McDonalds Management System

I. INTRODUCTION

A. Why Management Systems are required?

The benefits of an effective management system to an organisation include:

1. More efficient use of resources and improved financial performance.
2. Improved risk management.
3. Protection of people and the environment.

A successful business depends on many processes that work in sync with each other. **McDonalds Management System** is designed with features to help manage and operate different aspects of their restaurants more efficiently and more profitably.

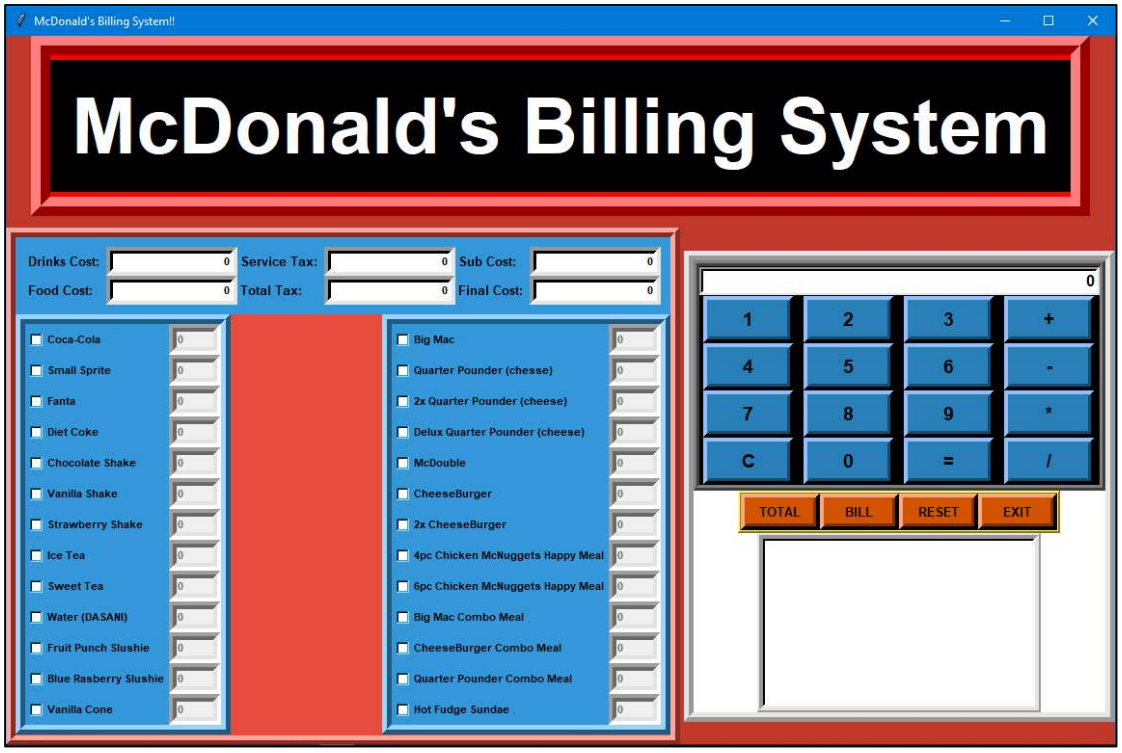
B. What does the management system do?

The management system displays the **menu of McDonalds**, the customer **selects the items** and the **number of items** required. The management system then returns a **sample bill**, including **taxes** paid, and the **costs** given for several items. If the customer needs to confirm again the calculation made by the application, the application comes with a **small calculator**.

