2274802010694 – TRẦN MINH PHÚC

PROJECT 2

I/ Giao diện

1. Bố cục chung:

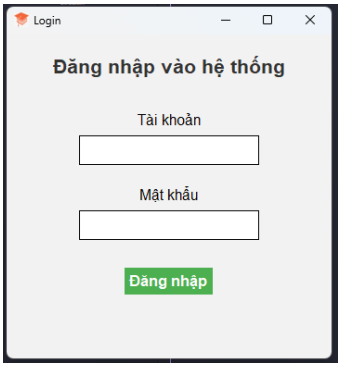
- Giao diện của dự án được xây dựng dựa trên mô hình MVC (Model-View-Controller). Điều này giúp tách biệt logic xử lý, giao diện người dùng và dữ liệu, tạo điều kiện thuận lợi cho việc bảo trì và phát triển.
- Giao diện chính được điều khiển bởi file View.py, trong đó chứa các thành phần giao diện như:
    - Các nút chức năng (insert, delete, update).
    - Danh sách Treeview hiển thị thông tin sinh viên (hoặc các đối tượng quản lý khác tùy vào bài toán).
    - Form nhập liệu để thêm hoặc chỉnh sửa thông tin.
- Phần main.py sẽ chịu trách nhiệm khởi tạo và quản lý phiên làm việc với giao diện.

2. Các thành phần chính của giao diện:

- **Form nhập liệu:** Cho phép người dùng nhập các thông tin cần thiết (như mã sinh viên, tên, email, và các thông tin khác).
- **Danh sách Treeview:** Hiển thị danh sách các đối tượng quản lý, bao gồm các cột như mã, tên, loại, trạng thái, v.v. Người dùng có thể thực hiện các thao tác cập nhật, xóa trên các bản ghi.
- **Nút chức năng:** Các nút như "Lưu", "Hủy" thực hiện chức năng xử lý thêm mới hoặc hủy bỏ hành động.

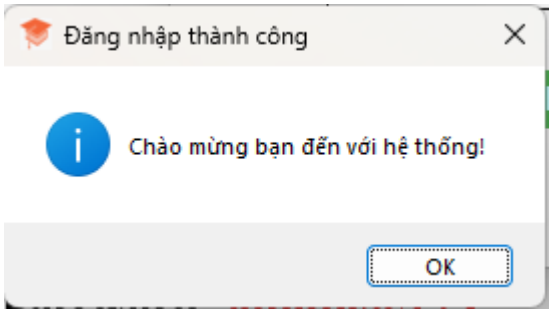
3. Giao diện

Chạy chương trình:



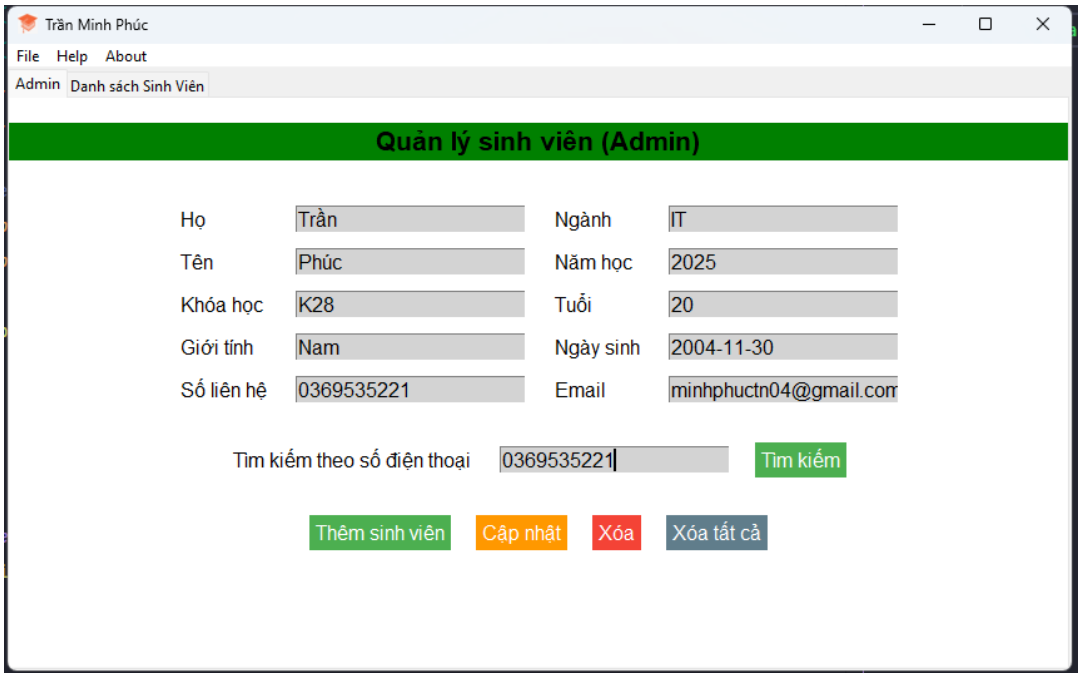
Nhập tài khoản: admin – Password: admin@123

