Source Code Of RMS Projects

Student.py

```
from tkinter import*
from PIL import Image,ImageTk  #pip install pillow
from tkinter import ttk,messagebox
import sqlite3
class studentClass:
    def __init__(self,root):
        self.root=root
        self.root.title("Student Result Management System")
        self.root.geometry("1200x480+80+170")
        self.root.config(bg="white")
        self.root.focus_force()
        #=====title======
        title=Label(self.root,text="Manage Student Details",font=("goudy old style",20,"bold"),bg="#033054",fg="white").place(x=10,y=15,width=1180,height=35)
        #=======Variables======
        self.var_roll=StringVar()
        self.var_name=StringVar()
        self.var_email=StringVar()
        self.var_gender=StringVar()
        self.var_dob=StringVar()
        self.var_contact=StringVar()
        self.var_course=StringVar()
        self.var_a_date=StringVar()
        self.var_state=StringVar()
        self.var_city=StringVar()
        self.var_pin=StringVar()
```

```
        #========Widgets ======================

        #==========column 1 ================

        lbl_roll=Label(self.root,text="Roll No.", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=60)

        lbl_Name=Label(self.root,text="Name", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=100)

        lbl_Email=Label(self.root,text="Email", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=140)

        lbl_gender=Label(self.root,text="Gender", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=180)

        lbl_state=Label(self.root,text="State", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=220)

        txt_state=Entry(self.root,textvariable=self.var_state, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=150,y=220,width=150)

        lbl_city=Label(self.root,text="City", font=("goudy old
style",15,'bold'),bg='white').place(x=310,y=220)

        txt_city=Entry(self.root,textvariable=self.var_city, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=380,y=220,width=100)

        lbl_pin=Label(self.root,text="Pin", font=("goudy old
style",15,'bold'),bg='white').place(x=500,y=220)

        txt_pin=Entry(self.root,textvariable=self.var_pin, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=560,y=220,width=120)

        lbl_address=Label(self.root,text="Address", font=("goudy old
style",15,'bold'),bg='white').place(x=10,y=260)




        #========Entry Fields=======


        self.txt_roll=Entry(self.root,textvariable=self.var_roll, font=("goudy old
style",15,'bold'),bg='lightyellow')
```

```
        self.txt_roll.place(x=150,y=60,width=200)

        txt_name=Entry(self.root,textvariable=self.var_name,font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=150,y=100,width=200)

        txt_email=Entry(self.root,textvariable=self.var_email, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=150,y=140,width=200)

        self.txt_gender=ttk.Combobox(self.root,textvariable=self.var_gender,values=("Se
lect","Male","Female","Other"), font=("goudy old
style",15,'bold'),state='readonly',justify=CENTER)

        self.txt_gender.place(x=150,y=180,width=200)

        self.txt_gender.current(0)



        #============column 2=============

        lbl_dob=Label(self.root,text="D.O.B", font=("goudy old
style",15,'bold'),bg='white').place(x=360,y=60)

        lbl_contact=Label(self.root,text="Contact", font=("goudy old
style",15,'bold'),bg='white').place(x=360,y=100)

        lbl_addmission=Label(self.root,text="Addmission", font=("goudy old
style",15,'bold'),bg='white').place(x=360,y=140)

        lbl_course=Label(self.root,text="Course", font=("goudy old
style",15,'bold'),bg='white').place(x=360,y=180)

        #========Entry Fields2=======

        self.course_list=[]

        #function_call to update to the list

        self.fetch_course()

        txt_dob=Entry(self.root,textvariable=self.var_dob, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=480,y=60,width=200)

        txt_contact=Entry(self.root,textvariable=self.var_contact,font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=480,y=100,width=200)

        txt_addmission=Entry(self.root,textvariable=self.var_a_date, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=480,y=140,width=200)

        self.txt_course=ttk.Combobox(self.root,textvariable=self.var_course,values=self.
