# Report for Health Care Management System

As a project work for Course

PYTHON PROGRAMMING (INT 213)

| | |
|---|---|
| Name | : Nikita Tayal |
| Registration Number | : 12016245 |
| Name | : Jiya Sharma |
| Registration Number | : 12016642 |
| Program | : B.Tech CSE |
| Semester | : Third |
| School | : School of Computer Science and Engineering |
| Date of Submission | : 20 November, 2021 |

Lovely Professional University Jalandhar, Punjab, India

# HEALTHCARE MANAGEMENT SYSTEM

20th November, 2021

## ABSTRACT :

Our e-Health Care portal is a secure online web portal that gives patients convenient, 24-hour access to book an appointment with the doctor via an Internet connection. The clearest benefits to a health portal is the added ability for communication between patients and hospital, and these benefits are felt strongest with regard to chronically ill patients. This saves the patient’s time, also keep them organized, up to date, and deliver a higher overall level of convenience.

## ACKNOWLEDGEMENT :

I would like to thank my mentor - Prof. Ankita Wadhawan who gave us the opportunity to do this project on the topic Healthcare management system. Also, I would like to thank my group member for her help and support.

## TABLE OF CONTENTS

# INTRODUCTION :

## 1.1 Context :

This project has been done as part of my course for the CSE at Lovely Professional University. Supervised by Prof. Ankita Wadhawan, I have one month to fulfill the requirements in order to succeed the module.

## 1.2 Motivation :

Being extremely interested in everything having a relation with the programming, the group project was a great occasion to give us the time to learn and confirm our interest for this field. The fact that we can make an online health portal which will be helping a lot of people by saving their time. We can use python and tkinter in Finance, Medicine, almost everywhere. That’s why I decided to conduct my project around this.

## 1.3 Idea :

As a first experience, we wanted to make a project by approaching every different steps of using tkinter to create the interface that will be helpful for people. We chose to take Health Care Portal as approach. The goal was to book the appointment with the doctor via Internet connection.

## **TEAM MEMBERS** :

Nikita Tayal (12016245)

Contributions :

- Coding
- GUI

Jiya Sharma (12016642)

Contributions :

- Coding
- GUI

## LIBRARIES AND MODULES :

### 1. 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.

### 2. CSV -

The CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. The csv module's reader and writer objects read and write sequences.

### 3. OS -

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

## 4. Tkinter.ttk -

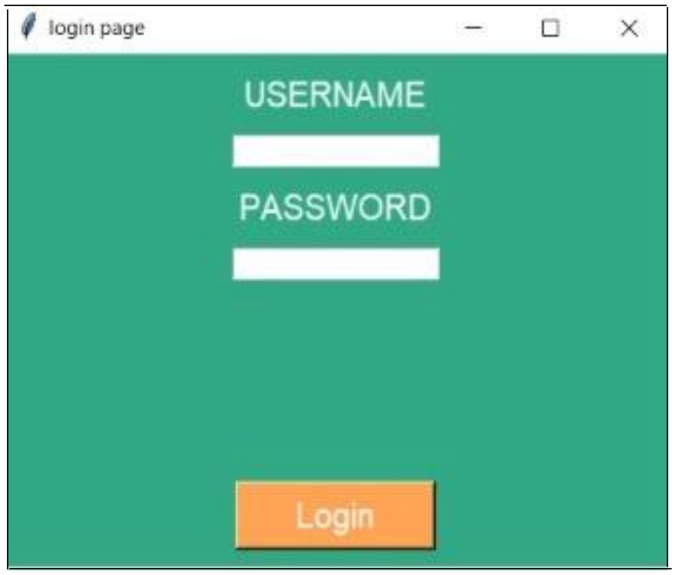
The tkinter.ttk module provides access to the Tk themed widget set, introduced in Tk 8.5. The basic idea for tkinter.ttk is to separate, to the extent possible, the code implementing a widget's behavior from the code implementing its appearance.