Sau khi đăng nhập sẽ hiển thị thông báo:

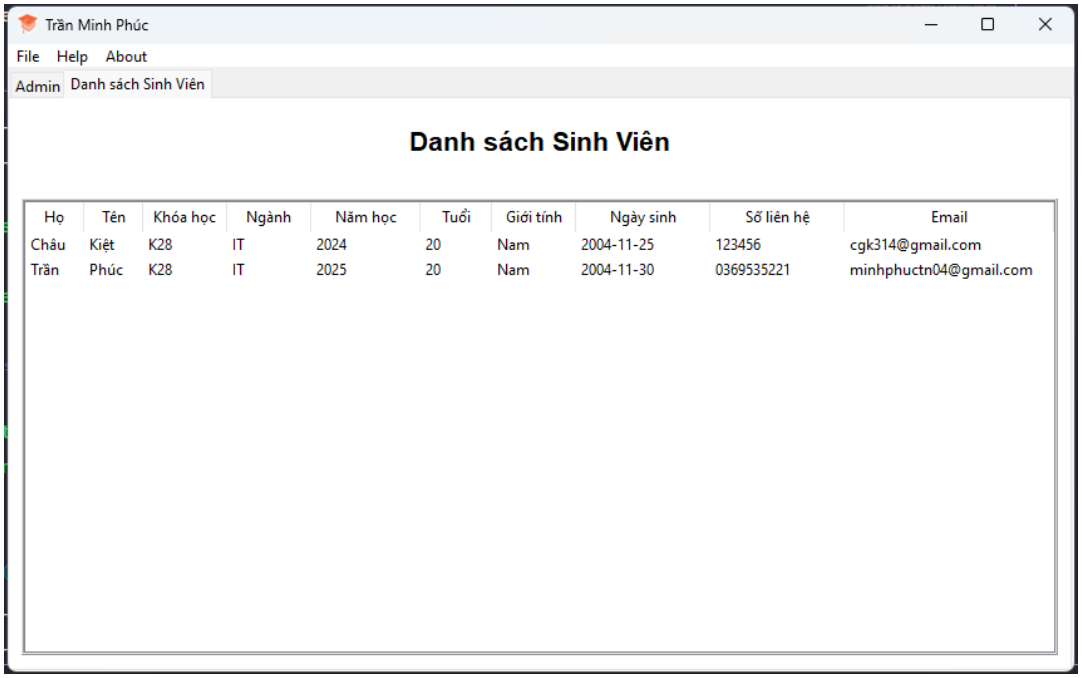


Khi bấm Ok sẽ hiển thị ra chương trình hệ thống:

Khi tìm kiếm sinh viên bằng số điện thoại, hệ thống sẽ hiển thị thêm 2 nút cập nhật và xóa:

