##Min Shwe Maung Htet  
  
##Python 3  
##Project Name: Clinic Portal System Project

Brief Explanation of the Client GUI Application Project

1. This Project has three files which are HomePage.py, InsertWindow.py and SearchDeleteWindow.py. The user needs to run HomePage.py as a main file.
2. After running the main file, it will show the first main home screen where user can see personal information about the project, use the function applied in the project and also use the services shown in the application.
3. If the user hits the Insert Button, it will pop up another window(Insert Data) where user call all fields of the client or patient as need. There are some restrictions about the program, the patient ID only must be numeric value, the email id must provide correct domain form and the phone field must have 10 digits as I make as an US number format. The user can finish the form by clicking insert button in the Insert Window or the user can clear out the entry form by clicking Reset button.
4. If the user hits the Search Button, the user can find the previous information of the individual data in the database by putting client’s id key.
5. If the user hits the Delete Button, it will delete individual’s personal information in the database by putting client’s id key. Note, it will delete as long as the user the clicks the Delete button. The user can check it by clicking the Display button in the main window whether the information exists in the database or not.
6. Finally, there is the Exit button and it basically destroys or closes the main window.
7. There is one special function in Menu bar section, if the user hits the Rate Me button, the user can rate on the scale value and it will store the value in rate.txt file.

HomePage.py

# Importing modules  
*from* Client\_GUI.SearchDeleteWindow *import* \*  
*from* Client\_GUI.InsertWindow *import* \*  
*from* tkinter *import* \*  
*import* tkinter.messagebox *as* tmsg  
*import* tkinter.ttk  
*import* tkinter.messagebox  
  
# This is the first page to interact with the user  
*class* HomePage:  
 # Constructing homepage window class  
 *def \_\_init\_\_*(self):  
 self.homePageWindow = tkinter.Tk()  
 self.homePageWindow.wm\_title("Project GUI")  
  
 # The following functions perform when the user click the menu in menubar choice  
 *def* exit():  
 a = tmsg.askquestion("Exit?", "Are you sure sir?")  
 *if* a == "yes":  
 b = tmsg.askquestion("Again!", "Think again sir!")  
 *if* b == "yes":  
 self.homePageWindow.destroy()  
  
 *def* help():  
 tmsg.showinfo("Help", "Contact Min for more information, Thank You :)")  
  
 *def* feedback():  
 l = tmsg.askquestion("Experience", "Was your experience good?")  
  
 *if* l == "yes":  
 tmsg.showinfo("Yes!", "Good, now go to our official website to rate us!")  
 *else*:  
 tmsg.showinfo("No!", "Sorry to hear that!")  
  
 *def* update():  
 msg = tmsg.askretrycancel("Sorry!", "Server is busy, so update is not available")  
 *if* msg:  
 tmsg.showinfo("Sorry", "Please try again later, update is not available!")  
  
 # This will pop up another window when the user click the Rate Me button  
 *def* RateWindow():  
 root = Tk()  
 root.geometry("400x100")  
 root.minsize(400, 100)  
 root.maxsize(400, 100)  
 root.title("Please rate Min!")  
  
 *def* rate():  
 tmsg.showinfo("Feedback", "Thank You for Rating Me")  
 *with* open("rate.txt", "a") *as* f:  
 f.write(f"Person rated = {Rate.get()}\n")  
 root.destroy()  
 Rate = Scale(root, from\_=0, to=50, tickinterval=10, orient=HORIZONTAL)  
 Rate.pack()  
 b0 = Button(root, text="Rate", bg="yellow", command=rate)  
 b0.pack()  
  
  
 # Creating Menubar  
 menubar = Menu(self.homePageWindow)  
 # Adding three menus in menu bar and related commands to each menu  
 menu1 = Menu(menubar, tearoff=0)  
 menubar.add\_cascade(label='Available Service', menu=menu1)  
 menu1.add\_command(label='Help', command=help)  
 menu1.add\_command(label='Feedback', command=feedback)  
 menu1.add\_command(label='Update', command=update)  
 menu1.add\_separator()  
 menu1.add\_command(label='Exit', command=exit)  
  
 menu2 = Menu(menubar, tearoff=0)  
 menubar.add\_cascade(label='Rate', menu=menu2)  
 menu2.add\_command(label="Rate Me", command=RateWindow)  
  
 menu3 = Menu(menubar, tearoff=0)  
 menubar.add\_cascade(label='View', menu=menu3)  
 show\_all = tkinter.BooleanVar()  
 show\_all.set(*True*)  
 show\_done = tkinter.BooleanVar()  
 show\_not\_done = tkinter.BooleanVar()  
 menu3.add\_checkbutton(label="Show All", onvalue=1, offvalue=0, variable=show\_all)  
 menu3.add\_checkbutton(label="Show Done", onvalue=1, offvalue=0, variable=show\_done)  
 menu3.add\_checkbutton(label="Show Not Done", onvalue=1, offvalue=0, variable=show\_not\_done)  
  
