**NMAM INSTITUTE OF TECHNOLOGY**

(A Unit of Nitte Education Trust)

Autonomous – NBA and NAAC Accredited – AICTE Approved – ISO 9001:2008

Certified – Affiliated to VTU, Belagavi

**Nitte– 574 110, Karkala Tk., Udupi Dist., Karnataka**

**Department of Computer Science and Engineering**

****

**Python project report on**

**RESTAURANT BILLING SYSTEM**

**Submitted by:**

Roline Stapny Saldanha(4nm15cs138)

P Chaithanya Chimbalkar(4nm15cs109)

**Under the guidance of:**

Mr. Vijaya Murari T

Assistant Professor

Department of CSE

NMAMIT

**CERTIFICATE**

Certified that the project work carried out by Roline Stapny Saldanha (4NM15CS138) , P Chaithanya Chimbalkar(USN 4NM15CS109) bonafide students of NMAM Institute of Technology, Nitte in fulfilment for the Python Programming lab in Computer Science and Engineering during the academic year 2018-2019.

Signature of the Examiners: Signature of the Guide:

1. 1.

2. 2.

**ACKNOWLEDGEMENT**

This report acknowledges some guidance, some perspiration and a lot of inspiration. It is time now to acknowledge our deep sense of gratitude to all those who have extended their cooperation all among our study tenure of project work.

We take immense pleasure in thanking **Prof. Niranjan N. Chiplunkar** and our H.O.D **Dr. K.R. Uday Kumar Reddy** for having permitted us to carry on this project work.

We wish to place our deep sense of gratitude to our project guide **Mr. Vijaya Murari T** for his wholehearted guidance and support without which this project would not have been possible. Thank you sir for being there, ready to help us anytime.

We are grateful to our college and department for providing us proper academic environment to do our project. We also thank all non-teaching staff for their support. We are also greatly indebted to all our classmates for their valuable suggestions and corrections that lead to the success of our project.

Finally, yet importantly, we would like to express my heartfelt thanks to our beloved parents for their blessings, our friends and all those who were directly and indirectly involved with the project leading to its successful realization.

Roline Stapny Saldanha(4nm15cs138)

P Chaithanya Chimbalkar(4nm15cs109)

**CONTENTS**

* Problem statement
* Introduction
* Specifications
* Hardware
* Software
* Implementation
* Output
* Conclusion

**PROBLEM STATEMENT**

The problem statement is digitalizing the billing procedure in the restaurant by using Python and MySQL database. The user has to enter the quantity of given items purchased and the software calculates the total amount including the tax. The system counts the customer bills automatically and stores it in the database for further analysis. Furthermore we have provided an onscreen calculator for manual cross verification of the calculations.

**INTRODUCTION**

Management in restaurants is one of the most daunting jobs in the restaurant industry. The restaurant professional software program is a comprehensive restaurant management tool designed for food service management of all types. It is simple to learn and easy to use. This system processes transaction and stores the resulting data. Reports will be generated from these data which help the manager to make appropriate business decisions for the restaurant. For example, knowing the number of customers for the particular time interval, the manager can decide whether more waiters and chefs are required. Restaurant management systems are essential to the successful operation of most foodservice establishments because they allow the business to track transactions in real time.

In our project, the restaurant manager can automatically calculate the total amount. He need not worry about the tax. It is internally calculated. A calculator is also provided. A unique reference number is automatically generated for each and every bill. We will also store these bills in a database for further uses. It contains total number of customers visited the restaurant.

The main advantage is that we can use restaurant billing software to enhance our business by saving time and improving your organizational capability.

Various Advantages:

* It increases operational efficiency.
* It saves time.
* It helps restaurant manager to manage the restaurant more effectively and efficiently by computerizing meal ordering, billing.
* It increases security
* It avoids paper work
* It is simple to learn and easy to use
* Portable

By opting for this smart and intuitive restaurant accounting solution, we can manage our restaurant finances in a better and focus on our core operations.

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

Tkinter − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python.

wxPython − This is an open-source Python interface for wxWindows

JPython − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine

Wamp – Stands for “Windows, Apache, MySQl and PHP”. It is used to create database and can also execute queries in it.

**Introduction to Tkinter:**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

* Import the Tkinter module.
* Create the GUI application main window.
* Add one or more of the above-mentioned widgets to the GUI application.
* Enter the main event loop to take action against each event triggered by the user.

**Event-driven programming**

Anything that happens in a user interface is an event. We say that an event is fired whenever the user does something – for example, clicks on a button or types a keyboard shortcut. Some events could also be triggered by occurrences which are not controlled by the user – for example, a background task might complete, or a network connection might be established or lost.

Our application needs to monitor, or listen for, all the events that we find interesting, and respond to them in some way if they occur. To do this, we usually associate certain functions with particular events. We call a function which performs an action in response to an event an event handler – we bind handlers to events.Since OpenGL is a graphics API and not a platform of its own,it requires a language to operate in like C, C++, Java etc.

