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 Tiến Thắng
Lớp	K58 KTPM
Giáo viên giảng dậy	TS.Nguyễn Văn Huy

Thái nguyên 2025

TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ

Bộ môn: Công nghệ Thông tin.

BÀI TẬP KẾT THÚC MÔN HỌC

TRÌNH QUẢN LÝ ỨNG DỤNG GUI

Sinh viên	Nguyễn Tiến Thắng
Lớp	K58 KTPM
Giáo viên giảng dậy	Ts.Nguyễn Văn Huy

Github	Youtube

Thái nguyên 2025

Biểu mẫu BTL02

TRƯỜNG ĐHKTCN **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ ***Độc lập - Tự do - Hạnh phúc***

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

Sinh viên: Nguyễn Tiến Thắng

Lớp: K58KTPM *Ngành:* Kĩ thuật máy tính

Giáo viên hướng dẫn:Ts. Nguyễn Văn Huy

Ngày giao đề 20/05/2025 *Ngày hoàn thành :* 06/06/2025

Tên đề tài : Trình quản lý thư mục GUI

Yêu cầu :

- *Sử dụng os để scan folder.*
- *Bắt lỗi không tìm thấy đường dẫn.*
- *GUI với Treeview (tkinter.ttk).*
- *Mở file bằng chương trình mặc định*

.....

.....

.....

.....

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày….tháng…..năm 20....

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

Mục Lục

Danh Mục Hình Ảnh

LỜI LẢM ƠN

Em xin gửi lời cảm ơn chân thành đến thầy Ts.Nguyễn Văn Huy – người đã tận tình giảng dạy môn Lập trình Python và hướng dẫn em trong quá trình thực hiện bài tập.

Nhờ sự chỉ bảo tận tình, kiến thức sâu rộng và tinh thần hỗ trợ của thầy, em đã có thể tiếp cận ngôn ngữ lập trình Python một cách hiệu quả, hiểu rõ hơn về tư duy lập trình cũng như ứng dụng thực tế của ngôn ngữ này.

Em xin trân trọng cảm ơn thầy và kính chúc thầy sức khỏe, thành công trong công tác giảng dạy và nghiên cứu.

LỜI NÓI ĐẦU

Trong bối cảnh công nghệ thông tin ngày càng phát triển, việc xây dựng các ứng dụng giao diện người dùng (GUI) để hỗ trợ thao tác trực quan, tiện lợi với hệ thống tệp là một yêu cầu phổ biến và thực tiễn. Với mục tiêu rèn luyện kỹ năng lập trình Python kết hợp với thư viện giao diện đồ họa tkinter, đề tài **"***Tạo ứng dụng GUI cho phép chọn thư mục, liệt kê file theo từng loại và cho phép mở file"* được lựa chọn nhằm giúp sinh viên vận dụng kiến thức đã học vào một bài toán cụ thể.

Ứng dụng được xây dựng với các chức năng chính như: cho phép người dùng chọn thư mục trên máy, tự động quét và liệt kê các tệp theo định dạng (.txt, .py, .jpg) trong một bảng Treeview, đồng thời cho phép mở tệp bằng chương trình mặc định của hệ thống. Trong quá trình phát triển, chương trình tận dụng thư viện os để xử lý đường dẫn và tệp, xử lý ngoại lệ khi không tìm thấy thư mục, và sử dụng tkinter.ttk để xây dựng giao diện thân thiện với người dùng.

Việc hoàn thành đề tài không chỉ giúp người thực hiện củng cố kiến thức về Python và lập trình hướng sự kiện với GUI, mà còn nâng cao tư duy thiết kế phần mềm ứng dụng thực tế.

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1 Mô tả đề tài

Đề tài yêu cầu phát triển một ứng dụng giao diện đồ họa (GUI) sử dụng ngôn ngữ lập trình Python và thư viện Tkinter trong môi trường Visual Studio Code. Ứng dụng cho phép người dùng chọn một thư mục, liệt kê các tệp tin có định dạng cụ thể (bao gồm .txt, .py, .jpg) trong một Treeview và hỗ trợ mở tệp tin bằng chương trình mặc định của hệ thống. Các thành phần chính bao gồm:

• Đầu vào: Một nút "Chọn thư mục" để người dùng chọn đường dẫn thư mục thông qua hộp thoại.

• Đầu ra: Một Treeview (hoặc Listbox) hiển thị danh sách các tệp tin với thông tin như tên tệp, loại tệp và đường dẫn; cùng với cơ chế mở tệp khi nhấp đúp chuột.

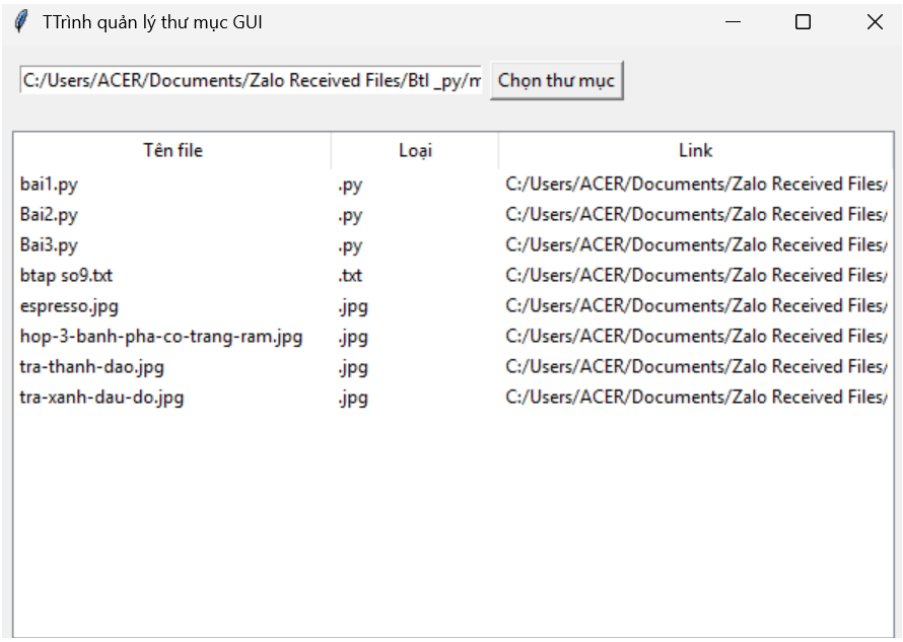
• Tính năng chính:

- Sử dụng module os để quét thư mục và lọc các tệp theo định dạng (.txt, .py, .jpg).
- Xử lý lỗi khi đường dẫn thư mục không hợp lệ (ví dụ: thư mục không tồn tại).
- Xây dựng giao diện người dùng với tkinter.ttk.Treeview để hiển thị danh sách tệp.
- Cho phép mở tệp bằng chương trình mặc định của hệ thống thông qua os.startfile.

• Kết quả mong đợi:

– Khi chọn một thư mục chứa ba tệp (ví dụ: file1.txt, script.py, image.jpg), Treeview hiển thị ba dòng tương ứng.

– Nhấp đúp vào tệp .txt sẽ mở tệp bằng Notepad (hoặc chương trình mặc định của hệ thống).



