```
from distutils import command
from tkinter import *
from tkinter import ttk
from typing_extensions import _AnnotatedAlias
from PIL import Image,ImageTk
import os
import pickle
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 regist_emp


def emp_screen():
    def click_delete_employee():
        """when clicked delete employees, it will require to select the employees and after selecting and
        performing the delete method, it will ask the admin either they are sure they want to delete that employee
        or not if yes then employee containing that id in employee 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 = employee_tree.focus()
            tree_view_items = employee_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 employee having id {tree_view_values}")
            if ask is True:
                query = f"delete from employees where employee_id={tree_view_values};"
                mycur.execute(query)
                spec.commit()
                #db_connection.delete(query, (tree_view_values,))
                messagebox.showinfo("Success", f" Employee 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")
```

```
58
59
60
61      def update_emp():
62          def tree_event_handle():
63              try:
64                  #obj_student_database = Model_class.student_registration.GetDatabase('use cms
    ;')
65                  #db_connection.create(obj_student_database.get_database())
66
67                  tree_view_content = employee_tree.focus()
68                  tree_view_items = employee_tree.item(tree_view_content)
69                  # print(tree_view_items)
70                  tree_view_values = tree_view_items['values']
71                  tree_view_id = tree_view_items['values'][0]
72                  # print("tree",tree_view_id)
73                  get_id.clear()
74                  get_id.append(tree_view_id)
75                  list_of_tree.clear()
76                  # print("Appended", ManageEmployee.get_id)
77                  for i in tree_view_values:
78                      list_of_tree.append(i)
79
80                  # print(ManageEmployee.list_of_tree)
81                  # print(tree_view_values)
82
83                  ask = messagebox.askyesno("Confirm",
84                                            f"Do you want to Update employee having id {
    tree_view_id}")
85                  if ask is True:
86                      #update()
87                      pass
88
89              except BaseException as msg:
90                  print(msg)
91                  messagebox.showerror("Error",
92                                       "There is some error updating the data\n Make sure you
    have Selected the data")
93
94          tree_event_handle()
95          def reg_as():
96              """ to validate and register employee as different roles"""
97              global department_combo
98              try:
99                  #obj_employee_database = Model_class.employee_registration.GetDatabase('use
    cms;')
100                 #db_connection.create(obj_employee_database.get_database())
101                 a=load_data()
102                 host=a[0]
103                 username = a[2]
104                 password = a[3]
105                 port=a[1]
106
107                 spec=sql.connect(host=host,user=username,password=password,port=port,database=
    "sms")
108                 mycur=spec.cursor()
109
110                 query = "select * from department"
111                 mycur.execute(query)
112
113                 department_tuple = mycur.fetchall()
114                 # print(department_tuple)
115                 department_list = []
```

```
116                 for i in department_tuple:
117                     department_name = i[2]
118                     department_list.append(department_name)
119                     # print(department_name)
120                     # print(department_list)
121 
122             except BaseException as msg:
123                 print(msg)
124             # ============================================================================
125             # ============================Department=============================
126             # ============================================================================
127             department_label = Label(personal_frame, text="Department ", bg="white", fg="#
    4f4e4d",
128                                                 font=("yu gothic ui", 13, "bold"))
129             department_label.place(x=370, y=170)  # (x=370, y=130) (x=370, y=170)
130 
131             department_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'),
    state='readonly',
132                                                     width=31)
133 
134             department_combo['values'] = department_list
135             try:
136                 department_combo.current(0)
137             except:
138                 messagebox.showerror("Error", "You must add Department first")
139             department_combo.place(x=605, y=437)
140 
141         def update():
142             """updates the data of employees from entry fields"""
143             try:
144                 #obj_student_database = Model_class.student_registration.GetDatabase('use cms
    ;')
145                 #db_connection.create(obj_student_database.get_database())
146                 a=load_data()
147                 host=a[0]
148                 username = a[2]
149                 password = a[3]
150                 port=a[1]
151 
152                 spec=sql.connect(host=host,user=username,password=password,port=port,database=
    "sms")
153                 mycur=spec.cursor()
154 
155                 #get_id = get_id
156                 data_id= get_id[0]
157                 # print(data_id)
158                 #obj_employee_database = Model_class.employee_registration.
