```
from distutils import command
from struct import pack
from tkinter import *
from tkinter import ttk
from typing_extensions import _AnnotatedAlias
from PIL import Image,ImageTk
import os
import pickle
from cv2 import magnitude
import mysql.connector  as sql
from tkinter import messagebox
from datetime import date
from datetime import time
from datetime import *
import requests
from bs4 import BeautifulSoup
import time
import dashboard
import csv
import regisf_stu

def stu_screen():
    def click_go_to_dashboard():
        """returns AdminDashboard class when clicked go to dashboard"""
        dashboard.dashboard()
        root.withdraw()

    def load_data():
        f=open("Credentials.csv","r")
        s=csv.reader(f,delimiter="-")
        d=[]
        for i in s:
            d.append(i)
        a=d[::-1]
        return (a[0])

    def click_view_all():
        """it will show all the data contains on the student table of cms database, when clicked by default this method
        is called while initializing the class ManageStudent. Exception is handled to avoid run time error which may
        cause by user.
        """
        try:
            #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
            #db_connection.create(obj_student_database.get_database())
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
            port=a[1]

            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
            mycur=spec.cursor()

            query = "select * from students;"
            mycur.execute(query)

            data = mycur.fetchall()

            # print(data)
            student_tree.delete(*student_tree.get_children())
```

```
            for values in data:
                data_list = [values[0], values[4], values[5], values[2], values[6], values[7],
                             values[8],
                             values[9], values[10], values[11], values[12], values[13],
values[14]]
                # print(data_list)
                student_tree.insert('', END, values=data_list)

        except BaseException as msg:
            print(msg)

    def click_delete_student():
        """when clicked delete students, it will require to select the students and after
selecting and
        performing the delete method, it will ask the admin either they are sure they want to
delete that student
        or not if yes then student containing that id in student table is deleted."""
        try:
            #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
            #db_connection.create(obj_student_database.get_database())
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
            port=a[1]

            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
")
            mycur=spec.cursor()

            tree_view_content = student_tree.focus()
            tree_view_items = student_tree.item(tree_view_content)
            tree_view_values = tree_view_items['values'][0]
            ask = messagebox.askyesno("Warning",
                                      f"Are you sure you want to delete Student having id {
tree_view_values}")
            if ask is True:
                query = f"delete from students where student_id={tree_view_values};"
                mycur.execute(query)
                spec.commit()
                #db_connection.delete(query, (tree_view_values,))
                messagebox.showinfo("Success", f" student id {tree_view_values} deleted
Successfully")

                click_view_all()
            else:
                pass

        except BaseException as msg:
            print(msg)
            messagebox.showerror("Error",
                                 "There is some error deleting the data\n Make sure you
have Selected the data")


    def update_stu():
        def tree_event_handle():
            try:
                #obj_student_database = Model_class.student_registration.GetDatabase('use cms
;')
                #db_connection.create(obj_student_database.get_database())

                tree_view_content = student_tree.focus()
```

```
116                 tree_view_items = student_tree.item(tree_view_content)
117                 # print(tree_view_items)
118                 tree_view_values = tree_view_items['values']
119                 tree_view_id = tree_view_items['values'][0]
120                 # print(tree_view_id)
121                 get_id.clear()
122                 list_of_tree.clear()
123                 get_id.append(tree_view_id)
124                 for i in tree_view_values:
125                     list_of_tree.append(i)
126                     # ManageStudent.get_id.append(tree_view_id)
127                 # print(ManageStudent.list_of_tree)
128                 # print(tree_view_values)
129 
130                 ask = messagebox.askyesno("Confirm",
131                                           f"Do you want to Update Student having id {
    tree_view_id}")
132                 if ask is True:
133                     #click_update_student()
134                     pass
135 
136             except BaseException as msg:
137                 print(msg)
138                 messagebox.showerror("Error",
139                                      "There is some error updating the data\n Make sure you
    have Selected the data")
140 
141         tree_event_handle()
142         def click_clear_button():
143             """this will clear entire field to default when click on clear button"""
144             username_entry.delete(0, END)
145             f_name_entry.delete(0, END)