Hình1.1 Trình quản lý thư mục GUI

1.2 Thách thức của đề tài

Đề tài đặt ra một số thách thức kỹ thuật và logic như sau:

Quản lý giao diện GUI là một trong những thách thức quan trọng của đề tài. Việc sử dụng Tkinter để tạo giao diện trực quan đòi hỏi phải bố trí các thành phần như nút, Entry và Treeview sao cho thân thiện và thuận tiện với người dùng. Bên cạnh đó, việc xử lý lỗi cũng rất cần thiết, đặc biệt trong các trường hợp ngoại lệ như thư mục không tồn tại hoặc tệp không thể mở, nhằm đảm bảo ứng dụng hoạt động ổn định và không gặp sự cố nghiêm trọng.

Quá trình quét và lọc tệp sử dụng module os cũng đòi hỏi hiểu biết kỹ thuật vững chắc về thao tác với hệ thống tệp, để duyệt thư mục hiệu quả và lọc chính xác các định dạng tệp yêu cầu. Đồng thời, việc tích hợp sự kiện, như kết nối sự kiện nhấp đúp chuột trên Treeview với hàm mở tệp, cần được xử lý chính xác để đảm bảo đường dẫn tệp được chọn là hợp lệ và thao tác mở file diễn ra mượt mà.

Ngoài ra, một thách thức không nhỏ là đảm bảo tính tương thích của tính năng os.startfile trên các hệ điều hành, đặc biệt là Windows, và xử lý kịp thời các lỗi phát sinh khi tệp không thể mở được. Tất cả những khó khăn này đã góp phần giúp em nâng cao kỹ năng lập trình và xử lý tình huống thực tế trong quá trình phát triển ứng dụng.

1.3 Kiến thức vận dụng

Để hoàn thành đề tài, cần vận dụng các kiến thức sau:

Trong quá trình phát triển ứng dụng "Trình quản lý thư mục GUI", em đã vận dụng thành thạo ngôn ngữ lập trình Python và các kiến thức về lập trình hướng đối tượng để xây dựng cấu trúc chương trình rõ ràng, dễ bảo trì. Em sử dụng thư viện Tkinter cùng module ttk để thiết kế giao diện người dùng với các widget như nút bấm và Treeview giúp hiển thị danh sách file một cách trực quan.

Bên cạnh đó, em áp dụng module os để thao tác với hệ thống file, thực hiện việc quét thư mục, lọc file theo định dạng và mở file bằng chương trình mặc định của hệ điều hành. Việc xử lý ngoại lệ cũng được chú trọng nhằm đảm bảo ứng dụng hoạt động ổn định khi người dùng không chọn thư mục hoặc chọn đường dẫn không hợp lệ.

Đồng thời, kỹ thuật xử lý sự kiện trong GUI như nhấn nút và double-click vào Treeview đã được sử dụng hiệu quả để tăng cường tương tác và trải nghiệm người dùng.

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

2.1 Danh sách list trong python

Danh sách (List) là một trong những kiểu dữ liệu có cấu trúc quan trọng và được sử dụng phổ biến trong Python. List cho phép lưu trữ tập hợp các giá trị (gọi là phần tử), có thể thuộc bất kỳ kiểu dữ liệu nào, và có thể thay đổi (mutable). Mỗi phần tử trong list được đánh chỉ số (index) bắt đầu từ 0, giúp việc truy cập và xử lý dữ liệu trở nên linh hoạt.

VD Cú pháp khai báo

```
my_list = [1, 2, 3, 4, 5]
```

Trong chương trình, list được sử dụng một cách gián tiếp nhưng hiệu quả thông qua các hàm trả về danh sách hoặc làm việc với danh sách:

a , os.listdir(path): Trả về một list chứa tên các tệp và thư mục trong thư mục được chỉ định. Danh sách này được duyệt bằng vòng lặp for để lọc và xử lý các tệp theo định dạng mong muốn

```
files = os.listdir(folder_path)
    for file in files:
```

b,tree.get_children(): Trả về danh sách các phần tử đang hiển thị trong bảng Treeview. Danh sách này được sử dụng để xóa các phần tử cũ trước khi hiển thị danh sách tệp mới

```
for item in tree.get_children():
   tree.delete(item)
```

c,allowed_extensions = ('.txt', '.py', '.jpg'): Mặc dù được khai báo dưới dạng tuple, danh sách định dạng tệp này đóng vai trò tương tự một list, cho phép kiểm tra phần mở rộng của tệp khi lọc.

```
if file_name.lower().endswith(allowed_extensions):
```

2.2 Widget Treeview

Treeview là một widget đặc biệt thuộc thư viện tkinter.ttk (themed Tkinter), dùng để hiển thị dữ liệu theo dạng bảng có nhiều cột, hỗ trợ lựa chọn từng hàng, sắp xếp, tổ chức dữ liệu phân cấp, và gán các sự kiện tương tác như click hoặc double-click.

Trong chương trình, Treeview được sử dụng để hiển thị danh sách các tệp hợp lệ từ thư mục do người dùng chọn. Mỗi hàng (record) trong Treeview đại diện cho một tệp, bao gồm các thông tin: Tệp, link, đường dẫn
ví dụ Đặt tiêu đề cho từng cột và cấu hình chiều rộng:

```
tree.heading("Tên file", text="Tên file")
tree.heading("Loại", text="Loại")
tree.heading("Link", text="Link")
tree.column("Tên file", width=200)
tree.column("Loại", width=100)
tree.column("Link", width=250)
```

Lợi ích của Treeview là giúp người dùng có thể duyệt dữ liệu theo cấu trúc rõ ràng, dễ quan sát, hỗ trợ thao tác lựa chọn và tương tác linh hoạt — đặc biệt phù hợp với các chương trình quản lý tệp hoặc cơ sở dữ liệu dạng bảng.

2.3 Thư viện Tkinter

Tkinter là một thư viện trong ngôn ngữ lập trình Python được sử dụng để tạo giao diện đồ họa người dùng (GUI). "Tkinter" là viết tắt của "Tk interface,", một toolkit đồ họa cung cấp các công cụ để phát triển giao diện người dùng.

Tkinter là một phần của thư viện tiêu chuẩn của Python và đã được tích hợp sẵn trong hầu hết các cài đặt Python. Điều này giúp cho Tkinter trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng với giao diện đồ họa đơn giản trong Python.