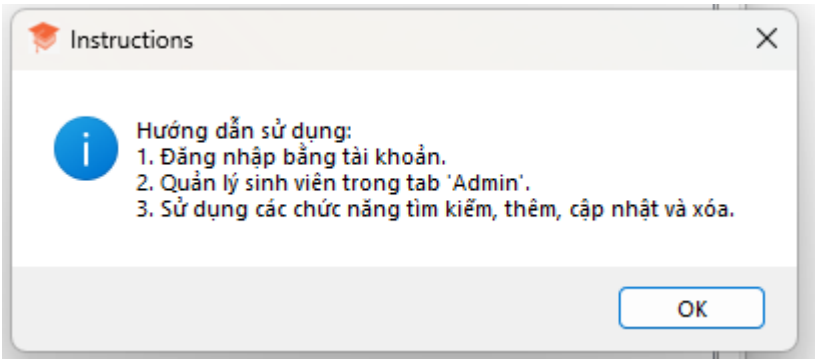


Tab2: TreeView để xem danh sách sinh viên

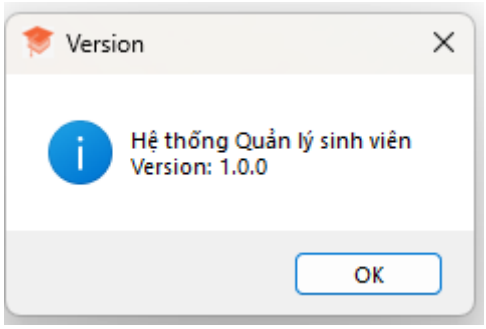


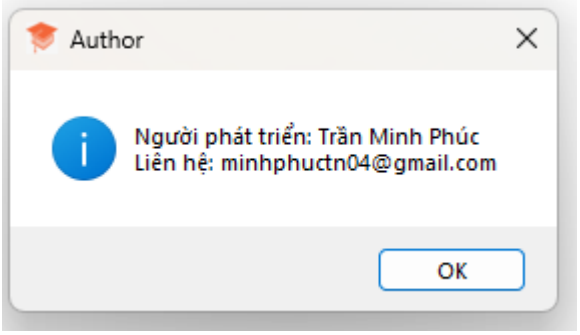
Menu:

+ Instructions



+ About/ Version và Author





II/ Chức năng

1. Chức năng chính:

- **Thêm mới đối tượng:**
    - Khi người dùng nhập đầy đủ thông tin và nhấn nút "Lưu", hệ thống sẽ gọi phương thức trong Controller.py để thêm đối tượng vào cơ sở dữ liệu.
- **Chỉnh sửa thông tin sinh viên:**
    - Cho phép người dùng chọn một đối tượng từ danh sách Treeview, sau đó sửa thông tin trực tiếp trên form và nhấn "Lưu" để cập nhật.
- **Xóa sinh viên:**
    - Khi chọn một đối tượng trong danh sách và nhấn nút "Xóa", hệ thống sẽ thực hiện thao tác xóa đối tượng khỏi cơ sở dữ liệu.
- **Tìm kiếm sinh viên:**
    - Chức năng tìm kiếm cho phép người dùng nhập số điện thoại để lọc danh sách sinh viên.

2. Cấu trúc mô hình MVC:

- **Model.py:** Quản lý các thao tác liên quan đến cơ sở dữ liệu như truy vấn, thêm, xóa, cập nhật thông tin.
- **View.py:** Quản lý giao diện và tương tác với người dùng.
- **Controller.py:** Xử lý logic chính, kết nối giữa giao diện và dữ liệu. Điều khiển các thao tác từ giao diện và cập nhật lại dữ liệu sau khi có thay đổi.

III/ Mã chương trình

Controller.py

```python
from Model import StudentModel
import psycopg2

class StudentController:
    Tabnine | Edit | Test | Explain | Document | Ask
    def __init__(self):
        pass

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def login(username, password):
        try:
            conn = psycopg2.connect(
                host='127.0.0.1',
                user='postgres',
                password='123456',
                dbname='student_management'
            )
            cursor = conn.cursor()
            cursor.execute("SELECT * FROM users WHERE username = %s AND password = %s", (username, password))
            result = cursor.fetchone()
            conn.close()
            return result
        except Exception as e:
            raise Exception(f"Lỗi kết nối: {str(e)}")

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def get_all_students():
        return StudentModel.get_all_students()

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def search_student_by_phone(phone):
        return StudentModel.search_student_by_phone(phone)

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def add_student(student_data):
        StudentModel.add_student(student_data)

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def update_student(student_data, phone):
        StudentModel.update_student(student_data, phone)

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def delete_student(phone):
        StudentModel.delete_student(phone)
```

