial'**, 14), fg=**'black'**).place(x=40, y=150)  
 Entry(self, textvariable=q, width=40).place(x=160, y=158)  
 Button(self, text=**"ISSUE"**, width=20, command=isb).place(x=200, y=200)  
  
  
issue().mainloop()

BookRemove.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
import sys  
  
py = sys.executable  
  
class rb(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(380,260)  
 self.title(**"Remove Book Details"**)  
 self.canvas = Canvas(width=500, height=500)  
 self.canvas.pack()  
 m= StringVar()  
  
 def aaa():  
 if len(m.get()) == 0:  
 messagebox.showerror(**"Error"**,**"Please Enter The Book Id"**)  
 else:  
 cnfrmMsg= messagebox.askyesno(**'Remove Book'**, **'Are You Sure You Want To Delete The Book'**)  
 if cnfrmMsg:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"DELETE FROM books WHERE Book\_Id = %s"**,(m.get()))  
 messagebox.showinfo(**'Remove'**, **'Successfully Removed'**)  
 self.conn.commit()  
 self.conn.close()  
 dialogMsg = messagebox.askyesno(**"Confirm"**,**"Do you want to remove another book"**)  
 if dialogMsg:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'BookRemove.py'**))  
 else:  
 self.destroy()  
 except Exception as ex:  
 print(ex)  
  
 lb = Label(self, text=**"-:Book Id:-"**,fg=**'red'**, font=(**'Comic Scan Ms'**, 22, **'bold'**))  
 lb.place(x=90, y=70)  
 e = Entry(self, textvariable=m, width=30).place(x=85, y=135)  
 bt = Button(self, text=**"Delete"**, width=20, command=aaa).place(x=100, y=170)  
  
rb().mainloop()

BookRenewal.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
from datetime import datetime,date  
import os  
import sys  
py = sys.executable  
class renew(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**"Book Renewal Form"**)  
 self.canvas = Canvas(width=500, height=417)  
 self.canvas.pack()  
 self.maxsize(450,300)  
 m = StringVar()  
 n = StringVar()  
 self.cal = 0  
  
 def days\_Gap(dateA, dateB):  
 dateA = datetime.strptime(dateA, **"%Y-%m-%d"**)  
 dateB = datetime.strptime(dateB, **"%Y-%m-%d"**)  
 return abs((dateB - dateA).days)  
  
 def qui():  
 if len(m.get()) == 0 or len(n.get()) == 0:  
 messagebox.showerror(**"Error"**,**"Please Enter both BookId/StudentId's"**)  
 else:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"Select BID from issue where BID = %s"**,(m.get()))  
 temp = self.pointTo.fetchone()  
 self.pointTo.execute(**"Select Fine from students where Student\_Id = %s"**, (n.get()))  
 fine = self.pointTo.fetchone()  
 self.pointTo.execute(**"Select Return\_date from issue where BID = %s and SID = %s"**, (m.get(), n.get()))  
 tValue = self.pointTo.fetchone()  
 if tValue:  
 currentDay = str(date.today())  
 newDay = str(tValue[0])  
 if currentDay < newDay:  
 messagebox.showinfo(**"Oops"**, **"Your Return Date Has not yet come"**)  
 else:  
 self.cal = days\_Gap(newDay, currentDay)  
 self.cal += int(fine[0])  
 if int(self.cal) >= 100:  
 messagebox.showinfo(**"Fine"**, **"Your Id is banned.Please pay the fine"**)  
 elif int(self.cal) > 0:  
 messagebox.showinfo(**'Warning'**,**'Please Return/Renew book Timely to avoid termination of id'**)  
 self.pointTo.execute(**"Update students set Fine = %s where Student\_Id = %s"**,(int(self.cal), n.get()))  
 self.pointTo.execute(**"UPDATE issue set Issue\_date = date('now') where BID =%s "**, (m.get()))  
 self.pointTo.execute(**"update issue set Return\_date = date('now','+15 days') where BID = %s"**,(m.get()))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Info'**, **'Successfully Renewed'**)  
 c = messagebox.askyesno(**"Confirm"**, **"Do you want to renew another book%s"**)  
 if c:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'BookRenewal.py'**))  
 else:  
 self.destroy()  
 else:  
 messagebox.showinfo(**"Oop's"**, **"The Book is not yet issued"**)  
 except TypeError:  
 messagebox.showerror(**"Error"**, **"Check The Texts"**)  
 except Exception as ex:  
 print(ex)  
  
 Label(self, text=**'Book Renewal Page'**, fg=**'red'**,font=(**'arial'**, 29, **'bold'**)).place(x=55, y=50)  
 Label(self, text=**'Book ID'**,font=(**'Comic Scan Ms'**, 11, **'bold'**)).place(x=30, y=150)  
 Entry(self, textvariable=m, width=40).place(x=170, y=155)  
 Label(self, text=**"Student Id"**,font=(**'Comic Scan Ms'**, 11, **'bold'**)).place(x=30, y=200)  
 Entry(self, textvariable=n, width=40).place(x=170, y=205)  
 Button(self, text=**"Renew"**, width=25, command=qui).place(x=180, y=240)  
  
renew().mainloop()

BookReturn.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os,sys  
from datetime import datetime,date  
pythonFile = sys.executable  
class ret(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**"Book Return Form"**)  
 self.maxsize(420,260)  
 self.canvas = Canvas(width=500, height=417)  
 self.canvas.pack()  
 self.cal = 0  
 m = StringVar()  
  
 def days\_Gap(dateA, dateB):  
 if dateB <= dateA:  
 return 0  
 else:  
 dateA = datetime.strptime(dateA, **"%Y-%m-%d"**)  
 dateB = datetime.strptime(dateB, **"%Y-%m-%d"**)  
 return abs((dateB - dateA).days)  
  
 def qui():  
 if len(m.get()) == **'0'**:  
 messagebox.showerror(**"Error"**,**"Please Enter The Book Id"**)  
 else:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"Select SID from issue where BID = %s"**, (m.get()))  
 studentID = list(self.pointTo.fetchone())  
 self.pointTo.execute(**"Select Books\_Issued from students where Student\_Id = %s"**, (studentID[0]))  
 gstudentID = list(self.pointTo.fetchone())  
 gstudentID[0] = gstudentID[0] - 1  
 self.pointTo.execute(**"Select BID from issue where BID = %s"**,(m.get()))  
 tValue = self.pointTo.fetchone()  
 self.pointTo.execute(**"Select Fine from students where Student\_Id = %s"**, (studentID[0]))  
 fineCalculation = self.pointTo.fetchone()  
 self.pointTo.execute(**"Select Return\_date from issue where BID = %s and SID = %s"**, (m.get(), [0]))  
 tValue1 = self.pointTo.fetchone()  
 dayarea = str(date.today())  
 earlyarea = str(tValue[0])  
 self.cal = days\_Gap(earlyarea, dayarea)  
 self.cal += int(fineCalculation[0])  
 if dayarea <= earlyarea and int(self.cal) == 0:  
 *#self.pointTo.execute("DELETE FROM issue WHERE BID = %s", (m.get()))* self.pointTo.execute(**"update books set Availiability = 1 where Book\_Id = %s"**, (m.get()))  
 self.pointTo.execute(**"update students set Books\_Issued = %s where Student\_Id = %s"**, (gstudentID[0],studentID[0]))  
 self.pointTo.execute(**"insert into breturns values(%s,%s,%s)"**, (m.get(),studentID[0],dayarea))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Info'**, **'Successfully Returned'**)  
 d = messagebox.askyesno(**"Confirm"**, **"Return more books%s"**)  
 if d:  
 self.destroy()  
 os.system(**'%s %s'** % (pythonFile, **'BookReturn.py'**))  
 else:  
 self.destroy()  
 elif len(tValue) > 0:  
 if int(self.cal) > 0:  
 messagebox.showinfo(**'Warning'**,**'Please Return/Renew book Timely to avoid termination of id'**)  
 self.pointTo.execute(**"Update students set Fine = %s where Student\_Id = %s"**,(int(self.cal), studentID[0]))  
 self.pointTo.execute(**"DELETE FROM issue WHERE BID = %s"**, (m.get()))  
 self.pointTo.execute(**"update books set Availiability = 1 where Book\_Id = %s"**, (m.get()))  
 self.pointTo.execute(**"update students set Books\_Issued = %s where Student\_Id = %s"**, (gstudentID[0],studentID[0]))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Info'**, **'Succesfully Returned'**)  
 d = messagebox.askyesno(**"Confirm"**, **"Return more books%s"**)  
 if d:  
 self.destroy()  
 os.system(**'%s %s'** % (pythonFile, **'BookReturn.py'**))  
 else:  
 self.destroy()  
 else:  
 self.pointTo.execute(**"DELETE FROM issue WHERE BID = %s"**, (m.get()))  
 self.pointTo.execute(**"update books set Availiability = 1 where Book\_Id = %s"**, (m.get()))  
 self.pointTo.execute(**"update students set Books\_Issued = %s where Student\_Id = %s"**, (gstudentID[0],studentID[0]))  
 self.conn.commit()  
 self.conn.close()  
 messagebox.showinfo(**'Info'**, **'Successfully Returned'**)  
 d = messagebox.askyesno(**"Confirm"**, **"Return more books%s"**)  
 if d:  
 self.destroy()  
 os.system(**'%s %s'** % (pythonFile, **'BookReturn.py'**))  
 else:  
 self.destroy()  
 else:  
 messagebox.showinfo(**"Oop's"**, **"Book not yet issued till Now."**)  
 except Exception as ex:  
 print(ex)  
 Label(self, text=**'Book Return Page'**, fg=**'red'**,font=(**'arial'**, 28, **'bold'**)).place(x=40, y=50)  
 Label(self, text=**'Enter Book ID'**, font=(**'Comic Scan Ms'**, 14, **'bold'**)).place(x=20, y=120)  
 Entry(self, textvariable=m, width=40).place(x=165, y=124)  
 Button(self, text=**"Return"**, width=25, command=qui).place(x=180, y=180)  
ret().mainloop()