Một số đặc điểm của Tkinter bao gồm khả năng tạo các thành phần giao diện như cửa sổ, nút, ô văn bản, và các widget khác để tương tác với người dùng. Tkinter cung cấp cả các sự kiện và phương thức để xử lý tương tác người dùng và thay đổi trạng thái của ứng dụng
VD tạo cửa sổ chính

```
root = tk.Tk()
root.title("Trình quản lý thư mục GUI")
root.geometry("600x400")
```

2.4 Module os

Module os là một thư viện tiêu chuẩn trong Python, cung cấp các phương thức giúp lập trình viên tương tác với hệ điều hành, đặc biệt là hệ thống tệp (file system). Trong ứng dụng, os đóng vai trò quan trọng trong việc truy cập thư mục, kiểm tra và xử lý các tệp tin.

a,os.listdir(path): Trả về danh sách các tệp và thư mục nằm trong đường dẫn path. Đây là cơ sở để chương trình duyệt và liệt kê các tệp có trong thư mục được chọn.

b,os.path.join(path, file): Dùng để kết hợp đường dẫn thư mục và tên tệp, đảm bảo tạo ra đường dẫn đầy đủ và đúng định dạng (phù hợp với hệ điều hành).

c,os.path.isfile(path): Kiểm tra xem một đường dẫn cụ thể có phải là tệp tin hay không. Điều này rất quan trọng để phân biệt tệp với thư mục, chỉ hiển thị các tệp trong danh sách.

d,os.startfile(path): Mở tệp bằng ứng dụng mặc định của hệ điều hành. Lưu ý: hàm này hoạt động chủ yếu trên hệ điều hành Windows

Ví dụ minh họa

```
import os

folder_path = "C:/Users/Documents"          # Đường dẫn đến thư mục
files = os.listdir(folder_path)             # Lấy danh sách tệp/thư mục

for file in files:
    full_path = os.path.join(folder_path, file)  # Kết hợp thành đường dẫn đầy đủ
    if os.path.isfile(full_path):               # Kiểm tra nếu là tệp
        print(full_path)                        # In đường dẫn tệp ra màn hình
```

2.5. Giao diện người dùng (GUI)

Giao diện người dùng (GUI) trong chương trình trên được thiết kế đơn giản và trực quan, sử dụng thư viện tkinter của Python. Khi chương trình được khởi chạy, một cửa sổ sẽ hiển thị với tiêu đề “Chọn thư mục”. Trong cửa sổ này, người dùng sẽ thấy một dòng nhãn hướng dẫn yêu cầu họ chọn một thư mục, kèm theo một ô nhập để hiển thị đường dẫn thư mục được chọn.

Ngay bên dưới ô nhập là một nút bấm có nhãn “Chọn thư mục”. Khi người dùng nhấn vào nút này, một hộp thoại sẽ xuất hiện cho phép họ duyệt qua các thư mục trên máy tính và chọn thư mục mong muốn. Sau khi chọn, đường dẫn đến thư mục sẽ tự động được chèn vào ô nhập, thay thế nội dung cũ (nếu có). Giao diện này phù hợp cho các ứng dụng cần nhập đầu vào là thư mục, như chọn nơi lưu trữ dữ liệu, tải file, hoặc xử lý hàng loạt tập tin trong một thư mục nhất định.

CHƯƠNG 3 THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

3.1 Sơ đồ khối hệ thống

3.1.1 Mô tả các module chính

Ứng dụng được chia thành ba module chính, mỗi module đảm nhiệm một chức năng cụ thể:

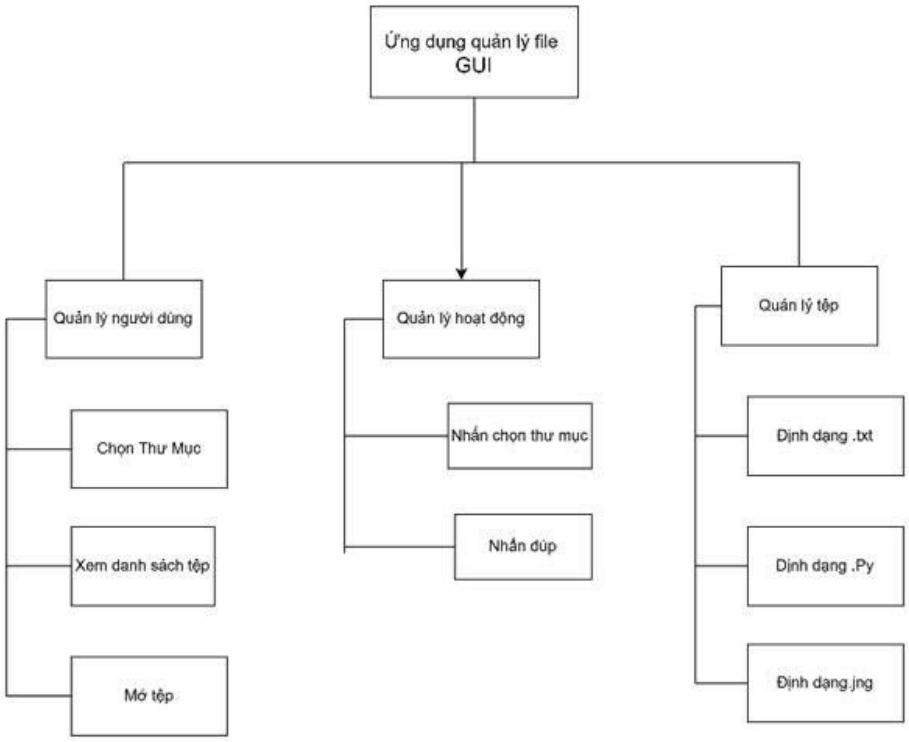
• Module giao diện người dùng (GUI): Sử dụng Tkinter để tạo cửa sổ chính, chứa các widget như Entry (ô nhập đường dẫn), Button (nút ”Chọn thư mục”), và Treeview (bảng hiển thị danh sách tệp). Module này chịu trách nhiệm hiển thị giao diện và tương tác với người dùng.

• Module quản lý tệp: Sử dụng module os để quét thư mục, lọc tệp theo định dạng (.txt, .py, .jpg), và cung cấp đường dẫn đầy đủ của tệp. Module này xử lý các thao tác liên quan đến hệ thống tệp.

• Module xử lý hoạt động : Quản lý các hoạt động như nhấp nút ”Chọn thư mục” hoặc nhấp đúp vào Treeview để mở tệp. Module này đảm bảo ứng dụng phản hồi chính xác với hành động của người dùng và xử lý lỗi (ví dụ: thư mục không tồn tại).

3.1.2 Biểu đồ phân cấp chức năng

Ứng dụng quản lý file với giao diện đồ họa (GUI) là một phần mềm hỗ trợ người dùng thao tác với các thư mục và tệp tin trên máy tính một cách trực quan. Thông qua giao diện đồ họa, người dùng có thể dễ dàng chọn thư mục, xem danh sách các tệp, mở và xử lý các tệp theo định dạng. Ứng dụng được xây dựng với các chức năng quản lý người dùng, Quản lý hoạt động và quản lý tệp



Hình 3.1 Sơ đồ phân cấp chức năng

Ứng dụng quản lý thư mục GUI bao gồm các chức năng chính được tổ chức theo phân cấp rõ ràng. Đầu tiên là chức năng Chọn thư mục, cho phép người dùng mở hộp thoại để lựa chọn thư mục cần quản lý. Sau khi người dùng chọn thư mục, đường dẫn của thư mục đó sẽ được hiển thị trong ô nhập liệu để người dùng dễ dàng theo dõi. Tiếp theo, ứng dụng sẽ gọi hàm để quét và hiển thị danh sách các file trong thư mục đã chọn.

Chức năng Hiển thị file trong thư mục đảm nhận việc quản lý bảng hiển thị dữ liệu. Ứng dụng sẽ xóa bỏ các mục dữ liệu cũ trong bảng trước khi tiến hành quét thư mục. Trong quá trình quét, chỉ những file có định dạng được phép như .txt, .py, hoặc .jpg mới được lọc ra và hiển thị lên bảng với các thông tin gồm tên file, loại file và đường dẫn đầy đủ.