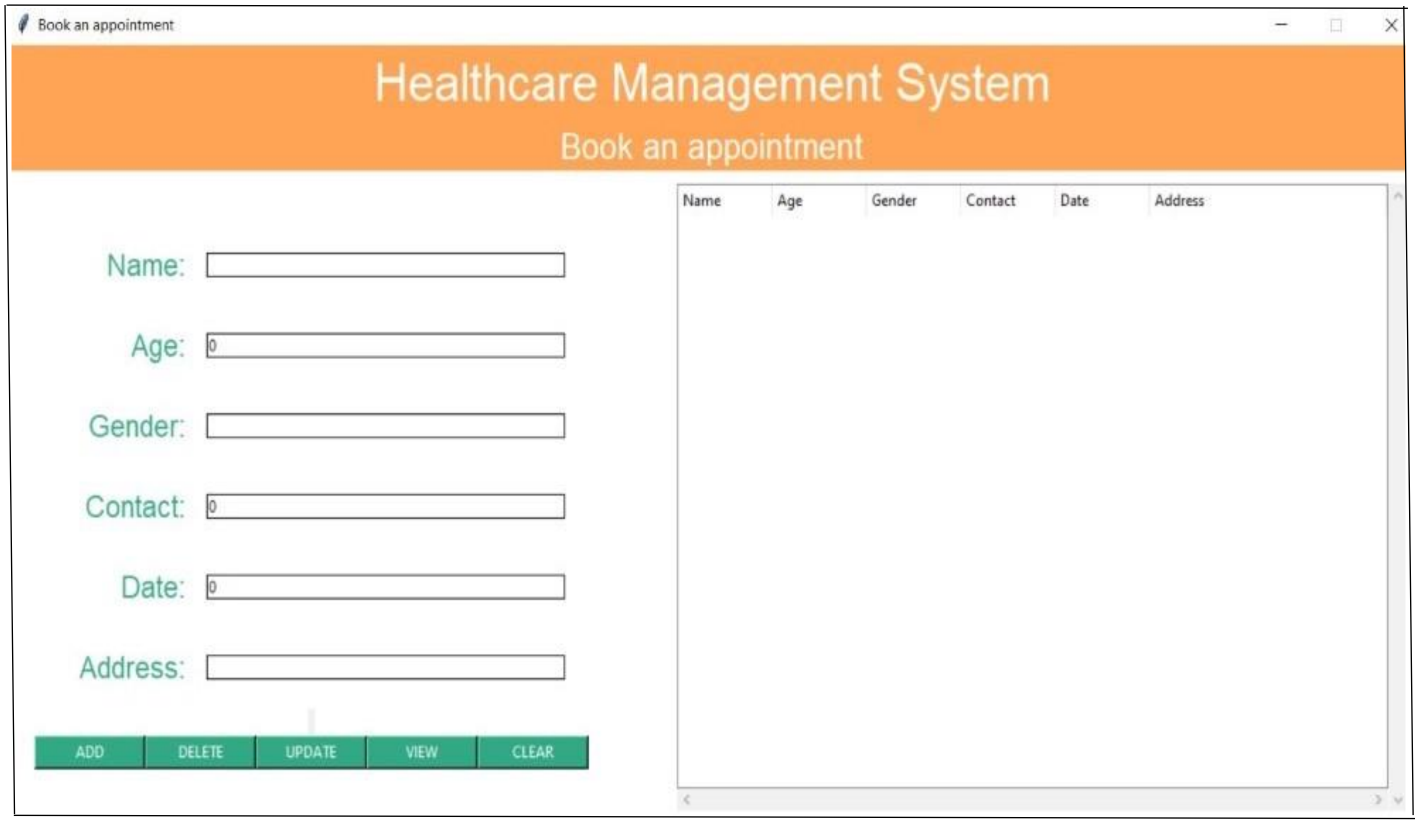
## 5.tkMessageBox -

The tkMessageBox module is used to display message boxes in your applications. This module provides a number of functions that you can use to display an appropriate message.

Some of these functions are showinfo, showwarning, showerror, askquestion, askokcancel, askyesno, and askretryignore.

# SCREENSHOT OF THE PORTAL



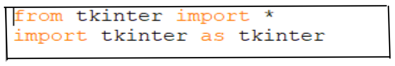


STEPS :

## [PAGE-1]

- **Importing Modules** –

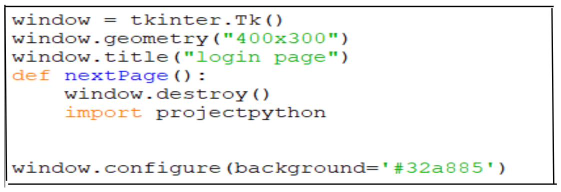
  Firstly in the setup if IDLE, we imported the tkinter module.



```
from tkinter import *
import tkinter as tkinter
```

- **Setting up the Main Frame and linking another page –**

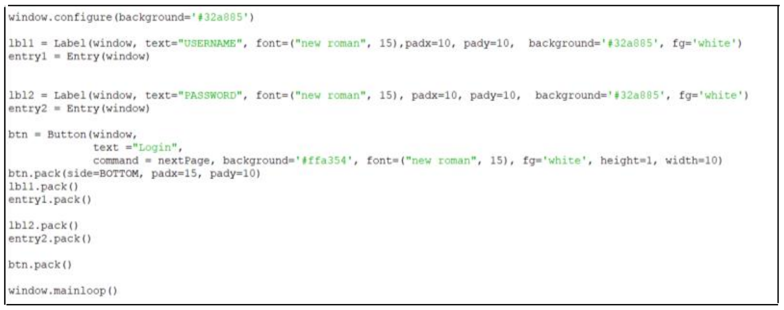
  After importing the modules, we will now then create the main frame for the application and linked the other page (projectpython) with the main login page.



```
window = tkinter.Tk()
window.geometry("400x300")
window.title("login page")
def nextPage():
    window.destroy()
    import projectpython


window.configure(background='#32a885')
```

- **Creating the Main Function and Layout –**

  In this we created the login page which consists of input of username and password from user to book an appointment.

