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 THỰC HIÊN | : | NGUYÊN KHÁNH DUY |
| MSV | : | K225480106008 |
| LỚP | : | K58KTP |
| GIÁO VIÊN GIẢNG DẠY | : | NGUYỄN VĂN HUY |


Link github: https://github.com/nguyenkhanhduy05/BTL_PyThon.git

Link youtube: https://www.youtube.com/watch?v=58jQam8ik3s

THÁI NGUYÊN - 2025

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

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

# BÀI TẬP KẾT THÚC MÔN

## MÔN HỌC: LẬP TRÌNH PYTHON

## BỘ MÔN: CÔNG NGHỆ THÔNG TIN

**Sinh viên:** Nguyên Khánh Duy Msv: K225480106008

**Lớp:** K58KTPM Ngành : KĨ thuật phần mềm

**Giáo viên giảng dậy:** Nguyễn Văn Huy

Ngày giao đề tài : 20/05/2025 Ngày hoàn thành: 1/06/2025

Tạo ứng dụng GUI cho phép chọn thư mục, liệt kê file theo từng loại (.txt, .py, .jpg), và cho phép mở file.Yêu cầu:

**Đầu vào – đầu ra:**

- Đầu vào: Nút “Chọn thư mục”.
- Đầu ra: Treeview hoặc Listbox hiển thị file, nút “Mở”.

**Tính năng 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

……

……

……

……

……

……

……

Xếp loại:…… Điểm :……

Thái Nguyên, ngày….tháng…..năm 20....

**GIÁO VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

# MỤC LỤC

# LỜI CAM ĐOAN

Tôi xin cam đoan rằng bài tập lớn môn Lập trình Python với đề tài "Trình quản lý thư mục GUI" được thực hiện bởi chính tôi, sinh viên Nguyên Khánh Duy, lớp K58KTPM, dưới sự hướng dẫn của thầy Nguyễn Văn Huy.

Toàn bộ nội dung trong báo cáo này, bao gồm quá trình thiết kế, phát triển, và thử nghiệm ứng dụng, là kết quả của sự nỗ lực và nghiên cứu độc lập của tôi. Tôi cam kết không sao chép hoặc sử dụng bất kỳ tài liệu, mã nguồn, hoặc nội dung nào từ các nguồn không được phép mà không trích dẫn rõ ràng. Các thông tin, số liệu, và kết quả thực nghiệm trình bày trong báo cáo đều trung thực và phản ánh đúng quá trình thực hiện đề tài.

Tôi chịu hoàn toàn trách nhiệm về tính trung thực và chính xác của nội dung báo cáo này. Nếu có bất kỳ sai phạm nào liên quan đến tính trung thực, tôi sẵn sàng chịu mọi hình thức xử lý theo quy định của nhà trường.

Tên Sinh Viên

Nguyên Khánh Duy

# LỜI NÓI ĐẦU

Bài tập lớn môn Lập trình Python là một cơ hội để sinh viên áp dụng các kiến thức đã học vào việc giải quyết một bài toán thực tế. Đề tài "Trình quản lý thư mục GUI" yêu cầu xây dựng một ứng dụng giao diện đồ họa sử dụng Python, cho phép người dùng chọn thư mục, liệt kê các file có phần mở rộng .txt, .py, .jpg, và mở file bằng chương trình mặc định của hệ điều hành. Ứng dụng không chỉ đòi hỏi kỹ năng lập trình mà còn yêu cầu sự hiểu biết về giao diện người dùng, xử lý hệ thống file, và quản lý lỗi.

Báo cáo này được thực hiện bởi sinh viên Nguyên Khánh Duy dưới sự hướng dẫn của thầy Nguyễn Văn Huy. Báo cáo trình bày chi tiết quá trình thiết kế, phát triển, và thử nghiệm ứng dụng, bao gồm cơ sở lý thuyết, phương pháp triển khai, và kết quả thực nghiệm. Mục tiêu là thể hiện sự độc lập trong tư duy, khả năng áp dụng kiến thức, và những bài học rút ra từ quá trình thực hiện đề tài. Trong quá trình thực hiện, tôi đã nỗ lực để đảm bảo ứng dụng đáp ứng đầy đủ yêu cầu đề bài, đồng thời học hỏi thêm về cách tích hợp giao diện đồ họa với xử lý hệ thống file. Báo cáo này không chỉ là kết quả của quá trình làm việc mà còn phản ánh sự cố gắng trong việc tìm hiểu và áp dụng các công cụ lập trình Python vào thực tiễn.

# CHƯƠNG 1: GIỚI THIỆU ĐẦU BÀI

## 1.1. Mô tả đề tài

Đề tài "Trình quản lý thư mục GUI" yêu cầu xây dựng một ứng dụng giao diện đồ họa (GUI) sử dụng ngôn ngữ lập trình Python để hỗ trợ người dùng quản lý và truy cập các file trong một thư mục cụ thể. Ứng dụng được thiết kế để cung cấp một giao diện trực quan, cho phép người dùng thực hiện các thao tác như chọn thư mục, xem danh sách file, và mở file bằng chương trình mặc định của hệ điều hành. Đề tài được giao bởi thầy Nguyễn Văn Huy cho sinh viên Nguyên Khánh Duy với thời gian thực hiện từ ngày 20/05/2025 đến ngày 01/06/2025.

Các tính năng chính của ứng dụng bao gồm:

a) Chọn thư mục: Người dùng nhấn nút "Chọn Thư Mục" để mở hộp thoại chọn một thư mục trên hệ thống. Hộp thoại này giúp người dùng dễ dàng duyệt và chọn thư mục cần quản lý.

b) Liệt kê file: Ứng dụng hiển thị danh sách các file có phần mở rộng .txt (văn bản), .py (mã nguồn Python), và .jpg (hình ảnh) trong thư mục đã chọn. Danh sách được trình bày trong một bảng (Treeview) với các cột: Tên File, Loại File, và Đường Dẫn đầy đủ.

c) Mở file: Người dùng có thể mở file bằng cách nhấn nút "Mở File" hoặc nhấp đúp chuột vào file trong bảng. File sẽ được mở bằng chương trình mặc định của hệ điều hành (ví dụ: file .txt mở bằng Notepad, file .jpg mở bằng trình xem ảnh).

d) Xử lý lỗi: Ứng dụng phải phát hiện và thông báo các lỗi như thư mục không tồn tại, không có quyền truy cập, hoặc file không thể mở, đảm bảo trải nghiệm người dùng mượt mà.

Mục tiêu của đề tài là tạo ra một công cụ thân thiện, giúp người dùng quản lý các file văn bản, mã nguồn, và hình ảnh một cách hiệu quả, đồng thời thể hiện khả năng tích hợp giao diện đồ họa với xử lý hệ thống file.

