

PROFESSIONAL TRAINING REPORT
AT
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology Degree in

Information Technology

By

BALAMURUGAN R
(Reg.No:38120019)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119

JANUARY 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119
www.sathyabamauniversity.ac.in**



SCHOOL OF COMPUTING

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **BALAMURUGAN R (Reg.No:38120019)** who carried out the project entitled **"RESTAURANT MANAGEMENT SYSTEM"** under my supervision August 2020 to January 2021.

Internal Guide Name

Dr.S.Revathy M.E., Ph.D.,

Head of the Department

Dr.R.SUBHASHINI ., M.E., Ph.D.,

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I BALAMURUGAN R (**Reg.no.38120019**) hereby declare that the Project Report entitled "RESTAURANT MANAGEMENT SYSTEM" done by me under the guidance of **Dr. S.REVATHY** and at **Udemy** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in INFORMATION TECHNOLOGY.

DATE:

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.SASIKALA., M.E.,Ph.D.**, Dean, School of Computing **Dr.R.SUBHASHINI M.E,Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.S.REVATHY M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of INFORMATION TECHNOLOGY** who were helpful in many ways for the completion of the project.



RETECH
SOLUTIONS PVT. LTD.
An ISO 9001:2015 certified company

INT: RE20S081

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **BALAMURUGAN R** student of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, BTECH/IT** has undergone the Internship Training through online in our concern entitled **PYTHON** from **12th Aug 2020 to 26th Aug 2020** in relevant departments related to their academic studies.

During the above period, the performance was good & we wish great success in all your future endeavours.

Technology drives the world. We are the people who are behind the technology.

Authorised Signature



No-31, 1st Floor, Alagesan Street, Tambaram West, Chennai-45.

Office : 044-42077204 | +91-8122266653

www.retechsolutions.in

ABSTRACT

Restaurant Bill Management System is developed using Python. The project file contains a python script (project.py). This is a simple GUI based project which is very easy to understand and use. While using this billing management system, you can easily calculate total bill of the customer. Moreover, the total bill is calculated including service charges and state tax. All you need to do is just fill up the blank boxes with item quantities and click on total button. The program will display your total bill. This simple program can be used in restaurants, cafe and food truck for calculating bills in a short period of time and it's not time consuming. It's easy to operate and understand by users. There is no any error and warning contents in the project. The design is so simple that user won't find it difficult to use and navigate. **Restaurant management system** helps you capture transactions and manage inventory with accuracy and generally run everyday processes more efficiently. RESTAURANT MANAGEMENT SYSTEM project fully developed in python language which is currently demanded in market using python GUI Tkinter. Python is a programming language that lets you work quickly and integrate systems more efficiently. Hence the project is developed proficiently to help restaurant. This system wake to provide service facility to restaurant and owners automate their billing operations. This system entirely reduces the unnecessary time waste inside the hotel as well as it reduces unnecessary calculation. Today's generation encourages high-tech services especially over the Internet.

TABLE OF CONTENT

CHAPTER NO:	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF ABBREVIATIONS	iv
	LIST OF FIGURES	iv
1	INTRODUCTION	1
	1.1. PYTHON LANGUAGE INTRODUCTION	1
	1.2. HISTORY OF PYTHON	2
	1.3. PYTHON FEATURES	3
2	AIM & SCOPE OF THE PRESENT INVESTIGATION	4
	2.1. AIM	4
	2.2. SCOPE OF THE PRESENT INVESTIGATION	4
	2.3. FUTURE WORK	4
3	EXPERIMENTAL OR MATERIALS AND METHODS ALGORITHMS USED	6
	3.1. MODULAR DESIGN	6
	3.2. PROGRAM	6
	3.3. TKINTER	7

4	RESULT AND DISCUSSION, PERFORMANCE ANALYSIS	9
	4.1. RESULT & DISCUSSION	9
	4.2. PERFORMANCE ANALYSIS	10
5	SUMMARY & CONCLUSION	11
	REFERENCES	12
	APPENDIX	13
	A.SCREENSHOT	13
	B.PSEUDOCODE	14
	C.SOURCE CODE	15

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
3.1.1	Library files	5
3.2	Program	6
4.1	Result	9
4.1.1	Python IDLE	10