BookSearch.py

from tkinter import \*  
from tkinter import ttk  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
class Sea(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 mValue = StringVar()  
 nValue = StringVar()  
 self.title(**"Search Book"**)  
 self.maxsize(800,500)  
 self.minsize(800,500)  
 self.canvas = Canvas(width=800, height=500)  
 self.canvas.pack()  
  
 l1=Label(self,text=**"Book Search Page"**,font=(**"Arial"**,20,**'bold'**)).place(x=290,y=20)  
 l = Label(self, text=**"Search Type"**, font=(**"Arial"**, 14, **'bold'**)).place(x=60, y=96)  
 def insert(data):  
 self.listForm.delete(\*self.listForm.get\_children())  
 for row in data:  
 self.listForm.insert(**""**, **'end'**, text=row[0], values=(row[1], row[2],row[4], **'Available'** if row[3] == 1 else **'Unavailable'**))  
 def ge():  
 if (len(nValue.get())) == 0:  
 messagebox.showinfo(**'Error'**, **'First select a item'**)  
 elif (len(mValue.get())) == 0:  
 messagebox.showinfo(**'Error'**, **'Enter the '**+nValue.get())  
 elif nValue.get() == **'Book Id'**:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.conn.cursor()  
 self.pointTo.execute(**"DROP VIEW booksavialability"**)  
 self.pointTo.execute(**"CREATE VIEW booksavialability AS SELECT book\_name,Availiability FROM books"**)  
  
 self.studentlist = self.pointTo.fetchall()  
 self.pointTo.execute(**"Select b.Book\_Id,b.Book\_name,b.Author,b.Availiability,count(iu.sid) from books b,issue iu where iu.BID=%s and b.Book\_Id LIKE %s "**, (mValue.get(),**'%'**+mValue.get()+**'%'**))  
 self.pc = self.pointTo.fetchall()  
 if self.pc:  
 insert(self.pc)  
 else:  
 messagebox.showinfo(**"Oop's"**,**"Either Book Id is incorrect or it is not available"**)  
 except Exception as ex:  
 print(ex)  
 b=Button(self,text=**"Search"**,width=15,font=(**"Arial"**,10,**'bold'**),command=ge).place(x=460,y=148)  
 c=ttk.Combobox(self,textvariable=nValue,values=[**"Book Id"**],width=40,state=**"readonly"**).place(x = 180, y = 100)  
 en = Entry(self,textvariable=mValue,width=43).place(x=180,y=155)  
 la = Label(self, text=**"Enter Value"**, font=(**"Arial"**, 14, **'bold'**)).place(x=60, y=150)  
  
 def handle(event):  
 if self.listForm.identify\_region(event.x,event.y) == **"separator"**:  
 return **"break"** self.listForm = ttk.Treeview(self, height=13,columns=(**'Book Name'**, **'Book Author'**, **'Issued Students'**,**'Availability'**))  
 self.vsb = ttk.Scrollbar(self,orient=**"vertical"**,command=self.listForm.yview)  
 self.listForm.configure(yscrollcommand=self.vsb.set)  
 self.listForm.heading(**"#0"**, text=**'Book ID'**, anchor=**'center'**)  
 self.listForm.column(**"#0"**, width=80, anchor=**'center'**)  
 self.listForm.heading(**"Book Name"**, text=**'Book Name'**)  
 self.listForm.column(**"Book Name"**, width=150, anchor=**'center'**)  
 self.listForm.heading(**"Book Author"**, text=**'Book Author'**)  
 self.listForm.column(**"Book Author"**, width=150, anchor=**'center'**)  
  
 self.listForm.heading(**"Issued Students"**, text=**'Issued Students'**)  
 self.listForm.column(**"Issued Students"**, width=150, anchor=**'center'**)  
  
  
 self.listForm.heading(**"Availability"**, text=**'Availability'**)  
 self.listForm.column(**"Availability"**, width=150, anchor=**'center'**)  
 self.listForm.bind(**'<Button-1>'**, handle)  
 self.listForm.place(x=40, y=200)  
 self.vsb.place(x=763,y=200,height=287)  
 ttk.Style().configure(**"Treeview"**, font=(**'Times new Roman'**, 15))  
  
Sea().mainloop()

BookWiseIssues.py

from tkinter import ttk  
import tkinter as tk  
import pymysql  
from pymysql import Error  
def dbCconnection():  
 connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 connection.close()  
def ViewContent():  
 connection =pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 pointTo.execute(**"SELECT issue.BID, books.book\_name,books.Author,issue.Issue\_date,issue.Return\_date FROM issue INNER JOIN books ON issue.BID=books.book\_id"**)  
 rowData = pointTo.fetchall()  
 for rowData in rowData :  
 formatTable.insert(**""**,**"end"**,text = rowData [4], values = (rowData [0],rowData [1],rowData [2],rowData [3],rowData [4]))  
 connection.close()  
*# connect to the database*dbCconnection()  
root = tk.Tk()  
formatTable = ttk.Treeview(root, column=(**"c1"**,**"c2"**,**"c3"**,**"c4"**,**"c5"**), show=**'headings'**)  
formatTable.column(**"#1"**, anchor=tk.CENTER)  
formatTable.heading(**"#1"**, text=**"Book ID"**)  
formatTable.column(**"#2"**, anchor=tk.CENTER)  
formatTable.heading(**"#2"**, text=**"Book Name"**)  
formatTable.column(**"#3"**, anchor=tk.CENTER)  
formatTable.heading(**"#3"**, text=**"Author"**)  
formatTable.column(**"#4"**, anchor=tk.CENTER)  
formatTable.heading(**"#4"**, text=**"Issue Date"**)  
formatTable.column(**"#5"**, anchor=tk.CENTER)  
formatTable.heading(**"#5"**, text=**"Return Date"**)  
formatTable.pack()  
button1 = tk.Button(text=**"View"**, command=ViewContent)  
button1.pack(pady=10)  
root.mainloop()

ClassficationUser.py

from tkinter import ttk  
import tkinter as tk  
import pymysql  
from pymysql import Error  
def dbCconnection():  
 connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 connection.close()  
def ViewContent():  
 connection =pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 pointTo.execute(**"SELECT b.book\_id,b.book\_name,b.author,(SELECT COUNT(\*) FROM issue WHERE issue.bid = b.book\_id) AS issue\_count,(SELECT COUNT(\*) FROM breturns WHERE breturns.bid IN ( SELECT bid FROM breturns WHERE breturns.bid = b.book\_id ) ) AS breturns\_count FROM books AS b ORDER BY breturns\_count DESC, issue\_count DESC, author DESC"**)  
 rowData = pointTo.fetchall()  
 for rowData in rowData :  
 formatTable.insert(**""**,**"end"**,text = rowData [4], values = (rowData [0],rowData [1],rowData [2],rowData [3],rowData [4]))  
 connection.close()  
*# connect to the database*dbCconnection()  
root = tk.Tk()  
formatTable = ttk.Treeview(root, column=(**"c1"**,**"c2"**,**"c3"**,**"c4"**,**"c5"**), show=**'headings'**)  
formatTable.column(**"#1"**, anchor=tk.CENTER)  
formatTable.heading(**"#1"**, text=**"Book ID"**)  
formatTable.column(**"#2"**, anchor=tk.CENTER)  
formatTable.heading(**"#2"**, text=**"Book Name"**)  
formatTable.column(**"#3"**, anchor=tk.CENTER)  
formatTable.heading(**"#3"**, text=**"Author Name"**)  
formatTable.column(**"#4"**, anchor=tk.CENTER)  
formatTable.heading(**"#4"**, text=**"No.of Book Issues"**)  
formatTable.column(**"#5"**, anchor=tk.CENTER)  
formatTable.heading(**"#5"**, text=**"No.of Book Returns"**)  
formatTable.pack()  
button1 = tk.Button(text=**"View"**, command=ViewContent)  
button1.pack(pady=10)  
root.mainloop()