## 1.2. Thách thức của đề tài

Đề tài đặt ra một số thách thức kỹ thuật và thiết kế, đòi hỏi sự kết hợp giữa kiến thức lập trình và kỹ năng giải quyết vấn đề:

a) Tích hợp giao diện đồ họa với hệ thống file: Việc xây dựng một giao diện trực quan đồng thời xử lý các thao tác hệ thống file như quét thư mục và mở file yêu cầu sự đồng bộ giữa các thành phần giao diện (GUI) và logic xử lý backend.

b) Lọc file theo phần mở rộng: Ứng dụng chỉ hiển thị các file có phần mở rộng .txt, .py, và .jpg, đòi hỏi phải xử lý chính xác phần mở rộng và không phân biệt chữ hoa/thường để đảm bảo tính nhất quán.

c) Xử lý lỗi đa dạng: Ứng dụng cần phát hiện và xử lý các trường hợp lỗi như thư mục không tồn tại, thiếu quyền truy cập, hoặc file không thể mở. Việc hiển thị thông báo lỗi thân thiện với người dùng là một thách thức để cải thiện trải nghiệm.

d) Thiết kế giao diện trực quan: Giao diện cần hiển thị thông tin file một cách rõ ràng, với các cột được định dạng hợp lý (Tên File, Loại File, Đường Dẫn) và hỗ trợ cuộn dọc khi danh sách file dài.

e) Tính tương thích với hệ điều hành: Ứng dụng sử dụng phương thức mở file mặc định của Windows, nhưng cần cân nhắc khả năng mở rộng cho các hệ điều hành khác như macOS hoặc Linux trong tương lai.

Những thách thức này yêu cầu sinh viên không chỉ nắm vững lý thuyết mà còn phải áp dụng linh hoạt các kỹ năng lập trình để xây dựng một ứng dụng ổn định và hiệu quả.

## 1.3. Kiến thức vận dụng

Để thực hiện đề tài, các kiến thức và công cụ sau đã được vận dụng:

a) Thư viện tkinter và ttk: Đây là các thư viện chuẩn của Python để xây dựng giao diện đồ họa. Thư viện tkinter cung cấp các thành phần như cửa sổ, khung, nhãn, và hộp thoại chọn thư mục. Module ttk (themed Tkinter) được

sử dụng để tạo các thành phần giao diện hiện đại như bảng Treeview, nút Button, và thanh cuộn Scrollbar.

b) Module os: Module os cung cấp các hàm để tương tác với hệ thống file, bao gồm liệt kê file trong thư mục, kiểm tra xem một đường dẫn có phải là file, tách phần mở rộng file, và mở file bằng chương trình mặc định.
c) Module mimetypes: Module này được sử dụng để xác định loại file (MIME type) dựa trên phần mở rộng, giúp hiển thị thông tin loại file trong bảng Treeview (ví dụ: text/plain cho .txt, image/jpeg cho .jpg).
d) Xử lý ngoại lệ: Ứng dụng áp dụng kỹ thuật xử lý ngoại lệ để phát hiện và xử lý các lỗi như thư mục không tồn tại, thiếu quyền truy cập, hoặc file không thể mở, đảm bảo chương trình hoạt động ổn định.
e) Thiết kế giao diện người dùng: Kiến thức về bố cục giao diện (layout) được áp dụng để sắp xếp các thành phần như nút, bảng, và nhãn một cách hợp lý, đảm bảo giao diện thân thiện và dễ sử dụng.

Những kiến thức này không chỉ giúp hoàn thành các yêu cầu của đề tài mà còn cung cấp cơ hội để sinh viên thực hành các kỹ năng lập trình Python trong một dự án thực tế.

## *Tóm tắt chương*

*Chương này đã giới thiệu tổng quan về đề tài "Trình quản lý thư mục GUI", bao gồm mô tả các tính năng chính như chọn thư mục, liệt kê file .txt, .py, .jpg, mở file, và xử lý lỗi. Các thách thức được nêu rõ, bao gồm tích hợp GUI với hệ thống file, lọc file theo phần mở rộng, xử lý lỗi, thiết kế giao diện trực quan, và cân nhắc tính tương thích. Các kiến thức vận dụng như thư viện tkinter, ttk, module os, mimetypes, và xử lý ngoại lệ đã được trình bày, làm nền tảng cho các chương tiếp theo về lý thuyết, thiết kế, và thực nghiệm của ứng dụng.*

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

Chương này trình bày các kiến thức lý thuyết và công cụ lập trình được sử dụng để phát triển ứng dụng "Trình quản lý thư mục GUI". Các nội dung bao gồm thư viện giao diện đồ họa `tkinter` và `ttk`, module `os` để xử lý hệ thống file, module `mimetypes` để xác định loại file, và kỹ thuật xử lý ngoại lệ để đảm bảo tính ổn định của chương trình. Những kiến thức này là nền tảng để thực hiện các tính năng của ứng dụng, bao gồm chọn thư mục, liệt kê file theo loại (.txt, .py, .jpg), mở file bằng chương trình mặc định, và xử lý lỗi.

## 2.1. Thư viện tkinter và ttk

### 2.1.1. Tổng quan về tkinter

`tkinter` là thư viện chuẩn của Python được thiết kế để xây dựng các ứng dụng giao diện đồ họa (GUI). Thư viện này cung cấp một bộ công cụ mạnh mẽ để tạo các thành phần giao diện như cửa sổ, nút, nhãn, bảng, và hộp thoại, cho phép người dùng tương tác với chương trình một cách trực quan. Trong đề tài này, `tkinter` được sử dụng để xây dựng giao diện chính của ứng dụng, bao gồm cửa sổ hiển thị, khung chứa các thành phần giao diện, và hộp thoại chọn thư mục.

Các thành phần chính của `tkinter` được sử dụng trong ứng dụng bao gồm:

a) Tk: Lớp chính để tạo cửa sổ ứng dụng, đóng vai trò là nền tảng cho toàn bộ giao diện. Cửa sổ này được cấu hình với tiêu đề ("Trình Quản Lý Thư Mục") và kích thước phù hợp (700x500 pixel).

b) Frame: Một widget chứa các thành phần giao diện khác, giúp tổ chức bố cục giao diện theo cách hợp lý và dễ quản lý.

c) Label: Được sử dụng để hiển thị văn bản tĩnh, chẳng hạn như đường dẫn của thư mục hiện tại, giúp người dùng biết họ đang làm việc với thư mục nào.

d) filedialog: Module con của `tkinter`, cung cấp hộp thoại chọn thư mục (`askdirectory`) để người dùng chọn thư mục cần quản lý một cách dễ dàng.

`tkinter` được chọn vì tính đơn giản, tích hợp sẵn trong Python, và khả năng tạo giao diện nhanh chóng mà không cần cài đặt thêm thư viện bên ngoài.

### 2.1.2. Module ttk

Module `ttk` (themed Tkinter) là một phần mở rộng của `tkinter`, cung cấp các widget giao diện hiện đại hơn, có khả năng tự động điều chỉnh theo giao diện của hệ điều hành (Windows, macOS, Linux). Trong ứng dụng này, `ttk` được sử dụng để tạo các thành phần giao diện với vẻ ngoài chuyên nghiệp và nhất quán. Các widget `ttk` được sử dụng bao gồm:

a) Treeview: Một widget hiển thị danh sách file dưới dạng bảng với ba cột: Tên File, Loại File, và Đường Dẫn. Treeview hỗ trợ hiển thị dữ liệu theo hàng và cột, cho phép người dùng dễ dàng xem thông tin chi tiết của từng file.
b) Button: Được sử dụng để tạo các nút "Chọn Thư Mục" và "Mở File", cho phép người dùng thực hiện các hành động tương ứng khi nhấn.
c) Scrollbar: Thanh cuộn dọc được gắn vào Treeview, hỗ trợ người dùng cuộn qua danh sách file khi số lượng file vượt quá kích thước hiển thị của bảng.

