

SUMMER INTERNSHIP REPORT-2023

BANARAS LOCOMOTIVE WORKS

VARANASI - 221004



SUBMITTED TO:

PRINCIPAL

Technical Training Center

BLW, VARANASI

SUBMITTED BY:

Name : RITU PARNA BANERJEE

Registration No : 2023OF0011

Training Duration : 6 WEEK

Course : Bachelor of Technology

Branch Name : COMPUTER SCIENCE & ENGINEERING

Year : 3RD YEAR

Collage Name : ASHOKA INSTITUTE OF TECHNOLOGY & MANAGEMENT

CONTENTS

Topic

Certificate

Abstract

Acknowledgement

CHAPTER 1 PROJECT DESCRIPTION

- 1.1** Introduction
- 1.2** Motivation for Project
- 1.3** Language Description

CHAPTER 2 PLANNING

- 2.1** Flowchart
- 2.2** ER Diagram

CHAPTER 3 COMPONENT LIST & DESCRIPTION

- 3.1** Entire component list

CHAPTER 4 PROJECT CODE

- 4.1** Output

CHAPTER 5 ADVANTAGES & DISADVANTAGES

- 5.1** Advantages
- 5.2** Disadvantages
- 5.3** Future scope

CHAPTER 6 CONCLUSION AND REFERENCE

Banaras Locomotive Works (BLW)

VARANASI

CERTIFICATE OF TRAINING

This is to certify that **RITU PARNA BANERJEE** student of Bachelor of Technology in Computer Science & Engineering branch of ASHOKA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, VARANASI has successfully completed 6 weeks of Industrial training from 03/07/2023 to 11/08/2023.

During the training she worked on project name **Billing System** in "EDP" under the guidance of

"**Mr Rahul**".

Her overall performance during the training period was

Mr. Rahul

DATE:

ABSTRACT:

A **Billing System** can be very useful within a business environment. Instead of making bills manually or to sum up the total manually, it is very much time consuming and also may have some human errors like adding up the wrong total or adding wrong items into the bill.

When making a hand written bill the owner and customer both have to repeatedly check the total, items added, etc.

It also sometimes results into a Bad Impression towards the Restaurant or shop from a Customer.

Ideally, user should be able to generate bill without any mistakes and quickly, enabling them to fasten or improve their process.

To overcome this problem, we have come up with this project, that is, Billing System Using Python.

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would have been possible without the kind support and help of many individuals. It is indeed a great pleasure to work on this project and presenting this report to my department. This project had provided me a good exposure to the real world problem and also a solution to that.

I would like to pay my heartiest thanks to **TTC and EDP Department** who provided me such a wonderful opportunity to pursue my project on such an interesting topics. My heartfelt thanks goes to all other faculties who provided valuable suggestions and kind co-operation. I would like to thanks our project guide **Mr. Rahul Sir** for importing his valuable guidance and support. He has not only provided suggestions but also rectified my problems whenever I have faced any problems.

I would like to express my special gratitude and thanks to persons who rendered their assistance directly or indirectly. I would like to express my gratitude towards my parents & friends for their kind co-operation and encouragement which help me in this project.

CHAPTER 1

PROJECT DESCRIPTION

1.1 INTRODUCTION

The purpose of this report is to provide a comprehensive overview of the billing system. The billing system is a software application used by retail businesses to manage and streamline their billing processes.

It automates the calculation of sales transactions, generates invoices, tracks inventory, and manages customer data.

Billing System Using Python can be very useful

within a business environment. Instead of doing manual work for making up a bill at Retailers, which gets tiring and time consuming, you can generate a bill including tax and service charges in just few clicks. When making up a bill manually at a Retailers may contain some human errors like adding wrong items into the bill or summing up their total also may end up wrong, it also sometimes results into a Bad Impression towards the Retailers from a Customer.

Ideally, user should be able to generate bill without any mistakes and quickly, enabling them to fasten or improve their process. To overcome this problem, we have come up with this project, that is, Billing System Using Python.

The Billing System Using Python is very useful to

small business or restaurant or cafe or food truck owners.

This helps the owner to fasten the process which is bug free and easy to use. It also has a calculator to ease the use of the user. This project firstly has the menu and then adds up the selected items by customer and sums up the total of all items adds tax and service charges and displays total. To perform any other operation like division, multiplication, etc calculator is also.

1.2 Motivation for Project:

The scope of a billing system report typically includes an overview of the system, its features and functionalities, its implementation process, and its impact on the retail business.

The Retail industry is enlarging rapidly and Retailers owners are keen to improve every section of their business. Though much attention is paid to digitalizing the Retail management and the menu, but not many business owners realize the importance of applying digital billing software in the restaurant. The customers' experience at your restaurant includes the billing and payment experiences too. Billing software provides some exclusive features that ease up the restaurant services. It upgrades the billing process and uplift the customers' experience. It enables customers to pay bills more easily. The software can generate detailed bills that eliminate the need to calculate bills separately when the guests wish to know total GST amount. Apart from billing, the software enables you to organize a number of processes at the restaurant. It makes your system more effective and helps you provide faster and easy services to the customers. So many times, customers leave unhappy due to improper billing. When the crowd is vast in the restaurant, it might take you some time to generate manual bills that may leave your customers unsatisfied. This is where the automated billing system can be used. It generates digital bills automatically and allows customers to make quick payments.

1.3 Language Description

PYTHON:- 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.

