FINAL SUBMISSION REPORT
FOR PYTHON GROUP PROJECT.

"School of Computer Science & Engineering"

COURSE CODE :- INT 213

PROJECT:-Design a System for booking Cabs within LPU using python.

SUBMITTED BY :-

NAME	REGISTRATION NO.	ROLL NO.
BIKIRAN BORAH	12111199	RK21WYA26
VISHAL KUMAR SINGH	12104228	RK21WYA01

TABLE OF CONTENTS

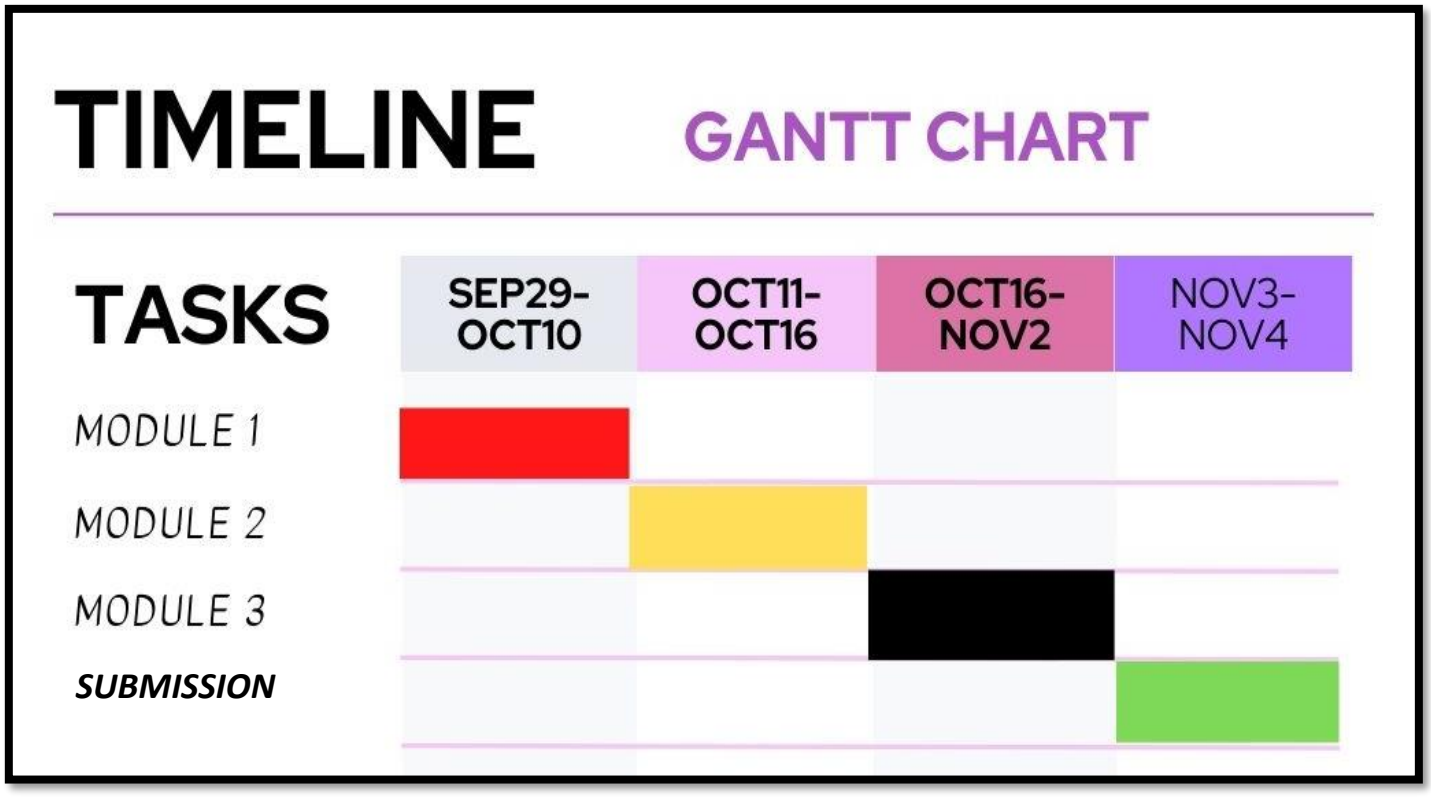
ABOUT PROJECT -

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

PRE SUBMISSION 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.

To make travelling easier and to ride safely and securely.