Việc sử dụng `ttk` giúp giao diện ứng dụng trở nên trực quan, dễ sử dụng, và phù hợp với trải nghiệm người dùng hiện đại.

## 2.2. Module os

Module `os` là một module chuẩn của Python, cung cấp các hàm để tương tác với hệ thống file của hệ điều hành. Trong đề tài này, `os` đóng vai trò quan trọng trong việc quét thư mục, kiểm tra file, và mở file bằng chương trình mặc định. Các chức năng chính của module `os` được sử dụng bao gồm:

a) os.listdir(path): Hàm này trả về danh sách tất cả các file và thư mục con trong một thư mục được chỉ định. Trong ứng dụng, hàm được dùng để quét toàn bộ nội dung của thư mục mà người dùng chọn, cung cấp danh sách các file để xử lý tiếp.

b) os.path.isfile(path): Hàm này kiểm tra xem một đường dẫn có trỏ đến một file hay không, giúp lọc bỏ các thư mục con khi hiển thị danh sách file trong Treeview.
c) os.path.splitext(filename): Hàm này tách tên file thành hai phần: tên và phần mở rộng (ví dụ: "document.txt" được tách thành "document" và ".txt"). Hàm được sử dụng để kiểm tra và lọc các file có phần mở rộng .txt, .py, hoặc .jpg.
d) os.startfile(path): Hàm này mở một file bằng chương trình mặc định của hệ điều hành Windows (ví dụ: file .txt mở bằng Notepad, file .jpg mở bằng trình xem ảnh). Đây là chức năng cốt lõi để thực hiện yêu cầu mở file của đề tài.

Module `os` được chọn vì tính linh hoạt và khả năng tương tác trực tiếp với hệ thống file, đáp ứng các yêu cầu về quét và xử lý file của ứng dụng.

## 2.3. Module mimetypes

Module `mimetypes` là một module chuẩn của Python, được sử dụng để xác định loại file (MIME type) dựa trên phần mở rộng của file. Trong ứng dụng, module này được dùng để lấy thông tin loại file và hiển thị trong cột "Loại File" của Treeview, giúp người dùng dễ dàng nhận biết đặc tính của file (ví dụ: văn bản, mã nguồn, hay hình ảnh).

Chức năng chính của `mimetypes` được sử dụng:

- mimetypes.guess_type(filename): Hàm này trả về một bộ gồm loại MIME và mã hóa (nếu có) của file. Ví dụ, file "note.txt" trả về ("text/plain", None), file "image.jpg" trả về ("image/jpeg", None). Nếu không xác định được loại file, hàm trả về None, và ứng dụng sẽ hiển thị "Không xác định" trong cột Loại File.

Module `mimetypes` giúp cung cấp thông tin bổ sung về file, tăng tính trực quan và chuyên nghiệp của giao diện ứng dụng.

## 2.4. Xử lý ngoại lệ

Xử lý ngoại lệ là một kỹ thuật quan trọng để đảm bảo ứng dụng hoạt động ổn định và cung cấp phản hồi thân thiện khi xảy ra lỗi. Trong đề tài này, các trường hợp lỗi được xử lý bao gồm:

a) FileNotFoundError: Xảy ra khi thư mục hoặc file được chọn không tồn tại. Ứng dụng sẽ hiển thị thông báo lỗi như "Thư mục không tồn tại!" hoặc "File không tồn tại!" để thông báo cho người dùng.
b) PermissionError: Xảy ra khi người dùng không có quyền truy cập vào thư mục hoặc file. Ứng dụng sẽ thông báo "Không có quyền truy cập thư mục!" để người dùng thử chọn thư mục khác.
c) OSError: Xảy ra khi không thể mở file bằng chương trình mặc định, ví dụ do file bị hỏng hoặc không được hỗ trợ. Thông báo lỗi như "Không thể mở file!" sẽ được hiển thị.

Kỹ thuật xử lý ngoại lệ được áp dụng trong các giai đoạn như quét thư mục, kiểm tra file, và mở file, đảm bảo ứng dụng không bị treo khi gặp lỗi và cung cấp thông tin rõ ràng để người dùng xử lý vấn đề.

## 2.5. Thiết kế giao diện người dùng

Thiết kế giao diện người dùng (UI) là một phần quan trọng của đề tài, đảm bảo ứng dụng dễ sử dụng và trực quan. Các nguyên tắc thiết kế UI được áp dụng bao gồm:

a) Bố cục hợp lý: Các thành phần giao diện như nút, bảng, và nhãn được sắp xếp trong một khung (Frame) với khoảng cách phù hợp, sử dụng hệ thống lưới (grid layout) để đảm bảo tính đồng nhất.
b) Tính trực quan: Bảng Treeview hiển thị thông tin file theo cột (Tên File, Loại File, Đường Dẫn) với kích thước cột được tối ưu hóa để dễ đọc. Thanh cuộn dọc hỗ trợ khi danh sách file dài.

c) Phản hồi người dùng: Thông báo lỗi được hiển thị qua hộp thoại khi xảy ra sự cố, và nhãn hiển thị đường dẫn thư mục giúp người dùng biết trạng thái hiện tại của ứng dụng.
d) Tương tác: Các nút "Chọn Thư Mục" và "Mở File" được đặt ở vị trí dễ thấy, và sự kiện nhấp đúp chuột trên Treeview được hỗ trợ để mở file nhanh chóng.

Thiết kế UI được xây dựng dựa trên các widget của `tkinter` và `ttk`, đảm bảo giao diện thân thiện và đáp ứng yêu cầu của đề tài.

## *Tóm tắt chương*

*Chương này đã trình bày các kiến thức lý thuyết cần thiết để phát triển ứng dụng "Trình quản lý thư mục GUI". Thư viện `tkinter` và `ttk` cung cấp các công cụ để xây dựng giao diện đồ họa trực quan, với các widget như Treeview, Button, và Scrollbar. Module `os` hỗ trợ quét và xử lý file, trong khi module `mimetypes` giúp xác định loại file. Kỹ thuật xử lý ngoại lệ đảm bảo tính ổn định, và thiết kế giao diện người dùng tạo ra trải nghiệm thân thiện. Những kiến thức này là nền tảng cho việc thiết kế và triển khai ứng dụng, được trình bày chi tiết trong các chương tiếp theo.*

# CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

## 3.1. Sơ đồ khối hệ thống

Trước khi bắt tay vào xây dựng một ứng dụng, việc thiết kế hệ thống là một bước quan trọng nhằm đảm bảo rằng chương trình có cấu trúc rõ ràng, dễ bảo trì và dễ mở rộng. Trong đề tài "Trình quản lý thư mục GUI", hệ thống được thiết kế dựa trên mô hình module hóa, tức là chia chương trình thành các phần (module) nhỏ hơn, mỗi phần đảm nhiệm một chức năng cụ thể, hoạt động độc lập nhưng có liên kết logic với nhau.

