

TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ
BỘ MÔN CÔNG NGHỆ THÔNG TIN



BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC

LẬP TRÌNH PYTHON

Sinh viên : Nguyễn Trung Hiếu

Lớp : K58KTP

Giáo viên giảng dạy : Nguyễn Văn Huy

Link Github: https://github.com/hieu181104/Laptrinh_Python

Link Video: <https://youtu.be/dej2-f2KMis>

Thái Nguyên – 2025

PHIẾU GIAO ĐỀ TÀI
BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

Sinh viên: Nguyễn Trung Hiếu

MSSV: K225480106019

Chuyên ngành : Kỹ thuật phần mềm

Lớp: K58KTP

Hệ đào tạo : Đại học chính quy

Giáo viên hướng dẫn: Nguyễn Văn Huy

Ngày giao đề: 20/5/2025

Ngày hoàn thành: 9/6/2025

1. Tên đề tài:

Ứng dụng quản lý danh bạ GUI

2. Yêu cầu của đề tài:

Đầu bài: Xây ứng dụng quản lý danh bạ đơn giản: thêm, sửa, xóa liên hệ với name, phone, email, lưu vào file JSON.

Tính năng yêu cầu:

- Đọc/ghi JSON với module json.
- Bắt lỗi format JSON, lưu khi thêm/sửa/xóa.
- GUI: Entry, Buttons, Treeview.
- Tìm kiếm theo tên.

GIÁO VIÊN HƯỚNG DẪN
(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày 10 tháng 6 năm 2025

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN CHẤM

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày 10 tháng 6 năm 2025

GIÁO VIÊN CHẤM

(Ký và ghi rõ họ tên)

MỤC LỤC

MỤC LỤC.....	4
LỜI CAM KẾT	5
LỜI NÓI ĐẦU	6
CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI.....	7
1.1. NỘI DUNG ĐỀ TÀI	7
1.2. PHÂN TÍCH ĐẦU BÀI.....	7
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	9
2.1. PYTHON.....	9
2.2. THƯ VIỆN TKINTER.....	9
2.3. TÙY CHỈNH GIAO DIỆN (ttk.Style)	9
2.4. BIẾN VÀ RÀNG BUỘC DỮ LIỆU	9
2.5. TREEVIEW	10
2.6. LIST VÀ DICTIONARY.....	10
2.7. FILE JSON.....	11
2.8. VÒNG LẶP VÀ CÂU LỆNH Rẽ NHÁNH.....	12
2.9. ROOT, LABEL VÀ BUTTON	13
CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH	15
3.1. SƠ ĐỒ KHỐI HỆ THỐNG.....	15
3.2. SƠ ĐỒ KHỐI CÁC THUẬT TOÁN CHÍNH	16
3.3. CẤU TRÚC DỮ LIỆU.....	19
3.4. CHƯƠNG TRÌNH	19
CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN	22
4.1. THỰC NGHIỆM.....	22
4.2. KẾT LUẬN	26
TÀI LIỆU THAM KHẢO.....	27

LỜI CAM KẾT

Em xin cam kết rằng tất cả nội dung của báo cáo này đều được thực hiện dựa trên quá trình tìm hiểu, học tập và thực hành chăm chỉ. Nội dung trong báo cáo được trình bày một cách cẩn thận và trung thực.

Em xin khẳng định rằng nếu có bất kỳ vi phạm nào sai sót hoặc liên quan đến bản quyền trong nội dung báo cáo, em xin hoàn toàn chịu trách nhiệm.

Sinh viên thực hiện

(Ký và ghi rõ họ tên)

Nguyễn Trung Hiếu

LỜI NÓI ĐẦU

Trong quá trình học tập và rèn luyện kỹ năng lập trình Python, việc áp dụng kiến thức vào các dự án thực tế luôn là một bước quan trọng giúp sinh viên nói chung và bản thân em nói riêng nắm vững và phát triển tư duy lập trình. Đề tài “Ứng dụng quản lý danh bạ GUI” là một bài tập cuối kỳ nhằm mục tiêu củng cố kiến thức đã học, từ xử lý dữ liệu JSON đến việc thiết kế giao diện người dùng với thư viện Tkinter.

Thông qua việc xây dựng một ứng dụng quản lý danh bạ đơn giản, đề tài giúp em: Hiểu rõ hơn về cách tổ chức và lưu trữ dữ liệu bằng file JSON; rèn luyện kỹ năng xây dựng giao diện đồ họa (GUI) trong Python; phát triển kỹ năng xử lý dữ liệu, bắt lỗi định dạng và triển khai các thao tác cơ bản như thêm, sửa, xóa, tìm kiếm. Bài tập cũng tạo điều kiện cho em tiếp cận quy trình phát triển phần mềm gồm nhiều bước: thiết kế giao diện, xây dựng chức năng xử lý, kiểm thử và đảm bảo tính ổn định của ứng dụng.

Với tinh thần học hỏi và mong muốn hoàn thiện kỹ năng lập trình, em xin trình bày bài tập cuối kỳ của mình dưới đây. Rất mong nhận được sự góp ý và nhận xét từ thầy Nguyễn Văn Huy để em có thể cải thiện kỹ năng và hoàn thiện hơn trong các dự án lập trình sau này.

Em xin chân thành cảm ơn!

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1. NỘI DUNG ĐỀ TÀI

1.1.1. Đầu bài:

Xây ứng dụng quản lý danh bạ đơn giản: thêm, sửa, xoá liên hệ với name, phone, email, lưu vào file JSON.

1.1.2. Đầu vào – đầu ra:

- Đầu vào: Form nhập name, phone, email.
- Đầu ra: Table hiển thị danh sách, lưu file contacts.json.

1.1.3. Tính năng yêu cầu:

- Đọc/ghi JSON với module json.
- Bắt lỗi format JSON, lưu khi thêm/sửa/xoá.
- GUI: Entry, Buttons, Treeview.
- Tìm kiếm theo tên.

1.1.4. Kiểm tra & kết quả mẫu:

- Thêm “Nguyen”, “0123” → xuất hiện trong Table.
- Sửa số → file JSON cập nhật.

1.1.5. Các bước triển khai:

- Module contacts.py chứa class quản lý list, load/save.
- GUI class: form nhập và Treeview.
- Map nút Add/Edit/Delete gọi methods.
- Tự động load file khi khởi động.

1.2. PHÂN TÍCH ĐẦU BÀI

Bài tập cuối kỳ này yêu cầu xây dựng một ứng dụng quản lý danh bạ đơn giản, với các tính năng cơ bản: thêm, sửa, xoá liên hệ (gồm thông tin name, phone, email) và lưu trữ vào file JSON. Ứng dụng sử dụng giao diện đồ hoạ (GUI) để người dùng dễ dàng

nhập dữ liệu, hiển thị danh sách liên hệ, đồng thời cung cấp các chức năng tìm kiếm và chỉnh sửa thuận tiện.