Model.py

```python
import psycopg2

class StudentModel:
    @staticmethod
    def get_all_students():
        try:
            connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
            curs = connection.cursor()
            curs.execute("SELECT * FROM student_register")
            rows = curs.fetchall()
            connection.close()
            return rows
        except Exception as e:
            raise Exception(f"Lỗi khi tải danh sách sinh viên: {str(e)}")

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def search_student_by_phone(phone):
        try:
            connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
            curs = connection.cursor()
            curs.execute("SELECT * FROM student_register WHERE contact = %s", (phone,))
            row = curs.fetchone()
            connection.close()
            return row
        except Exception as e:
            raise Exception(f"Lỗi khi tìm sinh viên: {str(e)}")

    Tabnine | Edit | Test | Explain | Document | Ask
    @staticmethod
    def add_student(student_data):
        try:
            connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
            curs = connection.cursor()
            curs.execute("INSERT INTO student_register (f_name, l_name, course, subject, year, age, gender, birth, contact, email) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                         student_data)
            connection.commit()
            connection.close()
        except Exception as e:
            raise Exception(f"Không thể thêm sinh viên: {str(e)}")
```

```python
Tabnine | Edit | Test | Explain | Document | Ask
@staticmethod
def update_student(student_data, phone):
    try:
        connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
        curs = connection.cursor()
        curs.execute("""
            UPDATE student_register
            SET f_name=%s, l_name=%s, course=%s, subject=%s, year=%s, age=%s, gender=%s, birth=%s, email=%s
            WHERE contact=%s
            """,
            student_data + (phone,))
        connection.commit()
        connection.close()
    except Exception as e:
        raise Exception(f"Lỗi khi cập nhật sinh viên: {str(e)}")

Tabnine | Edit | Test | Explain | Document | Ask
@staticmethod
def delete_student(phone):
    try:
        connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
        curs = connection.cursor()
        curs.execute("DELETE FROM student_register WHERE contact=%s", (phone,))
        connection.commit()
        connection.close()
    except Exception as e:
        raise Exception(f"Lỗi khi xóa sinh viên: {str(e)}")

Tabnine | Edit | Test | Explain | Document | Ask
@staticmethod
def get_all_students():
    try:
        connection = psycopg2.connect(host='127.0.0.1', user='postgres', password='123456', database='student_management')
        curs = connection.cursor()
        curs.execute("SELECT * FROM student_register")
        rows = curs.fetchall()
        connection.close()
        return rows
    except Exception as e:
        raise Exception(f"Lỗi khi tải danh sách sinh viên: {str(e)}")
```