    EmployeeRegistration(username_entry.get(),
159                                                                                         #
    email_entry.get(),
160                                                                                         #
    email_entry.get(),
161                                                                                         #
    f_name_entry.get(),
162                                                                                         #
    l_name_entry.get(),
163                                                                                         #
    dob_entry.get(),
164                                                                                         #
    gender_combo.get(),
165                                                                                         #
    address_entry.get(),
166                                                                                         #
```

```
166 contact_entry.get(),
167                                                                                 #
    job_type_combo.get(),
168                                                                                 #
    register_as_combo.get(),
169                                                                                 #
    qualification_entry.get(),
170                                                                                 #
    department_combo.get(),
171                                                                                 #
    reg_date)
172                 query = f"update employees 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()}'," \
173                         f"job_type='{job_type_combo.get()}', registered_as='{register_as_combo
    .get()}', qualification='{qualification_entry.get()}', department='{department_combo.get()}'
    where employee_id={data_id}"
174                 mycur.execute(query)
175                 spec.commit()
176                 #values = (obj_employee_database.get_email(), obj_employee_database.get_f_name
    (),
177                         #obj_employee_database.get_l_name(), obj_employee_database.get_dob(),
178                         #obj_employee_database.get_gender(), obj_employee_database.get_address
    (),
179                         #obj_employee_database.get_contact(), obj_employee_database.get_job(),
180                         #obj_employee_database.get_reg_as(), obj_employee_database.
    get_qualification(),
181                         #obj_employee_database.get_department(), data_id)
182
183                 #db_connection.update(query, values)
184
185                 ask=messagebox.askyesnocancel("Success", f"Data having \n Email={email_entry.
    get()} \n Updated Successfully\n"
186                                                       f"Do you want to Go Employee Dashboard")
187
188                 click_view_all()
189                 if ask is True:
190                     #win = Toplevel()
191                     #Frontend.manage_employee.ManageEmployee(win)
192                     #window.withdraw()
193                     #win.deiconify()
194                     pass
195
196             except BaseException as msg:
197                 print(msg)
198                 messagebox.showerror("Error",f"Error due to{msg}")
199
200         global username_entry,password_entry,email_entry,f_name_entry,l_name_entry,dob_entry,
    gender_combo,address_entry,contact_entry,job_type_combo,register_as_combo,qualification_entry,
    department_combo
201
202
203         root = Toplevel()
204         root.title('EMPLOYEE MANAGEMENT SYSTEM - COLLEGE MANAGEMENT SYSTEM')
205         root.geometry('1067x600')
206         root.config(bg="#f29844")
207
208         # ======================Backend connection=============
209         #db_connection = Backend.connection.DatabaseConnection()
210
211         reg_frame = Frame(root, bg="#ffffff", width=1000, height=560)
212         reg_frame.place(x=30, y=30)
213
```

```
214
215         heading = Label(reg_frame, text="Employee Registration Form", font=('yu gothic ui', 20
    , "bold"), bg="white",
216                                 fg='black',
217                                 bd=5,
218                                 relief=FLAT)
219         heading.place(x=200, y=0, width=600)
220         #slider()
221         #heading_color()
222
223         cred_frame = LabelFrame(reg_frame, text="Account Details", bg="white", fg="#4f4e4d",
    height=140,
224                                         width=800, borderwidth=2.4,
225                                         font=("yu gothic ui", 13, "bold"))
226         cred_frame.config(highlightbackground="red")
227         cred_frame.place(x=100, y=50)
228         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"))
229         info.place(x=20,y=80)
230         # ============================================================================
231         # ============================Username========================================
232         # ============================================================================
233
234         username_label = Label(cred_frame, text="Username ", bg="white", fg="#4f4e4d",
235                                     font=("yu gothic ui", 13, "bold"))