 # display Menu  
 self.homePageWindow.config(menu=menubar)  
  
  
 # Adding my personal information at the top of the project using Label and Frame  
 tkinter.Label(self.homePageWindow, text="Name: Min Shwe Maung Htet\n"  
 "Course: Python 2 (18516)\n"  
 "Professor: Javad Ameri\n"  
 "Topic: Clinic Portal System Project",  
 width=100).grid(pady=20, column=1, row=1)  
 instruction\_frame = Frame(self.homePageWindow, width=150).grid(pady=20, column=1, row=2)  
 Label(instruction\_frame, text="You can do the following services "  
 "in this application").grid(pady=20, column=1, row=3)  
  
  
 # Creating five buttons in homepage for interaction with the program  
 tkinter.Button(self.homePageWindow, width=20, text="Insert", command=self.Insert).grid(pady=15, column=1, row=5)  
 tkinter.Button(self.homePageWindow, width=20, text="Search", command=self.Search).grid(pady=15, column=1, row=6)  
 tkinter.Button(self.homePageWindow, width=20, text="Delete", command=self.Delete).grid(pady=15, column=1, row=7)  
 tkinter.Button(self.homePageWindow, width=20, text="Display", command=self.Display).grid(pady=15, column=1,  
 row=8)  
 tkinter.Button(self.homePageWindow, width=20, text="Exit", command=self.homePageWindow.destroy).grid(pady=15,  
 column=1,row=9)  
  
  
 # Creating main loop  
 self.homePageWindow.mainloop()  
  
  
 # This each function performs differently specifically when the button clicks  
 *def* Insert(self):  
 self.insertWindow = InsertWindow()  
  
 *def* Search(self):  
 self.searchWindow = SearchDeleteWindow("Search")  
  
 *def* Delete(self):  
 self.deleteWindow = SearchDeleteWindow("Delete")  
  
 *def* Display(self):  
 self.database = Database()  
 self.data = self.database.Display()  
 self.displayWindow = DatabaseView(self.data)  
  
  
# Creating an instance of the HomePage class  
homePage = HomePage()

InsertWindow.py

# Importing modules and class  
*from* Client\_GUI.SearchDeleteWindow *import* \*  
*import* tkinter.messagebox *as* tmsg  
*import* tkinter.ttk  
*import* tkinter.messagebox  
*import* sqlite3  
  
# It is an insert class to get all information from the beginning  
*class* InsertWindow:  
 *def \_\_init\_\_*(self):  
 self.window = tkinter.Tk()  
 self.window.wm\_title("Insert data")  
  
 # Initializing all the variables  
 self.id = tkinter.StringVar()  
 self.fName = tkinter.StringVar()  
 self.lName = tkinter.StringVar()  
 self.address = tkinter.StringVar()  
 self.phone = tkinter.StringVar()  
 self.email = tkinter.StringVar()  
 self.history = tkinter.StringVar()  
 self.doctor = tkinter.StringVar()  
 self.genderButton = tkinter.IntVar()  
 self.dateList = list(range(1, 32))  
 self.monthList = ["January", "February", "March", "April", "May", "June", "July",  
 "August", "September", "October", "November", "December"]  
 self.yearList = list(range(1900, 2020))  
 self.bloodGroupList = ["A+", "A-", "B+", "B-", "O+", "O-", "AB+", "AB-"]  
  
