NAME = Lalit Bhagwan Patil

PROJECT = School Management System

INPUT :-

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('SchoolManagement.db')

cursor = connector.cursor()

connector.execute(

"CREATE TABLE IF NOT EXISTS SCHOOL\_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 SCHOOL\_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 SCHOOL\_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 SCHOOL\_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])

def search\_record():

    global name\_strvar

    name = name\_strvar.get()

    if not name:

        mb.showerror('Error!', 'Please enter a name to search for.')

    else:

        cursor.execute('SELECT \* FROM SCHOOL\_MANAGEMENT WHERE NAME LIKE ?', ('%' + name + '%',))

        data = cursor.fetchall()

        tree.delete(\*tree.get\_children())

        for records in data:

            tree.insert('', END, values=records)

# Initializing the GUI window

main = Tk()

main.title('DataFlair School Management System')

main.geometry('1000x600')

main.resizable(0, 0)

# Creating the background and foreground color variables

lf\_bg = 'Light Blue' # bg color for the left\_frame

cf\_bg = 'Light Pink' # 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="SCHOOL MANAGEMENT SYSTEM", font=headlabelfont, bg='navajo white').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='Search Record', font=labelfont, command=search\_record, width=15).place(relx=0.1, rely=0.55)

# Placing components in the right frame

Label(right\_frame, text='Students Records', font=headlabelfont, bg='lemon chiffon', fg='black').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()

OUTPUT:-