View.py

```python
from tkinter import *
from tkinter import messagebox
from tkinter.ttk import Treeview, Notebook
from Controller import StudentController
from Model import StudentModel

class Login:
    Tabnine | Edit | Test | Explain | Document | Ask
    def __init__(self, root):
        self.root = root
        self.root.title("Login")
        self.root.geometry("350x350")
        self.root.configure(bg="#f2f2f2")
        self.root.iconbitmap("./favicon.ico")

        Label(self.root, text="Đăng nhập vào hệ thống", font=("Helvetica", 16, "bold"), fg="#333333", bg="#f2f2f2").pack(pady=20)

        Label(self.root, text="Tài khoản", font=("Helvetica", 12), bg="#f2f2f2").pack(pady=(10, 0))
        self.username_entry = Entry(self.root, font=("Helvetica", 12), bd=1, relief=SOLID)
        self.username_entry.pack(pady=(5, 10), ipadx=5, ipady=5)

        Label(self.root, text="Mật khẩu", font=("Helvetica", 12), bg="#f2f2f2").pack(pady=(10, 0))
        self.password_entry = Entry(self.root, show="*", font=("Helvetica", 12), bd=1, relief=SOLID)
        self.password_entry.pack(pady=(5, 20), ipadx=5, ipady=5)

        login_button = Button(self.root, text="Đăng nhập", command=self.check_login, font=("Helvetica", 12, "bold"),
                              fg="white", bg="#4CAF50", activebackground="#45a049", cursor="hand2", bd=0)
        login_button.pack(pady=10)
        login_button.bind("<Enter>", lambda e: login_button.config(bg="#45a049"))
        login_button.bind("<Leave>", lambda e: login_button.config(bg="#4CAF50"))

    Tabnine | Edit | Test | Explain | Document | Ask
    def check_login(self):
        username = self.username_entry.get()
        password = self.password_entry.get()
        try:
            result = StudentController.login(username, password)
            if result:
                messagebox.showinfo("Đăng nhập thành công", "Chào mừng bạn đến với hệ thống!")
                self.root.destroy()
                self.open_management_form()
            else:
                messagebox.showerror("Đăng nhập thất bại", "Tên người dùng hoặc mật khẩu không hợp lệ.")
        except Exception as e:
            messagebox.showerror("Error", str(e))

    Tabnine | Edit | Test | Explain | Document | Ask
    def open_management_form(self):
        root = Tk()
        app = Management(root)
        root.mainloop()
```

```python
class Management:
    Tabnine | Edit | Test | Explain | Document | Ask
    def __init__(self, root):
        self.window = root
        self.window.title("Trần Minh Phúc")
        self.window.geometry("850x500")
        self.window.config(bg="white")
        self.window.iconbitmap("./favicon.ico")
        # Menu Bar
        self.menu_bar = Menu(self.window)
        self.window.config(menu=self.menu_bar)

        # File Menu
        file_menu = Menu(self.menu_bar, tearoff=0)
        file_menu.add_command(label="Logout", command=self.logout)
        file_menu.add_separator()
        file_menu.add_command(label="Exit", command=self.window.quit)
        self.menu_bar.add_cascade(label="File", menu=file_menu)

        # Help Menu
        help_menu = Menu(self.menu_bar, tearoff=0)
        help_menu.add_command(label="Instructions", command=self.show_instructions)
        self.menu_bar.add_cascade(label="Help", menu=help_menu)

        # About Menu
        about_menu = Menu(self.menu_bar, tearoff=0)
        about_menu.add_command(label="Version", command=self.show_version)
        about_menu.add_command(label="Author", command=self.show_author)
        self.menu_bar.add_cascade(label="About", menu=about_menu)

        # Tab Control
        self.tab_control = Notebook(self.window)
        self.tab_control.pack(expand=1, fill='both')

        # Tab 1 - Admin (Nhập liệu sinh viên)
