```
import datetime
from tkinter import *
import tkinter.messagebox as mb
from tkinter import ttk
from tkcalendar import DateEntry  # pip install
tkcalendar
import sqlite3

# Creating the universal font variables
headlabelfont = ("Noto Sans CJK TC", 15, 'bold')
labelfont = ('Garamond', 14)
entryfont = ('Garamond', 12)

# Connecting to the Database where all information will
be stored
connector = sqlite3.connect('StudentManagement.db')
cursor = connector.cursor()

connector.execute(
"CREATE TABLE IF NOT EXISTS STUDENT_MANAGEMENT
(STUDENT_ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
NAME TEXT, EMAIL TEXT, PHONE_NO TEXT, GENDER TEXT, DOB
TEXT, STREAM TEXT)"
)

# Creating the functions
def reset_fields():
    global name_strvar, email_strvar, contact_strvar,
gender_strvar, dob, stream_strvar

    for i in ['name_strvar', 'email_strvar',
'contact_strvar', 'gender_strvar', 'stream_strvar']:
```

```
        exec(f"{i}.set('')")
    dob.set_date(datetime.datetime.now().date())


def reset_form():
    global tree
    tree.delete(*tree.get_children())

    reset_fields()


def display_records():
    tree.delete(*tree.get_children())

    curr = connector.execute('SELECT * FROM
STUDENT_MANAGEMENT')
    data = curr.fetchall()

    for records in data:
        tree.insert('', END, values=records)


def add_record():
    global name_strvar, email_strvar, contact_strvar,
gender_strvar, dob, stream_strvar

    name = name_strvar.get()
    email = email_strvar.get()
    contact = contact_strvar.get()
    gender = gender_strvar.get()
    DOB = dob.get_date()
    stream = stream_strvar.get()
```

```
    if not name or not email or not contact or not
gender or not DOB or not stream:
        mb.showerror('Error!', "Please fill all the
missing fields!!")
    else:
        try:
            connector.execute(
            'INSERT INTO STUDENT_MANAGEMENT (NAME,
EMAIL, PHONE_NO, GENDER, DOB, STREAM) VALUES
(?,?,?,?,?,?)', (name, email, contact, gender, DOB,
stream)
            )
            connector.commit()
            mb.showinfo('Record added', f"Record of
{name} was successfully added")
            reset_fields()
            display_records()
        except:
            mb.showerror('Wrong type', 'The type of the
values entered is not accurate. Pls note that the
contact field can only contain numbers')


def remove_record():
    if not tree.selection():
        mb.showerror('Error!', 'Please select an item
from the database')
    else:
        current_item = tree.focus()
        values = tree.item(current_item)
        selection = values["values"]
```

```
        tree.delete(current_item)

        connector.execute('DELETE FROM
STUDENT_MANAGEMENT WHERE STUDENT_ID=%d' % selection[0])
        connector.commit()

        mb.showinfo('Done', 'The record you wanted
deleted was successfully deleted.')

        display_records()


def view_record():
    global name_strvar, email_strvar, contact_strvar,
gender_strvar, dob, stream_strvar

    current_item = tree.focus()
    values = tree.item(current_item)
    selection = values["values"]

    date = datetime.date(int(selection[5][:4]),
int(selection[5][5:7]), int(selection[5][8:]))

    name_strvar.set(selection[1]);
email_strvar.set(selection[2])
    contact_strvar.set(selection[3]);
gender_strvar.set(selection[4])
    dob.set_date(date); stream_strvar.set(selection[6])


# Initializing the GUI window
```

```
main = Tk()
main.title('DataFlair Student Management System')
main.geometry('1000x600')
main.resizable(0, 0)

# Creating the background and foreground color variables
lf_bg = 'MediumSpringGreen' # bg color for the
left_frame
cf_bg = 'PaleGreen' # bg color for the center_frame

# Creating the StringVar or IntVar variables
name_strvar = StringVar()
email_strvar = StringVar()
contact_strvar = StringVar()