FineClearence.py

from tkinter import \*  
from tkinter import ttk  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import re  
import sys,os  
pytnCmd = sys.executable  
class Fine(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(440,250)  
 self.minsize(440, 250)  
 self.title(**"Fine Clearence Form"**)  
 self.canvas = Canvas(width=450, height=254)  
 self.canvas.pack()  
 *#creating variables* m = StringVar()  
 def clear():  
 if len(m.get()) == 0:  
 messagebox.showerror(**"Error"**,**"Please Enter The Student Id"**)  
 elif m.get().isdigit():  
 try:  
 self.connection =pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select Student\_Id from students"**)  
 stdntData = self.pointTo.fetchall()  
 liststdntData = list(stdntData)  
 if liststdntData:  
 for sid in liststdntData:  
  
 if m.get() in sid:  
 print(**"yes"**)  
 confirmMsg = messagebox.askyesno(**"Confirm"**,**"Are You Sure you want to clear the fine?"**)  
 if confirmMsg:  
  
 self.pointTo.execute(**"Update students set Fine=0 where Student\_Id= %s"**,(m.get()))  
 self.connection.commit()  
 self.connection.close()  
 messagebox.showinfo(**"Successful"**,**"All Fine Cleared"**)  
 d = messagebox.askyesno(**"Confirm"**,**"Do you want to clear another fine?"**)  
 if d:  
 self.destroy()  
 os.system(**'%s %s'**% (pytnCmd,**'FineClearence.py'**))  
 else:  
 self.destroy()  
 else:  
 messagebox.showinfo(**"Oops"**,**"The Given Student ID not found"**)  
 else:  
 messagebox.showerror(**"Error"**,**"Please Check The Student Id"**)  
 except Exception as ex:  
 print(ex)  
 else:  
 messagebox.showerror(**"Error"**,**"Please Check The Student Id"**)  
 Label(self,text=**"Enter Student Id"**, font = (**'arial'**,15,**'bold'**)).place(x=150,y=50)  
 Entry(self,textvariable=m,width=40).place(x=105,y=100)  
 Button(self, text=**'Pay Fine'**, width=20,command = clear).place(x=150, y=130)  
Fine().mainloop()

ForgotPassword.py

from tkinter import \*  
from tkinter import ttk  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
class Fp(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.iconbitmap(**r'libico.ico'**)  
 self.maxsize(480, 320)  
 self.title(**"Find Your Password Here"**)  
 self.canvas = Canvas(width=500, height=200)  
 self.canvas.pack()  
  
 m = StringVar()  
 n = StringVar()  
 o = StringVar()  
 p = StringVar()  
 q = StringVar()  
  
 def ins():  
 if (len(p.get())) < 8 or len(q.get()) < 8:  
 while True:  
 if not re.search(**"[a-z]"**, p.get()):  
 flag = -1  
 break  
 elif not re.search(**"[A-Z]"**, p.get()):  
 flag = -1  
 break  
 elif not re.search(**"[0-9]"**, p.get()):  
 flag = -1  
 break  
 elif not re.search(**"[\_@$]"**, p.get()):  
 flag = -1  
 break  
 elif re.search(**"\s"**, p.get()):  
 flag = -1  
 break  
 else:  
 flag = 0  
 break  
 if len(p.get()) == 0:  
 messagebox.showinfo(**"Error"**, **"Please Enter Your Password"**)  
 elif flag == -1:  
 messagebox.showinfo(**"Error"**,  
 **"Minimum 8 characters.**\n**The alphabets must be between [a-z]**\n**At least one alphabet should be of Upper Case [A-Z]**\n**At least 1 number or digit between [0-9].**\n**At least 1 character from [ \_ or @ or $ ]."**)  
 elif p.get() != q.get():  
 messagebox.showinfo(**"Error"**, **"New and retype password are not some"**)  
 else:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Update admin set password = ? where id = ?"**, (q.get(), m.get()))  
 self.connection.commit()  
 self.pointTo.close()  
 self.connection.close()  
 messagebox.showinfo(**"Confirm"**, **"Password Updated Successfully"**)  
 self.destroy()  
 except Error:  
 messagebox.showerror(**"Error"**, **"Something Goes Wrong"**)  
  
 def checkUser():  
 if len(m.get()) < 5:  
 messagebox.showinfo(**"Error"**, **"Please Enter User Id"**)  
 elif len(n.get()) == 0:  
 messagebox.showinfo(**"Error"**, **"Please Choose a question"**)  
 elif len(o.get()) == 0:  
 messagebox.showinfo(**"Error"**, **"Please Enter a answer"**)  
 else:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select id,secQuestion,secAnswer from admin where id = ?"**, (m.get()))  
 pc = self.pointTo.fetchone()  
 if not pc:  
 messagebox.showinfo(**"Error"**, **"Something Wrong in the Details"**)  
 elif str(pc[0]) == m.get() or str(pc[1]) == n.get() or str(pc[2]) == o.get():  
 Label(self, text=**"New Password"**, font=(**'arial'**, 15, **'bold'**)).place(x=40, y=220)  
 Entry(self, show=**"\*"**, textvariable=p, width=40).place(x=230, y=224)  
 Label(self, text=**"Retype Password"**, font=(**'arial'**, 15, **'bold'**)).place(x=40, y=270)  
 Entry(self, show=**"\*"**, textvariable=q, width=40).place(x=230, y=274)  
 Button(self, text=**"Submit"**, width=15, command=ins).place(x=230, y=324)  
 except Error:  
 messagebox.showerror(**"Error"**, **"Something Goes Wrong"**)  
  
 *# label and input box* Label(self, text=**"Enter User Id"**, fg=**'brown'**, font=(**'arial'**, 14, **'bold'**)).place(x=40, y=20)  
 Label(self, text=**"Security Question"**, fg=**'brown'**, font=(**'arial'**, 14, **'bold'**)).place(x=40, y=70)  
 Label(self, text=**"Security Answer"**, fg=**'brown'**, font=(**'arial'**, 14, **'bold'**)).place(x=40, y=120)  
 Entry(self, textvariable=m, width=40).place(x=230, y=24)  
 ttk.Combobox(self, textvariable=n,  
 values=[**"What is your school name?"**, **"What is your home name?"**, **"What is your Father name?"**,  
 **"What is your pet name?"**], width=37, state=**"readonly"**).place(x=230, y=74)  
 Entry(self, show=**"\*"**, textvariable=o, width=40).place(x=230, y=124)  
 Button(self, text=**'Verify'**, width=15, command=checkUser).place(x=275, y=170)  
  
  
Fp().mainloop()

LibraryMain.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
py=sys.executable  
class Lib(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.a = StringVar()  
 self.b = StringVar()  
 self.maxsize(500, 400)  
 self.minsize(500, 400)  
 self.state(**"zoomed"**)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 self.title(**"LIBRARY MANAGEMENT SYSTEM"**)  
  
  
*#verifying input* def chex():  
 if len(self.user\_text.get()) < 0:  
 messagebox.showinfo(**" INVALID USERNAME OR PASSWORD"** )  
 elif len(self.pass\_text.get()) < 0:  
 messagebox.showinfo(**" INVALID USERNAME OR PASSWORD"**)  
 else:  
 try:  
 self.conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.myCursor = self.conn.cursor()  
 self.myCursor.execute(**"Select \* from admin where id=%s AND password =%s"**,(self.user\_text.get(),self.pass\_text.get()))  
 self.pc = self.myCursor.fetchall()  
 self.myCursor.close()  
 self.conn.close()  
 if self.pc:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'Menu.py'**))  
 else:  
 messagebox.showinfo(**'Error'**, **'Username and password not found'**)  
 self.user\_text.delete(0, END)  
 self.pass\_text.delete(0, END)  
 except Exception as ex:  
 print(ex)  
 def fp():  
 os.system(**'%s %s'** % (py, **'ForgotPassword.py'**))  
  
 def check():  
 try:  
 conn = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 mycursor = conn.cursor()  
 mycursor.execute(**"Select \* from admin"**)  
 z = mycursor.fetchone()  
 mycursor.close()  
 conn.close()  
 if not z:  
 messagebox.showinfo(**"Error"**, **"Please Register A user"**)  
 x = messagebox.askyesno(**"Confirm"**,**"Do you want to register a user"**)  
 if x:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'Registration.py'**))  
 else:  
 self.lbleName = Label(self, text=**"USER LOGIN PAGE"**,fg = **'BROWN'**, font=(**"ARIAL"**, 24,**'bold'**))  
 self.lbleName.place(x=100, y=100)  
 self.lbleName1 = Label(self, text=**"Username"**,fg = **'red'**, font=(**"Times New roman"**, 18, **'bold'**))  
 self.lbleName1.place(x=130, y=150)  
 self.user\_text = Entry(self, textvariable=self.a, width=20)  
 self.user\_text.place(x=250, y=150)  
 self.lbleName2 = Label(self, text=**"Password"**,fg = **'red'**, font=(**"Times new roman"**, 18, **'bold'**))  
 self.lbleName2.place(x=130, y=195)  
 self.pass\_text = Entry(self, show=**'\*'**, textvariable=self.b, width=20)  
 self.pass\_text.place(x=250, y=190)  
 self.butt = Button(self, text=**"Login"**, font=10, width=8, command=chex).place(x=150, y=250)  
 self.butt2 = Button(self, text=**"Forgot Password"**,font=8, width=15, command=fp).place(x=150, y=300)  
 except Exception as ex:  
 print(ex)  
  
 check()  
  
Lib().mainloop()

Menu.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
import sys  
from tkinter import ttk  
pyCmnd = sys.executable  
class MainWin(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.configure(bg=**'red'**)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 self.maxsize(1320, 768)  
 self.minsize(1320, 768)  
 self.state(**'zoomed'**)  
 self.title(**'LIBRARY MANAGEMENT SYSTEM'**)  
 self.m = StringVar()  
 self.n = StringVar()  
 self.mymenu = Menu(self)  
  
