PACKAGES TO BE INSTALLED IN PYTHON INTERPRETER TO RUN THE CODE:

PACKAGE	VERSION
Pillow	8.0.0
PyMySQL	0.10.1
numpy	1.19.4
opencv-contrib-python	4.4.0.46
pandas	1.1.4
pip	20.2.4
python-dateutil	2.8.1
pytz	2020.4
setuptools	50.3.2
six	1.15.0

CODE :

```
from tkinter import *
from tkinter import ttk
import pymysql
from tkinter import messagebox
from PIL import ImageTk,Image
class Student:
  def __init__(self,root):
    self.root=root
    self.root.title("Scholar Stewardship Application")
    self.root.geometry("1350x700+0+0")
    self.root.configure(bg="AntiqueWhite1")


    title=Label(self.root,text="Scholar Stewardship Application",bd=10,relief=GROOVE,font=("times new roman",40,"bold"),bg="navy",fg="White")
    title.pack(side=TOP,fill=X)
```

```
    #---------------------------------------------------

    Manage_Frame=Frame(self.root,bd=4,relief=RIDGE,bg="crimson")
    Manage_Frame.place(x=20,y=100,width=450,height=580)

    m_title=Label(Manage_Frame,text="Administrate Student
Details",bg="crimson",fg="white",font=("times new roman",25,"bold"))
    m_title.grid(row=0,columnspan=2,pady=10)

    self.Roll_No_var=StringVar()
    self.name_var = StringVar()
    self.email_var = StringVar()
    self.gender_var = StringVar()
    self.contact_var = StringVar()
    self.dob_var = StringVar()
    self.address_var = StringVar()

    self.search_by=StringVar()
    self.search_txt=StringVar()



    lbl_roll = Label(Manage_Frame,text="Roll number",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
    lbl_roll.grid(row=1,column=0,pady=10,padx=20,sticky="w")

    txt_roll=Entry(Manage_Frame,textvariable=self.Roll_No_var,font=("times new roman",15,
"bold"),bd=5,relief=GROOVE)
    txt_roll.grid(row=1,column=1,pady=10,padx=20,sticky="w")

    lbl_name = Label(Manage_Frame, text="Name", bg="crimson", fg="white", font=("times new
roman", 20, "bold"))
    lbl_name.grid(row=2, column=0, pady=10, padx=20, sticky="w")

    txt_name = Entry(Manage_Frame,textvariable=self.name_var, font=("times new roman", 15,
"bold"), bd=5, relief=GROOVE)
    txt_name.grid(row=2, column=1, pady=10, padx=20, sticky="w")
```

```
    lbl_email = Label(Manage_Frame, text="Email", bg="crimson",fg="white",font=("times new roman", 20, "bold"))
    lbl_email.grid(row=3, column=0, pady=10, padx=20, sticky="w")

    txt_email = Entry(Manage_Frame,textvariable=self.email_var, font=("times new roman", 15, "bold"),bd=5, relief=GROOVE)
    txt_email.grid(row=3, column=1, pady=10, padx=20, sticky="w")

    lbl_gender = Label(Manage_Frame, text="Gender", bg="crimson",fg="white",font=("times new roman", 20, "bold"))
    lbl_gender.grid(row=4, column=0, pady=10, padx=20, sticky="w")

    combo_gender=ttk.Combobox(Manage_Frame,textvariable=self.gender_var,font=("times new roman",14))
    combo_gender['values']=("Male","Female","Other")
    combo_gender.grid(row=4,column=1,padx=20,pady=10)

    lbl_contact = Label(Manage_Frame, text="Contact", bg="crimson", fg="white", font=("times new roman", 20, "bold"))
    lbl_contact.grid(row=5, column=0, pady=10, padx=20, sticky="w")

    txt_contact = Entry(Manage_Frame,textvariable=self.contact_var, font=("times new roman",15, "bold"), bd=5, relief=GROOVE)
