INT 213 PYTHON PROJECT REPORT ON:

CAR/CAB BOOKING SYSTEM

LOVELY PROFESSIONAL UNIVERSITY

Submitted by:
Mir Munazir Hassan
11913937
Roll No. 19

Submitted to: **Mrs. Neha Bagga**

About Project

In order to book an appointment, the user has to enter the owner’s name, age, gender, location, entry time and phone number. To view all the available appointment details, the user just has to click on the “Next Booking” button on the Display module. Remaining feature is about updating and deleting records. Simply by entering owner’s name, the user can edit and update the details. The design is so simple that the user won’t find any difficulties while working on it.

System Requirements:

- A PC running Mac OS, Windows or Linux OS
- Python Interpretor
- Tkinter package

The Actual Code

Importing Tkinter Module

In the code given below, which is for the function for importing tkinter. Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications.

```
from tkinter import
```

Importing the Sqlite3 Module

In the code given below, which is for the function for importing SQLite3 . SQLite is a C library that provides a lightweight disk-based database that doesn't require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language.

```
import sqlite3
```

This module is for the appointment

In the code given below, which is for the function for appointment. Likewise, you can see the frames in the windows, the labels, the button use to perform command , entries for all the labels, and the function to call when the submit button is clicked. Getting the value of user inputs, checking if the user input is empty and you can see the displaying the logs on the right frame.

```
# import modules
from tkinter import *
import sqlite3
import tkinter.messagebox
# DB connection
conn = sqlite3.connect('database.db')
# cursor to move around the databse
c = conn.cursor()
# empty list to later append the ids from the database
ids = []
class Application:
    def __init__(self, master):
        self.master = master

        # creating the frames in the master
        self.left = Frame(master, width=800, height=720, bg='grey')
        self.left.pack(side=LEFT)

        self.right = Frame(master, width=600, height=720, bg='grey')
        self.right.pack(side=RIGHT)

        # labels for the window
        self.heading = Label(self.left, text="Car Booking", font=(
            'georgia 40 bold'), fg='black', bg='grey')
        self.heading.place(x=0, y=0)
        # patients name
        self.name = Label(self.left, text="Owner's Name", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.name.place(x=0, y=100)

        # age
        self.age = Label(self.left, text="Age", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.age.place(x=0, y=140)

        # gender
        self.gender = Label(self.left, text="Gender", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.gender.place(x=0, y=180)

        # location
        self.location = Label(self.left, text="Location", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.location.place(x=0, y=220)

        # appointment time
        self.time = Label(self.left, text="Entry Time", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.time.place(x=0, y=260)

        # phone
        self.phone = Label(self.left, text="Phone Number", font=(
            'georgia 18 bold'), fg='black', bg='grey')
        self.phone.place(x=0, y=300)

        # Entries for all labels==========================================================
        self.name_ent = Entry(self.left, width=30)
        self.name_ent.place(x=250, y=100)
```

```
        self.age_ent = Entry(self.left, width=30)
        self.age_ent.place(x=250, y=140)

        self.gender_ent = Entry(self.left, width=30)
        self.gender_ent.place(x=250, y=180)

        self.location_ent = Entry(self.left, width=30)
        self.location_ent.place(x=250, y=220)

        self.time_ent = Entry(self.left, width=30)
        self.time_ent.place(x=250, y=260)

        self.phone_ent = Entry(self.left, width=30)
        self.phone_ent.place(x=250, y=300)

        # button to perform a command
        self.submit = Button(self.left, text="Add Booking", width=20,
                             height=2, bg='white', command=self.add_appointment)
        self.submit.place(x=300, y=340)

        # getting the number of appointments fixed to view in the log
        sql2 = "SELECT ID FROM appointments "
        self.result = c.execute(sql2)
        for self.row in self.result:
            self.id = self.row[0]
            ids.append(self.id)

        # ordering the ids
        self.new = sorted(ids)
        self.final_id = self.new[len(ids)-1]
        # displaying the logs in our right frame
        self.logs = Label(self.right, text="Booking logs", font=(
            'georgia 28 bold'), fg='black', bg='grey')
        self.logs.place(x=70, y=10)

        self.box = Text(self.right, width=150, height=40)
        self.box.place(x=20, y=60)
        self.box.insert(END, "Total Bookings till now : " +
                        str(self.final_id) + " \n")
    # funtion to call when the submit button is clicked

    def add_appointment(self):
        # getting the user inputs
        self.val1 = self.name_ent.get()
        self.val2 = self.age_ent.get()
        self.val3 = self.gender_ent.get()
        self.val4 = self.location_ent.get()
        self.val5 = self.time_ent.get()
        self.val6 = self.phone_ent.get()

        # checking if the user input is empty
        if self.val1 == '' or self.val2 == '' or self.val3 == '' or self.val4 == '' or self.val5 == '':
            tkinter.messagebox.showinfo("Warning", "Please Fill Up All Boxes")
        else:
            # now we add to the database
            sql = "INSERT INTO 'appointments' (name, age, gender, location, scheduled_time, phone) VALUES(?, ?, ?, ?, ?, ?)"
            c.execute(sql, (self.val1, self.val2, self.val3,
                            self.val4, self.val5, self.val6))
            conn.commit()
            tkinter.messagebox.showinfo(
                "Success", "Booking for " + str(self.val1) + " has been created")
            self.box.insert(END, 'Booking fixed for ' +
                            str(self.val1) + ' at ' + str(self.val5))


# creating the object
root = Tk()
b = Application(root)
```

```
# resolution of the window
root.geometry("1366x768")

# preventing the resize feature
root.resizable(False, False)

# end the loop
root.mainloop()
```