        self.admin_tab = Frame(self.tab_control, bg="white")
        self.tab_control.add(self.admin_tab, text="Admin")
        self.create_admin_tab()

        # Tab 2 - Treeview
        self.treeview_tab = Frame(self.tab_control, bg="white")
        self.tab_control.add(self.treeview_tab, text="Danh sách Sinh Viên")
        self.create_treeview_tab()

    Tabnine | Edit | Test | Explain | Document | Ask
    def show_instructions(self):
        messagebox.showinfo("Instructions", "Hướng dẫn sử dụng:\n1. Đăng nhập bằng tài khoản.\n2. Quản lý sinh viên trong tab 'Admin'.\n3. Sử dụng các chức năng tìm kiếm, thêm, cập nhật và xóa.")

    Tabnine | Edit | Test | Explain | Document | Ask
    def show_version(self):
        messagebox.showinfo("Version", "Hệ thống Quản lý sinh viên\nVersion: 1.0.0")
```

```python
def show_author(self):
    messagebox.showinfo("Author", "Người phát triển: Trần Minh Phúc\nLiên hệ: minhphuctn04@gmail.com")

Tabnine | Edit | Test | Explain | Document | Ask
def create_admin_tab(self):
    Label(self.admin_tab, text="Quản lý sinh viên (Admin)", font=("Helvetica", 16, "bold"), bg="green").pack(pady=20, fill="x")

    # Frame for Input Fields
    input_frame = Frame(self.admin_tab, bg="white")
    input_frame.pack(pady=10)

    Label(input_frame, text="Họ", font=("Helvetica", 12), bg="white").grid(row=0, column=0, padx=10, pady=5, sticky=W)
    self.(variable) f_name_entry: Entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.f_name_entry.grid(row=0, column=1, padx=10, pady=5)

    Label(input_frame, text="Tên", font=("Helvetica", 12), bg="white").grid(row=1, column=0, padx=10, pady=5, sticky=W)
    self.l_name_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.l_name_entry.grid(row=1, column=1, padx=10, pady=5)

    Label(input_frame, text="Khóa học", font=("Helvetica", 12), bg="white").grid(row=2, column=0, padx=10, pady=5, sticky=W)
    self.course_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.course_entry.grid(row=2, column=1, padx=10, pady=5)

    Label(input_frame, text="Ngành", font=("Helvetica", 12), bg="white").grid(row=0, column=2, padx=10, pady=5, sticky=W)
    self.subject_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.subject_entry.grid(row=0, column=3, padx=10, pady=5)

    Label(input_frame, text="Năm học", font=("Helvetica", 12), bg="white").grid(row=1, column=2, padx=10, pady=5, sticky=W)
    self.year_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.year_entry.grid(row=1, column=3, padx=10, pady=5)

    Label(input_frame, text="Tuổi", font=("Helvetica", 12), bg="white").grid(row=2, column=2, padx=10, pady=5, sticky=W)
    self.age_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.age_entry.grid(row=2, column=3, padx=10, pady=5)

    Label(input_frame, text="Giới tính", font=("Helvetica", 12), bg="white").grid(row=3, column=0, padx=10, pady=5, sticky=W)
    self.gender_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.gender_entry.grid(row=3, column=1, padx=10, pady=5)

    Label(input_frame, text="Ngày sinh", font=("Helvetica", 12), bg="white").grid(row=3, column=2, padx=10, pady=5, sticky=W)
    self.birth_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.birth_entry.grid(row=3, column=3, padx=10, pady=5)

    Label(input_frame, text="Số liên hệ", font=("Helvetica", 12), bg="white").grid(row=4, column=0, padx=10, pady=5, sticky=W)
    self.contact_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.contact_entry.grid(row=4, column=1, padx=10, pady=5)

    Label(input_frame, text="Email", font=("Helvetica", 12), bg="white").grid(row=4, column=2, padx=10, pady=5, sticky=W)
    self.email_entry = Entry(input_frame, font=("Helvetica", 12), bg="lightgrey")
    self.email_entry.grid(row=4, column=3, padx=10, pady=5)