II. PROJECT INSIGHT

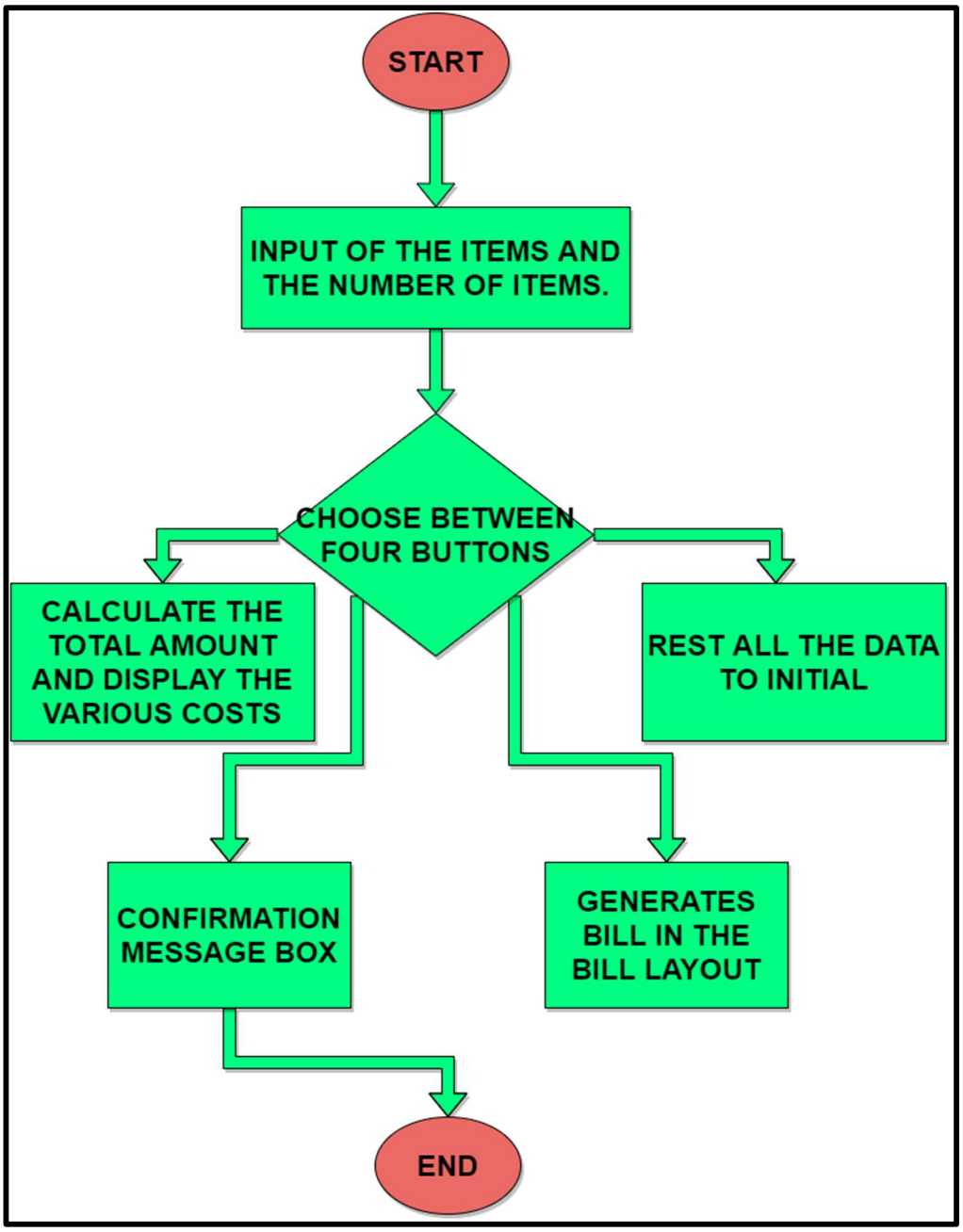
C. THE FINAL LAYOUT OF THE APPLICATION

The Final Layout is as shown below:



D. FLOWCHART FOR THE SYSTEM

The Flowchart for the application is shown in Figure below:



E. CODE FOR THE PROJECT

The link to the GitHub Repository for the project is https://github.com/Isha1504/McDonald-s-Billing-System

F. CODE EXPLANATION

The first step in any application building is **importing modules** required for the project. Figure1 shows the modules imported.

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

Figure1: Modules Imported

The next step is creating a basic layout for the system. For this application **Frames** were used to get better section by section design. Frame functions like a container that arranges the position of other widgets inside it and is organized with the rectangular sections in the monitor. Figure2 shows the frames which were used for the application.

```
window=Tk()
window.geometry("1150x780+0+0")
window.title("McDonald's Billing System!!")
window.configure(bg="#C0392B")

head=Frame(window,bg="Red",bd=20,pady=5,relief=RIDGE)
head.pack(side=TOP)
app_title=Label(head,text="McDonald's Billing System",
                font=('arial',60,'bold'),bd=21,
                bg="Black",fg="White",justify=CENTER)
app_title.grid(row=0,column=0)

bill=Frame(window,bg="White",bd=10,relief=RIDGE)
bill.pack(side=RIGHT)
bill1=Frame(bill,bg="White",bd=4,relief=RIDGE)
bill1.pack(side=BOTTOM)
cal_frame=Frame(bill,bg="Black",bd=6,relief=RIDGE)
cal_frame.pack(side=TOP)
btn_frame1=Frame(bill,bg="#F1C40F",bd=3,relief=RIDGE)
btn_frame1.pack(side=BOTTOM)

menu=Frame(window,bg="#E74C3C",bd=10,relief=RIDGE)
menu.pack(side=LEFT)
drinks_frame3=Frame(menu,bg="#3498DB",bd=4)
drinks_frame3.pack(side=BOTTOM)
cost_frame3=Frame(menu,bg="#3498DB",bd=10)
cost_frame3.pack(side=TOP)
drinks_frame3=Frame(menu,bg="#3498DB",bd=10,relief=RIDGE)
drinks_frame3.pack(side=LEFT)
burger_frame=Frame(menu,bg="#3498DB",bd=10,relief=RIDGE)
burger_frame.pack(side=RIGHT)
```

Figure2: Frames used in the application

The frames used in the application are as follows:

1. **Head Frame** – Contains the title of the application.
2. **Bill Frame** – Contains the final bill gets displayed.
3. **Calculator Frame** – Contains the calculator in the application.
4. **Button Frame** – Contains the main operating buttons.
5. **Menu Frame** – Contains the whole menu of McDonald's.
    a. **Drinks Frame** – Contains List of drinks.

b. **Burger/Food Frame** – Contains burgers and various meals list.