236         username_label.place(x=10, y=10)
237
238         username_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
239                                     font=("yu gothic ui semibold", 12))
240         username_entry.place(x=230, y=117, width=260)  # trebuchet ms
241
242         username_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0
    )
243         username_line.place(x=230, y=139)
244
245         # ============================================================================
246         # ============================Email===========================================
247         # ============================================================================
248
249         email_label = Label(cred_frame, text="Email ", bg="white", fg="#4f4e4d",
250                                     font=("yu gothic ui", 13, "bold"))
251         email_label.place(x=370, y=10)
252
253         email_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
254                                     font=("yu gothic ui semibold", 12))
255         email_entry.place(x=555, y=117, width=350)  # trebuchet ms
256
257         email_line = Canvas(root, width=350, height=1.5, bg="#bdb9b1", highlightthickness=0)
258         email_line.place(x=555, y=139)
259
260         # ============================================================================
261         # ============================Password========================================
262         # ============================================================================
263
264         password_label = Label(cred_frame, text="Password ", bg="white", fg="#4f4e4d",
265                                     font=("yu gothic ui", 13, "bold"))
266         password_label.place(x=10, y=50)
267
268         password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
269                                     font=("yu gothic ui semibold", 12),show = "*")
270         password_entry.place(x=230, y=157, width=260)  # trebuchet ms
```

```
271
272         password_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0
    )
273         password_line.place(x=230, y=180)
274
275         # ============================================================================
276         # ============================Confirm password============================
277         # ============================================================================
278
279         c_password_label = Label(cred_frame, text="Confirm Password ", bg="white", fg="#4f4e4d
    ",
280                                  font=("yu gothic ui", 13, "bold"))
281         c_password_label.place(x=370, y=50)
282
283         c_password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
284                                  font=("yu gothic ui semibold", 12), show = "*")
285         c_password_entry.place(x=650, y=157, width=255)  # trebuchet ms
286
287         c_password_line = Canvas(root, width=255, height=1.5, bg="#bdb9b1", highlightthickness
    =0)
288         c_password_line.place(x=650, y=180)
289
290         # ============================================================================
291         # ========================frame for personal credentails=====================
292         # ============================================================================
293
294         personal_frame = LabelFrame(reg_frame, text="Personal Details", bg="white", fg="#
    4f4e4d", height=265,
295                                     width=800, borderwidth=2.4,
296                                     font=("yu gothic ui", 13, "bold"))
297         personal_frame.config(highlightbackground="red")
298         personal_frame.place(x=100, y=210)
299
300
301         # ============================================================================
302         # ============================First name======================================
303         # ============================================================================
304         f_name_label = Label(personal_frame, text="First Name ", bg="white", fg="#4f4e4d",
305                              font=("yu gothic ui", 13, "bold"))
306         f_name_label.place(x=10, y=10)
307
308         f_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
309                              font=("yu gothic ui semibold", 12))
310         f_name_entry.place(x=235, y=277, width=260)  # trebuchet ms
311
312         f_name_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0)
313         f_name_line.place(x=235, y=299)
314
315         # ============================================================================
316         # ============================Last name=======================================
317         # ============================================================================
318
319         l_name_label = Label(personal_frame, text="Last Name ", bg="white", fg="#4f4e4d",
320                              font=("yu gothic ui", 13, "bold"))
321         l_name_label.place(x=370, y=10)
322
323         l_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
324                              font=("yu gothic ui semibold", 12))
325         l_name_entry.place(x=595, y=277, width=315)  # trebuchet ms
326
```

```
