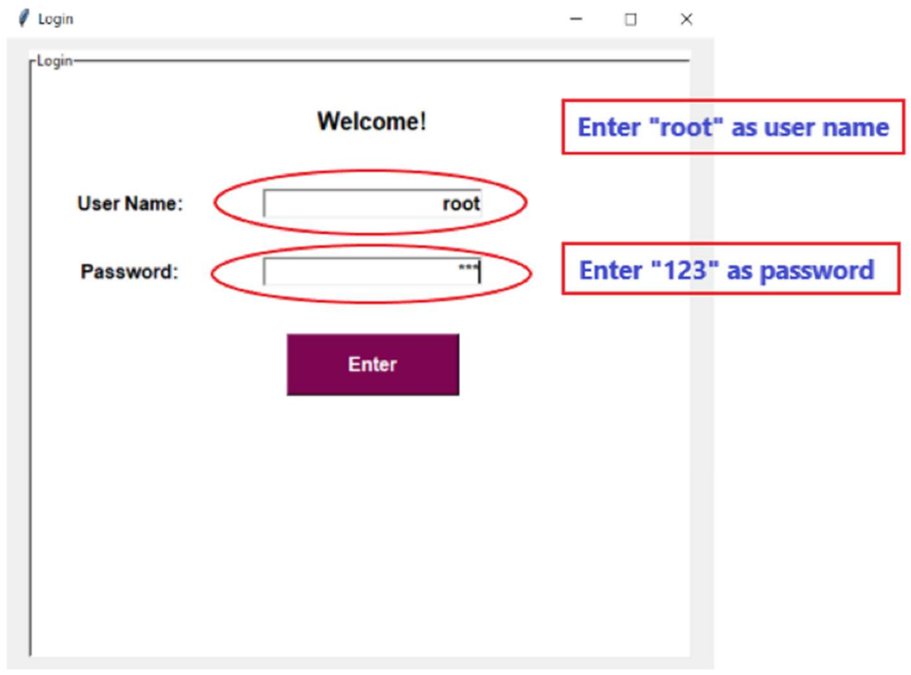
Credentials to Access GUI Interface –

User Name:	root
Password:	123



To get access to the main interface, user must provide above mentioned user name and password (i.e., “root” and “123”, respectively).

Objective

The primary objective of this project is to implement what we've learnt throughout.

Introduction

A restaurant management system (RMS) is an essential tool for any new restaurant. These systems are designed to keep your restaurant running by tracking inventory and sales. A typical RMS setup usually includes both software and hardware, such as a cash register, barcode scanner and receipt printer, depending on how your restaurant is organized. Most importantly, an RMS is a comprehensive tool that allows you to see your restaurant and its needs at a glance, which can simplify your workload on a day-to-day basis.

- **What is This Project is All About –**

  This project provides a very interactive graphical user interface, which can be used to manage the restaurant. Here, first user is asked to login, and by providing proper credentials, user gets access to the main interface.

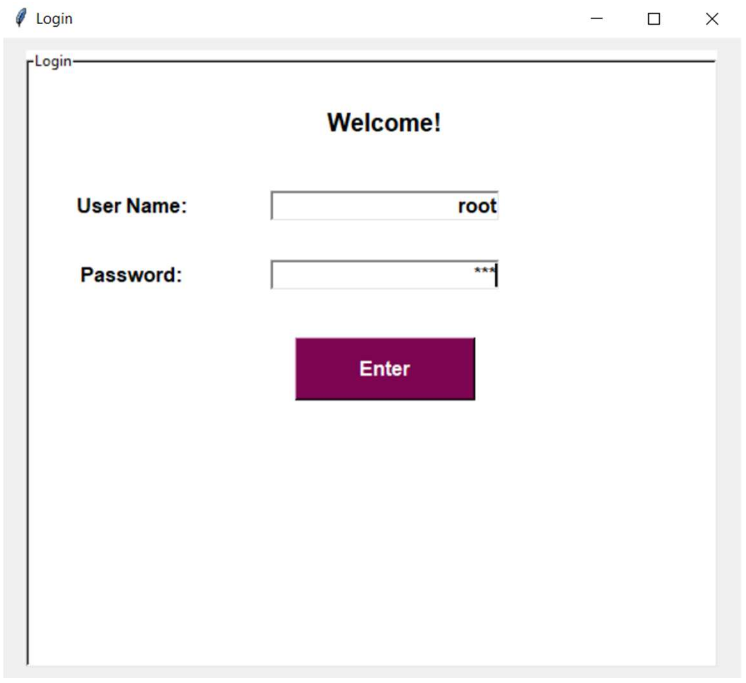
- **Features of GUI –**

  After getting access to the main interface, user can perform various different kinds of functionalities. Such as, user can enter meal quantity and get cost of meal on a single click. User can generate rate card. User can generate bill, by entering customer's name and phone number. User can also generate sales report to keep track of sales. User can exit GUI on a single click of exit button.

Screen Shots of GUI –

LOGIN SCREEN:

a) When user enters, correct user name and password, access is given to the main interface –



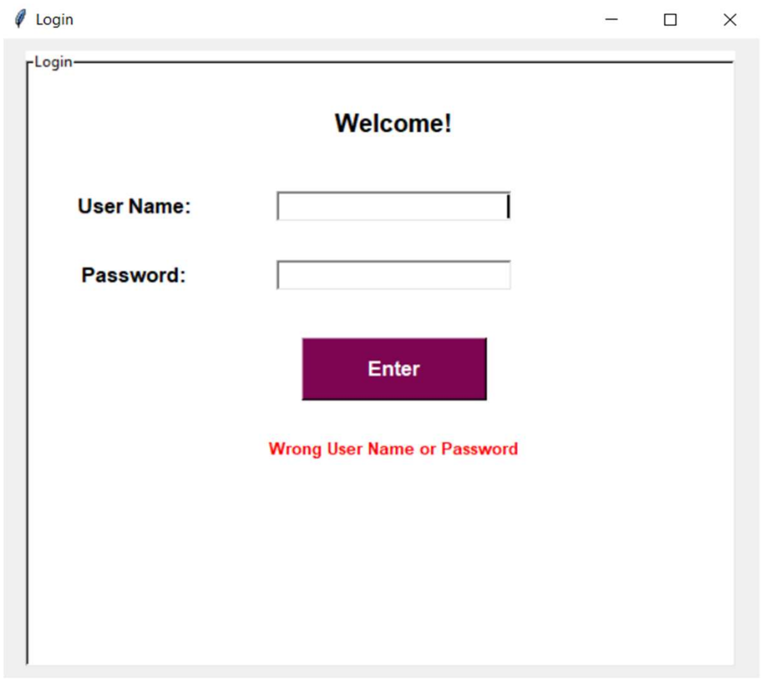
Login

Welcome!

User Name: root

Password: ***

b) When user enters, wrong user name or password (proper message is displayed on the screen) –



Login

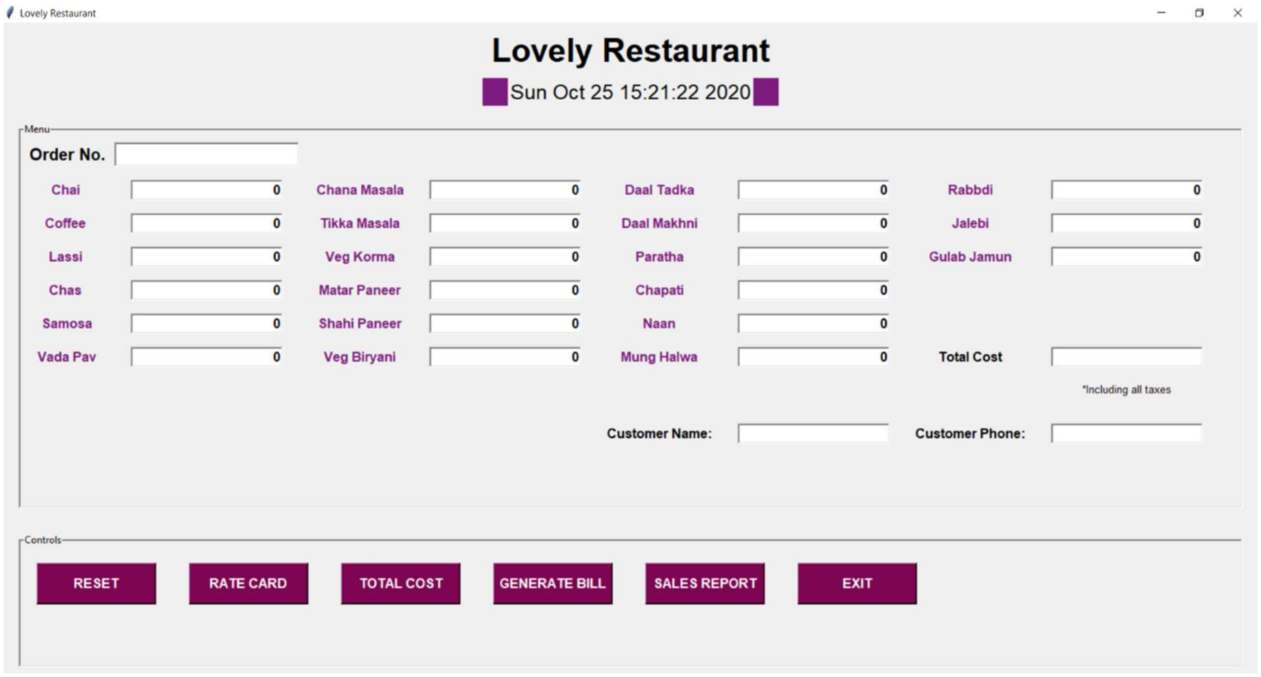
Welcome!