The next step on building layout is adding things to the layout. For Each drink in drink frame. We define a separate **Check buttons** and put them on separate rows as shown in Figure3.

```
cola=Checkbutton(drinks_frame3,text="Coca-Cola",variable=v0,onvalue=-1,
                 offvalue=0,font=('arial',8,'bold'),
                 bg="#3498DB",command=item1).grid(row=0,sticky=W)
sprite=Checkbutton(drinks_frame3,text="Small Sprite",variable=v1,onvalue=-1,
                   offvalue=0,font=('arial',8,'bold'),
                   bg="#3498DB",command=item2).grid(row=1,sticky=W)
```

Figure3: Check button for 2 out of 13 drinks

Similarly, we define each Check button for every burger, meals and ice creams as shown in Figure4.

```
cream2=Checkbutton(burger_frame,text="Hot Fudge Sundae",variable=v13,
                   onvalue=-1,offvalue=0,font=('arial',8,'bold'),
                   bg="#3498DB",command=item14).grid(row=12,sticky=W)
B1=Checkbutton(burger_frame,text="Big Mac",variable=v14,onvalue=-1,
               offvalue=0,font=('arial',8,'bold'),
               bg="#3498DB",command=item15).grid(row=0,sticky=W)
```

Figure4: Check buttons for 2 out of 13 Food stuff

We also define the variables for each of the check button for both drinks and food as shown in Figure5.

```
v0=IntVar()
v1=IntVar()
v2=IntVar()
v3=IntVar()
v4=IntVar()
v5=IntVar()
v6=IntVar()
v7=IntVar()
v8=IntVar()
v9=IntVar()
v10=IntVar()
v11=IntVar()
v12=IntVar()
v13=IntVar()
```

Figure5: Variables defined for 13 drinks

We then display the layout for each of the item to make sure the number of each items to order is also displayed on the screen. Figure6 shows the code for two of the 26 entry boxes used for the menu of both drinks and food. And Figure7 shows the output till now.

```
e1=Entry(drinks_frame3,font=('arial',8,'bold'),bd=8,width=6,
         justify=LEFT,textvariable=ev1,state=DISABLED)
e1.grid(row=0,column=1)
e2=Entry(drinks_frame3,font=('arial',8,'bold'),bd=8,width=6,
         justify=LEFT,textvariable=ev2,state=DISABLED)
e2.grid(row=1,column=1)
```

