```
from tkinter import *
import tkinter.ttk as ttk
import tkinter.messagebox as tkMessageBox
import sqlite3


#function to define database
def Database():
    global conn, cursor
    #creating contact database
    conn = sqlite3.connect("contact.db")
    cursor = conn.cursor()
    #creating REGISTRATION table
    cursor.execute(
        "CREATE TABLE IF NOT EXISTS REGISTRATION (RID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, FNAME TEXT, LNAME TEXT, GENDER TEXT, ADDRESS TEXT, CONTACT TEXT)")


#defining function for creating GUI Layout
def DisplayForm():
    #creating window
    display_screen = Tk()
    #setting width and height for window
    display_screen.geometry("900x400")
    #setting title for window
    display_screen.title("SM INFOTECH")
    global tree
    global SEARCH
```

```
    global fname,lname,gender,address,contact
    SEARCH = StringVar()
    fname = StringVar()
    lname = StringVar()
    gender = StringVar()
    address = StringVar()
    contact = StringVar()
    #creating frames for layout
    #topview frame for heading
    TopViewForm = Frame(display_screen, width=600, bd=1, relief=SOLID)
    TopViewForm.pack(side=TOP, fill=X)
    #first left frame for registration from
    LFrom = Frame(display_screen, width="350",bg="#15244C")
    LFrom.pack(side=LEFT, fill=Y)
    #seconf left frame for search form
    LeftViewForm = Frame(display_screen, width=500,bg="#0B4670")
    LeftViewForm.pack(side=LEFT, fill=Y)
    #mid frame for displaying lnames record
    MidViewForm = Frame(display_screen, width=600)
    MidViewForm.pack(side=RIGHT)
    #label for heading
    lbl_text = Label(TopViewForm, text="Contact Management System", font=('verdana', 18),
width=600,bg="cyan")
    lbl_text.pack(fill=X)
    #creating registration form in first left frame
    Label(LFrom, text="First Name  ", font=("Arial", 12),bg="#15244C",fg="white").pack(side=TOP)
```

```
    Entry(LFrom,font=("Arial",10,"bold"),textvariable=fname).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Last Name ", font=("Arial", 12),bg="#15244C",fg="white").pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=lname).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Gender ", font=("Arial", 12),bg="#15244C",fg="white").pack(side=TOP)
    #Entry(LFrom, font=("Arial", 10, "bold"),textvariable=gender).pack(side=TOP, padx=10, fill=X)
    gender.set("Select Gender")
    content={'Male','Female'}
    OptionMenu(LFrom,gender,*content).pack(side=TOP, padx=10, fill=X)


    Label(LFrom, text="Address ", font=("Arial", 12),bg="#15244C",fg="white").pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=address).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Contact ", font=("Arial", 12),bg="#15244C",fg="white").pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=contact).pack(side=TOP, padx=10, fill=X)
    Button(LFrom,text="Submit",font=("Arial", 10,
"bold"),command=register,bg="#15244C",fg="white").pack(side=TOP, padx=10,pady=5, fill=X)


    #creating search label and entry in second frame
    lbl_txtsearch = Label(LeftViewForm, text="Enter fname to Search", font=('verdana',
10),bg="#0B4670")
    lbl_txtsearch.pack()
    #creating search entry
    search = Entry(LeftViewForm, textvariable=SEARCH, font=('verdana', 15), width=10)
    search.pack(side=TOP, padx=10, fill=X)
    #creating search button
    btn_search = Button(LeftViewForm, text="Search", command=SearchRecord,bg="cyan")
```

```
    btn_search.pack(side=TOP, padx=10, pady=10, fill=X)
    #creating view button
    btn_view = Button(LeftViewForm, text="View All", command=DisplayData,bg="cyan")
    btn_view.pack(side=TOP, padx=10, pady=10, fill=X)
    #creating reset button
    btn_reset = Button(LeftViewForm, text="Reset", command=Reset,bg="cyan")
    btn_reset.pack(side=TOP, padx=10, pady=10, fill=X)
    #creating delete button
    btn_delete = Button(LeftViewForm, text="Delete", command=Delete,bg="cyan")
    btn_delete.pack(side=TOP, padx=10, pady=10, fill=X)
    #create update button
    btn_delete = Button(LeftViewForm, text="Update", command=Update,bg="cyan")
    btn_delete.pack(side=TOP, padx=10, pady=10, fill=X)
    #setting scrollbar
    scrollbarx = Scrollbar(MidViewForm, orient=HORIZONTAL)
    scrollbary = Scrollbar(MidViewForm, orient=VERTICAL)
    tree = ttk.Treeview(MidViewForm,columns=("Student Id", "Name", "Contact",
"Email","Rollno","Branch"),
                        selectmode="extended", height=100, 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)
    #setting headings for the columns
    tree.heading('Student Id', text="Id", anchor=W)
```

```
    tree.heading('Name', text="FirstName", anchor=W)
    tree.heading('Contact', text="LastName", anchor=W)
    tree.heading('Email', text="Gender", anchor=W)
    tree.heading('Rollno', text="Address", anchor=W)
    tree.heading('Branch', text="Contact", anchor=W)
    #setting width of the columns
    tree.column('#0', stretch=NO, minwidth=0, width=0)
    tree.column('#1', stretch=NO, minwidth=0, width=100)
    tree.column('#2', stretch=NO, minwidth=0, width=150)
    tree.column('#3', stretch=NO, minwidth=0, width=80)
    tree.column('#4', stretch=NO, minwidth=0, width=120)
    tree.pack()
    DisplayData()
#function to update data into database
def Update():
    Database()
    #getting form data
    fname1=fname.get()
    lname1=lname.get()
    gender1=gender.get()
    address1=address.get()
    contact1=contact.get()
    #applying empty validation
    if fname1=='' or lname1==''or gender1=='' or address1==''or contact1=='':
        tkMessageBox.showinfo("Warning","fill the empty field!!!")