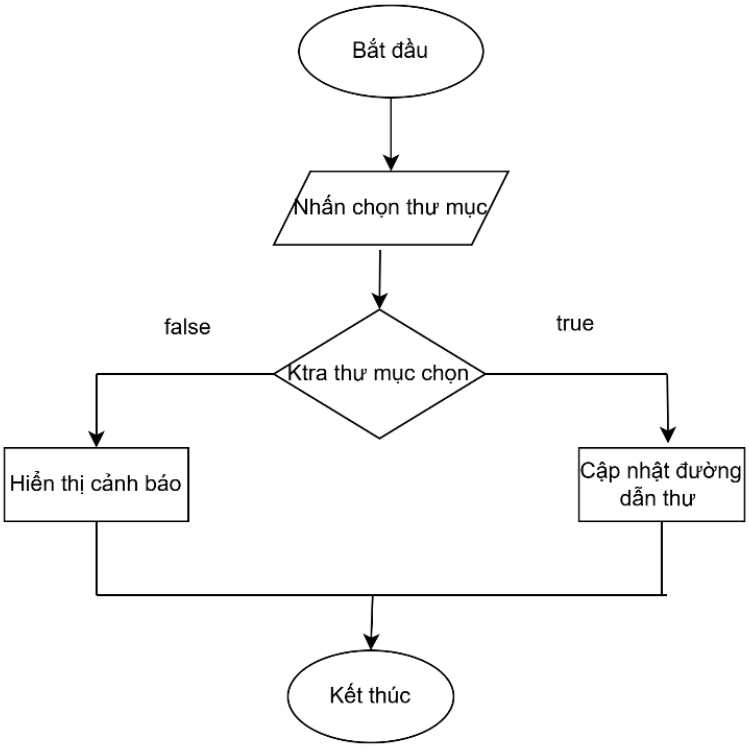
Chức năng Mở file giúp người dùng có thể truy cập nhanh đến nội dung file bằng cách bắt sự kiện nhấp đúp chuột trên dòng dữ liệu tương ứng trong bảng. Khi sự kiện này xảy ra, ứng dụng lấy đường dẫn file đã chọn và sử dụng chương trình mặc định trên hệ điều hành để mở file đó

Cuối cùng, phần Xử lý lỗi đảm bảo trải nghiệm người dùng được ổn định và trực quan hơn bằng cách hiển thị các cảnh báo và thông báo lỗi khi xảy ra các tình huống như người dùng không chọn thư mục, thư mục đã chọn không

tồn tại hoặc file không thể mở được do các nguyên nhân khác nhau. Điều này giúp ứng dụng tránh các lỗi không mong muốn và hướng dẫn người dùng xử lý tình huống một cách hợp lý.

3.2 Sơ đồ khối các thuật toán chính

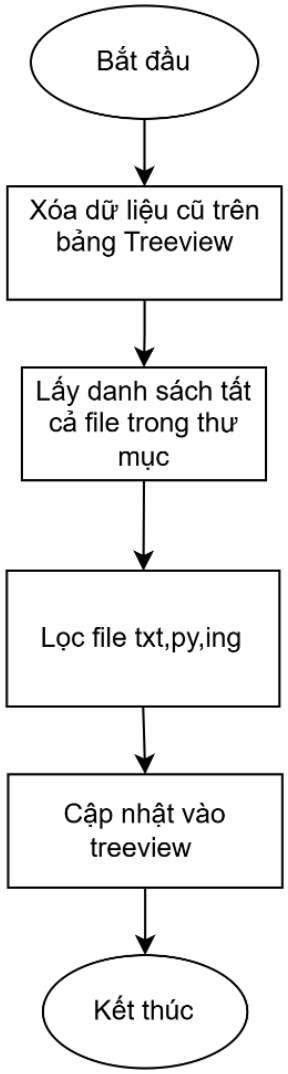
3.2.1 Thuật toán chọn thư mục



Hình 3.2 Sơ đồ khối chọn thư mục

Đây là thuật toán đầu tiên trong quy trình hoạt động của ứng dụng. Khi người dùng nhấn nút "Chọn thư mục", hệ thống mở hộp thoại chọn thư mục (dùng filedialog.askdirectory() của Tkinter). Sau khi người dùng chọn một thư mục, ứng dụng kiểm tra đầu vào. Nếu không có thư mục nào được chọn, chương trình sẽ hiển thị cảnh báo để yêu cầu chọn lại. Nếu có thư mục, đường dẫn sẽ được ghi vào ô nhập và chương trình gọi hàm tiếp theo để quét và hiển thị file trong thư mục đó. Thuật toán này đảm bảo tính tương tác ban đầu và kiểm soát đầu vào người dùng.

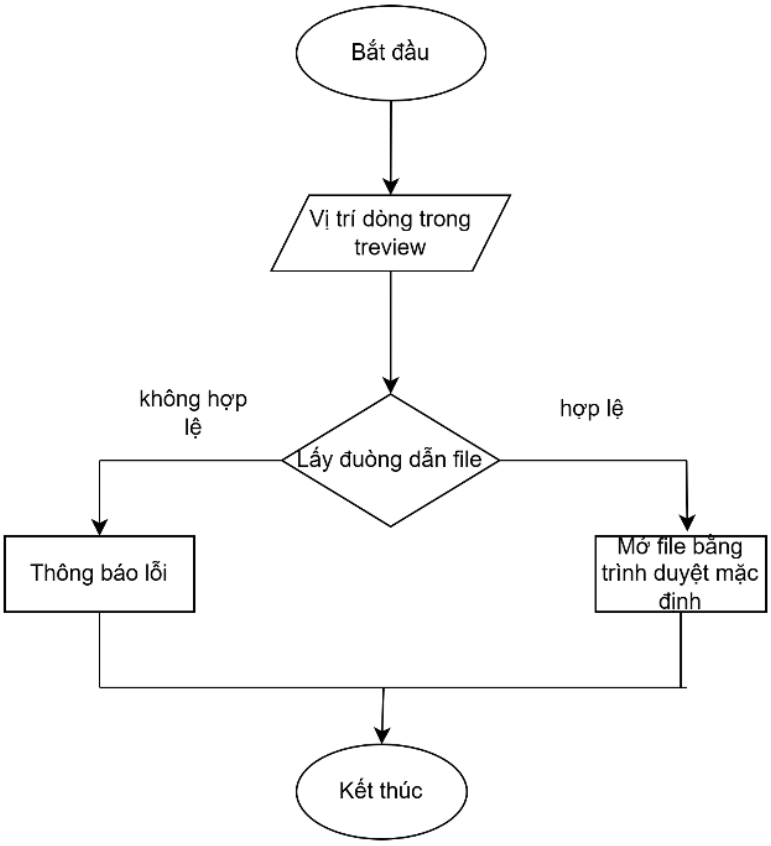
3.2.2 Thuật toán hiển thị danh sách tệp



Hình 3.3 Sơ đồ khối hiển thị danh sách tệp

Thuật toán này được gọi sau khi người dùng chọn một thư mục hợp lệ. Đầu tiên, chương trình xóa toàn bộ dữ liệu cũ đang hiển thị trong bảng Treeview. Sau đó, nó thực hiện việc quét danh sách các file trong thư mục đã chọn. File nào có đuôi nằm trong danh sách được phép (.txt, .py, .jpg) sẽ được xử lý tiếp: chương trình lấy tên file, phần mở rộng và đường dẫn đầy đủ rồi thêm thông tin đó vào bảng. Toàn bộ quá trình này được bao bọc bởi khối try-except để xử lý lỗi như không tìm thấy thư mục hoặc lỗi hệ thống.