Figure6: Code for the entry boxes of 2 of the 26 entry boxes of menu

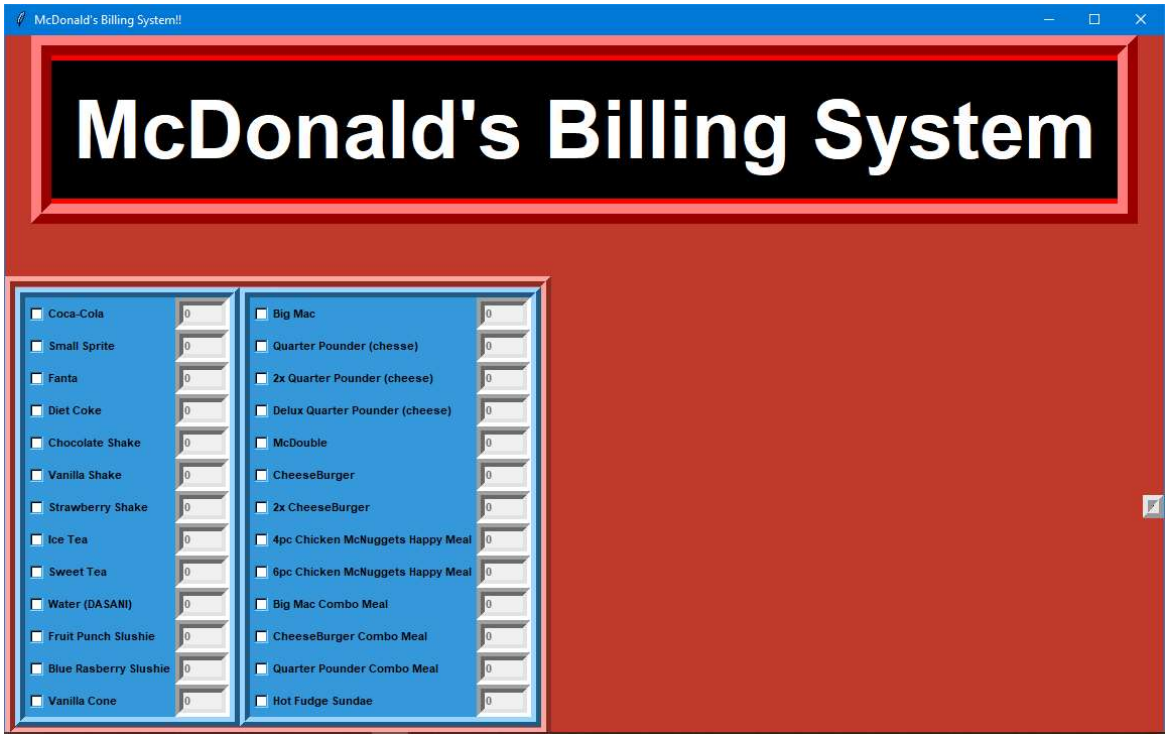


Figure7: Layout1 of the application

The next step is to create the Bill Layout and the buttons such as Total, Generate Bill, reset everything and Exit Button and Work over the same each button at a time. The same is shown in Figure8.

```
final_bill=Text(bill1,width=46,height=12,bg="White",bd=4,
                font=('arial',8,'bold'))
final_bill.grid(row=0,column=0)
total_btn=Button(btn_frame1,padx=16,pady=1,bd=7,fg="black",
                 font=('arial',10,'bold'),width=4,text="TOTAL",
                 bg="#D35400",command=calculate_cost).grid(row=0,column=0)
bill_btn=Button(btn_frame1,padx=16,pady=1,bd=7,fg="black",
                font=('arial',10,'bold'),width=4,text="BILL",
                bg="#D35400",command=GenerateBill).grid(row=0,column=1)
reset_btn=Button(btn_frame1,padx=16,pady=1,bd=7,fg="black",
                 font=('arial',10,'bold'),width=4,text="RESET",
                 bg="#D35400",command=Reset_data).grid(row=0,column=2)
exit_btn=Button(btn_frame1,padx=16,pady=1,bd=7,fg="black",
                font=('arial',10,'bold'),width=4,text="EXIT",
                bg="#D35400",command=ExitSystem).grid(row=0,column=3)
```

Figure8: Layout of the Bill generating area and main buttons

The output of the same is shown in Figure9. Then next part is making the buttons work. The buttons are as follows:

1. Exit Button – Opens a prompt to confirm whether user wants to exit or not. The code for the same function is shown in Figure10.
2. Reset Button – This button resets all the values in the menu and in the bill layout to initial value. Part of the code is shown in Figure11.
3. Bill Button – Generates the Bill using simple text insert function to insert the values at the right place. A part of the whole function is shown in Figure12.
4. Total Button – Total button is used to calculate the various cost which will come in the later sections.

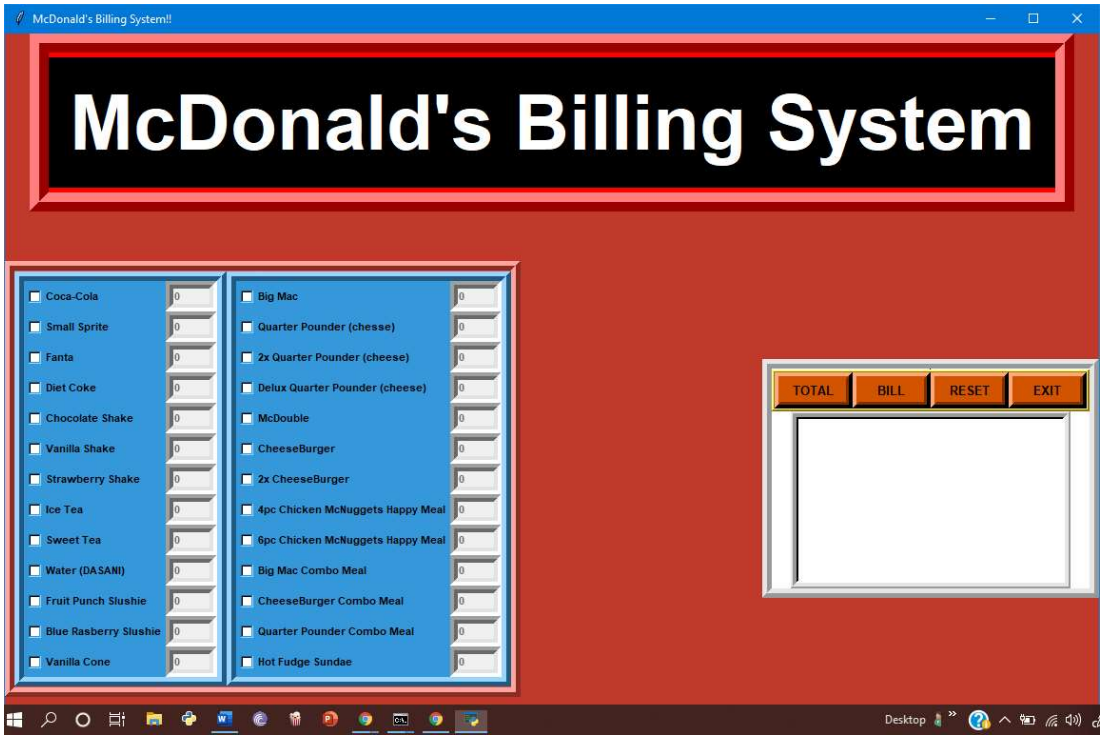


Figure9: The layout with the Bill generating area and main operating buttons

```
def ExitSystem():
    exitmsg=tkinter.messagebox.askyesno("EXIT McDonald's System ?",
                                        "Please Confirm!")
    if exitmsg>0:
        window.destroy()
        return
```

