```python
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_depart

def depart_screen():
    def click_delete_department():
        """when clicked delete departments, it will require to select the departments and after selecting and
        performing the delete method, it will ask the admin either they are sure they want to delete that department
        or not if yes then department containing that id in department table is deleted."""
        try:
            #obj_department_registration_database = Model_class.department_registration.GetDatabase('use cms;')
            #db_connection.create(obj_department_registration_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()

            department_view_content = department_tree.focus()
            department_view_items = department_tree.item(department_view_content)
            department_view_values = department_view_items['values'][0]
            ask = messagebox.askyesno("Warning",
                                        f"Are you sure you want to delete department having id {department_view_values}")
            if ask is True:
                query = f"delete from department where department_id={department_view_values};"
                mycur.execute(query)
                spec.commit()
                #db_connection.delete(query, (department_view_values,))
                messagebox.showinfo("Success", f" department id {department_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 click_clear_button():
        department_code_entry.delete(0, END)
        department_name_entry.delete(0, END)


    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_go_to_dashboard():
        """returns AdminDashboard class when clicked go to dashboard"""
        dashboard.dashboard()
        root.withdraw()

    def click_view_all():
            """it will show all the data contains on the department table of cms database, when clicked by default this method
            is called while initializing the class ManageDepartment. 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 department;"
                mycur.execute(query)

                data = mycur.fetchall()
                # print(data)
                department_tree.delete(*department_tree.get_children())
                for values in data:
                    data_list = [values[0], values[1], values[2], values[3]]
                    # print(data_list)
                    department_tree.insert('', END, values=data_list)

            except BaseException as msg:
                print(msg)


    def updatebutton():

        def tree_event_handle():
            try:
                #obj_department_database = Model_class.department_registration.GetDatabase('use cms;')
                #db_connection.create(obj_department_database.get_database())

                tree_view_content = department_tree.focus()
                tree_view_items = department_tree.item(tree_view_content)
                # print(tree_view_items)
                tree_view_values = tree_view_items['values']
                tree_view_id = tree_view_items['values'][0]
                list_of_tree.clear()
                # print(tree_view_id)
                get_id.clear()
                get_id.append(tree_view_id)
                for i in tree_view_values:
                    list_of_tree.append(i)

                ask = messagebox.askyesno("Confirm",
                                        f"Do you want to Update Student having id {tree_view_id}")
                if ask is True:
                    pass

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

        tree_event_handle()

        def up_dep():
            try:
                #obj_department_database = Model_class.department_registration.GetDatabase('use cms;')
                #db_connection.create(obj_department_database.get_database())

                #get_id = ManageDepartment.get_id
                data_id = get_id[0]
                # print(data_id)

                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()
                #obj_department_database = Model_class.department_registration.DepartmentRegistration(
                    #department_code_entry.get(),
                    #department_name_entry.get(),
                    #reg_date)

                query = f"update department set department_code='{department_code_entry.get()}',department_name='{department_name_entry.get()}' where department_id={data_id}"
                mycur.execute(query)
                spec.commit()
                click_view_all()

                #values = (obj_department_database.get_code(), obj_department_database.get_name(), data_id)

                #db_connection.update(query, values)

                ask = messagebox.askyesnocancel("Success",
                                                f"Data having \n department Name={department_code_entry.get()} \n Updated Successfully\n"
                                                f"Do you want to Go Department Dashboard")
                if ask is True:
                    pass
                    #win = Toplevel()
                    #Frontend.manage_department.ManageDepartment(win)
                    #window.withdraw()
                    #win.deiconify()

            except BaseException as msg:
                print(msg)
                messagebox.showerror("Error", f"Error due to{msg}")



        a = list_of_tree
        # print(a)


        global department_code_entry,department_name_entry

        root = Toplevel()
        root.title('DEPARTMENT REGISTRATION FORM - COLLEGE MANAGEMENT SYSTEM')
        root.geometry('1067x600')
        root.config(bg="#f29844")
        root.resizable(False,False)

        # ======================Backend connection=============
        #db_connection = Backend.connection.DatabaseConnection()

        # creating frame for Register
        # img = img
        # dummylabel = Label(root, image=img)
        # dummylabel.place(x=30, y=30)

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




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

        dept_frame = LabelFrame(reg_frame, text="Department Details", bg="white", fg="#4f4e4d", height=380,
                                        width=800, borderwidth=2.4,
                                        font=("yu gothic ui", 13, "bold"))
        dept_frame.config(highlightbackground="red")
        dept_frame.place(x=100, y=90)

        # ===============================================================
        # ===========================Key Bindings==================================
        # ===============================================================

        #root.bind("<Return>", click_enter_submit)
```

```python

        # ========================================================================
        # ============================Department Code label ====================================
        # ========================================================================

        department_code_label = Label(dept_frame, text="Department Code ", bg="white", fg="#4f4e4d",
                                            font=("yu gothic ui", 13, "bold"))
        department_code_label.place(x=160, y=80)

        department_code_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
                                            font=("yu gothic ui semibold", 12))
        department_code_entry.place(x=450, y=227, width=295)  # trebuchet ms

        department_code_line = Canvas(root, width=295, height=1.5, bg="#bdb9b1", highlightthickness=0)