```

```
# Search Frame
search_frame = Frame(self.admin_tab, bg="white")
search_frame.pack(pady=10)

Label(search_frame, text="Tìm kiếm theo số điện thoại", font=("Helvetica", 12), bg="white").grid(row=0, column=0, padx=10, pady=5, sticky=W)
self.search_entry = Entry(search_frame, font=("Helvetica", 12), bg="lightgrey")
self.search_entry.grid(row=0, column=1, padx=10, pady=5)
Button(search_frame, text="Tìm kiếm", command=self.search_student, font=("Helvetica", 12), fg="white", bg="#4CAF50",
       cursor="hand2", bd=0).grid(row=0, column=2, padx=10, pady=5)

# Buttons Frame
buttons_frame = Frame(self.admin_tab, bg="white")
buttons_frame.pack(pady=10)

# Create update and delete buttons but don't pack or grid them initially
self.update_button = Button(buttons_frame, text="Cập nhật", command=self.update_student, font=("Helvetica", 12), fg="white", bg="#FF9800",
                            cursor="hand2", bd=0)
self.delete_button = Button(buttons_frame, text="Xóa", command=self.delete_student, font=("Helvetica", 12), fg="white", bg="#F44336",
                            cursor="hand2", bd=0)

Button(buttons_frame, text="Thêm sinh viên", command=self.add_student, font=("Helvetica", 12), fg="white", bg="#4CAF50",
       cursor="hand2", bd=0).grid(row=0, column=0, padx=10, pady=5)
Button(buttons_frame, text="Xóa tất cả", command=self.reset_fields, font=("Helvetica", 12), fg="white", bg="#607D8B",
       cursor="hand2", bd=0).grid(row=0, column=3, padx=10, pady=5)
```

```
def create_treeview_tab(self):
    Label(self.treeview_tab, text="Danh sách Sinh Viên", font=("Helvetica", 16, "bold"), bg="white").pack(pady=20)

    self.list_frame = Frame(self.treeview_tab, bg="white", bd=2, relief=RIDGE)
    self.list_frame.pack(fill=BOTH, expand=True, padx=10, pady=10)

    self.student_tree = Treeview(self.list_frame, columns=("f_name", "l_name", "course", "subject", "year", "age", "gender", "birth", "contact", "email"), show="headings")

    self.student_tree.heading("f_name", text="Họ")
    self.student_tree.heading("l_name", text="Tên")
    self.student_tree.heading("course", text="Khóa học")
    self.student_tree.heading("subject", text="Ngành")
    self.student_tree.heading("year", text="Năm học")
    self.student_tree.heading("age", text="Tuổi")
    self.student_tree.heading("gender", text="Giới tính")
    self.student_tree.heading("birth", text="Ngày sinh")
    self.student_tree.heading("contact", text="Số liên hệ")
    self.student_tree.heading("email", text="Email")

    # Set column widths
    self.student_tree.column("f_name", width=40)
    self.student_tree.column("l_name", width=40)
    self.student_tree.column("course", width=60)
    self.student_tree.column("subject", width=60)
    self.student_tree.column("year", width=80)
    self.student_tree.column("age", width=50)
    self.student_tree.column("gender", width=60)
    self.student_tree.column("birth", width=100)
    self.student_tree.column("contact", width=100)
    self.student_tree.column("email", width=160)

    self.student_tree.pack(fill=BOTH, expand=True)
    self.refresh_treeview()
```

```python
def logout(self):
    self.window.destroy()
    root = Tk()
    Login(root)
    root.mainloop()

Tabnine | Edit | Test | Explain | Document | Ask
def refresh_treeview(self):
    # Xóa tất cả các mục trong treeview
    for i in self.student_tree.get_children():
        self.student_tree.delete(i)

    try:
        # Gọi phương thức từ model để lấy danh sách sinh viên
        rows = StudentModel.get_all_students()

        # Thêm các hàng vào treeview
        for row in rows:
            self.student_tree.insert('', 'end', values=row)
    except Exception as e:
        messagebox.showerror("Error", str(e))
```