course_list, font=("goudy old style",15,'bold'),state='readonly',justify=CENTER)

        self.txt_course.place(x=480,y=180,width=200)
```

```
        self.txt_course.set("Select")


        #==========Text Address=============


        self.txt_address=Text(self.root, font=("goudy old style",15,'bold'),bg='lightyellow')

        self.txt_address.place(x=150,y=260,width=540,height=100)


        #=====Buttons======


        self.btn_add=Button(self.root,text='Save',font=("goudy old
style",15,"bold"),bg="#2196f3",fg="white",cursor="hand2",command=self.add)

        self.btn_add.place(x=150,y=400,width=110,height=40)

        self.btn_update=Button(self.root,text='Update',font=("goudy old
style",15,"bold"),bg="#4caf50",fg="white",cursor="hand2",command=self.update)

        self.btn_update.place(x=270,y=400,width=110,height=40)

        self.btn_delete=Button(self.root,text='Delete',font=("goudy old
style",15,"bold"),bg="#f44336",fg="white",cursor="hand2",command=self.delete)

        self.btn_delete.place(x=390,y=400,width=110,height=40)

        self.btn_clear=Button(self.root,text='Clear',font=("goudy old
style",15,"bold"),bg="#607d8b",fg="white",cursor="hand2",command=self.clear)

        self.btn_clear.place(x=510,y=400,width=110,height=40)


        #=======Search Panel==========

        self.var_search=StringVar()

        lbl_search_roll=Label(self.root,text="Roll No. ", font=("goudy old
style",15,'bold'),bg='white').place(x=720,y=60)

        txt_search_roll=Entry(self.root,textvariable=self.var_search, font=("goudy old
style",15,'bold'),bg='lightyellow').place(x=870,y=60,width=180)

        btn_search=Button(self.root,text='Search',font=("goudy old
style",15,"bold"),bg="#03a9f4",fg="white",cursor="hand2",command=self.search).plac
e(x=1070,y=60,width=120,height=28)
```

```
        #========content=============
        self.C_Frame=Frame(self.root,bd=2,relief=RIDGE)
        self.C_Frame.place(x=720,y=100,width=470,height=340)

        scrolly=Scrollbar(self.C_Frame,orient=VERTICAL)
        scrollx=Scrollbar(self.C_Frame,orient=HORIZONTAL)
        self.CourseTable=ttk.Treeview(self.C_Frame,columns=("roll","name","email","gender","dob","contact","admission","course","state","city","pin","address"),xscrollcommand=scrollx.set,yscrollcommand=scrolly.set)

        scrollx.pack(side=BOTTOM,fill=X)
        scrolly.pack(side=RIGHT,fill=Y)
        scrollx.config(command=self.CourseTable.xview)
        scrolly.config(command=self.CourseTable.yview)

        self.CourseTable.heading("roll",text="Roll No")
        self.CourseTable.heading("name",text="Name")
        self.CourseTable.heading("email",text="Email")
        self.CourseTable.heading("gender",text="Gender")
        self.CourseTable.heading("dob",text="D.O.B")
        self.CourseTable.heading("contact",text=" Contact")
        self.CourseTable.heading("admission",text="Admission")
        self.CourseTable.heading("course",text="Course")
        self.CourseTable.heading("state",text="State")
        self.CourseTable.heading("city",text="City")
        self.CourseTable.heading("pin",text="PIN")
        self.CourseTable.heading("address",text="Address")
        self.CourseTable["show"]='headings'
        self.CourseTable.column("roll",width=100)
        self.CourseTable.column("name",width=100)
```

```
        self.CourseTable.column("email",width=100)
        self.CourseTable.column("gender",width=100)
        self.CourseTable.column("dob",width=150)
        self.CourseTable.column("contact",width=100)
        self.CourseTable.column("admission",width=100)
        self.CourseTable.column("course",width=100)
        self.CourseTable.column("state",width=100)
        self.CourseTable.column("city",width=100)
        self.CourseTable.column("pin",width=100)
        self.CourseTable.column("address",width=100)
        self.CourseTable.pack(fill=BOTH,expand=1)
        self.CourseTable.bind("<ButtonRelease-1>",self.get_data)
        self.show()


#================================================================================

    def clear(self):
        self.show()
        self.var_roll.set(""),
        self.var_name.set(""),
        self.var_email.set(""),
        self.var_gender.set("Select"),
        self.var_dob.set(""),
        self.var_contact.set(""),
        self.var_a_date.set(""),
        self.var_course.set("Select"),
        self.var_state.set(""),
        self.var_city.set(""),
        self.var_pin.set(""),
```

```
        self.txt_address.delete("1.0",END)
        self.txt_roll.config(state=NORMAL)
        self.var_search.set("")



    def delete(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            if self.var_roll.get()=="":
                messagebox.showerror("Error","Roll No. should be
required",parent=self.root)
            else:
                cur.execute("select * from student where roll=?",(self.var_roll.get(),))
                row=cur.fetchone()
                print(row)
                if row==None:
                    messagebox.showerror("Error","please select student from list
",parent=self.root)
                else:
                    op=messagebox.askyesno("Confirm","Do you really want to
delete?",parent=self.root)
                    if op==True:
                        cur.execute("delete from student where roll=?",(self.var_roll.get(),))