Figure10: The code for Exit System in the application

```
def Reset_data():
    dcost.set("0")
    bcost.set("0")
    STcost.set("0")
    TTcost.set("0")
    Subcost.set("0")
    Totcost.set("0")

    final_bill.delete("1.0",END)

    ev1.set("0")
    ev2.set("0")
    ev3.set("0")
    ev4.set("0")
    ev5.set("0")
    ev6.set("0")
    ev7.set("0")
    ev8.set("0")
    ev9.set("0")
    ev10.set("0")
    ev11.set("0")
    ev12.set("0")
    ev13.set("0")
    ev14.set("0")
    ev15.set("0")
    ev16.set("0")
    ev17.set("0")
    ev18.set("0")
    ev19.set("0")
    ev20.set("0")
    ev21.set("0")
    ev22.set("0")
    ev23.set("0")
    ev24.set("0")
    ev25.set("0")
    ev26.set("0")

    v0.set(0)
    v1.set(0)
    v2.set(0)
    v3.set(0)
    v4.set(0)
    v5.set(0)
    v6.set(0)
    v7.set(0)
    v8.set(0)
    v9.set(0)
```

Figure11: A part of the reset button code to reset all the data

```
def GenerateBill():
    final_bill.delete("1.0",END)
    calculate_cost()
    x=random.randint(10209,9808308)
    final_bill.insert(END,"\tWELCOME TO MCDONALD'S!")
    final_bill.insert(END,"\n\t    Here is your bill")
    final_bill.insert(END,"\n\t    Bill Number:"+str(x))
    final_bill.insert(END,"\n\n\t  Item"+"\t\t\t   Number")
    if(e1.get()!="0"):
        final_bill.insert(END,"\n\tCoca Cola\t\t\t:\t"+str(e1.get()))
    if(e2.get()!="0"):
        final_bill.insert(END,"\n\tSmall Sprite\t\t\t:\t"+str(e2.get()))
    if(e3.get()!="0"):
        final_bill.insert(END,"\n\tFanta\t\t\t:\t"+str(e3.get()))
    if(e4.get()!="0"):
        final_bill.insert(END,"\n\tDiet Coke\t\t\t:\t"+str(e4.get()))
    if(e5.get()!="0"):
        final_bill.insert(END,"\n\tChocolate Shake\t\t\t:\t"+str(e5.get()))
    if(e6.get()!="0"):
        final_bill.insert(END,"\n\tVanilla Shake\t\t\t:\t"+str(e6.get()))
    if(e7.get()!="0"):
        final_bill.insert(END,"\n\tStrawberry Shake\t\t\t:\t"+str(e7.get()))
    if(e8.get()!="0"):
        final_bill.insert(END,"\n\tIce Tea\t\t\t:\t"+str(e8.get()))
    if(e9.get()!="0"):
        final_bill.insert(END,"\n\tSweet Tea\t\t\t:\t"+str(e9.get()))
    if(e10.get()!="0"):
        final_bill.insert(END,"\n\tWater Bottle\t\t\t:\t"+str(e10.get()))
    if(e11.get()!="0"):
        final_bill.insert(END,"\n\tFruit Punch Slushie\t\t\t:\t"+str(e11.get()))
    if(e12.get()!="0"):
        final_bill.insert(END,"\n\tBlue Rasberry Slushie\t\t\t:\t"+str(e12.get()))
    if(e13.get()!="0"):
        final_bill.insert(END,"\n\tVanilla Cone\t\t\t:\t"+str(e13.get()))
    if(e26.get()!="0"):
        final_bill.insert(END,"\n\tHot Fudge Sundae\t\t\t:\t"+str(e26.get()))
    if(e14.get()!="0"):
        final_bill.insert(END,"\n\tBig Mac\t\t\t:\t"+str(e14.get()))
    if(e15.get()!="0"):
        final_bill.insert(END,"\n\tQuarter Pounder\t\t\t:\t"+str(e15.get()))
    if(e16.get()!="0"):
        final_bill.insert(END,"\n\t2x Quarter Pounder\t\t\t:\t"+str(e16.get()))
    if(e17.get()!="0"):
        final_bill.insert(END,"\n\tDelux Quarter Pounder\t\t\t:\t"+str(e17.get()))
    if(e18.get()!="0"):
```

Figure12: A part of Bill Generating code using simple insert function and checking which items are checked and which are not.

