from tkinter import \*

import sqlite3

import tkinter.messagebox

import time

from datetime import \*

from tkinter.ttk import Combobox

from tkinter import filedialog

from prettytable import PrettyTable

import sys

import pyttsx3

import threading

class Main\_window(object):

def hide(self):

self.mainwindow.withdraw()

def \_\_init\_\_(self, parent):

def open\_main\_screen():

database = sqlite3.connect("LMS.sqlite")

name = username\_Input.get()

passwordin = password\_Input.get()

if name is "":

tkinter.messagebox.showinfo('Login info', 'Username is empty')

elif passwordin is "":

tkinter.messagebox.showinfo('Login info', 'Password is empty')

elif name is not "":

for username, password in database.execute("SELECT \* FROM login WHERE username Like ? AND password Like ?", (name, passwordin)):

if password == passwordin and username == name:

login\_screen()

break

else:

tkinter.messagebox.showinfo('Login Information', 'Username and Password dose not match')

database.close()

# ================================= Other Toplavel Window =============================

def login\_screen():

def exitmain():

answer = tkinter.messagebox.askquestion('Exit confirmation', 'Do you want to close?')

if answer == 'yes':

sys.exit()

def student\_report():

from prettytable import PrettyTable

import os

conn = sqlite3.connect('LMS.sqlite')

curs = conn.cursor()

curs.execute('SELECT \* FROM student\_add ORDER BY studentname')

col\_names = [cn[0] for cn in curs.description]

rows = curs.fetchall()

x = PrettyTable(col\_names)

x.align[col\_names[0]] = "l"

x.align[col\_names[1]] = "l"

x.align[col\_names[2]] = "l"

x.align[col\_names[3]] = "l"

x.align[col\_names[4]] = "l"

x.align[col\_names[5]] = "l"

x.align[col\_names[7]] = "l"

x.align[col\_names[8]] = "l"

x.padding\_width = 1

for row in rows:

x.add\_row(row)

tabstring = x.get\_string()

output = open("export.txt", "w")

output.write("Book Report"+"\n")

output.write(tabstring)

output.close()

os.system('export.txt')

def book\_report():

from prettytable import PrettyTable

import os

conn = sqlite3.connect('LMS.sqlite')

curs = conn.cursor()

curs.execute('SELECT \* FROM book\_entry ORDER BY bookuid')

col\_names = [cn[0] for cn in curs.description]

rows = curs.fetchall()

x = PrettyTable(col\_names)

x.align[col\_names[0]] = "l"

x.align[col\_names[1]] = "l"

x.align[col\_names[2]] = "l"

x.align[col\_names[3]] = "l"

x.align[col\_names[4]] = "l"

x.align[col\_names[5]] = "l"

x.align[col\_names[6]] = "l"

x.padding\_width = 1

for row in rows:

x.add\_row(row)

tabstring = x.get\_string()

output = open("export.txt", "w")

output.write("Student Report"+"\n")

output.write(tabstring)

output.close()

os.system('export.txt')

def issue\_report():

from prettytable import PrettyTable

import os

conn = sqlite3.connect('LMS.sqlite')

curs = conn.cursor()

curs.execute('SELECT \* FROM issuebook ORDER BY bookuid')

col\_names = [cn[0] for cn in curs.description]

rows = curs.fetchall()

x = PrettyTable(col\_names)

x.align[col\_names[0]] = "l"

x.align[col\_names[1]] = "l"

x.align[col\_names[2]] = "l"

x.align[col\_names[3]] = "l"

x.align[col\_names[4]] = "l"

x.align[col\_names[5]] = "l"

x.align[col\_names[6]] = "l"

x.align[col\_names[7]] = "l"

x.align[col\_names[8]] = "l"

x.padding\_width = 5

for row in rows:

x.add\_row(row)

tabstring = x.get\_string()

output = open("export.txt", "w")

output.write("Book Issue Report"+"\n")

output.write(tabstring)

output.close()

os.system('export.txt')

self.mainwindow.withdraw()

main\_screen = Toplevel()

main\_screen.title("Library Management System")

main\_screen.state('zoomed')

# =====================Frame===========================

menu\_frame = Frame(main\_screen, width=1024, height=50, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

Label(menu\_frame, font=('arial', 20, 'bold'), text="Library Management System").pack()

frame1 = Frame(main\_screen, width=512, height=700, relief='sunken')

frame2 = Frame(main\_screen, width=640, height=700, relief='sunken')

frame3 = Frame(main\_screen, width=100, height=700, relief='sunken')

frame3.pack(side='left')

frame1.pack(side='left')

frame2.pack(side='right')

# ===============================Menu===========================================

menu = Menu(main\_screen)

main\_screen.config(menu=menu)

subMenu = Menu(menu)

menu.add\_cascade(label="File", menu=subMenu)

subMenu.add\_separator()

subMenu.add\_command(label="Exit", command=exitmain)

reportMenu = Menu(menu)

menu.add\_cascade(label="Report", menu=reportMenu)

reportMenu.add\_command(label="Student Report", command=student\_report)

reportMenu.add\_command(label="Book Report", command=book\_report)

reportMenu.add\_command(label="Book Issue Report", command=issue\_report)

AboutMenu = Menu(menu)

menu.add\_cascade(label="About", menu=AboutMenu)

AboutMenu.add\_command(label="About us", command=self.about\_from)

# ==============================Button==============================

book\_add\_button = Button(frame1, width=20, text="Entry A Book", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.book\_entry)

book\_management\_button = Button(frame1, width=20, text="Book Management", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.book\_management)

student\_add\_button = Button(frame1, width=20, text="Add A Student", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.student\_add)

student\_management\_button = Button(frame1, width=20, text="Student Management", fg='#fff', bg='#A38C20', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.student\_management)

issue\_button = Button(frame1, width=20, text="Issue A Book", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.issue\_book)

return\_button = Button(frame1, width=20, text="Return A Book", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=self.return\_from)

exit\_button = Button(frame1, width=20, text="Exit", bg='#A38C20', fg='#fff', font=('times new roman', 23, 'bold italic'), bd=2, padx=2, pady=2, command=exitmain)

book\_add\_button.grid(row=0, column=0)

book\_management\_button.grid(row=1, column=0)

student\_add\_button.grid(row=2, column=0)

student\_management\_button.grid(row=3, column=0)

issue\_button.grid(row=4, column=0)

return\_button.grid(row=5, column=0)

exit\_button.grid(row=6, column=0)

photo = PhotoImage(file="libraries\_lms.png")

lbl\_img = Label(frame2, width=640, height=480, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

self.mainwindow = parent

self.mainwindow.title("Welcome To RPI Library")

self.mainwindow.geometry('300x150+500+400')

username\_label = Label(self.mainwindow, font=('times new roman', 20, 'bold'), text="Username:")

password\_label = Label(self.mainwindow, font=('times new roman', 20, 'bold'), text="Password:")

username\_label.grid(row=0, column=0, sticky=E)

password\_label.grid(row=1, column=0, sticky=E)

