PYTHON PROJECT POST-SUBMISSION REPORT

Course:-Python Programming (INT 213)

Submitted To:-Navpreet Rupal

"School of Computer Science & Engineering"
Lovely Professional University

PROJECT TITLE-

CAB BOOKING SYSTEM IN LPU

LOVELY PROFESSIONAL UNIVERSITY

TEAM MEMBERS–

NAME	REG.NO	ROLL-NO
Rohan Dwivedi	12113276	RK21WYA15
Chipani Tarun Manikanta	12113465	RK21WYB37
Yanamandala Sri Sai Harshitha	12112419	RK21WYB62

ABOUT PROJECT -

Cab Booking system is an application which is used for Booking cabs using a computerised software . In this application we can perform many operations like storing CabMs account of every student in university, for available cabs, for available routes, car pool options , charges for particular route, maximum time to reach destination, drivers contact details.

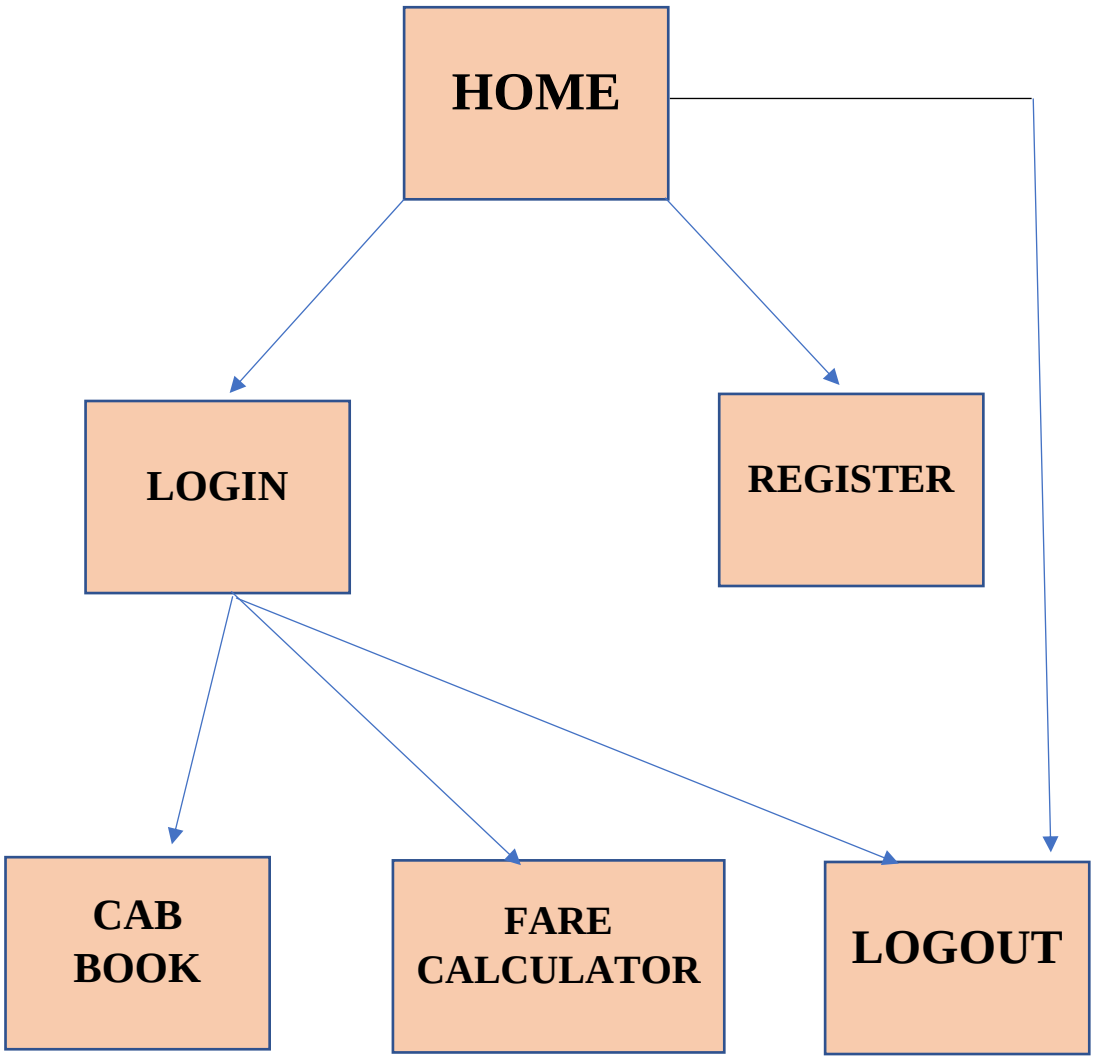
This project aims to provide software to car rental servicing center that maintains the information about the customer details, vehicle details, insurance details, driver details, booking details and transaction details of the customer.

This application helps everyone in lpu to book the transport for their destination from lpu only in the best possible way.

OBJECTIVES-

1) To develop a system that can replace the manual transport booking system.
2) Develop a database which stores user details and staff details.
3) User friendly booking procedure.
4) Location tracking and user friendly interface.
5) Arrival time estimation and fare precision.
6) To make travelling easier and to ride safely and securely.