146             l_name_entry.delete(0, END)
147             email_entry.delete(0, END)
148             password_entry.delete(0, END)
149             c_password_entry.delete(0, END)
150             dob_entry.delete(0, END)
151             gender_combo.current(0)
152             address_entry.delete(0, END)
153             contact_entry.delete(0, END)
154             shift_combo.current(0)
155             batch_combo.current(0)
156             course_combo.current(0)
157             batch_combo.current(0)
158             section_combo.current(0)
159 
160 
161         def update():
162             """updates the data of students from entry fields"""
163             try:
164                 #obj_student_database = Model_class.student_registration.GetDatabase('use cms
    ;')
165                 #db_connection.create(obj_student_database.get_database())
166 
167                 #get_id = ManageStudent.get_id
168                 data_id = get_id[0]
169                 # print(data_id)
170                 a=load_data()
171                 host=a[0]
172                 username = a[2]
173                 password = a[3]
174                 port=a[1]
175 
```

```
176                 spec=sql.connect(host=host,user=username,password=password,port=port,database=
    "sms")
177                 mycur=spec.cursor()
178 
179                 #obj_student_database = Model_class.student_registration.StudentRegistration(
    username_entry.get(),
180                                                                                              #
    email_entry.get(),
181                                                                                              #
    password_entry.get(),
182                                                                                              #
    f_name_entry.get(),
183                                                                                              #
    l_name_entry.get(),
184                                                                                              #
    dob_entry.get(),
185                                                                                              #
    gender_combo.get(),
186                                                                                              #
    address_entry.get(),
187                                                                                              #
    contact_entry.get(),
188                                                                                              #
    shift_combo.get(),
189                                                                                              #
    course_combo.get(),
190                                                                                              #
    batch_combo.get(),
191                                                                                              #
    section_combo.get(),
192                                                                                              #
    reg_date)
193                 query = f"update students set email='{email_entry.get()}',f_name='{
    f_name_entry.get()}', l_name='{l_name_entry.get()}',dob='{dob_entry.get()}',gender='{
    gender_combo.get()}',address='{address_entry.get()}',contact_no='{contact_entry.get()}'," \
194                         f"shift='{shift_combo.get()}', course_enrolled='{course_combo.get()}
    ', batch='{batch_combo.get()}', section_enrolled='{section_combo.get()}' where student_id={
    data_id}"
195                 mycur.execute(query)
196                 #values = (obj_student_database.get_email(), obj_student_database.get_f_name
    (),
197                         #obj_student_database.get_l_name(), obj_student_database.get_dob(),
198                         #obj_student_database.get_gender(), obj_student_database.get_address
    (),
199                         #obj_student_database.get_contact(), obj_student_database.get_shift(),
200                         #obj_student_database.get_course_id(), obj_student_database.
    get_batch_id(),
201                         #obj_student_database.get_section(), data_id)
202 
203                 #db_connection.update(query, values)
204                 spec.commit()
205                 click_view_all()
206                 ask = messagebox.askyesnocancel("Success", f"Data having \n Email={email_entry
    .get()} \n Updated Successfully\n"
207                                                            f"Do you want to Go Student Management
     Dashboard")
208                 if ask is True:
209                     #win = Toplevel()
210                     #Frontend.manage_student.ManageStudent(win)
211                     #window.withdraw()
212                     #win.deiconify()
213                     pass
214 
```

```
            except BaseException as msg:
                print(msg)
                messagebox.showerror("Error", f"Error due to{msg}")
 
 
 
 
 
 
        root = Toplevel()
        root.title('COLLEGE MANAGEMENT SYSTEM')
        root.geometry('1067x600')
        root.config(bg="#f29844")
        #root.iconbitmap('images\\logo.ico')
        root.resizable(False, False)
        #manage_student_frame = ImageTk.PhotoImage(file='Pics\\student_frame.png')

        # ======================Variables======================

        reg_frame = Frame(root, bg="#ffffff", width=1000, height=560)
        reg_frame.place(x=30, y=30)

        heading = Label(reg_frame, text="Student Registration Form", font=('yu gothic ui', 20