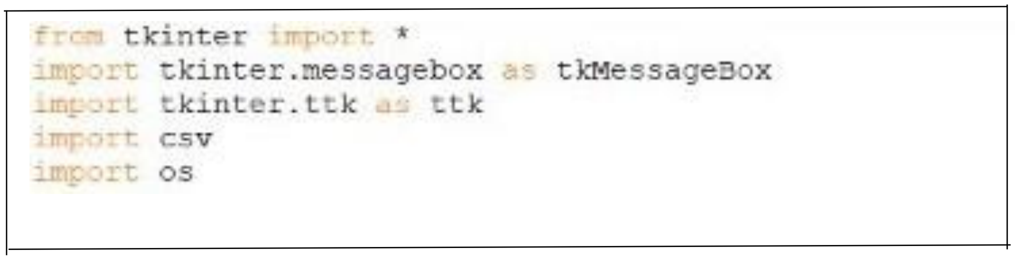


```
window.configure(background='#32a885')

lbl1 = Label(window, text="USERNAME", font=("new roman", 15),padx=10, pady=10,  background='#32a885', fg='white')
entry1 = Entry(window)


lbl2 = Label(window, text="PASSWORD", font=("new roman", 15), padx=10, pady=10,  background='#32a885', fg='white')
entry2 = Entry(window)

btn = Button(window,
             text ="Login",
             command = nextPage, background='#ffa354', font=("new roman", 15), fg='white', height=1, width=10)
btn.pack(side=BOTTOM, padx=15, pady=10)
lbl1.pack()
entry1.pack()

lbl2.pack()
entry2.pack()

btn.pack()

window.mainloop()
```

## [PAGE-2]

- **Importing Modules** –

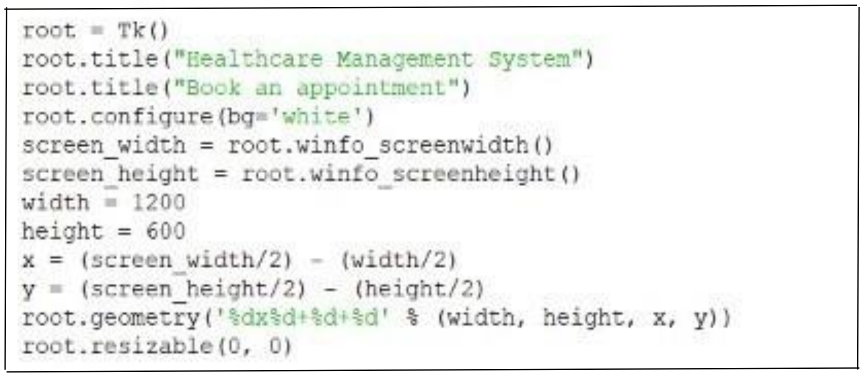
  Firstly, in the setup if IDLE, we imported the tkinter modules tkinter, tkinter.tk, csv, os.



```
from tkinter import *
import tkinter.messagebox as tkMessageBox
import tkinter.ttk as ttk
import csv
import os
```