 *# calling scripts* def a\_s():  
 os.system(**'%s %s'** % (pyCmnd, **'NewStudentAddition.py'**))  
  
 def a\_b():  
 os.system(**'%s %s'** % (pyCmnd, **'NewBookAddition.py'**))  
  
 def r\_b():  
 os.system(**'%s %s'** % (pyCmnd, **'BookRemove.py'**))  
  
 def r\_s():  
 os.system(**'%s %s'** % (pyCmnd, **'StudentRemove.py'**))  
  
 def ib():  
 os.system(**'%s %s'** % (pyCmnd, **'BookIssue.py'**))  
  
 def rb1():  
 os.system(**'%s %s'** % (pyCmnd, **'BookRenewal.py'**))  
  
 def ret():  
 os.system(**'%s %s'** % (pyCmnd, **'BookReturn.py'**))  
  
 def sea():  
 os.system(**'%s %s'** % (pyCmnd, **'BookSearch.py'**))  
  
 def log():  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmnd, **'LibraryMain.py'**))  
 exit()  
  
 def add\_user():  
 os.system(**'%s %s'** % (pyCmnd, **'viewIssues.py'**))  
  
 def rem\_user():  
 os.system(**'%s %s'** % (pyCmnd, **'RemoveUser.py'**))  
 def rem1\_user():  
 os.system(**'%s %s'** % (pyCmnd, **'BookWiseIssues.py'**))  
 def class\_user():  
 os.system(**'%s %s'** % (pyCmnd, **'ClassficationUser.py'**))  
  
 def cfine():  
 os.system(**'%s %s'** % (pyCmnd, **'FineClearence.py'**))  
  
 def sest():  
 os.system(**'%s %s'** % (pyCmnd, **'SearchStudent.py'**))  
  
 *# creating table* self.listForm = ttk.Treeview(self, height=14,columns=(**'SID'**, **'Name'**, **'Fine'**, **'Book Name'**, **'Issue Date'**, **'Return Date'**))  
 self.vsb = ttk.Scrollbar(self, orient=**"vertical"**, command=self.listForm.yview)  
 self.hsb = ttk.Scrollbar(self, orient=**"horizontal"**, command=self.listForm.xview)  
 self.listForm.configure(yscrollcommand=self.vsb.set, xscrollcommand=self.hsb.set)  
 self.listForm.heading(**"#0"**, text=**'Book ID'**, anchor=**'center'**)  
 self.listForm.column(**"#0"**, width=100, minwidth=100, anchor=**'center'**)  
 self.listForm.heading(**"#1"**, text=**'SID'**)  
 self.listForm.column(**"#1"**, width=100, minwidth=100, anchor=**'center'**)  
 self.listForm.heading(**"Name"**, text=**'Name'**)  
 self.listForm.column(**"Name"**, width=150, minwidth=150, anchor=**'center'**)  
 self.listForm.heading(**"Fine"**, text=**'Fine'**)  
 self.listForm.column(**"Fine"**, width=100, minwidth=100, anchor=**'center'**)  
 self.listForm.heading(**"Book Name"**, text=**'Book Name'**)  
 self.listForm.column(**"Book Name"**, width=200, minwidth=200, anchor=**'center'**)  
 self.listForm.heading(**"Return Date"**, text=**'Return Date'**)  
 self.listForm.column(**"Return Date"**, width=125, minwidth=125, anchor=**'center'**)  
 self.listForm.heading(**"Issue Date"**, text=**'Issue Date'**)  
 self.listForm.heading(**"Issue Date"**, text=**'Issue Date'**)  
 self.listForm.column(**"Issue Date"**, width=125, minwidth=125, anchor=**'center'**)  
 self.listForm.place(x=220, y=360)  
 self.vsb.place(x=1123, y=361, height=287)  
 self.hsb.place(x=220, y=650, width=922)  
 ttk.Style().configure(**"Treeview"**, font=(**'Times new Roman'**, 15))  
  
 list1 = Menu(self)  
 list1.add\_command(label=**"Student"**, command=a\_s)  
 list1.add\_command(label=**"Book"**, command=a\_b)  
  
  
 list2 = Menu(self)  
 list2.add\_command(label=**"Student"**, command=r\_s)  
 list2.add\_command(label=**"Book"**, command=r\_b)  
  
 list3 = Menu(self)  
 list3.add\_command(label=**"On Students wise"**, command=add\_user)  
 list3.add\_command(label=**"On Books wise"**, command=rem1\_user)  
 list3.add\_command(label=**"on Classification wise"**, command=class\_user)  
  
 self.mymenu.add\_cascade(label=**'Add'**, menu=list1)  
 self.mymenu.add\_cascade(label=**'Remove'**, menu=list2)  
 self.mymenu.add\_cascade(label=**'Book Issues'**, menu=list3)  
   
  
 self.config(menu=self.mymenu)  
  
 def ser():  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.change = int(self.m.get())  
 self.pointTo.execute(  
 **"Select issue.BID,issue.SID,students.name,students.Fine,books.Book\_name,issue.Issue\_date,issue.Return\_date from books,students,issue where issue.BID = books.Book\_Id and SID = %s"**,  
 (self.change))  
 self.pc = self.pointTo.fetchall()  
 if self.pc:  
 self.listForm.delete(\*self.listForm.get\_children())  
 for row in self.pc:  
 self.listForm.insert(**""**, **'end'**, text=row[0],values=(row[1], row[2], row[3], row[4], row[5], row[6]))  
 else:  
 messagebox.showinfo(**"Error"**, **"Either ID is wrong or The book is not yet issued on this ID"**)  
 except Exception as ex:  
 print(ex)  
  
 def ent():  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select issue.BID,issue.SID,students.name,students.Fine,books.Book\_name,issue.Issue\_date,issue.Return\_date from books,students, issue where issue.BID = books.Book\_Id and BID = %s"**,  
 (self.n.get()))  
 self.pc = self.pointTo.fetchall()  
 if self.pc:  
 self.listForm.delete(\*self.listForm.get\_children())  
 for row in self.pc:  
 self.listForm.insert(**""**, **'end'**, text=row[0],  
 values=(row[1], row[2], row[3], row[4], row[5], row[6]))  
 else:  
 messagebox.showinfo(**"Error"**, **"Please Enter a valid ID"**)  
 except Exception as ex:  
 print(ex)  
  
 def check():  
 try:  
 connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 pointTo.execute(**"Select \* from admin"**)  
 z = pointTo.fetchone()  
 if not z:  
 messagebox.showinfo(**"Error"**, **"Please Register A user"**)  
 confirmMessage = messagebox.askyesno(**"Confirm"**, **"Do you want to register a user"**)  
 if confirmMessage:  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmnd, **'Registration.py'**))  
 else:  
 *# label and input box* self.label3 = Label(self, text=**'LIBRARY MANAGEMENT SYSTEM'**, fg=**'red'**,  
 font=(**'Arial'**, 30, **'bold'**))  
 self.label3.place(x=400, y=22)  
 self.label4 = Label(self, text=**"STUDENT ID"**, font=(**'Arial'**, 18, **'bold'**))  
 self.label4.place(x=80, y=107)  
 self.e1 = Entry(self, textvariable=self.m, width=90).place(x=405, y=110)  
 self.srt = Button(self, text=**'Search'**, width=15, font=(**'arial'**, 10), command=ser).place(x=1000,  
 y=106)  
 self.label5 = Label(self, text=**"BOOK ID"**, font=(**'Arial'**, 18, **'bold'**))  
 self.label5.place(x=95, y=150)  
 self.e2 = Entry(self, textvariable=self.n, width=90).place(x=405, y=160)  
 self.nrt = Button(self, text=**'Find'**, width=15, font=(**'arial'**, 10), command=ent).place(x=1000, y=150)  
 self.label6 = Label(self, text=**"INFORMATION DETAILS"**, font=(**'Arial'**, 15, **'underline'**, **'bold'**))  
 self.label6.place(x=510, y=300)  
 self.nutton = Button(self, text=**'Search Student'**, width=25, font=(**'Arial'**, 10),  
 command=sest).place(x=240, y=250)  
 self.nutton = Button(self, text=**'Search Book'**, width=25, font=(**'Arial'**, 10), command=sea).place(  
 x=470, y=250)  
 self.nrt = Button(self, text=**"Issue Book"**, width=15, font=(**'Arial'**, 10), command=ib).place(x=700,  
 y=250)  
 self.nrt = Button(self, text=**"Renew Book"**, width=15, font=(**'Arial'**, 10), command=rb1).place(  
 x=850, y=250)  
 self.nrt = Button(self, text=**"Return Book"**, width=15, font=(**'Arial'**, 10), command=ret).place(  
 x=1000, y=250)  
 self.nrt = Button(self, text=**"LOGOUT"**, width=15, font=(**'Arial'**, 10), command=log).place(x=1150,  
 y=105)  
 self.nrt = Button(self, text=**"Fine Clear"**, width=15, font=(**'Arial'**, 10), command=cfine).place(  
 x=1150, y=150)  
 except Exception as ex:  
 print(ex)  
  
 check()  
  
  
MainWin().mainloop()

