**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙡🏶🙣----

A blue and white logo with a book and a graduation cap

Description automatically generated

**ĐỒ ÁN**

**HỌC PHẦN LẬP TRÌNH PYTHON**

**TÊN ĐỀ TÀI**

**ỨNG DỤNG QUẢN LÍ VÀ TÌM KIẾM PHIM ANIME**

**Giáo viên hướng dẫn: Ths. Đinh Nguyễn Trọng Nghĩa**

**Thành viên nhóm :**

Hoàng Văn Hoan-2001230258 (Nhóm trưởng)

Nguyễn Quốc Huy-2001230301

**Thành phố Hồ Chí Minh, tháng 3 năm 2025**

# 

# **LỜI MỞ DẦU**

Lời đầu tiên, nhóm chúng em xin chân thành cảm ơn thầy Đinh Nguyễn Trọng Nghĩa Giảng viên khoa Công Nghệ Thông Tin đã tận tình giảng dạy và hướng dẫn sinh viên trong suốt thời gian môn học.Nhờ vào những lời khuyên và chỉ bảo tận tình của cô, nhóm chúng em đã vượt qua những khó khăn khi thực hiện đồ án của nhóm.

Tiếp đến, chúng em xin gửi lời tri ân tới các thầy cô trường Đại học Công Thương Thành Phố Hồ Chí Minh - Những người đã cùng góp sức truyền đạt kiến thức để giúp chúng em có được nền tảng tốt như ngày hôm nay. Ngoài ra, không thể không nhắc tới gia đình, bạn bè người thân đã là hậu phương vững chắc, là chỗ dựa tinh thần của chúng em trong thời gian qua. Sự thành công của đồ án không thể không kể đến công ơn của mọi người.

Nhưng sau tất cả, chúng em nhận thức rằng với lượng kiến thức và kinh nghiệm ít ỏi của bản thân, chắc chắn đồ án sẽ khó tránh khỏi thiểu sót. Kính mong quý thầy cô thông cảm và góp ý để em ngày càng hoàn thiện hơn.

Chúng em xin chân thành cảm ơn !

# **BẢNG PHÂN CÔNG NHIỆM VỤ CỦA TỪNG THÀNH VIÊN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và tên** | **Nhiệm vụ** | **Đánh giá hoàn thành** | **Phần trăm hoàn thành** |
| Nguyễn Quốc Huy | Lên ý tưởng, thực hiện phần tiểu luận | Hoàn thành nhiệm vụ | 100% |
| Hoàng Văn Hoan | Thực hiện phần ứng dụng | Hoàn thành nhiệm vụ | 100% |

Mục lục

[**LỜI MỞ DẦU** 2](#_Toc199004251)

[**BẢNG PHÂN CÔNG NHIỆM VỤ CỦA TỪNG THÀNH VIÊN** 3](#_Toc199004252)

[I. Giới thiệu 5](#_Toc199004253)

[1. Tổng quan về đề tài 5](#_Toc199004254)

[2. Mục tiêu đề tài 5](#_Toc199004255)

[3. Phạm vi và đối tượng 5](#_Toc199004256)

[II. Công nghệ và công cụ 6](#_Toc199004257)

[1. Ngôn ngữ lập trình 6](#_Toc199004258)

[2. Thư viện sử dụng 7](#_Toc199004259)

[3. Các API sử dụng 7](#_Toc199004260)

[III. Phân tích yêu cầu 8](#_Toc199004261)

[1. Chức năng của hệ thống 8](#_Toc199004262)

[2. Yêu cầu kỹ thuật 10](#_Toc199004263)

[IV. Thiết kế hệ thống 11](#_Toc199004264)

[1. Kiến trúc hệ thống 11](#_Toc199004265)

[2. Thiết kế giao diện 12](#_Toc199004266)

[3. Cấu trúc dữ liệu 13](#_Toc199004267)

[V. Cài đặt và triển khai 14](#_Toc199004268)

[1. Quá trình phát triển 14](#_Toc199004269)

[2. Mô tả mã nguồn 16](#_Toc199004270)

[3. Hướng dẫn sử dụng ứng dụng 17](#_Toc199004271)

[VI. Kiểm thử và đánh giá 19](#_Toc199004272)

[1. Quá trình kiểm thử 19](#_Toc199004273)

[2. Kết quả kiểm thử 20](#_Toc199004274)

[VII. Kết luận. 22](#_Toc199004275)

[1. Tóm tắt kết quả 22](#_Toc199004276)

[2. Hướng phát triển 23](#_Toc199004277)

[**TÀI LIỆU THAM KHẢO** 24](#_Toc199004278)

[**PHỤ LỤC** 24](#_Toc199004279)

# Giới thiệu

## Tổng quan về đề tài

Trong thời đại công nghệ thông tin phát triển mạnh mẽ, nhu cầu giải trí của con người ngày càng tăng cao, đặc biệt là với văn hóa anime – một loại hình nghệ thuật độc đáo đến từ Nhật Bản. Anime không chỉ là một hình thức giải trí mà còn là một phần của văn hóa đại chúng, thu hút hàng triệu người hâm mộ trên toàn thế giới. Tuy nhiên, việc quản lý và tìm kiếm thông tin về các bộ anime yêu thích vẫn còn gặp nhiều khó khăn, đặc biệt khi người dùng phải truy cập nhiều nền tảng khác nhau để tìm kiếm và lưu trữ thông tin.