Cụ thể, yêu cầu bài tập như sau:

- Đầu vào là form nhập các trường: name, phone, email.
- Đầu ra là bảng hiển thị danh sách liên hệ, đồng thời dữ liệu được lưu trữ trong file `contacts.json`.

Chương trình cần thực hiện các chức năng chính:

- Đọc/ghi dữ liệu từ file JSON bằng module `json`.
- Phát hiện và xử lý lỗi định dạng JSON khi thêm/sửa/xoá dữ liệu.
- Thiết kế giao diện người dùng với các thành phần: Entry, Button và Treeview.
- Cho phép tìm kiếm liên hệ theo tên.

Trong quá trình thực hiện, bài tập đặt ra một số thử thách như:

- Đảm bảo dữ liệu được lưu trữ chính xác và có thể đọc lại khi chương trình khởi động.
- Quản lý việc hiển thị dữ liệu trên Treeview một cách hợp lý, tránh lỗi hiển thị hoặc sai dữ liệu.
- Việc bắt lỗi JSON cũng đòi hỏi hiểu biết về xử lý ngoại lệ và cấu trúc dữ liệu.

Để hoàn thành bài tập này, em cần vận dụng nhiều kiến thức đã học trong môn học Lập trình Python, bao gồm:

- Kỹ năng thao tác với file JSON (đọc, ghi, kiểm tra định dạng).
- Tổ chức và quản lý dữ liệu dưới dạng danh sách (list) và từ điển (dict).
- Thiết kế giao diện người dùng bằng Tkinter, phối hợp các thành phần Entry, Button và Treeview.
- Xử lý sự kiện (event handling) trong ứng dụng GUI.

Qua đó, bài tập giúp em củng cố khả năng lập trình Python cơ bản, đồng thời tiếp cận kỹ năng thiết kế ứng dụng có giao diện người dùng – một kỹ năng rất quan trọng trong thực tế phát triển phần mềm.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. PYTHON

Python là một ngôn ngữ lập trình bậc cao, có cú pháp đơn giản, dễ học và rất phù hợp với người mới bắt đầu. Python hỗ trợ lập trình hướng đối tượng, xử lý ngoại lệ, và đặc biệt có sẵn nhiều thư viện để xây dựng các ứng dụng thực tế, bao gồm cả ứng dụng có giao diện người dùng (GUI).

2.2. THƯ VIỆN TKINTER

Tkinter là thư viện chuẩn của Python để xây dựng GUI. Trong chương trình, Tkinter được sử dụng để:

- Tạo cửa sổ chính, đặt tiêu đề, kích thước, màu nền.
- Tạo các thành phần giao diện (widgets) như:
 - + Entry: các ô nhập liệu cho name, phone, email.
 - + Label: tiêu đề trường nhập.
 - + Button: các nút bấm thực hiện chức năng.
 - + Frame: khung để bố trí các thành phần.
- Tổ chức bố cục (sử dụng pack, grid), thiết lập màu nền, padding và các thuộc tính hiển thị.
- Tích hợp Treeview để hiển thị danh bạ dưới dạng bảng (với các cột: Name, Phone, Email).

2.3. TÙY CHỈNH GIAO DIỆN (ttk.Style)

Module ttk.Style được sử dụng để thay đổi kiểu hiển thị của bảng Treeview (chữ đậm, màu sắc tiêu đề, màu hàng được chọn). Giúp giao diện trực quan, dễ nhìn và thân thiện hơn với người dùng.

2.4. BIẾN VÀ RÀNG BUỘC DỮ LIỆU

Các biến tk.StringVar() được dùng để lưu trữ giá trị nhập vào (name, phone, email, search).

StringVar.trace() được sử dụng để ràng buộc sự kiện khi người dùng nhập tìm kiếm, từ đó tự động lọc dữ liệu và hiển thị kết quả trong bảng.

2.5. TREEVIEW

Treeview là một thành phần giao diện người dùng (UI) được sử dụng để hiển thị các mục dữ liệu có cấu trúc phân cấp (giống như dạng cây). Mỗi mục dữ liệu (gọi là node) có thể chứa các sub-node (node con), và có thể được mở rộng (expand) hoặc thu gọn (collapse) để người dùng dễ dàng duyệt qua.

Trong bài, Widget `tk.Ttreeview` sẽ được thiết lập với các cột ("Name", "Phone", "Email"), giúp hiển thị dữ liệu danh bạ rõ ràng và trực quan.

Hỗ trợ thao tác chọn dòng để phục vụ chức năng sửa và xóa.

2.6. LIST VÀ DICTIONARY

List trong Python là một danh sách có thứ tự (ordered), có thể chứa nhiều phần tử, kể cả các kiểu dữ liệu khác nhau. Danh sách có thể thay đổi được (mutable), và các phần tử có thể trùng lặp.

Cú pháp: `my_list = [item1, item2, item3, ...]`

- Các phần tử trong list được phân tách bởi dấu phẩy.
- Có thể truy cập phần tử bằng chỉ số: `my_list[index]`.

Trong bài, `self.contacts` là một list được tạo và chứa nhiều liên hệ. Khi thêm mới, sửa, xóa hoặc tìm kiếm, chương trình đều thao tác trực tiếp với list `self.contacts` này. Ngoài ra, dữ liệu để hiển thị lên giao diện là một list.

Dict (dictionary) là một cấu trúc dữ liệu lưu trữ các cặp khóa - giá trị. Các khóa phải duy nhất, còn giá trị thì có thể trùng lặp.

Cú pháp:

```
my_dict = {  
    "key1": value1,  
    "key2": value2,  
    ...  
}
```

Trong bài, mỗi liên hệ là một dict, dễ dàng quản lý thông tin (name, phone, email). Các thao tác thêm, sửa, xóa đều thực hiện bằng cách cập nhật dict trong list.

2.7. FILE JSON

2.7.1. Khái niệm

JSON là viết tắt của JavaScript Object Notation. Nó là một định dạng dữ liệu đơn giản, dễ đọc, thường dùng để lưu trữ và trao đổi dữ liệu giữa các ứng dụng.

JSON có cấu trúc cặp khóa – giá trị, rất giống với kiểu dict trong Python.

2.7.2. Cú pháp

Cú pháp JSON khá giống Python dict nhưng:

- Khóa & giá trị phải được đặt trong dấu ngoặc kép kép ("").
- Dấu phân cách là dấu phẩy.