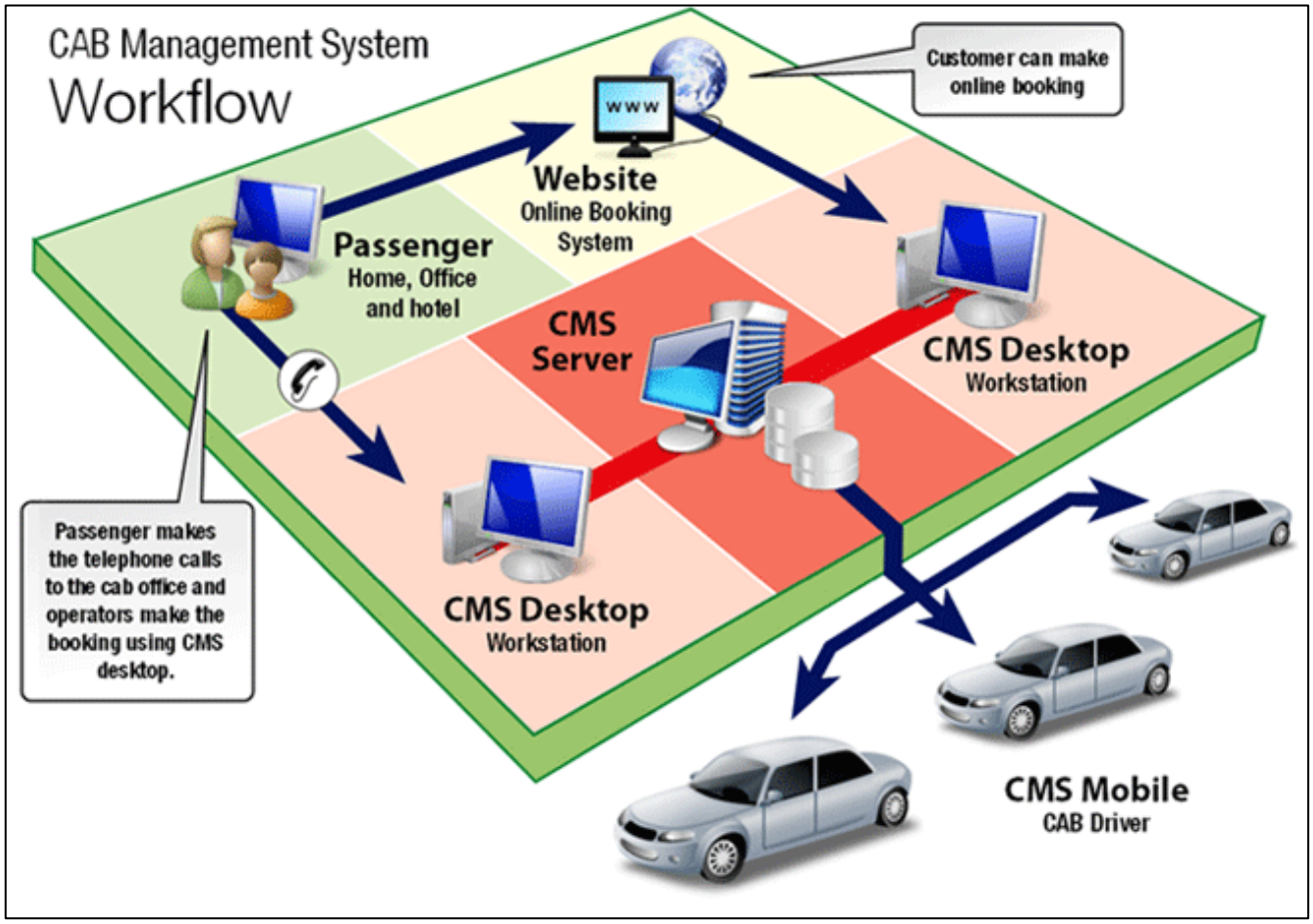
Introduction :

Online Cab Booking System specializing in Hiring cabs to customers. It is an online system through which customers can view available cabs; register the cabs, view profile and book cabs.Cab booking service is a major transport service provided by the various transport operators in a particular city. Mostly peoples use cab service for their daily transportations need. The company must be a registered and fulfils all the requirements and security standards set by the transport department. Online Cab Booking System is a web based platform that allows your customers to book their taxi's and executive taxis all online from the comfort of their own home or office. The platform should offer an administration interface where the taxi company can manage the content, and access all bookings and customer information. More and more Taxi companies are looking for integrated taxi booking systems as it makes life much easier for (1) The traveler - this is highly important and in today's internet age people should be able to book taxis online without having to pick up the phone and (2) the taxi company as all their bookings are now managed via an automated system which means they have an electronic record of future and historic bookings A Cab Booking/Hiring is a system that can be used temporarily for a period of time with a fee. Hiring a car assists people to get around even when they do not have access to their own personal vehicle or don't own a vehicle at all. The individual who want to hire/rent a car must first contact the cab hiring company for the desire vehicle. This can be done online. At this point, this person has to supply some information such as: dates of rental, and type of car. After these details are worked out, the individual renting the car must present a valid Identification Card. Most companies throughout the industry make a profit based of the type of cars.

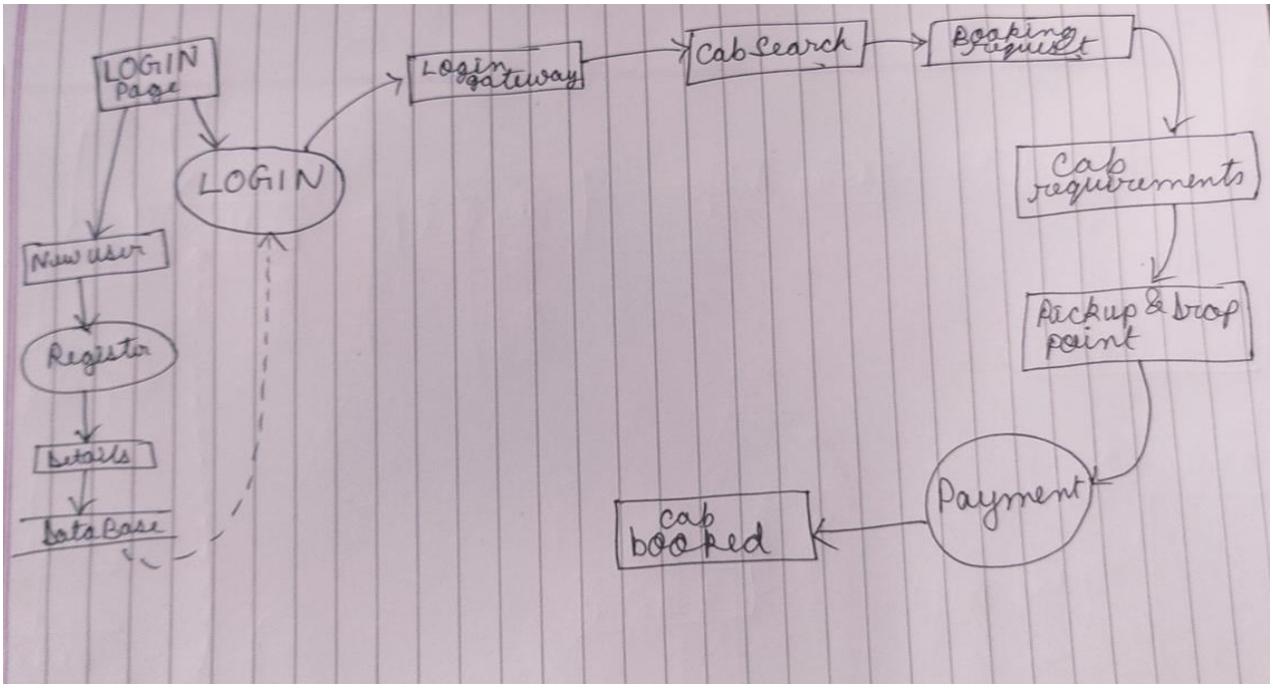
MAIN OBJECTIVE :- Online Cab Booking System specializing in Hiring cabs to customers. It is an online system through which customers can view available cabs; register the cabs, view profile and book cabs. Cab booking service is a major transport service provided by the various transport operators in a particular city. Mostly peoples use cab service for their daily transportations need. The company must be a registered and fulfils all the requirements and security standards set by the transport department. Online Cab Booking System is a web based platform that allows your customers to book their taxi's and executive taxis all online from the comfort of their own home or office. The platform should offer an administration interface where the taxi company can manage the content, and access all bookings and customer information. More and more Taxi companies are looking for integrated taxi booking systems as it makes life much easier for (1) The traveller - this is highly important and in today's internet age people should be able to book taxis online without having to pick up the phone and (2) the taxi company as all their bookings are now managed via an automated system which means they have an electronic record of future and historic bookings A Cab Booking/Hiring is a system that can be used temporarily for a period of time with a fee. Hiring a car assists people to get around even when they do not have access to their own personal vehicle or don't own a vehicle at all. The individual who want to hire/rent a car must first contact the cab hiring company for the desire vehicle. This can be done online. At this point, this person has to supply some information such as: dates of rental, and type of car. After these details are worked out, the individual renting the car must present a valid