    , "bold"), bg="white",
                                   fg='black',
                                   bd=5,
                                   relief=FLAT)
        heading.place(x=200, y=0, width=600)


        cred_frame = LabelFrame(reg_frame, text="Account Details", bg="white", fg="#4f4e4d",
    height=140,
                                            width=800, borderwidth=2.4,
                                            font=("yu gothic ui", 13, "bold"))
        cred_frame.config(highlightbackground="red")
        cred_frame.place(x=100, y=50)
        info = Label(cred_frame,text="*You cant change Username and password leave the fields
    blank",bg="white", fg="#4f4e4d",font=("yu gothic ui", 13, "bold"))
        info.place(x=20,y=80)
        # ============================================================================
        # ============================Username========================================
        # ============================================================================

        username_label = Label(cred_frame, text="Username ", bg="white", fg="#4f4e4d",
                                       font=("yu gothic ui", 13, "bold"))
        username_label.place(x=10, y=10)

        username_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
                                       font=("yu gothic ui semibold", 12))
        username_entry.place(x=230, y=117, width=260)  # trebuchet ms

        username_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0
    )
        username_line.place(x=230, y=140)

        # ============================================================================
        # ============================Email===========================================
        # ============================================================================

        email_label = Label(cred_frame, text="Email ", bg="white", fg="#4f4e4d",
                                       font=("yu gothic ui", 13, "bold"))
        email_label.place(x=370, y=10)

```

```
273         email_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
274                                     font=("yu gothic ui semibold", 12))
275         email_entry.place(x=555, y=117, width=350)  # trebuchet ms
276 
277         email_line = Canvas(root, width=350, height=1.5, bg="#bdb9b1", highlightthickness=0)
278         email_line.place(x=555, y=140)
279 
280         # ============================================================================
281         # ============================Password========================================
282         # ============================================================================
283 
284         password_label = Label(cred_frame, text="Password ", bg="white", fg="#4f4e4d",
285                                     font=("yu gothic ui", 13, "bold"))
286         password_label.place(x=10, y=50)
287 
288         password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
289                                     font=("yu gothic ui semibold", 12), show="*")
290         password_entry.place(x=230, y=157, width=260)  # trebuchet ms
291 
292         password_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0
    )
293         password_line.place(x=230, y=180)
294 
295         # ============================================================================
296         # ============================Confirm password================================
297         # ============================================================================
298 
299         c_password_label = Label(cred_frame, text="Confirm Password ", bg="white", fg="#4f4e4d
    ",
300                                       font=("yu gothic ui", 13, "bold"))
301         c_password_label.place(x=370, y=50)
302 
303         c_password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
304                                       font=("yu gothic ui semibold", 12), show="*")
305         c_password_entry.place(x=650, y=157, width=255)  # trebuchet ms
306 
307         c_password_line = Canvas(root, width=255, height=1.5, bg="#bdb9b1", highlightthickness
    =0)
308         c_password_line.place(x=650, y=180)
309 
310 
311         # ============================================================================
312         # ========================frame for personal credentials ==================
313         # ============================================================================
314 
315         personal_frame = LabelFrame(reg_frame, text="Personal Details", bg="white", fg="#
    4f4e4d", height=265,
316                                         width=800, borderwidth=2.4,
317                                         font=("yu gothic ui", 13, "bold"))
318         personal_frame.config(highlightbackground="red")
319         personal_frame.place(x=100, y=200)
320 
321 
322 
323         # ============================================================================
324         # ============================First name======================================
325         # ============================================================================
326         f_name_label = Label(personal_frame, text="First Name ", bg="white", fg="#4f4e4d",
327                                     font=("yu gothic ui", 13, "bold"))
328         f_name_label.place(x=10, y=10)
329 
```

```
330         f_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
331                                  font=("yu gothic ui semibold", 12))