- **Setting up the Main Frame –**

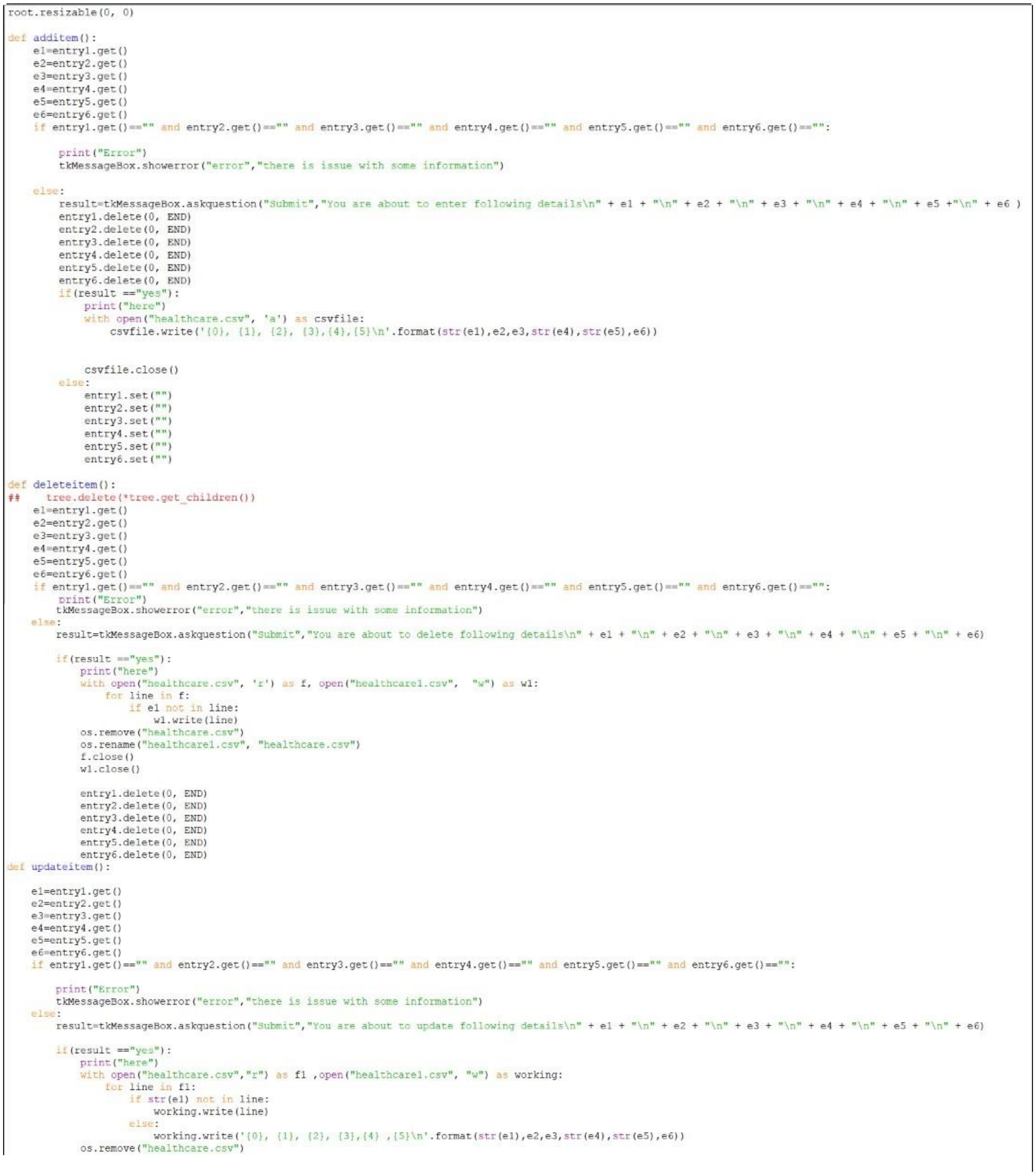
  After importing the modules, we will now then create the main frame for the application.



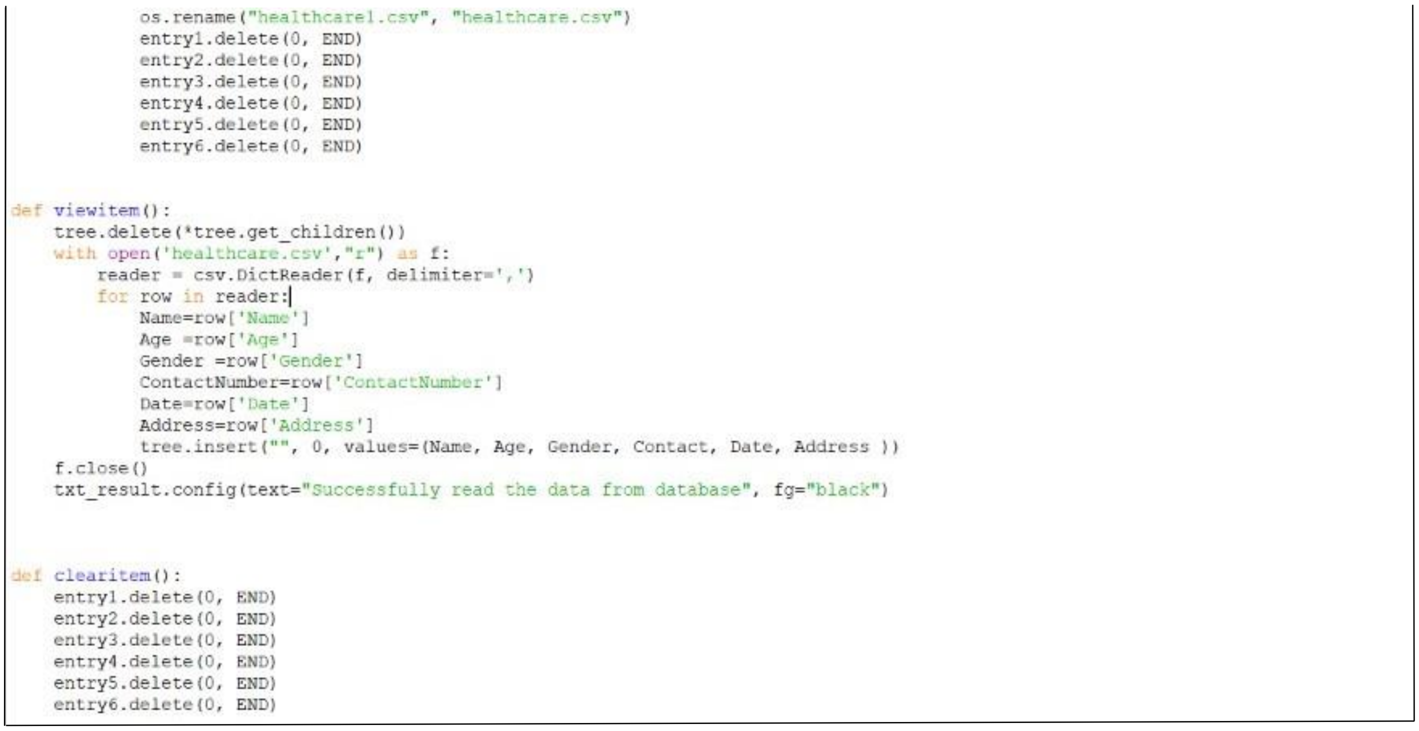
```
root = Tk()
root.title("Healthcare Management System")
root.title("Book an appointment")
root.configure(bg='white')
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
width = 1200
height = 600
x = (screen_width/2) - (width/2)
y = (screen_height/2) - (height/2)
root.geometry('%dx%d+%d+%d' % (width, height, x, y))
root.resizable(0, 0)
```