3.2.3 Thuật toán mở tệp



Hình 3.4 Sơ đồ khối mở tệp

Đây là thuật toán xử lý sự kiện khi người dùng nhấp đúp chuột vào một dòng trong bảng Treeview. Hệ thống kiểm tra xem có dòng nào đang được chọn không, sau đó lấy đường dẫn file từ dòng đó. Tiếp theo, chương trình cố gắng mở file bằng lệnh os.startfile() (trên Windows) hoặc các lệnh hệ điều hành tương ứng nếu triển khai đa nền tảng. Nếu việc mở file gặp lỗi, hệ thống sẽ thông báo cho người dùng. Thuật toán này cho phép người dùng truy cập nhanh vào file mà không cần rời khỏi ứng dụng.

3.3 Cấu trúc dữ liệu

Trong ứng dụng quản lý thư mục và file này, cấu trúc dữ liệu được thiết kế nhằm lưu trữ và xử lý thông tin các tập tin trong thư mục do người dùng chọn. Mỗi file được biểu diễn bằng một tập hợp các thuộc tính cơ bản bao gồm tên file, định dạng (đuôi file) và đường dẫn đầy đủ trên hệ thống. Thông tin này được thu thập bằng cách quét thư mục và lọc theo các loại file cho phép nhằm đảm bảo tính chính xác và phù hợp với yêu cầu người dùng.

Dữ liệu các file sau khi được thu thập sẽ được lưu trữ tạm thời và hiển thị trực tiếp trong widget bảng Treeview của thư viện Tkinter. Cấu trúc dữ liệu này giúp quản lý và truy xuất thông tin file một cách hiệu quả, đồng thời hỗ trợ các thao tác như mở file khi người dùng tương tác (double-click). Việc thiết kế cấu

trúc dữ liệu đơn giản nhưng đầy đủ này giúp ứng dụng hoạt động linh hoạt, dễ dàng mở rộng và bảo trì trong các phiên bản tiếp theo.

Ví dụ

Tên file	Loại	Đường dẫn
bai1.py	.py	C:/Users/ACER/Documents/Zalo Received Files/
Bai2.py	.py	C:/Users/ACER/Documents/Zalo Received Files/
Bai3.py	.py	C:/Users/ACER/Documents/Zalo Received Files/
espresso.jpg	.jpg	C:/Users/ACER/Documents/Zalo Received Files/
hop-3-banh-pha-co-trang-ram.jpg	.jpg	C:/Users/ACER/Documents/Zalo Received Files/
latte.jpg	.jpg	C:/Users/ACER/Documents/Zalo Received Files/
New Tài liệu văn bản.txt	.txt	C:/Users/ACER/Documents/Zalo Received Files/
tra-thanh-dao.jpg	.jpg	C:/Users/ACER/Documents/Zalo Received Files/
tra-xanh-dau-do.jpg	.jpg	C:/Users/ACER/Documents/Zalo Received Files/

Hình 3.5 trường thông tin

3.4 Chương trình

Chương trình được xây dựng với mục đích hiển thị danh sách các file trong một thư mục do người dùng lựa chọn, đồng thời cung cấp tính năng mở nhanh các file đó bằng cách nháy đúp chuột vào từng mục trong danh sách. Ba hàm chính của chương trình bao gồm:

3.4.1 Hàm select_folder()

```
def select_folder():  # Hàm xử lý sự kiện khi người dùng nhấn nút "Chọn thư mục"
    folder_path = filedialog.askdirectory()  # Hiển thị hộp thoại chọn thư mục và lưu đường dẫn đã chọn
    if folder_path:  # Kiểm tra nếu người dùng đã chọn một thư mục (không để trống)
        folder_entry.delete(0, tk.END)  # Xóa nội dung cũ trong ô nhập đường dẫn
        folder_entry.insert(0, folder_path)  # Hiển thị đường dẫn vừa chọn vào ô nhập
        show_files(folder_path)  # Gọi hàm show_files để hiển thị danh sách file trong thư mục đó
    else:
        messagebox.showwarning("Cảnh báo", "Vui lòng chọn một thư mục!")
# Thông báo nếu người dùng chưa chọn thư mục
```

Hàm select_folder() được thiết kế để xử lý sự kiện khi người dùng nhấn nút "Chọn thư mục" trong giao diện ứng dụng. Mục tiêu của hàm là giúp người dùng dễ dàng lựa chọn một thư mục trên hệ thống, đồng thời cập nhật giao diện và hiển thị danh sách các file trong thư mục đó.

Quá trình thực hiện của hàm gồm các bước sau:

1. Hiển thị hộp thoại chọn thư mục: Khi hàm được gọi, nó sẽ mở một cửa sổ hộp thoại để người dùng duyệt và chọn thư mục mong muốn trên máy tính, thông qua phương thức filedialog.askdirectory(). Đường dẫn của thư mục được người dùng chọn sẽ được lưu vào biến folder_path.
2. Kiểm tra thư mục đã chọn: Hàm sẽ kiểm tra xem người dùng có thực sự chọn một thư mục hay không (tức là biến folder_path không rỗng). Nếu người dùng không chọn, hàm sẽ thông báo cảnh báo yêu cầu người dùng chọn một thư mục hợp lệ.
3. Cập nhật giao diện: Nếu người dùng đã chọn thư mục, hàm sẽ xóa nội dung cũ trong ô nhập đường dẫn thư mục (folder_entry) và chèn đường dẫn mới được chọn vào đó, giúp người dùng dễ dàng theo dõi thư mục đang làm việc.
4. Hiển thị danh sách file: hàm gọi hàm show_files(folder_path) để hiển thị danh sách các file có định dạng được phép trong thư mục vừa chọn lên bảng Treeview.

3.4.2, hàm def show_files(folder_path):

```
def show_files(folder_path):  # Hàm hiển thị danh sách các file trong thư mục được chọn
    try:
        # Xóa dữ liệu cũ trong Treeview trước khi thêm dữ liệu mới
        for item in tree.get_children():
            tree.delete(item)

        # Định nghĩa các phần mở rộng file được phép hiển thị
        allowed_extensions = ('.txt', '.py', '.jpg')
        # Duyệt qua tất cả các tên file và thư mục trong folder_path
        for file_name in os.listdir(folder_path):
            # Kiểm tra file có phần mở rộng nằm trong allowed_extensions không
            if file_name.lower().endswith(allowed_extensions):
                file_path = os.path.join(folder_path, file_name)  # Tạo đường dẫn đầy đủ đến file
                if os.path.isfile(file_path):  # Kiểm tra chắc chắn đây là file, không phải thư mục
                    ext = os.path.splitext(file_name)[1].lower()  # Lấy phần mở rộng file và chuyển thành chữ thường
                    # Thêm dòng mới vào Treeview với các cột: tên file, phần mở rộng, đường dẫn
```

```
                tree.insert("", tk.END, values=(file_name, ext, file_path))
    except FileNotFoundError:  # Xử lý trường hợp thư mục không tồn tại hoặc không tìm thấy
        messagebox.showerror("Lỗi", "Không tìm thấy thư mục!")
    except Exception as e:  # Bắt các lỗi khác và hiển thị thông báo lỗi
        messagebox.showerror("Lỗi", f"Đã xảy ra lỗi: {str(e)}")
```

Hàm show_files(folder_path) có nhiệm vụ hiển thị danh sách các tệp tin trong thư mục được người dùng chọn lên giao diện dưới dạng bảng (Treeview). Hàm giúp lọc và liệt kê các file theo các định dạng được phép, đồng thời cập nhật lại dữ liệu hiển thị khi thư mục thay đổi.