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
IDLE	Integrated Development Environment
OpenCV	Open Source Computer Vision
SDK	Software Development Kit
Bg	Back ground
Fg	Fore ground
GUI	Graphical user interface

CHAPTER 1

INTRODUCTION

1.1 PYTHON LANGUAGE INTRODUCTION

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

1.2 HISTORY OF PYTHON

Python was developed by **Guido van Rossum** in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

1.2 PYTHON FEATURES

Python's features include

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

1.3 PYTHON GRAPHICAL USER INTERFACES (GUIs)

- **Tkinter** – Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.
- **wxPython** – This is an open-source Python interface for wxWindows <http://wxpython.org>.
- **JPython** – JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine <http://www.jython.org>.
- There are many other interfaces available, which you can find them on the net.

CHAPTER 2

AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 AIM

The aim of the project is to build an application in Python to maintain the orders and calculate the prices of orders at the restaurant. It allows a user to calculate the price, subtotal, taxes and total amount of the order by managing the quantities and prices of the items

2.2 SCOPE OF THE PRESENT INVESTIGATION

Restaurant management system is very much needed. It reduces a lot of hardwork and complexity. Nowadays restaurant are filled with people so it's very difficult to manage all the work manually. That's why these kinds of softwares are required.

2.3 FUTURE WORK

Restaurant management system is Desktop Application to restaurant management. Hence the project is developed proficiently to help restaurant. This system wake to provide service facility to restaurant and owners automate their billing operation

CHAPTER 3

EXPERIMENTAL OR MATERIALS AND METHODS

ALGORITHMS USED

3.1 MODULAR DESIGN

3.1.1 Modules involved in Program Development

3.1.1.1 Library Files



```
restaurant.py - C:\Users\AKHILL\Desktop\akshith\Restaurant mgmt PYTHON\restaurant.py (3.7.3)
File Edit Format Run Options Window Help
from tkinter import *
import random
import time
import datetime

root=Tk()
root.geometry("1600x800")
root.title("Restaurant Management System")

Tops=Frame(root, width=1600,relief=SUNKEN)
Tops.pack(side=TOP)

f1=Frame(root,width=800,height=700,relief=SUNKEN)
f1.pack(side=LEFT)

#=====
# TIME
#=====
localtime=time.asctime(time.localtime(time.time()))

lblInfo=Label(Tops,font=('ink free',50,'bold'),text="SRI ANNAPOORNA RESTAURANT ",fg="RED",bd=10,anchor='w')
lblInfo.grid(row=0,column=0)

lblInfo=Label(Tops,font=('ink free',20,'bold'),text=localtime,fg="BLUE",bd=10,anchor='w')
lblInfo.grid(row=1,column=0)

def Ref():
    x=random.randint(10908,500876)
    randomRef=str(x)
    rand.set(randomRef)

    if (samosa.get()==""):
        Cosamosa=0
    else:
        Cosamosa=float(samosa.get())

    if (Noodles.get()==""):
        CoNoodles=0
    else:
        CoNoodles=float(Noodles.get())

    if (Soup.get()==""):
        CoSoup=0
    else:
        CoSoup=0
```

Fig.3.1.1 Library files

Here we use the library files like add, insert, delete, root,font e.t.c.

The library file is the program in which the Dynamsoft SDK file is called upon to the program for executing the application software program intended to perform its task.

It involves commands to install the SDK file directly in to the command prompt of the system. It uses sys, os, NumPy which are libraries used in python for accessing the system access by the program.

3.2 PROGRAM

```
=====
#                                CALCULATION
#=====

Costofsamosa = Cosamosa * 10
CostofDrinks = CoD * 40
CostofNoodles = CoNoodles * 90
CostofSoup = CoSoup * 90
CostBurger = CoBurger * 120
CostSandwich = CoSandwich * 75

CostofMeal = "Rs", str('%.2f' % (Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSandwich))

PayTax = ((Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSandwich) * 0.12)
TotalCost = (Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSandwich)

Ser_Charge = ((Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSandwich)*0.06)
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)

def qExit():
    root.destroy()

def Reset():
    rand.set("")
    samosa.set("")
    Noodles.set("")
    Soup.set("")
    SubTotal.set("")
    Total.set("")
```

Fig.3.2 Program

This program is the source code of the application software. It consists of the library files such as OpenCV, NumPy, dbr, time and os from python repositories. This is which helps in getting the system access from the system for opening and accessing the camera module and also the information such as time and is used for scanning the barcodes and display its value as output from the program.