NewBookAddition.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
import os  
import sys  
from pymysql import Error  
  
pyCmd = sys.executable  
  
*#creating window*class Add(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(480,360 )  
 self.minsize(480,360)  
 self.title(**'Add Book Information'**)  
 self.canvas = Canvas(width=500, height=500)  
 self.canvas.pack()  
 m = StringVar()  
 n = StringVar()  
 p = StringVar()  
 *#verifying Input* def b\_q():  
 if len(m.get()) == 0 or len(n.get()) == 0:  
 messagebox.showerror(**"Error"**,**"Please Enter The Details"**)  
 else:  
 g = 1  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 edit\_insert\_statement = **" INSERT INTO books VALUES (%s,%s,%s,%s)"** record\_statement2 = (m.get(), n.get(), p.get(), g)  
 self.pointTo.execute(edit\_insert\_statement,record\_statement2)  
 self.connection.commit()  
 messagebox.showinfo(**'Info'**, **'Successfully Added'**)  
 askMessage = messagebox.askyesno(**"Confirm"**, **"Do you want to add another book?"**)  
 if askMessage:  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmd , **'NewBookAddition.py'**))  
 else:  
 self.destroy()  
 except Exception as ex:  
 print(ex)  
 Label(self, text=**''**).pack()  
 Label(self, text= **'Book Details'**,fg= **'red'**,font=(**'Times New Roman'**, 20, **'bold'**)).place(x=140, y=80)  
 Label(self, text=**''**).pack()  
 Label(self, text=**'Book Id:'**,fg=**'brown'**, font=(**'Comic Scan Ms'**, 10, **'bold'**)).place(x=60, y=130)  
 Entry(self, textvariable=m, width=30).place(x=170, y=132)  
 Label(self, text=**'Book Name:'**,fg=**'brown'**, font=(**'Comic Scan Ms'**, 10, **'bold'**)).place(x=60, y=180)  
 Entry(self, textvariable=n, width=30).place(x=170, y=182)  
 Label(self, text=**'Book Author:'**,fg=**'brown'**, font=(**'Comic Scan Ms'**, 10, **'bold'**)).place(x=60, y=230)  
 Entry(self, textvariable=p, width=30).place(x=170, y=232)  
 Button(self, text=**"Submit"**, command=b\_q).place(x=245, y=300)  
Add().mainloop()

NewStudentAddition.py

from tkinter import \*  
from tkinter import messagebox  
from tkinter import filedialog  
import pymysql  
from pymysql import Error  
import os  
import sys  
pyCmd = sys.executable  
class Add(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(500,417)  
 self.minsize(500,417)  
 self.title(**'Add New Student Details'**)  
 self.canvas = Canvas(width=500, height=417)  
 self.canvas.pack()  
  
 r = StringVar()  
 m = StringVar()  
 n = StringVar()  
 o = StringVar()  
 p = StringVar()  
 q = StringVar()  
*#uploading image* def convertToBinaryData(filename):  
 with open(filename, **'rb'**) as file:  
 blobData = file.read()  
 return blobData  
*#verifying input* def asi():  
 if len(r.get()) < 1:  
 messagebox.showinfo(**"Oop's"**, **"Enter Your library ID"**)  
 elif len(m.get()) < 1:  
 messagebox.showinfo(**"Oop's"**,**"Enter Your Student Name"**)  
 elif len(n.get()) < 1:  
 messagebox.showinfo(**"Oop's"**, **"Enter Your Student PId"**)  
 elif len(o.get()) < 1:  
 messagebox.showinfo(**"Oop's"**, **"Enter Your Student year"**)  
 elif len(p.get()) < 10 or len(p.get()) > 10:  
 messagebox.showinfo(**"Oop's"**, **"Enter Your Student Contact Number"**)  
 elif len(q.get()) < 1:  
 messagebox.showinfo(**"Oop's"**, **"Select an Image of id"**)  
 else:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 edit\_insert\_statement =**"Insert into students(Roll\_no,name,Student\_Id,class,Phone\_number,Image) values (%s,%s,%s,%s,%s,%s)"** record\_statement2 = (r.get(),m.get(),n.get(),o.get(),p.get(),convertToBinaryData(q.get()))  
 pc = self.pointTo.execute(edit\_insert\_statement,record\_statement2)  
 self.connection.commit()  
 if pc:  
 messagebox.showinfo(**"Done"**,**"Student Inserted Successfully"**)  
 asConfrmMsg = messagebox.askyesno(**"Confirm"**,**"Do you want to add another student?"**)  
 if asConfrmMsg:  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmd, **'NewStudentAddition.py'**))  
 else:  
 self.destroy()  
 else:  
 messagebox.showerror(**"Error"**, **'Something goes wrong'**)  
 self.pointTo.close()  
 self.connection.close()  
 except Exception as ex:  
 print(ex)  
  
 *# label and input box* label4 = Label(self, text=**'Add Student Details'**, fg=**'red'**, font=(**'Arial bold'**, 25, **'bold'**)).place(x=100, y=32)  
 lbl = Label(self, text=**'Library ID:'**, font=(**'Arial'**, 11, **'bold'**)).place(x=70, y=82)  
 S\_name = Entry(self, textvariable=r, width=30).place(x=200, y=84)  
 label = Label(self, text=**'Student Name:'**, font=(**''**, 11, **'bold'**)).place(x=70, y=130)  
 S\_name = Entry(self, textvariable=m, width=30).place(x=200, y=132)  
 label5 = Label(self, text=**'Student ID:'**, font=(**'Arial'**, 11, **'bold'**)).place(x=70, y=180)  
 S\_ID = Entry(self, textvariable=n, width=30).place(x=200, y=182)  
 label6 = Label(self, text=**'Course/Section:'**, font=(**'Arial'**, 11, **'bold'**)).place(x=70, y=230)  
 S\_Class = Entry(self, textvariable=o, width=30).place(x=200, y=232)  
 label7 = Label(self, text=**'Contact Number:'**, font=(**'Arial'**, 11, **'bold'**)).place(x=70, y=280)  
 def fileDialog():  
 filename = filedialog.askopenfilename(initialdir = **"/"**,title = **"Select A File"**,filetype = ((**"jpeg"**,**"\*.jpg"**),(**"png"**,**"\*.png"**),(**"All Files"**,**"\*.\*"**)))  
 q.set(filename)  
 label8 = Label(self, text=**"Upload image"**, font=(**'Arial'**, 11, **'bold'**)).place(x=70, y=330)  
 upload\_image = Entry(self, textvariable=q, width=30).place(x=200, y=330)  
 ZS\_phone\_number = Entry(self, textvariable=p, width=30).place(x=200, y=282)  
 butt = Button(self, text=**"Browse"**, width=7, command=fileDialog).place(x=400, y=328)  
 S\_butt = Button(self, text=**"Submit"**,width = 15,command=asi).place(x=230, y=370)  
  
Add().mainloop()

Registration.py

from tkinter import \*  
from tkinter import messagebox  
import re  
from tkinter import ttk  
import pymysql  
from pymysql import Error  
import os,sys  
pyCmd=sys.executable  
class reg(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**"LIBRARY MANAGEMENT SYSTEM"**)  
 self.maxsize(866, 668)  
 self.minsize(866, 668)  
 self.state(**"normal"**)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 m = StringVar()  
 n = StringVar()  
 x = StringVar()  
 p = StringVar()  
 v = StringVar()  
 u = StringVar()  
 s = StringVar()  
 r = StringVar()  
  
 def insert():  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 confrmMsg = self.pointTo.execute(**"Insert into admin values (%s,%s,%s,%s,%s,%s,%s)"**,(m.get(), n.get(), x.get(), p.get(), v.get(), s.get(), r.get()))  
 self.connection.commit()  
 self.pointTo.close()  
 self.connection.close()  
 if confrmMsg:  
 messagebox.showinfo(**"Confirm"**, **"Data Inserted Successfully"**)  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmd, **'Main.py'**))  
 except Exception as ex:  
 print(ex)  
*# verify input* def verifyUserCredentials():  
 if(len(m.get())) < 5:  
 messagebox.showinfo(**"Error"**,**"Enter User Id**\n**User Id should be greater than 5 letters"**)  
 elif (len(n.get())) < 3:  
 messagebox.showinfo(**"Error"**, **"Please Enter Your Full Name"**)  
 elif (len(x.get())) < 8:  
 while True:  
 if not re.search(**"[a-z]"**, x.get()):  
 flag = -1  
 break  
 elif not re.search(**"[A-Z]"**, x.get()):  
 flag = -1  
 break  
 elif not re.search(**"[0-9]"**, x.get()):  
 flag = -1  
 break  
 elif not re.search(**"[\_@$]"**, x.get()):  
 flag = -1  
 break  
 elif re.search(**"\s"**, x.get()):  
 flag = -1  
 break  
 else:  
 flag = 0  
 break  
 if len(x.get()) == 0:  
 messagebox.showinfo(**"Error"**,**"Please Enter Your Password"**)  
 elif flag == -1:  
 messagebox.showinfo(**"Error"**,**"Minimum 8 characters.**\n**The alphabets must be between [a-z]**\n**At least one alphabet should be of Upper Case [A-Z]**\n**At least 1 number or digit between [0-9].**\n**At least 1 character from [ \_ or @ or $ ]."**)  
 elif len(p.get()) == 0:  
 messagebox.showinfo(**"Error"**,**"Please select a question"**)  
 elif len(v.get()) == 0:  
 messagebox.showinfo(**"Error"**,**"Please write an answer"**)  
 elif len(s.get()) == 0 or len(s.get()) > 10 or len(s.get()) < 10:  
 messagebox.showinfo(**"Error"**,**"Enter Valid Phone Number"**)  
 elif len(s.get()) == 10:  
 if s.get().isdigit():  
 cas = re.fullmatch(**"[6-9][0-9]{9}"**, s.get())  
 if cas is None:  
 messagebox.showinfo(**"Error"**,**"Check Your Phone Number"**)  
 else:  
 insert()  
*#label and input* Label(self,text=**"Library Management System"**,font=(**"Arial"**,25,**'bold'**),fg=**"purple"**).place(x=200,y=80)  
 *#Label(self,text="Enter your details and click save",font=("Arial",20,'bold'),fg="brown").place(x=200,y=600)* Label(text = **"Library Information"**,fg=**'brown'**,font = (**"Arial"**,13,**"bold"**)).place(x=300,y=220)  
 Label( text=**"Username"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=260)  
 Label( text=**"Name"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=300)  
 Label( text=**"Password"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=340)  
 Label( text=**"Security Question"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=380)  
 Label( text=**"Security Answer"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=420)  
 Label( text=**"Phone"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=460)  
 Label( text=**"City"**,fg=**'brown'**, font=(**"Arial"**, 13, **"bold"**)).place(x=100, y=500)  
 Entry(textvariable=m,width=60).place(x=250,y=260)  
 Entry( textvariable=n, width=60).place(x=250, y=300)  
 Entry( show = **'\*'**,textvariable=x, width=60).place(x=250, y=340)  
 ttk.Combobox( textvariable = p, values=[**"What is your school name?"**, **"What is your home name?"**,**"What is your Father name?"**, **"What is your pet name?"**], width=57,state=**"readonly"**).place(x=250, y=380)  
 Entry( show = **'\*'**,textvariable=v, width=60).place(x=250, y=420)  
 Entry( textvariable=s, width=60).place(x=250, y=460)  
 Entry( textvariable=r, width=60).place(x=250, y=500)  
 Button( text=**"Save"**, width=10, font=(**"Arial"**, 13, **"bold"**), command=verifyUserCredentials).place(x=300, y=550)  
 Button( text=**"Cancel"**, width=10, font=(**"Arial"**, 13, **"bold"**)).place(x=460, y=550)  
  