DESIGN



DATA STRUCTURES USED

Dictionary: Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

```
def Kilo():
    if var2.get() == 0:
        self.txtKm.configure(state=DISABLED)
        Km.set("0")
    elif var2.get() == 1 and var11.get() != "" and var12.get() != "":
        self.txtKm.configure(state=NORMAL)
        if var11.get() == "BleckerStreet":
            switch ={"BrownAvenue": 10,"NorthAvenue": 8,"BoggessStreet":6,"BleckerStreet": 0}
            Km.set(switch[var12.get()])
        elif var11.get() == "BrownAvenue":
            switch ={"BrownAvenue": 0,"NorthAvenue": 2,"BoggessStreet":5,"BleckerStreet": 10}
            Km.set(switch[var12.get()])
        elif var11.get() == "NorthAvenue":
            switch ={"BrownAvenue": 2,"NorthAvenue": 0,"BoggessStreet":3,"BleckerStreet": 8}
            Km.set(switch[var12.get()])
        elif var11.get() == "BoggessStreet":
            switch ={"BrownAvenue": 5,"NorthAvenue": 3,"BoggessStreet":0,"BleckerStreet": 6}
            Km.set(switch[var12.get()])
```

SOURCE CODE:-

```
from tkinter import *
from tkinter import ttk
import random
import time
import datetime
from tkinter import messagebox as ms
import sqlite3

Item4 = 0

# make database and users (if not exists already) table at programme start
up
with sqlite3.connect('Users.db') as db:
    c = db.cursor()

c.execute('CREATE TABLE IF NOT EXISTS user (username TEXT NOT NULL
,password TEXT NOT NULL)')
db.commit()
db.close()


# main Class
class user:
    def __init__(self, master):
        # Window
        self.master = master
        # Some Usefull variables
        self.username = StringVar()
        self.password = StringVar()
        self.n_username = StringVar()
        self.n_password = StringVar()
        # Create Widgets
        self.widgets()

    # Login Function
    def login(self):
        # Establish Connection
        with sqlite3.connect('Users.db') as db:
            c = db.cursor()

        # Find user If there is any take proper action
        find_user = ('SELECT * FROM user WHERE username = ? and password =
?')
        c.execute(find_user, [(self.username.get()),
(self.password.get())])
        result = c.fetchall()
        if result:
```

```
            self.logf.pack_forget()
            self.head['text'] = "Welcome " + self.username.get()
            self.head.configure(fg="black")
            self.head.pack(fill=X)
            application = travel(root)

        else:
            ms.showerror('Oops!', 'Username Not Found.')

    def new_user(self):
        # Establish Connection
        with sqlite3.connect('Users.db') as db:
            c = db.cursor()

        # Find Existing username if any take proper action
        find_user = ('SELECT * FROM user WHERE username = ?')
        c.execute(find_user, [(self.username.get())])
        if c.fetchall():
            ms.showerror('Error!', 'Username Already Taken!')
        else:
            ms.showinfo('Success!', 'Account Created!')
            self.log()
        # Create New Account
        insert = 'INSERT INTO user(username,password) VALUES(?,?)'
        c.execute(insert, [(self.n_username.get()),
(self.n_password.get())])
        db.commit()

        # Frame Packing Methords

    def log(self):
        self.username.set('')
        self.password.set('')
        self.crf.pack_forget()
        self.head['text'] = 'Login'
        self.logf.pack()

    def cr(self):
        self.n_username.set('')
        self.n_password.set('')
        self.logf.pack_forget()
        self.head['text'] = 'Create Account'
        self.crf.pack()

    # Draw Widgets
    def widgets(self):
        self.head = Label(self.master, text='Login Panel', font=('', 30),
pady=10)
        self.head.pack()
        self.logf = Frame(self.master, padx=10, pady=10)
        Label(self.logf, text='Username: ', font=('', 20), pady=5,
padx=5).grid(sticky=W)
        Entry(self.logf, textvariable=self.username, bd=5, font=('',
15)).grid(row=0, column=1)
        Label(self.logf, text='Password: ', font=('', 20), pady=5,
padx=5).grid(sticky=W)
        Entry(self.logf, textvariable=self.password, bd=5, font=('', 15),
show='*').grid(row=1, column=1)
        Button(self.logf, text=' Login ', bd=3, font=('', 15), padx=5,
```

```
pady=5, command=self.login).grid()
        Button(self.logf, text=' Create Account ', bd=3, font=('', 15),
padx=5, pady=5, command=self.cr).grid(row=2,

column=1)
        self.logf.pack()

        self.crf = Frame(self.master, padx=10, pady=10)
        Label(self.crf, text='Username: ', font=('', 20), pady=5,
padx=5).grid(sticky=W)
        Entry(self.crf, textvariable=self.n_username, bd=5, font=('',
15)).grid(row=0, column=1)
        Label(self.crf, text='Password: ', font=('', 20), pady=5,
padx=5).grid(sticky=W)
        Entry(self.crf, textvariable=self.n_password, bd=5, font=('', 15),
show='*').grid(row=1, column=1)
        Button(self.crf, text='Create Account', bd=3, font=('', 15),
padx=5, pady=5, command=self.new_user).grid()
        Button(self.crf, text='Go to Login', bd=3, font=('', 15), padx=5,
pady=5, command=self.log).grid(row=2,

column=1)


class travel:

    def __init__(self, root):
        self.root = root
        self.root.title("Cab Booking System")
        self.root.geometry(geometry)
        self.root.configure(background='black')

        DateofOrder = StringVar()
        DateofOrder.set(time.strftime(" %d / %m / %Y "))
        Receipt_Ref = StringVar()
        PaidTax = StringVar()
        SubTotal = StringVar()
        TotalCost = StringVar()

        var1 = IntVar()
        var2 = IntVar()
        var3 = IntVar()
        var4 = IntVar()
        journeyType = IntVar()
        carType = IntVar()

        varl1 = StringVar()
        varl2 = StringVar()