3.3 TKINTER

Here to run the program we need a header file called tkinter.

A good way to systematically check whether your Tkinter support is working is the following.

Enter an interactive Python interpreter in a shell on an X console.

Step 1 - can `_tkinter` be imported?

Try the following command at the Python prompt:

```
>>> import _tkinter # with underscore, and lowercase 't'
```

- If it works, go to step 2.
- If it fails with "No module named _tkinter", your Python configuration needs to be modified to include this module (which is an extension module implemented in C). Do ****not**** edit Modules/Setup (it is out of date). You may have to install Tcl and Tk (when using RPM, install the -devel RPMs as well) and/or edit the setup.py script to point to the right locations where Tcl/Tk is installed. If you install Tcl/Tk in the default locations, simply rerunning "make" should build the _tkinter extension.
- If it fails with an error from the dynamic linker, see above (for Unix, check for a header/library file mismatch; for Windows, check that the TCL/TK DLLs can be found).

Step 2 - can Tkinter be imported?

Try the correct command for your version at the Python prompt:

```
>>> import Tkinter # no underscore, uppercase 'T' for versions prior to V3.0
```

```
>>> import tkinter # no underscore, lowercase 't' for V3.0 and later
```

- If it works, go to step 3.
- If it fails with "No module named Tkinter", your Python configuration need to be changed to include the directory that contains Tkinter.py in its default module search path. You have probably forgotten to define TKPATH in the Modules/Setup file. A temporary workaround would be to find that directory and add it to your PYTHONPATH environment variable. It is the subdirectory named "lib-tk" of the Python library directory (when using Python 1.4 or before, it is named "tkinter").

Step 3 - does Tkinter work?

Try the correct command for your Python version at the Python prompt:

```
>>> Tkinter._test() # note underscore in _test and uppercase 'T' for versions prior to V3.0
```

```
>>> tkinter._test() # note underscore in _test and lowercase 'T' for V3.0 and later
```

- This should pop up a small window with two buttons. Clicking the "Quit" button makes it go away and the command return. If this works, you're all set. (When running this test on Windows, from Python run in a MS-DOS console, the new window somehow often pops up **under** the console window. This can also occur when using iTerm on Mac OS X. Move it aside or locate the Tk window in the Taskbar / Dock.)

CHAPTER 4

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

4.1 RESULTS AND DISCUSSION

This project involves giving inputs to the program so that it is processed by the program for giving the required result as programmed. Here the program succeeds in getting the input from the user.

The screenshot displays the 'Restaurant Management System' window for 'SRI ANNAPURNA RESTAURANT'. The date and time are 'Thu Aug 13 11:38:41 2020'. The interface is divided into two main columns for input and calculation. The left column lists items with their prices and quantities entered in text boxes. The right column shows the corresponding calculations. At the bottom, there are three buttons: 'Total', 'Reset', and 'Exit'. A Windows watermark is visible in the bottom right corner.

Item	Price	Quantity	Item	Price	Quantity
Reference		359327	Drinks	[Rs45]	4
samosa	[Rs10]	4	Cost of Meal		Rs 935.00
Noodles	[Rs90]	2	Service Charge [6%]		Rs 56.10
Soup	[Rs100]	1	GST [12%]		Rs 112.20
Burger	[Rs75]	2	Sub Total		Rs 935.00
Sandwich	[Rs60]	3	Total Cost		Rs 1103.30

Buttons: Total, Reset, Exit

Activate Windows
Go to PC settings to activate Windows.