This module is for the booking

In the code given below, which is for the function for car booking. Also you can see the title, label, text entry, search button, delete button, creating the update form, declaring the variables to update and the delete function for the appointment.

```
# update the appointments
from tkinter import *
import tkinter.messagebox
import sqlite3
conn = sqlite3.connect('database.db')
c = conn.cursor()

class Application:
    def __init__(self, master):
        self.master = master
        # heading label
        self.heading = Label(master, text="Bookings ",  fg='grey', font=('arial 40 bold'))
        self.heading.place(x=150, y=20)

        # search criteria -->name
        self.name = Label(master, text="Enter Owner's Name", font=('arial 18 bold'))
        self.name.place(x=0, y=100)

        # entry for  the name
        self.namenet = Entry(master, width=30)
        self.namenet.place(x=280, y=92)

        # search button
        self.search = Button(master, text="Search", width=12, height=1, bg='steelblue', command=self.search_db)
        self.search.place(x=350, y=132)
    # function to search
    def search_db(self):
        self.input = self.namenet.get()
        # execute sql

        sql = "SELECT * FROM appointments WHERE name LIKE ?"
        self.res = c.execute(sql, (self.input,))
        for self.row in self.res:
            self.name1 = self.row[1]
            self.age = self.row[2]
            self.gender = self.row[3]
            self.location = self.row[4]
            self.time = self.row[6]
            self.phone = self.row[5]
        # creating the update form
        self.uname = Label(self.master, text="Owner's Name", font=('arial 18 bold'))
        self.uname.place(x=0, y=160)

        self.uage = Label(self.master, text="Age", font=('arial 18 bold'))
        self.uage.place(x=0, y=200)

        self.ugender = Label(self.master, text="Gender", font=('arial 18 bold'))
        self.ugender.place(x=0, y=240)

        self.ulocation = Label(self.master, text="Location", font=('arial 18 bold'))
        self.ulocation.place(x=0, y=280)

        self.utime = Label(self.master, text="Entry Time", font=('arial 18 bold'))
```

```
        self.utime.place(x=0, y=320)

        self.uphone = Label(self.master, text="Phone Number", font=('arial 18 bold'))
        self.uphone.place(x=0, y=360)

        # entries for each labels=========================================================
        # ====================filling the search result in the entry box to update
        self.ent1 = Entry(self.master, width=30)
        self.ent1.place(x=300, y=170)
        self.ent1.insert(END, str(self.name1))

        self.ent2 = Entry(self.master, width=30)
        self.ent2.place(x=300, y=210)
        self.ent2.insert(END, str(self.age))

        self.ent3 = Entry(self.master, width=30)
        self.ent3.place(x=300, y=250)
        self.ent3.insert(END, str(self.gender))

        self.ent4 = Entry(self.master, width=30)
        self.ent4.place(x=300, y=290)
        self.ent4.insert(END, str(self.location))

        self.ent5 = Entry(self.master, width=30)
        self.ent5.place(x=300, y=330)
        self.ent5.insert(END, str(self.time))

        self.ent6 = Entry(self.master, width=30)
        self.ent6.place(x=300, y=370)
        self.ent6.insert(END, str(self.phone))

        # button to execute update
        self.update = Button(self.master, text="Update?", width=20, height=2,fg='white', bg='black', command=self.update_db)
        self.update.place(x=400, y=410)

        # button to delete
        self.delete = Button(self.master, text="Delete?", width=20, height=2,fg='white', bg='black', command=self.delete_db)
        self.delete.place(x=150, y=410)
    def update_db(self):
        # declaring the variables to update
        self.var1 = self.ent1.get() #updated name
        self.var2 = self.ent2.get() #updated age
        self.var3 = self.ent3.get() #updated gender
        self.var4 = self.ent4.get() #updated location
        self.var5 = self.ent5.get() #updated phone
        self.var6 = self.ent6.get() #updated time
      query = "UPDATE appointments SET name=?, age=?, gender=?, location=?, phone=?, scheduled_time=? WHERE name LIKE
?"
        c.execute(query, (self.var1, self.var2, self.var3, self.var4, self.var5, self.var6, self.namenet.get(),))
        conn.commit()
        tkinter.messagebox.showinfo("Updated", "Successfully Updated.")
    def delete_db(self):
        # delete the appointment
        sql2 = "DELETE FROM appointments WHERE name LIKE ?"
        c.execute(sql2, (self.namenet.get(),))
        conn.commit()
        tkinter.messagebox.showinfo("Success", "Deleted Successfully")
        self.ent1.destroy()
        self.ent2.destroy()
        self.ent3.destroy()
        self.ent4.destroy()
        self.ent5.destroy()
        self.ent6.destroy()
# creating the object
root = Tk()
b = Application(root)
root.geometry("1366x768+0+0")
root.resizable(False, False)
root.mainloop()
```