OBJECTIVES {Online cab Booking System.Most companies throughout the industry make a profit based of the type of cars.

WORKFLOW/DESIGN FOR THE PROJECT:-



DFD:-



#DESCRIPTION OF MODULES –

1) **CABMS ACCOUNT** – It is the cab booking system account of the students and faculties.

2) **REGISTER NEW ACCOUNT**-This module helps in creating account for new users.

3) **BEST ROUTE**- This module helps in finding the best route for the driver.

4) **LOGIN PAGE -** This module helps in login the account.

5) **FARE PRECISION -**This module helps to view and calculate the fare of user.

6) **AVAILABILTY OF CABS-** This module helps to check the available cabs .

7) **BOOKING REQUEST-** This module helps in booking the cab.

8) **CONTACT DETAILS-** This module helps to give contact details of the driver.

9) **FEEDBACK / COMPLAIN-** This module helps to give the feedback to the user. Issues and complaints can also be raised in this module .

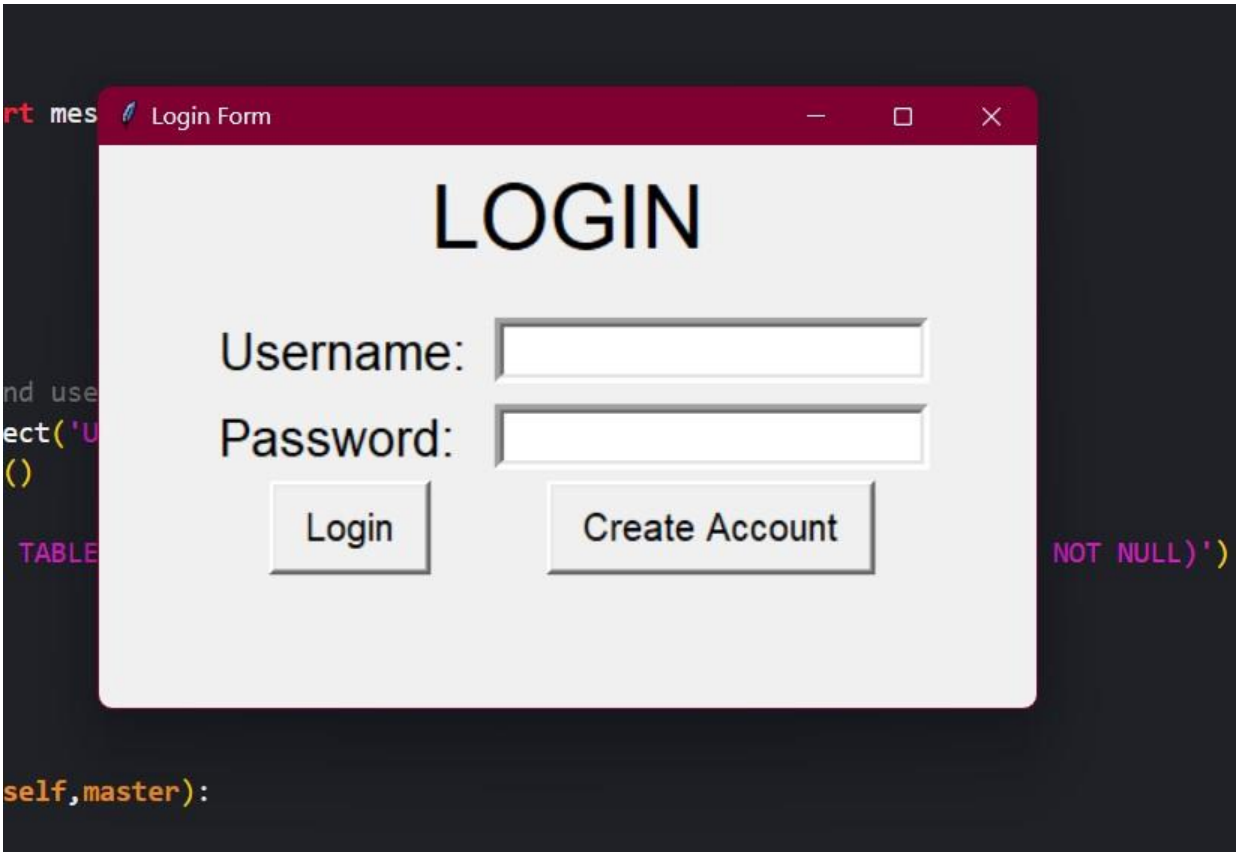
#ALGORITHM:- Cab Finder Algorithm is used to locate the nearest and safest cab for passengers using specific designed algorithm. This algorithm is designed to give most safe riding suggestions of all the cabs available in the vicinity.