Ví dụ:

```
[  
  {  
    "name": "Nguyen Van A",  
    "phone": "0123456789",  
    "email": "nguyenvana@example.com"  
  },  
  {  
    "name": "Tran Thi B",  
    "phone": "0987654321",  
    "email": "tranthib@example.com"  
  }  
]
```

2.7.3. Ứng dụng của JSON và cách dùng

Ứng dụng:

- Lưu trữ dữ liệu cấu trúc (vd: cấu hình, dữ liệu người dùng).
- Trao đổi dữ liệu giữa front-end và back-end của ứng dụng web.

- Lưu trữ dữ liệu tạm thời trong file (như danh bạ trong bài của bạn).
- Các API (ứng dụng web, dịch vụ đám mây) thường trả về dữ liệu ở định dạng JSON.

Cách dùng trong Python:

Sử dụng thư viện chuẩn: json

Các hàm chính:

- `json.load(f)`: đọc dữ liệu JSON từ file.
- `json.dump(data, f, indent=4)`: ghi dữ liệu (list, dict) thành JSON trong file.
- `json.loads(string)`: chuyển chuỗi JSON thành đối tượng Python.
- `json.dumps(data)`: chuyển đối tượng Python thành chuỗi JSON.

2.8. VÒNG LẶP VÀ CÂU LỆNH Rẽ NHÁNH

2.8.1. Vòng lặp for

Trong Python, câu lệnh lặp (loop) được dùng để thực hiện lặp đi lặp lại một khối lệnh, với số lần lặp xác định hoặc không xác định. Loại lặp chủ yếu và sẽ dùng trong bài là vòng lặp for (for loop).

Cú pháp:

for biến in tập_hợp:

Khối lệnh lặp

Trong đó:

- *biến*: tên biến dùng để nhận giá trị từ tập hợp.
- *tập_hợp*: có thể là list, dict, tuple hoặc str.
- Trong mỗi vòng lặp, biến lần lượt nhận từng giá trị của tập_hợp và thực hiện khối lệnh bên trong.

2.8.2. Câu lệnh rẽ nhánh If – else

Câu lệnh rẽ nhánh giúp chương trình tự động quyết định chạy khối lệnh nào dựa vào điều kiện đúng hay sai. Câu lệnh rẽ nhánh cho phép “chia nhánh” luồng chương trình tùy theo kết quả của điều kiện kiểm tra.

Cú pháp:

if điều_kiện:

Khối lệnh nếu điều kiện đúng

elif điều_kiện_khác:

Khối lệnh nếu điều kiện_khác đúng

else:

Khối lệnh nếu không có điều kiện nào đúng

2.9. ROOT, LABEL VÀ BUTTON

2.9.1. Root

Trong tkinter, root là cửa sổ chính của ứng dụng. Tất cả các thành phần giao diện (Label, Button, Entry, Treeview, ...) sẽ được chèn vào trong root hoặc các widget con của nó.

Cú pháp khởi tạo:

```
import tkinter as tk
```

```
root = tk.Tk()
```

Lệnh này tạo một cửa sổ gốc có thể chứa các thành phần giao diện khác.

Một số thao tác phổ biến như:

- `root.title("Tên cửa sổ")` : đặt tiêu đề cửa sổ
- `root.geometry("600x400")` : đặt kích thước cửa sổ
- `root.mainloop()` : chạy giao diện

2.9.2. Label

Label là thành phần giao diện dùng để hiển thị văn bản (nhãn), thường dùng để mô tả hoặc gợi ý cho các ô nhập liệu (Entry).

Cú pháp:

```
label = tk.Label(parent, text="Nội dung", bg="màu nền", fg="màu chữ", font="phông chữ")
```

```
label.pack() # hoặc .grid(), .place()
```

Trong đó:

- parent: widget cha (ví dụ: root hoặc một Frame).

Các tham số thông dụng:

- text: văn bản hiển thị.
- bg: màu nền.
- fg: màu chữ.
- font: kiểu chữ và kích thước.

2.9.3. Button

Button là thành phần giao diện dùng để tạo nút bấm. Mỗi nút thường được gán hành động khi bấm (qua tham số command).

Cú pháp:

```
btn = tk.Button(parent, text="Nội dung nút", bg="màu nền", fg="màu chữ",  
width=chiều_rộng, command=hàm_xử_lý)
```

```
btn.pack() # hoặc .grid(), .place()
```

Trong đó: command là tên hàm xử lý khi nút được bấm

Tóm lại, Root, Label và Button là các thành phần này cơ bản và quan trọng nhất trong giao diện tkinter, giúp xây dựng ứng dụng quản lý danh bạ dễ dùng và trực quan.

CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

3.1. SƠ ĐỒ KHỐI HỆ THỐNG

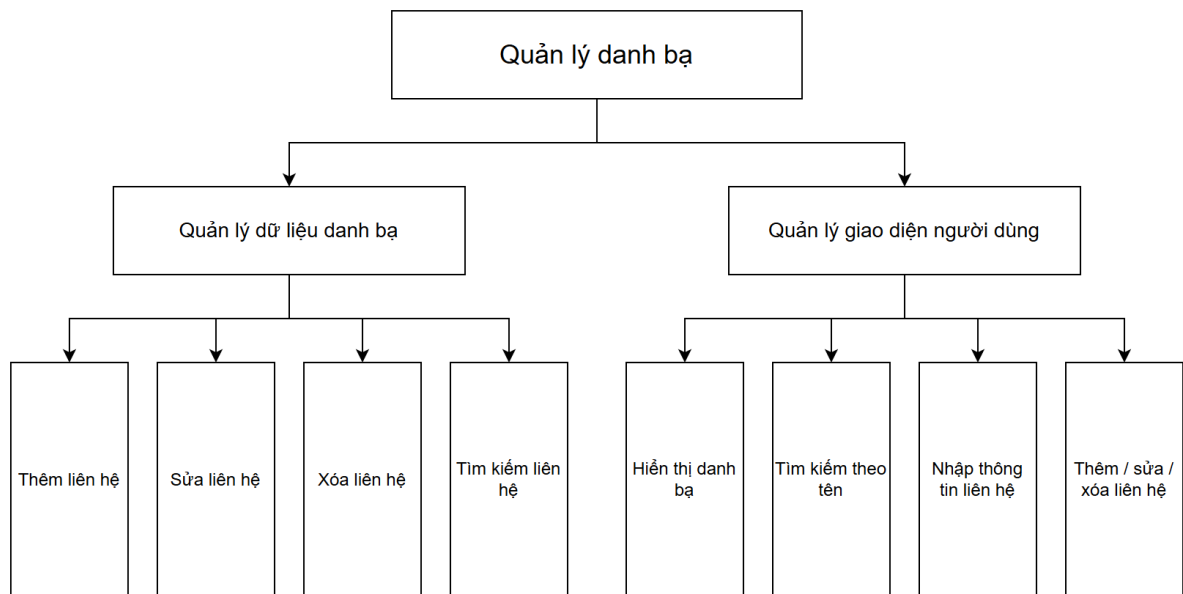
3.1.1. Các module chính trong chương trình