- **Creating the Main Function –**

  This is where the code that contains the main functions- ADD, DELETE, UPDATE, VIEW, CLEAR. This code will read the csv file then display all the data to the Tkinter TreeView.



```python
root.resizable(0, 0)

def additem():
    e1=entry1.get()
    e2=entry2.get()
    e3=entry3.get()
    e4=entry4.get()
    e5=entry5.get()
    e6=entry6.get()
    if entry1.get()=="" and entry2.get()=="" and entry3.get()=="" and entry4.get()=="" and entry5.get()=="" and entry6.get()=="":

        print("Error")
        tkMessageBox.showerror("error","there is issue with some information")

    else:
        result=tkMessageBox.askquestion("Submit","You are about to enter following details\n" + e1 + "\n" + e2 + "\n" + e3 + "\n" + e4 + "\n" + e5 +"\n" + e6 )
        entry1.delete(0, END)
        entry2.delete(0, END)
        entry3.delete(0, END)
        entry4.delete(0, END)
        entry5.delete(0, END)
        entry6.delete(0, END)
        if(result =="yes"):
            print("here")
            with open("healthcare.csv", 'a') as csvfile:
                csvfile.write('{0}, {1}, {2}, {3},{4},{5}\n'.format(str(e1),e2,e3,str(e4),str(e5),e6))


            csvfile.close()
        else:
            entry1.set("")
            entry2.set("")
            entry3.set("")
            entry4.set("")
            entry5.set("")
            entry6.set("")

def deleteitem():
##    tree.delete(*tree.get_children())
    e1=entry1.get()
    e2=entry2.get()
    e3=entry3.get()
    e4=entry4.get()
    e5=entry5.get()
    e6=entry6.get()
    if entry1.get()=="" and entry2.get()=="" and entry3.get()=="" and entry4.get()=="" and entry5.get()=="" and entry6.get()=="":
        print("Error")
        tkMessageBox.showerror("error","there is issue with some information")
    else:
        result=tkMessageBox.askquestion("Submit","You are about to delete following details\n" + e1 + "\n" + e2 + "\n" + e3 + "\n" + e4 + "\n" + e5 + "\n" + e6)

        if(result =="yes"):
            print("here")
            with open("healthcare.csv", 'r') as f, open("healthcare1.csv",  "w") as w1:
                for line in f:
                    if e1 not in line:
                        w1.write(line)
            os.remove("healthcare.csv")
            os.rename("healthcare1.csv", "healthcare.csv")
            f.close()
            w1.close()

            entry1.delete(0, END)
            entry2.delete(0, END)
            entry3.delete(0, END)
            entry4.delete(0, END)
            entry5.delete(0, END)
            entry6.delete(0, END)
def updateitem():

    e1=entry1.get()
    e2=entry2.get()
    e3=entry3.get()
    e4=entry4.get()
    e5=entry5.get()
    e6=entry6.get()
    if entry1.get()=="" and entry2.get()=="" and entry3.get()=="" and entry4.get()=="" and entry5.get()=="" and entry6.get()=="":

        print("Error")
        tkMessageBox.showerror("error","there is issue with some information")
    else:
        result=tkMessageBox.askquestion("Submit","You are about to update following details\n" + e1 + "\n" + e2 + "\n" + e3 + "\n" + e4 + "\n" + e5 + "\n" + e6)

        if(result =="yes"):
            print("here")
            with open("healthcare.csv","r") as f1 ,open("healthcare1.csv", "w") as working:
                for line in f1:
                    if str(e1) not in line:
                        working.write(line)
                    else:
                        working.write('{0}, {1}, {2}, {3},{4} ,{5}\n'.format(str(e1),e2,e3,str(e4),str(e5),e6))
            os.remove("healthcare.csv")
```