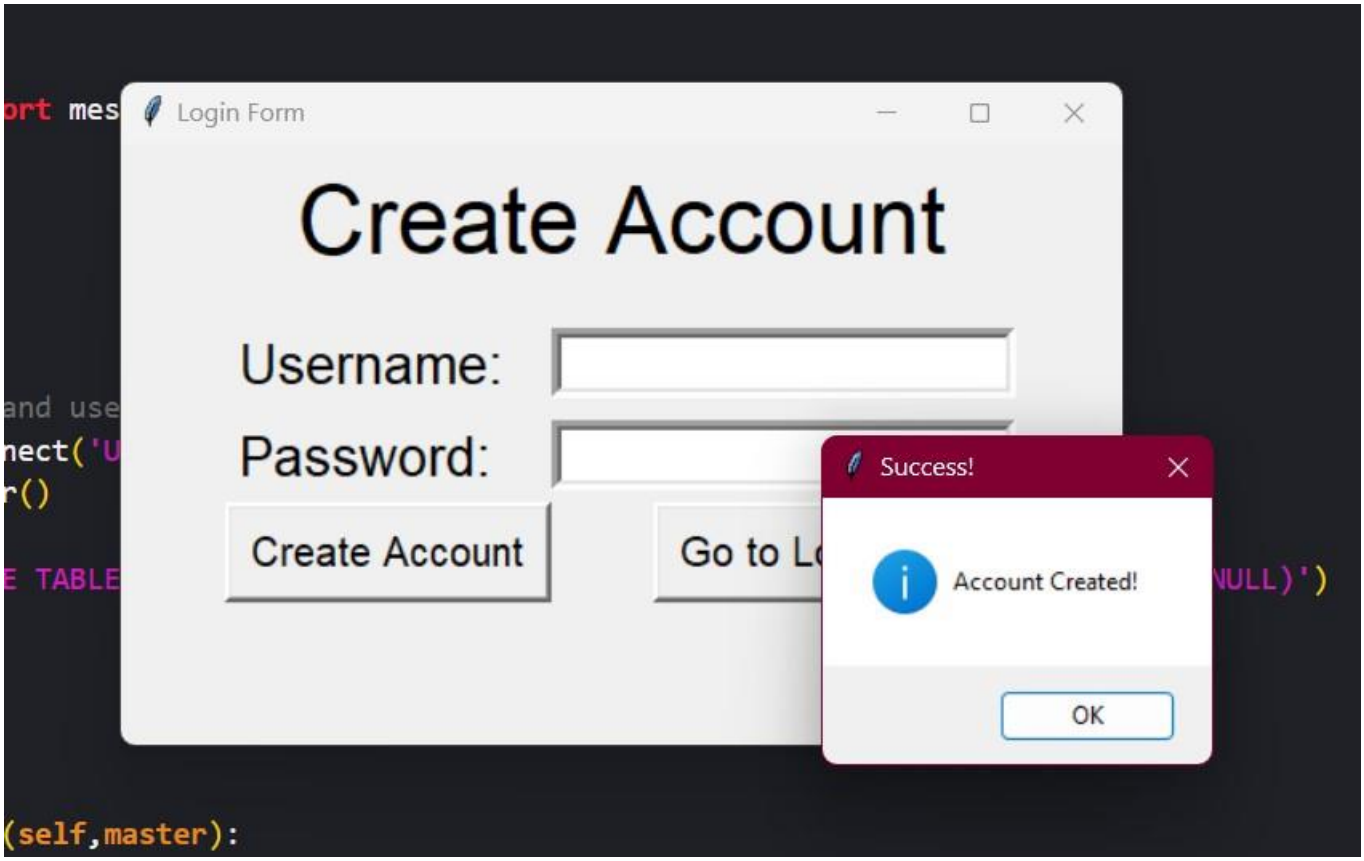
WORKING:-

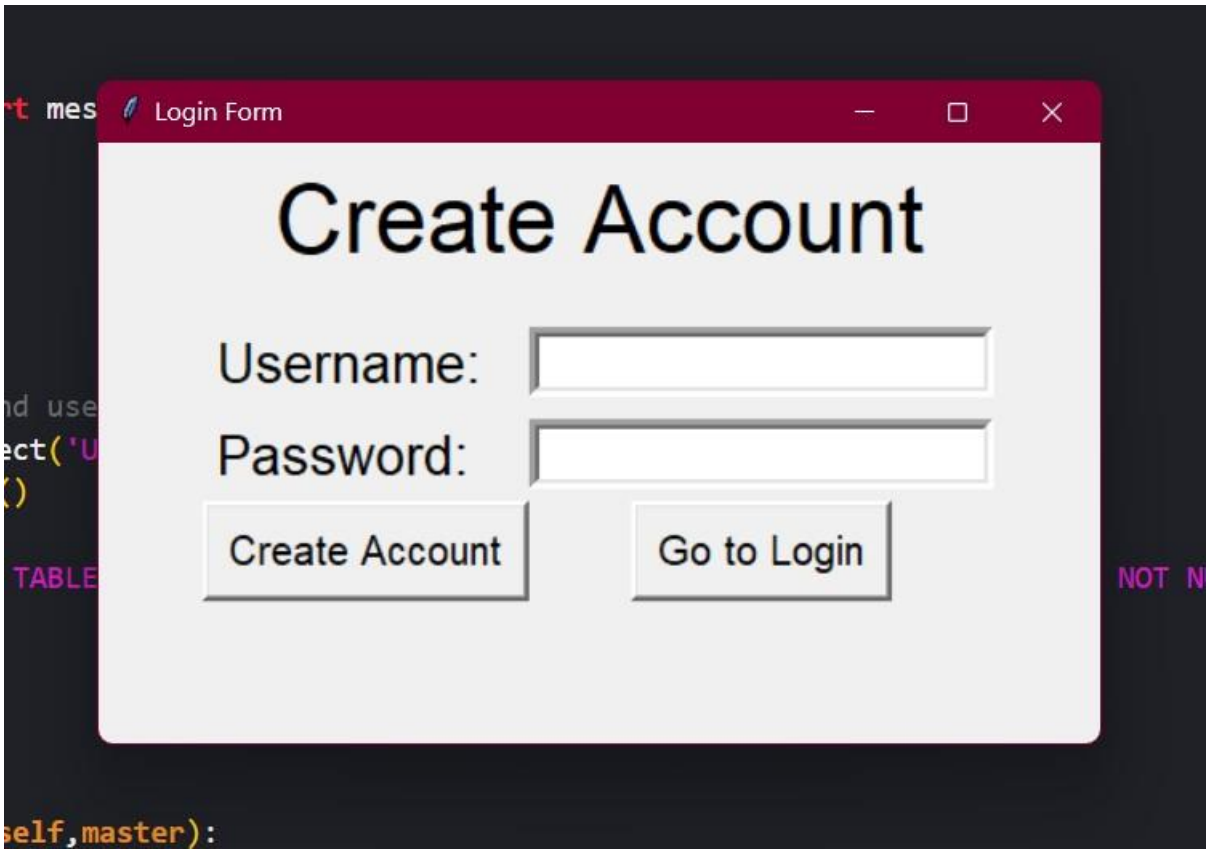
1. The co-ordinates of the taxis are stored regularly on to the server . Thus , in JSON format we have the details of the taxi (details like gps co-odinates of the taxi , rating of taxi driver and trip details)
2. That data is processed to make to find the distances of taxis from the current position of the customer using the formula of co-ordinate distance algorithm.