332         f_name_entry.place(x=235, y=267, width=260)  # trebuchet ms
333 
334         f_name_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0)
335         f_name_line.place(x=235, y=290)
336 
337         # ============================================================================
338         # ============================Last name=======================================
339         # ============================================================================
340 
341         l_name_label = Label(personal_frame, text="Last Name ", bg="white", fg="#4f4e4d",
342                                  font=("yu gothic ui", 13, "bold"))
343         l_name_label.place(x=370, y=10)
344 
345         l_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
346                                  font=("yu gothic ui semibold", 12))
347         l_name_entry.place(x=595, y=267, width=315)  # trebuchet ms
348 
349         l_name_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0)
350         l_name_line.place(x=595, y=290)
351 
352         # ============================================================================
353         # ============================DOB=============================================
354         # ============================================================================
355 
356         dob_label = Label(personal_frame, text="DOB ", bg="white", fg="#4f4e4d",
357                           font=("yu gothic ui", 13, "bold"))
358         dob_label.place(x=10, y=50)
359 
360         dob_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
361                           font=("yu gothic ui semibold", 12))
362         dob_entry.insert(0, "mm/dd/yyyy")
363         dob_entry.place(x=190, y=307, width=305)  # trebuchet ms
364         #dob_entry.bind("<1>", pick_date)
365 
366         dob_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
367         dob_line.place(x=190, y=329)
368 
369         # ============================================================================
370         # ============================Gender==========================================
371         # ============================================================================
372         style = ttk.Style()
373 
374         # style.map('TCombobox', selectbackground=[('readonly', 'grey')])
375         root.option_add("*TCombobox*Listbox*Foreground", '#f29844')
376 
377         gender_label = Label(personal_frame, text="Gender ", bg="white", fg="#4f4e4d",
378                                  font=("yu gothic ui", 13, "bold"))
379         gender_label.place(x=370, y=50)
380 
381         gender_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
382                                     width=35)
383 
384         gender_list = ['Male', 'Female', 'Rather not say']
385         gender_combo['values'] = gender_list
386         # gender_combo.current(0)
387         gender_combo.place(x=570, y=307)
388 
389         # gender_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0
```

```
389 )
390         # gender_line.place(x=595, y=369)
391 
392         # ============================================================================
393         # ============================Address=========================================
394         # ============================================================================
395 
396         address_label = Label(personal_frame, text="Address ", bg="white", fg="#4f4e4d",
397                                     font=("yu gothic ui", 13, "bold"))
398         address_label.place(x=10, y=90)
399 
400         address_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69
    ",
401                                     font=("yu gothic ui semibold", 12))
402         address_entry.place(x=215, y=347, width=280)  # trebuchet ms
403 
404         address_line = Canvas(root, width=280, height=1.5, bg="#bdb9b1", highlightthickness=0)
405         address_line.place(x=215, y=370)
406 
407         # ============================================================================
408         # ============================Contact no======================================
409         # ============================================================================
410 
411         contact_label = Label(personal_frame, text="Contact No. ", bg="white", fg="#4f4e4d",
412                                     font=("yu gothic ui", 13, "bold"))
413         contact_label.place(x=370, y=90)
414 
415         contact_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69
    ",
416                                     font=("yu gothic ui semibold", 12))
417         contact_entry.place(x=605, y=347, width=305)  # trebuchet ms
418 
419         contact_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
420         contact_line.place(x=605, y=370)
421 
422         # ============================================================================
423         # ============================Shift No========================================
424         # ============================================================================
425 
426         shift_label = Label(personal_frame, text="Shift ", bg="white", fg="#4f4e4d",
427                                     font=("yu gothic ui", 13, "bold"))
428         shift_label.place(x=10, y=130)
429 
430         shift_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
431                                        width=28)
432         shift_list = ["Morning", "Day", "Evening"]
433         shift_combo['values'] = shift_list
434         shift_combo.current(0)