gender_strvar = StringVar()
stream_strvar = StringVar()

# Placing the components in the main window
Label(main, text="STUDENT MANAGEMENT SYSTEM",
font=headlabelfont, bg='SpringGreen').pack(side=TOP,
fill=X)

left_frame = Frame(main, bg=lf_bg)
left_frame.place(x=0, y=30, relheight=1, relwidth=0.2)

center_frame = Frame(main, bg=cf_bg)
center_frame.place(relx=0.2, y=30, relheight=1,
relwidth=0.2)

right_frame = Frame(main, bg="Gray35")
right_frame.place(relx=0.4, y=30, relheight=1,
relwidth=0.6)
```

```
# Placing components in the left frame
Label(left_frame, text="Name", font=labelfont,
bg=lf_bg).place(relx=0.375, rely=0.05)
Label(left_frame, text="Contact Number", font=labelfont,
bg=lf_bg).place(relx=0.175, rely=0.18)
Label(left_frame, text="Email Address", font=labelfont,
bg=lf_bg).place(relx=0.2, rely=0.31)
Label(left_frame, text="Gender", font=labelfont,
bg=lf_bg).place(relx=0.3, rely=0.44)
Label(left_frame, text="Date of Birth (DOB)",
font=labelfont, bg=lf_bg).place(relx=0.1, rely=0.57)
Label(left_frame, text="Stream", font=labelfont,
bg=lf_bg).place(relx=0.3, rely=0.7)

Entry(left_frame, width=19, textvariable=name_strvar,
font=entryfont).place(x=20, rely=0.1)
Entry(left_frame, width=19, textvariable=contact_strvar,
font=entryfont).place(x=20, rely=0.23)
Entry(left_frame, width=19, textvariable=email_strvar,
font=entryfont).place(x=20, rely=0.36)
Entry(left_frame, width=19, textvariable=stream_strvar,
font=entryfont).place(x=20, rely=0.75)

OptionMenu(left_frame, gender_strvar, 'Male',
"Female").place(x=45, rely=0.49, relwidth=0.5)

dob = DateEntry(left_frame, font=("Arial", 12),
width=15)
dob.place(x=20, rely=0.62)
```

```
Button(left_frame, text='Submit and Add Record',
font=labelfont, command=add_record,
width=18).place(relx=0.025, rely=0.85)

# Placing components in the center frame
Button(center_frame, text='Delete Record',
font=labelfont, command=remove_record,
width=15).place(relx=0.1, rely=0.25)
Button(center_frame, text='View Record', font=labelfont,
command=view_record, width=15).place(relx=0.1,
rely=0.35)
Button(center_frame, text='Reset Fields',
font=labelfont, command=reset_fields,
width=15).place(relx=0.1, rely=0.45)
Button(center_frame, text='Delete database',
font=labelfont, command=reset_form,
width=15).place(relx=0.1, rely=0.55)

# Placing components in the right frame
Label(right_frame, text='Students Records',
font=headlabelfont, bg='DarkGreen',
fg='LightCyan').pack(side=TOP, fill=X)

tree = ttk.Treeview(right_frame, height=100,
selectmode=BROWSE,
                    columns=('Student ID', "Name",
"Email Address", "Contact Number", "Gender", "Date of
Birth", "Stream"))

X_scroller = Scrollbar(tree, orient=HORIZONTAL,
command=tree.xview)
```

```
Y_scroller = Scrollbar(tree, orient=VERTICAL,
command=tree.yview)

X_scroller.pack(side=BOTTOM, fill=X)
Y_scroller.pack(side=RIGHT, fill=Y)

tree.config(yscrollcommand=Y_scroller.set,
xscrollcommand=X_scroller.set)

tree.heading('Student ID', text='ID', anchor=CENTER)
tree.heading('Name', text='Name', anchor=CENTER)
tree.heading('Email Address', text='Email ID',
anchor=CENTER)
tree.heading('Contact Number', text='Phone No',
anchor=CENTER)
tree.heading('Gender', text='Gender', anchor=CENTER)
tree.heading('Date of Birth', text='DOB', anchor=CENTER)
tree.heading('Stream', text='Stream', anchor=CENTER)

tree.column('#0', width=0, stretch=NO)
tree.column('#1', width=40, stretch=NO)
tree.column('#2', width=140, stretch=NO)
tree.column('#3', width=200, stretch=NO)
tree.column('#4', width=80, stretch=NO)
tree.column('#5', width=80, stretch=NO)
tree.column('#6', width=80, stretch=NO)
tree.column('#7', width=150, stretch=NO)

tree.place(y=30, relwidth=1, relheight=0.9, relx=0)

display_records()
```

```
# Finalizing the GUI window
main.update()
main.mainloop()
```