 # Labels of the data  
 tkinter.Label(self.window, text = "Patient ID", width = 25).grid(pady = 5, column = 1, row = 1)  
 tkinter.Label(self.window, text = "First Name", width = 25).grid(pady = 5, column = 1, row = 2)  
 tkinter.Label(self.window, text = "Last Name", width = 25).grid(pady = 5, column = 1, row = 3)  
 tkinter.Label(self.window, text = "D.O.B", width = 25).grid(pady = 5, column = 1, row = 4)  
 tkinter.Label(self.window, text = "M.O.B", width = 25).grid(pady = 5, column = 1, row = 5)  
 tkinter.Label(self.window, text = "Y.O.B", width = 25).grid(pady = 5, column = 1, row = 6)  
 tkinter.Label(self.window, text = "Gender", width = 25).grid(pady = 5, column = 1, row = 7)  
 tkinter.Label(self.window, text = "Home Address", width = 25).grid(pady = 5, column = 1, row = 8)  
 tkinter.Label(self.window, text = "Phone Number", width = 25).grid(pady = 5, column = 1, row = 9)  
 tkinter.Label(self.window, text = "Email ID", width = 25).grid(pady = 5, column = 1, row = 10)  
 tkinter.Label(self.window, text = "Blood Group", width = 25).grid(pady = 5, column = 1, row = 11)  
 tkinter.Label(self.window, text = "Patient History", width = 25).grid(pady = 5, column = 1, row = 12)  
 tkinter.Label(self.window, text = "Doctor", width = 25).grid(pady = 5, column = 1, row = 13)  
  
 # Fields  
 # Entry widgets  
 self.idEntry = tkinter.Entry(self.window, width = 25, textvariable = self.id)  
 self.fNameEntry = tkinter.Entry(self.window, width = 25, textvariable = self.fName)  
 self.lNameEntry = tkinter.Entry(self.window, width = 25, textvariable = self.lName)  
 self.addressEntry = tkinter.Entry(self.window, width = 25, textvariable = self.address)  
 self.phoneEntry = tkinter.Entry(self.window, width = 25, textvariable = self.phone)  
 self.emailEntry = tkinter.Entry(self.window, width = 25, textvariable = self.email)  
 self.historyEntry = tkinter.Entry(self.window, width = 25, textvariable = self.history)  
 self.doctorEntry = tkinter.Entry(self.window, width = 25, textvariable = self.doctor)  
  
 self.idEntry.grid(pady = 5, column = 3, row = 1)  
 self.fNameEntry.grid(pady = 5, column = 3, row = 2)  
 self.lNameEntry.grid(pady = 5, column = 3, row = 3)  
 self.addressEntry.grid(pady = 5, column = 3, row = 8)  
 self.phoneEntry.grid(pady = 5, column = 3, row = 9)  
 self.emailEntry.grid(pady = 5, column = 3, row = 10)  
 self.historyEntry.grid(pady = 5, column = 3, row = 12)  
 self.doctorEntry.grid(pady = 5, column = 3, row = 13)  
  
 # Combobox widgets  
 self.dobBox = tkinter.ttk.Combobox(self.window, values = self.dateList, width = 20)  
 self.mobBox = tkinter.ttk.Combobox(self.window, values = self.monthList, width = 20)  
 self.yobBox = tkinter.ttk.Combobox(self.window, values = self.yearList, width = 20)  
  
 # Radio button for male and female button  
 self.genderButton.set(1)  
 self.male\_button = tkinter.ttk.Radiobutton(self.window, text="Male", variable=self.genderButton,  
 value = 1, width = 20)  
 self.female\_button = tkinter.ttk.Radiobutton(self.window, text="Female", variable=self.genderButton,  
 value = 2, width=20)  
 self.bloodGroupBox = tkinter.ttk.Combobox(self.window, values = self.bloodGroupList, width = 20)  
  
  
 # Putting all combobox and radiobutton in grid format  
 self.dobBox.grid(pady = 5, column = 3, row = 4)  
 self.mobBox.grid(pady = 5, column = 3, row = 5)  
 self.yobBox.grid(pady = 5, column = 3, row = 6)  
 self.male\_button.grid(padx = 5, column = 3, row = 7)  
 self.female\_button.grid(padx=5, column=4, row=7)  
 self.bloodGroupBox.grid(pady = 5, column = 3, row = 11)  
  
  
 # Button widgets in the very bottom of the window  
 tkinter.Button(self.window, width = 20, text = "Insert", command = self.Insert)\  
 .grid(pady = 15, padx = 5, column = 1, row = 14)  
 tkinter.Button(self.window, width = 20, text = "Reset", command = self.Reset)\  
 .grid(pady = 15, padx = 5, column = 2, row = 14)  
 tkinter.Button(self.window, width = 20, text = "Close", command = self.window.destroy)\  
 .grid(pady = 15, padx = 5, column = 3, row = 14)  
  