        varl3 = StringVar()
        reset_counter = 0

        Firstname = StringVar()
        Surname = StringVar()
        Address = StringVar()
        Postcode = StringVar()
        Mobile = StringVar()
        Telephone = StringVar()
        Email = StringVar()
```

```
        CabTax = StringVar()
        Km = StringVar()
        Travel_Ins = StringVar()
        Luggage = StringVar()
        Receipt = StringVar()

        Standard = StringVar()
        FordGalaxy = StringVar()
        FordMondeo = StringVar()

        CabTax.set("0")
        Km.set("0")
        Travel_Ins.set("0")
        Luggage.set("0")

        Standard.set("0")
        FordGalaxy.set("0")
        FordMondeo.set("0")

        # ==========================================Define
Functiom==================================================

        def iExit():
            iExit = ms.askyesno("Prompt!", "Do you want to exit?")
            if iExit > 0:
                root.destroy()
                return

        def Reset():
            CabTax.set("0")
            Km.set("0")
            Travel_Ins.set("0")
            Luggage.set("0")

            Standard.set("0")
            FordGalaxy.set("0")
            FordMondeo.set("0")

            Firstname.set("")
            Surname.set("")
            Address.set("")
            Postcode.set("")
            Mobile.set("")
            Telephone.set("")
            Email.set("")

            PaidTax.set("")
            SubTotal.set("")
            TotalCost.set("")
            self.txtReceipt1.delete("1.0", END)
            self.txtReceipt2.delete("1.0", END)

            var1.set(0)
            var2.set(0)
            var3.set(0)
            var4.set(0)
            journeyType.set(0)
            carType.set(0)
            varl1.set("0")
```

```
            varl2.set("0")
            varl3.set("0")

            self.cboPickup.current(0)
            self.cboDrop.current(0)
            self.cboPooling.current(0)

            self.txtCabTax.configure(state=DISABLED)
            self.txtKm.configure(state=DISABLED)
            self.txtTravel_Ins.configure(state=DISABLED)
            self.txtLuggage.configure(state=DISABLED)

            self.txtStandard.configure(state=DISABLED)
            self.txtFordGalaxy.configure(state=DISABLED)
            self.txtFordMondeo.configure(state=DISABLED)
            self.reset_counter = 1

        def Receiptt():
            if reset_counter == 0 and Firstname.get() != "" and
Surname.get() != "" and Address.get() != "" and Postcode.get() != "" and
Mobile.get() != "" and Telephone.get() != "" and Email.get() != "":
                self.txtReceipt1.delete("1.0", END)
                self.txtReceipt2.delete("1.0", END)
                x = random.randint(10853, 500831)
                randomRef = str(x)
                Receipt_Ref.set(randomRef)

                self.txtReceipt1.insert(END, "Receipt Ref:\n")
                self.txtReceipt2.insert(END, Receipt_Ref.get() + "\n")
                self.txtReceipt1.insert(END, 'Date:\n')
                self.txtReceipt2.insert(END, DateofOrder.get() + "\n")
                self.txtReceipt1.insert(END, 'Cab No:\n')
                self.txtReceipt2.insert(END, 'TR ' + Receipt_Ref.get() + "
BW\n")
                self.txtReceipt1.insert(END, 'Firstname:\n')
                self.txtReceipt2.insert(END, Firstname.get() + "\n")
                self.txtReceipt1.insert(END, 'Surname:\n')
                self.txtReceipt2.insert(END, Surname.get() + "\n")
                self.txtReceipt1.insert(END, 'Address:\n')
                self.txtReceipt2.insert(END, Address.get() + "\n")
                self.txtReceipt1.insert(END, 'Postal Code:\n')
                self.txtReceipt2.insert(END, Postcode.get() + "\n")
                self.txtReceipt1.insert(END, 'Telephone:\n')
                self.txtReceipt2.insert(END, Telephone.get() + "\n")
                self.txtReceipt1.insert(END, 'Mobile:\n')
                self.txtReceipt2.insert(END, Mobile.get() + "\n")
                self.txtReceipt1.insert(END, 'Email:\n')
                self.txtReceipt2.insert(END, Email.get() + "\n")
                self.txtReceipt1.insert(END, 'From:\n')
                self.txtReceipt2.insert(END, varl1.get() + "\n")
                self.txtReceipt1.insert(END, 'To:\n')
                self.txtReceipt2.insert(END, varl2.get() + "\n")
                self.txtReceipt1.insert(END, 'Pooling:\n')
                self.txtReceipt2.insert(END, varl3.get() + "\n")
                self.txtReceipt1.insert(END, 'Standard:\n')
                self.txtReceipt2.insert(END, Standard.get() + "\n")
                self.txtReceipt1.insert(END, 'Prime Sedan:\n')
                self.txtReceipt2.insert(END, FordGalaxy.get() + "\n")
                self.txtReceipt1.insert(END, 'Premium Sedan:\n')
```

```
                self.txtReceipt2.insert(END, FordMondeo.get() + "\n")
                self.txtReceipt1.insert(END, 'Paid:\n')
                self.txtReceipt2.insert(END, PaidTax.get() + "\n")
                self.txtReceipt1.insert(END, 'SubTotal:\n')
                self.txtReceipt2.insert(END, str(SubTotal.get()) + "\n")
                self.txtReceipt1.insert(END, 'Total Cost:\n')
                self.txtReceipt2.insert(END, str(TotalCost.get()))

            else:
                self.txtReceipt1.delete("1.0", END)
                self.txtReceipt2.delete("1.0", END)
                self.txtReceipt1.insert(END, "\nNo Input")

        def Cab_Tax():
            global Item1
            if var1.get() == 1:
                self.txtCabTax.configure(state=NORMAL)
                Item1 = float(50)
                CabTax.set("Rs " + str(Item1))
            elif var1.get() == 0:
                self.txtCabTax.configure(state=DISABLED)
                CabTax.set("0")
                Item1 = 0

        def Kilo():
            if var2.get() == 0:
                self.txtKm.configure(state=DISABLED)
                Km.set("0")
            elif var2.get() == 1 and varl1.get() != "" and varl2.get() !=
"":
                self.txtKm.configure(state=NORMAL)
                if varl1.get() == "BleckerStreet":
                    switch = {"BrownAvenue": 10, "NorthAvenue": 8,
"BoggessStreet": 6, "BleckerStreet": 0}
                    Km.set(switch[varl2.get()])
                elif varl1.get() == "BrownAvenue":
                    switch = {"BrownAvenue": 0, "NorthAvenue": 2,
"BoggessStreet": 5, "BleckerStreet": 10}
                    Km.set(switch[varl2.get()])
                elif varl1.get() == "NorthAvenue":
                    switch = {"BrownAvenue": 2, "NorthAvenue": 0,
"BoggessStreet": 3, "BleckerStreet": 8}
                    Km.set(switch[varl2.get()])
                elif varl1.get() == "BoggessStreet":
                    switch = {"BrownAvenue": 5, "NorthAvenue": 3,
"BoggessStreet": 0, "BleckerStreet": 6}
                    Km.set(switch[varl2.get()])

        def Travelling():
            global Item3
            if var3.get() == 1:
                self.txtTravel_Ins.configure(state=NORMAL)
                Item3 = float(10)
                Travel_Ins.set("Rs " + str(Item3))
            elif var3.get() == 0:
                self.txtTravel_Ins.configure(state=DISABLED)
                Travel_Ins.set("0")
                Item3 = 0
```

```
        def Lug():
            global Item4
            if (var4.get() == 1):
                self.txtLuggage.configure(state=NORMAL)
                Item4 = float(30)
                Luggage.set("Rs " + str(Item4))
            elif var4.get() == 0:
                self.txtLuggage.configure(state=DISABLED)
                Luggage.set("0")
                Item4 = 0

        def selectCar():
            global Item5
            if carType.get() == 1:
                self.txtFordGalaxy.configure(state=DISABLED)
                FordGalaxy.set("0")
                self.txtFordMondeo.configure(state=DISABLED)
                FordMondeo.set("0")
                self.txtStandard.configure(state=NORMAL)
                Item5 = float(8)
                Standard.set("Rs " + str(Item5))
            elif carType.get() == 2:
                self.txtStandard.configure(state=DISABLED)
                Standard.set("0")
                self.txtFordMondeo.configure(state=DISABLED)
                FordMondeo.set("0")
                self.txtFordGalaxy.configure(state=NORMAL)
                Item5 = float(15)
                FordGalaxy.set("Rs " + str(Item5))
            else:
                self.txtStandard.configure(state=DISABLED)
                Standard.set("0")
                self.txtFordGalaxy.configure(state=DISABLED)
                FordGalaxy.set("0")
                self.txtFordMondeo.configure(state=NORMAL)
                Item5 = float(22)
                FordMondeo.set("Rs " + str(Item5))

        def Total_Paid():
            if ((
                    var1.get() == 1 and var2.get() == 1 and var3.get() ==
1 or var4.get() == 1) and carType.get() != 0 and journeyType.get() != 0
and (
                    varl1.get() != "" and varl2.get() != "")):
                if journeyType.get() == 1:
                    Item2 = Km.get()
                    Cost_of_fare = (Item1 + (float(Item2) * Item5) + Item3
+ Item4)

                    Tax = "Rs " + str('%.2f' % ((Cost_of_fare) * 0.09))
                    ST = "Rs " + str('%.2f' % ((Cost_of_fare)))
                    TT = "Rs " + str('%.2f' % (Cost_of_fare +
((Cost_of_fare) * 0.9)))
                elif journeyType.get() == 2:
                    Item2 = Km.get()
                    Cost_of_fare = (Item1 + (float(Item2) * Item5) * 1.5 +
Item3 + Item4)

                    Tax = "Rs " + str('%.2f' % ((Cost_of_fare) * 0.09))
```

```
                    ST = "Rs " + str('%.2f' % ((Cost_of_fare)))
                    TT = "Rs " + str('%.2f' % (Cost_of_fare +
((Cost_of_fare) * 0.9)))
                else:
                    Item2 = Km.get()
                    Cost_of_fare = (Item1 + (float(Item2) * Item5) * 2 +
Item3 + Item4)

                    Tax = "Rs " + str('%.2f' % ((Cost_of_fare) * 0.09))
                    ST = "Rs " + str('%.2f' % ((Cost_of_fare)))
                    TT = "Rs " + str('%.2f' % (Cost_of_fare +
((Cost_of_fare) * 0.9)))

                PaidTax.set(Tax)
                SubTotal.set(ST)
                TotalCost.set(TT)
            else:
                w = ms.showwarning("Error !", "Invalid Input\nPlease try
again !!!")

        #
==================================================mainframe=================
=========================================================

        MainFrame = Frame(self.root)
        MainFrame.pack(fill=BOTH, expand=True)

        Tops = Frame(MainFrame, bd=10, width=1350, relief=RIDGE)
        Tops.pack(side=TOP, fill=BOTH)

        self.lblTitle = Label(Tops, font=('arial', 50, 'bold'), text="\t
Cab Booking System ")
        self.lblTitle.grid()

        #
==================================================customerframedetail=======
=========================================================
        CustomerDetailsFrame = LabelFrame(MainFrame, width=1350,
height=500, bd=20, pady=5, relief=RIDGE)
        CustomerDetailsFrame.pack(side=BOTTOM, fill=BOTH, expand=True)

        FrameDetails = Frame(CustomerDetailsFrame, width=880, height=400,
bd=10, relief=RIDGE)
        FrameDetails.pack(side=LEFT, fill=BOTH, expand=True)

        CustomerName = LabelFrame(FrameDetails, width=150, height=250,