Next part of the code is to build the calculator and the labels and entry boxes for various cost labels which are as follows:

1. Cost of Drinks
2. Cost of Food
3. Service Tax
4. Total Tax
5. Sub Cost
6. Total Cost

The code for the label and entry of the various cost is shown in Figure13 and Figure 14 shows to calculate the various cost and display it.

```
drink_cost=Label(cost_frame3,font=('arial',10,'bold'),text="Drinks Cost: ",bg="#3498DB",fg="Black")
drink_cost.grid(row=0,column=0,sticky=W)
drink_cost_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                       font=('arial',8,'bold'),textvariable=dcost)
drink_cost_entry.grid(row=0,column=1)
burger_cost=Label(cost_frame3,font=('arial',10,'bold'),text="Food Cost: ",bg="#3498DB",fg="Black")
burger_cost.grid(row=1,column=0,sticky=W)
burger_cost_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                        font=('arial',8,'bold'),textvariable=bcost)
burger_cost_entry.grid(row=1,column=1)

serv_tax=Label(cost_frame3,font=('arial',10,'bold'),text="Service Tax: ",bg="#3498DB",fg="Black")
serv_tax.grid(row=0,column=2,sticky=W)
serv_tax_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                     font=('arial',8,'bold'),textvariable=STcost)
serv_tax_entry.grid(row=0,column=3)
total_tax=Label(cost_frame3,font=('arial',10,'bold'),text="Total Tax: ",bg="#3498DB",fg="Black")
total_tax.grid(row=1,column=2,sticky=W)
total_tax_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                      font=('arial',8,'bold'),textvariable=TTcost)
total_tax_entry.grid(row=1,column=3)

sub_cost=Label(cost_frame3,font=('arial',10,'bold'),text="Sub Cost: ",bg="#3498DB",fg="Black")
sub_cost.grid(row=0,column=4,sticky=W)
sub_cost_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                     font=('arial',8,'bold'),textvariable=Subcost)
sub_cost_entry.grid(row=0,column=5)
total_cost=Label(cost_frame3,font=('arial',10,'bold'),text="Final Cost: ",bg="#3498DB",fg="Black")
total_cost.grid(row=1,column=4,sticky=W)
total_cost_entry=Entry(cost_frame3,bd=7,bg="White",insertwidth=2,justify=RIGHT,
                       font=('arial',8,'bold'),textvariable=Totcost)
total_cost_entry.grid(row=1,column=5)
```

Figure13: Code for the labels and entries of the various cost types

```
def calculate_cost():
    i1=ev1.get()
    i2=ev2.get()
    i3=ev3.get()
    i4=ev4.get()
    i5=ev5.get()
    i6=ev6.get()
    i7=ev7.get()
    i8=ev8.get()
    i9=ev9.get()
    i10=ev10.get()
    i11=ev11.get()
    i12=ev12.get()
    i13=ev13.get()
    i14=ev14.get()
    i15=ev15.get()
    i16=ev16.get()
    i17=ev17.get()
    i18=ev18.get()
    i19=ev19.get()
    i20=ev20.get()
    i21=ev21.get()
    i22=ev22.get()
    i23=ev23.get()
    i24=ev24.get()
    i25=ev25.get()
    i26=ev26.get()

    costdrink= (i1*10)+(i2*10)+(i3*10)+(i4*10)+(i5*10)+(i6*10)+(i7*10)+(i8*10)+(i9*10)+(i10*10)+(i11*1
    costburger= (i14*10)+(i15*10)+(i16*10)+(i17*10)+(i18*10)+(i19*10)+(i20*10)+(i21*10)+(i22*10)+(i23*

    drinktext=str(costdrink)
    burgertext=str(costburger)

    dcost.set(drinktext)
    bcost.set(burgertext)
    STcost.set("10")
    SUBCOST=(costdrink+costburger+10)
    Subcost.set(str(SUBCOST))
    TAX=SUBCOST*0.15
    Totalcost=SUBCOST+TAX
    TTcost.set(str(TAX))
    Totcost.set(str(Totalcost))
```

Figure14: Calculation of the cost function

Now the next task is to build the calculator. The calculator code is written in calculator frame. The code for the same is mentioned in Figure15 and Figure16 shows the function to do the calculation.

```
calculator=Entry(cal_frame,width=45,bg="White",bd=4,font=('arial',12,'bold'),
                 justify=RIGHT,textvariable=text)
calculator.grid(row=0,column=0,columnspan=4,pady=1)
calculator.insert(0,"0")
total_btn=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),width=4,text="1",bg
btn0=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="2",bg="#2980B9",command=lambda:Number(2)).grid(row=1,column=1)
btn1=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="3",bg="#2980B9",command=lambda:Number(3)).grid(row=1,column=2)
btn2=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="+",bg="#2980B9",command=lambda:Number('+')).grid(row=1,column=3)
btn3=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="4",bg="#2980B9",command=lambda:Number(4)).grid(row=2,column=0)
btn4=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="5",bg="#2980B9",command=lambda:Number(5)).grid(row=2,column=1)
btn5=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="6",bg="#2980B9",command=lambda:Number(6)).grid(row=2,column=2)
btn6=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="-",bg="#2980B9",command=lambda:Number('-')).grid(row=2,column=3)
btn7=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="7",bg="#2980B9",command=lambda:Number(7)).grid(row=3,column=0)
btn8=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="8",bg="#2980B9",command=lambda:Number(8)).grid(row=3,column=1)
btn9=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
            width=4,text="9",bg="#2980B9",command=lambda:Number(9)).grid(row=3,column=2)
btn10=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
             width=4,text="*",bg="#2980B9",command=lambda:Number('*')).grid(row=3,column=3)
btn11=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),w
             idth=4,text="C",bg="#2980B9",command=Clear).grid(row=4,column=0)
btn12=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
             width=4,text="0",bg="#2980B9",command=lambda:Number(0)).grid(row=4,column=1)
btn13=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
             width=4,text="=",bg="#2980B9",command=Calculate).grid(row=4,column=2)
btn14=Button(cal_frame,padx=14,pady=1,bd=7,fg="black",font=('arial',14,'bold'),
             width=4,text="/",bg="#2980B9",command=lambda:Number('/')).grid(row=4,column=3)
```