 # Creating mainloop  
 self.window.mainloop()  
  
 # This insert function interacts with the class to get input valid data  
 *def* Insert(self):  
 self.values = Values()  
 self.database = Database()  
 self.test = self.values.Validate(self.idEntry.get(), self.fNameEntry.get(), self.lNameEntry.get(),  
 self.phoneEntry.get(), self.emailEntry.get(), self.historyEntry.get(),  
 self.doctorEntry.get())  
 *if* (self.test == "SUCCESS"):  
 self.database.Insert(self.idEntry.get(), self.fNameEntry.get(), self.lNameEntry.get(),  
 self.dobBox.get(), self.mobBox.get(), self.yobBox.get(),  
 self.genderButton.get(), self.addressEntry.get(), self.phoneEntry.get(),  
 self.emailEntry.get(), self.bloodGroupBox.get(), self.historyEntry.get(),  
 self.doctorEntry.get())  
 tkinter.messagebox.showinfo("Inserted data", "Successfully inserted the above data in the database")  
 *else*:  
 self.valueErrorMessage = "Invalid input in field " + self.test  
 tkinter.messagebox.showerror("Value Error", self.valueErrorMessage)  
  
 # This function clear out all the input data in insert field  
 *def* Reset(self):  
 self.idEntry.delete(0, tkinter.END)  
 self.fNameEntry.delete(0, tkinter.END)  
 self.lNameEntry.delete(0, tkinter.END)  
 self.dobBox.set("")  
 self.mobBox.set("")  
 self.yobBox.set("")  
 self.genderButton.set("")  
 self.addressEntry.delete(0, tkinter.END)  
 self.phoneEntry.delete(0, tkinter.END)  
 self.emailEntry.delete(0, tkinter.END)  
 self.bloodGroupBox.set("")  
 self.historyEntry.delete(0, tkinter.END)  
 self.doctorEntry.delete(0, tkinter.END)  
  
  
# This value class gets input data with valid information  
*class* Values:  
 *def* Validate(self, *id*, *fName*, *lName*, *phone*, *email*, *history*, *doctor*):  
 *if not* (*id*.isdigit()):  
 *return* "id"  
 *elif not* (*fName*.isalpha()):  
 *return* "fName"  
 *elif not* (*lName*.isalpha()):  
 *return* "lName"  
 *elif not* (*phone*.isdigit() *and* (len(*phone*) == 10)):  
 *return* "phone"  
 *elif not* (*email*.count("@") == 1 *and email*.count(".") > 0):  
 *return* "email"  
 *elif not* (*history*.isalpha()):  
 *return* "history"  
 *elif not* (*doctor*.isalpha()):  
 *return* "doctor"  
 *else*:  
 *return* "SUCCESS"  
  
*class* Database:  
 *def \_\_init\_\_*(self):  
 self.dbConnection = sqlite3.connect("dbFile.db")  
 self.dbCursor = self.dbConnection.cursor()  
 self.dbCursor.execute("CREATE TABLE IF NOT EXISTS patient\_info (id PRIMARYKEY text, fName text, lName text, "  
 "dob text, mob text, yob text, gender text, address text, phone text, email text, "  
 "bloodGroup text, history text, doctor text)")  
  
 *def \_\_del\_\_*(self):  
 self.dbCursor.close()  
 self.dbConnection.close()  
  
 # This insert function adds information to the database  
 *def* Insert(self, *id*, *fName*, *lName*, *dob*, *mob*, *yob*, *gender*, *address*,  
 *phone*, *email*, *bloodGroup*, *history*, *doctor*):  
 self.dbCursor.execute("INSERT INTO patient\_info VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)",  
 (*id*, *fName*, *lName*, *dob*, *mob*, *yob*, *gender*, *address*,  
 *phone*, *email*, *bloodGroup*, *history*, *doctor*))  
 self.dbConnection.commit()  
  
 # This update function updates information to the database  
 *def* Update(self, *fName*, *lName*, *dob*, *mob*, *yob*, *gender*, *address*,  
 *phone*, *email*, *bloodGroup*, *history*, *doctor*, *id*):  
 self.dbCursor.execute("UPDATE patient\_info SET fName = ?, lName = ?, dob = ?, mob = ?, yob = ?, gender = ?, "  
 "address = ?, phone = ?, email = ?, bloodGroup = ?, history = ?, doctor = ? WHERE id = ?",  
 (*fName*, *lName*, *dob*, *mob*, *yob*, *gender*, *address*,  
 *phone*, *email*, *bloodGroup*, *history*, *doctor*, *id*))  
 self.dbConnection.commit()  
  
 # This function performs to get information with patient id key  
 *def* Search(self, *id*):  
 self.dbCursor.execute("SELECT \* FROM patient\_info WHERE id = ?", (*id*, ))  
 searchResults = self.dbCursor.fetchall()  
 *return* searchResults  
  