## **Tkinter basics**

## tkinter provides us with a variety of common GUI elements which we can use to build our interface – such as buttons, menus and various kinds of entry fields and display areas. We call these elements widgets. We are going to construct a tree of widgets for our GUI – each widget will have a parent widget, all the way up to the root window of our application. For example, a button or a text field needs to be inside some kind of containing window.

### **Widget classes**

There are many different widget classes built into tkinter:

* A Frame is a container widget which is placed inside a window, which can have its own border and background – it is used to group related widgets together in an application’s layout.
* Top level is a container widget which is displayed as a separate window.
* Canvas is a widget for drawing graphics. In advanced usage, it can also be used to create custom widgets – because we can draw anything we like inside it, and make it interactive.
* Text displays formatted text, which can be editable and can have embedded images.
* A Button usually maps directly onto a user action – when the user clicks on a button, something should happen.
* A Label is a simple widget which displays a short piece of text or an image, but usually isn’t interactive.
* Checkbutton, Radiobutton and Entry  are different kinds of input widgets – they allow the user to enter information into the program.

**SPECIFICATIONS**

**HARDWARE REQUIREMENTS**:

1. Processor: Pentium III 630MHZ or more

2. RAM: 128MB or more

3. Hard disc: 20GB or more

4. Monitor: Color monitor

5. Keyboard: Any keyboard

6. Mouse: Any mouse

7. Printer: Incase of printing reports

**SOFTWARE REQUIREMENTS**:

1. Operating System: Windows7, Windows 98, Windows 10, Windows XP

2. Language: Python Programming language

3. Database: MYSQL

**IMPLEMENTATION**

from tkinter import \*

import random

import time

import pymysql

import sys

import ctypes

from PIL import ImageTk

operator=""

#time

localtime=time.asctime(time.localtime(time.time()))

def nextpage():

fpage.destroy()

root=Tk()

root.title("Restaurant Billing Systems")

#database connection

try:

conn=pymysql.connect(host='localhost',user='root',password='',db='restaurant\_bills')

print("connection"+str(conn))

except:

print("connection refused")

sys.exit

cursor=conn.cursor()

print(cursor)

#message box

def Mbox(title, text, style):

return ctypes.windll.user32.MessageBoxW(0, text, title, style)

text\_Input=StringVar()

Tops=Frame(root,width=1600,height=50,bg="powder blue",relief=SUNKEN)

Tops.pack(side=TOP)

f1=Frame(root,width=800,height=700,relief=SUNKEN)

f1.pack(side=LEFT)

f2=Frame(root,width=300,height=700,relief=SUNKEN)

f2.pack(side=RIGHT)

#info

lblInfo=Label(Tops,font=('arial',50,'bold'),text="Restaurant Billing System",fg="Steel Blue",bd=10,anchor='w')

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

lblInfo=Label(Tops,font=('arial',20,'bold'),text=localtime,fg="Steel Blue",bd=10,anchor='w')

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

#calculator

def btnClick(numbers):

global operator

operator=operator+str(numbers)

text\_Input.set(operator)

def btnClearDisplay():

global operator

operator=""

text\_Input.set("")

def btnEqualsInput():

global operator

sumup=str(eval(operator))

text\_Input.set(sumup)

operator=""

def Ref():

x=random.randint(10908,500876)

randRef=str(x)

rand.set(randRef)

CoF=float(Fries.get())

CoD=float(Drinks.get())

CoFilet=float(Filet.get())

CoBurger=float(Burger.get())

CoChicBurger=float(Chicken\_Burger.get())

CoCheese\_burger=float(Cheese\_Burger.get())

CostofFries=CoF \* 0.99

CostofDrinks=CoD \* 1.00

CostofFilet=CoFilet \* 2.99

CostofBurger=CoBurger \* 2.87

CostChicBurger=CoChicBurger \* 2.89

CostCheese\_Burger=CoCheese\_burger \* 2.69

CostofMeal="Rs.",str('%.2f' % (CostofFries+ CostofDrinks+ CostofFilet+ CostofBurger+ CostChicBurger+ CostCheese\_Burger))

PayTax=((CostofFries + CostofDrinks + CostofFilet + CostofBurger + CostChicBurger+ CostCheese\_Burger)\*0.2)

TotalCost=(CostofFries+ CostofDrinks+ CostofFilet+ CostofBurger+ CostChicBurger+ CostCheese\_Burger)

Ser\_Charge=((CostofFries + CostofDrinks + CostofFilet + CostofBurger + CostChicBurger + CostCheese\_Burger)\*0.05)

Service="Rs.",str('%.2f' %(Ser\_Charge))

OverAllCost="Rs.",str('%.2f' % (PayTax+TotalCost+Ser\_Charge))

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

Service\_Charge.set(Service)

Cost.set(CostofMeal)

Tax.set(PaidTax)

SubTotal.set(CostofMeal)