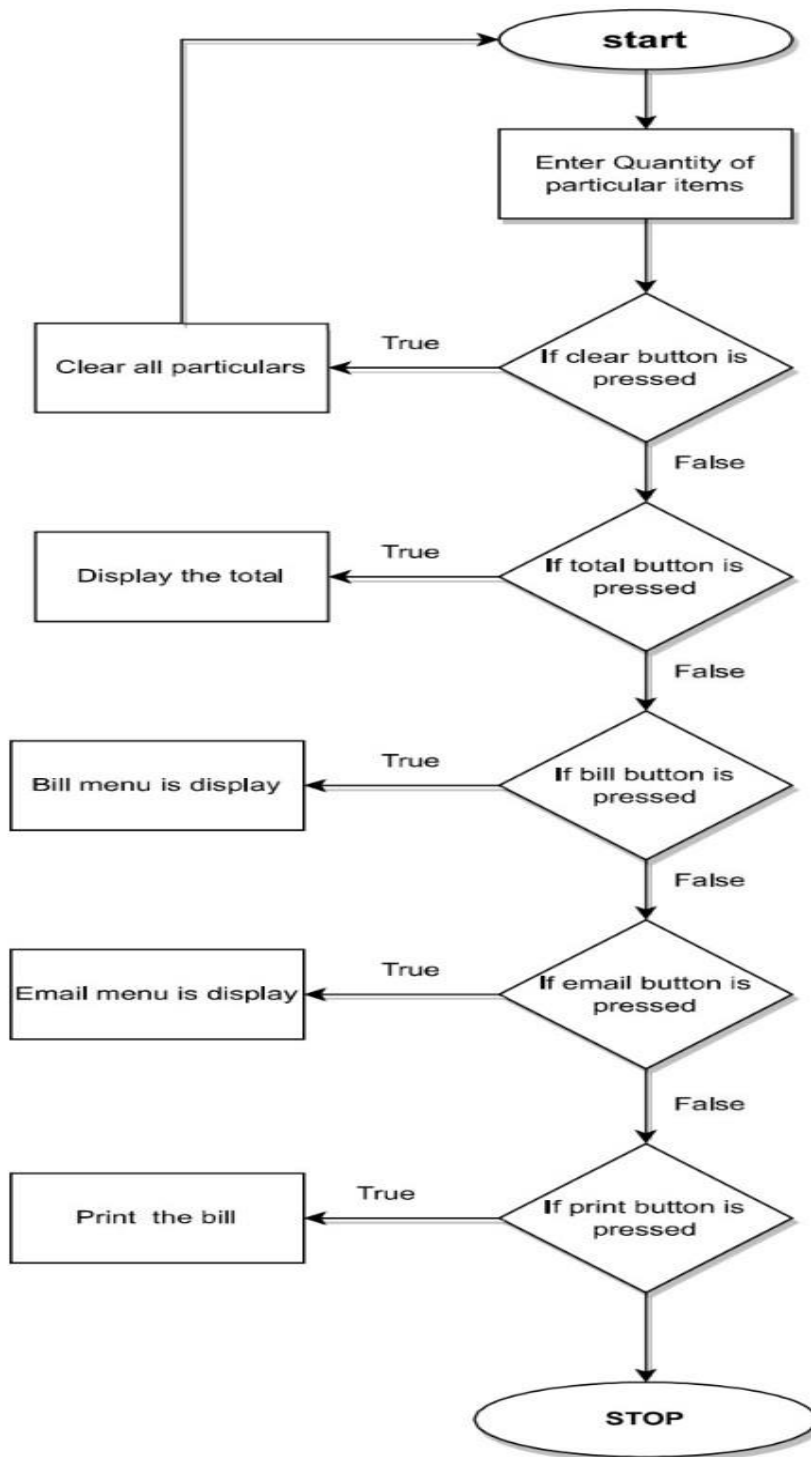
- Python is Interactive - You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- 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.

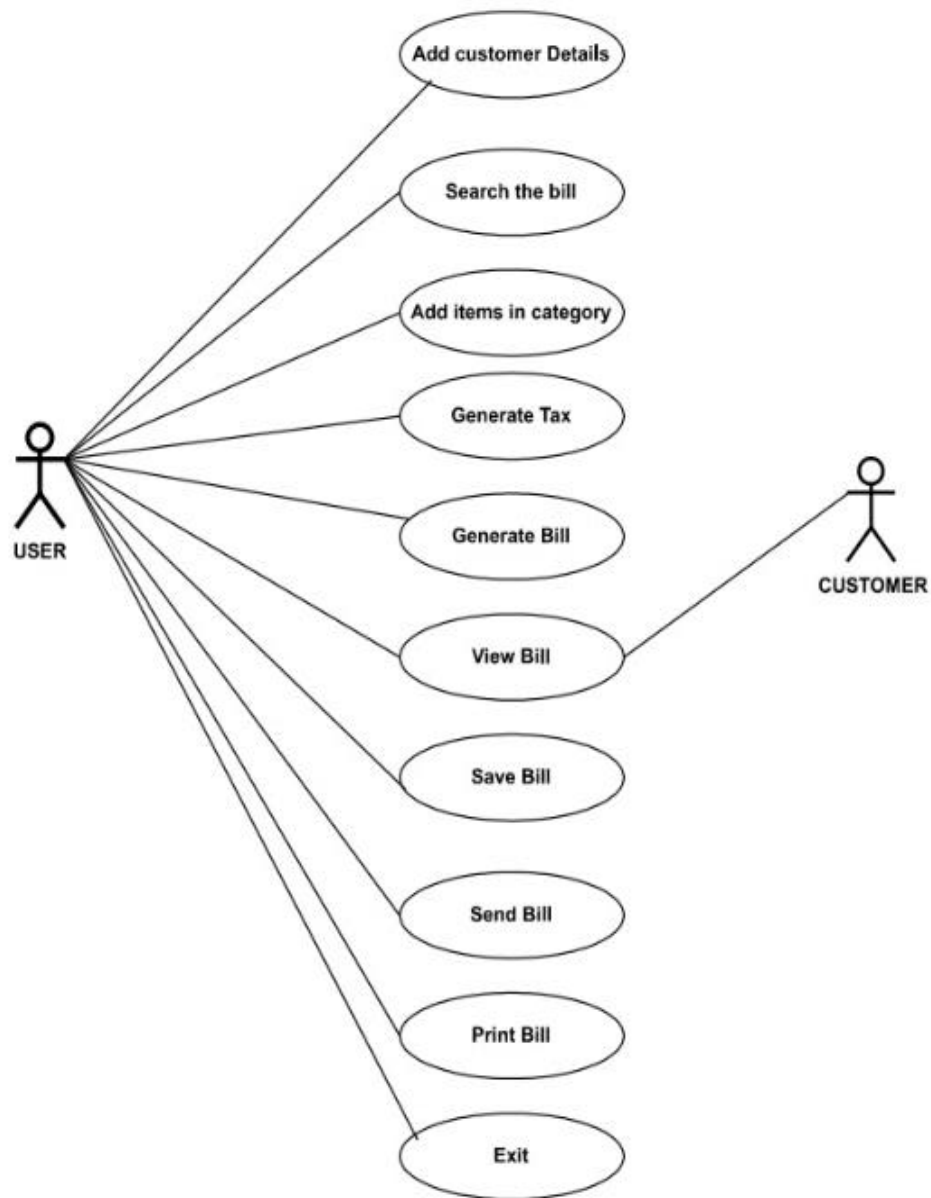
CHAPTER 2

PLANNING

2.1 Flowchart



2.2 ER Diagram



CHAPTER 3

COMPONENT LIST & DESCRIPTION

Hardware Requirements:

Processor - Intel i3 7th generation

RAM - 4 GB

System Type - 64-bit windows operating system

Software Requirements:

Technologies used - Python 3.6.