# =======================Input==============================

username\_Input = StringVar()

password\_Input = StringVar()

username\_entry = Entry(self.mainwindow, textvariable=username\_Input, bd=5, bg='powder blue')

password\_entry = Entry(self.mainwindow, textvariable=password\_Input, bd=5, show='\*', bg='powder blue')

username\_entry.grid(row=0, column=1, pady=5, padx=10)

password\_entry.grid(row=1, column=1, pady=5, padx=10)

# =======================Button===============================

login\_button = Button(self.mainwindow, font=('arial', 12, 'bold'), bd=1, text="Login", width=13, command=open\_main\_screen, activeforeground='red')

exit\_button = Button(self.mainwindow, font=('arial', 12, 'bold'), bd=1, text="Exit", width=13, activeforeground='green', command=sys.exit)

login\_button.grid(row=2, column=0, pady=10)

exit\_button.grid(row=2, column=1, pady=10)

def onCloseOtherWindow(self, otherWindow):

otherWindow.destroy()

self.show()

def show(self):

self.mainwindow.update()

self.mainwindow.deiconify()

# =======================================================================================

def book\_entry(self):

def exitbutt():

bookentry.destroy()

def clearfield():

bookName\_Input.set("")

WriterNme\_Input.set("")

bookUid\_Input.set("")

NumberOfCopies\_Input.set("")

prize\_Input.set("")

publisher\_Input.set("")

bookCategory\_Input.set("")

def savedata():

database = sqlite3.connect("LMS.sqlite")

data = bookUid\_Input.get()

for bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory in database.execute("SELECT \* FROM book\_entry WHERE bookuid Like ?", (data,)):

if bookuid == data:

tkinter.messagebox.showinfo('Book Information', 'Book already exist')

break

else:

cursor = database.execute("SELECT \* FROM book\_entry")

bookname = bookName\_Input.get()

writername = WriterNme\_Input.get()

bookuid = bookUid\_Input.get()

numberofcopies = NumberOfCopies\_Input.get()

prize = prize\_Input.get()

publisher = publisher\_Input.get()

bookcategory = bookCategory\_Input.get()

cursor.execute("INSERT INTO book\_entry VALUES(?, ?, ?, ?, ?, ?, ?)",

(bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory))

cursor.connection.commit()

cursor.close()

tkinter.messagebox.showinfo('Book Information', 'Book saved successfully')

def go\_to\_bookmanagement():

bookentry.destroy()

self.book\_management()

bookentry = Toplevel()

bookentry.title("Book Entry Form")

bookentry.maxsize()

# ===================== Frame ===========================

menu\_frame = Frame(bookentry, width=1024, height=18, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

Label(menu\_frame, text="Book Entry Form", font=('arial', 30, 'bold')).pack()

frame1 = Frame(bookentry, width=100, height=750, relief='sunken')

frame2 = Frame(bookentry, width=462, height=700, relief='sunken')

frame3 = Frame(bookentry, width=462, height=700, relief='sunken')

frame1.pack(side='left')

frame2.pack(side='left')

frame3.pack(side='right')

# ============================== Lable and Entry =============================

bookName = Label(frame2, width=20, text="Book Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

WriterNme = Label(frame2, width=20, text="Writer Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookUid = Label(frame2, width=20, text="Book UID", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

NumberOfCopies = Label(frame2, width=20, text="Number Of Copies", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

prize = Label(frame2, width=20, text="Price", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

publisher = Label(frame2, width=20, text="Publisher", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookCategory = Label(frame2, width=20, text="Book Category", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookName\_Input = StringVar()

WriterNme\_Input = StringVar()

bookUid\_Input = IntVar()

NumberOfCopies\_Input = IntVar()

prize\_Input = IntVar()

publisher\_Input = StringVar()

bookCategory\_Input = StringVar()

BookName\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookName\_Input, bd=10)

WriterNme\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=WriterNme\_Input, bd=10)

bookUid\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookUid\_Input, bd=10)

NumberOfCopies\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=NumberOfCopies\_Input, bd=10)

prize\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=prize\_Input, bd=10)

publisher\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=publisher\_Input, bd=10)

bookCategory\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookCategory\_Input, bd=10)

bookUid\_Input.set("")

NumberOfCopies\_Input.set("")

prize\_Input.set("")

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

WriterNme.grid(row=1, column=0, pady=2, padx=2)

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

NumberOfCopies.grid(row=3, column=0, pady=2, padx=2)

prize.grid(row=4, column=0, pady=2, padx=2)

publisher.grid(row=5, column=0, pady=2, padx=2)

bookCategory.grid(row=6, column=0, pady=2, padx=2)

BookName\_entry.grid(row=0, column=1)

WriterNme\_entry.grid(row=1, column=1)

bookUid\_entry.grid(row=2, column=1)

NumberOfCopies\_entry.grid(row=3, column=1)

prize\_entry.grid(row=4, column=1)

publisher\_entry.grid(row=5, column=1)

bookCategory\_entry.grid(row=6, column=1)

# ================================== Button =====================================

clear\_button = Button(frame3, width=10, text="Clear", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=clearfield)

save\_button = Button(frame3, width=10, text="Save", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=savedata)

go\_to\_bookmanagement\_button = Button(frame3, width=10, text="B M", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=go\_to\_bookmanagement)

exit\_button = Button(frame3, width=10, text="Exit", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=exitbutt)

clear\_button.grid(row=0, column=2, pady=12)

save\_button.grid(row=1, column=2, pady=13)

go\_to\_bookmanagement\_button.grid(row=2, column=2, pady=13)

exit\_button.grid(row=3, column=2, pady=10)

def book\_management(self):

def exitbutt():

book\_management.destroy()

def clearfield():

bookName\_Input.set("")

WriterNme\_Input.set("")

bookUid\_Input.set("")

NumberOfCopies\_Input.set("")

prize\_Input.set("")

publisher\_Input.set("")

bookCategory\_Input.set("")

def searchdata():

database = sqlite3.connect("LMS.sqlite")

data = bookUid\_Input.get()

for bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory in database.execute("SELECT \* FROM book\_entry WHERE bookuid Like ?", (data,)):

if bookuid == data:

bookName\_Input.set(bookname)

WriterNme\_Input.set(writername)

bookUid\_Input.set(bookuid)