Cấu trúc hệ thống bao gồm các thành phần chính như sau:

a) **Giao diện người dùng (GUI - Graphical User Interface)**: Đây là phần tiếp xúc trực tiếp với người dùng, nơi họ thực hiện các thao tác như chọn thư mục, quan sát danh sách file, hoặc mở file bằng cách nhấp chuột. Giao diện được xây dựng bằng thư viện tkinter và ttk, cung cấp các widget hiện đại như cửa sổ, nút bấm, nhãn hiển thị, bảng dữ liệu (Treeview) và thanh cuộn. Việc sử dụng giao diện đồ họa giúp người dùng không cần sử dụng dòng lệnh, qua đó tăng trải nghiệm và tính trực quan.

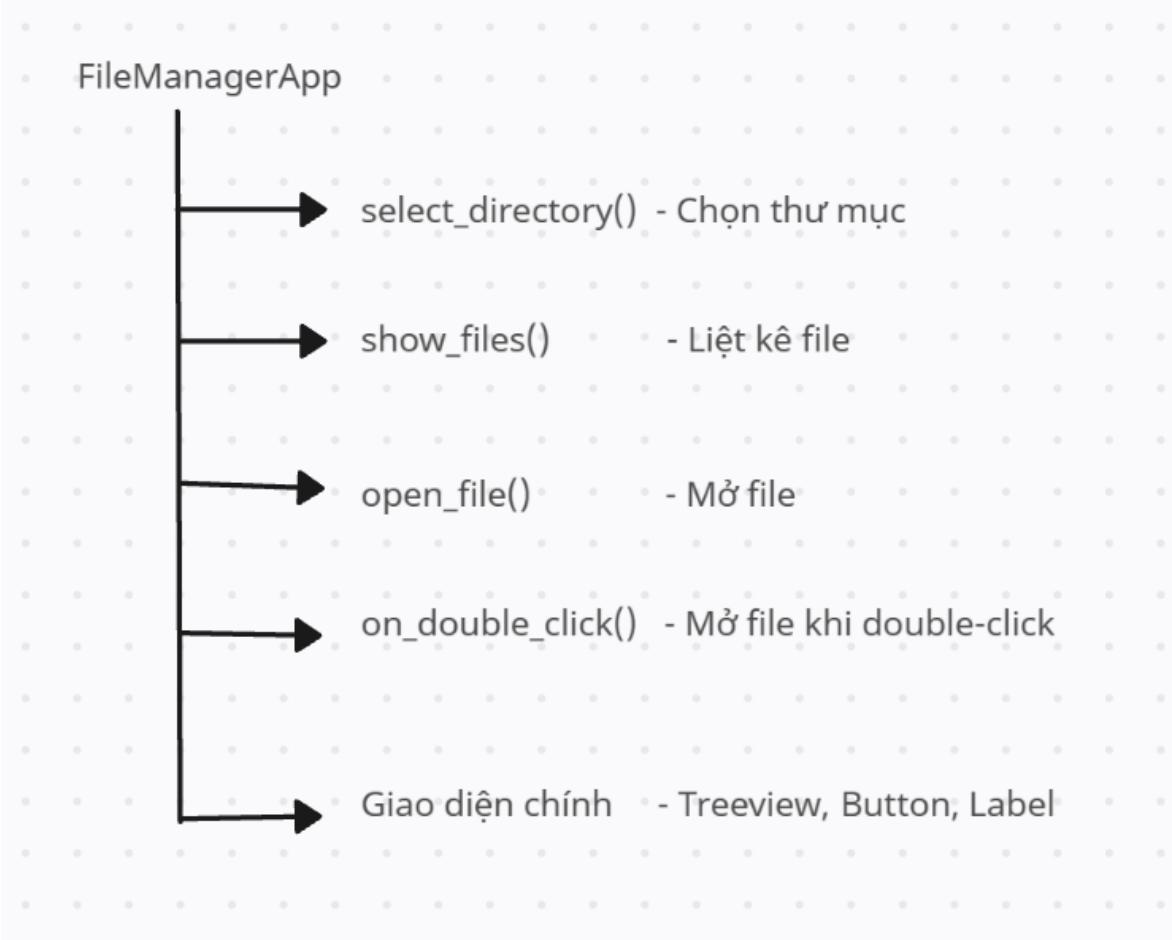
b) **Xử lý chọn thư mục**: Khi người dùng nhấn vào nút "Chọn Thư Mục", chương trình sẽ kích hoạt một hộp thoại hệ thống (filedialog) để người dùng duyệt và chọn một thư mục có sẵn trên máy tính. Sau khi chọn xong, hệ thống sẽ lưu đường dẫn và tiến hành quét thư mục đó để hiển thị file.

c) **Lọc và liệt kê file**: Sau khi có được đường dẫn thư mục, chương trình sẽ sử dụng thư viện os để duyệt toàn bộ nội dung bên trong. Chương trình chỉ giữ lại các file có định dạng .txt, .py và .jpg vì đây là yêu cầu đã được đặt ra từ đầu đề. Các file không đúng định dạng sẽ bị loại bỏ. Sau đó, danh sách các file hợp lệ sẽ được hiển thị lên giao diện Treeview kèm theo tên file, loại file (dựa trên MIME type) và đường dẫn đầy đủ.

d) **Xử lý mở file**: Khi người dùng nhấn nút "Mở File" hoặc nhấp đúp vào một file trong bảng, chương trình sẽ sử dụng os.startfile() để yêu cầu hệ điều hành mở file đó bằng phần mềm mặc định đã thiết lập. Ví dụ, file .txt có thể mở bằng Notepad, .jpg mở bằng ứng dụng xem ảnh.

e) **Xử lý lỗi và phản hồi người dùng**: Trong quá trình thao tác với hệ thống file, rất nhiều lỗi có thể phát sinh như: thư mục không tồn tại, người dùng không có quyền truy cập, hoặc file đã bị xóa. Vì vậy, chương trình được thiết kế để bắt các ngoại lệ này và hiển thị thông báo lỗi rõ ràng dưới dạng hộp thoại (messagebox). Điều này giúp người dùng hiểu được sự cố đang xảy ra, thay vì làm ứng dụng bị "treo" hoặc dừng đột ngột.

**Biểu đồ phân cấp chức năng:**




*Hình 3.1 Biểu đồ phân cấp chức năng*

## 3.2. Sơ đồ khối các thuật toán chính

Để chương trình thực hiện được các chức năng đã đề ra, bên trong nó cần có các thuật toán điều khiển luồng xử lý dữ liệu. Các thuật toán này không quá phức tạp, nhưng phải được tổ chức hợp lý để vừa đảm bảo tính đúng đắn, vừa đảm bảo hiệu suất và độ ổn định.

### a) Chọn thư mục

Khi người dùng nhấn nút "Chọn Thư Mục", chương trình sẽ gọi hộp thoại duyệt thư mục. Hệ thống chờ người dùng chọn và nhấn OK. Nếu thư mục được chọn là hợp lệ, đường dẫn sẽ được lưu vào biến current_directory, và một nhãn trên giao diện sẽ hiển thị đường dẫn này. Sau đó, chương trình gọi hàm show_files() để bắt đầu quá trình liệt kê file.