3. After getting the distances of taxis , other details are taken into account like Rating of the driver ,Number of rides the person has got in that day ,gender of the customer booking the taxi .
4. Then the algorithm is used to calculate total cost for different taxis available.

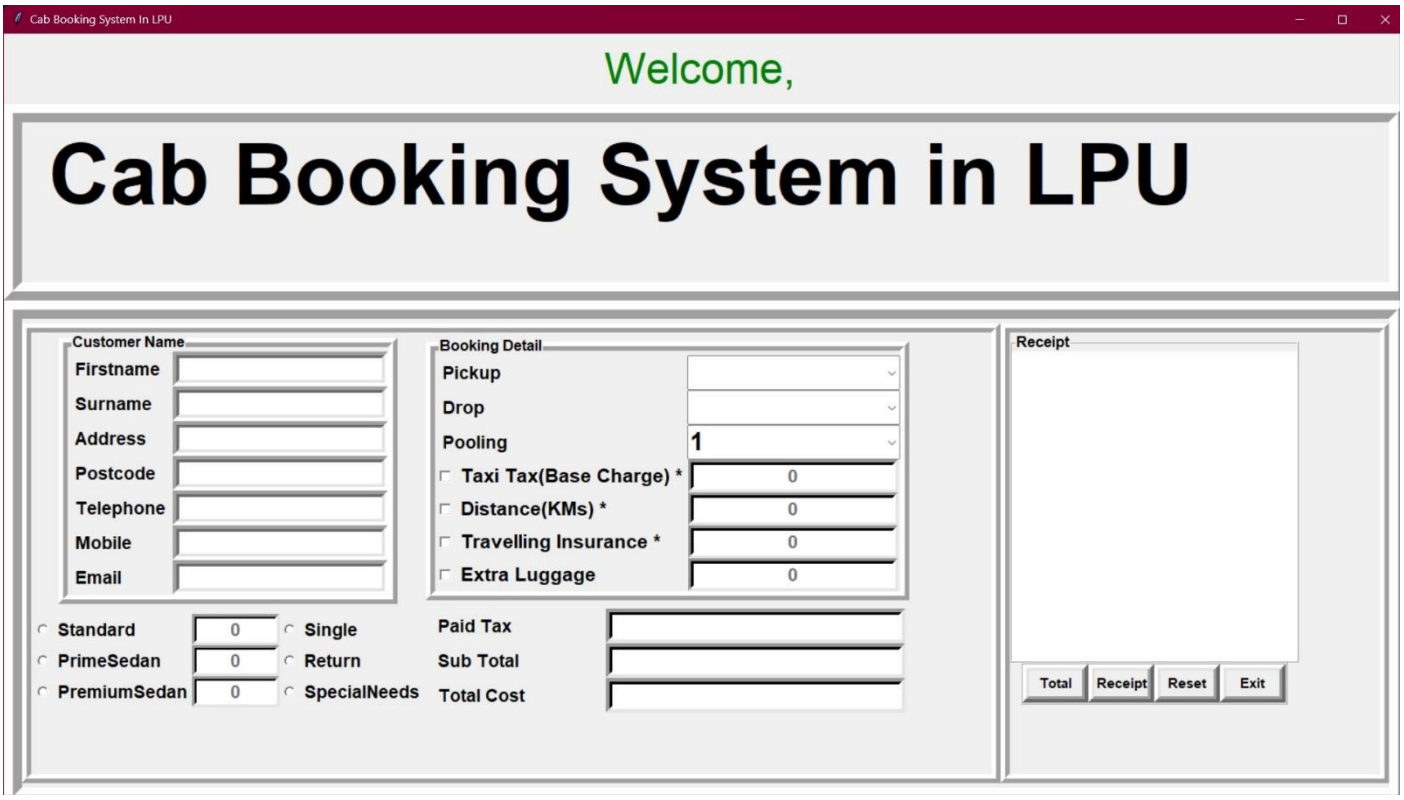
#CONCLUSION:- Thus we conclude our project in which we learnt about GUI and its various functions through which we can make various projects like Capstone portal, scientific calculator , etc in future .