Python Tkinter GUI

Language used – Python

Python graphical user interfaces (GUIs) used in this Project

Tkinter

Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. 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. Tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

1. **pack() method:**It organizes the widgets in blocks before placing in the parent widget.
2. **grid() method:**It organizes the widgets in grid (table-like structure) before placing in the parent widget.
3. **place() method:**It organizes the widgets by placing them on specific positions directed by the programmer.



MODULES

tempfile module

Tempfile is a Python module used in a situation, where we need to read multiple files, change or access the data in the file, and gives output files based on the result of processed data. Each of the output files produced during the program execution was no longer needed after the program was done. In this case, a problem arose that many output files were created and this cluttered the file system with unwanted files that would require deleting every time the program ran.

OS Module in Python

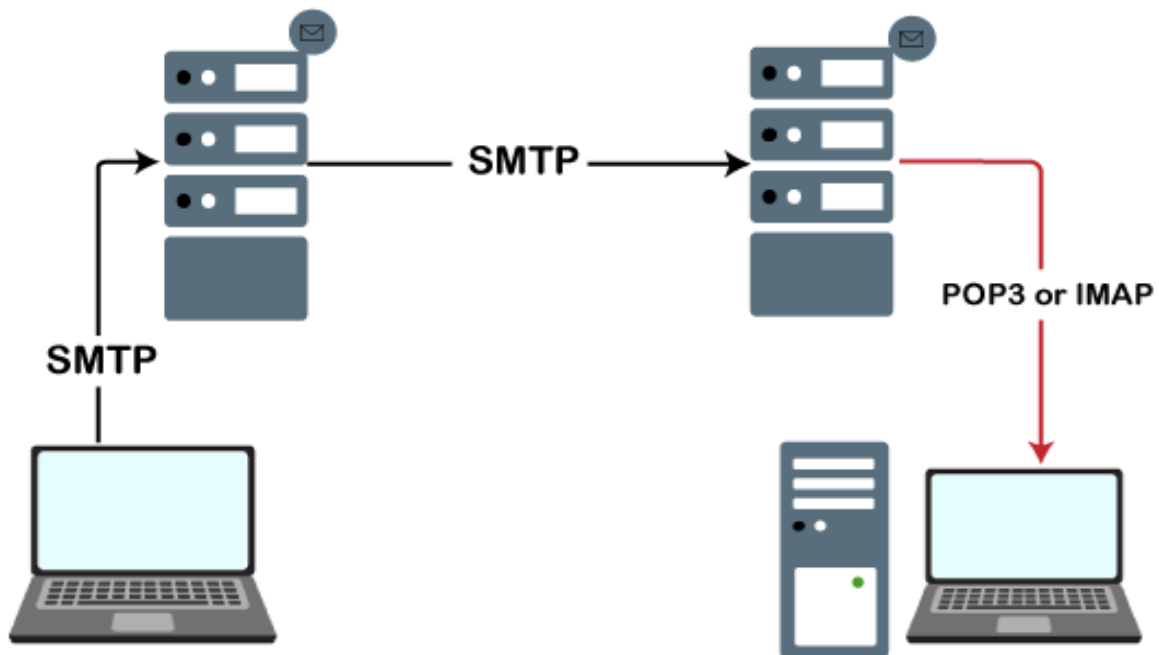
The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The `*os*` and `*os.path*` modules include many functions to interact with the file system.

Random Module

Python **Random module** is an in-built module of Python that is used to generate random numbers in [Python](#). These are pseudo-random numbers means they are not truly random. This module can be used to perform random actions such as generating random numbers, printing random a value for a list or string, etc.

Python Sending Email using SMTP

Simple Mail Transfer Protocol (SMTP) is used as a protocol to handle the email transfer using Python. It is used to route emails between email servers. It is an application layer protocol which allows to users to send mail to another. The receiver retrieves email using the protocols **POP(Post Office Protocol)** and **IMAP(Internet Message Access Protocol)**.



When the server listens for the TCP connection from a client, it initiates a connection on port 587.

Python provides a **smtplib** module, which defines an the SMTP client session object used to send emails to an internet machine. For this purpose, we have to import the **smtplib** module using the import statement.

CHAPTER 4

PROJECT CODE

```
from tkinter import *
from tkinter import messagebox
import random, os
import random, os, tempfile
import smtplib
#functionality part
# ptihykahluadct

def clear():

    bathsoapEntry.delete(0,END)
    facecreamEntry.delete(0,END)
    facewashEntry.delete(0,END)
    hairgelEntry.delete(0,END)
    hairsprayEntry.delete(0,END)
    bodylotionEntry.delete(0,END)
    daalEntry.delete(0,END)
    wheatEntry.delete(0,END)
    sugarEntry.delete(0,END)
    teaEntry.delete(0,END)
    riceEntry.delete(0,END)
    oilEntry.delete(0,END)
    pepsiEntry.delete(0,END)
    dewEntry.delete(0,END)
    frootiEntry.delete(0,END)
    spriteEntry.delete(0,END)
    cococolaEntry.delete(0,END)
    maazaEntry.delete(0,END)
    bathsoapEntry.insert(0,0)
    facecreamEntry.insert(0,0)
    facewashEntry.insert(0,0)
    hairgelEntry.insert(0,0)
    hairsprayEntry.insert(0,0)
    bodylotionEntry.insert(0,0)
    daalEntry.insert(0,0)
    wheatEntry.insert(0,0)
    sugarEntry.insert(0,0)
    teaEntry.insert(0,0)
    riceEntry.insert(0,0)
    oilEntry.insert(0,0)
```

```

pepsiEntry.insert(0,0)
dewEntry.insert(0,0)
frootiEntry.insert(0,0)
spriteEntry.insert(0,0)
cococolaEntry.insert(0,0)
maazaEntry.insert(0,0)
cosmeticstaxEntry.delete(0,END)
grocerytaxEntry.delete(0,END)
drinkstaxEntry.delete(0,END)
cosmeticspriceEntry.delete(0,END)
grocerypriceEntry.delete(0,END)
drinkspriceEntry.delete(0,END)
nameEntry.delete(0,END)
phoneEntry.delete(0,END)
billnumberEntry.delete(0,END)
textarea.delete(1.0,END)

def send_email():

def send_gmail():
    try:
        ob=smtplib.SMTP('smtp.gmail.com',587)
        ob.starttls()
        ob.login(senderEntry.get(),passwordEntry.get())
        message=email_textarea.get(1.0,END)
        # reciever_address=recieverEntry.get()
        ob.sendmail(senderEntry.get(),recieverEntry.get(),message)
        ob.quit()
        messagebox.showinfo('success', 'Bill is Successfully Send')
    except:
        messagebox.showerror('Error', 'Something went wrong, please try
again',parent=root1)
        root1.destroy()
    if textarea.get(1.0,END)=='\n':
        messagebox.showerror('Error','Bill is Empty')
    else:
        root1=Toplevel()
        root1.grab_set()
        root1.title('SEND GMAIL')
        root1.config(bg='gray20')
        root1.resizable(0,0)
        senderFrame=LabelFrame(root1,text='SENDER',font=('arial',16,'bold'),bd
=6,bg='gray20',fg='white')
        senderFrame.grid(row=0,column=0,padx=40,pady=20)

        senderLabel=Label(senderFrame,text="Sender's
Email",font=('arial',14,'bold'),bg='gray20',fg='white')
        senderLabel.grid(row=0,column=0,padx=10,pady=8)