Total.set(OverAllCost)

meal\_cost=CostofFries+ CostofDrinks+ CostofFilet+ CostofBurger+ CostChicBurger+ CostCheese\_Burger

overall\_cost=PayTax+TotalCost+Ser\_Charge

#insert the valus in the database

sql="INSERT INTO `bills`(`Reference`, `Large\_Fries`, `Burger\_Meal`, `Filet\_O\_Meal`, `Chicken\_Meal`, `Cheese`, `Drinks`,`Cost\_Of\_Meal`, `Service\_Charge`, `State\_Tax`, `Sub\_Total`, `Total\_Cost`) VALUES (%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d)" %(x,CoF,CoBurger,CoFilet,CoChicBurger,CoCheese\_burger,CoD,meal\_cost,Ser\_Charge,PayTax,TotalCost,overall\_cost)

try:

cursor.execute(sql)

conn.commit()

print("inserted")

Mbox('Database message', 'Succesfully Inserted into the database', 1)

except:

conn.rollback()

def qExit():

root.destroy()

conn.close()

def Reset():

rand.set("")

Fries.set("")

Burger.set("")

Filet.set("")

SubTotal.set("")

Total.set("")

Service\_Charge.set("")

Drinks.set("")

Tax.set("")

Cost.set("")

Chicken\_Burger.set("")

Cheese\_Burger.set("")

#restaurant info 1

rand=StringVar()

Fries=StringVar()

Burger=StringVar()

Filet=StringVar()

SubTotal=StringVar()

Total=StringVar()

Service\_Charge=StringVar()

Drinks=StringVar()

Tax=StringVar()

Cost=StringVar()

Chicken\_Burger=StringVar()

Cheese\_Burger=StringVar() txtDisplay=Entry(f2,font=('arial',20,'bold'),textvariable=text\_Input,bd=30,insertwidth=4,bg="powder blue",justify='right')

txtDisplay.grid(columnspan=4)

#buttons for calculator

btn7=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="7",bg="powder blue",command=lambda:btnClick(7)).grid(row=2,column=0)

btn8=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="8",bg="powder blue",command=lambda:btnClick(8)).grid(row=2,column=1)

btn9=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="9",bg="powder blue",command=lambda:btnClick(9)).grid(row=2,column=2)

Addition=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="+",bg="powder blue",command=lambda:btnClick("+")).grid(row=2,column=3)

btn4=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="4",bg="powder blue",command=lambda:btnClick(4)).grid(row=3,column=0)

btn5=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="5",bg="powder blue",command=lambda:btnClick(5)).grid(row=3,column=1)

btn6=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="6",bg="powder blue",command=lambda:btnClick(6)).grid(row=3,column=2)

Subtraction=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="-",bg="powder blue",command=lambda:btnClick("-")).grid(row=3,column=3)

btn1=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="1",bg="powder blue",command=lambda:btnClick(1)).grid(row=4,column=0)

btn2=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="2",bg="powder blue",command=lambda:btnClick(2)).grid(row=4,column=1)

btn3=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="3",bg="powder blue",command=lambda:btnClick(3)).grid(row=4,column=2)

Multiply=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="\*",bg="powder blue",command=lambda:btnClick("\*")).grid(row=4,column=3)

btn0=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="0",bg="powder blue",command=lambda:btnClick(0)).grid(row=5,column=0)

btnClear=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="C",bg="powder blue",command=btnClearDisplay).grid(row=5,column=1)

btnEquals=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="=",bg="powder blue",command=btnEqualsInput).grid(row=5,column=2)

Division=Button(f2,padx=16,pady=16,bd=8,fg="black",font=('arial',20,'bold'),text="/",bg="powder blue",command=lambda:btnClick("/")).grid(row=5,column=3)

lblReference=Label(f1,font=('arial',16,'bold'),text="Reference",bd=16,anchor='w')

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

txtReference=Entry(f1,font=('arial',16,'bold'),textvariable=rand,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

lblFries=Label(f1,font=('arial',16,'bold'),text="Large Fries",bd=16,anchor='w')

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

txtFries=Entry(f1,font=('arial',16,'bold'),textvariable=Fries,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

lblBurger=Label(f1,font=('arial',16,'bold'),text="Burger Meal",bd=16,anchor='w')

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

txtBurger=Entry(f1,font=('arial',16,'bold'),textvariable=Burger,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

lblFilet=Label(f1,font=('arial',16,'bold'),text="Filet\_O\_Meal",bd=16,anchor='w')

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

txtFilet=Entry(f1,font=('arial',16,'bold'),textvariable=Filet,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtFilet.grid(row=3,column=1)

lblChicken=Label(f1,font=('arial',16,'bold'),text="Chicken Meal",bd=16,anchor='w')

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

txtChicken=Entry(f1,font=('arial',16,'bold'),textvariable=Chicken\_Burger,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtChicken.grid(row=4,column=1)

lblCheese=Label(f1,font=('arial',16,'bold'),text="Cheese",bd=16,anchor='w')

lblCheese.grid(row=5,column=0)

txtCheese=Entry(f1,font=('arial',16,'bold'),textvariable=Cheese\_Burger,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtCheese.grid(row=5,column=1)

#restaurant info 2

lblDrinks=Label(f1,font=('arial',16,'bold'),text="Drinks",bd=16,anchor='w')

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

txtDrinks=Entry(f1,font=('arial',16,'bold'),textvariable=Drinks,bd=10,insertwidth=4,bg="powder blue",justify='right')