bd=10, font=('arial', 12, 'bold'),
                                  text="Customer Info", relief=RIDGE)
        CustomerName.grid(row=0, column=0)

        TravelFrame = LabelFrame(FrameDetails, bd=10, width=300,
height=250, font=('arial', 12, 'bold'),
                                 text="Booking Detail", relief=RIDGE)
        TravelFrame.grid(row=0, column=1)

        Book_Frame = LabelFrame(FrameDetails, width=300, height=150,
relief=FLAT)
        Book_Frame.grid(row=1, column=0)
```

```
        CostFrame = LabelFrame(FrameDetails, width=150, height=150, bd=5,
relief=FLAT)
        CostFrame.grid(row=1, column=1)

        #
==================================================recipt=====================
=================================================
        Receipt_BottonFrame = LabelFrame(CustomerDetailsFrame, bd=10,
width=450, height=400, relief=RIDGE)
        Receipt_BottonFrame.pack(side=RIGHT, fill=BOTH, expand=True)

        ReceiptFrame = LabelFrame(Receipt_BottonFrame, width=350,
height=300, font=('arial', 12, 'bold'),
                                  text="Receipt", relief=RIDGE)
        ReceiptFrame.grid(row=0, column=0)

        ButtonFrame = LabelFrame(Receipt_BottonFrame, width=350,
height=100, relief=RIDGE)
        ButtonFrame.grid(row=1, column=0)
        #
==========================================================CustomerName=====
=================================================

        self.lblFirstname = Label(CustomerName, font=('arial', 14,
'bold'), text="Firstname", bd=7)
        self.lblFirstname.grid(row=0, column=0, sticky=W)
        self.txtFirstname = Entry(CustomerName, font=('arial', 14,
'bold'), textvariable=Firstname, bd=7, insertwidth=2,
                                  justify=RIGHT)
        self.txtFirstname.grid(row=0, column=1)

        self.lblSurname = Label(CustomerName, font=('arial', 14, 'bold'),
text="Surname", bd=7)
        self.lblSurname.grid(row=1, column=0, sticky=W)
        self.txtSurname = Entry(CustomerName, font=('arial', 14, 'bold'),
textvariable=Surname, bd=7, insertwidth=2,
                                justify=RIGHT)
        self.txtSurname.grid(row=1, column=1, sticky=W)

        self.lblAddress = Label(CustomerName, font=('arial', 14, 'bold'),
text="Address", bd=7)
        self.lblAddress.grid(row=2, column=0, sticky=W)
        self.txtAddress = Entry(CustomerName, font=('arial', 14, 'bold'),
textvariable=Address, bd=7, insertwidth=2,
                                justify=RIGHT)
        self.txtAddress.grid(row=2, column=1)

        self.lblPostcode = Label(CustomerName, font=('arial', 14, 'bold'),
text="Postcode", bd=7)
        self.lblPostcode.grid(row=3, column=0, sticky=W)
        self.txtPostcode = Entry(CustomerName, font=('arial', 14, 'bold'),
textvariable=Postcode, bd=7, insertwidth=2,
                                 justify=RIGHT)
        self.txtPostcode.grid(row=3, column=1)

        self.lblTelephone = Label(CustomerName, font=('arial', 14,
'bold'), text="Telephone", bd=7)
        self.lblTelephone.grid(row=4, column=0, sticky=W)
        self.txtTelephone = Entry(CustomerName, font=('arial', 14,
```

```
'bold'), textvariable=Telephone, bd=7, insertwidth=2,
                                justify=RIGHT)
        self.txtTelephone.grid(row=4, column=1)

        self.lblMobile = Label(CustomerName, font=('arial', 14, 'bold'),
text="Mobile", bd=7)
        self.lblMobile.grid(row=5, column=0, sticky=W)
        self.txtMobile = Entry(CustomerName, font=('arial', 14, 'bold'),
textvariable=Mobile, bd=7, insertwidth=2,
                              justify=RIGHT)
        self.txtMobile.grid(row=5, column=1)

        self.lblEmail = Label(CustomerName, font=('arial', 14, 'bold'),
text="Email", bd=7)
        self.lblEmail.grid(row=6, column=0, sticky=W)
        self.txtEmail = Entry(CustomerName, font=('arial', 14, 'bold'),
textvariable=Email, bd=7, insertwidth=2,
                             justify=RIGHT)
        self.txtEmail.grid(row=6, column=1)

        # ================================================Cab
Information==================================================================
        self.lblPickup = Label(TravelFrame, font=('arial', 14, 'bold'),
text="Pickup", bd=7)
        self.lblPickup.grid(row=0, column=0, sticky=W)

        self.cboPickup = ttk.Combobox(TravelFrame, textvariable=varl1,
state='readonly', font=('arial', 20, 'bold'),
                                 width=14)
        self.cboPickup['value'] = ('', 'BleckerStreet', 'BoggessStreet',
'NorthAvenue', 'BrownAvenue')
        self.cboPickup.current(0)
        self.cboPickup.grid(row=0, column=1)

        self.lblDrop = Label(TravelFrame, font=('arial', 14, 'bold'),
text="Drop", bd=7)
        self.lblDrop.grid(row=1, column=0, sticky=W)

        self.cboDrop = ttk.Combobox(TravelFrame, textvariable=varl2,
state='readonly', font=('arial', 20, 'bold'),
                                width=14)
        self.cboDrop['value'] = ('', 'BrownAvenue', 'NorthAvenue',
'BleckerStreet', 'BoggessStreet')
        self.cboDrop.current(0)
        self.cboDrop.grid(row=1, column=1)

        self.lblPooling = Label(TravelFrame, font=('arial', 14, 'bold'),
text="Pooling", bd=7)
        self.lblPooling.grid(row=2, column=0, sticky=W)

        self.cboPooling = ttk.Combobox(TravelFrame, textvariable=varl3,
state='readonly', font=('arial', 20, 'bold'),
                                 width=14)
        self.cboPooling['value'] = ('', '1', '2', '3', '4')
        self.cboPooling.current(1)
        self.cboPooling.grid(row=2, column=1)

        # ================================================Cab
Information==================================================================
```