        l_name_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0)
        l_name_line.place(x=595, y=299)

        # ============================================================================
        # ============================DOB=============================================
        # ============================================================================

        dob_label = Label(personal_frame, text="DOB ", bg="white", fg="#4f4e4d",
                                font=("yu gothic ui", 13, "bold"))
        dob_label.place(x=10, y=50)

        dob_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
                                font=("yu gothic ui semibold", 12))
        dob_entry.insert(0, "mm/dd/yyyy")
        dob_entry.place(x=190, y=317, width=305)  # trebuchet ms
        #dob_entry.bind("<1>", pick_date)

        dob_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
        dob_line.place(x=190, y=339)

        # ============================================================================
        # ============================Gender==========================================
        # ============================================================================
        style = ttk.Style()

        # style.map('TCombobox', selectbackground=[('readonly', 'grey')])
        root.option_add("*TCombobox*Listbox*Foreground", '#f29844')

        gender_label = Label(personal_frame, text="Gender ", bg="white", fg="#4f4e4d",
                                    font=("yu gothic ui", 13, "bold"))
        gender_label.place(x=370, y=50)

        gender_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
                                            width=35)
        gender_combo['values'] = ('Male', 'Female', 'Rather not say')
        gender_combo.current(0)
        gender_combo.place(x=570, y=317)

        # gender_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0
    )
        # gender_line.place(x=595, y=369)
        #aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
        # ============================================================================
        # ============================Address=========================================
        # ============================================================================

        address_label = Label(personal_frame, text="Address ", bg="white", fg="#4f4e4d",
                                    font=("yu gothic ui", 13, "bold"))
        address_label.place(x=10, y=90)

        address_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69
    ",
                                    font=("yu gothic ui semibold", 12))
        address_entry.place(x=215, y=357, width=280)  # trebuchet ms

        address_line = Canvas(root, width=280, height=1.5, bg="#bdb9b1", highlightthickness=0)
        address_line.place(x=215, y=379)

        # ============================================================================
        # ============================Contact no======================================
        # ============================================================================

```

```
387         contact_label = Label(personal_frame, text="Contact No. ", bg="white", fg="#4f4e4d",
388                                     font=("yu gothic ui", 13, "bold"))
389         contact_label.place(x=370, y=90)
390 
391         contact_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69
    ",
392                                     font=("yu gothic ui semibold", 12))
393         contact_entry.place(x=605, y=357, width=305)  # trebuchet ms
394 
395         contact_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
396         contact_line.place(x=605, y=379)
397 
398         # ============================================================================
399         # ============================Job Type========================================
400         # ============================================================================
401 
402         job_type_label = Label(personal_frame, text="Job Type ", bg="white", fg="#4f4e4d",
403                                     font=("yu gothic ui", 13, "bold"))
404         job_type_label.place(x=10, y=130)
405 
406         job_type_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state=
    'readonly',
407                                       width=27)
408         job_type_list = ["Part time", "Full time"]
409         job_type_combo['values'] = job_type_list
410         job_type_combo.current(0)
411         job_type_combo.place(x=223, y=397)
412 
413         # ============================================================================
414         # ============================Register as=====================================
415         # ============================================================================
416 
417         register_as_label = Label(personal_frame, text="Register as ", bg="white", fg="#4f4e4d
    ",
418                                       font=("yu gothic ui", 13, "bold"))
419         register_as_label.place(x=370, y=130)
420 
421         register_as_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'),
    state='readonly',
422                                          width=31)
423 
424         register_as_list = ["Department Head", "Instructor", "Employee"]