                        con.commit()
                        messagebox.showinfo("Delete","Student deleted
Successfully",parent=self.root)
                        self.clear()


        except Exception as ex:
```

```
            messagebox.showerror("Error",f"Error due to {str(ex)}")



    def get_data(self,ev):
        self.txt_roll.config(state='readonly')

        r=self.CourseTable.focus()
        content=self.CourseTable.item(r)
        row=content["values"]
        self.var_roll.set(row[0]),
        self.var_name.set(row[1]),
        self.var_email.set(row[2]),
        self.var_gender.set(row[3]),
        self.var_dob.set(row[4]),
        self.var_contact.set(row[5]),
        self.var_a_date.set(row[6]),
        self.var_course.set(row[7]),
        self.var_state.set(row[8]),
        self.var_city.set(row[9]),
        self.var_pin.set(row[10]),
        self.txt_address.delete("1.0",END)
        self.txt_address.insert(END,row[11])



    def add(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            if self.var_roll.get()=="":
```

```
            messagebox.showerror("Error","Roll No should be
required",parent=self.root)
        else:
            cur.execute("select * from student where roll=?",(self.var_roll.get(),))
            row=cur.fetchone()
            print(row)
            if row!=None:
                messagebox.showerror("Error","Roll No already
avilable",parent=self.root)
            else:
                cur.execute("insert into
student(roll,name,email,gender,dob,contact,admission,course,state,city,pin,address)
values(?,?,?,?,?,?,?,?,?,?,?,?)",(
                    self.var_roll.get(),
                    self.var_name.get(),
                    self.var_email.get(),
                    self.var_gender.get(),
                    self.var_dob.get(),
                    self.var_contact.get(),
                    self.var_a_date.get(),
                    self.var_course.get(),
                    self.var_state.get(),
                    self.var_city.get(),
                    self.var_pin.get(),
                    self.txt_address.get("1.0",END)
                ))
                con.commit()
                messagebox.showinfo("Success","Student Added
Successfully",parent=self.root)
                self.show()
    except Exception as ex:
```

```
            messagebox.showerror("Error",f"Error due to {str(ex)}")


    def update(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            if self.var_roll.get()=="":
                messagebox.showerror("Error","Roll No should be
required",parent=self.root)
            else:
                cur.execute("select * from student where roll=?",(self.var_roll.get(),))
                row=cur.fetchone()
                print(row)
                if row==None:
                    messagebox.showerror("Error","Select student from
list",parent=self.root)
                else:
                    cur.execute("update student set
name=?,email=?,gender=?,dob=?,contact=?,admission=?,course=?,state=?,city=?,pin=?,a
ddress=? where roll=?",(


                        self.var_name.get(),
                        self.var_email.get(),
                        self.var_gender.get(),
                        self.var_dob.get(),
                        self.var_contact.get(),
                        self.var_a_date.get(),
                        self.var_course.get(),
                        self.var_state.get(),
                        self.var_city.get(),
                        self.var_pin.get(),
```

```
                    self.txt_address.get("1.0",END),
                    self.var_roll.get()
                ))
                con.commit()
                messagebox.showinfo("Success","Student update
Successfully",parent=self.root)
                self.show()
        except Exception as ex:
            messagebox.showerror("Error",f"Error due to {str(ex)}")


    def show(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            cur.execute("select * from student ")
            rows=cur.fetchall()
            self.CourseTable.delete(*self.CourseTable.get_children())
            for row in rows:
                self.CourseTable.insert('',END,values=row)


        except Exception as ex:
            messagebox.showerror("Error",f"Error due to {str(ex)}")


    def fetch_course(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            cur.execute("select name from course")
            rows=cur.fetchall()
            if len(rows)>0:
```

```
            for row in rows:
                self.course_list.append(row[0])
        except Exception as ex:
            messagebox.showerror("Error",f"Error due to {str(ex)}")


    def search(self):
        con=sqlite3.connect(database="rms.db")
        cur=con.cursor()
        try:
            cur.execute(f"select * from student where roll=?",(self.var_search.get(),))
            row=cur.fetchone()
            if row!=None:
                self.CourseTable.delete(*self.CourseTable.get_children())
                self.CourseTable.insert('',END,values=row)
            else:
                messagebox.showerror("Error","No record found",parent=self.root)


        except Exception as ex:
            messagebox.showerror("Error",f"Error due to {str(ex)}")


if __name__=="__main__":
    root=Tk()
    obj=studentClass(root)
    root.mainloop()
```