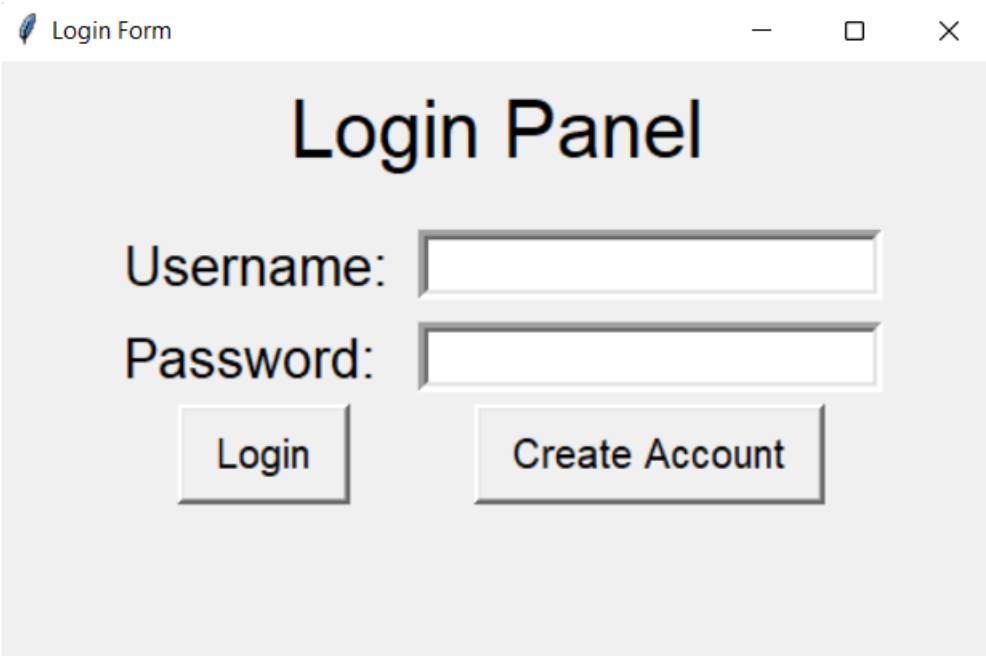
```
        self.chkCabTax = Checkbutton(TravelFrame, text="Base Charge *",
variable=var1, onvalue=1, offvalue=0,
                                font=('arial', 16, 'bold'),
command=Cab_Tax).grid(row=3, column=0, sticky=W)
        self.txtCabTax = Label(TravelFrame, font=('arial', 14, 'bold'),
textvariable=CabTax, bd=6, width=18, bg="white",
                               state=DISABLED, justify=RIGHT,
relief=SUNKEN)
        self.txtCabTax.grid(row=3, column=1)

        self.chkKm = Checkbutton(TravelFrame, text="Distance(KMs) *",
variable=var2, onvalue=1, offvalue=0,
                                 font=('arial', 16, 'bold'),
command=Kilo).grid(row=4, column=0, sticky=W)
        self.txtKm = Label(TravelFrame, font=('arial', 14, 'bold'),
textvariable=Km, bd=6, width=18, bg="white",
                           state=DISABLED, justify=RIGHT, relief=SUNKEN,
highlightthickness=0)
        self.txtKm.grid(row=4, column=1)

        self.chkTravel_Ins = Checkbutton(TravelFrame, text="Travelling
Insurance *", variable=var3, onvalue=1,
                                         offvalue=0, font=('arial', 16,
'bold'), command=Travelling).grid(row=5,

column=0,

sticky=W)
        self.txtTravel_Ins = Label(TravelFrame, font=('arial', 14,
'bold'), textvariable=Travel_Ins, bd=6, width=18,
                                   bg="white", state=DISABLED,
justify=RIGHT, relief=SUNKEN)
        self.txtTravel_Ins.grid(row=5, column=1)

        self.chkLuggage = Checkbutton(TravelFrame, text="Extra Luggage",
variable=var4, onvalue=1, offvalue=0,
                                      font=('arial', 16, 'bold'),
command=Lug).grid(row=6, column=0, sticky=W)
        self.txtLuggage = Label(TravelFrame, font=('arial', 14, 'bold'),
textvariable=Luggage, bd=6, width=18,
                                bg="white", state=DISABLED, justify=RIGHT,
relief=SUNKEN)
        self.txtLuggage.grid(row=6, column=1)

        # ================================payment information
=============================================================================
=

        self.lblPaidTax = Label(CostFrame, font=('arial', 14, 'bold'),
text="Paid Tax\t\t", bd=7)
        self.lblPaidTax.grid(row=0, column=2, sticky=W)
        self.txtPaidTax = Label(CostFrame, font=('arial', 14, 'bold'),
textvariable=PaidTax, bd=7, width=10,
                                justify=RIGHT, bg="white", relief=SUNKEN)
        self.txtPaidTax.grid(row=0, column=3)

        self.lblSubTotal = Label(CostFrame, font=('arial', 14, 'bold'),
text="Sub Total", bd=7)
```

```
        self.lblSubTotal.grid(row=1, column=2, sticky=W)
        self.txtSubTotal = Label(CostFrame, font=('arial', 14, 'bold'),
textvariable=SubTotal, bd=7, width=10,
                                 justify=RIGHT, bg="white", relief=SUNKEN)
        self.txtSubTotal.grid(row=1, column=3)

        self.lblTotalCost = Label(CostFrame, font=('arial', 14, 'bold'),
text="Total Cost", bd=7)
        self.lblTotalCost.grid(row=2, column=2, sticky=W)
        self.txtTotalCost = Label(CostFrame, font=('arial', 14, 'bold'),
textvariable=TotalCost, bd=7, width=10,
                                  justify=RIGHT, bg="white",
relief=SUNKEN)
        self.txtTotalCost.grid(row=2, column=3)

        #
==========================================================Cabselect=======
==================================================================

        self.chkStandard = Radiobutton(Book_Frame, text="Standard Cab",
value=1, variable=carType,
                                       font=('arial', 14, 'bold'),
command=selectCar).grid(row=0, column=0, sticky=W)
        self.txtStandard = Label(Book_Frame, font=('arial', 14, 'bold'),
width=7, textvariable=Standard, bd=5,
                                 state=DISABLED, justify=RIGHT,
bg="white", relief=SUNKEN)
        self.txtStandard.grid(row=0, column=1)

        self.chkFordGalaxyd = Radiobutton(Book_Frame, text="Ford Galaxy
Cab", value=2, variable=carType,
                                          font=('arial', 14, 'bold'),
command=selectCar).grid(row=1, column=0, sticky=W)
        self.txtFordGalaxy = Label(Book_Frame, font=('arial', 14, 'bold'),
width=7, textvariable=FordGalaxy, bd=5,
                                   state=DISABLED, justify=RIGHT,
bg="white", relief=SUNKEN)
        self.txtFordGalaxy.grid(row=1, column=1)

        self.chkFordMondeo = Radiobutton(Book_Frame, text="Ford Mondeo
Cab", value=3, variable=carType,
                                         font=('arial', 14, 'bold'),
command=selectCar).grid(row=2, column=0)
        self.txtFordMondeo = Label(Book_Frame, font=('arial', 14, 'bold'),
width=7, textvariable=FordMondeo, bd=5,
                                   state=DISABLED, justify=RIGHT,
bg="white", relief=SUNKEN)
        self.txtFordMondeo.grid(row=2, column=1)

        self.chkSingle = Radiobutton(Book_Frame, text="Single", value=1,
variable=journeyType,
                                     font=('arial', 14,
'bold')).grid(row=0, column=2, sticky=W)
        self.chkReturn = Radiobutton(Book_Frame, text="Return", value=2,
variable=journeyType,
                                     font=('arial', 14,
'bold')).grid(row=1, column=2, sticky=W)
        self.chkSpecialsNeeds = Radiobutton(Book_Frame,
text="SpecialNeeds", value=3, variable=journeyType,
```