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

lblCost=Label(f1,font=('arial',16,'bold'),text="Cost of Meal",bd=16,anchor='w')

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

txtCost=Entry(f1,font=('arial',16,'bold'),textvariable=Cost,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtCost.grid(row=1,column=3)

lblService=Label(f1,font=('arial',16,'bold'),text="Service Charge",bd=16,anchor='w')

lblService.grid(row=2,column=2)

txtService=Entry(f1,font=('arial',16,'bold'),textvariable=Service\_Charge,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtService.grid(row=2,column=3)

lblStateTax=Label(f1,font=('arial',16,'bold'),text="State Tax",bd=16,anchor='w')

lblStateTax.grid(row=3,column=2)

txtStateTax=Entry(f1,font=('arial',16,'bold'),textvariable=Tax,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtStateTax.grid(row=3,column=3)

lblSubTotal=Label(f1,font=('arial',16,'bold'),text="Sub Total",bd=16,anchor='w')

lblSubTotal.grid(row=4,column=2)

txtSubTotal=Entry(f1,font=('arial',16,'bold'),textvariable=SubTotal,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtSubTotal.grid(row=4,column=3)

lblTotalCost=Label(f1,font=('arial',16,'bold'),text="Total Cost",bd=16,anchor='w')

lblTotalCost.grid(row=5,column=2)

txtTotalCost=Entry(f1,font=('arial',16,'bold'),textvariable=Total,bd=10,insertwidth=4,bg="powder blue",justify='right')

txtTotalCost.grid(row=5,column=3)

#Buttons

btnTotal=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Total",bg="powder blue",command=Ref).grid(row=7,column=1)

btnReset=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Reset",bg="powder blue",command=Reset).grid(row=7,column=2)

btnExit=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=10,text="Exit",bg="powder blue",command=qExit).grid(row=7,column=3)

root.mainloop()

fpage=Tk()

fpage.title("Restaurant Billing Systems")

fpage.configure(background="#000000")

bg\_image = ImageTk.PhotoImage(file = "C:/Users/User/Desktop/img.gif")

x = Label (image = bg\_image)

x.grid(row = 10, column = 10)

fpage.geometry("1200x1000")

btn=Button(fpage,padx=16,pady=16,bd=8,fg="black",font=('arial',15,'bold'),text="START",bg="pink",command=nextpage)

btn.grid(row=10, column=10)

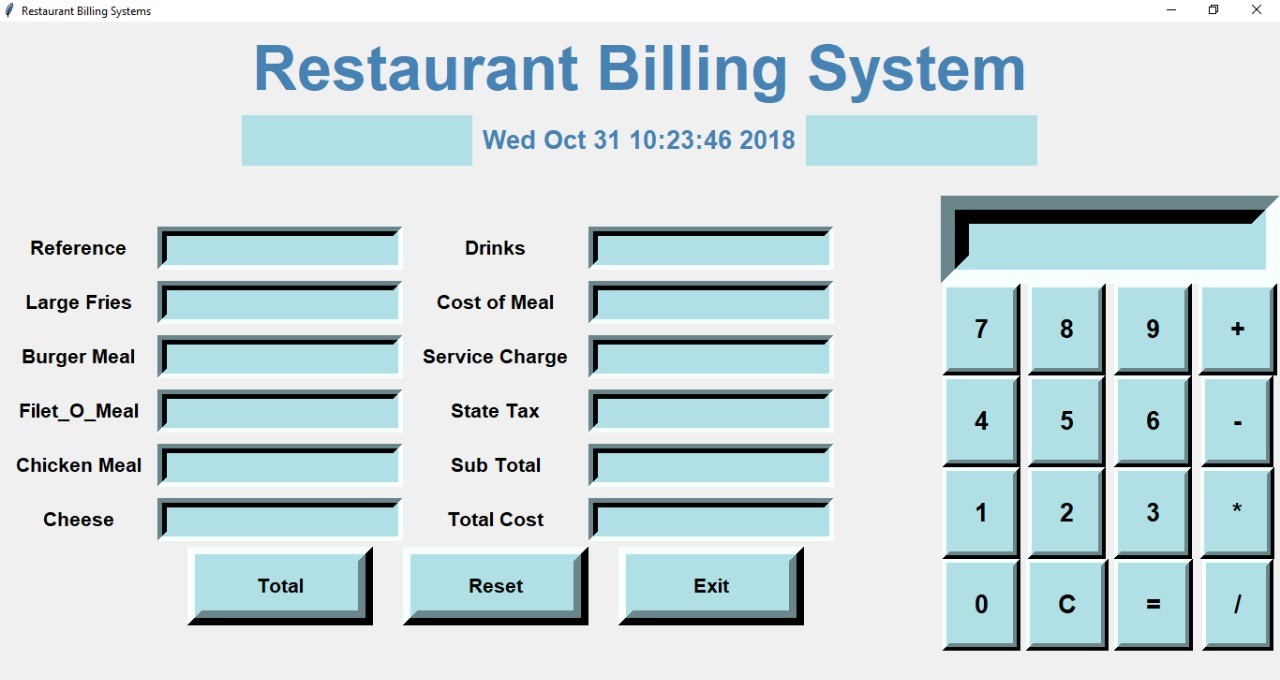
fpage.mainloop()