Quá trình thực hiện của hàm bao gồm các bước chính như sau:

1. Xóa dữ liệu cũ: Trước khi hiển thị dữ liệu mới, hàm sẽ xóa toàn bộ các dòng hiện có trong widget Treeview bằng cách duyệt và gọi tree.delete() cho từng mục, nhằm tránh việc hiển thị chồng lặp hoặc sai lệch.
2. Định nghĩa phần mở rộng file cho phép: Hàm chỉ hiển thị các tệp tin có phần mở rộng nằm trong bộ lọc cho phép, gồm: .txt (file văn bản), .py (file mã nguồn Python), và .jpg (file ảnh). Việc này giúp giới hạn danh sách file phù hợp với mục đích sử dụng và tăng tính trực quan cho người dùng.
3. Duyệt thư mục và lọc file: Hàm sử dụng os.listdir() để lấy danh sách tất cả tên file và thư mục con trong đường dẫn folder_path. Với từng tên file, hàm kiểm tra xem nó có phần mở rộng thuộc bộ lọc hay không, và chắc chắn đây là file chứ không phải thư mục con (sử dụng os.path.isfile()).
4. Hiển thị file lên Treeview: Đối với mỗi file hợp lệ, hàm sẽ lấy phần mở rộng, sau đó chèn một dòng mới vào bảng Treeview với các thông tin gồm: tên file, phần mở rộng, và đường dẫn đầy đủ đến file.
5. Xử lý lỗi: Hàm có cơ chế bắt lỗi đối với các trường hợp thư mục không tồn tại hoặc không thể truy cập, hiển thị hộp thoại cảnh báo lỗi cho người dùng. Đồng thời, các lỗi phát sinh khác cũng được bắt và thông báo cụ thể.

3.4.3 Hàm def open_file(event):

```
def open_file(event):  # Hàm xử lý sự kiện khi người dùng double-click vào một hàng trong Treeview
    selected_item = tree.selection()  # Lấy danh sách các mục (items) được chọn trong Treeview
    if selected_item:  # Nếu có ít nhất một mục được chọn
        # Lấy đường dẫn file từ cột thứ 3 (index 2) của mục được chọn
        file_path = tree.item(selected_item)['values'][2]
        try:
            os.startfile(file_path)  # Mở file bằng ứng dụng mặc định của hệ điều hành (chỉ trên Windows)
        except Exception as e:  # Nếu xảy ra lỗi khi mở file (ví dụ file bị xóa hoặc không có quyền truy cập)
            messagebox.showerror("Lỗi", f"Không thể mở file: {str(e)}")
# Hiển thị thông báo lỗi cho người dùng
```

Hàm open_file(event) được xây dựng nhằm xử lý sự kiện khi người dùng thực hiện thao tác nháy đúp (double-click) vào một hàng dữ liệu trong widget Treeview. Mục đích của hàm là giúp người dùng có thể mở nhanh tệp tin tương ứng với dòng dữ liệu được chọn mà không cần phải tìm kiếm thủ công.
Cụ thể, quá trình thực hiện của hàm bao gồm các bước sau:

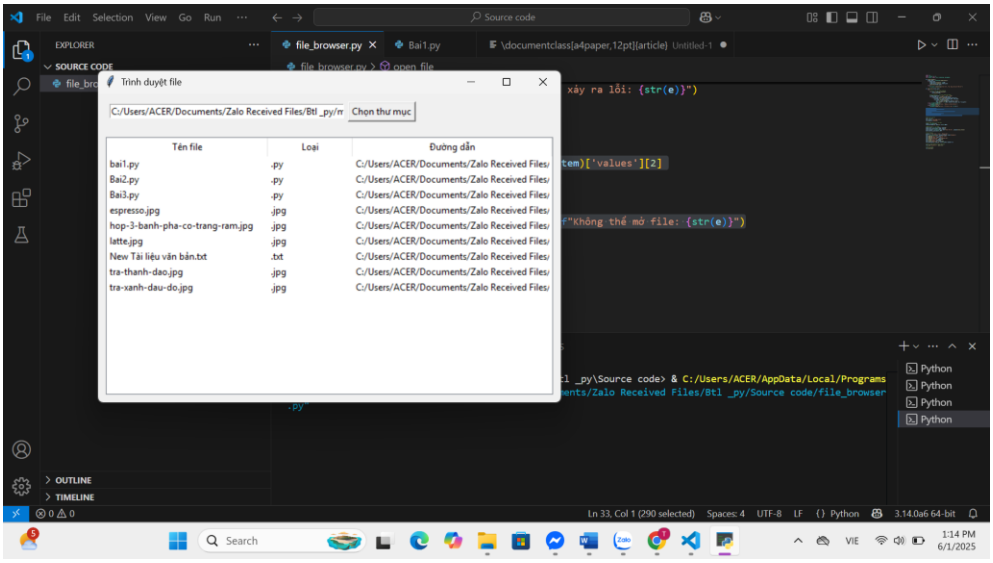
1. Lấy mục được chọn: Hàm sẽ lấy danh sách các mục (items) đang được chọn trong Treeview thông qua phương thức tree.selection(). Hàm sẽ kiểm tra xem có ít nhất một mục được chọn hay không.
2. Lấy đường dẫn file: Nếu có mục được chọn, hàm sẽ truy xuất đến giá trị của cột thứ 3 (chỉ số 2) trong hàng đó, đây là nơi chứa đường dẫn tuyệt đối hoặc tương đối của file cần mở.
3. Mở file: Hàm sử dụng phương thức os.startfile(file_path) để mở file bằng ứng dụng mặc định của hệ điều hành Windows. Đây là cách thức đơn giản và hiệu quả để khởi chạy tài liệu hoặc file đa phương tiện ngay trong giao diện phần mềm.
4. Xử lý lỗi: Trong trường hợp xảy ra lỗi khi mở file (ví dụ file đã bị xóa, đường dẫn không tồn tại hoặc không có quyền truy cập), hàm sẽ bắt ngoại lệ và hiển thị hộp thoại cảnh báo lỗi (messagebox.showerror) với thông báo cụ thể nhằm thông báo cho người dùng biết nguyên nhân gây lỗi.