Trong ứng dụng quản lý danh bạ GUI của em, chương trình được tổ chức thành 3 module chính, bao gồm:

- Giao diện người dùng (main.py)
 - + Nhiệm vụ:
 - Hiển thị cửa sổ chính, các nhãn, ô nhập, nút bấm.
 - Cho phép người dùng tương tác (nhập, tìm kiếm, chỉnh sửa).
 - + Thành phần chính:
 - tkinter: Xây dựng giao diện.
 - ContactApp: Lớp quản lý giao diện.
- Xử lý dữ liệu (contacts.py)
 - + Nhiệm vụ:
 - Quản lý danh sách danh bạ: thêm, xóa, chỉnh sửa, tìm kiếm.
 - Kiểm tra dữ liệu hợp lệ
 - Đọc và ghi file contacts.json (dùng json module).
- File dữ liệu (contacts.json)
 - + Nhiệm vụ:
 - Lưu trữ thông tin danh bạ ở dạng JSON.
 - Giúp dữ liệu được bền vững, không bị mất khi đóng ứng dụng.

3.1.2. Biểu đồ phân cấp chức năng

Dựa theo yêu cầu của bài tập và các phân tích, ta có biểu đồ phân cấp chức năng của hệ thống như sau:

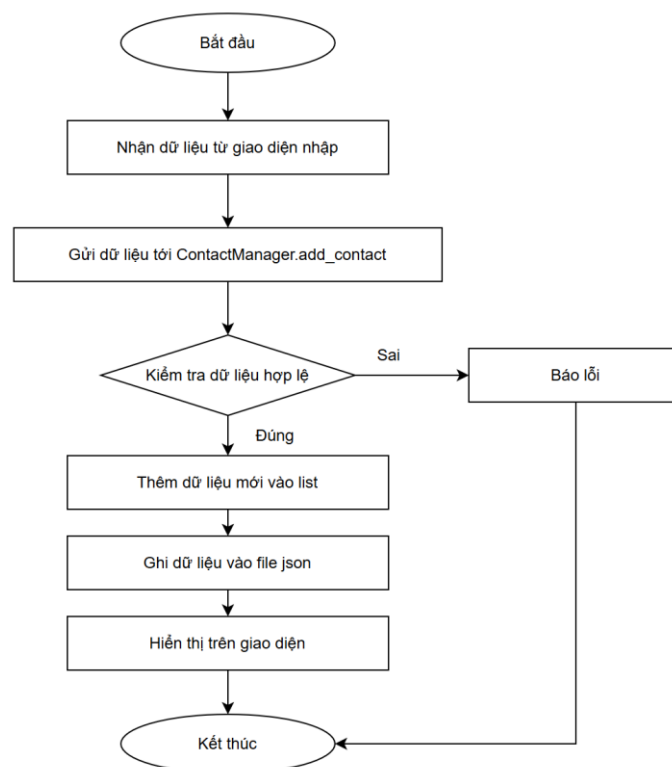


Hình 3.1. Biểu đồ phân cấp chức năng

3.2. SƠ ĐỒ KHỐI CÁC THUẬT TOÁN CHÍNH

3.2.1. Thuật toán Thêm liên hệ (Add)

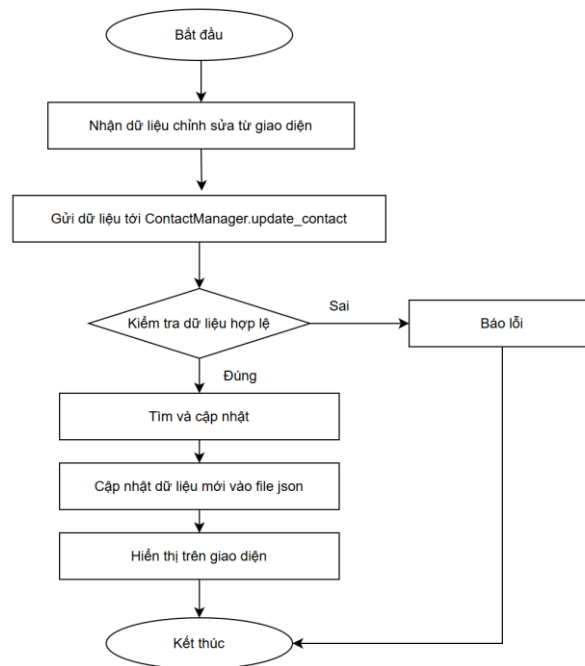
Thuật toán này dùng để thêm một liên hệ vào danh bạ:



Hình 3.2. Sơ đồ khối thuật toán Thêm liên hệ

3.2.2. Thuật toán Sửa liên hệ (Edit)

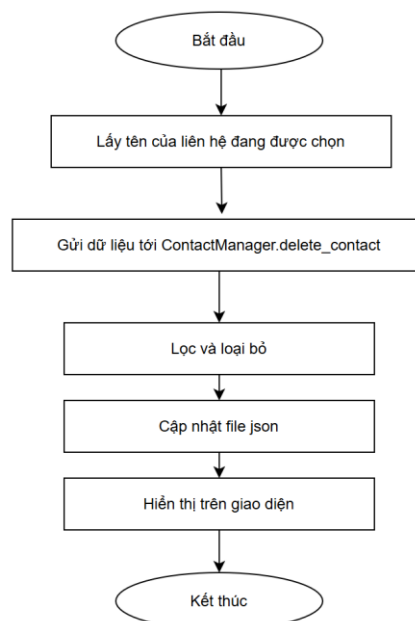
Thuật toán này cho phép sửa thông tin của một liên hệ trong danh bạ:



Hình 3.3. Sơ đồ khối thuật toán Sửa liên hệ

3.2.3. Thuật toán Xóa liên hệ (Delete)

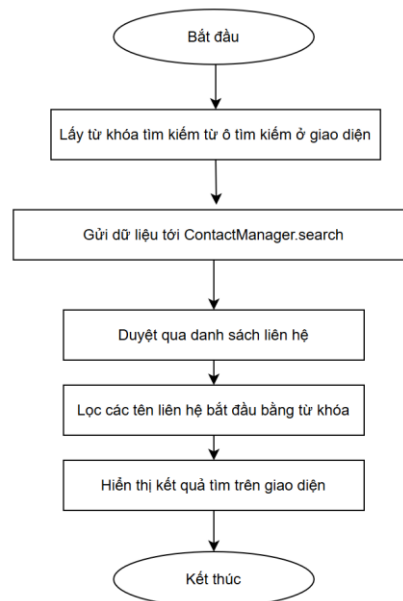
Thuật toán này cho phép xóa thông tin liên hệ:



Hình 3.4. Sơ đồ khối thuật toán Xóa liên hệ

3.2.4. Thuật toán Tìm kiếm (Search)

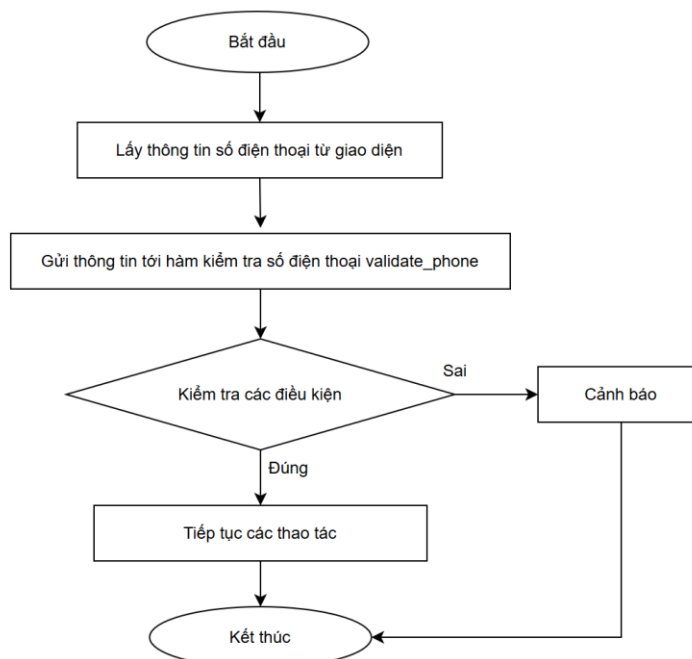
Thuật toán này cho phép tìm kiếm liên hệ trong danh bạ theo tên:



Hình 3.5. Sơ đồ khối thuật toán Tìm kiếm

3.2.5. Thuật toán Kiểm tra số điện thoại

Thuật toán này dùng để kiểm tra xem số điện thoại có đúng quy cách không:



Hình 3.5. Sơ đồ khối thuật toán Kiểm tra số điện thoại

3.3. CẤU TRÚC DỮ LIỆU

Trong chương trình của em, dữ liệu được lưu trong tệp `contact.json`, dạng file là json với kiểu dữ liệu là danh sách (list) chứa nhiều từ điển (dict), mỗi một dict đại diện cho một liên hệ trong danh bạ.

Mỗi một liên hệ được biểu diễn với ba trường thông tin chính:

Trường dữ liệu	Ý nghĩa	Kiểu
name	Tên liên hệ	string
phone	Số điện thoại của người liên hệ	string
email	Địa chỉ email	string

Các trường dữ liệu `name`, `phone`, `email` luôn đi cùng nhau và tạo thành thông tin hoàn chỉnh của một liên hệ.

Danh sách các liên hệ (contacts) chứa toàn bộ các liên hệ, dùng cho việc: thêm mới, chỉnh sửa, xóa, tìm kiếm và hiển thị.

3.4. CHƯƠNG TRÌNH

Một số hàm trong chương trình chính:

```
def load_contacts(self, filtered=None):  
    for i in self.tree.get_children():  
        self.tree.delete(i)  
  
    data = filtered if filtered else self.manager.contacts  
  
    for contact in data:  
        self.tree.insert("", tk.END, values=(contact["name"], contact["phone"],  
contact["email"]))  
  
def add_contact(self):  
    name, phone, email = self.get_input()  
  
    try:  
        self.manager.add_contact(name, phone, email)
```

```
        self.load_contacts()

    except ValueError as e:

        messagebox.showerror("Error", str(e))


def delete_contact(self):

    selected = self.tree.selection()

    if selected:

        item = self.tree.item(selected[0])

        name = item["values"][0]

        self.manager.delete_contact(name)

        self.load_contacts()


def edit_contact(self):

    selected = self.tree.selection()

    if selected:

        old_item = self.tree.item(selected[0])

        old_name = old_item["values"][0]

        new_name, phone, email = self.get_input()

        try:

            self.manager.update_contact(old_name, new_name, phone, email)

            self.load_contacts()

        except ValueError as e:

            messagebox.showerror("Error", str(e))


def search_contacts(self):
```

```
keyword = self.search_var.get()

if keyword:

    results = self.manager.search(keyword)

    self.load_contacts(results)

else:

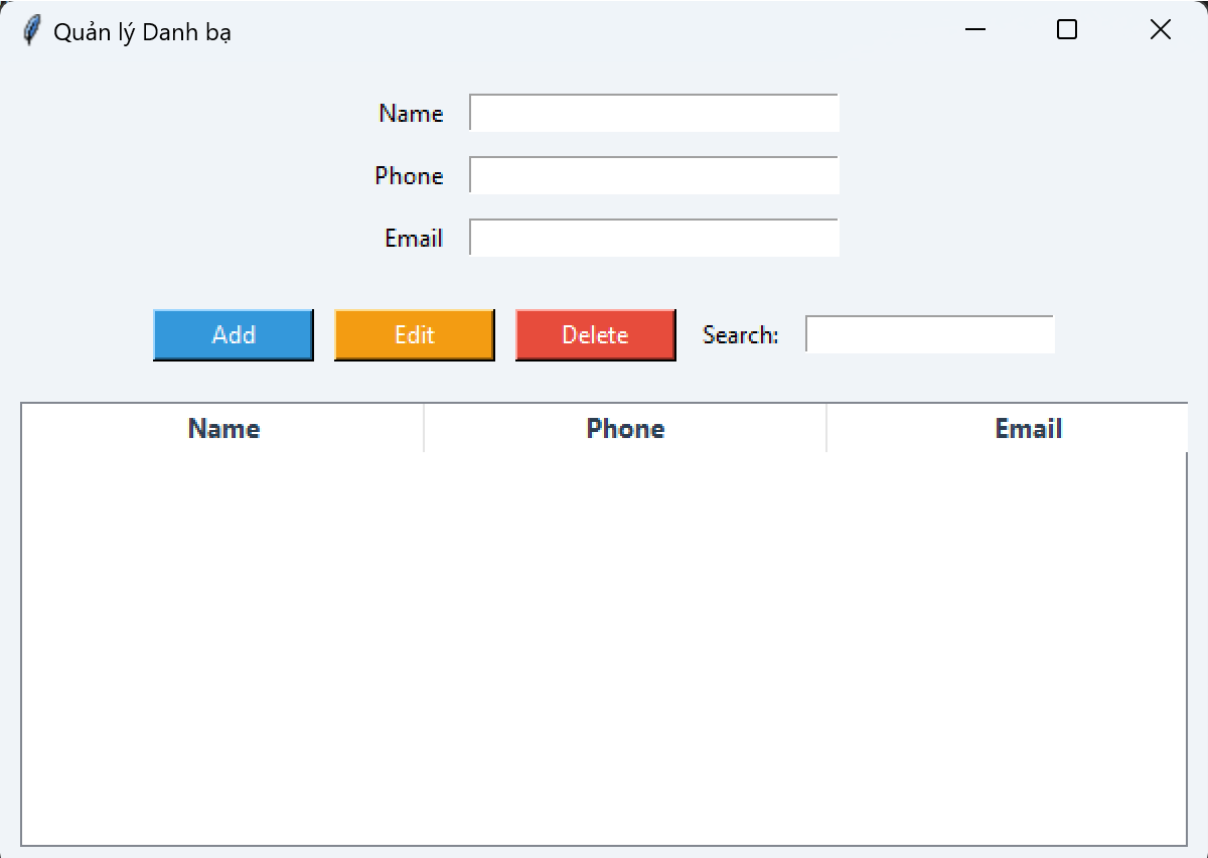
    self.load_contacts()
```

CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN

4.1. THỰC NGHIỆM

Để đảm bảo sản phẩm hoạt động chính xác và đáp ứng các yêu cầu đề ra, em đã tiến hành chạy thử nghiệm chương trình. Các bài test đã được thực hiện để kiểm tra toàn bộ tính năng của sản phẩm, bao gồm: thêm liên hệ, chỉnh sửa, xóa, tìm kiếm và hiển thị danh bạ.

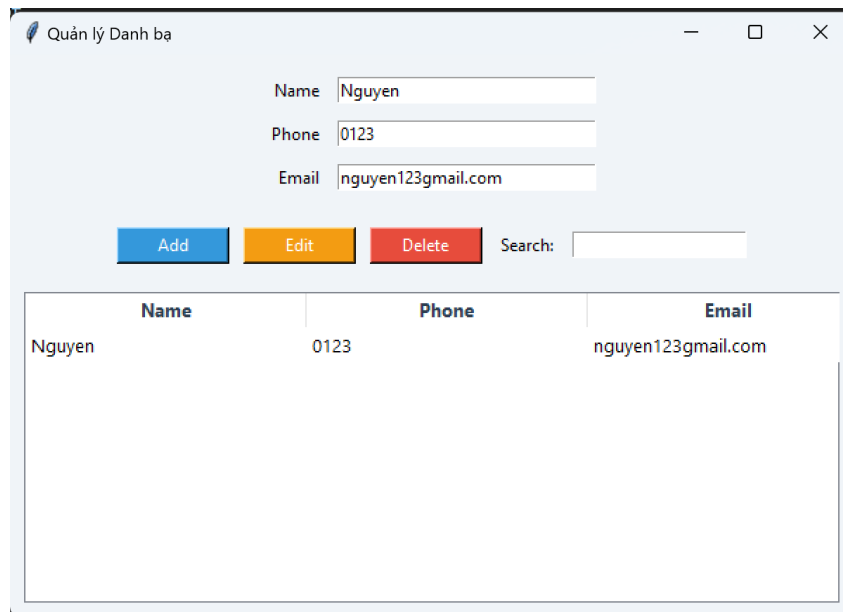
Trong quá trình thực nghiệm, nhóm đã ghi nhận kết quả của các bài test, đồng thời chụp lại các màn hình quan trọng minh họa cho từng chức năng:



Name	Phone	Email
------	-------	-------

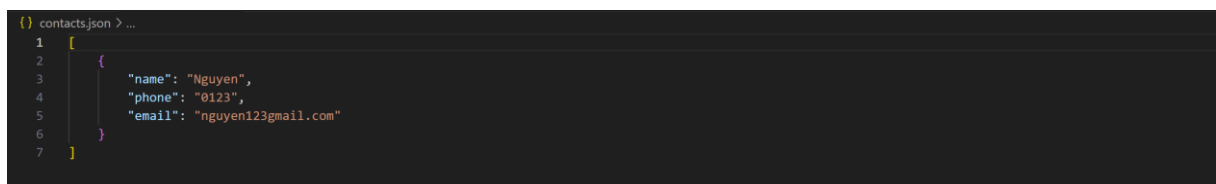
Hình 4.1

Sau khi chạy code, ứng dụng hiển thị đầy đủ các yêu cầu của đề bài bao gồm giao diện đồ họa, các nút thêm, sửa, xóa, ô tìm kiếm và danh sách hiển thị danh bạ.

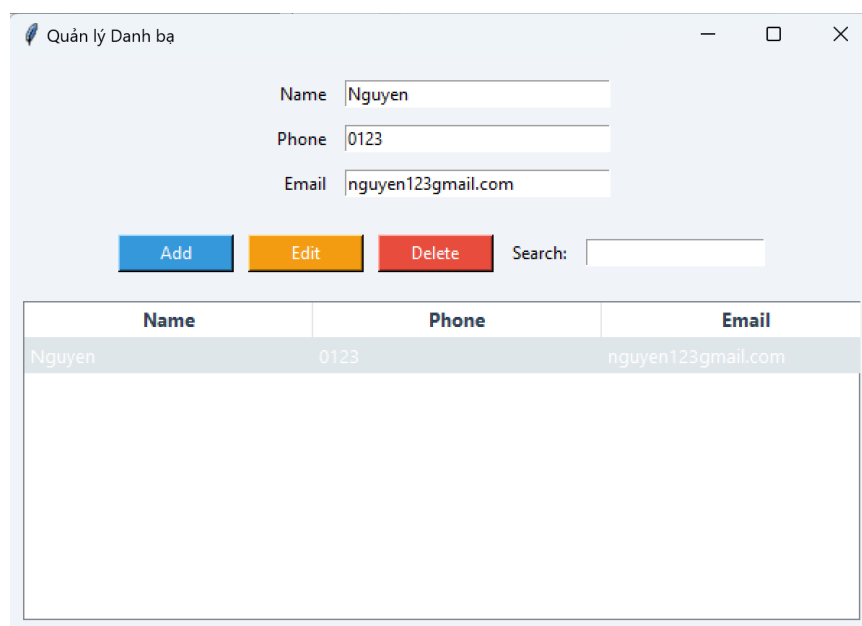


Hình 4.2

Sau khi nhập thông tin liên hệ, ta thấy dữ liệu của người liên hệ vừa nhập đã hiển thị trên giao diện. Ngoài ra file json cũng đã cập nhật dữ liệu vừa lưu (Hình 4.3)