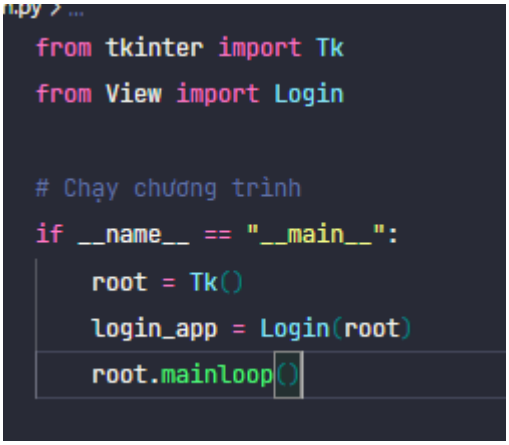
```python
def search_student(self):
    phone = self.search_entry.get()
    if phone == "":
        messagebox.showerror("Error", "Vui lòng nhập số điện thoại để tìm kiếm!")
    else:
        try:
            row = StudentModel.search_student_by_phone(phone)
            if row:
                # Populate fields with data
                self.f_name_entry.delete(0, END)
                self.f_name_entry.insert(0, row[0])
                self.l_name_entry.delete(0, END)
                self.l_name_entry.insert(0, row[1])
                self.course_entry.delete(0, END)
                self.course_entry.insert(0, row[2])
                self.subject_entry.delete(0, END)
                self.subject_entry.insert(0, row[3])

                # Gán giá trị cho ô năm học
                self.year_entry.delete(0, END)
                self.year_entry.insert(0, row[4] if row[4] is not None else "")

                self.age_entry.delete(0, END)
                self.age_entry.insert(0, row[5] if row[5] is not None else "")

                self.(variable) gender_entry: Entry
                self.gender_entry.insert(0, row[6])
                self.birth_entry.delete(0, END)
                self.birth_entry.insert(0, row[7])
                self.contact_entry.delete(0, END)
                self.contact_entry.insert(0, row[8])
                self.email_entry.delete(0, END)
                self.email_entry.insert(0, row[9])

                # Make the update and delete buttons visible
                self.update_button.grid(row=0, column=1, padx=10, pady=5)
                self.delete_button.grid(row=0, column=2, padx=10, pady=5)
            else:
                messagebox.showerror("Error", "Không tìm thấy sinh viên!")
        except Exception as e:
            messagebox.showerror("Error", f"Lỗi khi tìm sinh viên: {str(e)}")
```

```python
def add_student(self):
    if self.f_name_entry.get() == "" or self.l_name_entry.get() == "":
        messagebox.showerror("Error", "Vui lòng điền đầy đủ thông tin!")
    else:
        try:
            student_data = (
                self.f_name_entry.get(),
                self.l_name_entry.get(),
                self.course_entry.get(),
                self.subject_entry.get(),
                self.year_entry.get() if self.year_entry.get() != "" else None,
                self.age_entry.get() if self.age_entry.get() != "" else None,
                self.gender_entry.get(),
                self.birth_entry.get(),
                self.contact_entry.get(),
                self.email_entry.get()
            )

            StudentModel.add_student(student_data)
            messagebox.showinfo("Success", "Thêm sinh viên thành công!")
            self.reset_fields()
            self.refresh_treeview()
        except Exception as e:
            messagebox.showerror("Error", str(e))
```

```python
def update_student(self):
    phone = self.contact_entry.get()
    if phone == "":
        messagebox.showerror("Error", "Vui lòng nhập số điện thoại!")
    else:
        try:
            # Kiểm tra các trường dữ liệu số, thay chuỗi rỗng bằng None
            year = self.year_entry.get() if self.year_entry.get() != "" else None
            age = self.age_entry.get() if self.age_entry.get() != "" else None

            student_data = (
                self.f_name_entry.get(),
                self.l_name_entry.get(),
                self.course_entry.get(),
                self.subject_entry.get(),
                year,    # sử dụng None nếu giá trị rỗng
                age,     # sử dụng None nếu giá trị rỗng
                self.gender_entry.get(),
                self.birth_entry.get(),
                self.email_entry.get()
            )

            StudentModel.update_student(student_data, phone)
            messagebox.showinfo("Success", "Cập nhật thành công!")
            self.reset_fields()
            self.refresh_treeview()
        except Exception as e:
            messagebox.showerror("Error", str(e))

def delete_student(self):
    phone = self.contact_entry.get()
    if phone == "":
        messagebox.showerror("Error", "Vui lòng nhập số điện thoại!")
    else:
        try:
            StudentModel.delete_student(phone)
            messagebox.showinfo("Success", "Xóa sinh viên thành công!")
            self.reset_fields()
            self.refresh_treeview()
        except Exception as e:
            messagebox.showerror("Error", str(e))
```

```
def reset_fields(self):
    self.f_name_entry.delete(0, END)
    self.l_name_entry.delete(0, END)
    self.course_entry.delete(0, END)
    self.subject_entry.delete(0, END)
    self.year_entry.delete(0, END)
    self.age_entry.delete(0, END)
    self.gender_entry.delete(0, END)
    self.birth_entry.delete(0, END)
    self.contact_entry.delete(0, END)
    self.email_entry.delete(0, END)
    self.search_entry.delete(0, END)

    # Hide the update and delete buttons again after reset
    self.update_button.grid_remove()
    self.delete_button.grid_remove()
```

Main.py



IV/ Link github

https://github.com/TranMinhPhuc04/Python-Programming.git