User Name:

Password:

Wrong User Name or Password

Main Interface:



Lovely Restaurant

Sun Oct 25 15:21:22 2020

Order No.

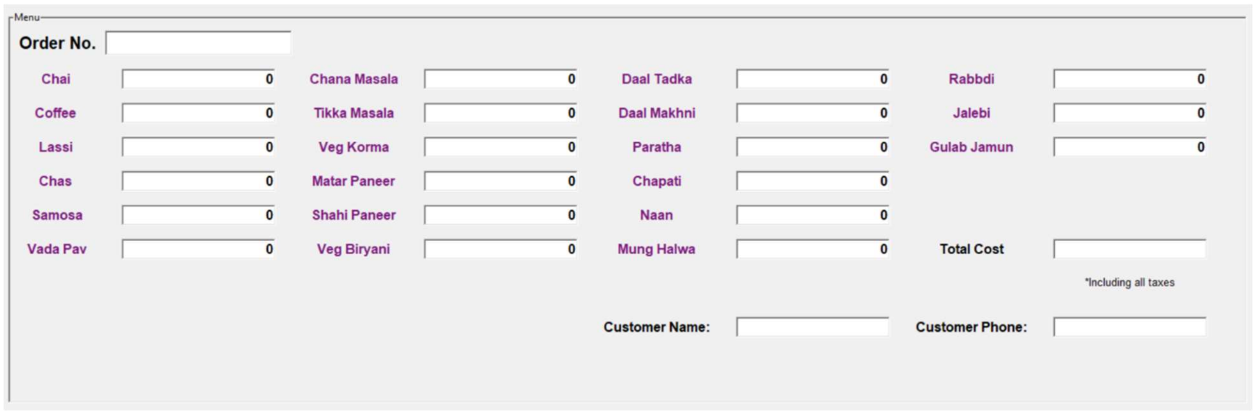
Chai	0	Chana Masala	0	Daal Tadka	0	Rabbdi	0
Coffee	0	Tikka Masala	0	Daal Makhni	0	Jalebi	0
Lassi	0	Veg Korma	0	Paratha	0	Gulab Jamun	0
Chas	0	Matar Paneer	0	Chapati	0		
Samosa	0	Shahi Paneer	0	Naan	0		
Vada Pav	0	Veg Biryani	0	Mung Halwa	0	Total Cost	

*Including all taxes

Customer Name: Customer Phone:

The main interface is divided into two major parts –

1. Menu (where user can give input to the system):



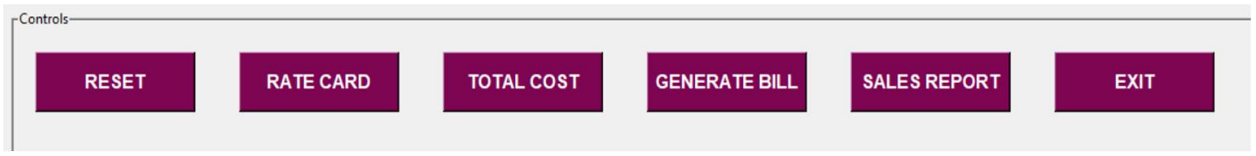
Order No.

Chai	0	Chana Masala	0	Daal Tadka	0	Rabbdi	0
Coffee	0	Tikka Masala	0	Daal Makhni	0	Jalebi	0
Lassi	0	Veg Korma	0	Paratha	0	Gulab Jamun	0
Chas	0	Matar Paneer	0	Chapati	0		
Samosa	0	Shahi Paneer	0	Naan	0		
Vada Pav	0	Veg Biryani	0	Mung Halwa	0	Total Cost	

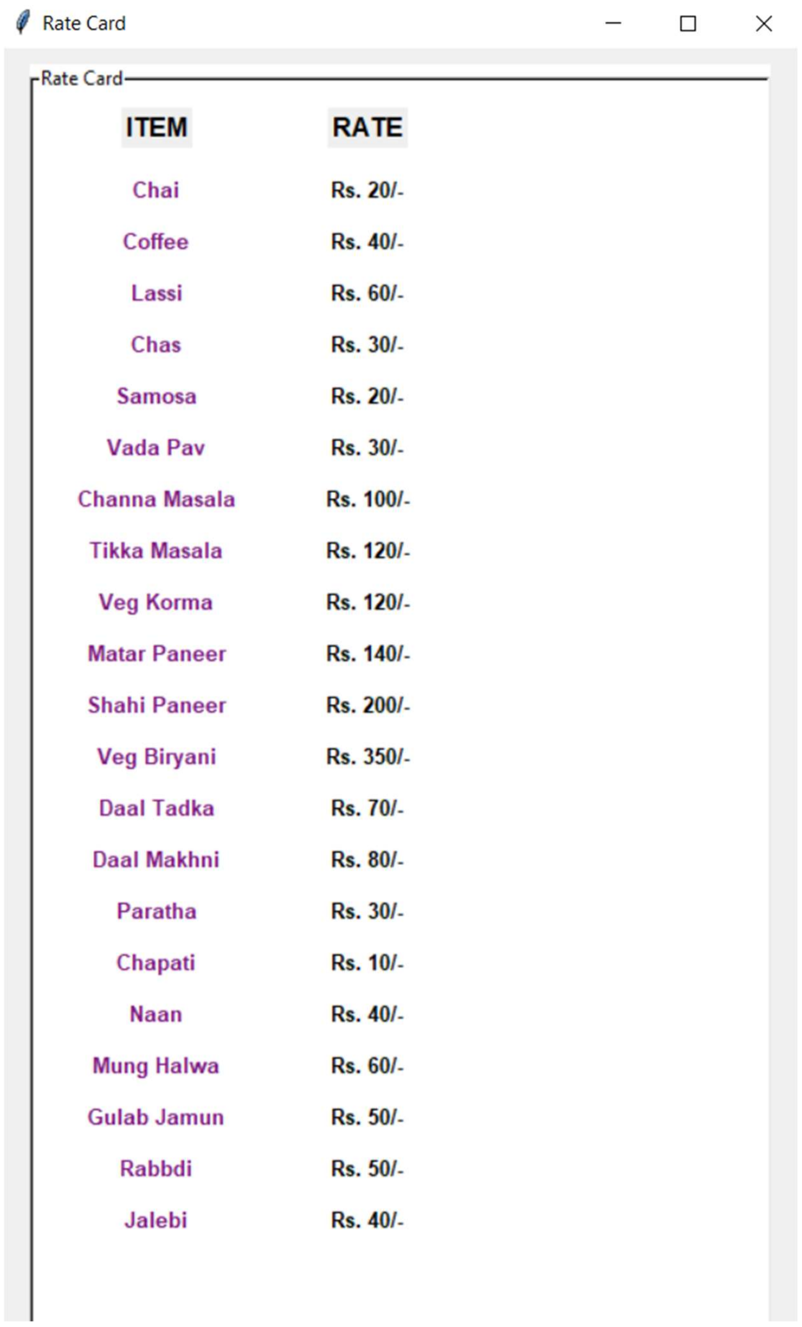
*Including all taxes

Customer Name: Customer Phone:

2. Controls (which consists of buttons, from where user can control the interface):



<u>Rate Card</u> –

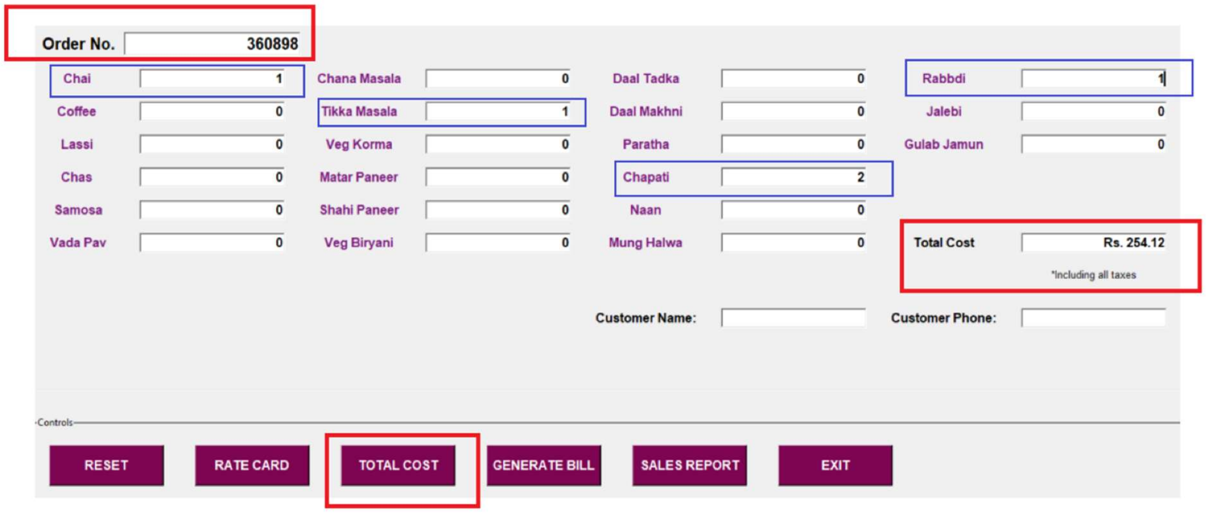

Rate Card

Rate Card

ITEM	RATE
Chai	Rs. 20/-
Coffee	Rs. 40/-
Lassi	Rs. 60/-
Chas	Rs. 30/-
Samosa	Rs. 20/-
Vada Pav	Rs. 30/-
Channa Masala	Rs. 100/-
Tikka Masala	Rs. 120/-
Veg Korma	Rs. 120/-
Matar Paneer	Rs. 140/-
Shahi Paneer	Rs. 200/-
Veg Biryani	Rs. 350/-
Daal Tadka	Rs. 70/-
Daal Makhni	Rs. 80/-
Paratha	Rs. 30/-
Chapati	Rs. 10/-
Naan	Rs. 40/-
Mung Halwa	Rs. 60/-
Gulab Jamun	Rs. 50/-
Rabbdi	Rs. 50/-
Jalebi	Rs. 40/-

When user clicks on “RATE CARD” button, rate card gets generated and displayed on a new window.

<u>Total Cost Function</u> (*on clicking 'TOTAL COST', after entering meal quantity*) –



Order No. 360898

Chai	1	Chana Masala	0	Daal Tadka	0	Rabbdi	1
Coffee	0	Tikka Masala	1	Daal Makhni	0	Jalebi	0
Lassi	0	Veg Korma	0	Paratha	0	Gulab Jamun	0
Chas	0	Matar Paneer	0	Chapati	2		
Samosa	0	Shahi Paneer	0	Naan	0		
Vada Pav	0	Veg Biryani	0	Mung Halwa	0	Total Cost	Rs. 254.12

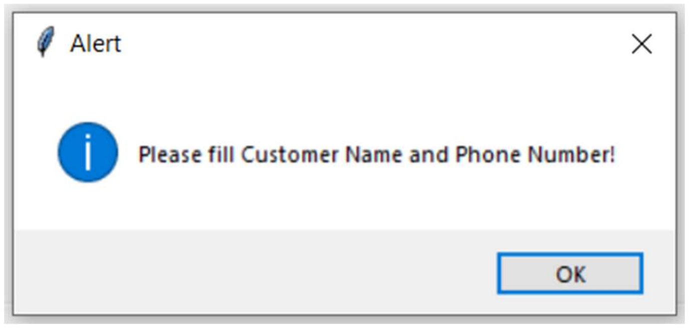
*including all taxes

Customer Name:

Customer Phone:

When user enters meal quantity and clicks on "TOTAL COST" button, then a unique order no. gets generated, along with that, the total cost of meal gets calculated and displayed on "Total Cost" entry box.

<u>Message Box</u> –

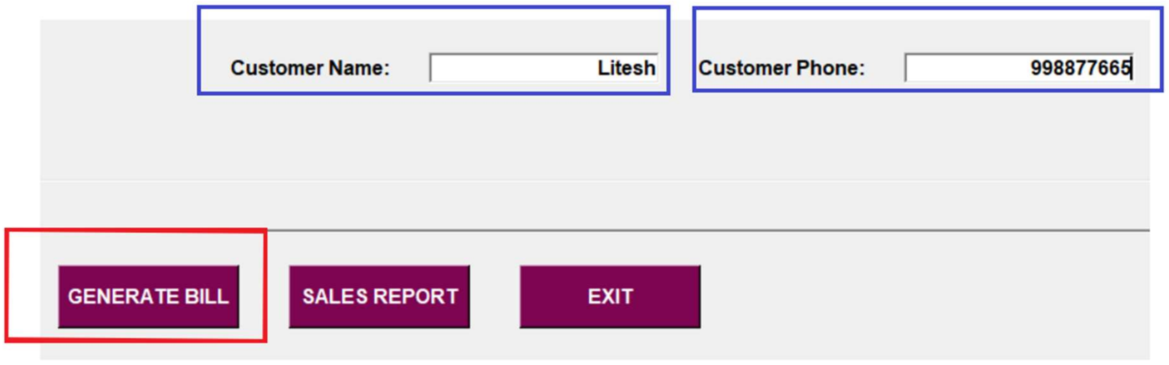


Alert

Please fill Customer Name and Phone Number!

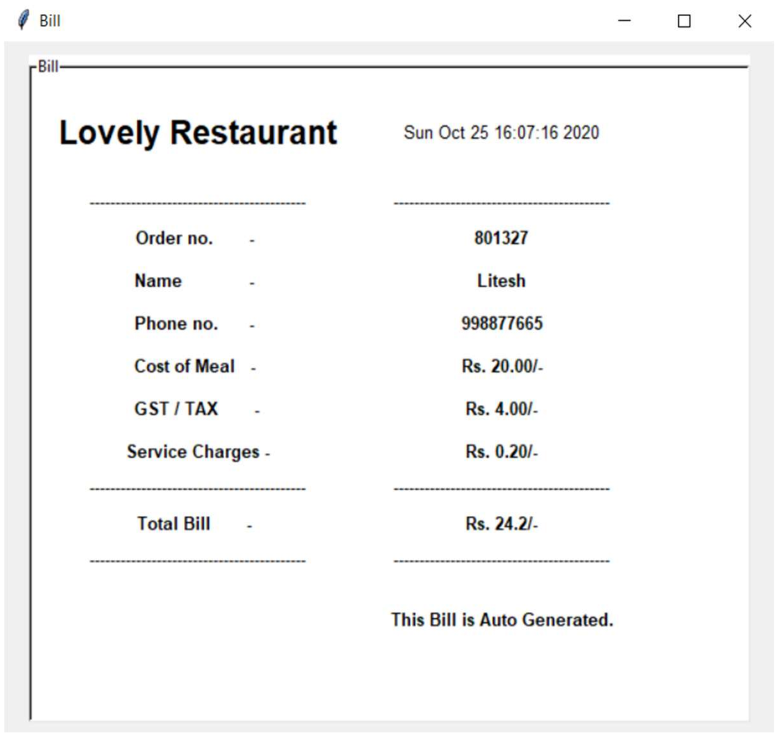
When user clicks on "Generate Bill" Button, without entering customer's name and phone number then, message box gets displayed with proper error message.

BILL GENERATOR –



When user enters proper values in "Customer Name" and "Customer Phone" entry boxes and clicks on "GENERATE BILL" Button, then proper bill gets generated.

PROPER BILL – (*after clicking 'Generate Bill'*)



Bill

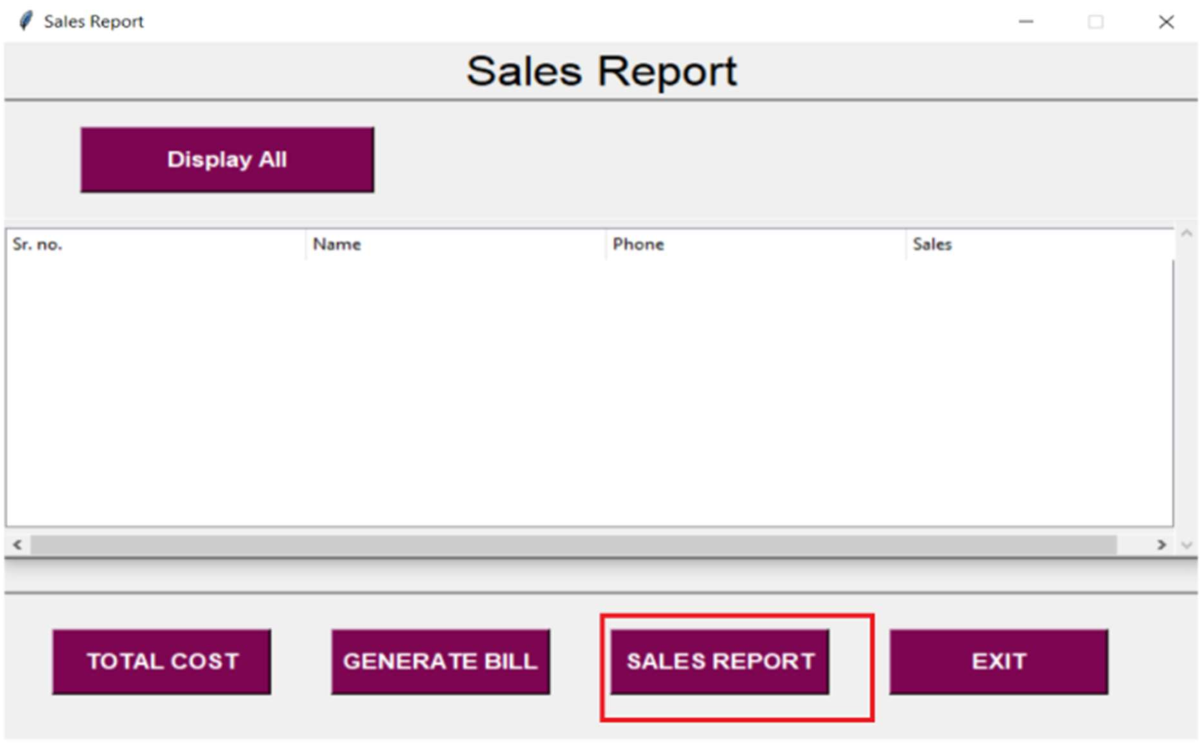
Lovely Restaurant Sun Oct 25 16:07:16 2020