This module is for the display bookings

In the code given below which is for the function for display bookings, you can see the title, a button to replace or change the bookings and the function to speak the text and update the text and the connection to database use in the system.

```
from tkinter import *
import sqlite3
import pyttsx3
# connection to database
conn = sqlite3.connect('database.db')
c = conn.cursor()
# empty lists to append later
number = []
patients = []
sql = "SELECT * FROM appointments"
res = c.execute(sql)
for r in res:
    ids = r[0]
    name = r[1]
    number.append(ids)
    patients.append(name)

# window
class Application:
    def __init__(self, master):
        self.master = master

        self.x = 0

        # heading
        self.heading = Label(master, text="Bookings", font=('arial 60 bold'), fg='green')
        self.heading.place(x=350, y=0)

        # button to change bookings
        self.change = Button(master, text="Next Booking", width=25, height=2, bg='steelblue', command=self.func)
        self.change.place(x=500, y=600)

        # empty text labels to later config
        self.n = Label(master, text="", font=('arial 200 bold'))
        self.n.place(x=500, y=100)

        self.pname = Label(master, text="", font=('arial 80 bold'))
        self.pname.place(x=300, y=400)
    # function to speak the text and update the text
    def func(self):
        self.n.config(text=str(number[self.x]))
        self.pname.config(text=str(patients[self.x]))
        engine = pyttsx3.init()
        voices = engine.getProperty('voices')
        rate = engine.getProperty('rate')
        engine.setProperty('rate', rate-50)
        engine.say('Booking number ' + str(number[self.x]) + str(patients[self.x]))
        engine.runAndWait()
        self.x += 1
root = Tk()
b = Application(root)
root.geometry("1366x768+0+0")
root.resizable(False, False)
root.mainloop()
```