```

```
    else:
        #getting selected data
        curItem = tree.focus()
        contents = (tree.item(curItem))
        selecteditem = contents['values']
        #update query
        conn.execute('UPDATE REGISTRATION SET FNAME=?,LNAME=?,GENDER=?,ADDRESS=?,CONTACT=?
WHERE RID = ?',(fname1,lname1,gender1,address1,contact1, selecteditem[0]))
        conn.commit()
        tkMessageBox.showinfo("Message","Updated successfully")
        #reset form
        Reset()
        #refresh table data
        DisplayData()
        conn.close()


def register():
    Database()
    #getting form data
    fname1=fname.get()
    lname1=lname.get()
    gender1=gender.get()
    address1=address.get()
    contact1=contact.get()
    #applying empty validation
    if fname1=='' or lname1==''or gender1=='' or address1==''or contact1==':
```

```
        tkMessageBox.showinfo("Warning","fill the empty field!!!")
    else:
        #execute query
        conn.execute('INSERT INTO REGISTRATION (FNAME,LNAME,GENDER,ADDRESS,CONTACT) \
            VALUES (?,?,?,?,?)',(fname1,lname1,gender1,address1,contact1));
        conn.commit()
        tkMessageBox.showinfo("Message","Stored successfully")
        #refresh table data
        DisplayData()
        conn.close()
def Reset():
    #clear current data from table
    tree.delete(*tree.get_children())
    #refresh table data
    DisplayData()
    #clear search text
    SEARCH.set("")
    fname.set("")
    lname.set("")
    gender.set("")
    address.set("")
    contact.set("")
def Delete():
    #open database
    Database()
```

```
    if not tree.selection():
        tkMessageBox.showwarning("Warning","Select data to delete")
    else:
        result = tkMessageBox.askquestion('Confirm', 'Are you sure you want to delete this record?',
                                          icon="warning")
        if result == 'yes':
            curItem = tree.focus()
            contents = (tree.item(curItem))
            selecteditem = contents['values']
            tree.delete(curItem)
            cursor=conn.execute("DELETE FROM REGISTRATION WHERE RID = %d" % selecteditem[0])
            conn.commit()
            cursor.close()
            conn.close()


#function to search data
def SearchRecord():
    #open database
    Database()
    #checking search text is empty or not
    if SEARCH.get() != "":
        #clearing current display data
        tree.delete(*tree.get_children())
        #select query with where clause
        cursor=conn.execute("SELECT * FROM REGISTRATION WHERE FNAME LIKE ?", ('%' +
str(SEARCH.get()) + '%',))
```

```
        #fetch all matching records
        fetch = cursor.fetchall()
        #loop for displaying all records into GUI
        for data in fetch:
            tree.insert('', 'end', values=(data))
        cursor.close()
        conn.close()
#defining function to access data from SQLite database
def DisplayData():
    #open database
    Database()
    #clear current data
    tree.delete(*tree.get_children())
    #select query
    cursor=conn.execute("SELECT * FROM REGISTRATION")
    #fetch all data from database
    fetch = cursor.fetchall()
    #loop for displaying all data in GUI
    for data in fetch:
        tree.insert('', 'end', values=(data))
        tree.bind("<Double-1>",OnDoubleClick)
    cursor.close()
    conn.close()
def OnDoubleClick(self):
    #getting focused item from treeview
```

```
    curItem = tree.focus()
    contents = (tree.item(curItem))
    selecteditem = contents['values']
    #set values in the fields
    fname.set(selecteditem[1])
    lname.set(selecteditem[2])
    gender.set(selecteditem[3])
    address.set(selecteditem[4])
    contact.set(selecteditem[5])

#calling function
DisplayForm()
if __name__=='__main__':
    #Running Application
    mainloop()
```