NumberOfCopies\_Input.set(numberofcopies)

prize\_Input.set(prize)

publisher\_Input.set(publisher)

bookCategory\_Input.set(bookcategory)

tkinter.messagebox.showinfo('Book Information', 'Book Found')

break

else:

tkinter.messagebox.showinfo('Book Information', 'Book Not Found')

database.close()

def go\_to\_bookentry():

book\_management.destroy()

self.book\_entry()

def deletedata():

answer = tkinter.messagebox.askquestion('Delete confirmation', 'Do you want to delete?')

if answer == 'yes':

database = sqlite3.connect("LMS.sqlite")

data = bookUid\_Input.get()

database.execute("DELETE FROM book\_entry WHERE bookuid Like ?", (data,))

database.commit()

tkinter.messagebox.showinfo('Book Information', 'Successfully Deleted')

database.close()

book\_management = Toplevel()

book\_management.title("Book Entry From")

#book\_management.geometry('1024x768+500+100')

book\_management.maxsize()

# ===================== Frame ===========================

menu\_frame = Frame(book\_management, width=1024, height=18, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

text\_LMS = Label(menu\_frame, text="Book Management From", font=('arial', 20, 'bold')).pack()

frame1 = Frame(book\_management, width=100, height=750, relief='sunken')

frame2 = Frame(book\_management, width=462, height=700, relief='sunken')

frame3 = Frame(book\_management, width=462, height=700, relief='sunken')

frame1.pack(side='left')

frame2.pack(side='left')

frame3.pack(side='right')

# ============================== Lable and Entry =============================

bookName = Label(frame2, width=20, text="Book Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

WriterNme = Label(frame2, width=20, text="Writer Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookUid = Label(frame2, width=20, text="Book UID", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

NumberOfCopies = Label(frame2, width=20, text="Number Of Copies", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

prize = Label(frame2, width=20, text="Prize", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

publisher = Label(frame2, width=20, text="Publisher", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookCategory = Label(frame2, width=20, text="Book Category", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

control = Label(frame3, width=10, text="Controls", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2)

bookName\_Input = StringVar()

WriterNme\_Input = StringVar()

bookUid\_Input = IntVar()

NumberOfCopies\_Input = IntVar()

prize\_Input = IntVar()

publisher\_Input = StringVar()

bookCategory\_Input = StringVar()

BookName\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookName\_Input, bd=10)

WriterNme\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=WriterNme\_Input, bd=10)

bookUid\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookUid\_Input, bd=10)

NumberOfCopies\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=NumberOfCopies\_Input, bd=10)

prize\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=prize\_Input, bd=10)

publisher\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=publisher\_Input, bd=10)

bookCategory\_entry = Entry(frame2, font=('arial', 12, 'bold'), bg='#CCCCCC', fg='#000000', textvariable=bookCategory\_Input, bd=10)

bookUid\_Input.set("")

NumberOfCopies\_Input.set("")

prize\_Input.set("")

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

WriterNme.grid(row=1, column=0, padx=2, pady=2)

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

NumberOfCopies.grid(row=3, column=0, padx=2, pady=2)

prize.grid(row=4, column=0, padx=2, pady=2)

publisher.grid(row=5, column=0, padx=2, pady=2)

bookCategory.grid(row=6, column=0, padx=2, pady=2)

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

BookName\_entry.grid(row=0, column=1)

WriterNme\_entry.grid(row=1, column=1)

bookUid\_entry.grid(row=2, column=1)

NumberOfCopies\_entry.grid(row=3, column=1)

prize\_entry.grid(row=4, column=1)

publisher\_entry.grid(row=5, column=1)

bookCategory\_entry.grid(row=6, column=1)

# ================================== Button =====================================

search\_button = Button(frame3, width=11, text="Search",bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=searchdata)

clear\_button = Button(frame3, width=11, text="Clear",bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=clearfield)

delete\_button = Button(frame3, width=11, text="Delete",bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=deletedata)

go\_to\_bookentry\_button = Button(frame3, width=11, text="Book Entry",bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=go\_to\_bookentry)

exit\_button = Button(frame3, width=11, text="Exit",bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=exitbutt)

search\_button.grid(row=1, column=0)

clear\_button.grid(row=2, column=0)

delete\_button.grid(row=4, column=0)

go\_to\_bookentry\_button.grid(row=5, column=0)

exit\_button.grid(row=6, column=0)

def student\_add(self):

def exitbutt():

studentadd.destroy()

def savedata():

database = sqlite3.connect("LMS.sqlite")

cursor = database.execute("SELECT \* FROM student\_add")

data = roll\_Input.get()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (data,)):

if roll == data:

tkinter.messagebox.showinfo('Student Information', 'Student already exist')

break

else:

studentname = studentName\_Input.get()

address = address\_Input.get()

phoneno = phone\_no\_Input.get()

roll = roll\_Input.get()

department = department\_Input.get()

semester = semester\_Input.get()

shift = shift\_Input.get()

date = dateformat()

d = day\_Input.get()

m = month\_Input.get()

if int(d) >31:

tkinter.messagebox.showinfo('Student Information', 'Date is not correct')

elif int(m) >12:

tkinter.messagebox.showinfo('Student Information', 'Month is not correct')

else:

bookonhand = 0

cursor.execute("INSERT INTO student\_add VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?)", (studentname, address, phoneno, roll, department, semester, shift, date, bookonhand))

cursor.connection.commit()

cursor.close()

tkinter.messagebox.showinfo('Student Information', 'Student Saved Successfully')

def cleardata():

studentName\_Input.set("")

address\_Input.set("")

phone\_no\_Input.set("")

roll\_Input.set("")

department\_Input.set("")

semester\_Input.set("")

shift\_Input.set("")

day\_Input.set(0)

month\_Input.set(0)

year\_Input.set(2017)

photo = PhotoImage(file='student\_image.png')

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def dateformat():

d = day\_Input.get()

m = month\_Input.get()

y = year\_Input.get()

date\_Input = d + '/' + m + '/' + y

return date\_Input

def student\_image():

con = sqlite3.connect('imagedb.sqlite')

cur = con.cursor()

image\_id = roll\_Input.get()

image\_location = filedialog.askopenfilename()

cur.execute("INSERT INTO image (id, name) VALUES(?, ?)", (image\_id, image\_location))

photo = PhotoImage(file=image\_location)

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

cur.connection.commit()

cur.close()

def go\_to\_student\_management():

studentadd.destroy()

self.student\_management()

studentadd = Toplevel()

studentadd.title("Student Add From")

studentadd.maxsize()

#===================== Frame ===========================

menu\_frame = Frame(studentadd, width=1024, height=18, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