CHƯƠNG 4 THỰC NGHIỆM VÀ KẾT LUẬN

4.1. Thực nghiệm

Chạy và ghi lại kết quả các bài test các tính năng cùa sản phẩm, đánh giá chất lượng

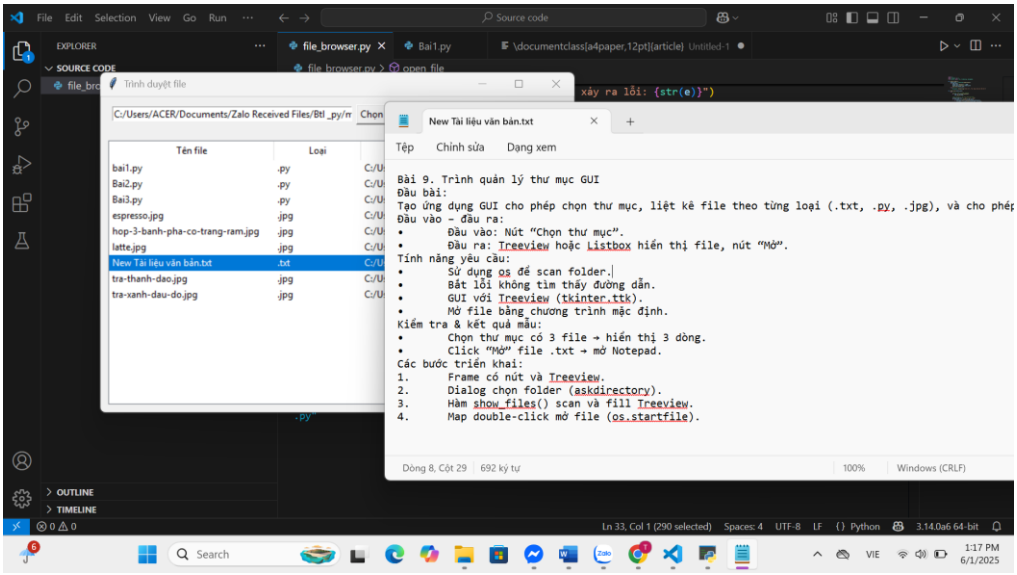
- Giao diện hoạt động đúng yêu cầu: chọn thư mục, lọc file đúng định dạng cho phép và hiển thị trong bảng.
- Hiển thị đầy đủ thông tin theo 3 cột: tên, định dạng và đường dẫn.
- Có thể mở file trực tiếp bằng cách nhấp đúp chuột (như đã cài đặt với hàm open_file(event)).



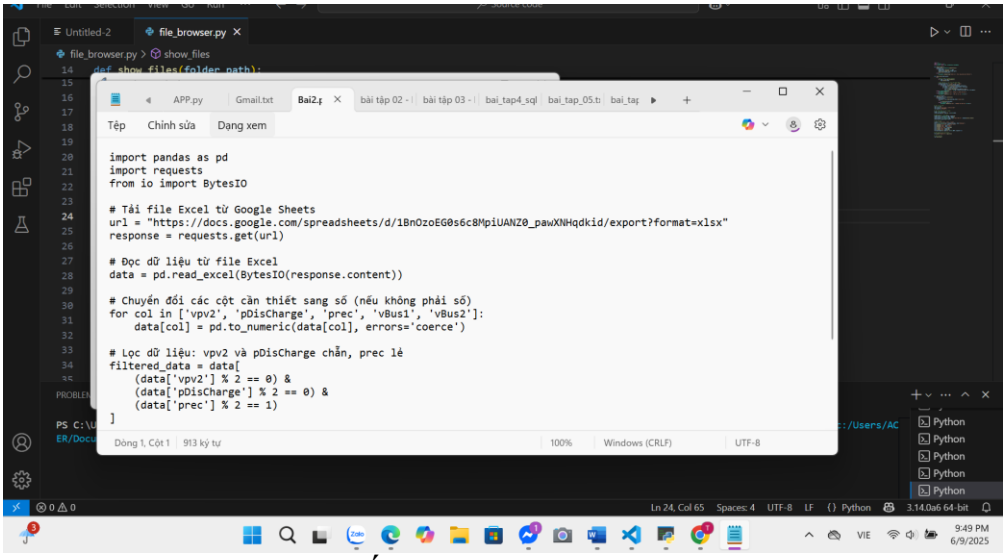
Hình 4.1 Giao diện thực nghiệm

Kiểm tra & kết quả mẫu:

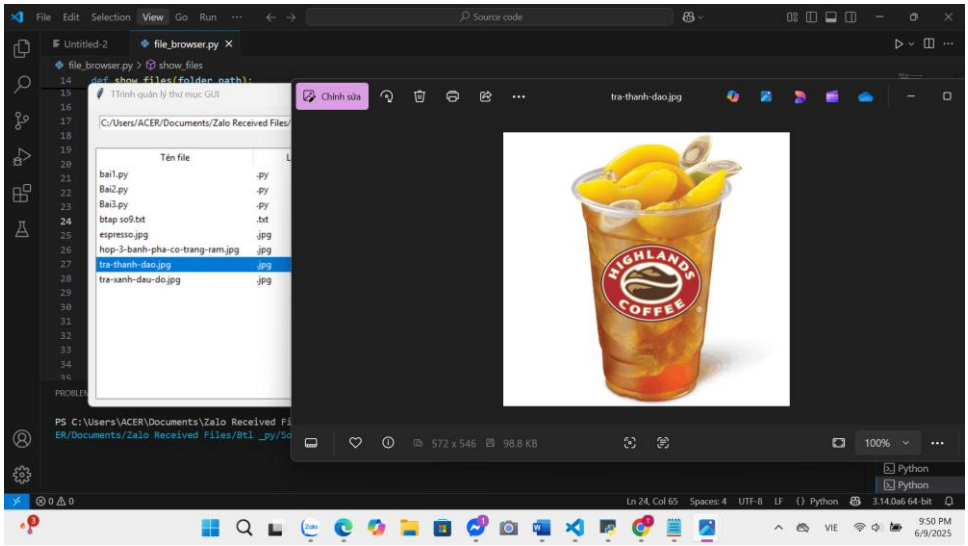
- Chọn thư mục có 3 file → hiển thị 3 dòng.
- Click "Mở" file .txt → mở Notepad.
- Click "Mở" file .py→ mở Notepad.(mặc định)
- Click "Mở" file .jpg → mở Ảnh
-



Hình 4.2 kết quả thực nghiệm mở file txt



Hình 4.3 kết quả thực nghiệm mở file py



Hình 4.4 kết quả thực nghiệm mở file jpg

4.2 Kết Luận

4.2.1 Sản phẩm đã làm được :

Người dùng có thể chọn một thư mục thông qua nút "Chọn thư mục", sử dụng hộp thoại filedialog.askdirectory. Đường dẫn của thư mục được chọn sẽ được hiển thị trong ô nhập (Entry). Đồng thời, chương trình kiểm tra tính hợp lệ của đường dẫn để đảm bảo người dùng không chọn sai hoặc đường dẫn không tồn tại.

Ứng dụng sử dụng module os để quét thư mục đã chọn và lọc ra các tệp có định dạng .txt, .py và .jpg. Sau đó, thông tin về tệp như tên tệp, loại tệp và đường dẫn được hiển thị trên giao diện bằng widget Treeview, giúp người dùng dễ dàng quan sát và thao tác với các tệp đó một cách trực quan.