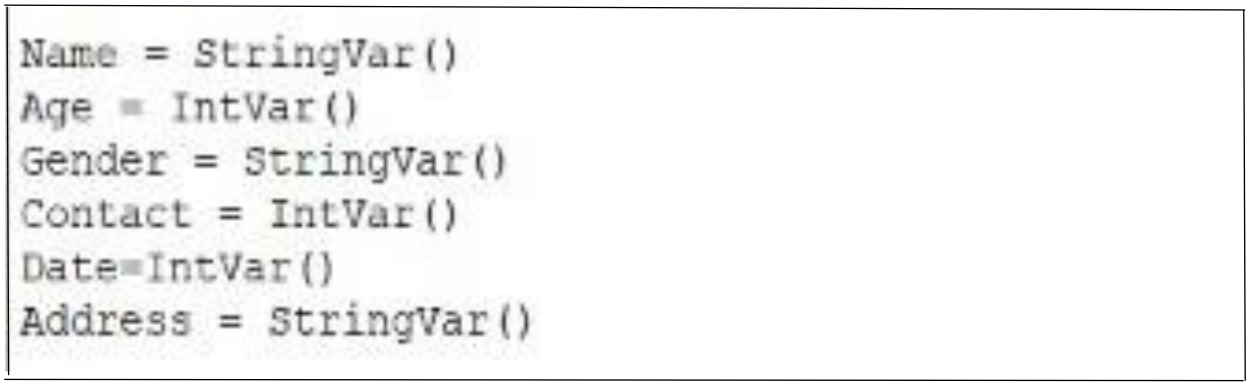
```
            os.rename("healthcare1.csv", "healthcare.csv")
            entry1.delete(0, END)
            entry2.delete(0, END)
            entry3.delete(0, END)
            entry4.delete(0, END)
            entry5.delete(0, END)
            entry6.delete(0, END)


def viewitem():
    tree.delete(*tree.get_children())
    with open('healthcare.csv',"r") as f:
        reader = csv.DictReader(f, delimiter=',')
        for row in reader:
            Name=row['Name']
            Age =row['Age']
            Gender =row['Gender']
            ContactNumber=row['ContactNumber']
            Date=row['Date']
            Address=row['Address']
            tree.insert("", 0, values=(Name, Age, Gender, Contact, Date, Address ))
    f.close()
    txt_result.config(text="Successfully read the data from database", fg="black")



def clearitem():
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)
    entry6.delete(0, END)
```

- **Tkinter Variables –**

  Tkinter supports some variables which are used to manipulate the values of Tkinter widgets. These variables work like normal variables.

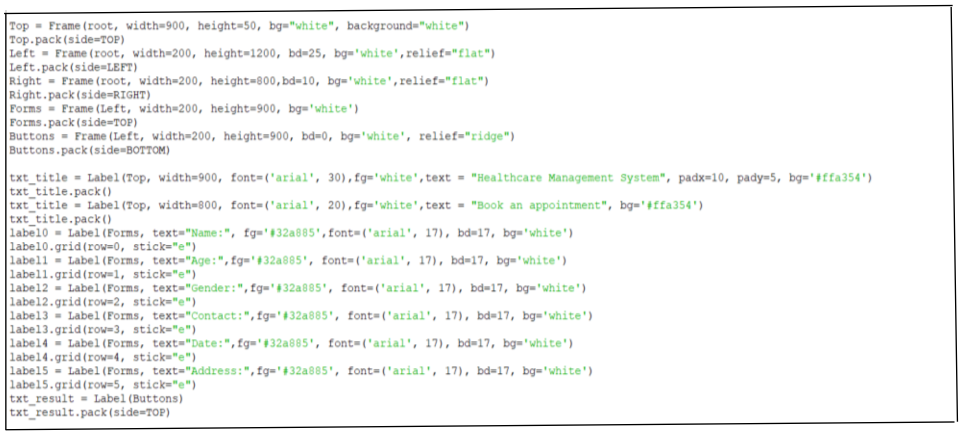
  1.StringVar()

  2. Intvar()



```
Name = StringVar()
Age = IntVar()
Gender = StringVar()
Contact = IntVar()
Date=IntVar()
Address = StringVar()
```