Order no. -	801327
Name -	Litesh
Phone no. -	998877665
Cost of Meal -	Rs. 20.00/-
GST / TAX -	Rs. 4.00/-
Service Charges -	Rs. 0.20/-
Total Bill -	Rs. 24.2/-

This Bill is Auto Generated.

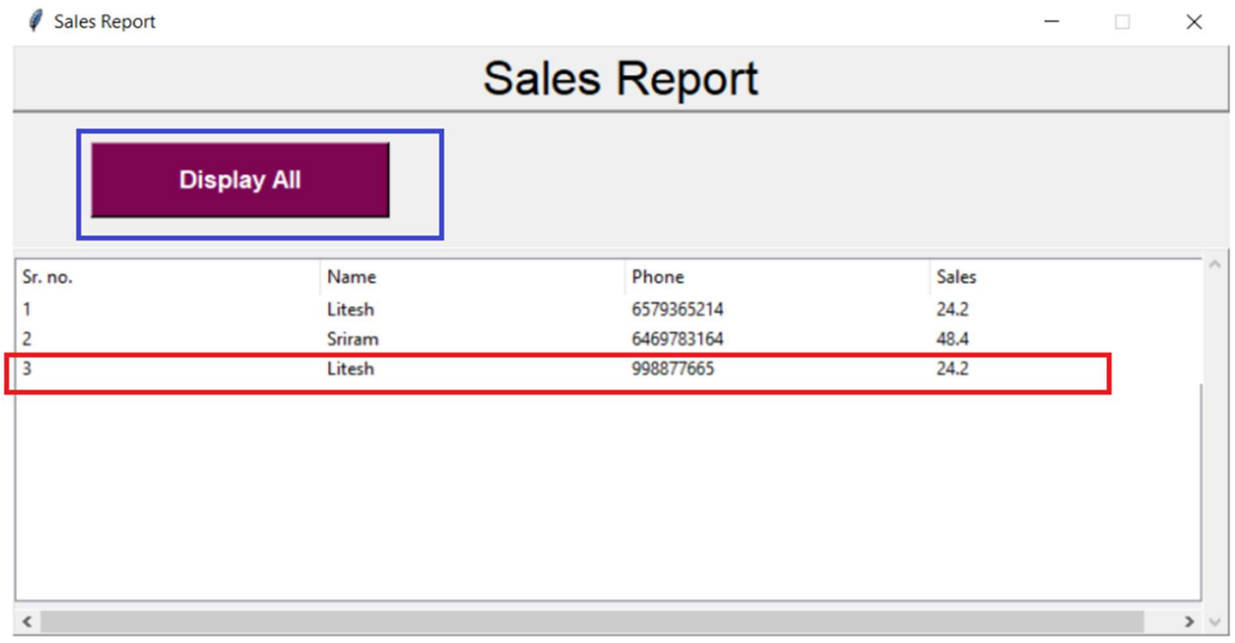
When user clicks on "Generate Bill" button after entering proper customer name and phone number value, then bill gets generated in new window.

<u>SALES REPORT FUNCTION</u> – (*on clicking 'SALES REPORT' button*)



When user clicks on "SALES REPORT" button, a new window gets generated. The new window has "Display All" button and a tree view. If the user clicks on "Display All" button then records gets displayed, as shown below-

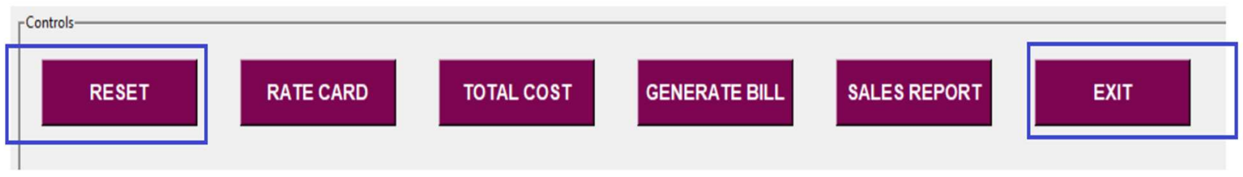
Sales Report –



Sr. no.	Name	Phone	Sales
1	Litesh	6579365214	24.2
2	Sriram	6469783164	48.4
3	Litesh	998877665	24.2

When user clicks on “Display all” button, all the records from the database gets fetched and displayed on the tree view, with the latest entered record at the bottom.

RESET and EXIT Button –



When user clicks on “RESET” button the interface gets reset and when user clicks on the “EXIT” button, then the interface gets closed.

Source Code:

Link –

https://github.com/LiteshGhute/Restaurant-Management-System-Using-Python-and-Sqlite

Video Demonstration:

Link – https://youtu.be/yLNGhHCE9l4

Libraries/Packages/modules used –

- **tkinter**

  def: Tkinter library is used to create graphical user interface (GUI).

- **from functools partial**

  def: Partial functions allow us to fix a certain number of arguments of a function and generate a new function.

- **from tkinter messagebox**

  def: The messagebox module is used to display the message boxes in the python applications.

- **ttk from tkinter.ttk**

  def: We are using this module in our project to create tree view.

- **random**

  def: To generate random numbers.

- **time**

  def: To get local time of the computer, and store it into variable.

- **sqlite3**

  def: To connect and store records to the database. And to retrieve record from database.

<u>Functions used in the project:</u>

- **<u>app()</u>**

  def: app function is the main function in the program, which is having the main source code, and this function is having other functions listed below:

  - **<u>Cost()</u>**

    def: This function is fetching values from entry boxes and calculating the cost of meal, and after calculating cost, it returning those values.

  - **<u>destroy()</u>**

    def: This function is destroying the root window.

  - **<u>reset()</u>**

    def: This function is resetting the values of entry boxes.

  - **<u>customer()</u>**

    def: This function is fetching value from customer name and phone number entry boxes, and displaying message if entry boxes is not filled.

- **<u>Database()</u>**

  def: This function is making connection with database and creating table if it does not exist.

- **<u>insert(cname, cphone, csales)</u>**

  def: This function is taking three arguments and storing the values of these arguments into the database.

- **<u>Rate()</u>**

  def: This function is generating rate card (price card) in a new window.

- **<u>bill()</u>**

  def: This function is collecting the values returned by Cost function and using those values to generate bill in a separate window.

- **<u>report()</u>**

  def: This function is creating separate window to display sales report.

- **<u>populateView()</u>**

  def: This function is fetching values from the database and inserting those values in tree view.

➢ **<u>validateLogin()</u>**

def: This function is validating login and calling app function (main function) if entered user name and password is correct.

➢ <u>**Important variables used in the code:**</u>

- **logWin:** variable used to store login window details.
- **root:** variable used to store root window or main window details.
- **Heading:** variable used to store heading frame details.
- **Menu:** variable used to store menu frame details.
- **Controls:** variables used to store control frame details.
- **localtime:** variable used to store local time of the system.
- **lblhead:** variables used to store heading label details.
- **logFrame:** variable used to store login frame details.
- **password:** variable used to store entered password.
- **username:** variable used to store entered username.
- **btnEnter:** variable used to store login button details.
- **btnReset:** variable used to store reset button details.
- **btnRate:** variable used to store rate card button details.
- **btnTotal:** variable used to store total button details.
- **btnBill:** variable used to store generate bill button details.
- **btnsalesReport:** variable used to store sales report button details.
- **btnExit:** variable used to store exit button details.
- **conn:** variable used to store connection details.
- **cursor:** variable used to store cursor details.
- **lblinfo:** variable used to store label information.
- **repo:** variable used to store sales report window details.
- **billWin:** variable used to store bill window details.
- **rateWin:** variable used to store rate card/price card details.
- **Chai:** variable used to store chai entry box value.
- **Coffee:** variable used to store coffee entry box value.
- **Lassi:** variable used to store lassi entry box value.
- **Coffee:** variable used to store coffee entry box value.
- **Chas:** variable used to store chas entry box value.
- **Samosa:** variable used to store samosa entry box value.
- **coffee:** variable used to store coffee entry box value.
- **Vada_Pav:** variable used to store vada pav entry box value.
- **Chana_Masala:** variable used to store chana masala entry box value.