        department_code_line.place(x=450, y=249)

        # ========================================================================
        # ============================Department Name=======================================
        # ========================================================================

        department_name_label = Label(dept_frame, text="Department Name ", bg="white", fg="#4f4e4d",
                                            font=("yu gothic ui", 13, "bold"))
        department_name_label.place(x=160, y=140)

        department_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
                                            font=("yu gothic ui semibold", 12))
        department_name_entry.place(x=450, y=287, width=295)  # trebuchet ms

        department_name_line = Canvas(root, width=295, height=1.5, bg="#bdb9b1", highlightthickness=0)
        department_name_line.place(x=450, y=309)

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



        # ========================================================================
        # ============================Register options=====================================
        # ========================================================================
        submit_img = ImageTk.PhotoImage(file='Pics\\submit.png')
        submit = Button(dept_frame, image=submit_img,
                                font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="white"
                                , borderwidth=0, background="white", cursor="hand2",command=up_dep)
        submit.image = submit_img
        submit.place(x=90, y=267)

        clear_img = ImageTk.PhotoImage(file='Pics\\clear.png')
        clear_button = Button(dept_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=270)

        back_img = ImageTk.PhotoImage(file='Pics\\back.png')
        back_button = Button(dept_frame, image=back_img,
                                    font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="white"
                                    , borderwidth=0, background="white", cursor="hand2")
        back_button.image = back_img
        back_button.place(x=410, y=270)


        exit_img = ImageTk.PhotoImage(file='Pics\\exit.png')
        exit_button = Button(dept_frame, image=exit_img,
                                    font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="white"
                                    , borderwidth=0, background="white", cursor="hand2", command=exit)
        exit_button.image = exit_img
        exit_button.place(x=570, y=270)

        try:
            department_code_entry.insert(0, a[1])
            department_name_entry.insert(0, a[2])

        except IndexError as msg:
            print(msg)


    list_of_tree = []
    get_id = []

    root = Toplevel()
    root.geometry("1067x600")
    root.title("Manage Department - College Management System")
    #root.iconbitmap('images\\logo.ico')
    root.resizable(False, False)


    manage_student_frame_r = Image.open('Pics\\student_frame.png').resize((1067,600),Image.ANTIALIAS)
    manage_student_frame = ImageTk.PhotoImage(manage_student_frame_r)
    image_panel = Label(root, image=manage_student_frame)
    image_panel.image = manage_student_frame
    image_panel.pack(fill='both', expand='yes')



    heading = Label(root, text="Department Management Dashboard", font=('yu gothic ui', 20, "bold"), bg="white",
                            fg='black',
                            bd=5,
                            relief=FLAT)
    heading.place(x=420, y=23)


    # ==================================================================
    # =======================Left frame ===============================
    # ==================================================================

    #left frame
    left_view_frame = Frame(root, bg="white")
    left_view_frame.place(x=35, y=89, height=470, width=250)


    #tree view frame
    tree_view_frame = Frame(root, bg="white")
    tree_view_frame.place(x=301, y=90, height=473, width=730)

    # =======================================================================
    # ========================frame for personal credentials =================
    # =======================================================================

    personal_frame = LabelFrame(left_view_frame, text="Batch Management Options", bg="white", fg="#4f4e4d",height=460,width=240, borderwidth=2.4,font=("yu gothic ui", 12, "bold"))
    personal_frame.config(highlightbackground="red")
    personal_frame.place(x=5, y=8)


    # ========================================================================
    # ============================Add Department button===============================
    # ========================================================================

    add_department_r = Image.open('Pics\\add_department.png').resize((225,25),Image.ANTIALIAS)
    add_department = ImageTk.PhotoImage(add_department_r)
    add_department_button = Button(personal_frame, image=add_department, relief=FLAT, borderwidth=0,
                                        activebackground="white", bg="white", cursor="hand2", command= regist_depart.regist_depart)
    add_department_button.image = add_department
    add_department_button.place(x=8, y=100)
    # add_student_button.place(x=36, y=295)

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

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

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

    delete_department_r = Image.open('Pics\\delete_department.png').resize((225,25),Image.ANTIALIAS)
    delete_department = ImageTk.PhotoImage(delete_department_r)
    delete_department_button = Button(personal_frame, image=delete_department, relief=FLAT, borderwidth=0,
                                            activebackground="white", bg="white", cursor="hand2",command=click_delete_department)
    delete_department_button.image = delete_department
    delete_department_button.place(x=8, y=230)
    # delete_student_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)
    department_tree = ttk.Treeview(tree_view_frame,
                                        columns=(
                                            "DEPARTMENT ID", "DEPARTMENT CODE", "DEPARTMENT NAME", "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=department_tree.xview)
    scroll_y.config(command=department_tree.yview)

    # ==========================TreeView Heading====================
    department_tree.heading("DEPARTMENT ID", text="DEPARTMENT ID")
    department_tree.heading("DEPARTMENT CODE", text="DEPARTMENT CODE")
    department_tree.heading("DEPARTMENT NAME", text="DEPARTMENT NAME")
    department_tree.heading("REGISTRATION DATE", text="REGISTRATION DATE")
    department_tree["show"] = "headings"

    # ==========================TreeView Column====================
    department_tree.column("DEPARTMENT ID", width=100)
    department_tree.column("DEPARTMENT CODE", width=100)
    department_tree.column("DEPARTMENT NAME", width=170)
    department_tree.column("REGISTRATION DATE", width=100)
    department_tree.pack(fill=BOTH, expand=1)

    #department_tree.bind("<Delete>", click_delete_key)
    #department_tree.bind("<Button-3>", do_popup)
    #department_tree.bind("<Double-Button-1>", tree_double_click)
    #department_tree.bind("<Return>", tree_double_click)
    #search_start()
    #search_by.current(0)
    click_view_all()

    #root.mainloop()


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