- **Layout –**

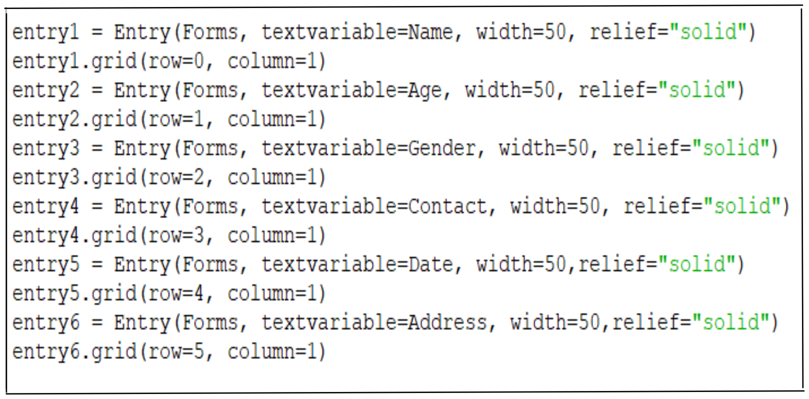


```
Top = Frame(root, width=900, height=50, bg="white", background="white")
Top.pack(side=TOP)
Left = Frame(root, width=200, height=1200, bd=25, bg='white',relief="flat")
Left.pack(side=LEFT)
Right = Frame(root, width=200, height=800,bd=10, bg='white',relief="flat")
Right.pack(side=RIGHT)
Forms = Frame(Left, width=200, height=900, bg='white')
Forms.pack(side=TOP)
Buttons = Frame(Left, width=200, height=900, bd=0, bg='white', relief="ridge")
Buttons.pack(side=BOTTOM)

txt_title = Label(Top, width=900, font=('arial', 30),fg='white',text = "Healthcare Management System", padx=10, pady=5, bg='#ffa354')
txt_title.pack()
txt_title = Label(Top, width=800, font=('arial', 20),fg='white',text = "Book an appointment", bg='#ffa354')
txt_title.pack()
label0 = Label(Forms, text="Name:", fg='#32a885',font=('arial', 17), bd=17, bg='white')
label0.grid(row=0, stick="e")
label1 = Label(Forms, text="Age:",fg='#32a885', font=('arial', 17), bd=17, bg='white')
label1.grid(row=1, stick="e")
label2 = Label(Forms, text="Gender:",fg='#32a885', font=('arial', 17), bd=17, bg='white')
label2.grid(row=2, stick="e")
label3 = Label(Forms, text="Contact:",fg='#32a885', font=('arial', 17), bd=17, bg='white')
label3.grid(row=3, stick="e")
label4 = Label(Forms, text="Date:",fg='#32a885', font=('arial', 17), bd=17, bg='white')
label4.grid(row=4, stick="e")
label5 = Label(Forms, text="Address:",fg='#32a885', font=('arial', 17), bd=17, bg='white')
label5.grid(row=5, stick="e")
txt_result = Label(Buttons)
txt_result.pack(side=TOP)
```

- **Entry Widget & Grid Layout–**

  Entry() widgets are the basic widgets of Tkinter, which is used to get input, i.e., text strings, from the user of an application. This widget allows the user to enter a single line of text.

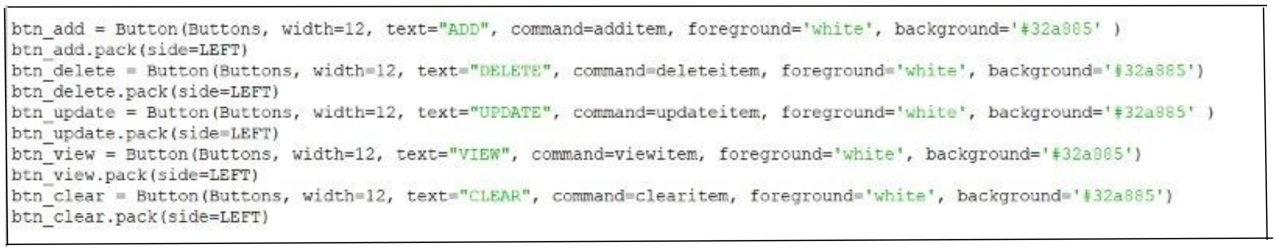
  To use the grid() method for placing the labels in the proper place by passing attributes row and column. If we don't pass anything in the grid() method, the default is 0 for both row and column.



```
entry1 = Entry(Forms, textvariable=Name, width=50, relief="solid")
entry1.grid(row=0, column=1)
entry2 = Entry(Forms, textvariable=Age, width=50, relief="solid")
entry2.grid(row=1, column=1)
entry3 = Entry(Forms, textvariable=Gender, width=50, relief="solid")
entry3.grid(row=2, column=1)
entry4 = Entry(Forms, textvariable=Contact, width=50, relief="solid")
entry4.grid(row=3, column=1)
entry5 = Entry(Forms, textvariable=Date, width=50,relief="solid")
entry5.grid(row=4, column=1)
entry6 = Entry(Forms, textvariable=Address, width=50,relief="solid")
entry6.grid(row=5, column=1)
```