425         register_as_combo['values'] = register_as_list
426         register_as_combo.current(2)
427         register_as_combo.place(x=605, y=397)
428         #register_as_combo.bind('<<ComboboxSelected>>', reg_as_event_handle)
429 
430         # ============================================================================
431         # ============================Qualification===================================
432         # ============================================================================
433 
434         qualification_label = Label(personal_frame, text="Qualification ", bg="white", fg="#
    4f4e4d",
435                                     font=("yu gothic ui", 13, "bold"))
436         qualification_label.place(x=10, y=170)
437 
438         qualification_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
439                                     font=("yu gothic ui semibold", 12))
440         qualification_entry.place(x=250, y=437, width=240)  # trebuchet ms
441 
442         qualification_line = Canvas(root, width=240, height=1.5, bg="#bdb9b1",
    highlightthickness=0)
```

```
443         qualification_line.place(x=250, y=460)
444 
445         reg_date = time.strftime("%Y/%m/%d")
446 
447         reg_as()
448 
449         # ============================================================================
450         # ============================Register options================================
451         # ============================================================================
452 
453         options_frame = LabelFrame(reg_frame, text="Register Options", bg="white", fg="#4f4e4d
    ", height=80,
454                                    width=800, borderwidth=2.4,
455                                    font=("yu gothic ui", 13, "bold"))
456         options_frame.config(highlightbackground="red")
457         options_frame.place(x=100, y=475)
458 
459         # ============================================================================
460         # ============================Register options================================
461         # ============================================================================
462 
463         submit_img = ImageTk.PhotoImage(file='Pics\\submit.png')
464         submit = Button(options_frame, image=submit_img,
465                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
466                                 , borderwidth=0, background="white", cursor="hand2",command=
    update)
467         submit.image = submit_img
468         submit.place(x=90, y=10)
469 
470         clear_img = ImageTk.PhotoImage(file='Pics\\clear.png')
471         clear_button = Button(options_frame, image=clear_img,
472                                     font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
473                                     , borderwidth=0, background="white", cursor="hand2")
474         clear_button.image = clear_img
475         clear_button.place(x=250, y=13)
476 
477         back_img = ImageTk.PhotoImage(file='Pics\\back.png')
478         back_button = Button(options_frame, image=back_img,
479                                     font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
480                                     , borderwidth=0, background="white", cursor="hand2")
481         back_button.image = back_img
482         back_button.place(x=410, y=13)
483 
484         exit_img = ImageTk.PhotoImage(file='Pics\\exit.png')
485         exit_button = Button(options_frame, image=exit_img,
486                                     font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
487                                     , borderwidth=0, background="white", cursor="hand2",
    command=exit)
488         exit_button.image = exit_img
489         exit_button.place(x=570, y=13)
490 
491         a = list_of_tree
492         print(a)
493         try:
494             #username_entry.insert(0,)
495             email_entry.insert(0, a[3])
496             f_name_entry.insert(0, a[1])
497             l_name_entry.insert(0, a[2])
498             dob_entry.delete(0, END)
```

```
499             dob_entry.insert(0, a[4])
500             gender_combo.set(a[5])
501             address_entry.insert(0, a[6])
502             contact_entry.insert(0, a[7])
503             job_type_combo.set(a[8])
504             register_as_combo.set(a[11])
505             qualification_entry.insert(0,a[10])
506             department_combo.set(a[9])
507         except IndexError as msg:
508             print(msg)
509 
510 
511     def load_data():
512         f=open("Credentials.csv","r")
513         s=csv.reader(f,delimiter="-")
514         d=[]
515         for i in s:
516             d.append(i)
517         a=d[::-1]
518         return (a[0])
519 
520     def click_go_to_dashboard():
521         """returns AdminDashboard class when clicked go to dashboard"""
522         dashboard.dashboard()
523         root.withdraw()
524 
525     def click_view_all():
526         """it will show all the data contains on the employee table of cms database, when
    clicked by default this method
527         is called while initializing the class ManageEmployee. Exception is handled to avoid
    run time error which may
528         cause by user."""