```

```

        senderEntry=Entry(senderFrame,font=('arial',14,'bold'),bd=2,width=23,relief=RIDGE)
        senderEntry.grid(row=0,column=1,padx=10,pady=8)

        passwordLabel=Label(senderFrame,text="Password",font=('arial',14,'bold'),bg='gray20',fg='white')
        passwordLabel.grid(row=1,column=0,padx=10,pady=8)
        passwordEntry=Entry(senderFrame,font=('arial',14,'bold'),bd=2,width=23,relief=RIDGE,show='*')
        passwordEntry.grid(row=1,column=1,padx=10,pady=8)

recipientFrame=LabelFrame(root1,text='RECIPIENT',font=('arial',16,'bold'),bd=6,bg='gray20',fg='white')
        recipientFrame.grid(row=1,column=0,padx=40,pady=20)
        recieverLabel=Label(recipientFrame,text="Email Address",font=('arial',14,'bold'),bg='gray20',fg='white')
        recieverLabel.grid(row=0,column=0,padx=10,pady=8)

        recieverEntry=Entry(recipientFrame,font=('arial',14,'bold'),bd=2,width=23,relief=RIDGE)
        recieverEntry.grid(row=0,column=1,padx=10,pady=8)
        messageLabel=Label(recipientFrame,text="Message",font=('arial',14,'bold'),bg='gray20',fg='white')
        messageLabel.grid(row=1,column=0,padx=10,pady=8)
        email_textarea=Text(recipientFrame, font=('arial', 14, 'bold'),bd=2,relief=SUNKEN,width=42,height=11)
        email_textarea.grid(row=2,column=0,columnspan=2)
        email_textarea.delete(1.0,END)
        email_textarea.insert(END,textarea.get(1.0,END).replace('=','').replace('-',')).replace('\t\t\t','\t\t'))
        sendButton=Button(root1,text='SEND', font=('arial', 16, 'bold'),width=15,command=send_gmail)
        sendButton.grid(row=2,column=0,pady=20)
        root1.mainloop()

def print_bill():
    if textarea.get(1.0,END)=='\n':
        messagebox.showerror('Error','Bill is Empty')
    else:
        file=tempfile.mktemp('.txt')
        open(file,'w').write(textarea.get(1.0, END))
        os.startfile(file,'print')
def search_bill():

```

```

for i in os.listdir('bills/'):

    if i.split('.')[0] == billnumberEntry.get():
        f = open(f'bills/{i}', 'r')
        textarea.delete('1.0', END)
        for data in f:
            textarea.insert(END, data)
        f.close()
        break
    else:
        messagebox.showerror('Error', 'Invalid Bill Number')
if not os.path.exists('bills'):

    os.mkdir('bills')

def save_bill():

    global billnumber
    result=messagebox.askyesno('Confirm', 'Do you Want to save the bill?')
    if result:
        bill_content=textarea.get(1.0,END)
        file=open(f'bills/{billnumber} .txt', 'w')
        # .pdf is used for your choise
        file.write(bill_content)
        file.close()
        messagebox.showinfo('Success', f' Bill number {billnumber} is saved
successfully')
        billnumber = random.randint(500,15000)

billnumber=random.randint(500,15000)

def bill_area():
    if nameEntry.get()==' ' or phoneEntry.get()==' ':
        messagebox.showerror('Error', 'Customer Details Are Required')
    elif cosmeticspriceEntry.get()==' ' or grocerypriceEntry.get()==' ' or
drinkspriceEntry.get()==' ':
        messagebox.showerror('Error', 'No Product are Selected')
    elif cosmeticspriceEntry.get()=='0 Rs' and grocerypriceEntry.get()=='0 Rs'
and drinkspriceEntry.get()=='0 Rs':
        messagebox.showerror('Error', 'No Product are Selected')
    else:
        textarea.delete(1.0,END)
        textarea.insert(END, '\t\t**WELCOME CUSTOMER**\n')
        textarea.insert(END, f'\nBill Number:{billnumber}\n')
        textarea.insert(END, f'\nCustomer Name:{nameEntry.get()}\n')
        textarea.insert(END, f'\nCustomer Phone Number:{phoneEntry.get()}\n')
        textarea.insert(END, '\n=====')
        =====')

```



```

        textarea.insert(END, f'\nProduct\t\t\tQuantity\t\t\tPrice')
        textarea.insert(END, '\n=====')
    =====')
    if bathsoapEntry.get() != '0':
        textarea.insert(END, f'\nBath Soap
\t\t\t{bathsoapEntry.get()}\t\t\t{soapprice} Rs')
    if facecreamEntry.get() != '0':
        textarea.insert(END, f'\nFace Cream
\t\t\t{facecreamEntry.get()}\t\t\t{facecreamprice} Rs')
    if facewashEntry.get() != '0':
        textarea.insert(END, f'\nFace Wash
\t\t\t{facewashEntry.get()}\t\t\t{facewashprice} Rs')
    if hairsprayEntry.get() != '0':
        textarea.insert(END, f'\nHair Spray
\t\t\t{hairsprayEntry.get()}\t\t\t{hairsprayprice} Rs')
    if hairgelEntry.get() != '0':
        textarea.insert(END, f'\nHair Gel
\t\t\t{hairgelEntry.get()}\t\t\t{hairgelprice} Rs')
    if bodylotionEntry.get() != '0':
        textarea.insert(END, f'\nBody Lotion
\t\t\t{bodylotionEntry.get()}\t\t\t{bodylotionprice} Rs')

    if riceEntry.get() != '0':
        textarea.insert(END, f'\nRice
\t\t\t{riceEntry.get()}\t\t\t{riceprice} Rs')

    if oilEntry.get() != '0':
        textarea.insert(END, f'\nOil \t\t\t{oilEntry.get()}\t\t\t{oilprice
} Rs')

    if daalEntry.get() != '0':
        textarea.insert(END, f'\nDaal\t\t\t{daalEntry.get()}\t\t\t{daalprice} Rs')

    if wheatEntry.get() != '0':
        textarea.insert(END, f'\nWheat\t\t\t{wheatEntry.get()}\t\t\t{wheatprice} Rs')

    if sugarEntry.get() != '0':
        textarea.insert(END, f'\nSugar\t\t\t{sugarEntry.get()}\t\t\t{sugarp
rice} Rs')

    if teaEntry.get() != '0':
        textarea.insert(END, f'\nTea \t\t\t{teaEntry.get()}\t\t\t{teaprice}
Rs')

    if maazaEntry.get() != '0':