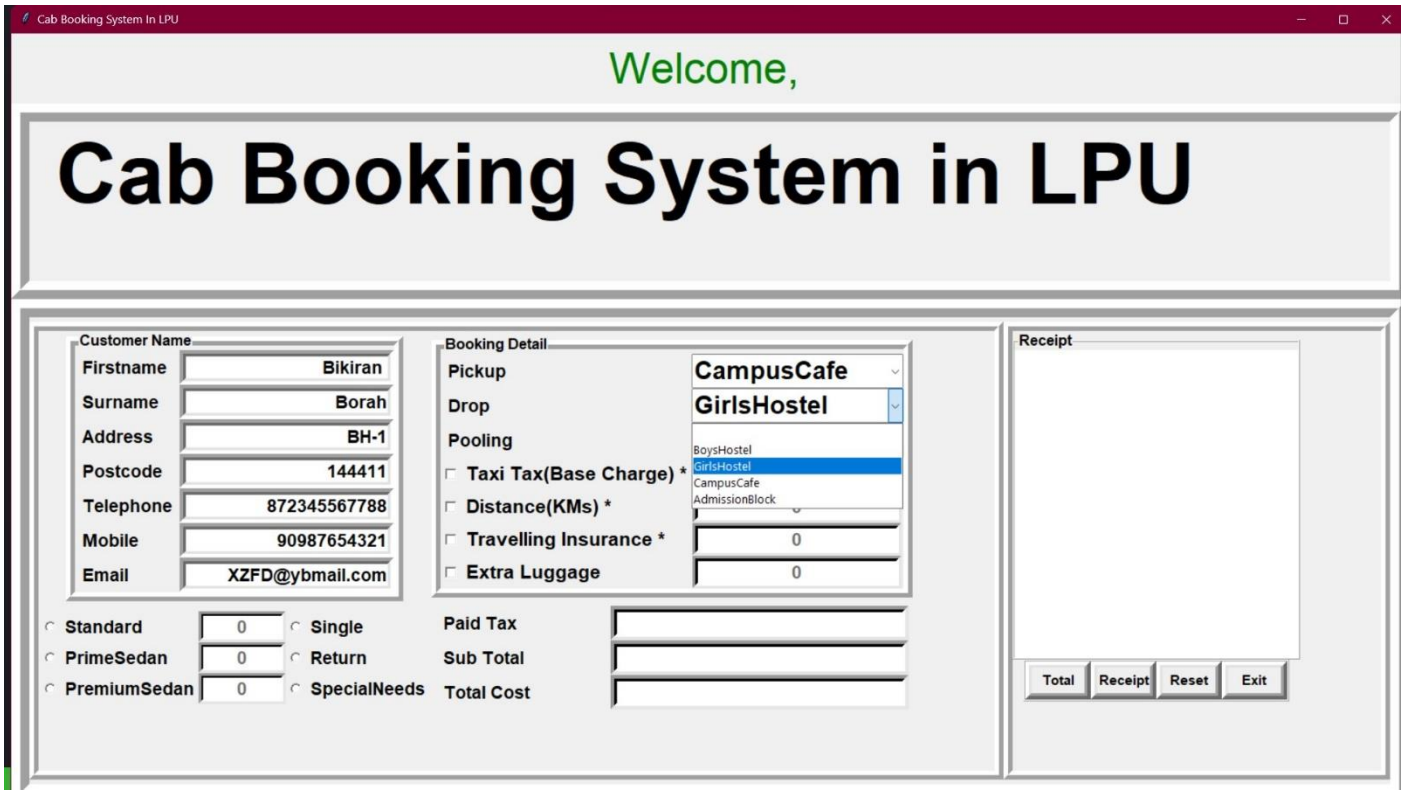
#RESULT SCREENSHOTS:-

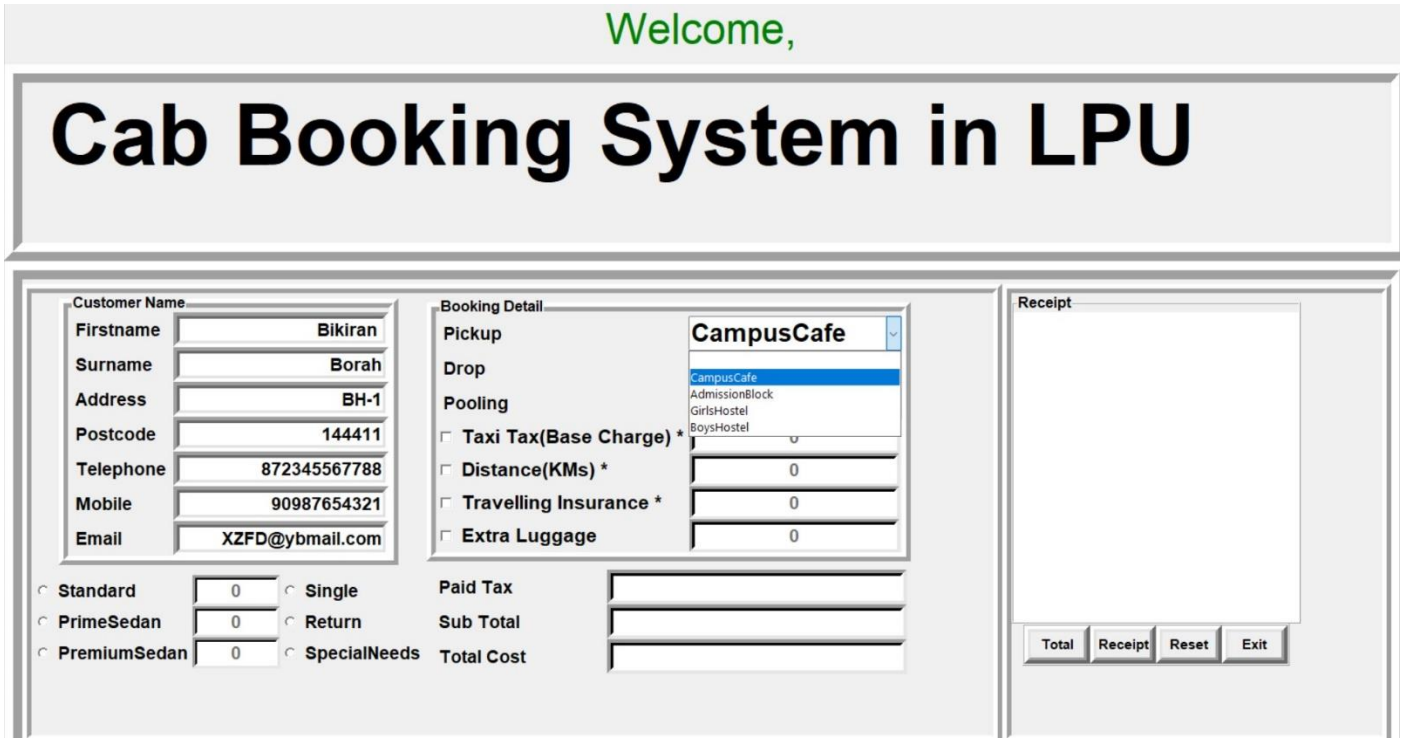


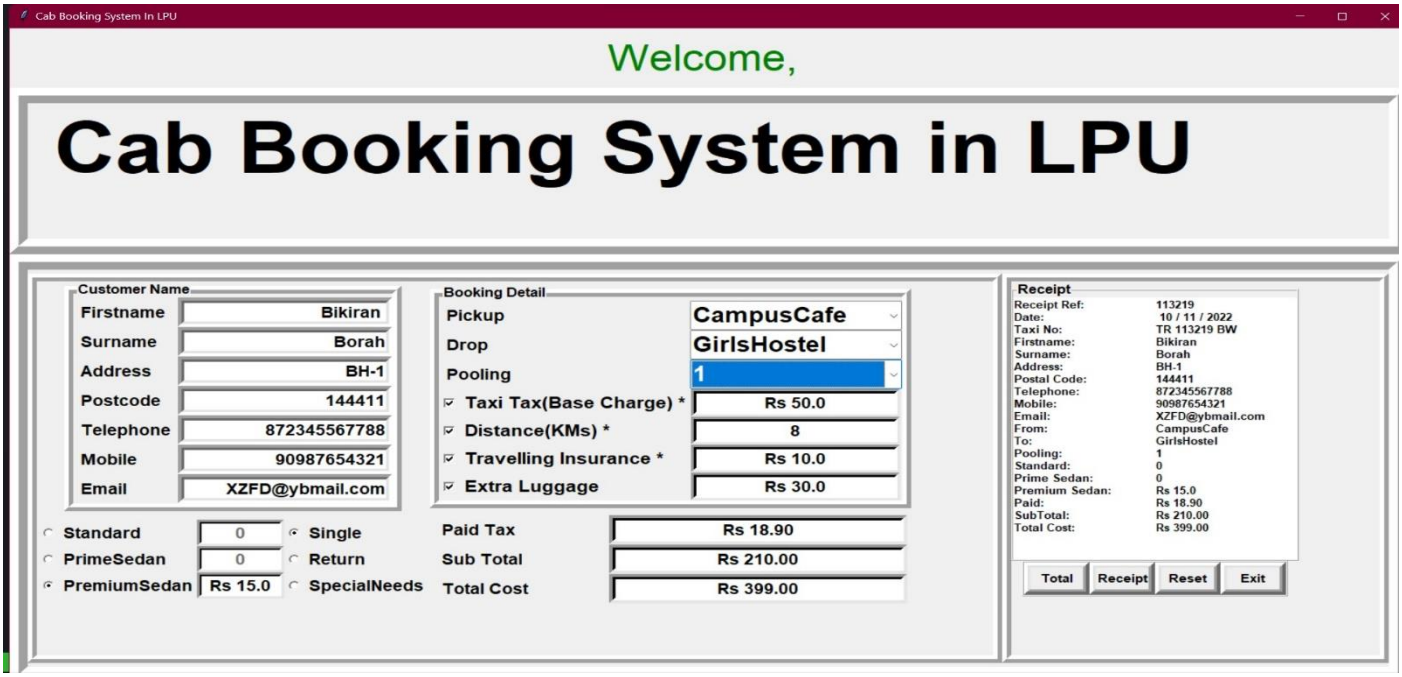



Login Form
Create Account
Username:
Password:
Create Account
Go to Login


Cab Booking System In LPU
Welcome,
Cab Booking System in LPU
Customer Name
Firstname
Surname
Address
Postcode
Telephone
Mobile
Email
Booking Detail
Pickup
Drop
Pooling
1
Taxi Tax(Base Charge) *
Distance(KMs) *
Travelling Insurance *
Extra Luggage
Standard
PrimeSedan
PremiumSedan
Single
Return
SpecialNeeds
Paid Tax
Sub Total
Total Cost
Receipt
Total
Receipt
Reset
Exit


Cab Booking System In LPU
Welcome,
Cab Booking System in LPU
Customer Name
Firstname
Bikiran
Surname
Borah
Address
BH-1
Postcode
144411
Telephone
872345567788
Mobile
90987654321
Email
XZFD@ybmail.com
Booking Detail
Pickup
CampusCafe
Drop
GirlsHostel
Pooling
BoysHostel
GirlsHostel
CampusCafe
AdmissionBlock
Taxi Tax(Base Charge) *
Distance(KMs) *
Travelling Insurance *
0
Extra Luggage
0
Standard
0
Single
PrimeSedan
0
Return
PremiumSedan
0
SpecialNeeds
Paid Tax
Sub Total
Total Cost
Receipt
Total
Receipt
Reset
Exit


Welcome,
Cab Booking System in LPU
Customer Name
Firstname
Bikiran
Surname
Borah
Address
BH-1
Postcode
144411
Telephone
872345567788
Mobile
90987654321
Email
XZFD@ybmail.com
Booking Detail
Pickup
CampusCafe
Drop
CampusCafe
AdmissionBlock
GirlsHostel
BoysHostel
Pooling
Taxi Tax(Base Charge) *
Distance(KMs) *
0
Travelling Insurance *
0
Extra Luggage
0
Standard
0
Single
PrimeSedan
0
Return
PremiumSedan
0
SpecialNeeds
Paid Tax
Sub Total
Total Cost
Receipt
Total
Receipt
Reset
Exit



REFERENCES: -

Python GUI – tkinter – GeeksforGeeks
tkinter- Python interface to Tcl/Tk – Python 3.11.0 documentation.