*Hình 3.2 Sơ đồ thuật toán chọn thư mục*

### b) Liệt kê file

Đây là thuật toán trọng tâm của chương trình. Sau khi nhận được đường dẫn thư mục:

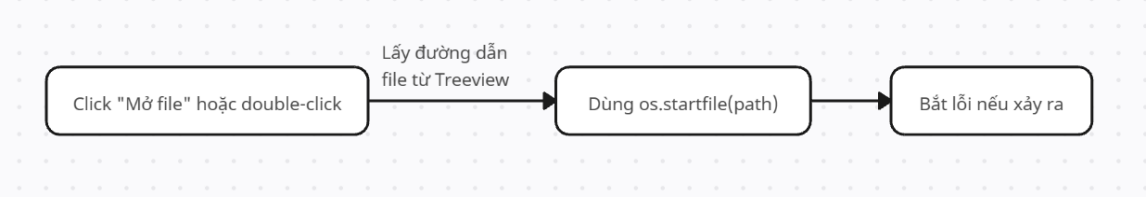
- Chương trình dùng os.listdir() để lấy danh sách tất cả các phần tử trong thư mục.
- Duyệt từng phần tử, dùng os.path.isfile() để kiểm tra xem đó có phải là file không.
- Nếu đúng là file, chương trình tách phần mở rộng bằng os.path.splitext() và kiểm tra xem có nằm trong danh sách phần mở rộng cho phép không.
- Nếu hợp lệ, chương trình dùng mimetypes.guess_type() để xác định loại MIME của file, sau đó thêm thông tin vào bảng Treeview.


*Hình 3.3 Sơ đồ thuật toán liệt kê file*

c) **Mở file**

Thuật toán này thực hiện khi người dùng chọn một file trong bảng và nhấn nút "Mở File", hoặc khi người dùng double-click vào dòng tương ứng. Hệ thống lấy đường dẫn file từ dòng được chọn, rồi gọi os.startfile() để mở. Trong quá trình này, có thể phát sinh lỗi (ví dụ file đã bị xóa hoặc không còn tồn tại), nên cần đặt trong khối try-except.

Tất cả các thuật toán đều được kiểm soát tốt bằng các câu lệnh điều kiện và khối xử lý lỗi, đảm bảo chương trình vận hành ổn định và thân thiện với người dùng.




*Hình 3.4 Sơ đồ thuật toán mơ file*

## 3.3. Cấu trúc dữ liệu

Chương trình sử dụng cấu trúc dữ liệu đơn giản và trực quan. Dữ liệu không được lưu vĩnh viễn mà được xử lý và hiển thị tạm thời trong thời gian người dùng thao tác.

Các biến và cấu trúc dữ liệu chính:

a) **self.current_directory**: Biến chuỗi chứa đường dẫn thư mục hiện tại mà người dùng đã chọn. Đây là biến trung tâm giúp các hàm khác biết cần quét thư mục nào.

b) **self.allowed_extensions**: Là một tuple bất biến gồm ba phần tử: ('.txt', '.py', '.jpg'). Đây là danh sách phần mở rộng mà chương trình chấp nhận hiển thị. Những file có đuôi khác sẽ bị loại bỏ để đảm bảo tập trung đúng vào yêu cầu đề tài.

c) **Treeview**: Đây là thành phần quan trọng nhất trong giao diện, đồng thời là nơi chứa dữ liệu file dưới dạng bảng ba cột:

- **Tên File**: chỉ tên của file
- **Loại File**: xác định theo kiểu MIME như “text/plain” hay “image/jpeg”
- **Đường Dẫn**: là đường dẫn đầy đủ, dùng để mở file khi cần

Việc sử dụng Treeview không chỉ giúp hiển thị dữ liệu gọn gàng mà còn hỗ trợ thao tác chọn file, mở rộng, cuộn dọc rất tiện lợi. Mỗi lần người dùng chọn thư mục mới, bảng Treeview sẽ được làm sạch và cập nhật lại toàn bộ.

## 3.4. Chương trình

Toàn bộ chương trình được đóng gói trong một class duy nhất có tên là FileManagerApp. Đây là một cách tổ chức theo mô hình hướng đối tượng (OOP), giúp dễ quản lý và mở rộng trong tương lai.

**Cấu trúc chương trình gồm các thành phần chính:**

a) **Hàm __init__()**:

- Khởi tạo cửa sổ chính của ứng dụng với kích thước xác định.
- Tạo tất cả các thành phần giao diện: frame, nhãn, nút, bảng, thanh cuộn.
- Thiết lập cấu trúc lưới (grid layout) để đảm bảo các thành phần sắp xếp khoa học.
- Gán các sự kiện xử lý cho nút và thao tác người dùng.

b) **Hàm select_directory()**:

- Mở hộp thoại chọn thư mục.
- Nếu có kết quả → cập nhật biến đường dẫn và gọi hàm liệt kê file.
- Nếu không → hiển thị thông báo "Chưa chọn thư mục".

c) **Hàm show_files()**:

- Duyệt thư mục, lọc file theo định dạng phù hợp.
- Lấy thông tin MIME, rồi đưa các dòng vào Treeview.

d) **Hàm open_file()**:

- Kiểm tra người dùng đã chọn dòng nào.
- Lấy đường dẫn tương ứng và mở bằng phần mềm mặc định.
- Nếu gặp lỗi → hiển thị hộp thoại thông báo.

e) **Hàm on_double_click()**:

- Kích hoạt khi người dùng nhấn đúp chuột vào dòng file.
- Gọi lại open_file() để thực hiện thao tác mở nhanh.

Toàn bộ chương trình chạy trong một cửa sổ chính (mainloop) và dừng lại khi người dùng thoát ứng dụng.

