```
#import libraries
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 student database
    conn = sqlite3.connect("student.db")
    cursor = conn.cursor()
    #creating STUD_REGISTRATION table
    cursor.execute(
        "CREATE TABLE IF NOT EXISTS STUD_REGISTRATION (STU_ID INTEGER PRIMARY KEY
AUTOINCREMENT NOT NULL, STU_NAME TEXT, STU_CONTACT TEXT, STU_EMAIL TEXT, STU_ROLLNO
TEXT, STU_BRANCH 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 name,contact,email,rollno,branch
    SEARCH = StringVar()
    name = StringVar()
    contact = StringVar()
    email = StringVar()
    rollno = StringVar()
    branch = 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")
    LFrom.pack(side=LEFT, fill=Y)
    #seconf left frame for search form
    LeftViewForm = Frame(display_screen, width=500,bg="gray")
    LeftViewForm.pack(side=LEFT, fill=Y)
    #mid frame for displaying students record
    MidViewForm = Frame(display_screen, width=600)
    MidViewForm.pack(side=RIGHT)
    #label for heading
    lbl_text = Label(TopViewForm, text="Student Management System", font=('verdana', 18),
width=600,bg="#1C2833",fg="white")
    lbl_text.pack(fill=X)
```

```
    #creating registration form in first left frame
    Label(LFrom, text="Name  ", font=("Arial", 12)).pack(side=TOP)
    Entry(LFrom,font=("Arial",10,"bold"),textvariable=name).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Contact ", font=("Arial", 12)).pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=contact).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Email ", font=("Arial", 12)).pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=email).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Rollno ", font=("Arial", 12)).pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=rollno).pack(side=TOP, padx=10, fill=X)
    Label(LFrom, text="Branch ", font=("Arial", 12)).pack(side=TOP)
    Entry(LFrom, font=("Arial", 10, "bold"),textvariable=branch).pack(side=TOP, padx=10, fill=X)
    Button(LFrom,text="Submit",font=("Arial", 10, "bold"),command=register).pack(side=TOP,
padx=10,pady=5, fill=X)


    #creating search label and entry in second frame
    lbl_txtsearch = Label(LeftViewForm, text="Enter name to Search", font=('verdana', 10),bg="gray")
    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)
    btn_search.pack(side=TOP, padx=10, pady=10, fill=X)
    #creating view button
    btn_view = Button(LeftViewForm, text="View All", command=DisplayData)
    btn_view.pack(side=TOP, padx=10, pady=10, fill=X)
```

```
    #creating reset button
    btn_reset = Button(LeftViewForm, text="Reset", command=Reset)
    btn_reset.pack(side=TOP, padx=10, pady=10, fill=X)
    #creating delete button
    btn_delete = Button(LeftViewForm, text="Delete", command=Delete)
    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="Student Id", anchor=W)
    tree.heading('Name', text="Name", anchor=W)
    tree.heading('Contact', text="Contact", anchor=W)
    tree.heading('Email', text="Email", anchor=W)
    tree.heading('Rollno', text="Rollno", anchor=W)
    tree.heading('Branch', text="Branch", 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 insert data into database
def register():
    Database()
    #getting form data
    name1=name.get()
    con1=contact.get()
    email1=email.get()
    rol1=rollno.get()
    branch1=branch.get()
    #applying empty validation
    if name1=='' or con1==''or email1=='' or rol1==''or branch1=='':
        tkMessageBox.showinfo("Warning","fill the empty field!!!")
    else:
        #execute query
        conn.execute('INSERT INTO STUD_REGISTRATION
(STU_NAME,STU_CONTACT,STU_EMAIL,STU_ROLLNO,STU_BRANCH) \
            VALUES (?,?,?,?,?)',(name1,con1,email1,rol1,branch1));
        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("")
    name.set("")
    contact.set("")
    email.set("")
    rollno.set("")
    branch.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 STUD_REGISTRATION WHERE STU_ID = %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 STUD_REGISTRATION WHERE STU_NAME 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 STUD_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))
    cursor.close()
    conn.close()

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