 # This function performs to delete information with patient id key  
 *def* Delete(self, *id*):  
 self.dbCursor.execute("DELETE FROM patient\_info WHERE id = ?", (*id*, ))  
 self.dbConnection.commit()  
  
 # This function shows all information of the patient data  
 *def* Display(self):  
 self.dbCursor.execute("SELECT \* FROM patient\_info")  
 records = self.dbCursor.fetchall()  
 *return* records

SearchDeleteWindow.py

# Importing modules and class  
*from* Client\_GUI.InsertWindow *import* \*  
*import* tkinter.ttk  
*import* tkinter.messagebox  
  
# This class performs for the search and delete button when the user clicks  
*class* SearchDeleteWindow:  
 *def \_\_init\_\_*(self, *task*):  
 window = tkinter.Tk()  
 window.wm\_title(*task* + " data")  
  
 # Initializing all the variables  
 self.id = tkinter.StringVar()  
 self.fName = tkinter.StringVar()  
 self.lName = tkinter.StringVar()  
 self.heading = "Please enter Patient ID to " + *task* # Labels  
 tkinter.Label(window, text=self.heading, width=50).grid(pady=20, row=1)  
 tkinter.Label(window, text="Patient ID", width=10).grid(pady=5, row=2)  
  
 # Entry widgets  
 self.idEntry = tkinter.Entry(window, width=5, textvariable=self.id)  
  
 self.idEntry.grid(pady=5, row=3)  
  
 # Button widgets  
 *if* (*task* == "Search"):  
 tkinter.Button(window, width=20, text=*task*, command=self.Search).grid(pady=15, padx=5, column=1, row=14)  
 *elif* (*task* == "Delete"):  
 tkinter.Button(window, width=20, text=*task*, command=self.Delete).grid(pady=15, padx=5, column=1, row=14)  
  
 *def* Search(self):  
 self.database = Database()  
 self.data = self.database.Search(self.idEntry.get())  
 self.databaseView = DatabaseView(self.data)  
  
 *def* Delete(self):  
 self.database = Database()  
 self.database.Delete(self.idEntry.get())  
  
  
# This class stores and shows all the information that the user has given  
*class* DatabaseView:  
 *def \_\_init\_\_*(self, *data*):  
 self.databaseViewWindow = tkinter.Tk()  
 self.databaseViewWindow.wm\_title("Database View")  
  
 # Label widgets  
 tkinter.Label(self.databaseViewWindow, text = "Database View Window", width = 25).grid(pady = 5, column = 1, row = 1)  
  
 self.databaseView = tkinter.ttk.Treeview(self.databaseViewWindow)  
 self.databaseView.grid(pady = 5, column = 1, row = 2)  
 self.databaseView["show"] = "headings"  
 self.databaseView["columns"] = ("id", "fName", "lName", "dob", "mob", "yob", "gender", "address",  
 "phone", "email", "bloodGroup", "history", "doctor")  
  
 # Treeview column headings  
 self.databaseView.heading("id", text = "ID")  
 self.databaseView.heading("fName", text = "First Name")  
 self.databaseView.heading("lName", text = "Last Name")  
 self.databaseView.heading("dob", text = "D.O.B")  
 self.databaseView.heading("mob", text = "M.O.B")  
 self.databaseView.heading("yob", text = "Y.O.B")  
 self.databaseView.heading("gender", text = "Gender")  
 self.databaseView.heading("address", text = "Home Address")  
 self.databaseView.heading("phone", text = "Phone Number")  
 self.databaseView.heading("email", text = "Email ID")  
 self.databaseView.heading("bloodGroup", text = "Blood Group")  
 self.databaseView.heading("history", text = "History")  
 self.databaseView.heading("doctor", text = "Doctor")  
  
 # Treeview columns  
 self.databaseView.column("id", width = 40)  
 self.databaseView.column("fName", width = 100)  
 self.databaseView.column("lName", width = 100)  
 self.databaseView.column("dob", width = 60)  
 self.databaseView.column("mob", width = 60)  
 self.databaseView.column("yob", width = 60)  
 self.databaseView.column("gender", width = 60)  
 self.databaseView.column("address", width = 200)  
 self.databaseView.column("phone", width = 100)  
 self.databaseView.column("email", width = 200)  
 self.databaseView.column("bloodGroup", width = 100)  
 self.databaseView.column("history", width = 100)  
 self.databaseView.column("doctor", width = 100)  
  
 # Retrieve information in data file  
 *for* record *in data*:  
 self.databaseView.insert('', 'end', values=(record))  
  
 # Creating mainloop  
 self.databaseViewWindow.mainloop()