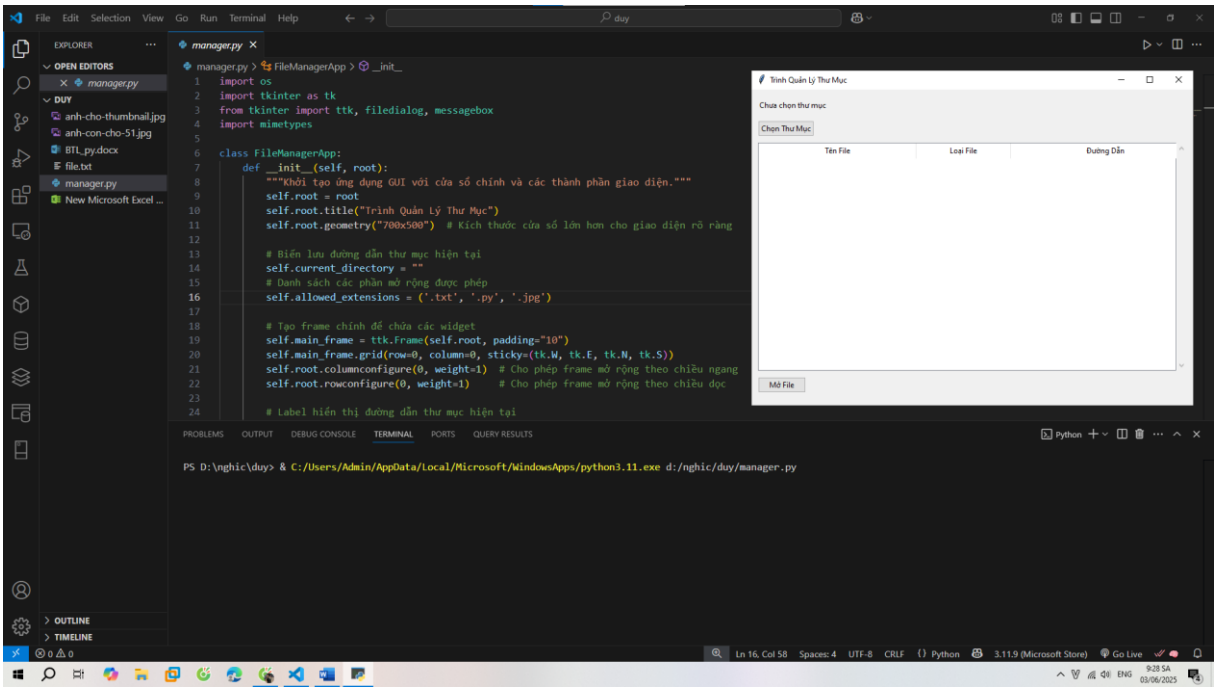
## *Tóm tắt chương*

*Trong chương này, chúng ta đã trình bày toàn bộ quá trình thiết kế và xây dựng chương trình "Trình quản lý thư mục GUI". Từ việc xây dựng sơ đồ khối hệ thống, xác định rõ ràng chức năng từng module, thiết kế các thuật toán chính cho đến tổ chức cấu trúc dữ liệu và chương trình theo hướng đối tượng. Mỗi phần trong chương trình đều được gói gọn và xử lý tốt, đảm bảo độ ổn định, thân thiện và dễ sử dụng. Bằng cách sử dụng các thư viện chuẩn của Python, đề tài không chỉ hoàn thành mục tiêu được giao mà còn có tiềm năng mở rộng thành một ứng dụng quản lý file hoàn chỉnh hơn trong tương lai.*

# CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN

## 4.1. Thực nghiệm

Sau khi hoàn thiện quá trình xây dựng và lập trình, ứng dụng "Trình quản lý thư mục GUI" đã được triển khai chạy thử để đánh giá mức độ đáp ứng yêu cầu đề bài, cũng như kiểm tra tính ổn định, hiệu quả và khả năng phản hồi trong các tình huống khác nhau. Quá trình thực nghiệm được thực hiện dựa trên các hành vi người dùng phổ biến, mô phỏng thao tác thật sự trong môi trường sử dụng thực tế.



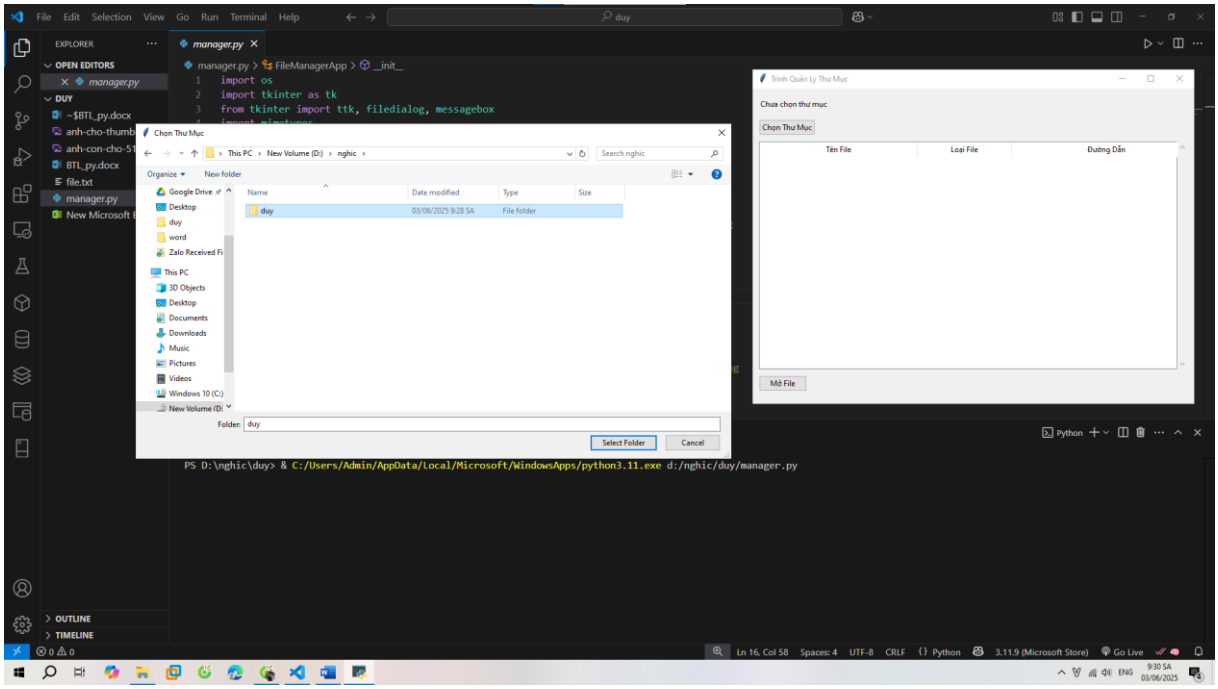

*Hình 4.1 chạy chương trình*

### a) Kiểm thử chức năng chọn thư mục

Trong thử nghiệm đầu tiên, người dùng nhấn vào nút "Chọn Thư Mục" để mở hộp thoại chọn thư mục hệ thống. Hộp thoại xuất hiện nhanh chóng, cho phép người dùng duyệt đến bất kỳ vị trí nào trên ổ đĩa. Sau khi lựa chọn một thư mục cụ thể và xác nhận, giao diện chính sẽ lập tức cập nhật đường dẫn thư mục hiện tại. Việc này được phản ánh trực quan qua một dòng nhãn phía trên giao diện, giúp người dùng luôn biết mình đang thao tác với thư mục nào.

Đây là bước khởi đầu trong luồng thao tác và được đánh giá là trơn tru, thân thiện. Dù người dùng chọn một thư mục trống hoặc thư mục có hàng chục file,

chương trình đều phản hồi nhanh và không có hiện tượng giật, lag hay treo ứng dụng.