435         shift_combo.place(x=213, y=387)
436 
437         def load_data():
438             f=open("Credentials.csv","r")
439             s=csv.reader(f,delimiter="-")
440             d=[]
441             for i in s:
442                 d.append(i)
443             a=d[::-1]
444             return (a[0])
445 
446         try:
447             #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
448             #db_connection.create(obj_student_database.get_database())
```

```
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
            port=a[1]


            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
    ")
            mycur=spec.cursor()

            query = "select * from section"
            mycur.execute(query)
            #data=mycur.fetchall()
            query = "select * from section"
            section_tuple = mycur.fetchall()
            # print(section_tuple)
            section_list = []
            for i in section_tuple:
                section_name = i[2]
                section_list.append(section_name)
                # print(section_name)
                # print(section_list)

        except BaseException as msg:
            print(msg)

        try:
            #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
            #db_connection.create(obj_student_database.get_database())
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
            port=a[1]


            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
    ")
            mycur=spec.cursor()

            query = "select * from course"
            mycur.execute(query)
            course_tuple = mycur.fetchall()

            # print(course_tuple)
            course_list = []
            for i in course_tuple:
                course_name = i[1]
                course_list.append(course_name)
                # print(course_name)
                # print(course_list)

        except BaseException as msg:
            print(msg)

        try:
            #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
            #db_connection.create(obj_student_database.get_database())
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
```

```
            port=a[1]


            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
            mycur=spec.cursor()

            query = "select * from batch"
            mycur.execute(query)
            batch_tuple = mycur.fetchall()
            # print(batch_tuple)
            batch_list = []
            for i in batch_tuple:
                batch_name = i[1]
                batch_list.append(batch_name)
                # print(batch_name)
                # print(batch_list)

        except BaseException as msg:
            print(msg)









        # ========================================================================
        # ============================Course enrolled=============================
        # ========================================================================

        course_label = Label(personal_frame, text="Course Enrolled ", bg="white", fg="#4f4e4d",
                                     font=("yu gothic ui", 13, "bold"))
        course_label.place(x=370, y=130)

        course_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='readonly',
                                            width=28)

        course_combo['values'] = course_list
        try:
            course_combo.current(0)
        except:
            messagebox.showerror("Error","You must add course first")
        course_combo.place(x=635, y=387)

        # ========================================================================
        # ============================Batch=======================================
        # ========================================================================

        batch_label = Label(personal_frame, text="Batch ", bg="white", fg="#4f4e4d",
                                     font=("yu gothic ui", 13, "bold"))
        batch_label.place(x=10, y=170)

        batch_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='readonly',
                                           width=28)

        batch_combo['values'] = batch_list
        try:
```

```
            batch_combo.current(0)
        except:
            messagebox.showerror("Error", "You must add batch first")
        batch_combo.place(x=213, y=427)

        # ============================================================================
        # ============================Section=========================================
        # ============================================================================

        section_label = Label(personal_frame, text="Section ", bg="white", fg="#4f4e4d",
                                    font=("yu gothic ui", 13, "bold"))
        section_label.place(x=370, y=170)

        section_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='readonly',
                                             width=28)

        section_combo['values'] = section_list
        try:
            section_combo.current(0)
        except:
            messagebox.showerror("Error", "You must add section first")
        section_combo.place(x=635, y=427)

        reg_date = time.strftime("%Y/%m/%d")

        # ============================================================================
        # ============================Register options================================
        # ============================================================================