    txt_contact.grid(row=5, column=1, pady=10, padx=20, sticky="w")

    lbl_dob = Label(Manage_Frame, text="DOB", bg="crimson", fg="white", font=("times new roman", 20, "bold"))
    lbl_dob.grid(row=6, column=0, pady=10, padx=20, sticky="w")

    txt_dob = Entry(Manage_Frame,textvariable=self.dob_var, font=("times new roman", 15, "bold"), bd=5, relief=GROOVE)
    txt_dob.grid(row=6, column=1, pady=10, padx=20, sticky="w")

    lbl_address = Label(Manage_Frame, text="Address", bg="crimson", fg="white", font=("times new roman", 20, "bold"))
    lbl_address.grid(row=7, column=0, pady=10, padx=20, sticky="w")

    self.txt_address=Text(Manage_Frame,width=30,height=4,font=("",10))
    self.txt_address.grid(row=7,column=1,pady=10,padx=20,sticky="w")
```

```
    #_____________________________________________________

    btn_Frame = Frame(Manage_Frame, bd=4, relief=RIDGE, bg="crimson")
    btn_Frame.place(x=15,y=500, width=420)

    Addbtn = Button(btn_Frame, text="Add", width=10,command=self.add_students).grid(row=0,
column=0, padx=10, pady=10)
    updatebtn = Button(btn_Frame, text="Update",
width=10,command=self.update_data).grid(row=0, column=1, padx=10, pady=10)
    deletebtn = Button(btn_Frame, text="Delete", width=10,command=self.delete_data).grid(row=0,
column=2, padx=10, pady=10)
    Clearbtn = Button(btn_Frame, text="Clear", width=10,command=self.clear).grid(row=0,
column=3, padx=10, pady=10)



    #----------------------------------------
    Detail_Frame=Frame(self.root,bd=4,relief=RIDGE,bg="green3")
    Detail_Frame.place(x=500,y=100,width=800,height=560)

    lbl_Search=Label(Detail_Frame, text="Search", bg="green3", fg="white", font=("times new
roman", 20, "bold"))
    lbl_Search.grid(row=0, column=0, pady=10, padx=20, sticky="w")

    combo_search = ttk.Combobox(Detail_Frame,width=10,textvariable=self.search_by,
font=("times new roman", 13,"bold"),state='readonly')
    combo_search['values'] = ("Roll_No", "Name", "Contact")
    combo_search.grid(row=0, column=1, padx=20, pady=10)

    txt_search = Entry(Detail_Frame,width=20,textvariable=self.search_txt, font=("times new
roman", 10, "bold"), bd=5, relief=GROOVE)
    txt_search.grid(row=0, column=2, pady=10, padx=20, sticky="w")

    searchbtn = Button(Detail_Frame, text="Search",
width=10,pady=5,command=self.search_data).grid(row=0, column=3, padx=10, pady=10)
    showallbtn = Button(Detail_Frame, text="Showall",
width=10,pady=5,command=self.fetch_data).grid(row=0, column=4, padx=10, pady=10)

    #_____________________________________________________
```

```
    Table_Frame=Frame(Detail_Frame,bd=4,relief=RIDGE,bg="grey1")
    Table_Frame.place(x=10,y=70,width=760,height=480)

    scroll_x=Scrollbar(Table_Frame,orient=HORIZONTAL)
    scroll_y = Scrollbar(Table_Frame, orient=VERTICAL)

self.Student_table=ttk.Treeview(Table_Frame,columns=("roll","name","email","gender","contact","d
ob","address"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set)
    scroll_x.pack(side=BOTTOM,fill=X)
    scroll_y.pack(side=RIGHT,fill=Y)
    scroll_x.config(command=self.Student_table.xview)
    scroll_y.config(command=self.Student_table.yview)
    self.Student_table.heading("roll",text="Roll No.")