Figure15: Layout design code for calculator

```
cal_text=""
text=StringVar()

def Number(x):
    global cal_text
    cal_text = cal_text+str(x)
    text.set(cal_text)
def Clear():
    global cal_text
    cal_text = ""
    text.set("")
def Calculate():
    global cal_text
    result=str(eval(cal_text))
    text.set(result)
    cal_text=""
```

Figure16: Code for the working of calculator

The final Layout is shown in Figure17 with the calculator.

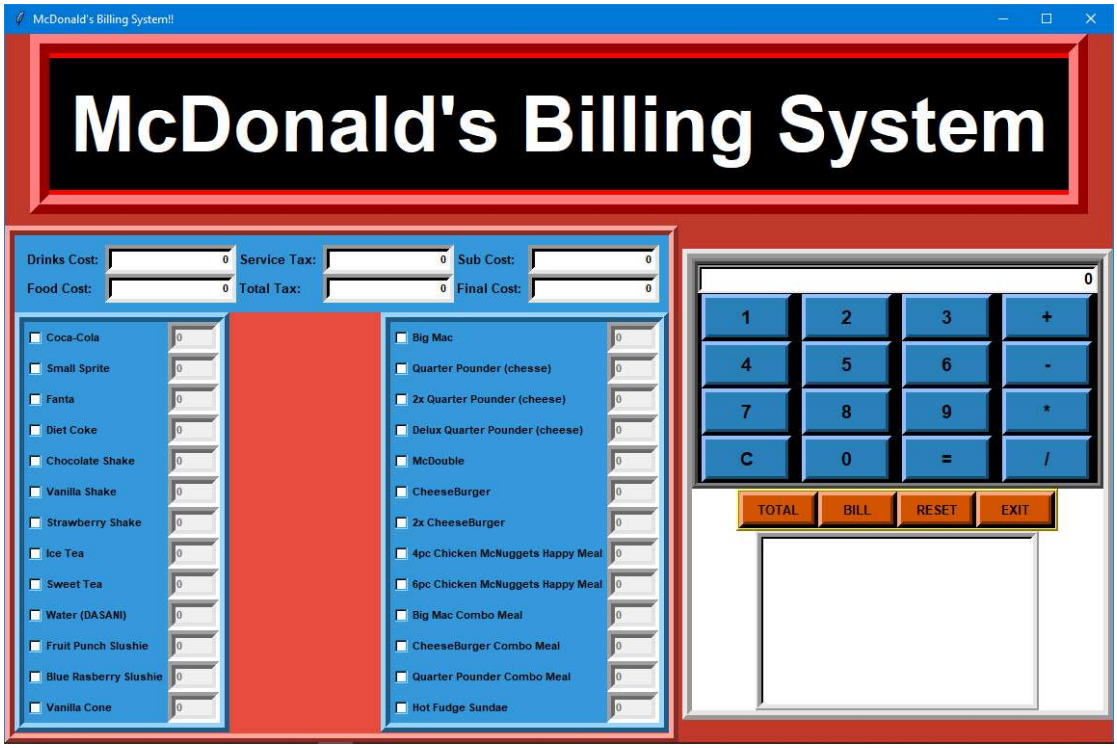


Figure17: Final Layout of the system

G. SAMPLE OUTPUTS

Figure shows the sample output with items entered and bill generated

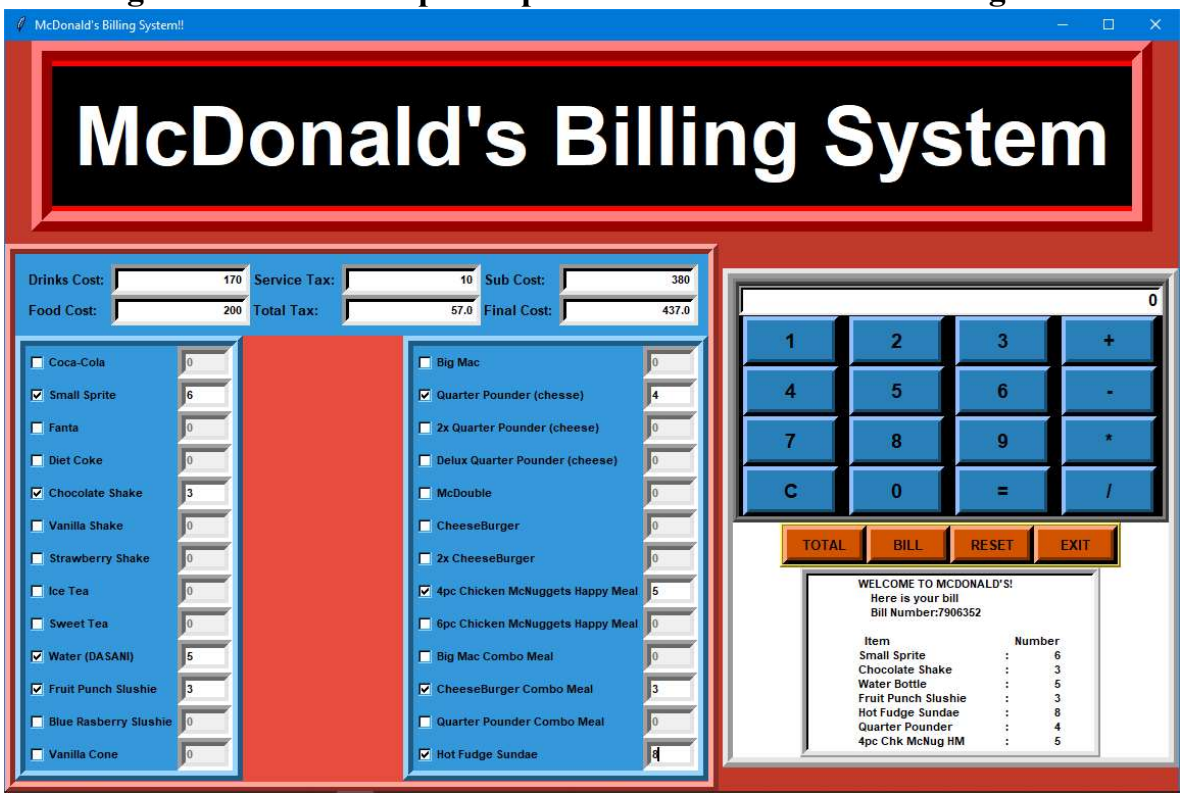
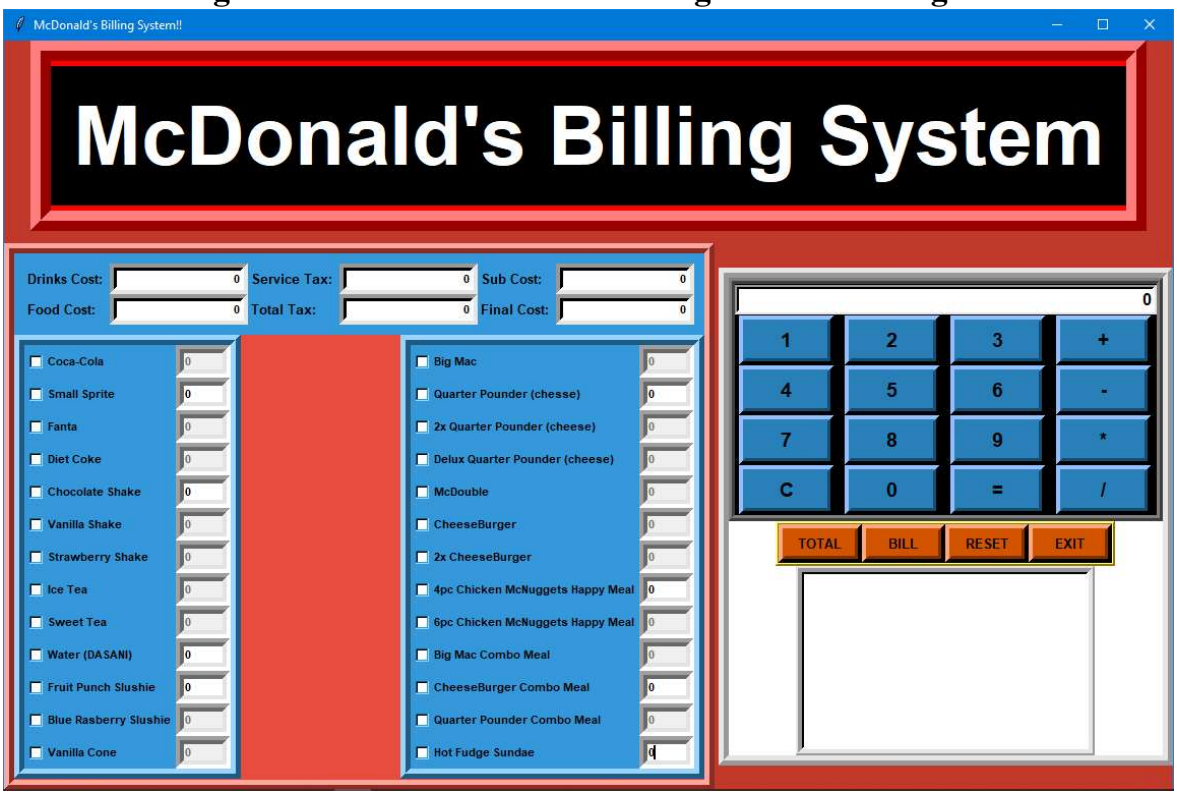


Figure shows the screen on clicking reset Button again



The Last Figure shows the exit confirmation on the screen