Fig 4.1 Result

```

#-----RESTAURANT INFO 2-----#
lblDrinks= Label(f1, font=('arial', 16, 'bold'),text="Drinks\t[Rs45]",bd=16,anchor="w")
lblDrinks.grid(row=0, column=3)
txtDrinks=Entry(f1, font=('arial',16,'bold'),textvariable=Drinks,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtDrinks.grid(row=0,column=4)

lblCost= Label(f1, font=('arial', 16, 'bold'),text="Cost of Meal",bd=16,anchor="w")
lblCost.grid(row=1, column=3)
txtCost=Entry(f1, font=('arial',16,'bold'),textvariable=Cost,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtCost.grid(row=1,column=4)

lblService= Label(f1, font=('arial', 16, 'bold'),text="Service Charge [6%]",bd=16,anchor="w")
lblService.grid(row=2, column=3)
txtService=Entry(f1, font=('arial',16,'bold'),textvariable=Service_Charge,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtService.grid(row=2,column=4)

lblStateTax= Label(f1, font=('arial', 16, 'bold'),text="State Tax [12%]",bd=16,anchor="w")
lblStateTax.grid(row=3, column=3)
txtStateTax=Entry(f1, font=('arial',16,'bold'),textvariable=Tax,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtStateTax.grid(row=3,column=4)

lblSubTotal= Label(f1, font=('arial', 16, 'bold'),text="Sub Total",bd=16,anchor="w")
lblSubTotal.grid(row=4, column=3)
txtSubTotal=Entry(f1, font=('arial',16,'bold'),textvariable=SubTotal,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtSubTotal.grid(row=4,column=4)

lblTotalCost= Label(f1, font=('arial', 16, 'bold'),text="Total Cost",bd=16,anchor="w")
lblTotalCost.grid(row=5, column=3)
txtTotalCost=Entry(f1, font=('arial',16,'bold'),textvariable=Total,bd=10,insertwidth=4,bg="powder blue",justify='right')
txtTotalCost.grid(row=5,column=4)

#-----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=2)
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=3)
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=4)

root.mainloop()

```

Fig4.1.1 Python IDLE

The IDLE is the shell of the Python language where the compiled results and errors are show in it. Here the program has successfully analysed the given input and has given the result to it accordingly.

4.2 PERFORMANCE ANALYSIS

The GUI of this application is built using Python Tkinter library. 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. The tkinter components used in this application are: Button Label Entry.

CHAPTER 5

SUMMARY AND CONCLUSION

5.1 SUMMARY

As Python developers, we are constantly trying to address the problem of inconsistencies between the renderings by different Methods and Cascade classifiers. This necessitates either time consuming double / multiple coding, or coding for a single Detection which makes it harder, If not impossible. Detecting errors in accordance with the standards shortens Library development time and makes Detection easier to maintain. Debugging and troubleshooting become easier, because the code follows a standard. No longer do you have to worry about the coding and maintenance for several versions of code that are supposed to accomplish the same presentation. This it improves itself in further training.

5.2 CONCLUSION

This project has really been faithful and informative. It has made us learn and understand the many trivial concepts of Python Language. As we have used python Tkinter as a GUI it provides various controls, such as buttons, labels and text boxes to build a user friendly application. The fast growing use of internet confirms the good future and scope of the proposed project.

Finally it has taught us a valuable lifelong lesson about the improvements and working and interacting in a group.

REFERENCES

[1] <https://stackoverflow.com/>

[2] <https://www.programiz.com/python-programming/modules/random>

[3] <https://www.pythonforbeginners.com/random/how>

[4] <https://www.w3schools.com/>

[5] <https://www.geeksforgeeks.org>

APPENDIX

A.1 SCREEN SHOT OF CONSOLE

Restaurant Management System

SRI ANNAPURNA RESTAURANT

Thu Aug 13 11:38:41 2020

Reference		Drinks [Rs45]	
samosa [Rs10]		Cost of Meal	
Noodles [Rs90]		Service Charge [6%]	
Soup [Rs100]		GST [12%]	
Burger [Rs75]		Sub Total	
Sandwich [Rs60]		Total Cost	

Activate Windows
Go to PC settings to activate Windows.

A.2 SCREEN SHOT OF CONSOLE WITH RESULT

Restaurant Management System

SRI ANNAPURNA RESTAURANT

Thu Aug 13 11:38:41 2020

Reference	359327	Drinks [Rs45]	4
samosa [Rs10]	4	Cost of Meal	Rs 935.00
Noodles [Rs90]	2	Service Charge [6%]	Rs 56.10
Soup [Rs100]	1	GST [12%]	Rs 112.20
Burger [Rs75]	2	Sub Total	Rs 935.00
Sandwich [Rs60]	3	Total Cost	Rs 1103.30

Activate Windows
Go to PC settings to activate Windows.