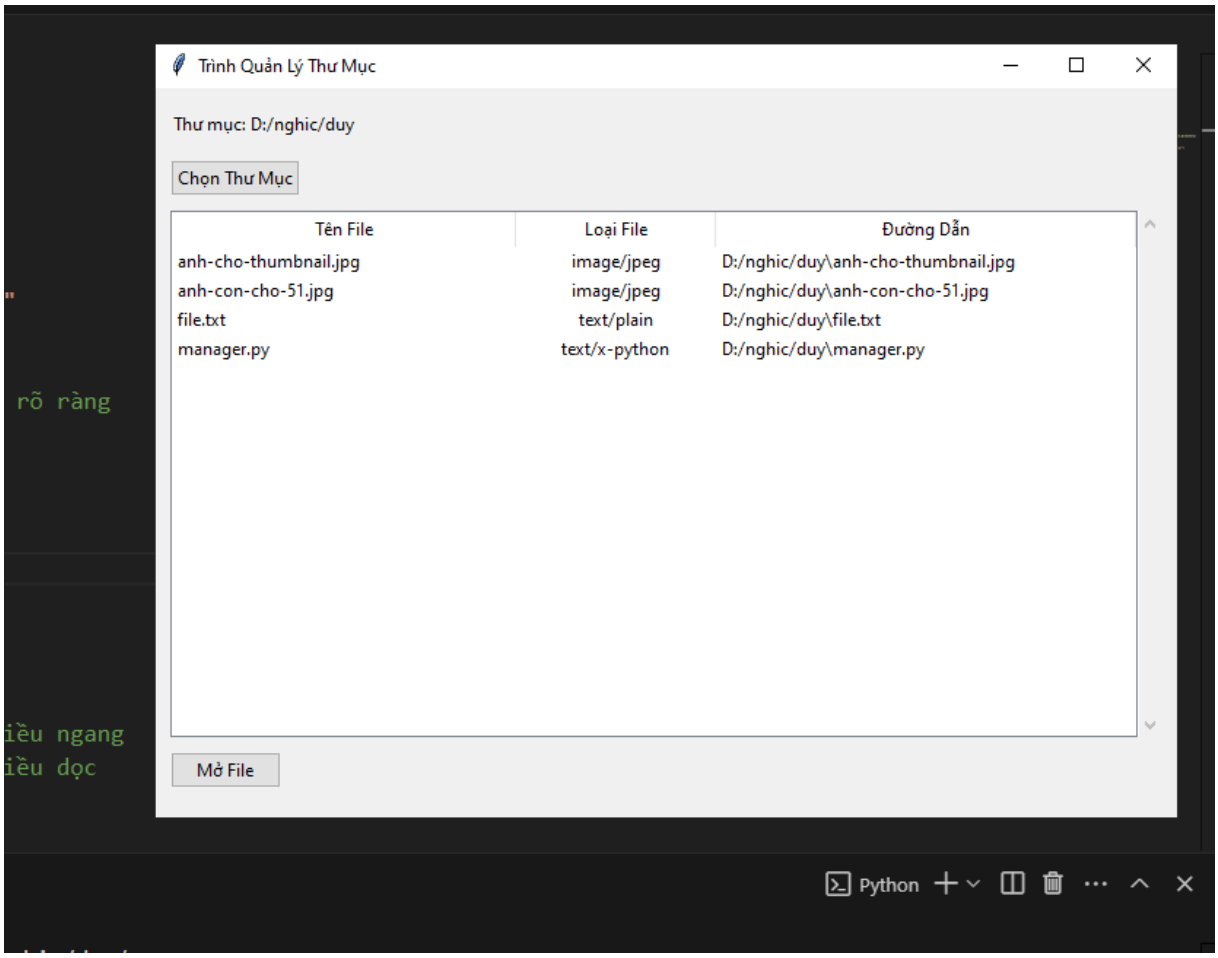
*Hình 4.2 Kiểm thử chức năng chọn thư mục*

**b) Kiểm thử chức năng liệt kê và lọc file**

Sau khi thư mục được chọn, chương trình tự động quét toàn bộ nội dung trong thư mục đó, lọc ra các file có phần mở rộng là .txt, .py, và .jpg. Danh sách các file hợp lệ được hiển thị dưới dạng bảng gồm ba cột: **Tên File**, **Loại File**, và **Đường Dẫn** đầy đủ. Giao diện bảng được tổ chức rõ ràng, dễ quan sát, có thể cuộn dọc nếu danh sách dài.

Một điểm nổi bật là chương trình không phân biệt chữ hoa - chữ thường khi lọc file (ví dụ .JPG cũng được hiển thị như .jpg), điều này thể hiện sự tinh tế trong xử lý dữ liệu đầu vào. Mỗi dòng được thêm vào bảng đều là kết quả của một quá trình kiểm tra kỹ càng: từ kiểm tra có phải là file hay không, kiểm tra phần mở rộng, đến xác định loại MIME thông qua mimetypes.

Từ góc độ người dùng, thao tác này mang lại sự hài lòng cao vì mọi thông tin đều được hiển thị một cách ngắn gọn nhưng đầy đủ. Người dùng không cần mở thư mục bằng Windows Explorer để xem nội dung, tất cả đã được tổng hợp và sắp xếp khoa học.




*Hình 4.3 Kiểm thử chức năng liệt kê và lọc file*

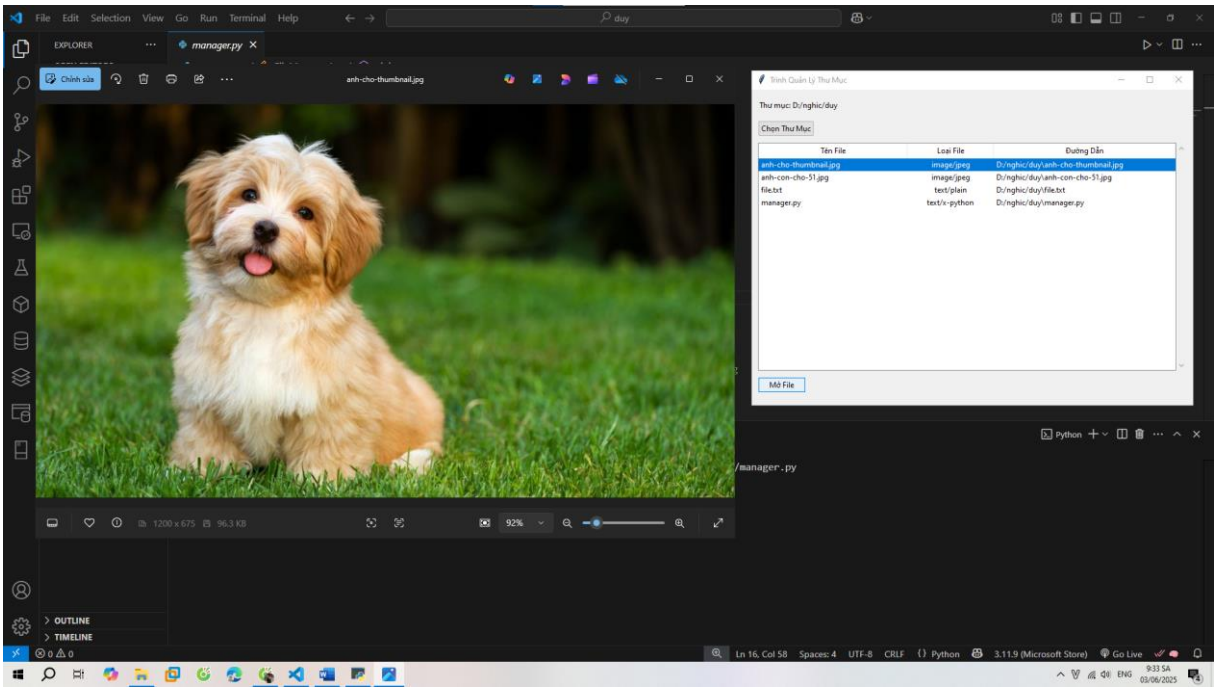
**c) Kiểm thử chức năng mở file**

Một trong những chức năng quan trọng và thực tế nhất của chương trình là cho phép người dùng mở file ngay từ giao diện. Khi người dùng chọn một file trong bảng và nhấn vào nút "Mở File", hoặc đơn giản là nhấp đúp chuột vào dòng đó, chương trình sẽ gửi yêu cầu đến hệ điều hành để mở file bằng phần mềm mặc định.