        options_frame = LabelFrame(reg_frame, text="Register Options", bg="white", fg="#4f4e4d", height=83,
                                           width=800, borderwidth=2.4,
                                           font=("yu gothic ui", 13, "bold"))
        options_frame.config(highlightbackground="red")
        options_frame.place(x=100, y=470)

        # ============================================================================
        # ============================Register options================================
        # ============================================================================
        submit_img = ImageTk.PhotoImage \
            (file='Pics\\submit.png')

        submit = Button(options_frame, image=submit_img,
                                  font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="white"
                                  , borderwidth=0, background="white", cursor="hand2"
                                  , command=update)

        submit.image = submit_img
        submit.place(x=90, y=10)

        clear_img = ImageTk.PhotoImage \
            (file='Pics\\clear.png')
        clear_button = Button(options_frame, image=clear_img,
                                       font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="white"
                                       , borderwidth=0, background="white", cursor="hand2"
                                       ,command=click_clear_button)
        clear_button.image = clear_img
        clear_button.place(x=250, y=13)

        back_img = ImageTk.PhotoImage \
```

```
628             (file='Pics\\back.png')
629         back_button = Button(options_frame, image=back_img,
630                                     font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
631                                     , borderwidth=0, background="white", cursor="hand2")
632                                     #, command=click_back_button)
633         back_button.image = back_img
634         back_button.place(x=410, y=13)
635 
636         exit_img = ImageTk.PhotoImage \
637             (file='Pics\\exit.png')
638         exit_button = Button(options_frame, image=exit_img,
639                                     font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
640                                     , borderwidth=0, background="white", cursor="hand2",
    command=exit)
641         exit_button.image = exit_img
642         exit_button.place(x=570, y=13)
643 
644         a = list_of_tree
645         # print(a)
646         try:
647             email_entry.insert(0, a[3])
648             f_name_entry.insert(0, a[1])
649             l_name_entry.insert(0, a[2])
650             dob_entry.delete(0, END)
651             dob_entry.insert(0, a[4])
652             gender_combo.set(a[5])
653             address_entry.insert(0, a[6])
654             contact_entry.insert(0, a[7])
655             shift_combo.set(a[8])
656             course_combo.set(a[9])
657             batch_combo.set(a[10])
658             section_combo.set(a[11])
659         except IndexError as msg:
660             print(msg)
661 
662     list_of_tree = []
663     get_id = []
664 
665     root = Toplevel()
666     root.geometry("1067x600")
667     root.title("Student Management Dashboard - College Management System")
668     #root.iconbitmap('images\\logo.ico')
669     root.resizable(False, False)
670 
671     manage_student_frame_r = Image.open('Pics\\student_frame.png').resize((1067,600),Image.
    ANTIALIAS)
672     manage_student_frame = ImageTk.PhotoImage(manage_student_frame_r)
673     image_panel = Label(root, image=manage_student_frame)
674     image_panel.image = manage_student_frame
675     image_panel.pack(fill='both', expand='yes')
676 
677     #db_connection = Backend.connection.DatabaseConnection()
678 
679     txt = "Student Management Dashboard"
680     heading = Label(root, text=txt, font=('yu gothic ui', 20, "bold"), bg="white",fg='black',
    bd=5,relief=FLAT)
681     heading.place(x=420, y=23)
682 
683     # ============================================================================
684     # ========================Left frame ========================================
685     # ============================================================================
```

```
686
687     left_view_frame = Frame(root, bg="white")
688     left_view_frame.place(x=35, y=89, height=470, width=250)
689
690     # ========================================================================
691     # ========================Tree view frame=================================
692     # ========================================================================
693
694     tree_view_frame = Frame(root, bg="white")
695     tree_view_frame.place(x=301, y=90, height=473, width=730)
696
697     # ========================================================================
698     # ========================frame for personal credentials ==================
699     # ========================================================================
700
701     personal_frame = LabelFrame(left_view_frame, text="Student Management Options", bg="white"
    , fg="#4f4e4d",height=460,width=240, borderwidth=2.4,font=("yu gothic ui", 12, "bold"))
702     personal_frame.config(highlightbackground="red")
703     personal_frame.place(x=5, y=8)
704
705     # ========================================================================
706     # ============================Add student button==============================
707     # ========================================================================