**OUTPUT**

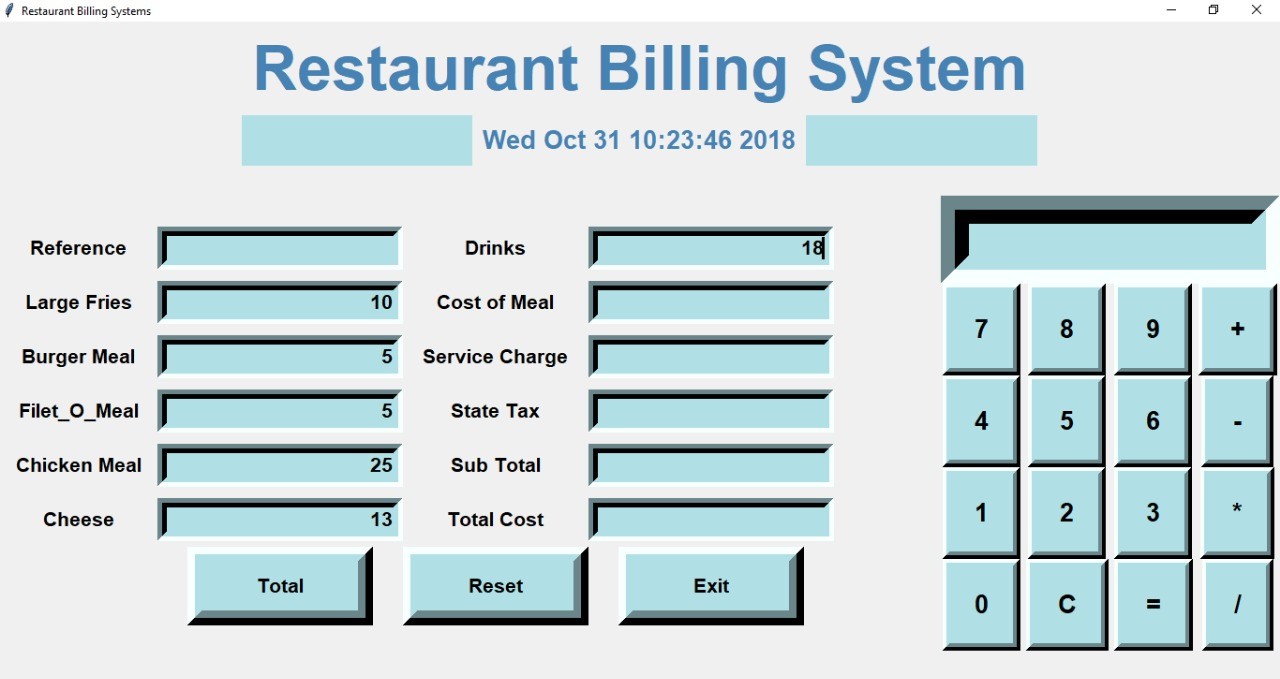
**First page**



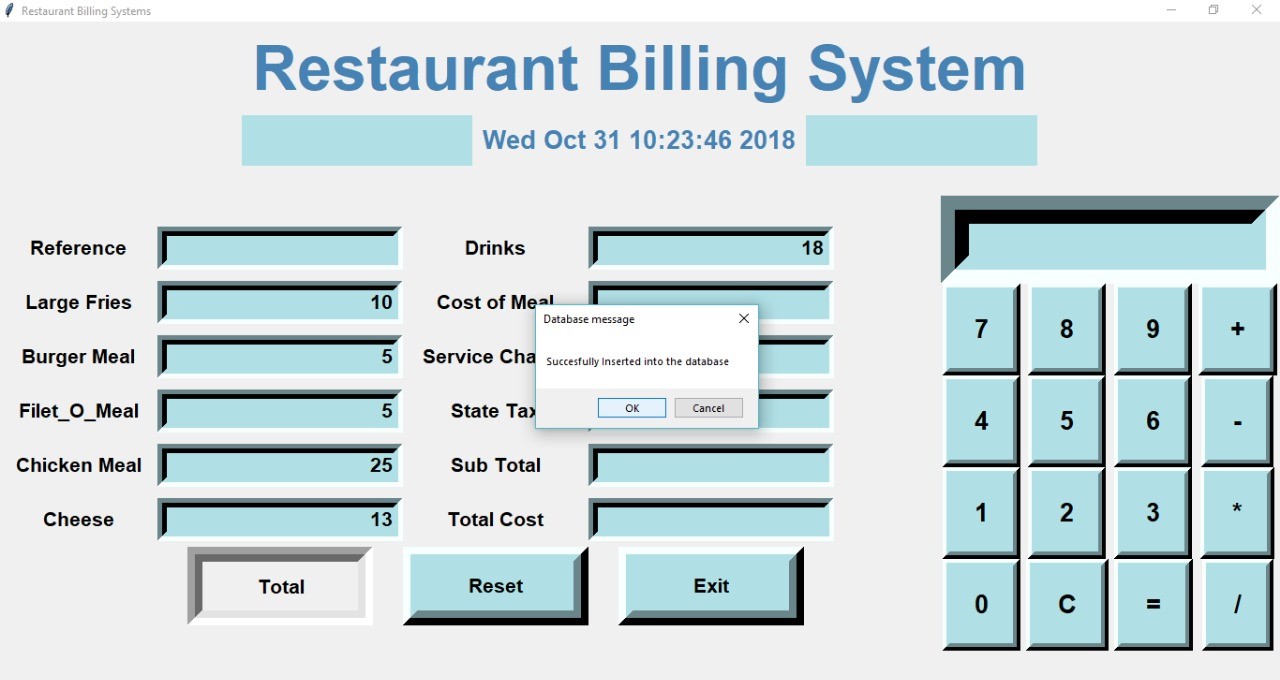
**Billing Page**

****

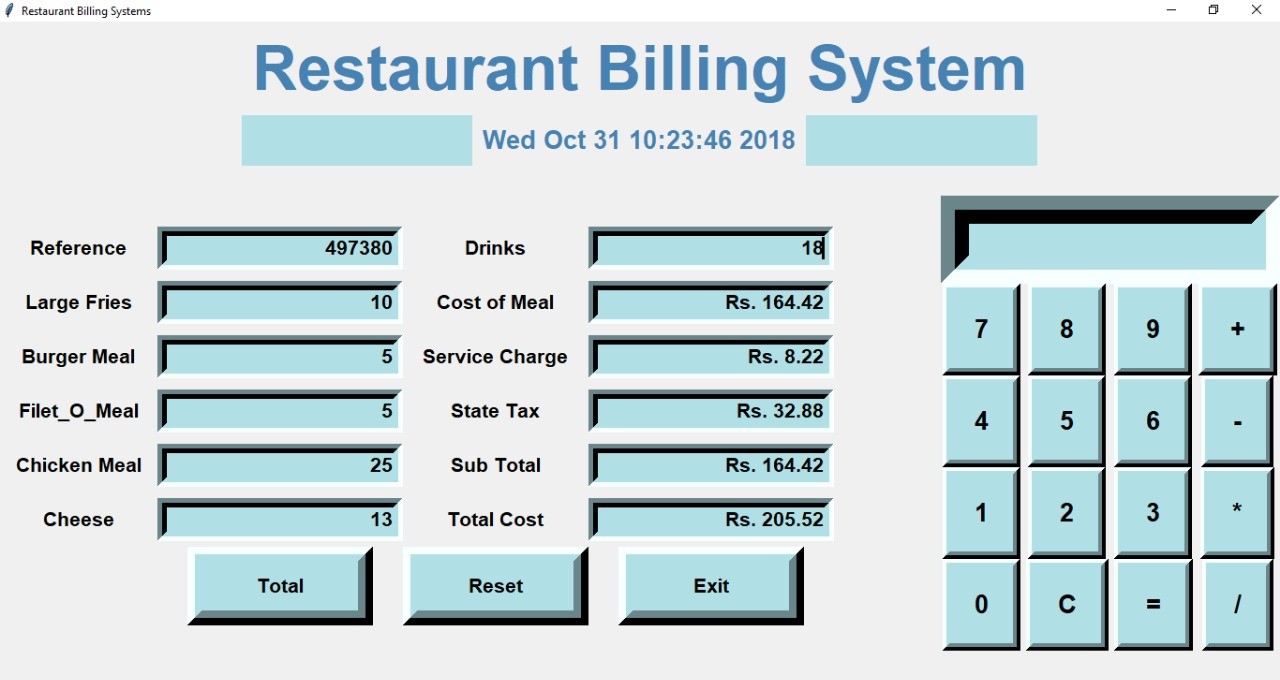
**Entering the items**

****

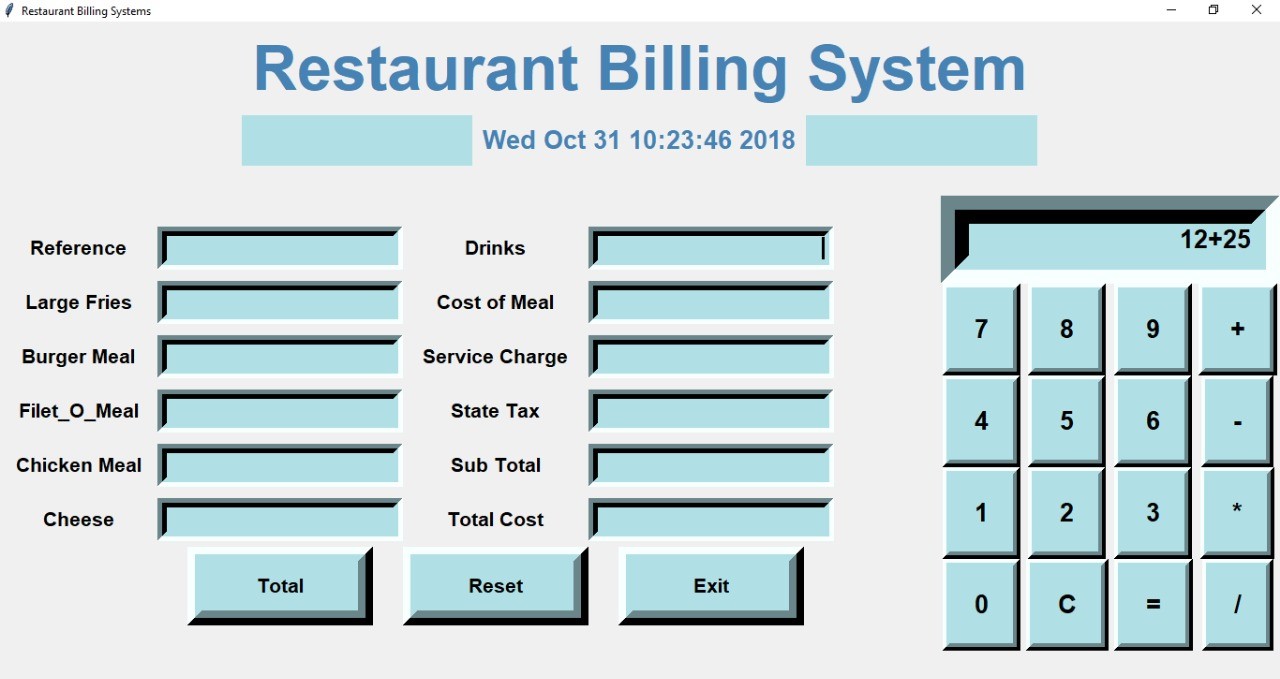
**Inserting into database**

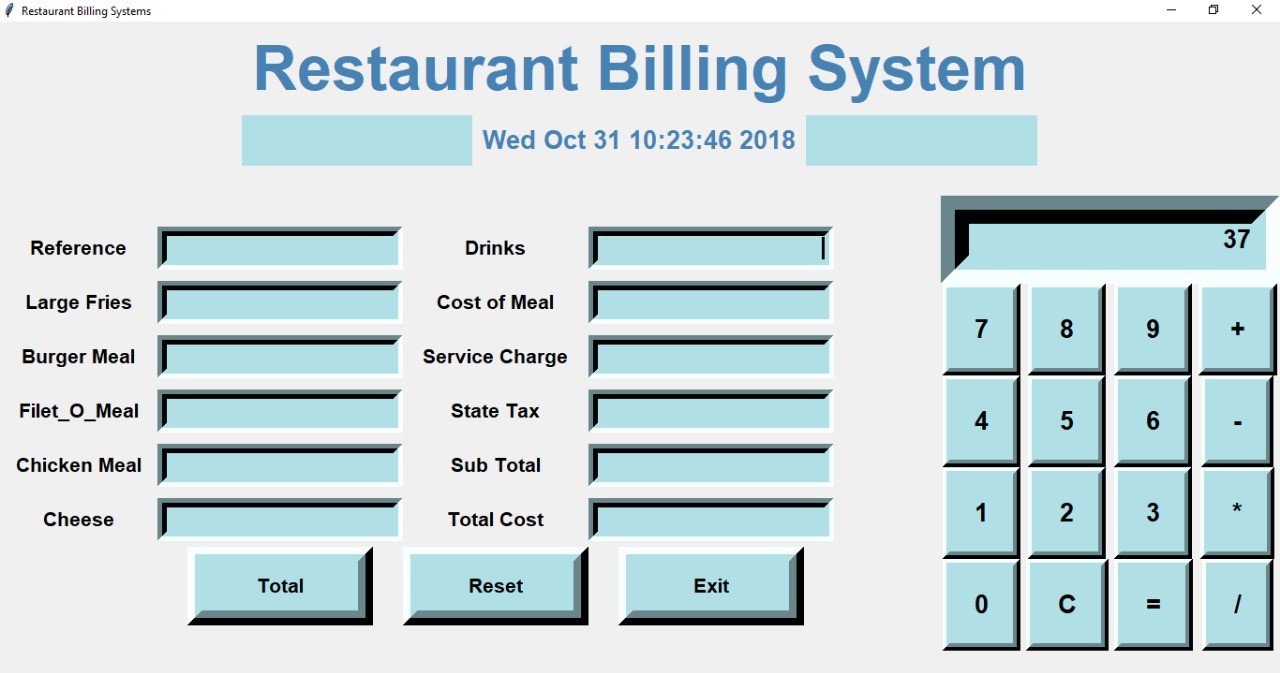
****

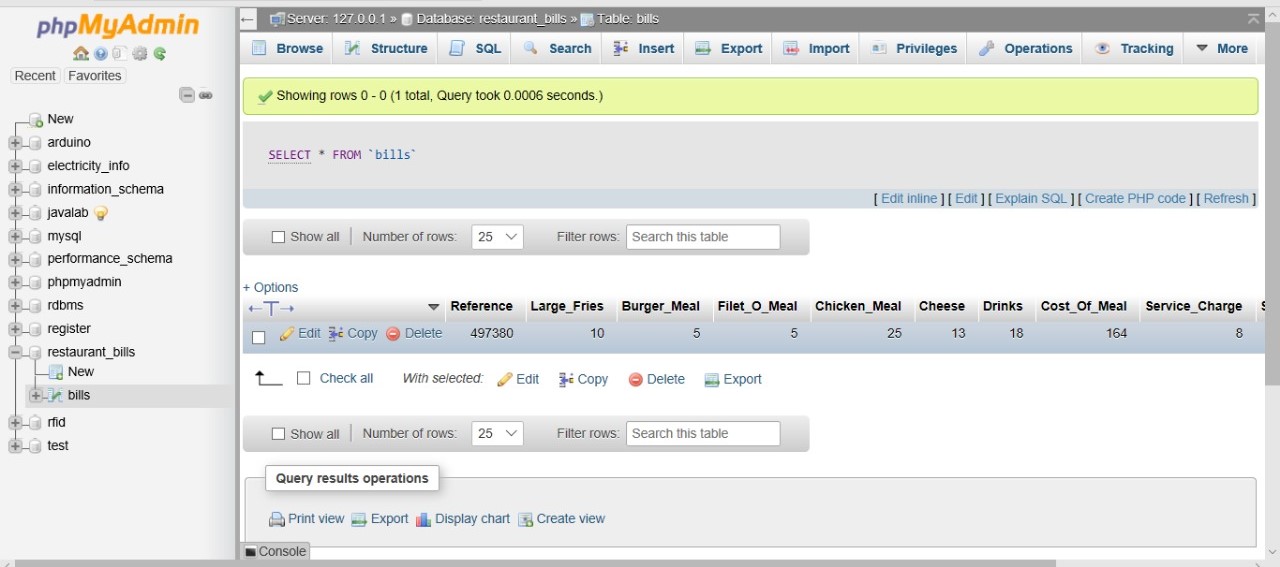