Chức năng này hoạt động rất ổn định. File .txt được mở bằng Notepad, .jpg được mở bằng ứng dụng xem ảnh mặc định, còn .py được mở bằng trình soạn thảo mã nguồn. Việc tương tác này tạo cảm giác liền mạch, giúp chương trình không chỉ dừng lại ở việc "liệt kê", mà còn trở thành một **cổng điều hướng tệp tin** thông minh và hiệu quả.

Quan trọng hơn, nếu file đã bị xóa, hoặc người dùng không có quyền mở file đó, chương trình sẽ không bị dừng đột ngột. Thay vào đó, hộp thoại cảnh báo

sẽ hiển thị thông báo rõ ràng như: “File không tồn tại!” hoặc “Không thể mở file!”. Cách xử lý này giúp người dùng không cảm thấy khó hiểu hoặc bị động khi gặp sự cố.



*Hình 3.4 Kiểm thử chức năng mở file*

**d) Đánh giá tổng thể giao diện và trải nghiệm người dùng**

Giao diện chương trình tuy được xây dựng bằng thư viện cơ bản của Python (tkinter) nhưng vẫn thể hiện được tính thẩm mỹ nhất định: gọn gàng, dễ nhìn, dễ sử dụng. Các thành phần như nhãn, nút bấm, bảng dữ liệu và thanh cuộn đều được bố trí hợp lý, với khoảng cách phù hợp giúp người dùng không bị rối mắt.

Không có hiện tượng đè layout, bảng không bị tràn ra ngoài, tất cả các dòng dữ liệu đều nằm trong vùng hiển thị quy định. Nhờ đó, thao tác với ứng dụng mang lại cảm giác thoải mái và chuyên nghiệp, nhất là khi so sánh với các phần mềm dòng lệnh truyền thống.

## 4.2. Kết luận

Sau quá trình nghiên cứu, xây dựng và kiểm thử, chương trình “Trình quản lý thư mục GUI” đã hoàn thành đầy đủ các yêu cầu đặt ra trong đề tài. Đây không chỉ là một bài tập lập trình, mà còn là một bước đệm quan trọng giúp sinh viên tiếp cận gần hơn với việc xây dựng ứng dụng thực tế.

### a) Những kết quả đã đạt được

Chương trình đã thực hiện thành công các chức năng sau:

- Giao diện trực quan, hiện đại, dễ sử dụng, không cần nhập lệnh thủ công.
- Khả năng chọn thư mục linh hoạt, hỗ trợ mọi ổ đĩa, thư mục bất kỳ trên hệ thống.
- Tự động quét và lọc các file có định dạng cụ thể (.txt, .py, .jpg), hiển thị rõ ràng.
- Mở file bằng chương trình mặc định của hệ điều hành, hỗ trợ thao tác nhanh chóng.
- Xử lý lỗi hợp lý, không để chương trình bị treo hay crash, đảm bảo độ tin cậy.

Những kết quả này không chỉ cho thấy khả năng hoàn thành bài toán kỹ thuật mà còn phản ánh sự đầu tư nghiêm túc trong việc tổ chức mã nguồn, xử lý ngoại lệ và cải thiện trải nghiệm người dùng.

### b) Những bài học kinh nghiệm

Thông qua đề tài này, người viết đã rút ra nhiều bài học quan trọng, tiêu biểu như:

- **Tầm quan trọng của thiết kế trước khi viết mã**: Khi cấu trúc chương trình rõ ràng, quá trình lập trình diễn ra suôn sẻ hơn rất nhiều.
- **Xử lý ngoại lệ là yếu tố không thể thiếu**: Trong mọi chương trình có tương tác với hệ thống, khả năng xảy ra lỗi là không tránh khỏi. Việc xử lý lỗi tốt giúp chương trình "chuyên nghiệp" hơn.
- **Giao diện trực quan tạo ra sự khác biệt**: Một chương trình với logic tốt nhưng giao diện kém sẽ khiến người dùng khó tiếp cận. Ngược lại, một giao diện thân thiện giúp tăng sự hài lòng và hiệu quả sử dụng.

### c) Hướng phát triển tương lai

Mặc dù chương trình đã đáp ứng tốt các yêu cầu cơ bản, vẫn còn nhiều hướng mở để cải tiến và mở rộng:

- **Hỗ trợ nhiều định dạng hơn**: Như .pdf, .docx, .xlsx, .png…

- **Bổ sung chức năng xóa, đổi tên, di chuyển file**: Giúp chương trình trở thành một công cụ quản lý file hoàn chỉnh.
- **Tích hợp tìm kiếm nhanh**: Cho phép lọc theo tên file hoặc loại file.
- **Tính năng xem trước nội dung file**: Đặc biệt hữu ích với file văn bản hoặc hình ảnh.
- **Đa nền tảng**: Thêm cơ chế kiểm tra hệ điều hành để hoạt động tốt trên macOS, Linux, không chỉ riêng Windows.

## *Tóm tắt chương*

*Chương này đã trình bày quá trình kiểm thử ứng dụng "Trình quản lý thư mục GUI" một cách toàn diện, từ chọn thư mục, liệt kê file, mở file cho đến xử lý lỗi. Thực nghiệm cho thấy chương trình đáp ứng tốt các yêu cầu, hoạt động ổn định và tạo được trải nghiệm người dùng mượt mà. Từ đó, người thực hiện rút ra được nhiều bài học thực tiễn trong việc tổ chức phần mềm, xử lý dữ liệu, cũng như thiết kế giao diện. Đây là nền tảng vững chắc để tiếp tục học tập và phát triển các ứng dụng lớn hơn trong tương lai.*

# TÀI LIỆU THAM KHẢO


1. Python Software Foundation. *The Python Standard Library*. Truy cập từ: https://docs.python.org/3/library/index.html
2. Python Software Foundation. *tkinter — Python interface to Tcl/Tk*. Truy cập từ: https://docs.python.org/3/library/tk.html
3. Python Software Foundation. *os — Miscellaneous operating system interfaces*. Truy cập từ: https://docs.python.org/3/library/os.html
4. Python Software Foundation. *mimetypes — Mapping of filenames to MIME types*. Truy cập từ: https://docs.python.org/3/library/mimetypes.html
5. Grayson, John E. *Python and Tkinter Programming*. Manning Publications, 2000.
6. Sweigart, Al. *Automate the Boring Stuff with Python*. No Starch Press, 2015. Truy cập từ: https://automatetheboringstuff.com
7. Lutz, Mark. *Programming Python* (4th Edition). O'Reilly Media, 2010.
8. TutorialsPoint. *Python GUI Programming (Tkinter)*. Truy cập từ: https://www.tutorialspoint.com/python/python_gui_programming.htm
9. GeeksforGeeks. *Python | Working with tkinter widgets*. Truy cập từ: https://www.geeksforgeeks.org/python-gui-tkinter/
10. Real Python. *Build a File Explorer With Python and Tkinter*. Truy cập từ: https://realpython.com/python-gui-tkinter/