```

```

        textarea.insert(END,f'\nMaaza
\t\t\t{maazaEntry.get()}\t\t\t{maazaprice} Rs')

        if pepsiEntry.get()!='0':
            textarea.insert(END,f'\nPepsi
\t\t\t{pepsiEntry.get()}\t\t\t{pepsiprice} Rs')

        if spriteEntry.get()!='0':
            textarea.insert(END,f'\nSprite
\t\t\t{spriteEntry.get()}\t\t\t{spriteprice} Rs')

        if dewEntry.get()!='0':
            textarea.insert(END,f'\nDew \t\t\t{dewEntry.get()}\t\t\t{dewprice}
Rs')

        if frootiEntry.get()!='0':
            textarea.insert(END,f'\nFrooti
\t\t\t{frootiEntry.get()}\t\t\t{frootiprice} Rs')

        if cococolaEntry.get()!='0':
            textarea.insert(END,f'\nCoco Cola
\t\t\t{cococolaEntry.get()}\t\t\t{cococolaprice} Rs')

        textarea.insert(END,'\n-----
-----')

        if cosmeticstaxEntry.get()!='0.0Rs':
            textarea.insert(END,f'\nCosmetic
Tax\t\t\t\t{cosmeticstaxEntry.get()}\t\t\t\t')

        if grocerytaxEntry.get()!='0.0Rs':
            textarea.insert(END,f'\nGrocery
Tax\t\t\t\t\t{grocerytaxEntry.get()}\t\t\t\t\t')

        if drinkstaxEntry.get()!='0.0Rs':
            textarea.insert(END,f'\nDrink Tax\t\t\t\t\t{drinkstaxEntry.get()}\t\t\t\t\t')
            textarea.insert(END,'\n-----
-----')

        textarea.insert(END,f'\n\nTotal Bill \t\t\t\t\t {totalbill}\t\t\t\t\t')

        textarea.insert(END,'\n-----
-----')

        textarea.insert(END,'\t\t\t\t\t**THANKYOU VISIT AGAIN**\n')

```

```
save_bill()
```

```
def total():
    global
    soapprice, facecreamprice, facewashprice, hairsprayprice, hairgelprice, bodylotionp
    rice
    global riceprice, oilprice, daalprice, wheatprice, sugarprice, teaprice
    global
    maazaprice, pepsiprice, spriteprice, dewprice, frootiprice, cococolaprice
    global totalbill
    #cosmetics price calculation
    soapprice=int(bathsoapEntry.get())*20
    facecreamprice=int(facecreamEntry.get())*50
    facewashprice=int(facewashEntry.get())*100
    hairsprayprice=int(hairsprayEntry.get())*150
    hairgelprice=int(hairgelEntry.get())*80
    bodylotionprice=int(bodylotionEntry.get())*60

    totalcosmeticprice=soapprice+facewashprice+facecreamprice+hairgelprice+hai
    rsprayprice+bodylotionprice
    cosmeticspriceEntry.delete(0,END)
    cosmeticspriceEntry.insert(0,f'{totalcosmeticprice} Rs')
    cosmeticstax=totalcosmeticprice*0.12
    cosmeticstaxEntry.delete(0,END)
    cosmeticstaxEntry.insert(0,str(cosmeticstax)+ 'Rs')

    #grocery price calculation

    riceprice=int(riceEntry.get())*200
    oilprice=int(oilEntry.get())*150
    daalprice=int(daalEntry.get())*110
    wheatprice=int(wheatEntry.get())*25
    sugarprice=int(sugarEntry.get())*45
    teaprice=int(teaEntry.get())*55

    totalgroceryprice=riceprice+oilprice+daalprice+wheatprice+sugarprice+teapr
    ice
    grocerypriceEntry.delete(0,END)
    grocerypriceEntry.insert(0,f'{totalgroceryprice} Rs')

    grocerytax=totalgroceryprice*0.05
    grocerytaxEntry.delete(0,END)
    grocerytaxEntry.insert(0,str( grocerytax)+ 'Rs')
```

```

#drinks price colculation
maazaprice=int(maazaEntry.get())*90
pepsiprice=int(pepsiEntry.get())*90
spriteprice=int(spriteEntry.get())*95
dewprice=int(dewEntry.get())*85
frootiprice=int(frootiEntry.get())*80
cococolaprice=int(cococolaEntry.get())*95

totaldrinksprice=
maazaprice+pepsiprice+spriteprice+dewprice+frootiprice+cococolaprice
drinkspriceEntry.delete(0,END)
drinkspriceEntry.insert(0,f'{totaldrinksprice} Rs')

drinkstax=totaldrinksprice*0.08
drinkstaxEntry.delete(0,END)
drinkstaxEntry.insert(0,str( drinkstax) + 'Rs')

totalbill=totalcosmeticprice+totalgroceryprice+totaldrinksprice+cosmeticst
ax+grocerytax+drinkstax

```

#GUI part

```

#header sections start
root=Tk()
root.title('Billing System')
root.geometry('1270x685')
root.iconbitmap('icon.ico')
headingLabel=Label(root,text='BILLING SYSTEM',font=('times new
roman',30,'bold'),bg='gray20',fg='gold',bd=12,relief=GROOVE)
headingLabel.pack(fill=X)

#header sections end

# customer details start

customer_details_frame=LabelFrame(root,text='Customer Deatils',font=('times
new roman',15,'bold'),fg='gold',bd=12,relief=GROOVE,bg='gray20')
customer_details_frame.pack(fill=X)

#Name Label start

nameLabel=Label(customer_details_frame,text='Name',font=('times new
roman',15,'bold'),bg='gray20',fg='white')
nameLabel.grid(row=0,column=0,padx=20,pady=2)