reg().mainloop()

Remove\_student.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
import sys  
  
py = sys.executable  
  
class Rem(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(450,250)  
 self.minsize(450,250)  
 self.title(**"Remove Student Details"**)  
 self.canvas = Canvas(width=500, height=417)  
 self.canvas.pack()  
 a = StringVar()  
  
 def iii():  
 if len(a.get()) == 0:  
 messagebox.showerror(**"Error"**, **"Please Enter The Student Id"**)  
 else:  
 c = messagebox.askyesno(**'Remove Book'**, **'Are You Sure You Want To Remove The Student'**)  
 if c:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"DELETE FROM students WHERE Student\_Id = %s"**, (a.get()))  
 messagebox.showinfo(**'Remove'**, **'Successfully Removed'**)  
 self.connection.commit()  
 self.connection.close()  
 d = messagebox.askyesno(**"Confirm"**, **"Do you want to remove another student"**)  
 if d:  
 self.destroy()  
 os.system(**'%s %s'** % (py, **'Remove\_student.py'**))  
 else:  
 self.destroy()  
 except Error:  
 messagebox.showerror(**"Error"**, **"Something Goes Wrong"**)  
  
 self.lb = Label(self, text=**"Enter Student Id"**, font=(**'Arial'**, 15, **'bold'**))  
 self.lb.place(x=30, y=70)  
 self.e1 = Entry(self, textvariable=a, width=30).place(x=230, y=77)  
 self.butt1234 = Button(self, text=**"Remove"**, width=20, command=iii).place(x=230, y=120)  
Rem().mainloop()

RemoveUser.py

from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
class Rem(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(400, 200)  
 self.minsize(400, 200)  
 self.title(**"Remove User Details"**)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 a = StringVar()  
 def ent():  
 dConfirmMessage = messagebox.askyesno(**"Confirm"**, **"Are you sure you want to remove the user?"**)  
 if dConfirmMessage:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 print(a.get())  
 self.pointTo.execute(**"Delete from admin where id = %s"**,(a.get()))  
 temp = self.pointTo.fetchone()  
 if temp:  
 messagebox.showinfo(**"Oop's"**,**"User Not Found"**)  
 a.set(**""**)  
 else:  
 self.connection.commit()  
 self.pointTo.close()  
 self.connection.close()  
 messagebox.showinfo(**"Confirm"**,**"User Removed Successfully"**)  
 a.set(**""**)  
 except Exception as ex:  
 print(ex)  
 Label(self, text = **"Username: "**,fg=**'brown'**,font=(**'Arial'**, 15, **'bold'**)).place(x = 20,y = 40)  
 Entry(self,textvariable = a,width = 37).place(x = 160,y = 44)  
 Button(self, text=**'Remove'**, width=15, font=(**'arial'**, 10),command = ent).place(x=200, y = 90)  
  
  
  
Rem().mainloop()

SearchStudent.py

from tkinter import \*  
from tkinter import ttk  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
from PIL import ImageTk,Image  
import os,glob  
  
  
class Sst(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 f = StringVar()  
 g = StringVar()  
 self.title(**"Search Student Details"**)  
 self.maxsize(770,320)  
 self.canvas = Canvas(width=1366, height=768)  
 self.canvas.pack()  
 l1=Label(self,text=**"Search Student"**,font=(**"Arial"**,19,**'bold'**)).place(x=290,y=40)  
 l = Label(self, text=**"Search Type"**, font=(**"Arial"**, 14, **'bold'**)).place(x=180, y=100)  
 def writeTofile(data,filename):  
 with open(filename,**'wb'**) as file:  
 file.write(data)  
  
 def insert(data):  
 self.listForm.delete(\*self.listForm.get\_children())  
 for row in data:  
 self.listForm.insert(**""**,**"end"**,text = row[2], values = (row[1],row[4],row[6]))  
  
 def photo(pic):  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select \* from students where Student\_Id = %s"**, (pic))  
 pcuteStudents = self.pointTo.fetchone()  
 if pcuteStudents[5] != **''**:  
 photoPath = **"TempImages**\\**"** + pcuteStudents[1] + **".jpeg"** writeTofile(pcuteStudents[5], photoPath)  
 self.photo = ImageTk.PhotoImage(Image.open(**"TempImages**\\**"** + pcuteStudents[1] + **".jpeg"**))  
 Label(image=self.photo, width=150, height=150).place(x=625, y=20)  
 flist = glob.glob(**"TempImages\\*.jpeg"**)  
 for file in flist:  
 os.remove(file)  
 else:  
 self.photo = ImageTk.PhotoImage(Image.open(**"TempImages**\\**48-512.png"**))  
 Label(image=self.photo, width=150, height=150).place(x=625, y=20)  
 except Error:  
 messagebox.showerror(**"Error"**, **"Something goes wrong"**)  
  
*#clicking the record will open the picture* def select(a):  
 currentItem = self.listForm.focus()  
 selectedItem = self.listForm.item(currentItem)  
 pic = str(selectedItem[**'text'**])  
 photo(pic)  
  
  
 def ge():  
 if (len(g.get())) == 0:  
 messagebox.showinfo(**'Error'**, **'First select a item'**)  
 elif (len(f.get())) == 0:  
 messagebox.showinfo(**'Error'**, **'Enter the '**+g.get())  
 elif g.get() == **'Name'**:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select \* from students where name like %s"**,(**'%'**+f.get()+**'%'**))  
 pcuteStudents = self.pointTo.fetchall()  
 if pcuteStudents:  
 insert(pcuteStudents)  
 else:  
 messagebox.showinfo(**"Oop's"**,**"Name not found"**)  
 except Error:  
 messagebox.showerror(**"Error"**, **"Something goes wrong"**)  
 elif g.get() == **'ID'**:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"Select \* from students where Student\_Id like %s"**, (**'%'** + f.get() + **'%'**))  
 pc = self.pointTo.fetchall()  
 if pc:  
 insert(pc)  
 else:  
 messagebox.showinfo(**"Oop's"**, **"Id not found"**)  
 except Exception as ex:  
 print(ex)  
  
  
 b=Button(self,text=**"Search"**,width=8,font=(**"Arial"**,8,**'bold'**),command=ge).place(x=400,y=170)  
 c=ttk.Combobox(self,textvariable=g,values=[**"Name"**,**"ID"**],width=40,state=**"readonly"**).place(x = 310, y = 105)  
 en = Entry(self,textvariable=f,width=43).place(x=310,y=145)  
 la = Label(self, text=**"Enter"**, font=(**"Arial"**, 15, **'bold'**)).place(x=180, y=140)  
  
 def handle(event):  
 if self.listForm.identify\_region(event.x,event.y) == **"separator"**:  
 return **"break"** self.listForm = ttk.Treeview(self, height=3,columns=(**'Student Name'**, **'Phone Number'**, **'No. Of Books Issued'**))  
 self.vsb = ttk.Scrollbar(self,orient=**"vertical"**,command=self.listForm.yview)  
 self.listForm.configure(yscrollcommand=self.vsb.set)  
 self.listForm.heading(**"#0"**, text=**'Student ID'**, anchor=**'w'**)  
 self.listForm.column(**"#0"**, width=100, anchor=**'w'**)  
 self.listForm.heading(**"Student Name"**, text=**'Student Name'**)  
 self.listForm.column(**"Student Name"**, width=200, anchor=**'center'**)  
 self.listForm.heading(**"Phone Number"**, text=**'Phone Number'**)  
 self.listForm.column(**"Phone Number"**, width=200, anchor=**'center'**)  
 self.listForm.heading(**"No. Of Books Issued"**, text=**'No. Of Books Issued'**)  
 self.listForm.column(**"No. Of Books Issued"**, width=200, anchor=**'center'**)  
 self.listForm.bind(**"<Button-1>"**, handle)  
 self.listForm.bind(**"<ButtonRelease-1>"**,select)  
 self.listForm.place(x=40, y=200)  
 self.vsb.place(x=743,y=200,height=287)  
 ttk.Style().configure(**"Treeview"**, font=(**'Times new Roman'**, 15))  
  