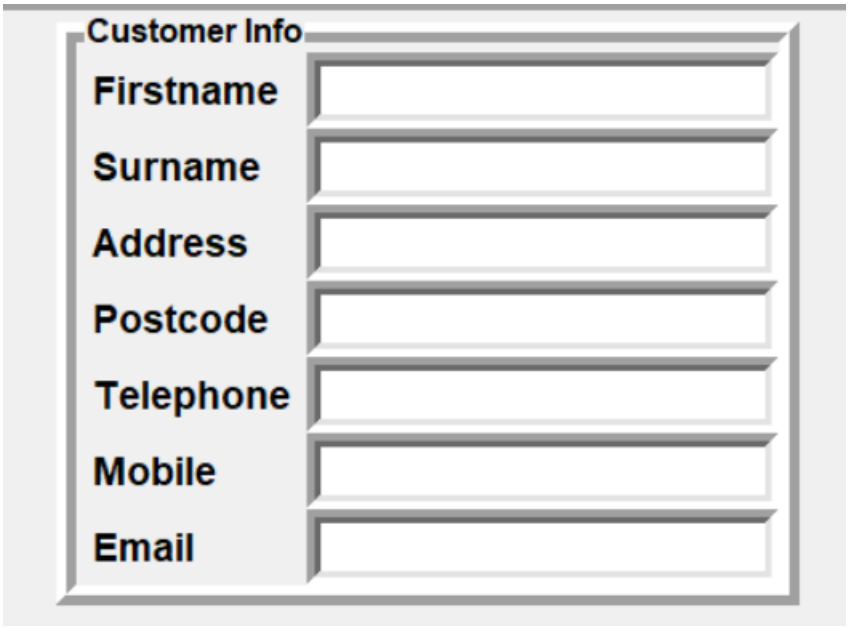
```
                                                font=('arial', 14,
'bold')).grid(row=2, column=2, sticky=W)

        #
=========================================Receipt============================
=========================================================

        self.txtReceipt1 = Text(ReceiptFrame, width=22, height=21,
font=('arial', 10, 'bold'), borderwidth=0)
        self.txtReceipt1.grid(row=0, column=0, columnspan=2)
        self.txtReceipt2 = Text(ReceiptFrame, width=22, height=21,
font=('arial', 10, 'bold'), borderwidth=0)
        self.txtReceipt2.grid(row=0, column=2, columnspan=2)

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

        self.btnTotal = Button(ButtonFrame, padx=18, bd=7, font=('arial',
11, 'bold'), width=2, text='Total',
                                command=Total_Paid).grid(row=0, column=0)
        self.btnReceipt = Button(ButtonFrame, padx=18, bd=7,
font=('arial', 11, 'bold'), width=2, text='Receipt',
                                  command=Receiptt).grid(row=0, column=1)
        self.btnReset = Button(ButtonFrame, padx=18, bd=7, font=('arial',
11, 'bold'), width=2, text='Reset',
                                command=Reset).grid(row=0, column=2)
        self.btnExit = Button(ButtonFrame, padx=18, bd=7, font=('arial',
11, 'bold'), width=2, text='Exit',
                               command=iExit).grid(row=0, column=3)

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


if __name__ == '__main__':
    root = Tk()

    # ============================================ Getting Screen Width
=================================================================
    w = root.winfo_screenwidth()
    h = root.winfo_screenheight()
    geometry = "%dx%d+%d+%d" % (w, h, 0, 0)

    root.geometry("500x300+320+200")
    root.title('Login Form')
    application = user(root)
    root.mainloop()
```