```

```

nameEntry=Entry(customer_details_frame,font=('arial',15),bd=7,width=18)
nameEntry.grid(row=0,column=1,padx=8)

#Name Label end


#Phone Label start
phoneLabel=Label(customer_details_frame,text='Phone Number',font=('times new
roman',15,'bold'),bg='gray20',fg='white')
phoneLabel.grid(row=0,column=2,padx=20)

phoneEntry=Entry(customer_details_frame,font=('arial',15),bd=7,width=18)
phoneEntry.grid(row=0,column=3,padx=8)

#Phone Label end


#Bill number Label start

billnumberLabel=Label(customer_details_frame,text='Bill Number',font=('times
new roman',15,'bold'),bg='gray20',fg='white')
billnumberLabel.grid(row=0,column=4,padx=20,pady=2)

billnumberEntry=Entry(customer_details_frame,font=('arial',15),bd=7,width=18)
billnumberEntry.grid(row=0,column=5,padx=8)

#Bill number Label end
# customer details end


#button class start
searchButton=Button(customer_details_frame,text='SEARCH',font=('arial',12,'bol
d'),bd=7,width=10,command=search_bill)
searchButton.grid(row=0,column=6,padx=20,pady=8)

#button class end


#products Frame section start

productsFrame=Frame(root)
productsFrame.pack()

#cosmeticsFrame sections start

```

```
cosmeticsFrame=LabelFrame(productsFrame,text='cosmetics',font=('times new  
roman',15,'bold'),fg='gold',bd=12,relief=GROOVE,bg='gray20')  
cosmeticsFrame.grid(row=0,column=0)
```

```
bathsoapLabel=Label(cosmeticsFrame,text='Bath Soap',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
bathsoapLabel.grid(row=0,column=0,pady=9,padx=10)
```

```
bathsoapEntry=Entry(cosmeticsFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)  
bathsoapEntry.grid(row=0,column=1,padx=10)  
bathsoapEntry.insert(0,0)
```

```
facecreamLabel=Label(cosmeticsFrame,text='Face Cream',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
facecreamLabel.grid(row=1,column=0,pady=9,padx=10)
```

```
facecreamEntry=Entry(cosmeticsFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)  
facecreamEntry.grid(row=1,column=1,padx=10)  
facecreamEntry.insert(0,0)
```

```
facewashLabel=Label(cosmeticsFrame,text='Face Wash',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
facewashLabel.grid(row=2,column=0,pady=9,padx=10)
```

```
facewashEntry=Entry(cosmeticsFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)  
facewashEntry.grid(row=2,column=1,padx=10)  
facewashEntry.insert(0,0)
```

```
hairsprayLabel=Label(cosmeticsFrame,text='Hair Spray',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
hairsprayLabel.grid(row=3,column=0,pady=9,padx=10,sticky='w')
```

```
hairsprayEntry=Entry(cosmeticsFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)
```

```
hairsprayEntry.grid(row=3,column=1,padx=10)
hairsprayEntry.insert(0,0)
```

```
hairgelLabel=Label(cosmeticsFrame,text='Hair Gel',font=('times new
roman',15,'bold'),bg='gray20',fg='white')
```

```
hairgelLabel.grid(row=4,column=0,pady=9,padx=10,sticky='w')
```

```
hairgelEntry=Entry(cosmeticsFrame,font=('times new
roman',15,'bold'),width=10,bd=5)
hairgelEntry.grid(row=4,column=1,padx=10)
hairgelEntry.insert(0,0)
```

```
bodylotionLabel=Label(cosmeticsFrame,text='Body Lotion',font=('times new
roman',15,'bold'),bg='gray20',fg='white')
```

```
bodylotionLabel.grid(row=5,column=0,pady=9,padx=10,sticky='w')
```

```
bodylotionEntry=Entry(cosmeticsFrame,font=('times new
roman',15,'bold'),width=10,bd=5)
bodylotionEntry.grid(row=5,column=1,padx=10)
bodylotionEntry.insert(0,0)
```

```
#cosmeticsFrame sections end
```

```
#groceryFrame sections start
```

```
groceryFrame=LabelFrame(productsFrame,text='Grocery',font=('times new
roman',15,'bold'),fg='gold',bd=12,relief=GROOVE,bg='gray20')
groceryFrame.grid(row=0,column=1)
```

```
riceLabel=Label(groceryFrame,text='Rice',font=('times new
roman',15,'bold'),bg='gray20',fg='white')
```

```
riceLabel.grid(row=0,column=0,pady=9,padx=10,sticky='w')
```

```
riceEntry=Entry(groceryFrame,font=('times new roman',15,'bold'),width=10,bd=5)
riceEntry.grid(row=0,column=1,padx=10)
riceEntry.insert(0,0)
```

```
oilLabel=Label(groceryFrame,text='Oil',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
oilLabel.grid(row=1,column=0,pady=9,padx=10,sticky='w')
```

```
oilEntry=Entry(groceryFrame,font=('times new roman',15,'bold'),width=10,bd=5)  
oilEntry.grid(row=1,column=1,padx=10)  
oilEntry.insert(0,0)
```

```
daalLabel=Label(groceryFrame,text='Daal',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
daalLabel.grid(row=2,column=0,pady=9,padx=10,sticky='w')
```

```
daalEntry=Entry(groceryFrame,font=('times new roman',15,'bold'),width=10,bd=5)  
daalEntry.grid(row=2,column=1,padx=10)  
daalEntry.insert(0,0)
```

```
wheatLabel=Label(groceryFrame,text='Wheat',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
wheatLabel.grid(row=3,column=0,pady=9,padx=10,sticky='w')
```

```
wheatEntry=Entry(groceryFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)  
wheatEntry.grid(row=3,column=1,padx=10)  
wheatEntry.insert(0,0)
```

```
sugarLabel=Label(groceryFrame,text='Sugar',font=('times new  
roman',15,'bold'),bg='gray20',fg='white')
```

```
sugarLabel.grid(row=4,column=0,pady=9,padx=10,sticky='w')
```

```
sugarEntry=Entry(groceryFrame,font=('times new  
roman',15,'bold'),width=10,bd=5)  
sugarEntry.grid(row=4,column=1,padx=10)  
sugarEntry.insert(0,0)
```



```

teaLabel=Label(groceryFrame,text='Tea',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

teaLabel.grid(row=5,column=0,pady=9,padx=10,sticky='w')

teaEntry=Entry(groceryFrame,font=('times new roman',15,'bold'),width=10,bd=5)
teaEntry.grid(row=5,column=1,padx=10)
teaEntry.insert(0,0)

#groceryFrame sections end


#drinksFrame sections start

drinksFrame=LabelFrame(productsFrame,text='Cold Drinks',font=('times new
roman',15,'bold'),fg='gold',bd=12,relief=GROOVE,bg='gray20')
drinksFrame.grid(row=0,column=2)

maazaLabel=Label(drinksFrame,text='Maaza',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

maazaLabel.grid(row=0,column=0,pady=9,padx=10,sticky='w')

maazaEntry=Entry(drinksFrame,font=('times new roman',15,'bold'),width=10,bd=5)
maazaEntry.grid(row=0,column=2,padx=10)
maazaEntry.insert(0,0)


pepsiLabel=Label(drinksFrame,text='Pepsi',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

pepsiLabel.grid(row=1,column=0,pady=9,padx=10,sticky='w')

pepsiEntry=Entry(drinksFrame,font=('times new roman',15,'bold'),width=10,bd=5)
pepsiEntry.grid(row=1,column=2,padx=10)
pepsiEntry.insert(0,0)


spriteLabel=Label(drinksFrame,text='Sprite',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

spriteLabel.grid(row=2,column=0,pady=9,padx=10,sticky='w')