Label(menu\_frame, text="Student Add From", font=('arial', 30, 'bold')).pack()

frame1 = Frame(studentadd, width=100, height=750, relief='sunken')

frame2 = Frame(studentadd, width=462, height=700, relief='sunken')

frame3 = Frame(studentadd, width=462, height=700, relief='sunken')

frame1.pack(side='left')

frame2.pack(side='left')

frame3.pack(side='right')

#============================== Lable=============================

studentName = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Student Name:", font=('times new roman', 23, 'bold'), bd=2)

address = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Address:", font=('times new roman', 23, 'bold'), bd=2)

phone\_no = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Phone No:", font=('times new roman', 23, 'bold'), bd=2)

roll = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Roll:", font=('times new roman', 23, 'bold'), bd=2)

department = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Department:", font=('times new roman', 23, 'bold'), bd=2)

semester = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Semester:", font=('times new roman', 23, 'bold'), bd=2)

shift = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Shift:", font=('times new roman', 23, 'bold'), bd=2)

date = Label(frame2, width=20, bg='#666666', fg='#ffffff', text="Date:", font=('times new roman', 23, 'bold'), bd=2)

control = Label(frame3, width=10, text="Controls", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2)

studentName\_Input = StringVar()

address\_Input = StringVar()

phone\_no\_Input = StringVar()

roll\_Input = IntVar()

department\_Input = StringVar()

semester\_Input = StringVar()

shift\_Input = StringVar()

day\_Input = StringVar()

month\_Input = StringVar()

year\_Input = StringVar()

#==============================Entry and combobox =============================

studentName\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=studentName\_Input, bd=10)

address\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=address\_Input, bd=10)

phone\_no\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=phone\_no\_Input, bd=10)

roll\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=roll\_Input, bd=10)

department\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=department\_Input, value='Computer Civil Electrical Electronics Mechanical Power Elector-Mechanical Mechatonics ')

semester\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=semester\_Input, value='1st 2nd 3rd 4th 5th 6th 7th')

shift\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=shift\_Input, value='1st 2nd')

date\_frame = Frame(frame2)

daty\_label = Label(date\_frame, width=5, text="Day", font=('times new roman', 12, 'bold'), bd=2)

month\_label = Label(date\_frame, width=5, text="Month", font=('times new roman', 12, 'bold'), bd=2)

year\_label = Label(date\_frame, width=8, text="Year", font=('times new roman', 12, 'bold'), bd=2)

date\_spinbox = Spinbox(date\_frame, width=4, buttonup=RAISED, textvariable=day\_Input, from\_=0, to=31)

date\_month = Spinbox(date\_frame, width=4, buttonup=RAISED, textvariable=month\_Input, from\_=0, to=12)

date\_year = Spinbox(date\_frame, width=8, buttonup=RAISED, textvariable=year\_Input, from\_=2017, to=2099)

daty\_label.grid(row=0, column=0)

month\_label.grid(row=0, column=1)

year\_label.grid(row=0, column=2)

date\_spinbox.grid(row=1, column=0)

date\_month.grid(row=1, column=1)

date\_year.grid(row=1, column=2)

roll\_Input.set("")

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

address.grid(row=1, column=0, padx=2, pady=2)

phone\_no.grid(row=2, column=0, padx=2, pady=2)

roll.grid(row=3, column=0, padx=2, pady=2)

department.grid(row=4, column=0, padx=2, pady=2)

semester.grid(row=5, column=0, padx=2, pady=2)

shift.grid(row=6, column=0, padx=2, pady=2)

date.grid(row=7, column=0, padx=2, pady=2)

studentName\_entry.grid(row=0, column=1)

address\_entry.grid(row=1, column=1)

phone\_no\_entry.grid(row=2, column=1)

roll\_entry.grid(row=3, column=1)

department\_combobox.grid(row=4, column=1)

semester\_combobox.grid(row=5, column=1)

shift\_combobox.grid(row=6, column=1)

date\_frame.grid(row=7, column=1)

#================================== Button =====================================

clear\_button = Button(frame3, width=10, text="Clear", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=cleardata)

save\_button = Button(frame3, width=10, text="Save", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=savedata)

student\_management\_button = Button(frame3, width=10, bg='#BDCCD4', text="S M", font=('times new roman', 23, 'bold'), bd=2, command=go\_to\_student\_management)

exit\_button = Button(frame3, width=10, text="Exit", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=exitbutt)

image\_button = Button(frame3, width=10, text="Browse", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=student\_image)

#================================== Image Button ==========================================================

lbl\_frame = Frame(frame3, width=200, height=200, bg='pink')

photo = PhotoImage(file='student\_image.png')

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

lbl\_frame.grid(row=0, column=1)

control.grid(row=1, column=1)

image\_button.grid(row=2, column=1)

clear\_button.grid(row=3, column=1)

save\_button.grid(row=4, column=1)

student\_management\_button.grid(row=5, column=1)

exit\_button.grid(row=6, column=1)

def student\_management(self):

def exitbutt():

student\_management.destroy()

def cleardata():

studentName\_Input.set("")

address\_Input.set("")

phone\_no\_Input.set("")

roll\_Input.set("")

department\_Input.set("")

semester\_Input.set("")

shift\_Input.set("")

date\_Input.set("")

photo = PhotoImage(file='student\_image.png')

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def deletedata():

answer = tkinter.messagebox.askquestion('Delete confirmation', 'Do you want to delete?')

if answer == 'yes':

database = sqlite3.connect("LMS.sqlite")

data = roll\_Input.get()

database.execute("DELETE FROM student\_add WHERE roll Like ?", (data,))

con = sqlite3.connect('imagedb.sqlite')

con.execute("DELETE FROM image WHERE id Like ?", (data,))

database.commit()

con.commit()

database.close()

con.close()

tkinter.messagebox.showinfo('Student Information', 'Student Successfully Deleted')

def editdata():

database = sqlite3.connect("LMS.sqlite")

data = roll\_Input.get()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (data,)):

if roll == data:

studentname = studentName\_Input.get()

address = address\_Input.get()

phoneno = phone\_no\_Input.get()

roll = roll\_Input.get()

department = department\_Input.get()

semester = semester\_Input.get()

shift = shift\_Input.get()

date = date\_Input.get()

database.execute("UPDATE student\_add SET studentname = ? WHERE (roll = ?) ", (studentname, data))

database.execute("UPDATE student\_add SET address = ? WHERE (roll = ?) ", (address, data))

database.execute("UPDATE student\_add SET phoneno = ? WHERE (roll = ?) ", (phoneno, data))

database.execute("UPDATE student\_add SET roll = ? WHERE (roll = ?) ", (roll, data))

database.execute("UPDATE student\_add SET department = ? WHERE (roll = ?) ", (department, data))