529         try:
530             #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
531             #db_connection.create(obj_student_database.get_database())
532             a=load_data()
533             host=a[0]
534             username = a[2]
535             password = a[3]
536             port=a[1]
537 
538             spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
    ")
539             mycur=spec.cursor()
540             query = "select * from employees;"
541             mycur.execute(query)
542             data = mycur.fetchall()
543             # print(data)
544             employee_tree.delete(*employee_tree.get_children())
545             for values in data:
546                 data_list = [values[0], values[4], values[5], values[2], values[6], values[7],
547                              values[8],
548                              values[9], values[10], values[13], values[12], values[11],
    values[14]]
549                 # print(data_list)
550                 employee_tree.insert('', END, values=data_list)
551 
552         except BaseException as msg:
553             print(msg)
554 
555     list_of_tree = []
556     get_id = []
557 
```

```
558     root = Toplevel()
559     root.geometry("1067x600")
560     root.title("Manage Employee - College Management System")
561     #root.iconbitmap('images\\logo.ico')
562     root.resizable(False, False)
563 
564 
565     manage_student_frame_r = Image.open('Pics\\student_frame.png').resize((1067,600),Image.
    ANTIALIAS)
566     manage_student_frame = ImageTk.PhotoImage(manage_student_frame_r)
567     image_panel = Label(root, image=manage_student_frame)
568     image_panel.image = manage_student_frame
569     image_panel.pack(fill='both', expand='yes')
570 
571     #db_connection = Backend.connection.DatabaseConnection()
572 
573 
574     heading = Label(root, text="Employee Management Dashboard", font=('yu gothic ui', 20, "
    bold"), bg="white",
575                             fg='black',
576                             bd=5,
577                             relief=FLAT)
578     heading.place(x=420, y=26, width=640)
579 
580 
581     #left frame
582     left_view_frame = Frame(root, bg="white")
583     left_view_frame.place(x=35, y=89, height=470, width=250)
584 
585 
586     #tree view frame
587     tree_view_frame = Frame(root, bg="white")
588     tree_view_frame.place(x=301, y=90, height=473, width=730)
589 
590 
591     # ================================================================================
592     # ========================frame for personal credentials ==================
593     # ================================================================================
594 
595     personal_frame = LabelFrame(left_view_frame, text="Employee Management Options", bg="white
    ", fg="#4f4e4d",
596                                 height=460,
597                                 width=240, borderwidth=2.4,
598                                 font=("yu gothic ui", 13, "bold"))
599     personal_frame.config(highlightbackground="red")
600     personal_frame.place(x=5, y=8)
601 
602     # ================================================================================
603     # ============================Add employee button================================
604     # ================================================================================
605 
606     #add_admin = ImageTk.PhotoImage(file='images\\add_admin.png')
607     #add_admin_button = Button(personal_frame, image=add_admin, relief=FLAT, borderwidth=0,
608                                         #activebackground="white", bg="white", cursor="hand2")
609     #add_admin_button.place(x=8, y=60)
610 
611     # ================================================================================
612     # ============================Add employee button================================
613     # ================================================================================
614 
615     add_employee_r = Image.open('Pics\\add_employee.png').resize((225,25),Image.ANTIALIAS)
616     add_employee = ImageTk.PhotoImage(add_employee_r)
617     add_employee_button = Button(personal_frame, image=add_employee, relief=FLAT, borderwidth=
```

```
0,
                                        activebackground="white", bg="white", cursor="hand2",
command=regist_emp.regist_emp)
    add_employee_button.image = add_employee
    add_employee_button.place(x=8, y=100)
    # add_employee_button.place(x=36, y=295)

    # ========================================================================
    # ============================Update employee button=============================
    # ========================================================================

    update_employee_r = Image.open('Pics\\update_employee.png').resize((225,25),Image.
ANTIALIAS)
    update_employee = ImageTk.PhotoImage(update_employee_r)
    update_employee_button = Button(personal_frame, image=update_employee, relief=FLAT,
borderwidth=0,
                                            activebackground="white", bg="white", cursor="
hand2",command= update_emp)
    update_employee_button.image = update_employee
    update_employee_button.place(x=8, y=165)
    # update_employee_button.place(x=36, y=355)

    # ========================================================================
    # ============================Delete employee button=============================
    # ========================================================================

    delete_employee_r = Image.open('Pics\\delete_employee.png').resize((225,25),Image.