  
  
  
  
  
  
  
  
Sst().mainloop()

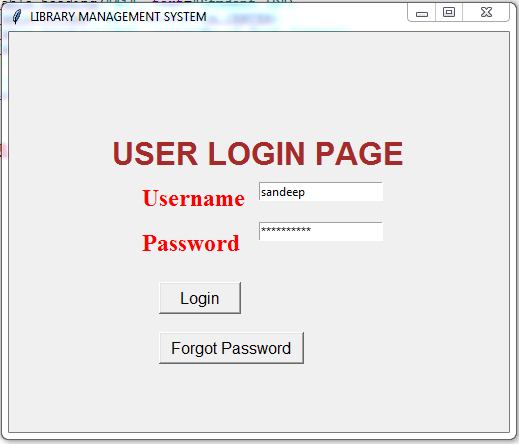
StudentRemove.py

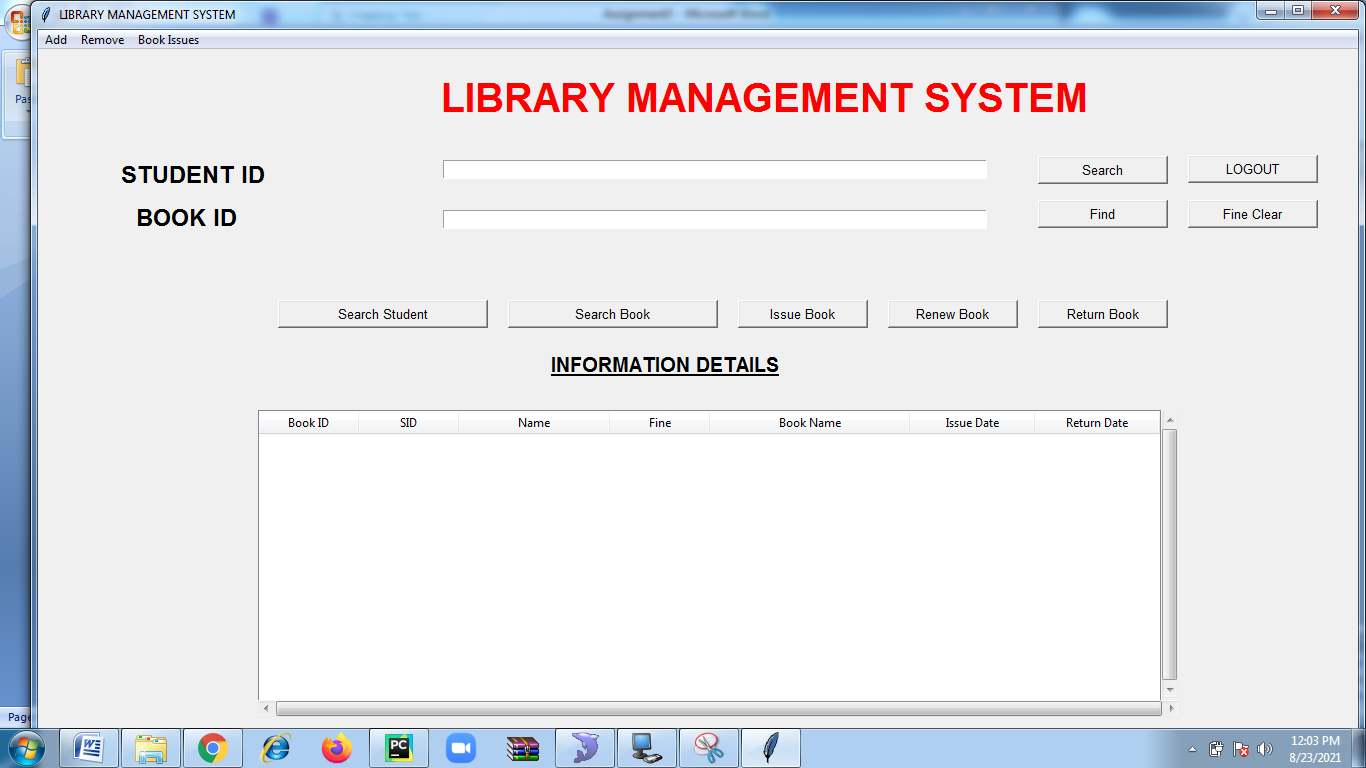
from tkinter import \*  
from tkinter import messagebox  
import pymysql  
from pymysql import Error  
import os  
import sys  
pyCmd = sys.executable  
class Rem(Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.maxsize(450,250)  
 self.minsize(450,250)  
 self.title(**"Remove Student Details"**)  
 self.canvas = Canvas(width=500, height=417)  
 self.canvas.pack()  
 a = StringVar()  
  
 def iii():  
 if len(a.get()) == 0:  
 messagebox.showerror(**"Error"**, **"Please Enter The Student Id"**)  
 else:  
 confirmMessag = messagebox.askyesno(**'Remove Book'**, **'Are You Sure You Want To Remove The Student'**)  
 if confirmMessag:  
 try:  
 self.connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 self.pointTo = self.connection.cursor()  
 self.pointTo.execute(**"DELETE FROM students WHERE Student\_Id = %s"**, (a.get()))  
 messagebox.showinfo(**'Remove'**, **'Successfully Removed'**)  
 self.connection.commit()  
 self.connection.close()  
 dconfirmMsg = messagebox.askyesno(**"Confirm"**, **"Do you want to remove another student"**)  
 if dconfirmMsg:  
 self.destroy()  
 os.system(**'%s %s'** % (pyCmd , **'StudentRemove.py'**))  
 else:  
 self.destroy()  
 except Exception as ex:  
 print(ex)  
  
 self.label = Label(self, text=**"Enter Student Id"**, font=(**'Arial'**, 15, **'bold'**))  
 self.label.place(x=30, y=70)  
 self.label = Entry(self, textvariable=a, width=30).place(x=230, y=77)  
 self.labelButton = Button(self, text=**"Delete"**, width=20, command=iii).place(x=230, y=120)  
Rem().mainloop()

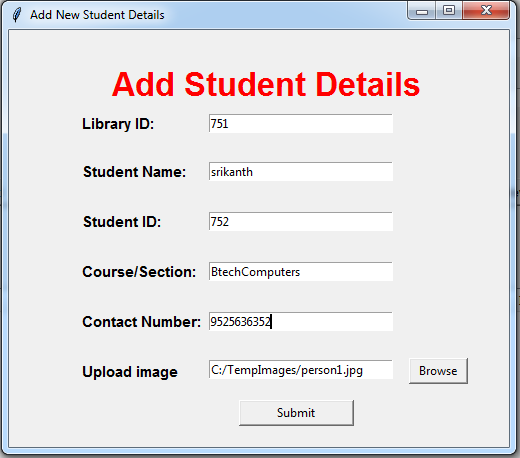
viewIssues.py

from tkinter import ttk  
import tkinter as tk  
import pymysql  
from pymysql import Error  
def dbCconnection():  
 connection = pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 connection.close()  
def ViewContent():  
 connection =pymysql.connect(host=**"localhost"**, user=**"root"**, password=**"root"**, database=**"library"**)  
 pointTo = connection.cursor()  
 pointTo.execute(**"select sid,count(bid) from issue group by sid"**)  
 rowData = pointTo.fetchall()  
 for rowData in rowData :  
 formatTable.insert(**""**,**"end"**,text = rowData [1], values = (rowData [0],rowData [1]))  
 connection.close()  
*# connect to the database*dbCconnection()  
root = tk.Tk()  
formatTable = ttk.Treeview(root, column=(**"c1"**, **"c2"**), show=**'headings'**)  
formatTable.column(**"#1"**, anchor=tk.CENTER)  
formatTable.heading(**"#1"**, text=**"Student ID"**)  
formatTable.column(**"#2"**, anchor=tk.CENTER)  
formatTable.heading(**"#2"**, text=**"No.of Book Issues"**)  
formatTable.pack()  
button1 = tk.Button(text=**"View"**, command=ViewContent)  
button1.pack(pady=10)  
root.mainloop()