- **Tikka_Masala:** variable used to store tikka masala entry box value.
- **Veg_Korma:** variable used to store veg korma entry box value.
- **Matar_Paneer:** variable used to store matar paneer entry box value.
- **Shahi_Paneer:** variable used to store shahi paneer entry box value.
- **Veg_Biryani:** variable used to store veg biryani entry box value.
- **Daal_Tadka:** variable used to store daal tadka entry box value.
- **Daal_Makhni:** variable used to store daal makhni entry box value.
- **Paratha:** variable used to store Paratha entry box value.
- **Roti:** variable used to store chapati entry box value.
- **Naan:** variable used to store naan entry box value.
- **Mung_Halwa:** variable used to store mung halwa entry box value.
- **Gulab_Jamun:** variable used to store gulab jamun entry box value.
- **Rabbdi:** variable used to store rabbdi entry box value.
- **Jalebi:** variable used to store jalebi entry box value.
- **Total:** variable used to store total entry box value.
- **rand:** variable used to store order number entry box value.
- **cusName:** variable used to store customer's name entry box value.
- **cusPhone:** variable used to store customer's phone entry box value.
- **quantity_chai:** variable used to store quantity of chai.
- **quantity_coffee:** variable used to store quantity of coffee.
- **quantity_samosa:** variable used to store quantity of samosa.
- **quantity_chas:** variable used to store quantity of chas.
- **quantity_vada_pav:** variable used to store quantity of vada pav.
- **quantity_chana_masala:** variable used to store quantity of chana masala.
- **quantity_tikka_masala:** variable used to store quantity of tikka masala.
- **quantity_veg_korma:** variable used to store quantity of veg korma.
- **quantity_matar_paneer:** variable used to store quantity of matar paneer.
- **quantity_shahi_paneer:** variable used to store quantity of shahi paneer.
- **quantity_veg_biryani:** variable used to store quantity of veg biryani.
- **quantity_daal_tadka:** variable used to store quantity of daal tadka.
- **quantity_daal_makhni:** variable used to store quantity of daal makhni.
- **quantity_paratha:** variable used to store quantity of chapati.
- **quantity_naan:** variable used to store quantity of naan.

- **quantity_mung_halwa:** variable used to store quantity of mung halwa.
- **quantity_gulab_jamun:** variable used to store quantity of gulab jamun.
- **quantity_rabbdi:** variable used to store quantity of rabbdi.
- **quantity_jalebi:** variable used to store quantity of jalebi.
- **costofChai:** variable used to store total calculated cost of chai.
- **costofCoffee:** variable used to store total calculated cost of coffee.
- **costofChai:** variable used to store total calculated cost of chai.
- **costofChas:** variable used to store total calculated cost of chas.
- **costofSamosa:** variable used to store total calculated cost of samosa.
- **costofVadaPav:** variable used to store total calculated cost of vada pav.
- **costofChanaMasala:** variable used to store total calculated cost of chana masala.
- **costofTikkaMasala:** variable used to store total calculated cost of tikka masala.
- **costofVegKorma:** variable used to store total calculated cost of veg korma.
- **costofMatarPaneer:** variable used to store total calculated cost of matar paneer.
- **costofShahiPaneer:** variable used to store total calculated cost of shahi paneer.
- **costofDaalTadka:** variable used to store total calculated cost of daal tadka.
- **costofDaalMakhni:** variable used to store total calculated cost of daal makhni.
- **costofBiryani:** variable used to store total calculated cost of biryani.
- **costofRoti:** variable used to store total calculated cost of chapati.
- **costofParatha:** variable used to store total calculated cost of paratha.
- **costofNaan:** variable used to store total calculated cost of naan.
- **costofMungHalwa:** variable used to store total calculated cost of mung halwa.
- **costofGulabJamun:** variable used to store total calculated cost of gulab jamun.
- **costofRabbdi:** variable used to store total calculated cost of rabbdi.
- **costofJalebi:** variable used to store total calculated cost of jalebi.

Results

We finally got the end product as a ‘Restaurant Management System’ that includes all the mentioned modules. We learnt how to make a GUI using Tkinter in Python and also learnt to implement database connectivity using sqlite3.

- The final product is capable of checking proper login credentials and giving access if entered credentials are correct.
- The final product is capable of calculating the cost of meal with taxes.
- The final product is capable of displaying menu card (Rate card).
- The final product is capable of generating a proper bill.
- The final product is capable of displaying proper error message in massage box, if user doesn’t enter customer’s name and phone number and tries to generate bill.
- The final product is capable of displaying sales record and storing those record into the database.
- The final product is capable of resetting the whole interface on a single click of reset button and also capable of closing the whole interface on a single of exit button.
- The final product is properly tested and ready to deploy on the field.