    self.Student_table.heading("name",text="Name")
    self.Student_table.heading("email", text="Email")
    self.Student_table.heading("gender", text="Gender")
    self.Student_table.heading("contact", text="Conatct")
    self.Student_table.heading("dob", text="DOB")
    self.Student_table.heading("address", text="Address")

    self.Student_table['show']='headings'
    self.Student_table.column("roll",width=100)
    self.Student_table.column("name", width=100)
    self.Student_table.column("email", width=100)
    self.Student_table.column("gender", width=100)
    self.Student_table.column("contact", width=100)
    self.Student_table.column("dob", width=100)
    self.Student_table.column("address", width=150)

    self.Student_table.pack(fill=BOTH,expand=1)
    self.Student_table.bind("<ButtonRelease-1>",self.get_cursor)
    self.fetch_data()

  def add_students(self):
    if self.Roll_No_var.get()=="" or self.name_var.get()=="":
      messagebox.showerror("Eroor","All fields are required!!!")
    else:
      con = pymysql.connect(host="localhost", user="root", password="", database="stm")
      cur = con.cursor()
```

```
    cur.execute("insert into students values(%s,%s,%s,%s,%s,%s,%s)", (self.Roll_No_var.get(),
                                        self.name_var.get(),
                                        self.email_var.get(),
                                        self.gender_var.get(),
                                        self.contact_var.get(),
                                        self.dob_var.get(),
                                        self.txt_address.get('1.0', END)
                                        ))
    con.commit()
    self.fetch_data()
    self.clear()
    con.close()
    messagebox.showinfo("Sucess","Record has been inserted")



def fetch_data(self):
  con = pymysql.connect(host="localhost", user="root", password="", database="stm")
  cur = con.cursor()
  cur.execute("select * from students")
  rows=cur.fetchall()
  if len(rows)!=0:
    self.Student_table.delete(*self.Student_table.get_children())
    for row in rows:
      self.Student_table.insert("",END,values=row)
      con.commit()
  con.close()

def clear(self):
  self.Roll_No_var.set("")
  self.name_var.set("")
  self.email_var.set("")
  self.gender_var.set("")
  self.contact_var.set("")
  self.dob_var.set("")
  self.txt_address.delete("1.0",END)

def get_cursor(self,ev):
  cursor_row=self.Student_table.focus()
```

```
    contents=self.Student_table.item(cursor_row)
    row=contents['values']
    self.Roll_No_var.set(row[0])
    self.name_var.set(row[1])
    self.email_var.set(row[2])
    self.gender_var.set(row[3])
    self.contact_var.set(row[4])
    self.dob_var.set(row[5])
    self.txt_address.delete("1.0", END)
    self.txt_address.insert(END,row[6])

  def update_data(self):
    con = pymysql.connect(host="localhost", user="root", password="", database="stm")
    cur = con.cursor()
    cur.execute("update students set
name=%s,email=%s,gender=%s,contact=%s,dob=%s,address=%s where roll_no=%s",(
                                        self.name_var.get(),
                                        self.email_var.get(),
                                        self.gender_var.get(),
                                        self.contact_var.get(),
                                        self.dob_var.get(),
                                        self.txt_address.get('1.0', END),
                                        self.Roll_No_var.get()
                                        ))
    con.commit()
    self.fetch_data()
    self.clear()
    con.close()

  def delete_data(self):
    con = pymysql.connect(host="localhost", user="root", password="", database="stm")
    cur = con.cursor()
    cur.execute("delete from students where roll_no=%s",self.Roll_No_var.get())
    con.commit()
    con.close()
    self.fetch_data()
    self.clear()

  def search_data(self):
```

```
    con = pymysql.connect(host="localhost", user="root", password="", database="stm")
    cur = con.cursor()
    cur.execute("select * from students where " +str(self.search_by.get())+" LIKE
'%"+str(self.search_txt.get())+"%'")
    rows=cur.fetchall()
    if len(rows)!=0:
      self.Student_table.delete(*self.Student_table.get_children())
      for row in rows:
        self.Student_table.insert("",END,values=row)
        con.commit()
    con.close()


root=Tk()
ob=Student(root)
root.mainloop()
```