ANTIALIAS)
    delete_employee = ImageTk.PhotoImage(delete_employee_r)
    delete_employee_button = Button(personal_frame, image=delete_employee, relief=FLAT,
borderwidth=0,
                                            activebackground="white", bg="white", cursor="
hand2",command=click_delete_employee)
    delete_employee_button.image = delete_employee
    delete_employee_button.place(x=8, y=230)
    # delete_employee_button.place(x=36, y=405)


    # ========================================================================
    # ============================Goto Main dashboard button=============================
    # ========================================================================

    goto_dashboard_r = Image.open('Pics\\goto_dashboard.png').resize((225,25),Image.ANTIALIAS)
    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")

    scroll_x = Scrollbar(tree_view_frame, orient=HORIZONTAL)
    scroll_y = Scrollbar(tree_view_frame, orient=VERTICAL)
    employee_tree = ttk.Treeview(tree_view_frame,
                                        columns=(
                                            "EMPLOYEE ID", "FNAME", "LNAME", "EMAIL", "DOB", "
GENDER", "ADDRESS",
```

```
                                                "CONTACT NO", "JOB TYPE", "DEPARTMENT", "
    QUALIFICATION", "REGISTERED AS",
                                                "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=employee_tree.xview)
        scroll_y.config(command=employee_tree.yview)

        # ==========================TreeView Heading====================
        employee_tree.heading("EMPLOYEE ID", text="EMPLOYEE ID")
        employee_tree.heading("FNAME", text="FNAME")
        employee_tree.heading("LNAME", text="LNAME")
        employee_tree.heading("EMAIL", text="EMAIL")
        employee_tree.heading("DOB", text="DOB")
        employee_tree.heading("GENDER", text="GENDER")
        employee_tree.heading("ADDRESS", text="ADDRESS")
        employee_tree.heading("CONTACT NO", text="CONTACT NO")
        employee_tree.heading("JOB TYPE", text="JOB TYPE")
        employee_tree.heading("DEPARTMENT", text="DEPARTMENT")
        employee_tree.heading("QUALIFICATION", text="QUALIFICATION")
        employee_tree.heading("REGISTERED AS", text="REGISTERED AS")
        employee_tree.heading("REGISTRATION DATE", text="REGISTRATION DATE")
        employee_tree["show"] = "headings"

        # ==========================TreeView Column====================
        employee_tree.column("EMPLOYEE ID", width=150)
        employee_tree.column("FNAME", width=170)
        employee_tree.column("LNAME", width=170)
        employee_tree.column("EMAIL", width=200)
        employee_tree.column("ADDRESS", width=150)
        employee_tree.column("GENDER", width=150)
        employee_tree.column("CONTACT NO", width=150)
        employee_tree.column("DOB", width=150)
        employee_tree.column("JOB TYPE", width=150)
        employee_tree.column("DEPARTMENT", width=150)
        employee_tree.column("QUALIFICATION", width=150)
        employee_tree.column("REGISTERED AS", width=150)
        employee_tree.column("REGISTRATION DATE", width=150)
        employee_tree.pack(fill=BOTH, expand=1)
        # emp_tree.bind("<ButtonRelease-1>", getcredon_click)
        #employee_tree.bind("<Button-3>", do_popup)
        #employee_tree.bind("<Delete>", click_delete_key)
        #employee_tree.bind("<Return>", tree_double_click)
        #employee_tree.bind("<Double-Button-1>", tree_double_click)
        #search_start()
        #search_by.current(0)
        click_view_all()

        #root.mainloop()

if __name__ == "__main__":
        emp_screen()
```