Complete Source Code –

```
#                              Restaurant Management System
# user name: root
# password:  123
from tkinter import *
from functools import partial
from tkinter import messagebox
import tkinter.ttk as ttk
import random
import time
import sqlite3

def app():                                    # main funtion

   logWin.destroy()                              # destroying login window
   root = Tk()                                # creating main root window
   root.geometry("1600x750+0+0")
   root.title("Lovely Restaurant")

   Heading = Frame(root,bg="#7D1B7E",width = 1600,height=50,relief=SUNKEN)
#creating frame for headers
   Heading.pack(side=TOP)

   Menu = LabelFrame(root,width = 900,height=640,relief=SUNKEN, text="Menu")
#creating label frame for menus
   Menu.pack(fill= "both", expand="yes", padx=20, pady=20)



   Controls = LabelFrame(root,width = 900,height=300,relief=SUNKEN, text="Controls")
#creatinf label fram for controls
```

```
    Controls.pack(fill= "both", expand="yes", padx=20, pady=10)




    localtime=time.asctime(time.localtime(time.time()))             # storing local time in variable



    # setting hearders
    lblhead = Label(Heading, font=( 'aria' ,30, 'bold' ),text="Lovely 
Restaurant",fg="Black",bd=10,anchor='w')
    lblhead.grid(row=0,column=0)
    lblhead = Label(Heading, font=( 'aria' ,19, ),text=localtime,fg="black",anchor=W)
    lblhead.grid(row=1,column=0)




    def Cost():                                         # funtion to calculate cost of meal and return those 
values
        x = random.randint(10000, 1000000)                  # storing random number in variable
        randomRef = str(x)
        rand.set(randomRef)

        #extracting values from boxes and type-casting it to float

        quantity_chai         = float(Chai.get())
        quantity_coffee       = float(Coffee.get())
        quantity_lassi        = float(Lassi.get())
        quantity_chas         = float(Chas.get())
        quantity_samosa       = float(Samosa.get())
        quantity_vada_pav     = float(Vada_Pav.get())
```

```
quantity_chana_masala = float(Chana_Masala.get())
quantity_tikka_masala = float(Tikka_Masala.get())
quantity_veg_korma    = float(Veg_Korma.get())
quantity_matar_paneer = float(Matar_Paneer.get())
quantity_shahi_paneer = float(Shahi_Paneer.get())

quantity_veg_biryani  = float(Veg_Biryani.get())
quantity_daal_tadka   = float(Daal_Tadka.get())
quantity_daal_makhni  = float(Daal_Makhni.get())
quantity_paratha      = float(Paratha.get())
quantity_roti         = float(Roti.get())
quantity_naan         = float(Naan.get())

quantity_mung_halwa   = float(Mung_Halwa.get())
quantity_gulab_jamun  = float(Gulab_Jamun.get())
quantity_rabbdi       = float(Rabbdi.get())
quantity_jalebi       = float(Jalebi.get())


# calculating cost of items ordered

costofChai           = quantity_chai     * 20
costofCoffee          = quantity_coffee   * 40
costofLassi          = quantity_lassi    * 60
costofChas           = quantity_chas     * 30
costofsamosa          = quantity_samosa   * 20
costofVadaPav         = quantity_vada_pav * 30
```

```
    costofChanaMasala     = quantity_chana_masala * 100
    costofTikkaMasala     = quantity_tikka_masala * 120
    costofVegKorma        = quantity_veg_korma    * 120
    costofMatarPaneer     = quantity_matar_paneer * 140
    costofShahiPaneer     = quantity_shahi_paneer * 200


    costofVegBiryani      = quantity_veg_biryani * 350
    costofDaalTadka       = quantity_daal_tadka  * 70
    costofDaalMakhni      = quantity_daal_makhni * 80
    costofParatha         = quantity_paratha     * 30
    costofRoti            = quantity_roti        * 10
    costofNaan            = quantity_naan        * 40


    costofMungHalwa       = quantity_mung_halwa  * 60
    costofGulabJamun      = quantity_gulab_jamun * 50
    costofRabbdi          = quantity_rabbdi      * 50
    costofJalebi          = quantity_jalebi      * 40



    # performing various calculations and storing there values in variables

    costofmeal = ("Rs.",str('%.2f'% (costofChai + costofCoffee + costofLassi + costofChas +
costofsamosa + costofVadaPav + costofChanaMasala + costofTikkaMasala + costofVegKorma +
costofMatarPaneer + costofShahiPaneer + costofVegBiryani + costofDaalTadka +
costofDaalMakhni + costofParatha + costofRoti + costofNaan + costofMungHalwa +
costofGulabJamun + costofRabbdi + costofJalebi)))

    PayTax=((costofChai + costofCoffee + costofLassi + costofChas + costofsamosa +
costofVadaPav + costofChanaMasala + costofTikkaMasala + costofVegKorma +
costofMatarPaneer + costofShahiPaneer + costofVegBiryani + costofDaalTadka +
costofDaalMakhni + costofParatha + costofRoti + costofNaan + costofMungHalwa +
costofGulabJamun + costofRabbdi + costofJalebi)*0.2)

    Totalcost=(costofChai + costofCoffee + costofLassi + costofChas + costofsamosa +
costofVadaPav + costofChanaMasala + costofTikkaMasala + costofVegKorma +
costofMatarPaneer + costofShahiPaneer + costofVegBiryani + costofDaalTadka +
costofDaalMakhni + costofParatha + costofRoti + costofNaan + costofMungHalwa +
costofGulabJamun + costofRabbdi + costofJalebi)

    Ser_Charge=((costofChai + costofCoffee + costofLassi + costofChas + costofsamosa +
costofVadaPav + costofChanaMasala + costofTikkaMasala + costofVegKorma +
costofMatarPaneer + costofShahiPaneer + costofVegBiryani + costofDaalTadka +
costofDaalMakhni + costofParatha + costofRoti + costofNaan + costofMungHalwa +
costofGulabJamun + costofRabbdi + costofJalebi)/99.2)

    Service="Rs.",str('%.2f'% Ser_Charge)

    OverAllCost="Rs.",str(round((PayTax + Totalcost + Ser_Charge),2))

    PaidTax="Rs.",str('%.2f'% PayTax)


    Service_Charge.set(Service)

    cost.set(costofmeal)

    Tax.set(PaidTax)

    Subtotal.set(costofmeal)

    Total.set(OverAllCost)

    lst = [costofmeal, PaidTax, Service, OverAllCost, randomRef]

    return lst      # returning values for future use



def exit():                     # funtion to destroy main root window

    root.destroy()




def reset():                    # funtion to reset the different values specified in boxes
```

```
Chai.set(0)
Coffee.set(0)
Lassi.set(0)
Chas.set(0)
Samosa.set(0)
Vada_Pav.set(0)
Chana_Masala.set(0)
Tikka_Masala.set(0)
Veg_Korma.set(0)
Matar_Paneer.set(0)
Shahi_Paneer.set(0)
Veg_Biryani.set(0)
Daal_Tadka.set(0)
Daal_Makhni.set(0)
Paratha.set(0)
Roti.set(0)
Naan.set(0)
Mung_Halwa.set(0)
Gulab_Jamun.set(0)
Rabbdi.set(0)
Jalebi.set(0)
rand.set("")
Subtotal.set("")
Total.set("")
Service_Charge.set("")
Tax.set("")
cost.set("")
CusName.set("")
CusPhone.set("")
```

```
# declaring various String Variables for future use

Chai = StringVar()
Coffee = StringVar()
Lassi = StringVar()
Chas = StringVar()
Samosa = StringVar()
Vada_Pav = StringVar()
Chana_Masala = StringVar()
Tikka_Masala = StringVar()
Veg_Korma = StringVar()
Matar_Paneer = StringVar()
Shahi_Paneer = StringVar()
Veg_Biryani = StringVar()
Daal_Tadka = StringVar()
Daal_Makhni = StringVar()
Paratha = StringVar()
Roti = StringVar()
Naan = StringVar()
Mung_Halwa = StringVar()
Gulab_Jamun = StringVar()
Rabbdi = StringVar()
Jalebi = StringVar()

rand = StringVar()
Subtotal = StringVar()
Total = StringVar()
Service_Charge = StringVar()
```

```
    Tax = StringVar()
    cost = StringVar()

    CusName = StringVar()
    CusPhone = StringVar()

    # creating various menus on main interface

    lblorder = Label(Menu, font=( 'aria' ,15, 'bold' ),text="Order
No.",fg="Black",bd=10,anchor='w')
    lblorder.grid(row=0,column=0)
    txtorder = Entry(Menu,font=('ariel' ,15,'bold'), textvariable=rand , bd=2 ,justify='right')
    txtorder.grid(row=0,column=1)



    lblChai = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Chai
",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblChai.grid(row=2,column=0)
    txtChai = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Chai , bd=2,justify='right')
    txtChai.grid(row=2,column=1)

    lblCoffee = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Coffee
",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblCoffee.grid(row=3,column=0)
    txtCoffee = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Coffee , bd=2,justify='right')
    txtCoffee.grid(row=3,column=1)

    lblLassi = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Lassi
",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblLassi.grid(row=4,column=0)
    txtLassi = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Lassi , bd=2,justify='right')
    txtLassi.grid(row=4,column=1)


    lblChas = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Chas
",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblChas.grid(row=5,column=0)
    txtChas = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Chas , bd=2,justify='right')
    txtChas.grid(row=5,column=1)


    lblSamosa = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Samosa",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblSamosa.grid(row=6,column=0)
    txtSamosa = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Samosa , bd=2,justify='right')
    txtSamosa.grid(row=6,column=1)


    lblVada_Pav = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Vada
Pav",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblVada_Pav.grid(row=7,column=0)
    txtVada_Pav = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Vada_Pav ,
bd=2,justify='right')
    txtVada_Pav.grid(row=7,column=1)




    lblChana_Masala = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Chana
Masala",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblChana_Masala.grid(row=2,column=3,padx=10)
    txtChana_Masala = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Chana_Masala ,
bd=2,justify='right')
    txtChana_Masala.grid(row=2,column=4)


    lblTikka_Masala = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Tikka
Masala",fg="#7D1B7E",bd=10,anchor='w',justify='right')
```

```
    lblTikka_Masala.grid(row=3,column=3,padx=10)
    txtTikka_Masala = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Tikka_Masala ,
bd=2,justify='right')
    txtTikka_Masala.grid(row=3,column=4)


    lblVeg_Korma = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Veg
Korma",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblVeg_Korma.grid(row=4,column=3,padx=10)
    txtVeg_Korma = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Veg_Korma ,
bd=2,justify='right')
    txtVeg_Korma.grid(row=4,column=4)


    lblMatar_Paneer = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Matar
Paneer",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblMatar_Paneer.grid(row=5,column=3,padx=10)
    txtMatar_Paneer = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Matar_Paneer ,
bd=2,justify='right')
    txtMatar_Paneer.grid(row=5,column=4,padx=10)


    lblShahi_Paneer = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Shahi
Paneer",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblShahi_Paneer.grid(row=6,column=3,padx=10)
    txtShahi_Paneer = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Shahi_Paneer ,
bd=2,justify='right')
    txtShahi_Paneer.grid(row=6,column=4,padx=10)