```

```

spriteEntry=Entry(drinksFrame,font=('times new
roman',15,'bold'),width=10,bd=5)
spriteEntry.grid(row=2,column=2,padx=10)
spriteEntry.insert(0,0)
dewLabel=Label(drinksFrame,text='Dew',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

dewLabel.grid(row=3,column=0,pady=9,padx=10,sticky='w')


dewEntry=Entry(drinksFrame,font=('times new roman',15,'bold'),width=10,bd=5)
dewEntry.grid(row=3,column=2,padx=10)
dewEntry.insert(0,0)
frootiLabel=Label(drinksFrame,text='Frooti',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

frootiLabel.grid(row=4,column=0,pady=9,padx=10,sticky='w')


frootiEntry=Entry(drinksFrame,font=('times new
roman',15,'bold'),width=10,bd=5)
frootiEntry.grid(row=4,column=2,padx=10)
frootiEntry.insert(0,0)
cococolaLabel=Label(drinksFrame,text='Coco Cola',font=('times new
roman',15,'bold'),bg='gray20',fg='white')

cococolaLabel.grid(row=5,column=0,pady=9,padx=10,sticky='w')


cococolaEntry=Entry(drinksFrame,font=('times new
roman',15,'bold'),width=10,bd=5)

cococolaEntry.grid(row=5,column=2,padx=10)
cococolaEntry.insert(0,0)

#drinksFrame sections end
#products Frame section end

#billframe section start


billframe=Frame(productsFrame,bd=8,relief=GROOVE)
billframe.grid(row=0,column=3,padx=10)
billareaLabel=Label(billframe,text='Bill Area',font=('times new
roman',15,'bold'),bd=8,relief=GROOVE)

billareaLabel.pack(fill=X)


#VERTICAL view of text area

```

```

Scrollbar=Scrollbar(billframe,orient=VERTICAL)
Scrollbar.pack(side=RIGHT,fill=Y)
textarea=Text(billframe,height=16,width=55,yscrollcommand=Scrollbar.set)
textarea.pack()

#Scrollbar config

Scrollbar.config(command=textarea.yview)
#billframe section start
billmenuFrame=LabelFrame(root,text='Bill Menu',font=('times new
roman',15,'bold'),fg='gold',bd=12,relief=GROOVE,bg='gray20')

billmenuFrame.pack()

cosmeticspriceLabel=Label(billmenuFrame,text='Cosmetic Price',font=('times new
roman',14,'bold'),bg='gray20',fg='white')
cosmeticspriceLabel.grid(row=0,column=0,pady=6,padx=10,sticky='w')
cosmeticspriceEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)
cosmeticspriceEntry.grid(row=0,column=1,padx=10,pady=6)
grocerypriceLabel=Label(billmenuFrame,text='Grocery Price',font=('times new
roman',14,'bold'),bg='gray20',fg='white')

grocerypriceLabel.grid(row=1,column=0,pady=6,padx=10,sticky='w')

grocerypriceEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)

grocerypriceEntry.grid(row=1,column=1,padx=10,pady=6)

drinkspriceLabel=Label(billmenuFrame,text='Cold Drink Price',font=('times new
roman',14,'bold'),bg='gray20',fg='white')

drinkspriceLabel.grid(row=2,column=0,pady=6,padx=10,sticky='w')

drinkspriceEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)
drinkspriceEntry.grid(row=2,column=1,padx=10,pady=6)

cosmeticstaxLabel=Label(billmenuFrame,text='Cosmetic Tax',font=('times new
roman',14,'bold'),bg='gray20',fg='white')

cosmeticstaxLabel.grid(row=0,column=2,pady=6,padx=10,sticky='w')

cosmeticstaxEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)
cosmeticstaxEntry.grid(row=0,column=3,padx=10,pady=6)

```

```

grocerytaxLabel=Label(billmenuFrame,text='Grocery Tax',font=('times new
roman',14,'bold'),bg='gray20',fg='white')

grocerytaxLabel.grid(row=1,column=2,pady=6,padx=10,sticky='w')

grocerytaxEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)
grocerytaxEntry.grid(row=1,column=3,padx=10,pady=6)

drinkstaxLabel=Label(billmenuFrame,text='Cold Drink Tax',font=('times new
roman',14,'bold'),bg='gray20',fg='white')

drinkstaxLabel.grid(row=2,column=2,pady=6,padx=10,sticky='w')

drinkstaxEntry=Entry(billmenuFrame,font=('times new
roman',14,'bold'),width=10,bd=5)
drinkstaxEntry.grid(row=2,column=3,padx=10,pady=6)


buttonFrame=Frame(billmenuFrame,bd=8,relief=GROOVE)
buttonFrame.grid(row=0,column=4,rowspan=3)

totalButton=Button(buttonFrame,text='Total',font=('arial',16,'bold'),bg='gray2
0',fg='white',bd=5,width=8,pady=10,command=total)
totalButton.grid(row=0,column=0,pady=20,padx=5)

billButton=Button(buttonFrame,text='Bill',font=('arial',16,'bold'),bg='gray20'
,fg='white',bd=5,width=8,pady=10,command=bill_area)
billButton.grid(row=0,column=1,pady=20,padx=5)

emailButton=Button(buttonFrame,text='Email',font=('arial',16,'bold'),bg='gray2
0',fg='white',bd=5,width=8,pady=10,command=send_email)
emailButton.grid(row=0,column=2,pady=20,padx=5)

printButton=Button(buttonFrame,text='Print',font=('arial',16,'bold'),bg='gray2
0',fg='white',bd=5,width=8,pady=10,command=print_bill)
printButton.grid(row=0,column=3,pady=20,padx=5)