database.execute("UPDATE student\_add SET semester = ? WHERE (roll = ?) ", (semester, data))

database.execute("UPDATE student\_add SET shift = ? WHERE (roll = ?) ", (shift, data))

database.execute("UPDATE student\_add SET date = ? WHERE (roll = ?) ", (date, data))

database.commit()

tkinter.messagebox.showinfo('Student Information', 'Successfully Updated')

database.close()

def searchdata():

database = sqlite3.connect("LMS.sqlite")

data = roll\_Input.get()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (data,)):

if roll == data:

studentName\_Input.set(studentname)

address\_Input.set(address)

phone\_no\_Input.set(phoneno)

roll\_Input.set(roll)

department\_Input.set(department)

semester\_Input.set(semester)

shift\_Input.set(shift)

date\_Input.set(date)

student\_image()

tkinter.messagebox.showinfo('Student Information', 'Student Found')

break

else:

tkinter.messagebox.showinfo('Student Information', 'Student Not Found')

database.close()

def student\_image():

con = sqlite3.connect('imagedb.sqlite')

cur = con.cursor()

image\_id = roll\_Input.get()

for id, name in con.execute("SELECT \* FROM image WHERE id Like ?", (image\_id,)):

if id == image\_id:

photo = PhotoImage(file=name)

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

cur.close()

def go\_to\_student\_add():

student\_management.destroy()

self.student\_add()

student\_management = Toplevel()

student\_management.title("Student Add From")

student\_management.maxsize()

#===================== Frame ===========================

menu\_frame = Frame(student\_management, width=1024, height=18, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

Label(menu\_frame, text="Student Management From", font=('arial', 20, 'bold') ).pack()

frame1 = Frame(student\_management, width=100, height=750, relief='sunken')

frame2 = Frame(student\_management, width=462, height=700, relief='sunken')

frame3 = Frame(student\_management, width=462, height=700, relief='sunken')

frame1.pack(side='left')

frame2.pack(side='left')

frame3.pack(side='right')

#============================== Lable and Entry =============================

studentName = Label(frame2, width=20, text="Student Name:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

address = Label(frame2, width=20, text="Address:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

phone\_no = Label(frame2, width=20, text="Phone No:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

roll = Label(frame2, width=20, text="Roll:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

department = Label(frame2, width=20, text="Department:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

semester = Label(frame2, width=20, text="semester:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

shift = Label(frame2, width=20, text="shift:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

date = Label(frame2, width=20, text="date:", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

control = Label(frame3, width=12, text="Controls", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2)

studentName\_Input = StringVar()

address\_Input = StringVar()

phone\_no\_Input = StringVar()

roll\_Input = IntVar()

department\_Input = StringVar()

semester\_Input = StringVar()

shift\_Input = StringVar()

date\_Input = StringVar()

studentName\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=studentName\_Input, bd=10)

address\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=address\_Input, bd=10)

phone\_no\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=phone\_no\_Input, bd=10)

roll\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=roll\_Input, bd=5)

department\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=department\_Input, value='Computer Civil Electrical Electronics Mechanical Power Elector-Mechanical Mechatonics ')

semester\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=semester\_Input, value='1st 2nd 3rd 4th 5th 6th 7th')

shift\_combobox = Combobox(frame2, font=('arial', 12, 'bold'), textvariable=shift\_Input, value='1st 2nd')

date\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=date\_Input, bd=10)

roll\_Input.set("")

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

address.grid(row=1, column=0, padx=2, pady=2)

phone\_no.grid(row=2, column=0, padx=2, pady=2)

roll.grid(row=3, column=0, padx=2, pady=2)

department.grid(row=4, column=0, padx=2, pady=2)

semester.grid(row=5, column=0, padx=2, pady=2)

shift.grid(row=6, column=0, padx=2, pady=2)

date.grid(row=7, column=0, padx=2, pady=2)

studentName\_entry.grid(row=0, column=1)

address\_entry.grid(row=1, column=1)

phone\_no\_entry.grid(row=2, column=1)

roll\_entry.grid(row=3, column=1)

department\_combobox.grid(row=4, column=1)

semester\_combobox.grid(row=5, column=1)

shift\_combobox.grid(row=6, column=1)

date\_entry.grid(row=7, column=1)

#================================== Button =====================================

search\_button = Button(frame3, width=10, text="Search", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=searchdata)

clear\_button = Button(frame3, width=10, text="Clear", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=cleardata)

delete\_button = Button(frame3, width=10, text="Delete", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=deletedata)

edit\_button = Button(frame3, width=10, text="Update", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=editdata)

go\_to\_student\_add\_button = Button(frame3, width=10, bg='#BDCCD4', text="S A", font=('times new roman', 23, 'bold'), bd=2, command=go\_to\_student\_add)

exit\_button = Button(frame3, width=10, text="Exit", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=exitbutt)

#================================== Image Button ==========================================================

lbl\_frame = Frame(frame3, width=200, height=200, bg='pink')

photo = PhotoImage(file='student\_image.png')

lbl\_img = Label(lbl\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

lbl\_frame.grid(row=0, column=1)

control.grid(row=1, column=1)

search\_button.grid(row=2, column=1)

clear\_button.grid(row=3, column=1)

delete\_button.grid(row=4, column=1)

edit\_button.grid(row=5, column=1)

go\_to\_student\_add\_button.grid(row=6, column=1)

exit\_button.grid(row=7, column=1)

def issue\_book(self):

def exitbutt():

bookissue.destroy()

def cleardata():

studentName\_Input.set("")

roll\_Input.set("")

bookName\_Input.set("")

writerName\_Input.set("")

bookUid\_Input.set("")

prize\_Input.set("")

publisher\_Input.set("")

issueDate\_Input.set("")

returnDate\_Input.set("")

day\_Input.set(0)

month\_Input.set(0)

year\_Input.set(2000)

day\_Input2.set(0)

month\_Input2.set(0)

year\_Input2.set(2000)

def searchdata\_roll():

database = sqlite3.connect("LMS.sqlite")

data = roll\_Input.get()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (data,)):

if roll == data:

studentName\_Input.set(studentname)

roll\_Input.set(roll)

database.close()

tkinter.messagebox.showinfo('Student Information', 'Student Found')

def searchdata\_book():

database = sqlite3.connect("LMS.sqlite")

data2 = bookUid\_Input.get()

for bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory in database.execute("SELECT \* FROM book\_entry WHERE bookuid Like ?", (data2,)):

if bookuid == data2:

bookName\_Input.set(bookname)

writerName\_Input.set(writername)

bookUid\_Input.set(bookuid)

prize\_Input.set(prize)

publisher\_Input.set(publisher)

database.close()

tkinter.messagebox.showinfo('Book Information', 'Book Found')