    lblVeg_Biryani = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Veg
Biryani",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblVeg_Biryani.grid(row=7,column=3,padx=10)
    txtVeg_Biryani = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Veg_Biryani ,
bd=2,justify='right')
    txtVeg_Biryani.grid(row=7,column=4,padx=10)
    lblDaal_Tadka = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Daal
Tadka",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblDaal_Tadka.grid(row=2,column=5,padx=10)
    txtDaal_Tadka = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Daal_Tadka ,
bd=2,justify='right')
    txtDaal_Tadka.grid(row=2,column=6,padx=10)

    lblDaal_Makhni = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Daal
Makhni",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblDaal_Makhni.grid(row=3,column=5,padx=10)
    txtDaal_Makhni = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Daal_Makhni ,
bd=2,justify='right')
    txtDaal_Makhni.grid(row=3,column=6,padx=10)

    lblParatha = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Paratha",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblParatha.grid(row=4,column=5,padx=10)
    txtParatha = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Paratha , bd=2,justify='right')
    txtParatha.grid(row=4,column=6,padx=10)

    lblRoti = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Chapati",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblRoti.grid(row=5,column=5,padx=10)
    txtRoti = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Roti , bd=2,justify='right')
    txtRoti.grid(row=5,column=6,padx=10)

    lblNaan = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Naan",fg="#7D1B7E",bd=10,anchor='w',justify='right')
    lblNaan.grid(row=6,column=5,padx=10)
    txtNaan = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Naan , bd=2,justify='right')
    txtNaan.grid(row=6,column=6,padx=10)


    lblMung_Halwa = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Mung
Halwa",fg="#7D1B7E",bd=10,anchor='w',justify='right')

    lblMung_Halwa.grid(row=7,column=5,padx=10)

    txtMung_Halwa = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Mung_Halwa ,
bd=2,justify='right')

    txtMung_Halwa.grid(row=7,column=6,padx=10)


    lblRabbdi = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Rabbdi",fg="#7D1B7E",bd=10,anchor='w',justify='right')

    lblRabbdi.grid(row=2,column=7,padx=10)

    txtRabbdi = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Rabbdi , bd=2,justify='right')

    txtRabbdi.grid(row=2,column=8,padx=10)


    lblJalebi = Label(Menu, font=( 'aria' ,12, 'bold'
),text="Jalebi",fg="#7D1B7E",bd=10,anchor='w',justify='right')

    lblJalebi.grid(row=3,column=7,padx=10)

    txtJalebi = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Jalebi , bd=2,justify='right')

    txtJalebi.grid(row=3,column=8,padx=10)


    lblGulab_Jamun = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Gulab
Jamun",fg="#7D1B7E",bd=10,anchor='w',justify='right')

    lblGulab_Jamun.grid(row=4,column=7,padx=10)

    txtGulab_Jamun = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Gulab_Jamun ,
bd=2,justify='right')

    txtGulab_Jamun.grid(row=4,column=8,padx=10)




    lblTotal = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Total
Cost",fg="Black",bd=10,anchor='w',justify='right')

    lblTotal.grid(row=7,column=7,padx=10)
    txtTotal = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=Total , bd=2,justify='right')
    txtTotal.grid(row=7,column=8,padx=10)

    lblCond = Label(Menu, font=( 'aria' ,10 ),text="*Including all
taxes",fg="Black",bd=10,anchor='w',justify='right')
    lblCond.grid(row=8,column=8)

    lblTotal = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Customer
Name:",fg="Black",bd=10,anchor='w',justify='right')
    lblTotal.grid(row=9,column=5,padx=10,pady=15)
    txtTotal = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=CusName , bd=2,justify='right')
    txtTotal.grid(row=9,column=6,padx=10, pady=15)

    lblTotal = Label(Menu, font=( 'aria' ,12, 'bold' ),text="Customer
Phone:",fg="Black",bd=10,anchor='w',justify='right')
    lblTotal.grid(row=9,column=7,padx=10,pady=15)
    txtTotal = Entry(Menu,font=('ariel' ,12,'bold'), textvariable=CusPhone , bd=2,justify='right')
    txtTotal.grid(row=9,column=8,padx=10,pady=15)



    # function to pop-up message box, when user tries to generate bill without specifing name and
phone number

    def customer():
        Name = CusName.get()
        Phone = CusPhone.get()

        if (Name == "" or Phone == ""):
            messagebox.showinfo("Alert", "Please fill Customer Name and Phone Number!")
        else:
            bill(Name, Phone)
```

```
# function to create database if it's not created earlier


def Database():
    global conn, cursor
    conn = sqlite3.connect('db_saleReports.db')
    cursor = conn.cursor()
    cursor.execute("CREATE TABLE IF NOT EXISTS `SalesRepo` (sales_id INTEGER
PRIMARY KEY AUTOINCREMENT NOT NULL, Name TEXT, Phone TEXT, Sales TEXT)")


# funtion to insert values into the database


def insert(cname, cphone, csales):
    Database()
    sql="INSERT INTO SalesRepo(Name, Phone, Sales) VALUES(?,?,?)"
    val=cname,cphone,csales
    cursor.execute(sql, val)
    conn.commit()


# function to display rate card on different window


def Rate():
    rateWin = Tk()
    rateWin.geometry("500x820+0+0")
    rateWin.title("Rate Card")


    RateCard = LabelFrame(rateWin,width = 400,height=800,relief=SUNKEN, text="Rate
Card", bg="white")
    RateCard.pack(fill= "both", expand="yes", padx=20, pady=10)


    lblinfo = Label(RateCard, font=('aria', 12, 'bold'), text="ITEM", fg="Black", bd=2, anchor
='w',justify='left')
```

```
        lblinfo.grid(row=0, column=0,padx=25, pady=10)


        lblinfo = Label(RateCard, font=('aria', 12, 'bold'), text="RATE", fg="Black",
anchor='w',justify='left')

        lblinfo.grid(row=0, column=3,padx=25, pady=10)


        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Chai", fg="#7D1B7E",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=1, column=0,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 20/-", fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=1, column=3,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Coffee", fg="#7D1B7E",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=2, column=0,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 40/-", fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=2, column=3,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Lassi", fg="#7D1B7E",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=3, column=0,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 60/-", fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=3, column=3,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Chas", fg="#7D1B7E",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=4, column=0,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 30/-", fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=4, column=3,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Samosa", fg="#7D1B7E",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=5, column=0,padx=25, pady=5)
```

```
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 20/-", fg="Black", anchor='w',
bg="white")
    lblinfo.grid(row=5, column=3,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Vada Pav", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
    lblinfo.grid(row=6, column=0,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 30/-", fg="Black", anchor='w',
bg="white")
    lblinfo.grid(row=6, column=3,padx=25, pady=5)


    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Channa Masala", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
    lblinfo.grid(row=7, column=0,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 100/-", fg="Black", anchor='w',
bg="white")
    lblinfo.grid(row=7, column=3,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Tikka Masala", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
    lblinfo.grid(row=8, column=0,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 120/-", fg="Black", anchor='w',
bg="white")
    lblinfo.grid(row=8, column=3,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Veg Korma", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
    lblinfo.grid(row=9, column=0,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 120/-", fg="Black", anchor='w',
bg="white")
    lblinfo.grid(row=9, column=3,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Matar Paneer", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
    lblinfo.grid(row=10, column=0,padx=25, pady=5)
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 140/-", fg="Black", anchor='w',
bg="white")
```

```
        lblinfo.grid(row=10, column=3,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Shahi Paneer", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
        lblinfo.grid(row=11, column=0,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 200/-", fg="Black", anchor='w',
bg="white")
        lblinfo.grid(row=11, column=3,padx=25, pady=5)


        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Veg Biryani", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
        lblinfo.grid(row=12, column=0,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 350/-", fg="Black", anchor='w',
bg="white")
        lblinfo.grid(row=12, column=3,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Daal Tadka", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
        lblinfo.grid(row=13, column=0,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 70/-", fg="Black", anchor='w',
bg="white")
        lblinfo.grid(row=13, column=3,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Daal Makhni", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
        lblinfo.grid(row=14, column=0,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 80/-", fg="Black", anchor='w',
bg="white")
        lblinfo.grid(row=14, column=3,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Paratha", fg="#7D1B7E",
anchor='w',justify='left', bg="white")
        lblinfo.grid(row=15, column=0,padx=25, pady=5)
        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 30/-", fg="Black", anchor='w',
bg="white")
        lblinfo.grid(row=15, column=3,padx=25, pady=5)
```

```
    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Chapati", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")

    lblinfo.grid(row=16, column=0,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 10/-", fg="Black", anchor='w', 
bg="white")

    lblinfo.grid(row=16, column=3,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Naan", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")

    lblinfo.grid(row=17, column=0,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 40/-", fg="Black", anchor='w', 
bg="white")

    lblinfo.grid(row=17, column=3,padx=25, pady=5)


    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Mung Halwa", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")

    lblinfo.grid(row=18, column=0,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 60/-", fg="Black", anchor='w', 