RESULT SCREENSHOT

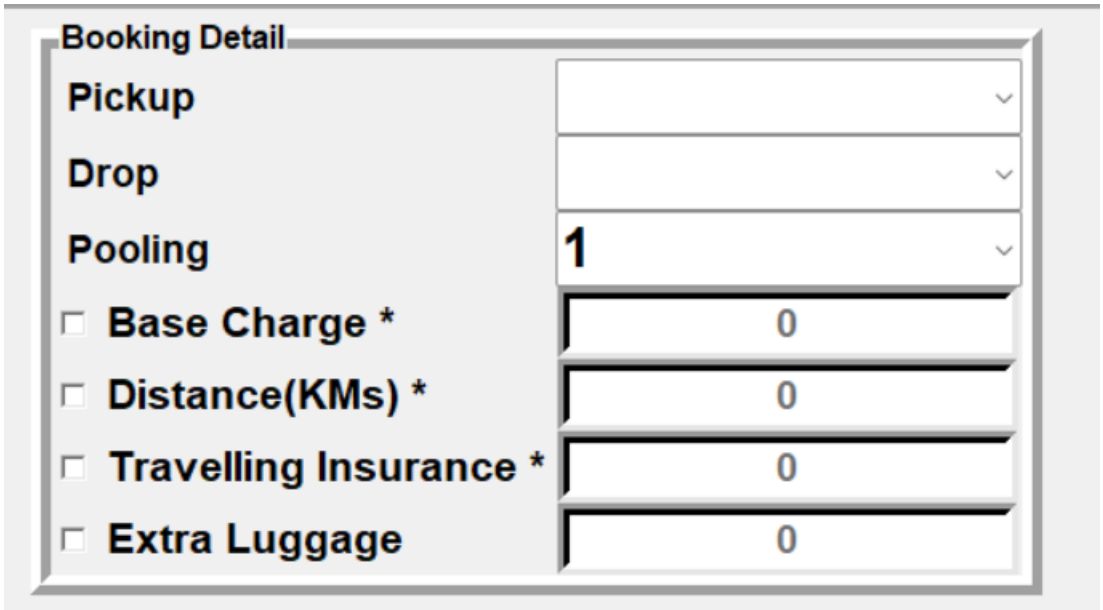
LOGIN PAGE



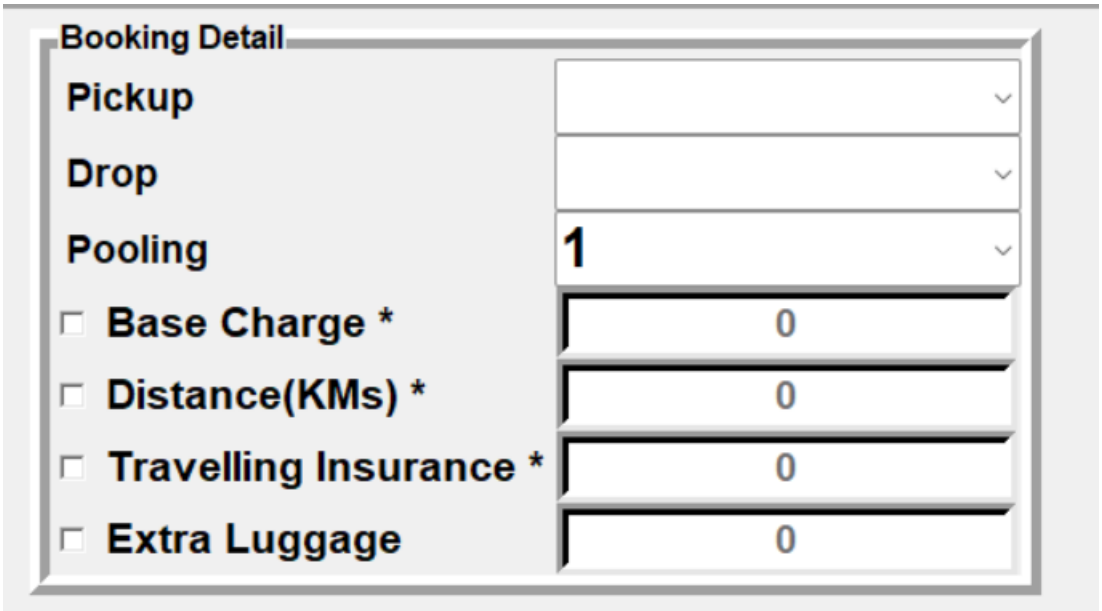
Customer Info

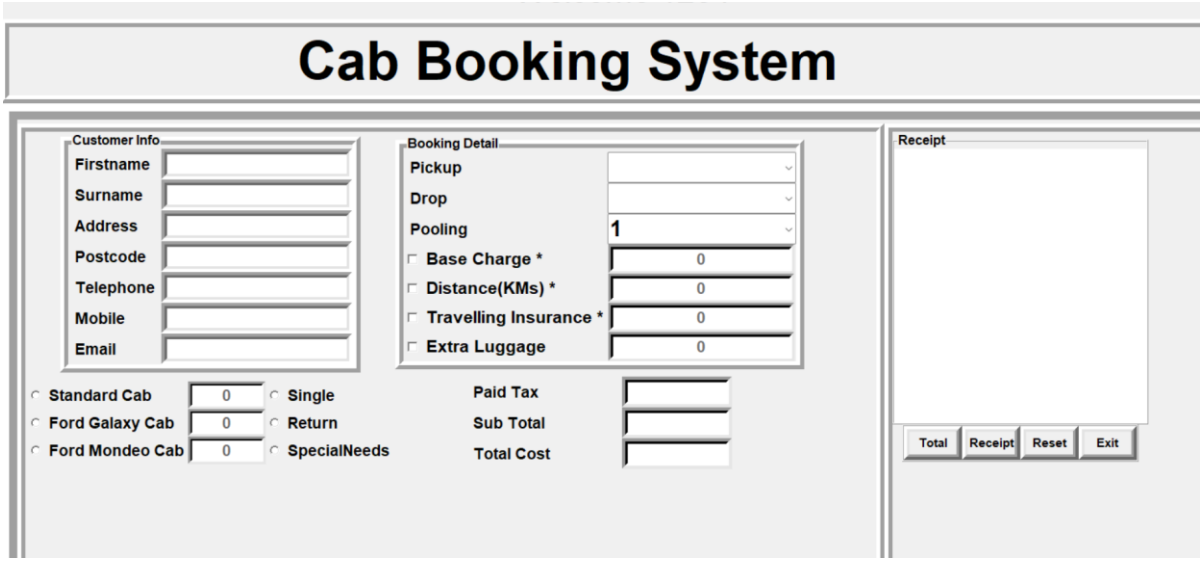


Booking Detail



Receipt


Booking Detail
Pickup
Drop
Pooling
1
Base Charge *
0
Distance(KMs) *
0
Travelling Insurance *
0
Extra Luggage
0


Cab Booking System
Customer Info
Firstname
Surname
Address
Postcode
Telephone
Mobile
Email
Booking Detail
Pickup
Drop
Pooling
1
Base Charge *
0
Distance(KMs) *
0
Travelling Insurance *
0
Extra Luggage
0
Standard Cab
0
Ford Galaxy Cab
0
Ford Mondeo Cab
0
Single
Return
SpecialNeeds
Paid Tax
Sub Total
Total Cost
Receipt
Total
Receipt
Reset
Exit

CONCLUSION

Customers can use an online booking system to rent cabs. Customers may use this online system to browse available taxis, view profiles, and book cabs. Taxi booking is a typical kind of transportation that is offered by a number of different transportation firms in a particular city. The bulk of people rely on taxi services for their daily transportation needs. The company must be registered and fulfil all of the transportation department's requirements and security requirements. This paper demonstrates an effective taxi booking system. This project included a wide variety of topics, from corporate principles to computer science, and required the completion of a number of courses in order to reach the deadline.

REFERENCES

- **GOOGLE**
- **Wikipedia**
- **Stackflow**
- **Taxi company:**
  1-UBER
  2-OLA