708
709     add_student_r = Image.open('Pics\\add_student.png').resize((225,25),Image.ANTIALIAS)
710     add_student = ImageTk.PhotoImage(add_student_r)
711     add_student_button = Button(personal_frame, image=add_student, relief=FLAT, borderwidth=0,
    activebackground="white", bg="white", cursor="hand2",command=regisf_stu.regist_stu)
712     add_student_button.image = add_student
713     add_student_button.place(x=5, y=100)
714     # add_student_button.place(x=36, y=295)
715
716     # ========================================================================
717     # ============================Update student button==============================
718     # ========================================================================
719
720     update_student_r = Image.open('Pics\\update_student.png').resize((225,25),Image.ANTIALIAS)
721     update_student = ImageTk.PhotoImage(update_student_r)
722     update_student_button = Button(personal_frame, image=update_student, relief=FLAT,
    borderwidth=0,activebackground="white", bg="white", cursor="hand2",command=update_stu)
723     update_student_button.image = update_student
724     update_student_button.place(x=5, y=165)
725     # update_student_button.place(x=36, y=355)
726
727     # ========================================================================
728     # ============================Delete student button==============================
729     # ========================================================================
730
731     delete_student_r = Image.open('Pics\\delete_student.png').resize((225,25),Image.ANTIALIAS)
732     delete_student = ImageTk.PhotoImage(delete_student_r)
733     delete_student_button = Button(personal_frame, image=delete_student, relief=FLAT,
    borderwidth=0,activebackground="white", bg="white", cursor="hand2",command=
    click_delete_student)
734     delete_student_button.image = delete_student
735     delete_student_button.place(x=5, y=230)
736     # delete_student_button.place(x=36, y=405)
737
738     # ========================================================================
739     # ============================Goto Main dashboard button==============================
740     # ========================================================================
741
742     goto_dashboard_r = Image.open('Pics\\goto_dashboard.png').resize((225,25),Image.ANTIALIAS)
743     goto_dashboard = ImageTk.PhotoImage (goto_dashboard_r)
```

```
    goto_dashboard_button = Button(personal_frame, image=goto_dashboard, relief=FLAT,
borderwidth=0,activebackground="white", bg="white", cursor="hand2",command=
click_go_to_dashboard)
    goto_dashboard_button.image = goto_dashboard
    goto_dashboard_button.place(x=5, y=295)

    # ============================================================================
    # ========================Starting Tree View==============================
    # ============================================================================
    style = ttk.Style()
    style.configure("Treeview.Heading", font=('yu gothic ui', 10, "bold"), foreground="red")
    style.configure("Treeview", font=('yu gothic ui', 9, "bold"), foreground="#f29844")

    scroll_x = Scrollbar(tree_view_frame, orient=HORIZONTAL)
    scroll_y = Scrollbar(tree_view_frame, orient=VERTICAL)
    student_tree = ttk.Treeview(tree_view_frame,
                                        columns=(
                                            "STUDENT ID", "FNAME", "LNAME", "EMAIL", "DOB", "
GENDER", "ADDRESS",
                                            "CONTACT NO", "SHIFT", "COURSE ENROLLED", "BATCH"
, "SECTION",
                                            "REGISTRATION DATE"),
                                        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=student_tree.xview)
    scroll_y.config(command=student_tree.yview)

    # ============================TreeView Heading====================
    student_tree.heading("STUDENT ID", text="STUDENT ID")
    student_tree.heading("FNAME", text="FNAME")
    student_tree.heading("LNAME", text="LNAME")
    student_tree.heading("EMAIL", text="EMAIL")
    student_tree.heading("DOB", text="DOB")
    student_tree.heading("GENDER", text="GENDER")
    student_tree.heading("ADDRESS", text="ADDRESS")
    student_tree.heading("CONTACT NO", text="CONTACT NO")
    student_tree.heading("SHIFT", text="SHIFT")
    student_tree.heading("COURSE ENROLLED", text="COURSE ENROLLED")
    student_tree.heading("BATCH", text="BATCH")
    student_tree.heading("SECTION", text="SECTION")
    student_tree.heading("REGISTRATION DATE", text="REGISTRATION DATE")
    student_tree["show"] = "headings"

    # ============================TreeView Column====================
    student_tree.column("STUDENT ID", width=150)
    student_tree.column("FNAME", width=170)
    student_tree.column("LNAME", width=170)
    student_tree.column("EMAIL", width=200)
    student_tree.column("ADDRESS", width=150)
    student_tree.column("GENDER", width=150)
    student_tree.column("CONTACT NO", width=150)
    student_tree.column("DOB", width=150)
    student_tree.column("SHIFT", width=150)
    student_tree.column("COURSE ENROLLED", width=150)
    student_tree.column("BATCH", width=100)
    student_tree.column("SECTION", width=100)
    student_tree.column("REGISTRATION DATE", width=150)
    student_tree.pack(fill=BOTH, expand=1)

    click_view_all()

```

```
802
803     #root.mainloop()
804
805 if __name__ == "__main__":
806     stu_screen()
```