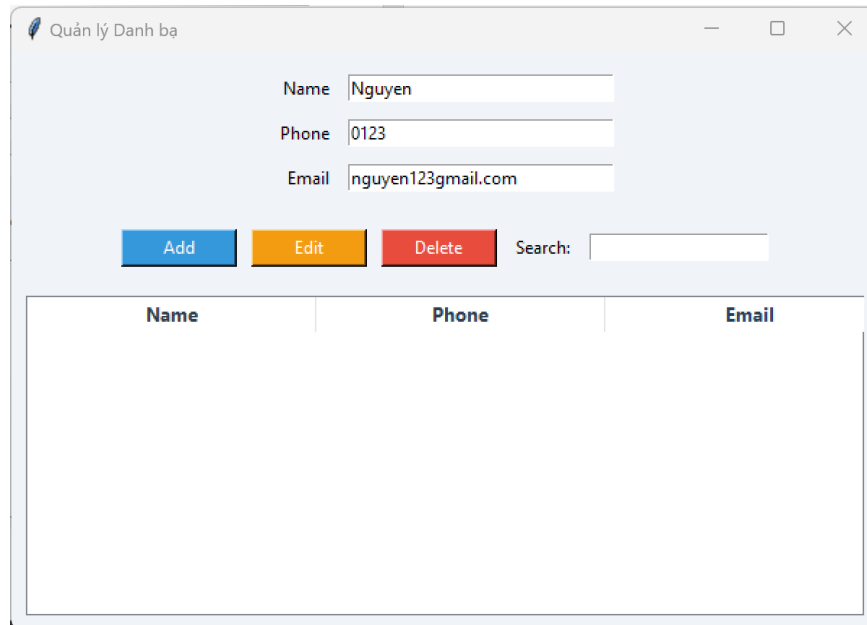


Hình 4.3



Hình 4.4

Bấm chọn một người liên hệ để tiến hành thao tác Edit hoặc Delete, ở đây em sẽ ví dụ trường hợp xóa một liên hệ.

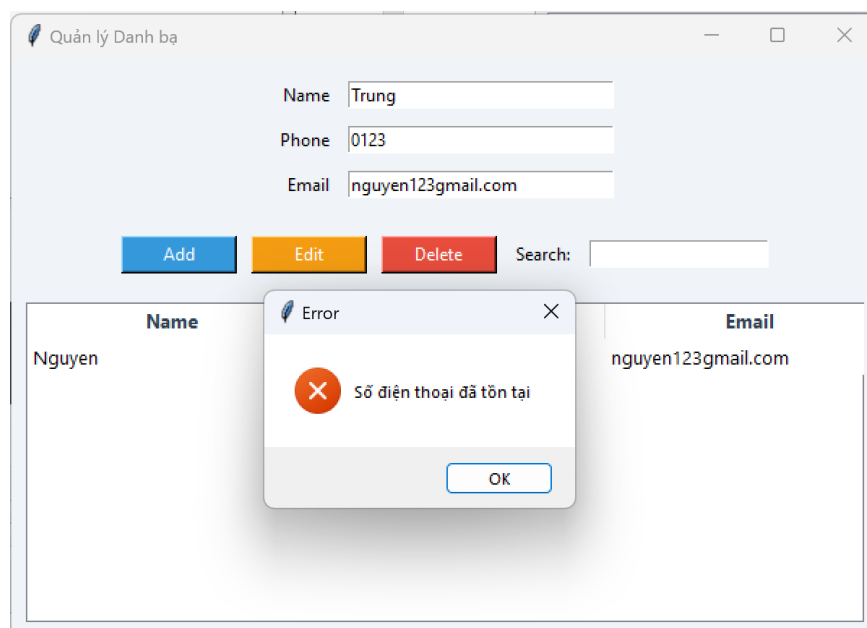


Hình 4.5

Sau khi chọn Delete, dữ liệu trên giao diện đã bị xóa đi và json cũng được cập nhật ngay