B.PSEUDO CODE

1. Start the program.
2. Name of the restaurant is displayed.
3. Date and time is displayed.
4. Reference id is genrated randomly between 10908 and 500876.
5. Name of the food items with their costs are displayed.
6. Enter the quantity of food items.
7. Total:
 - Total cost of the food odered will be displayed.
 - Total cost includes cost of food items,service charge and state tax.
8. Reset:
 - All fields of items quantities are cleared.
9. Exit:
 - Application will be closed.
- 10.Stop the program.

C.SOURCE CODE

```
from tkinter import*
import random
import time
import datetime

root=Tk()
root.geometry("1600x8000")
root.title("Restaurant Management System")

Tops=Frame(root, width=1600,relief=SUNKEN)
Tops.pack(side=TOP)
f1=Frame(root,width=800,height=700,relief=SUNKEN)
f1.pack(side=LEFT)
localtime=time.asctime(time.localtime(time.time()))
lblInfo=Label(Tops,font=('ink free',50,'bold'),text="SRI ANNAPURNA RESTAURANT",fg="RED",bd=10,anchor='w')
lblInfo.grid(row=0,column=0)
lblInfo=Label(Tops,font=('ink free',20,'bold'),text=localtime,fg="BLUE",bd=10,anchor='w')
lblInfo.grid(row=1,column=0)
def Ref():
    x=random.randint(10908,500876)
    randomRef=str(x)
    rand.set(randomRef)
if (samosa.get()==""):
    Cosamosa=0
else:
    Cosamosa=float(samosa.get())
if (Noodles.get()==""):
    CoNoodles=0
```

```

else:
    CoNoodles=float(Noodles.get())
if (Soup.get()==""):
    CoSoup=0
else:
    CoSoup=float(Soup.get())
if (Burger.get()==""):
    CoBurger=0
else:
    CoBurger=float(Burger.get())
if (Sandwich.get()==""):
    CoSandwich=0
else:
    CoSandwich=float(Sandwich.get())
if (Drinks.get()==""):
    CoD=0
else:
    CoD=float(Drinks.get())
Costofsamosa =Cosamosa * 10
CostofDrinks=CoD * 40
CostofNoodles = CoNoodles* 90
CostofSoup = CoSoup * 90
CostBurger = CoBurger* 120
CostSandwich=CoSandwich * 75
CostofMeal= "Rs", str('%0.2f' %
(Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSandw
ich))
PayTax=((Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+C
ostSandwich) * 0.12)
TotalCost=(Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+
CostSandwich)
Ser_Charge=
((Costofsamosa+CostofDrinks+CostofNoodles+CostofSoup+CostBurger+CostSand
wich)*0.0
Service = "Rs", str ('%0.2f' % 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)

```

```

def qExit():
    root.destroy()

```

```

def Reset():
    rand.set("")
    samosa.set("")
    Noodles.set("")
    Soup.set("")
    SubTotal.set("")
    Total.set("")
    Service_Charge.set("")
    Drinks.set("")
    Tax.set("")
    Cost.set("")
    Burger.set("")
    Sandwich.set("")
rand = StringVar()
samosa=StringVar()
Noodles=StringVar()
Soup=StringVar()
SubTotal=StringVar()
Total=StringVar()
Service_Charge=StringVar()
Drinks=StringVar()
Tax=StringVar()
Cost=StringVar()
Burger=StringVar()