ANNEXURE A- CODE OF THE PROJECT

```
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="green")
            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', font=('', 35), 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 In LPU")
        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()

        TaxiTax = StringVar()
        Km = StringVar()
        Travel_Ins = StringVar()
        Luggage = StringVar()
        Receipt = StringVar()

        Standard = StringVar()
        PrimeSedan = StringVar()
        PremiumSedan = StringVar()

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

        Standard.set("0")
        PrimeSedan.set("0")
        PremiumSedan.set("0")

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

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

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

            Standard.set("0")
            PrimeSedan.set("0")
            PremiumSedan.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)
            var11.set("0")
            var12.set("0")
            var13.set("0")

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

            self.txtTaxiTax.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.txtPrimeSedan.configure(state=DISABLED)
            self.txtPremiumSedan.configure(state=DISABLED)
            self.reset_counter = 1
```

```
        def Receiptt():
            if reset_counter is 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, 'Taxi 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, PrimeSedan.get() + "\n")
                self.txtReceipt1.insert(END, 'Premium Sedan:\n')
                self.txtReceipt2.insert(END, PremiumSedan.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 Taxi_Tax():
        global Item1
        if var1.get() == 1:
            self.txtTaxiTax.configure(state=NORMAL)
            Item1 = float(50)
            TaxiTax.set("Rs " + str(Item1))
        elif var1.get() == 0:
            self.txtTaxiTax.configure(state=DISABLED)
            TaxiTax.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() == "CampusCafe":
                switch = {"BoysHostel": 10, "GirlsHostel": 8,
                          "AdmissionBlock": 6, "CampusCafe": 0}
                Km.set(switch[varl2.get()])
            elif varl1.get() == "BoysHostel":
                switch = {"BoysHostel": 0, "GirlsHostel": 2,
                          "AdmissionBlock": 5, "CampusCafe": 10}
                Km.set(switch[varl2.get()])
            elif varl1.get() == "GirlsHostel":
                switch = {"BoysHostel": 2, "GirlsHostel": 0,
                          "AdmissionBlock": 3, "CampusCafe": 8}
                Km.set(switch[varl2.get()])
            elif varl1.get() == "AdmissionBlock":
                switch = {"BoysHostel": 5, "GirlsHostel": 3,
                          "AdmissionBlock": 0, "CampusCafe": 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.txtPrimeSedan.configure(state=DISABLED)
                PrimeSedan.set("0")
                self.txtPremiumSedan.configure(state=DISABLED)
                PremiumSedan.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.txtPremiumSedan.configure(state=DISABLED)
                PremiumSedan.set("0")
                self.txtPrimeSedan.configure(state=NORMAL)
                Item5 = float(10)
                PrimeSedan.set("Rs " + str(Item5))
            else:
                self.txtStandard.configure(state=DISABLED)
                Standard.set("0")
                self.txtPrimeSedan.configure(state=DISABLED)
                PrimeSedan.set("0")
                self.txtPremiumSedan.configure(state=NORMAL)
                Item5 = float(15)
                PremiumSedan.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=20, width=1350, relief=RIDGE)
        Tops.pack(side=TOP, fill=BOTH, expand=True)

        self.lblTitle = Label(Tops, font=('arial', 70, 'bold'),
                              text=" Cab Booking System in LPU ")
        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 Name", 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)
```

```
    #
==============================================receipt=================================
================================
        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)

    # ==========================================Taxi
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'] = (
            '', 'CampusCafe', 'AdmissionBlock', 'GirlsHostel', 'BoysHostel')
        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'] = (
            '', 'BoysHostel', 'GirlsHostel', 'CampusCafe', 'AdmissionBlock')
        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)
```

```
    # ============================================Taxi
Information=====================================================================

        self.chkTaxiTax = Checkbutton(TravelFrame, text="Taxi Tax(Base Charge) *",
variable=var1, onvalue=1,
                                      offvalue=0, font=('arial', 16, 'bold'),
command=Taxi_Tax).grid(row=3, column=0, sticky=W)
        self.txtTaxiTax = Label(TravelFrame, font=('arial', 14, 'bold'),
textvariable=TaxiTax,
                                bd=6, width=18, bg="white", state=DISABLED, justify=RIGHT,
relief=SUNKEN)
        self.txtTaxiTax.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=26, 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=26, 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=26, justify=RIGHT,
bg="white", relief=SUNKEN)
        self.txtTotalCost.grid(row=2, column=3)

    #
==========================================================taxiselect=====================
================================================

        self.chkStandard = Radiobutton(Book_Frame, text="Standard", 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.chkPrimeSedand = Radiobutton(Book_Frame, text="PrimeSedan", value=2,
variable=carType, font=(
            'arial', 14, 'bold'), command=selectCar).grid(row=1, column=0, sticky=W)
        self.txtPrimeSedan = Label(Book_Frame, font=('arial', 14, 'bold'), width=7,
                                   textvariable=PrimeSedan, bd=5, state=DISABLED,
justify=RIGHT, bg="white", relief=SUNKEN)
        self.txtPrimeSedan.grid(row=1, column=1)

        self.chkPremiumSedan = Radiobutton(Book_Frame, text="PremiumSedan", value=3,
variable=carType, font=(
            'arial', 14, 'bold'), command=selectCar).grid(row=2, column=0)
        self.txtPremiumSedan = Label(Book_Frame, font=('arial', 14, 'bold'), width=7,
                                     textvariable=PremiumSedan, bd=5, state=DISABLED,
justify=RIGHT, bg="white", relief=SUNKEN)
        self.txtPremiumSedan.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)
```

```
    #
=====================================Recipt=============================================
=======================================

        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()
```