def issuebook():

database = sqlite3.connect("LMS.sqlite")

databook = bookUid\_Input.get()

datastudent = roll\_Input.get()

d = day\_Input.get()

m = month\_Input.get()

d2 = day\_Input2.get()

m2 = month\_Input2.get()

if int(d) >31:

tkinter.messagebox.showinfo('Student Information', 'Date is not correct')

elif int(m) >12:

tkinter.messagebox.showinfo('Student Information', 'Month is not correct')

elif int(d2) >31:

tkinter.messagebox.showinfo('Student Information', 'Date is not correct')

elif int(m2) >12:

tkinter.messagebox.showinfo('Student Information', 'Month is not correct')

else:

for bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory in database.execute("SELECT \* FROM book\_entry WHERE bookuid Like ?", (databook,)):

if bookuid == databook:

if numberofcopies <=0:

tkinter.messagebox.showinfo('Book Information', 'Book out of stock')

break

else:

newbookvalue = numberofcopies - 1

database.execute("UPDATE book\_entry SET numberofcopies = ? WHERE (bookuid = ?) ", (newbookvalue, databook))

database.commit()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (datastudent,)):

if roll == datastudent:

if bookonhand >= 2:

tkinter.messagebox.showinfo('Student Information', 'Student reached maximum book laval on hand')

break

else:

newstudentvalue = bookonhand + 1

database.execute("UPDATE student\_add SET bookonhand = ? WHERE (roll = ?) ", (newstudentvalue, datastudent))

cursor = database.execute("SELECT \* FROM issuebook")

studentname = studentName\_Input.get()

roll = roll\_Input.get()

bookname = bookName\_Input.get()

writername = writerName\_Input.get()

bookuid = bookUid\_Input.get()

prize = prize\_Input.get()

publisher = publisher\_Input.get()

issuedate = dateformat()

returndate = dateformat2()

cursor.execute("INSERT INTO issuebook VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?)",

(studentname, roll, bookname, writername, bookuid, prize, publisher, issuedate, returndate))

cursor.connection.commit()

tkinter.messagebox.showinfo('Book Information', 'Book Successfully Issued')

database.commit()

database.close()

def dateformat():

d = day\_Input.get()

m = month\_Input.get()

y = year\_Input.get()

date\_Input = d + '/' + m + '/' + y

return date\_Input

def dateformat2():

d = day\_Input2.get()

m = month\_Input2.get()

y = year\_Input2.get()

date\_Input2 = d + '/' + m + '/' + y

return date\_Input2

bookissue = Toplevel()

bookissue.title("Book Issue From")

bookissue.maxsize()

#===================== Frame ===========================

menu\_frame = Frame(bookissue, width=1024, height=18, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

Label(menu\_frame, text="Book Issue From", font=('arial', 20, 'bold') ).pack()

frame1 = Frame(bookissue, width=100, height=750, relief='sunken')

frame2 = Frame(bookissue, width=462, height=700, relief='sunken')

frame3 = Frame(bookissue, width=462, height=700, bg='red', relief='sunken')

frame1.pack(side='left')

frame2.pack(side='left')

frame3.pack(side='right')

#============================== Lable and Entry =============================

studentName = Label(frame2, width=20, text="Student Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

roll = Label(frame2, width=20, text="Roll", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookName = Label(frame2, width=20, text="Book Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

WriterNme = Label(frame2, width=20, text="Writer Name", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookUid = Label(frame2, width=20, text="Book UID", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

prize = Label(frame2, width=20, text="Price", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

publisher = Label(frame2, width=20, text="Publisher", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

issueDate = Label(frame2, width=20, text="Issue Date", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

returnDate = Label(frame2, width=20, text="return Date", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

studentName\_Input = StringVar()

roll\_Input = IntVar()

bookName\_Input = StringVar()

writerName\_Input = StringVar()

bookUid\_Input = IntVar()

prize\_Input = IntVar()

publisher\_Input = StringVar()

issueDate\_Input = StringVar()

returnDate\_Input = StringVar()

day\_Input = StringVar()

month\_Input = StringVar()

year\_Input = StringVar()

day\_Input2 = StringVar()

month\_Input2 = StringVar()

year\_Input2 = StringVar()

studentName\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=studentName\_Input, bd=10)

roll\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=roll\_Input, bd=10)

bookName\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=bookName\_Input, bd=10)

WriterNme\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=writerName\_Input, bd=10)

bookUid\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=bookUid\_Input, bd=10)

prize\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=prize\_Input, bd=10)

publisher\_entry = Entry(frame2, font=('arial', 12, 'bold'), textvariable=publisher\_Input, bd=10)

#============================================== Date spinbox ============================================================

issueDate\_frame = Frame(frame2)

daty\_label = Label(issueDate\_frame, width=5, text="Day", font=('times new roman', 12, 'bold'), bd=2)

month\_label = Label(issueDate\_frame, width=5, text="Month", font=('times new roman', 12, 'bold'), bd=2)

year\_label = Label(issueDate\_frame, width=8, text="Year", font=('times new roman', 12, 'bold'), bd=2)

date\_spinbox = Spinbox(issueDate\_frame, width=4, buttonup=RAISED, textvariable=day\_Input, from\_=0, to=31)

date\_month = Spinbox(issueDate\_frame, width=4, buttonup=RAISED, textvariable=month\_Input, from\_=0, to=12)

date\_year = Spinbox(issueDate\_frame, width=8, buttonup=RAISED, textvariable=year\_Input, from\_=2017, to=2099)

daty\_label.grid(row=0, column=0)

month\_label.grid(row=0, column=1)

year\_label.grid(row=0, column=2)

date\_spinbox.grid(row=1, column=0)

date\_month.grid(row=1, column=1)

date\_year.grid(row=1, column=2)

returnDate\_frame = Frame(frame2)

daty\_label = Label(returnDate\_frame, width=5, text="Day", font=('times new roman', 12, 'bold'), bd=2)

month\_label = Label(returnDate\_frame, width=5, text="Month", font=('times new roman', 12, 'bold'), bd=2)

year\_label = Label(returnDate\_frame, width=8, text="Year", font=('times new roman', 12, 'bold'), bd=2)

date\_spinbox = Spinbox(returnDate\_frame, width=4, buttonup=RAISED, textvariable=day\_Input2, from\_=0, to=31)

date\_month = Spinbox(returnDate\_frame, width=4, buttonup=RAISED, textvariable=month\_Input2, from\_=0, to=12)

date\_year = Spinbox(returnDate\_frame, width=8, buttonup=RAISED, textvariable=year\_Input2, from\_=2017, to=2099)

daty\_label.grid(row=0, column=0)

month\_label.grid(row=0, column=1)

year\_label.grid(row=0, column=2)

date\_spinbox.grid(row=1, column=0)

date\_month.grid(row=1, column=1)

date\_year.grid(row=1, column=2)

roll\_Input.set("")

bookUid\_Input.set("")

prize\_Input.set("")