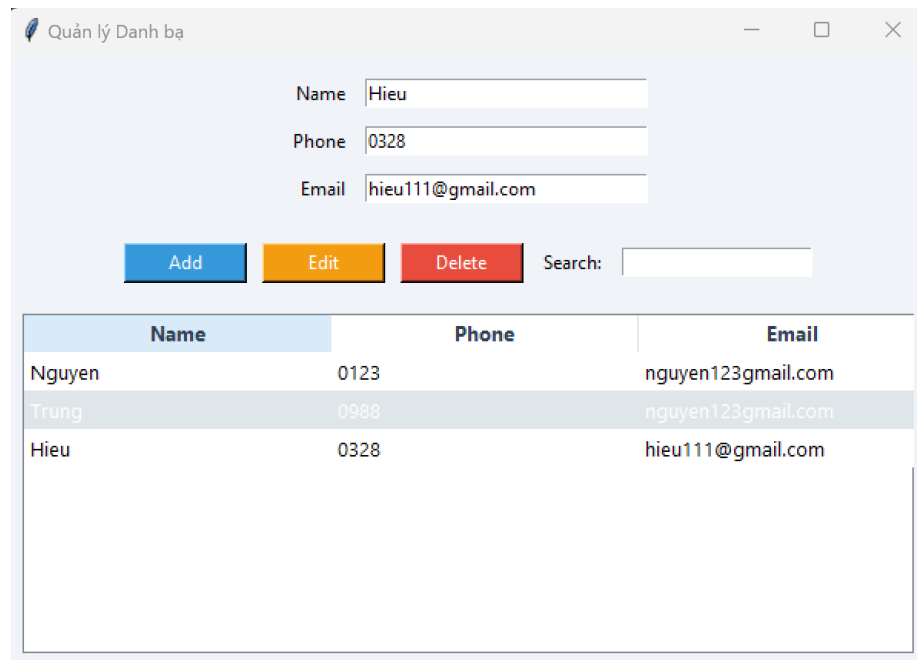


Hình 4.6



Hình 4.7

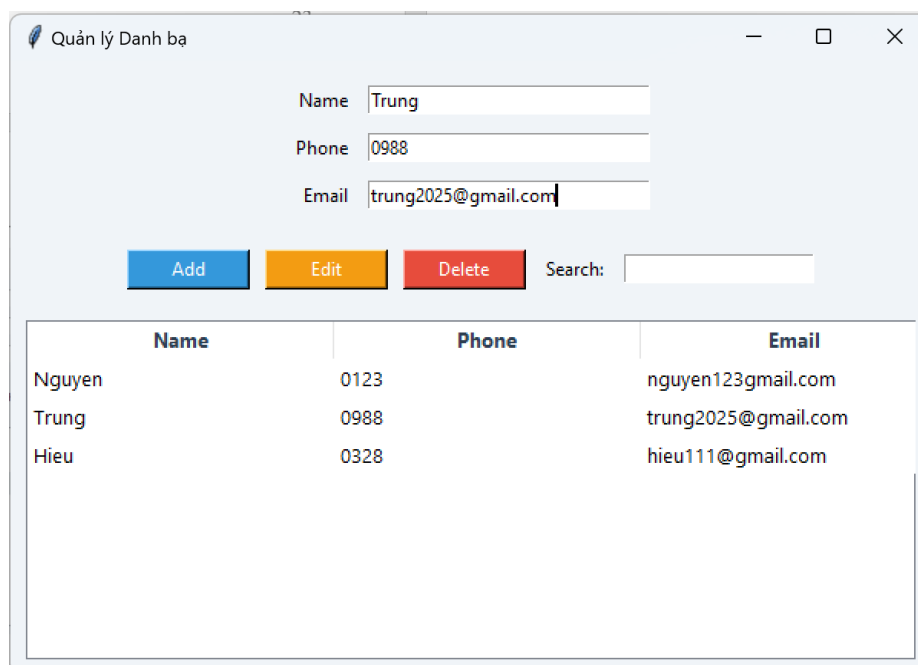
Ngoài ra, nếu ta thêm một liên hệ bị trùng số điện thoại với liên hệ trước đó, hệ thống sẽ cảnh báo và không cho thực hiện tiếp (Hình 4.7).



Name	Phone	Email
Nguyen	0123	nguyen123gmail.com
Trung	0988	nguyen123gmail.com
Hieu	0328	hieu111@gmail.com

Hình 4.8

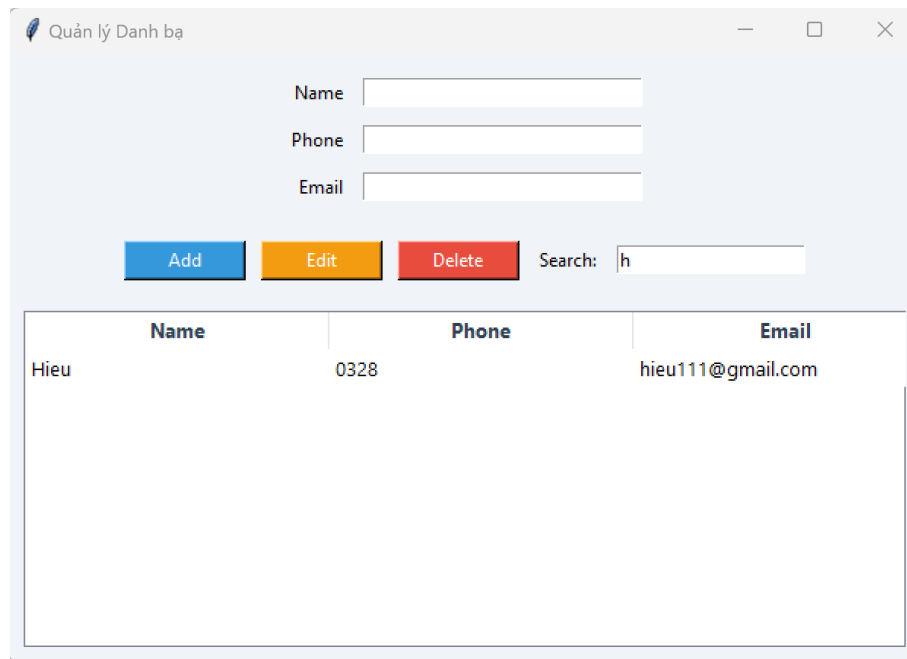
Ở đây có một dòng email em đã nhập sai, em muốn sửa lại thì sẽ chọn vào dòng đó và sửa lại thông tin trên phần nhập, sau đó bấm Edit, dữ liệu sẽ được thay đổi.



Name	Phone	Email
Nguyen	0123	nguyen123gmail.com
Trung	0988	trung2025@gmail.com
Hieu	0328	hieu111@gmail.com

Hình 4.9

Hình 4.9 là kết quả hiển thị sau khi Edit.



Hình 4.10

Để tìm kiếm theo tên, ta click vào hộp Search rồi nhập kí tự cần tìm, ứng dụng sẽ lọc ra các liên hệ bắt đầu bằng kí tự đó (Hình 4.10).

Đánh giá: Qua một số thử nghiệm trên ứng dụng, có thể thấy ứng dụng đã đáp ứng cơ bản các thao tác của người dùng. Tốc độ xử lý nhanh và ổn định, file JSON cũng được cập nhật kịp thời và nhanh chóng, đảm bảo dữ liệu được lưu trữ và duy trì khi tắt ứng dụng.

4.2. KẾT LUẬN

Chương trình quản lý danh bạ đã đáp ứng được các yêu cầu cơ bản: thêm, sửa, xóa, tìm kiếm và hiển thị danh bạ. Giao diện đồ họa được xây dựng thân thiện với người dùng, các thao tác xử lý nhanh, dữ liệu được lưu trữ ổn định dưới dạng file JSON, đảm bảo an toàn và duy trì sau khi thoát ứng dụng.

Trong quá trình phát triển chương trình, bản thân em đã học được cách sử dụng thư viện tkinter để xây dựng giao diện người dùng; quản lý dữ liệu bằng cách đọc, ghi file JSON; thiết kế và triển khai các thuật toán xử lý dữ liệu danh bạ; nắm vững cách kiểm tra, debug, và tối ưu chương trình để chạy ổn định.

Mặc dù ứng dụng đã hoạt động tốt, nhưng vẫn còn nhiều điểm có thể cải thiện như bổ sung tính năng xuất danh bạ ra file excel; cải thiện giao diện người dùng bắt mắt, hiện đại hơn; hỗ trợ tìm kiếm nâng cao hơn hoặc tìm theo nhiều tiêu chí; bổ sung tính năng phân quyền, bảo mật dữ liệu người dùng.

TÀI LIỆU THAM KHẢO

- [1]. Python Software Foundation. (2024). The tkinter package. Python documentation. Retrieved from <https://docs.python.org/3/library/tk.html>.
- [2]. Python Software Foundation. (2024). The json module. Python documentation. Retrieved from <https://docs.python.org/3/library/json.html>.
- [3]. Moore, A. D. (2018). Python GUI Programming with Tkinter. Packt Publishing.
- [4]. Michael Dawson – Python Programming for the Absolute Beginner – 3rd Edition
- [5]. <https://chatgpt.com/>