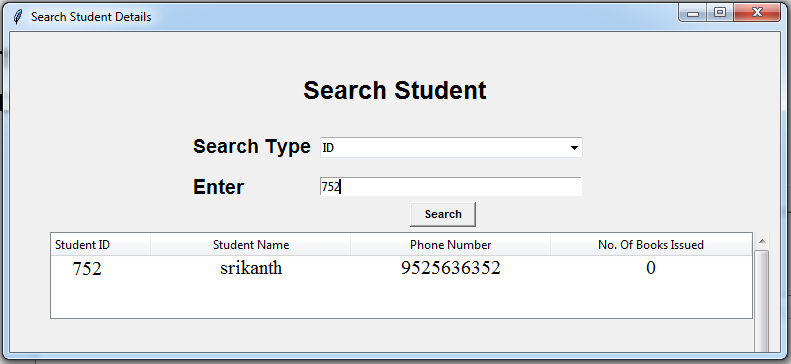
**Output screens**

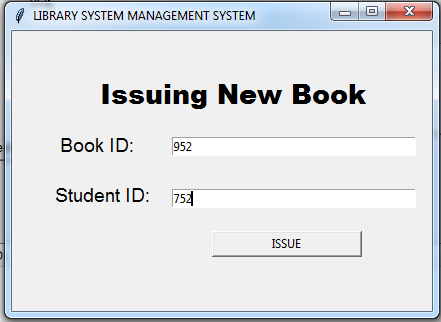


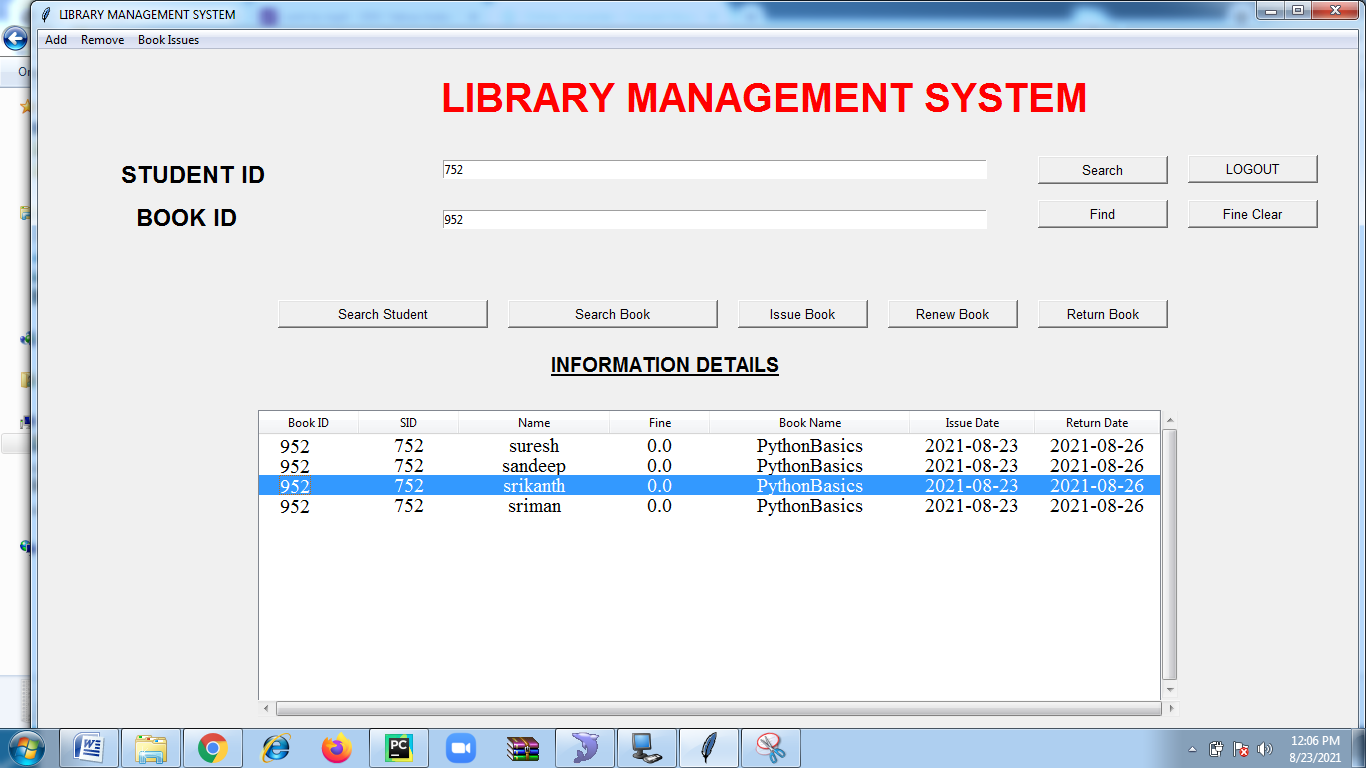


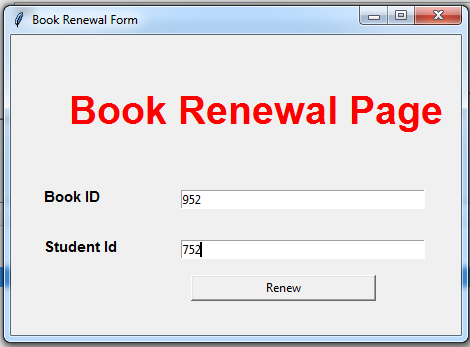


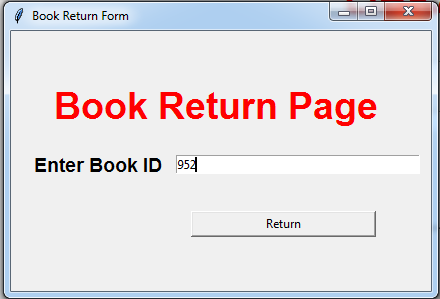


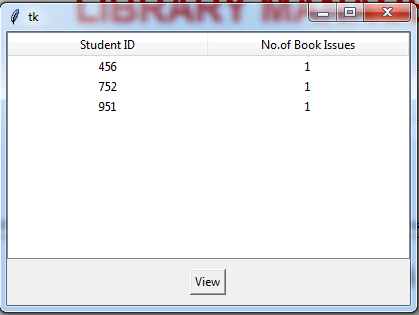


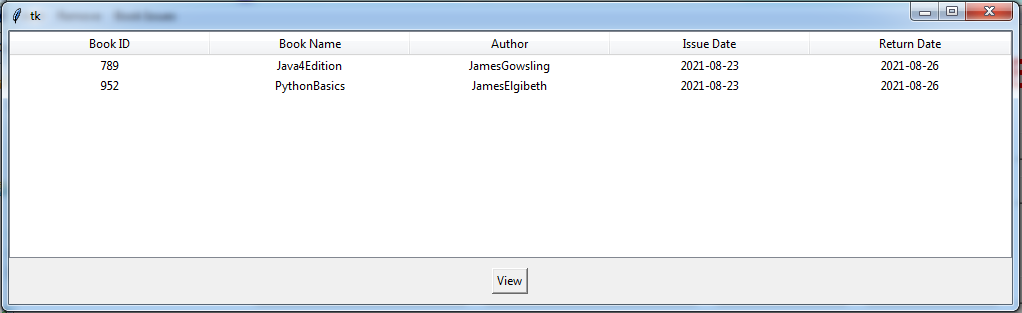


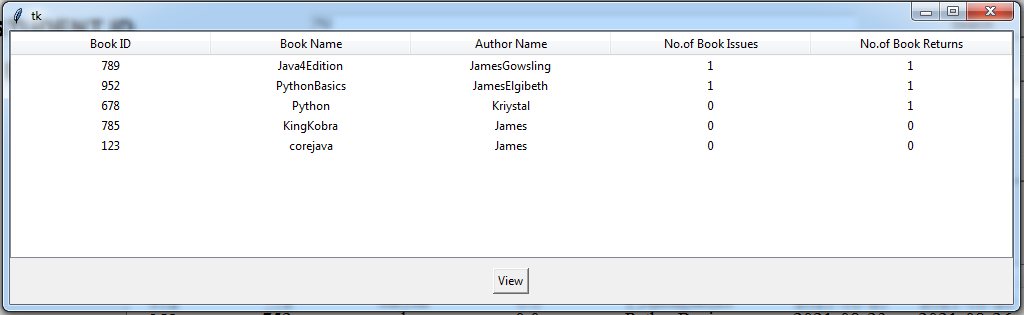


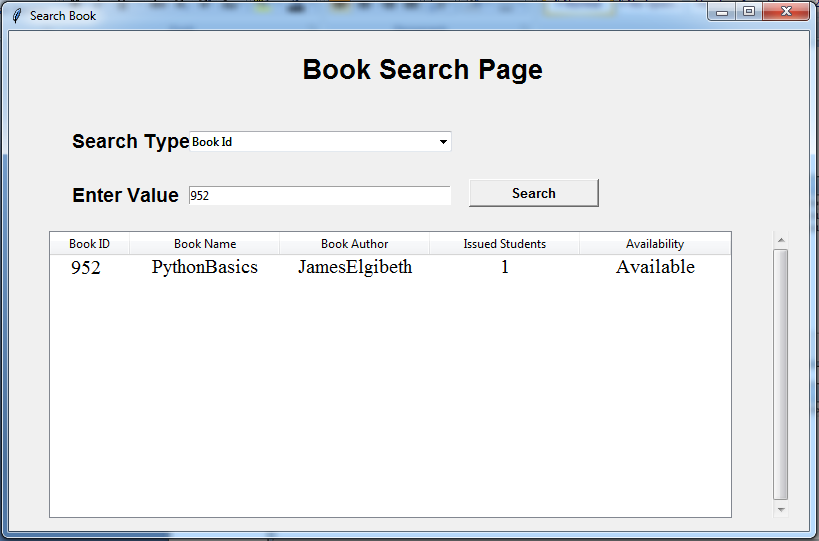












**Task-5: Supplemental video**

**Please watch the project demonstration Video by using following link**

https://youtu.be/1LYdZd79qZ8