Khi người dùng nhấp đúp vào một hàng trong Treeview, ứng dụng sẽ mở tệp tương ứng bằng chương trình mặc định của hệ thống. Ví dụ, tệp .txt sẽ được mở bằng Notepad trên Windows hoặc ứng dụng tương tự trên hệ điều hành khác, giúp thao tác trở nên thuận tiện và nhanh chóng.

Ứng dụng được trang bị cơ chế bắt lỗi nhằm xử lý các tình huống như thư mục không tồn tại (FileNotFoundError) hoặc người dùng không chọn thư mục. Các lỗi này sẽ được thông báo cho người dùng thông qua hộp thoại messagebox với các thông báo thân thiện, giúp người dùng dễ hiểu và xử lý tình huống.

Qua quá trình thử nghiệm, ứng dụng hoạt động ổn định. Khi chọn một thư mục chứa ba tệp ví dụ như file1.txt, script.py, và image.jpg, Treeview hiển thị chính xác ba dòng tương ứng. Ngoài ra, thao tác nhấp đúp vào tệp .txt sẽ mở tệp đó bằng chương trình mặc định, đáp ứng đầy đủ yêu cầu của bài toán.

4.2.2 Học được từ Sản Phẩm

Trong quá trình phát triển ứng dụng, tôi đã nắm vững cách sử dụng các cấu trúc dữ liệu như danh sách (list) để lọc và quản lý tệp. Bên cạnh đó, việc xử lý ngoại lệ bằng khối lệnh try-except được áp dụng nhằm đảm bảo chương trình hoạt động ổn định, không bị gián đoạn khi gặp các lỗi phát sinh trong quá trình xử lý dữ liệu.

Tôi đã tìm hiểu và thực hành xây dựng giao diện đồ họa bằng thư viện Tkinter. Qua đó, tôi sử dụng các widget phổ biến như Frame, Entry, Button và Treeview để tạo các thành phần giao diện trực quan, dễ sử dụng. Ngoài ra, việc

gắn sự kiện (bind) được áp dụng nhằm xử lý các hành động từ người dùng một cách hiệu quả và linh hoạt.

Module os được sử dụng để tương tác với hệ thống tệp trong máy tính. Tôi đã vận dụng các hàm như os.listdir để liệt kê các tệp trong thư mục, os.path.join để ghép nối đường dẫn, os.path.isfile để kiểm tra xem một đối tượng có phải là tệp tin hay không, và os.startfile để mở các tệp bằng ứng dụng mặc định trên hệ điều hành.

Quá trình phát triển ứng dụng được thực hiện trên môi trường Visual Studio Code, nơi tôi đã thành thạo trong việc cấu hình các tiện ích mở rộng hỗ trợ lập trình Python, đặc biệt là Pylance. Việc chạy mã, kiểm tra lỗi và gỡ lỗi thông qua breakpoint giúp tối ưu hóa thời gian phát triển và nâng cao chất lượng phần mềm.

Tôi đã học cách thiết kế phần mềm theo mô hình phân chia thành các module riêng biệt như giao diện, quản lý tệp và xử lý sự kiện. Đồng thời, việc xây dựng sơ đồ khối chức năng giúp tổ chức cấu trúc chương trình một cách rõ ràng và dễ dàng trong việc bảo trì, phát triển về sau.

4.2.3 Cải tiến Sản phẩm

Ứng dụng có thể được bổ sung tính năng cho phép người dùng lựa chọn loại tệp muốn hiển thị. Ví dụ, người dùng có thể chọn chỉ xem các tệp .txt hoặc .jpg thông qua menu thả xuống hoặc checkbox. Điều này giúp nâng cao khả năng tùy biến và giúp người dùng tìm kiếm tệp nhanh hơn.

Ngoài việc hiển thị tên và loại tệp, Treeview có thể được mở rộng thêm các cột thông tin như kích thước tệp, ngày sửa đổi gần nhất hoặc quyền truy cập. Việc này giúp người dùng có cái nhìn chi tiết và đầy đủ hơn về các tệp trong thư mục.

Hàm os.startfile chỉ hoạt động trên Windows, nên để ứng dụng có thể chạy trên macOS và Linux, có thể thay thế bằng subprocess.run hoặc các phương thức khác tương thích đa hệ điều hành. Điều này mở rộng phạm

Có thể thêm các nút chức năng mới như “Xóa” hoặc “Sao chép tệp” để hỗ trợ thao tác trực tiếp trên các tệp trong giao diện. Ngoài ra, hỗ trợ kéo-thả thư mục vào ô nhập đường dẫn sẽ giúp người dùng thao tác nhanh hơn và thuận tiện hơn

Tính năng lưu lại các thư mục đã chọn trước đó sẽ giúp người dùng dễ dàng truy cập lại các thư mục thường dùng mà không cần phải chọn lại từ đầu, từ đó nâng cao trải nghiệm và hiệu quả sử dụng.

KẾT LUẬN

Qua quá trình thực hiện đề tài "Trình quản lý thư mục GUI", em đã xây dựng thành công một ứng dụng với giao diện đồ họa thân thiện, cho phép người dùng dễ dàng chọn thư mục, hiển thị danh sách các file theo định dạng .txt, .py, .jpg trên Treeview, đồng thời hỗ trợ mở file bằng chương trình mặc định của hệ điều hành.

Việc phát triển ứng dụng đã giúp em nâng cao kỹ năng lập trình Python, đặc biệt là kỹ năng sử dụng thư viện tkinter để thiết kế giao diện GUI, thao tác với hệ thống file qua module os, cũng như xử lý các tình huống ngoại lệ như lỗi khi không tìm thấy đường dẫn hoặc người dùng không chọn thư mục. Ngoài ra, em còn hiểu rõ hơn về cách tổ chức giao diện người dùng sao cho trực quan và tiện lợi, cũng như cách kết hợp các sự kiện như click nút, double-click để nâng cao trải nghiệm người dùng.

Mặc dù ứng dụng đã hoàn thành các yêu cầu cơ bản, em nhận thấy vẫn còn một số điểm có thể cải tiến như tối ưu hiệu suất khi xử lý thư mục có nhiều file, bổ sung thêm các loại file hỗ trợ, hoặc mở rộng các tính năng quản lý như xóa, đổi tên file ngay trên giao diện. Đây sẽ là hướng phát triển tiếp theo giúp ứng dụng hoàn thiện và hữu ích hơn.

Tổng kết lại, đề tài không chỉ giúp em áp dụng kiến thức lý thuyết vào thực tế mà còn trang bị cho em những kỹ năng thực hành cần thiết để phát triển các ứng dụng phần mềm có giao diện người dùng trong tương lai.

Tài liệu tham khảo

1.Lutz, M. (2013). *Learning Python* (5th ed.). O'Reilly Media.

2.Sweigart, A. (2015). *Automate the Boring Stuff with Python: Practical Programming for Total Beginners*. No Starch Press. https://automatetheboringstuff.com

3.Python Software Foundation. *The Python Tutorial*. Truy cập tại: https://docs.python.org/3/tutorial/

4.Beazley, D., & Jones, B. (2013). *Python Cookbook* (3rd ed.). O'Reilly Media.

5.Martelli, A., Ravenscroft, A., & Ascher, D. (2005). *Python in a Nutshell* (2nd ed.). O'Reilly Media.