- **Buttons :**

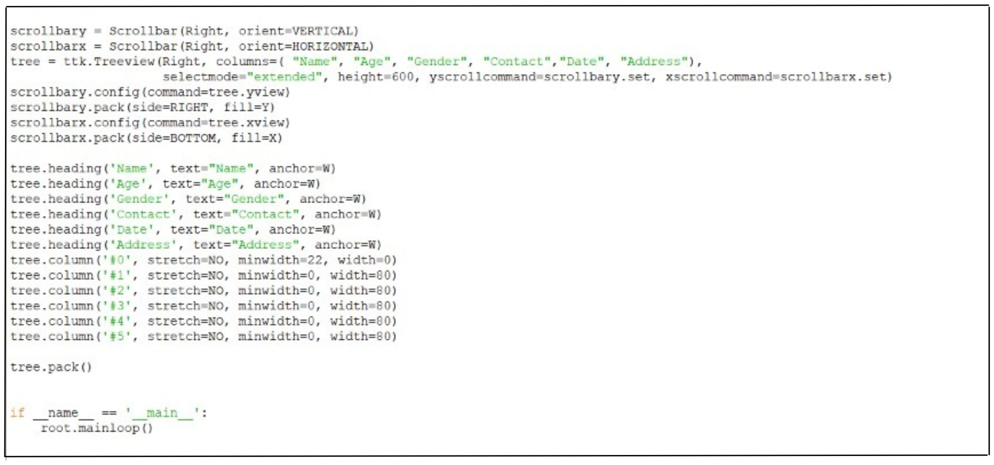
  The Button widget is used to add buttons in a Python application. These buttons can display text or images that convey the purpose of the buttons. You can attach a function or a method to a button which is called automatically when you click the button.



```
btn_add = Button(Buttons, width=12, text="ADD", command=additem, foreground='white', background='#32a885' )
btn_add.pack(side=LEFT)
btn_delete = Button(Buttons, width=12, text="DELETE", command=deleteitem, foreground='white', background='#32a885')
btn_delete.pack(side=LEFT)
btn_update = Button(Buttons, width=12, text="UPDATE", command=updateitem, foreground='white', background='#32a885' )
btn_update.pack(side=LEFT)
btn_view = Button(Buttons, width=12, text="VIEW", command=viewitem, foreground='white', background='#32a885')
btn_view.pack(side=LEFT)
btn_clear = Button(Buttons, width=12, text="CLEAR", command=clearitem, foreground='white', background='#32a885')
btn_clear.pack(side=LEFT)
```

- **Treeview Scrollbar –**

  When a scrollbar uses treeview widgets, then that type of scrollbar is called as treeview scrollbar. Where, a treeview widget is helpful in displaying more than one feature of every item listed in the tree to the right side of the tree in the form of columns. However, it can be implemented using tkinter in python with the help of some widgets and geometry management methods as supported by tkinter.



```
scrollbary = Scrollbar(Right, orient=VERTICAL)
scrollbarx = Scrollbar(Right, orient=HORIZONTAL)
tree = ttk.Treeview(Right, columns=( "Name", "Age", "Gender", "Contact","Date", "Address"),
                    selectmode="extended", height=600, yscrollcommand=scrollbary.set, xscrollcommand=scrollbarx.set)
scrollbary.config(command=tree.yview)
scrollbary.pack(side=RIGHT, fill=Y)
scrollbarx.config(command=tree.xview)
scrollbarx.pack(side=BOTTOM, fill=X)

tree.heading('Name', text="Name", anchor=W)
tree.heading('Age', text="Age", anchor=W)
tree.heading('Gender', text="Gender", anchor=W)
tree.heading('Contact', text="Contact", anchor=W)
tree.heading('Date', text="Date", anchor=W)
tree.heading('Address', text="Address", anchor=W)
tree.column('#0', stretch=NO, minwidth=22, width=0)
tree.column('#1', stretch=NO, minwidth=0, width=80)
tree.column('#2', stretch=NO, minwidth=0, width=80)
tree.column('#3', stretch=NO, minwidth=0, width=80)
tree.column('#4', stretch=NO, minwidth=0, width=80)
tree.column('#5', stretch=NO, minwidth=0, width=80)

tree.pack()


if __name__ == '__main__':
    root.mainloop()
```

## CONCLUSION

It is our team’s hope that this document will be of huge help with understanding of our little project of healthcare management system. We have used basic modules and libraries with the help of python and Tkinter.

## REFERENCES

To conduct this project the following tools have been used:

● Python IDLE

Reference website:

(Geeks For Geeks) - https://www.geeksforgeeks.org/python-gui-tkinter/

We have used this site for our basic knowledge gain of the methods that will be used in the project.