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

roll.grid(row=1, column=0, padx=2, pady=2)

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

WriterNme.grid(row=3, column=0, padx=2, pady=2)

bookUid.grid(row=4, column=0, padx=2, pady=2)

prize.grid(row=5, column=0, padx=2, pady=2)

publisher.grid(row=6, column=0, padx=2, pady=2)

issueDate.grid(row=7, column=0, padx=2, pady=2)

returnDate.grid(row=8, column=0, padx=2, pady=2)

studentName\_entry.grid(row=0, column=1)

roll\_entry.grid(row=1, column=1)

bookName\_entry.grid(row=2, column=1)

WriterNme\_entry.grid(row=3, column=1)

bookUid\_entry.grid(row=4, column=1)

prize\_entry.grid(row=5, column=1)

publisher\_entry.grid(row=6, column=1)

issueDate\_frame.grid(row=7, column=1)

returnDate\_frame.grid(row=8, column=1)

#================================== Button =====================================

searchrool\_button = Button(frame3, width=10, text="Search Roll", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=searchdata\_roll)

searchbookuid\_button = Button(frame3, width=10, text="Search Book", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=searchdata\_book)

clear\_button = Button(frame3, width=10, text="Clear", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=cleardata)

issue\_button = Button(frame3, width=10, text="Issue", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=issuebook)

exit\_button = Button(frame3, width=10, text="Exit", bg='#BDCCD4', font=('times new roman', 23, 'bold'), bd=2, command=exitbutt)

searchrool\_button.grid(row=0, column=1)

searchbookuid\_button.grid(row=1, column=1)

clear\_button.grid(row=2, column=1)

issue\_button.grid(row=3, column=1)

exit\_button.grid(row=4, column=1)

def return\_from(self):

def exitreturnbook():

returnbook.destroy()

def cleardata():

roll\_Input.set("")

bookUid\_Input.set("")

returndate\_Input.set("")

fine\_Input.set("")

rate\_Input.set("")

day\_Input.set(0)

month\_Input.set(0)

year\_Input.set(2017)

def bookreturn():

database = sqlite3.connect("LMS.sqlite")

databook = bookUid\_Input.get()

datastudent = roll\_Input.get()

for studentname, address, phoneno, roll, department, semester, shift, date, bookonhand in database.execute("SELECT \* FROM student\_add WHERE roll Like ?", (datastudent,)):

if roll == datastudent:

if bookonhand <=0:

tkinter.messagebox.showinfo('Book Information', 'No Book on hand !')

else:

newstudentvalue = bookonhand - 1

database.execute("UPDATE student\_add SET bookonhand = ? WHERE (roll = ?) ", (newstudentvalue, datastudent))

for bookname, writername, bookuid, numberofcopies, prize, publisher, bookcategory in database.execute("SELECT \* FROM book\_entry WHERE bookuid Like ?", (databook,)):

if bookuid == databook:

newbookvalue = numberofcopies + 1

database.execute("UPDATE book\_entry SET numberofcopies = ? WHERE (bookuid = ?) ", (newbookvalue, databook))

database.commit()

tkinter.messagebox.showinfo('Book Information', 'Book successfully returned')

cleardata()

database.commit()

database.close()

def finecalculation():

database = sqlite3.connect("LMS.sqlite")

databook = bookUid\_Input.get()

rate = rate\_Input.get()

for studentname, roll, bookname, writername, bookuid, prize, publisher, issuedate, returndate in database.execute("SELECT \* FROM issuebook WHERE bookuid Like ?", (databook,)):

if bookuid == databook:

date\_format = "%d/%m/%Y"

date\_stat = returndate

date\_end = dateformat()

date\_start = datetime.strptime(date\_stat, date\_format)

date\_end = datetime.strptime(date\_end, date\_format)

fine = date\_end - date\_start

fine\_Input.set(fine.days\*rate)

def dateformat():

d = day\_Input.get()

m = month\_Input.get()

y = year\_Input.get()

date\_Input = d + '/' + m + '/' + y

return date\_Input

returnbook = Toplevel()

returnbook.title("Return Book From")

#returnbook.geometry('650x380+600+300')

returnbook.maxsize()

menu\_frame = Frame(returnbook, width=550, height=50, bg='pink', relief='sunken')

menu\_frame.pack(side='top')

text\_LMS = Label(menu\_frame, text="Book Return From", font=('arial', 20, 'bold') ).pack()

frame1 = Frame(returnbook, width=50, height=700, relief='sunken')

frame2 = Frame(returnbook, width=500, height=700, relief='sunken')

frame1.pack(side='left')

frame2.pack(side='right')

#============================== Lable and Entry =============================

roll = Label(frame2, width=20, text="Roll", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

bookUid = Label(frame2, width=20, text="Book Uid", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

returndate = Label(frame2, width=20, text="Return Date", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

fine = Label(frame2, width=20, text="Fine", bg='#666666', fg='#ffffff',font=('times new roman', 23, 'bold'), bd=2)

rate = Label(frame2, width=20, text="Rate", bg='#666666', fg='#ffffff', font=('times new roman', 23, 'bold'), bd=2)

roll\_Input = IntVar()

bookUid\_Input = IntVar()

returndate\_Input = StringVar()

fine\_Input = IntVar()

rate\_Input = IntVar()

roll\_entry = Entry(frame2, font=('times new roman', 18, 'bold'), textvariable=roll\_Input, bd=10)

bookUid\_entry = Entry(frame2, font=('times new roman', 18, 'bold'), textvariable=bookUid\_Input, bd=10)

fine\_entry = Entry(frame2,font=('times new roman', 18, 'bold'), textvariable=fine\_Input, bd=10)

rate\_entry = Entry(frame2,font=('times new roman', 18, 'bold'), textvariable=rate\_Input, bd=10)

day\_Input = StringVar()

month\_Input = StringVar()

year\_Input = StringVar()

date\_frame = Frame(frame2)

date\_spinbox = Spinbox(date\_frame, width=4, buttonup=RAISED, bg='#CCCCCC', fg='#000000', textvariable=day\_Input, from\_=0, to=31)

date\_month = Spinbox(date\_frame, width=4, buttonup=RAISED, bg='#CCCCCC', fg='#000000', textvariable=month\_Input, from\_=0, to=12)

date\_year = Spinbox(date\_frame, width=8, buttonup=RAISED, bg='#CCCCCC', fg='#000000', textvariable=year\_Input, from\_=2017, to=2099)

date\_spinbox.grid(row=0, column=0)

date\_month.grid(row=0, column=1)

date\_year.grid(row=0, column=2)

roll\_Input.set("")

bookUid\_Input.set("")

returndate\_Input.set("")

fine\_Input.set("")

rate\_Input.set("")

roll.grid(row=0, column=0)

bookUid.grid(row=1, column=0)

returndate.grid(row=2, column=0)

rate.grid(row=3, column=0)

fine.grid(row=4, column=0)

button\_frame = Frame(frame2)

button\_frame.grid(row=5, columnspan=2)

roll\_entry.grid(row=0, column=1)

bookUid\_entry.grid(row=1, column=1)

date\_frame.grid(row=2, column=1)

rate\_entry.grid(row=3, column=1)

fine\_entry.grid(row=4, column=1)