```

```
clearButton=Button(buttonFrame,text='Clear',font=('arial',16,'bold'),bg='gray20',fg='white',bd=5,width=8,pady=10,command=clear)
clearButton.grid(row=0,column=4,pady=20,padx=5)
```

```
root.mainloop()
```

USER INTERFACE LAYOUT

BILLING SYSTEM

Customer Details

Name Phone Number Bill Number **SEARCH**

cosmetics	Grocery	Cold Drinks
Bath Soap <input type="text"/>	Rice <input type="text"/>	Maaza <input type="text"/>
Face Cream <input type="text"/>	Oil <input type="text"/>	Pepsi <input type="text"/>
Face Wash <input type="text"/>	Daal <input type="text"/>	Sprite <input type="text"/>
Hair Spray <input type="text"/>	Wheat <input type="text"/>	Dew <input type="text"/>
Hair Gel <input type="text"/>	Sugar <input type="text"/>	Frooti <input type="text"/>
Body Lotion <input type="text"/>	Tea <input type="text"/>	Coco Cola <input type="text"/>

Bill Area

Bill Menu

Cosmetic Price <input type="text"/>	Cosmetic Tax <input type="text"/>	Total Bill Email Print Clear
Grocery Price <input type="text"/>	Grocery Tax <input type="text"/>	
Cold Drink Price <input type="text"/>	Cold Drink Tax <input type="text"/>	

4.1 OUTPUT

Billing System

BILLING SYSTEM

Customer Deatils

Name
AVANISH
Phone Number
7390987389
Bill Number
SEARCH

cosmetics

Bath Soap
3
Face Cream
3
Face Wash
3
Hair Spray
1
Hair Gel
1
Body Lotion
1

Grocery

Rice
1
Oil
1
Daal
1
Wheat
1
Sugar
1
Tea
1

Cold Drinks

Maaza
1
Pepsi
1
Sprite
1
Dew
1
Frooti
1
Coco Cola
1

Bill Area

WELCOME CUSTOMER

Bill Number:10102

Customer Name:AVANISH

Customer Phone Number:7390987389

Product	Quantity	Price
Bath Soap	3	60 Rs
Face Cream	3	150 Rs
Face Wash	3	300 Rs
Hair Spray	1	150 Rs
Hair Gel	1	80 Rs

Bill Menu

Cosmetic Price
800 Rs
Cosmetic Tax
96.0Rs

Grocery Price
585 Rs
Grocery Tax
29.25Rs

Cold Drink Price
535 Rs
Cold Drink Tax
42.80000000

Total
Bill
Email
Print
Clear

29°C
Mostly cloudy

Search

ENG
IN

07:06
22-07-2023

SAVE THE BILL

****WELCOME CUSTOMER****

Bill Number:4102

Customer Name:xyz

Customer Phone Number:1234567809

```
=====
Product                Quantity                Price
=====
Bath Soap              1                20 Rs
Face Cream             5               250 Rs
Face Wash              7               700 Rs
Hair Spray            5               750 Rs
Wheat                  1
Pepsi                  1
```


```
-----
Cosmetic Tax                206.4Rs
Grocery Tax                 1.25Rs
Drink Tax                   7.2Rs
-----
```

```
-----
Total Bill                2049.85
-----
```

****THANKYOU VISIT AGAIN****

****WELCOME CUSTOMER****

SENDING GMAIL

 SEND GMAIL—□×

SENDER

Sender's Email

sumi.skb1960@gmail.com

Password

RECIPIENT

Email Address

sumi.skb1960@gmail.com

Message

****WELCOME CUSTOMER****

Bill Number:7445

Customer Name:xyz

Customer Phone Number:5876990078

Product	Quantity	Price
---------	----------	-------

SEND

BILLING SYSTEM

Customer Deatils

Name

Phone Number

Bill Number

SEARCH

Cosmetics

Grocery

Cold Drinks

Bill Area

WELCOME CUSTOMER

Bill Number:7445
 Customer Name:xyz
 Customer Phone Number:5876990078

Product	Quantity	Price
Bath Soap	1	20 Rs
Hair Spray	1	150 Rs
Hair Gel	5	400 Rs
Oil	5	750 Rs
Daal	3	330 Rs

Bath Soap

Rice

Maaza

Face Cream

Oil

Face Wash

Daal

Hair Spray

Wheat

Hair Gel

Sugar

Body Lotion

Tea

Coco Cola

Bill Menu

Cosmetic Price

Cosmetic Tax

Grocery Price

Grocery Tax

Cold Drink Price

Cold Drink Tax

Total
Bill
Email
Print
Clear

Inbox x



****WELCOME CUSTOMER****

Customer Name:xyz

Product	Quantity	Price
---------	----------	-------

Cosmetic Tax	68.39999999999999Rs
Grocery Tax	54.0Rs
Drink Tax	51.2Rs

⬅ Reply ➡ Forward

CHAPTER 5

ADVANTAGES & DISADVANTAGES

5.1 Advantages

- It will give curate accurate result as compared to manual calculation.
- Affordable and cost-effective
- User friendly: no expert knowledge required
- Reducing long queues: customer don't have to wait long
- Reduce manual working: No need of manual calculation.
- Time saving: User don't need to look for menu card, they can directly see on application.

5.2 DISADVANTAGES

- Backend is not included, so it can't save bill in database.

5.3 Future scope

This application may be modified in the future to include additional food products. There will also be a lot of new features introduced. The project will undoubtedly be improved in terms of aesthetic and usability as well. There will be an addition of numerous new features. Calculators can also be used for some enhancement. Currently, this application only generates bills, but in the future, it will be improved so that it can also send WhatsApp messages. It can also be applied broadly. There are many more modifications that can be made to the menu, prices, or tax. All potential consumers will find it to be simple to use and bug-free. Future improvements can also be made in accordance with consumer needs.

CHAPTER 6

CONCLUSION AND REFERENCE

6.1 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 fastest 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 improvement and working.

6.2 Reference

<https://www.slideshare.net/>

<https://stackoverflow.com/>

<https://www.google.com/>