```

Sandwich=StringVar()

```
lblReference= Label(f1, font=('arial', 16,  
'bold'),text="Reference\t",bd=16,anchor="w")  
lblReference.grid(row=0, column=1)  
txtReference =Entry(f1,  
font=('arial',16,'bold'),textvariable=rand,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtReference.grid(row=0,column=2)
```

```
lblsamosa= Label(f1, font=('arial', 16,  
'bold'),text="samosa\t[Rs10]",bd=16,anchor="w")  
lblsamosa.grid(row=1, column=1)  
txtsamosa=Entry(f1,  
font=('arial',16,'bold'),textvariable=samosa,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtsamosa.grid(row=1,column=2)
```

```
lblNoodles= Label(f1, font=('arial', 16,  
'bold'),text="Noodles\t[Rs90]",bd=16,anchor="w")  
lblNoodles.grid(row=2, column=1)  
txtNoodles=Entry(f1,  
font=('arial',16,'bold'),textvariable=Noodles,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtNoodles.grid(row=2,column=2)
```

```
lblSoup= Label(f1, font=('arial', 16, 'bold'),text="Soup\t[Rs100]",bd=16,anchor="w")  
lblSoup.grid(row=3, column=1)  
txtSoup=Entry(f1,  
font=('arial',16,'bold'),textvariable=Soup,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtSoup.grid(row=3,column=2)
```

```
lblBurger= Label(f1, font=('arial', 16, 'bold'),text="Burger\t[Rs75]",bd=16,anchor="w")  
lblBurger.grid(row=4, column=1)
```

```
txtBurger=Entry(f1,  
font=('arial',16,'bold'),textvariable=Burger,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtBurger.grid(row=4,column=2)
```

```
lblSandwich= Label(f1, font=('arial', 16, 'bold'),text="Sandwich  
[Rs60]",bd=16,anchor="w")  
lblSandwich.grid(row=5, column=1)  
txtSandwich=Entry(f1,  
font=('arial',16,'bold'),textvariable=Sandwich,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtSandwich.grid(row=5,column=2)
```

```
lblDrinks= Label(f1, font=('arial', 16, 'bold'),text="Drinks\t[Rs45]",bd=16,anchor="w")  
lblDrinks.grid(row=0, column=3)  
txtDrinks=Entry(f1,  
font=('arial',16,'bold'),textvariable=Drinks,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtDrinks.grid(row=0,column=4)
```

```
lblCost= Label(f1, font=('arial', 16, 'bold'),text="Cost of Meal",bd=16,anchor="w")  
lblCost.grid(row=1, column=3)  
txtCost=Entry(f1,  
font=('arial',16,'bold'),textvariable=Cost,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtCost.grid(row=1,column=4)
```

```
lblService= Label(f1, font=('arial', 16, 'bold'),text="Service Charge  
[6%]",bd=16,anchor="w")  
lblService.grid(row=2, column=3)  
txtService=Entry(f1,  
font=('arial',16,'bold'),textvariable=Service_Charge,bd=10,insertwidth=4,bg="powder  
blue",justify='right')  
txtService.grid(row=2,column=4)
```

```

lblStateTax= Label(f1, font=('arial', 16, 'bold'),text="State Tax
[12%]",bd=16,anchor="w")
lblStateTax.grid(row=3, column=3)
txtStateTax=Entry(f1,
font=('arial',16,'bold'),textvariable=Tax,bd=10,insertwidth=4,bg="powder
blue",justify='right')
txtStateTax.grid(row=3,column=4)

```

```

lblSubTotal= Label(f1, font=('arial', 16, 'bold'),text="Sub Total",bd=16,anchor="w")
lblSubTotal.grid(row=4, column=3)
txtSubTotal=Entry(f1,
font=('arial',16,'bold'),textvariable=SubTotal,bd=10,insertwidth=4,bg="powder
blue",justify='right')
txtSubTotal.grid(row=4,column=4)

```

```

lblTotalCost= Label(f1, font=('arial', 16, 'bold'),text="Total Cost",bd=16,anchor="w")
lblTotalCost.grid(row=5, column=3)
txtTotalCost=Entry(f1,
font=('arial',16,'bold'),textvariable=Total,bd=10,insertwidth=4,bg="powder
blue",justify='right')
txtTotalCost.grid(row=5,column=4)

```

```

btnTotal=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=1
0,text="Total",bg="powder blue",command=Ref).grid(row=7,column=2)

```

```

btnReset=Button(f1,padx=16,pady=8,bd=16,fg="black",font=('arial',16,'bold'),width=1
0,text="Reset",bg="powder blue",command=Reset).grid(row=7,column=3)

```

```

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=4)
root.mainloop()

```