bg="white")

    lblinfo.grid(row=18, column=3,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Gulab Jamun", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")

    lblinfo.grid(row=19, column=0,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 50/-", fg="Black", anchor='w', 
bg="white")

    lblinfo.grid(row=19, column=3,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rabbdi", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")

    lblinfo.grid(row=20, column=0,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 50/-", fg="Black", anchor='w', 
bg="white")

    lblinfo.grid(row=20, column=3,padx=25, pady=5)

    lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Jalebi", fg="#7D1B7E", 
anchor='w',justify='left', bg="white")
```

```
        lblinfo.grid(row=21, column=0,padx=25, pady=5)

        lblinfo = Label(RateCard, font=('aria', 10, 'bold'), text="Rs. 40/-", fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=21, column=3,padx=25, pady=5)


        RateCard.mainloop()


    # funtion to generate bill and display it on diffent window


    def bill(nam, phn):


        billWin = Tk()

        billWin.geometry("600x520+550+50")

        billWin.title("Bill")


        billFrame = LabelFrame(billWin,width = 580,height=500,relief=SUNKEN, text="Bill",
bg="white")

        billFrame.pack(fill= "both", expand="yes", padx=20, pady=10)



        rtn = Cost()       # calling cost function to retrive values



        lbinfo = Label(billFrame, font=( 'aria' ,19, 'bold' ),text="Lovely
Restaurant",fg="Black",bd=10,anchor='w',bg="white")

        lbinfo.grid(row=0,column=0,padx=10, pady=15)

        lbinfo = Label(billFrame, font=( 'aria' ,10,
),text=localtime,fg="black",anchor=W,bg="white")

        lbinfo.grid(row=0,column=1,padx=10, pady=5)
        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="----------------------------------------
",fg="black",anchor=W,bg="white")

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

        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="----------------------------------------
",fg="black",anchor=W,bg="white")

        lbinfo.grid(row=2,column=1)


        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Order no.       - ", fg="Black",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=3, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text=rtn[4], fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=3, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Name            - ", fg="Black",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=4, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text=nam, fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=4, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Phone no.      - ", fg="Black",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=5, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text=phn, fg="Black", anchor='w',
bg="white")

        lblinfo.grid(row=5, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Cost of Meal   - ", fg="Black",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=6, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Rs. "+ rtn[0][1] + "/-", fg="Black",
anchor='w', bg="white")

        lblinfo.grid(row=6, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="GST / TAX       -", fg="Black",
anchor='w',justify='left', bg="white")
```

```
        lblinfo.grid(row=7, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Rs. "+ rtn[1][1] + "/-", fg="Black",
anchor='w', bg="white")

        lblinfo.grid(row=7, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Service Charges -", fg="Black",
anchor='w',justify='left', bg="white")

        lblinfo.grid(row=8, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Rs. "+ rtn[2][1] + "/-", fg="Black",
anchor='w', bg="white")

        lblinfo.grid(row=8, column=1,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Total Bill       - ", fg="Black",
anchor='w',justify='left', bg="white")


        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="------------------------------------------
",fg="black",anchor=W,bg="white")

        lbinfo.grid(row=9,column=0)

        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="------------------------------------------
",fg="black",anchor=W,bg="white")

        lbinfo.grid(row=9,column=1)


        lblinfo.grid(row=10, column=0,padx=25, pady=5)

        lblinfo = Label(billFrame, font=('aria', 10, 'bold'), text="Rs. "+ rtn[3][1] + "/-", fg="Black",
anchor='w', bg="white")

        lblinfo.grid(row=10, column=1,padx=25, pady=5)


        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="------------------------------------------
",fg="black",anchor=W,bg="white")

        lbinfo.grid(row=11,column=0)

        lbinfo = Label(billFrame, font=( 'aria' ,10, ),text="------------------------------------------
",fg="black",anchor=W,bg="white")

        lbinfo.grid(row=11,column=1)
        lbinfo = Label(billFrame, font=( 'aria' ,10, 'bold' ),text="This Bill is Auto 
Generated.",fg="Black",bd=10,anchor='w',bg="white")
        lbinfo.grid(row=13,column=1,padx=10, pady=15)



        insert(nam, phn, rtn[3][1])   # calling insert function to insert values into the data base
        billWin.mainloop()

    # function to generate sales report

    def report():

        repo = Tk()
        repo.title("Sales Report")
        screen_width = repo.winfo_screenwidth()
        screen_height = repo.winfo_screenheight()
        width = 800
        height = 400
        x = (screen_width/2) - (width/2)
        y = (screen_height/2) - (height/2)
        repo.geometry('%dx%d+%d+%d' % (width, height, x, y))
        repo.resizable(0, 0)



        # funtion to insert values into tree view

        def populateView():
            conn = sqlite3.connect('db_saleReports.db')
            cursor = conn.cursor()
            tree.delete(*tree.get_children())
            cursor.execute("SELECT * FROM `SalesRepo` ORDER BY `sales_id` ASC")
            fetch = cursor.fetchall()
            for data in fetch:
                tree.insert('', 'end', values=(data[0],data[1], data[2], data[3]))

        # creating frames and buttons


        Top = Frame(repo, width=700, height=50, bd=2, relief="raise")
        Top.pack(side=TOP)
        Button_Group=Frame(repo, width=700, height=50)
        Button_Group.pack(side=TOP)
        Buttons = Frame(Button_Group, width=200, height=50)
        Buttons.pack(side=LEFT)
        Buttons1 = Frame(Button_Group, width=500, height=50)
        Buttons1.pack(side=RIGHT)
        Body = Frame(repo, width=700, height=300, bd=2, relief="raise")
        Body.pack(side=BOTTOM)



        txt_title = Label(Top, width=300, font=('arial', 24), text = "Sales Report")
        txt_title.pack()



        btn_display = Button(Buttons,padx=19,pady=10,font=('ariel' ,12,'bold'), width=15,
text="Display All", command=populateView,bg="#7D0552",fg='white')
        btn_display.pack(side=LEFT,padx=20, pady=20)



        # creating scroll-bar on tree view
```

```
        scrollbary = Scrollbar(Body, orient=VERTICAL)

        scrollbarx = Scrollbar(Body, orient=HORIZONTAL)

        tree = ttk.Treeview(Body, columns=("sales_id","Name", "Phone", "Sales"),
selectmode="extended", height=300, yscrollcommand=scrollbary.set,
xscrollcommand=scrollbarx.set)

        scrollbary.config(command=tree.yview)

        scrollbary.pack(side=RIGHT, fill=Y)

        scrollbarx.config(command=tree.xview)

        scrollbarx.pack(side=BOTTOM, fill=X)


        # setting various headings and columns on tree view


        tree.heading('sales_id', text="Sr. no.", anchor=W)

        tree.heading('Name', text="Name", anchor=W)

        tree.heading('Phone', text="Phone", anchor=W)

        tree.heading('Sales', text="Sales", anchor=W)

        tree.column('#0', stretch=NO, minwidth=0, width=0)

        tree.column('#1', stretch=NO, minwidth=0, width=200)

        tree.column('#2', stretch=NO, minwidth=0, width=200)

        tree.column('#3', stretch=NO, minwidth=0, width=200)

        tree.column('#4', stretch=NO, minwidth=0, width=200)

        tree.pack(pady=5)


        repo.mainloop()


    # creating varoius control buttons to control the whole GUI interface


    btnReset=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="RESET", bg="#7D0552",command=reset)

    btnReset.grid(row=0, column=1, padx=20, pady=20)
    btnRate=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="RATE CARD", bg="#7D0552",command=Rate)
    btnRate.grid(row=0, column=2, padx=20, pady=20)


    btnTotal=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="TOTAL COST", bg="#7D0552",command=Cost)
    btnTotal.grid(row=0, column=3, padx=20, pady=20)


    btnBill=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="GENERATE BILL", bg="#7D0552",command=customer)
    btnBill.grid(row=0, column=4, padx=20, pady=20)


    btnsalesReport=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="SALES REPORT", bg="#7D0552",command=report)
    btnsalesReport.grid(row=0, column=5, padx=20, pady=20)


    btnExit=Button(Controls,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="EXIT", bg="#7D0552",command=exit)
    btnExit.grid(row=0, column=6,padx=20, pady=20)


    reset()
    root.mainloop()

# funtion to validate user login

def validateLogin(username, password):
    user= username.get()
    passwd = password.get()

    if (user == "root" and passwd == "123"):
        app()

    else:

        errorLabel = Label(logFrame, font=('aria', 10, 'bold'), text="Wrong User Name or 
Password", fg="red", bd=10, anchor='w', justify='left', bg='white').grid(row=4,column=1)
        username.set("")
        password.set("")


# creating login window

logWin = Tk()
logWin.geometry("600x520+550+50")
logWin.title("Login")

logFrame = LabelFrame(logWin,width = 580,height=500,relief=SUNKEN, bg="white", 
text="Login")
logFrame.pack(fill= "both", expand="yes", padx=20, pady=10)

WlcmLabel = Label(logFrame, font=( 'aria' ,15, 'bold' ), 
text="Welcome!",fg="Black",bd=10,anchor='w',justify='right', bg="white").grid(row=0, 
column=1,padx=28, pady=19)

usernameLabel = Label(logFrame, font=( 'aria' ,12, 'bold' ), text="User 
Name:",fg="Black",bd=10,anchor='w',justify='right', bg="white").grid(row=1, 
column=0,padx=28, pady=5)
username = StringVar()
usernameEntry = Entry(logFrame,font=('ariel' ,12,'bold'),bd=2,justify='right', 
textvariable=username).grid(row=1, column=1,padx=28, pady=5)
```

```
passwordLabel = Label(logFrame,text="Password:",font=( 'aria' ,12, 'bold'
),fg="Black",bd=10,anchor='w',justify='right', bg="white").grid(row=2, column=0,padx=28,
pady=10)

password = StringVar()

passwordEntry = Entry(logFrame, textvariable=password, show='*',font=('ariel' ,12,'bold'),
bd=2,justify='right').grid(row=2, column=1,padx=28, pady=10)


validateLogin = partial(validateLogin, username, password)



btnEnter=Button(logFrame,padx=19,pady=10, bd=2 ,fg="white",font=('ariel'
,12,'bold'),width=10, text="Enter", bg="#7D0552",command=validateLogin)

btnEnter.grid(row=3, column=1, padx=28, pady=20)


logWin.mainloop()
```