**Calculating total cost**

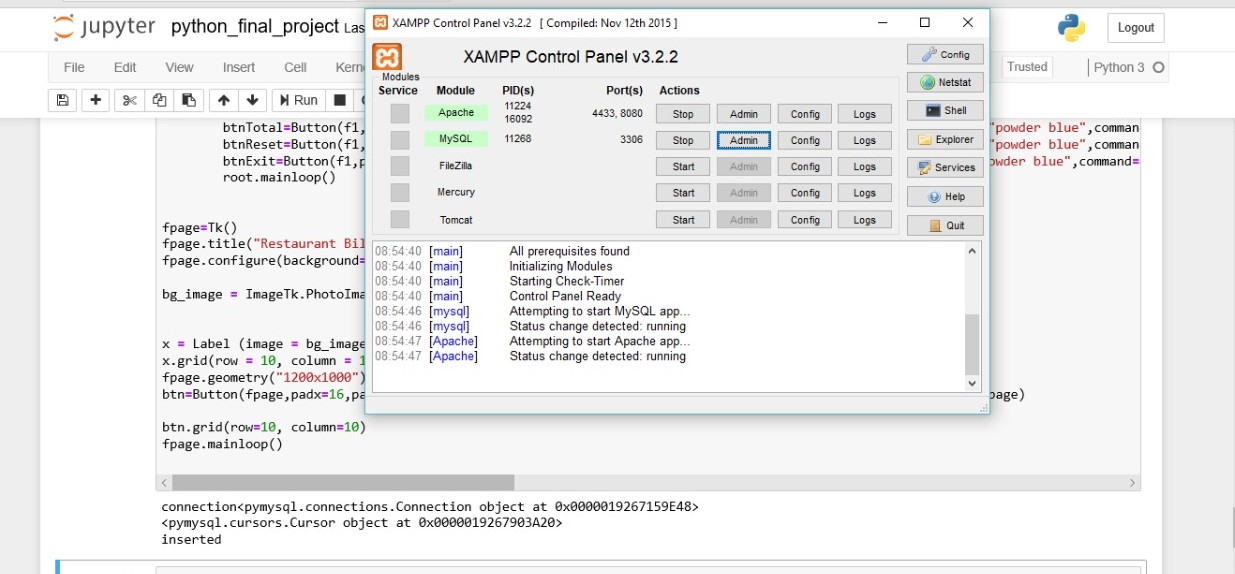
****

**Calculator**

****

****

****

****

**CONCLUSION**

Restaurant is a kind of business that serves people all over world with readymade food. Restaurant business is pretty competitive and in order to survive in this business one needs to work diligently. The main goals are gaining customer loyalty and main profits. Billing is by far the most crucial of all the accounting processes.

In our project entitled “Restaurant billing system” we have tried our best to fulfill all the billing requirements of a restaurant. The project being simple and flexible is running successfully. This helps the restaurant manager to manage the restaurant more effectively and efficiently by computerizing the billing system. With this system, billing becomes easier and systematic. This replaces the traditional manual method of using papers. Our system processes transaction and stores the data in the database which can be used for the further analysis.