#=========================== Button ============================================

returnButton = Button(button\_frame, width=10, text="Return", bg='#BDCCD4', font=('arial', 20, 'bold'), bd=2, command=bookreturn)

close = Button(button\_frame, width=10, text="close", bg='#BDCCD4', font=('arial', 20, 'bold'), bd=2, command=exitreturnbook)

fine = Button(button\_frame, width=10, text="Fine", bg='#BDCCD4', font=('arial', 20, 'bold'), bd=2, command=finecalculation)

returnButton.grid(row=0, column=0)

fine.grid(row=0, column=1)

close.grid(row=0, column=2)

def about\_from(self):

def cleardata():

studentName\_Input.set("")

roll\_Input.set("")

department\_Input.set("")

shift\_Input.set("")

semester\_Input.set("")

def touhid():

name = "Touhidur Rahman"

roll = "Roll: 808516"

department = "Department: Computer"

shift = "Shift: 1st"

semester = "Semester: 7th"

studentName\_Input.set(name)

roll\_Input.set(roll)

department\_Input.set(department)

shift\_Input.set(shift)

semester\_Input.set(semester)

photo = PhotoImage(file='image\Touhid.png')

lbl\_img = Label(image\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def subrato():

cleardata()

name = "Subroto Roy"

roll = "Roll: 868199"

department = "Department: Computer"

shift = "Shift: 1st"

semester = "Semester: 7th"

studentName\_Input.set(name)

roll\_Input.set(roll)

department\_Input.set(department)

shift\_Input.set(shift)

semester\_Input.set(semester)

photo = PhotoImage(file = 'image\Subrato.png')

lbl\_img = Label(image\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def raisul():

cleardata()

name = "Raisul Islam"

roll = "Roll: 808505"

department = "Department: Computer"

shift = "Shift: 1st"

semester = "Semester: 7th"

studentName\_Input.set(name)

roll\_Input.set(roll)

department\_Input.set(department)

shift\_Input.set(shift)

semester\_Input.set(semester)

photo = PhotoImage(file = "image\Raisul.png")

lbl\_img = Label(image\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def nishan():

cleardata()

name = "Nishan Ahmed"

roll = "Roll: 808519"

department = "Department: Computer"

shift = "Shift: 1st"

semester = "Semester: 7th"

studentName\_Input.set(name)

roll\_Input.set(roll)

department\_Input.set(department)

shift\_Input.set(shift)

semester\_Input.set(semester)

photo = PhotoImage(file = "image\808519.png")

lbl\_img = Label(image\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

def tanisha():

cleardata()

name = "Tanisha Akter"

roll = "Roll: 808525"

department = "Department: Computer"

shift = "Shift: 1st"

semester = "Semester: 7th"

studentName\_Input.set(name)

roll\_Input.set(roll)

department\_Input.set(department)

shift\_Input.set(shift)

semester\_Input.set(semester)

photo = PhotoImage(file = "image\Tanisha.png")

lbl\_img = Label(image\_frame, width=200, height=250, image=photo)

lbl\_img.image = photo

lbl\_img.grid(row=0, column=0)

about = Toplevel()

about.title("About US")

about.geometry("800x650+500+300")

Label(about, text="ABOUT US", font=('arial', 20, 'bold')).pack(side='top')

frame1 = Frame(about, width=600, height=430, relief='sunken')

frame1.pack(side='top')

frame2 = Frame(about, width=300, height=350)

frame2.pack(side='bottom')

image\_frame = Frame(about, width=200, height=250)

image\_frame.pack(side='top')

touhid\_button = Button(frame1, text="Team leader Touhid", width=20, font=('times new roman', 12, 'bold italic'), activeforeground='green', bd=2, command=touhid)

subrato\_button = Button(frame1, text="Subrato Roy", width=20, font=('times new roman', 12, 'bold italic'), activeforeground='pink', bd=2, command=subrato)

raisul\_button = Button(frame1, text="Raisul Islam", width=20, font=('times new roman', 12, 'bold italic'), activeforeground='blue', bd=2, command=raisul)

nishan\_button = Button(frame1, text="Nishan Ahmed", width=20, font=('times new roman', 12, 'bold italic'), activeforeground='blue', bd=2, command=nishan)

tanisha\_button = Button(frame1, text="Tanisha Akter", width=20, font=('times new roman', 12, 'bold italic'), activeforeground='blue', bd=2, command=tanisha)

touhid\_button.grid(row=0, column=0)

subrato\_button.grid(row=0, column=1)

raisul\_button.grid(row=0, column=2)

nishan\_button.grid(row=0, column=3)

tanisha\_button.grid(row=1, columnspan=4)

studentName\_Input = StringVar()

roll\_Input = StringVar()

department\_Input = StringVar()

shift\_Input = StringVar()

semester\_Input = StringVar()

name\_entry = Entry(frame2, font=('times new roman', 20, 'bold italic'), state='readonly', textvariable=studentName\_Input, relief=RAISED)

roll\_entry = Entry(frame2, font=('times new roman', 20, 'bold italic'), state='readonly', textvariable=roll\_Input, relief=RAISED)

department\_entry = Entry(frame2, font=('times new roman', 20, 'bold italic'), state='readonly', textvariable=department\_Input, relief=RAISED)

shift\_entry = Entry(frame2, font=('times new roman', 20, 'bold italic'), state='readonly', textvariable=shift\_Input, relief=RAISED)

semester\_entry = Entry(frame2, font=('times new roman', 20, 'bold italic'), state='readonly', textvariable=semester\_Input, relief=RAISED)

name\_entry.grid(row=0, column=0)

roll\_entry.grid(row=1, column=0)

department\_entry.grid(row=2, column=0)

shift\_entry.grid(row=3, column=0)

semester\_entry.grid(row=4, column=0)

if \_\_name\_\_ == "\_\_main\_\_":

login = Tk()

app = Main\_window(login)

login.mainloop()