Xuất phát từ thực tế đó, nhóm chúng em đã quyết định phát triển một ứng dụng quản lý anime sử dụng ngôn ngữ lập trình Python và giao diện đồ họa Tkinter. Ứng dụng này không chỉ giúp người dùng quản lý danh sách anime yêu thích mà còn cung cấp các tính năng tìm kiếm, sắp xếp và hiển thị thông tin anime một cách trực quan và tiện lợi. Dự án được thực hiện nhằm đáp ứng nhu cầu của người dùng yêu thích anime, đồng thời là cơ hội để nhóm chúng tôi áp dụng kiến thức lập trình vào thực tiễn.

## Mục tiêu đề tài

Mục tiêu chính của dự án là xây dựng một ứng dụng desktop cho phép người dùng tìm kiếm thông tin anime từ API công cộng và ứng dụng đã đạt được các mục tiêu chính sau:

* Xây dựng một ứng dụng desktop có giao diện đồ họa người dùng (GUI) thân thiện để quản lý thông tin anime.
* Cho phép người dùng đăng ký, đăng nhập để cá nhân hóa trải nghiệm.
* Cung cấp chức năng tìm kiếm anime từ một nguồn dữ liệu trực tuyến (API).
* Hiển thị thông tin chi tiết của anime bao gồm tiêu đề, hình ảnh, điểm số, số tập.
* Cho phép người dùng quản lý danh sách anime yêu thích cá nhân.
* Cung cấp các tùy chọn sắp xếp và lọc danh sách anime.
* Đảm bảo ứng dụng hoạt động ổn định và phản hồi nhanh chóng.

## Phạm vi và đối tượng

Dự án tập trung vào việc phát triển một ứng dụng desktop nhằm hỗ trợ người dùng quản lý thông tin về anime một cách hiệu quả và tiện lợi. Cụ thể, phạm vi của ứng dụng bao gồm các khía cạnh sau:

* Chức năng chính: Ứng dụng cung cấp các chức năng cơ bản như đăng nhập và đăng ký tài khoản người dùng, hiển thị danh sách anime, tìm kiếm anime theo từ khóa, sắp xếp danh sách anime theo các tiêu chí (điểm số, số tập, tiêu đề), và quản lý danh sách anime yêu thích. Ngoài ra, ứng dụng cũng hỗ trợ hiển thị thông tin chi tiết của anime như tiêu đề, điểm số, số tập, trạng thái phát hành và hình ảnh bìa.
* Nguồn dữ liệu: Với nguồn dữ liệu thì sẽ được lấy trực tiếp từ Jikan API – một API miễn phí cung cấp thông tin về anime từ cơ sở dữ liệu MyAnimeList. Ứng dụng sử dụng API này để lấy danh sách top 20 anime nổi bật nhất và hỗ trợ tìm kiếm anime theo từ khóa do người dùng nhập. Dữ liệu sau khi lấy từ API được lưu trữ cục bộ trong tệp du\_lieu\_anime.json để giảm thiểu việc gọi API liên tục, từ đó cải thiện hiệu suất.
* Quản lí người dùng: Ứng dụng hỗ trợ quản lý tài khoản người dùng thông qua tệp nguoi\_dung.json, các chức năng như đăng nhập, đăng ký, thêm/xóa anime yêu thích đều được triển khai để đảm bảo trải nghiệm cá nhân hóa cho từng người dùng.

Ứng dụng hướng đến các nhóm đối tượng cụ thể, bao gồm:

* Người yêu thích anime: Đây là đối tượng chính của ứng dụng, bao gồm những người đam mê anime và muốn quản lý danh sách anime yêu thích một cách hiệu quả. Họ có thể là những người xem anime thường xuyên, muốn theo dõi các bộ anime nổi bật, hoặc tìm kiếm anime mới dựa trên từ khóa.
* Người dùng phổ thông: Ứng dụng được thiết kế đơn giản và trực quan, phù hợp với cả những người không có nhiều kinh nghiệm về công nghệ. Các thao tác như đăng nhập, tìm kiếm, sắp xếp và quản lý danh sách yêu thích đều dễ dàng thực hiện, không đòi hỏi kỹ năng lập trình hay kiến thức chuyên môn.
* Cộng đồng người hâm mộ anime: Ứng dụng có thể phục vụ các nhóm cộng đồng yêu thích anime, đặc biệt là những người muốn chia sẻ danh sách anime yêu thích hoặc tìm kiếm các bộ anime phù hợp với sở thích cá nhân.

# Công nghệ và công cụ

## Ngôn ngữ lập trình

Ứng dụng được phát triển bằng Python, một ngôn ngữ lập trình cấp cao, mã nguồn mở, được biết đến với cú pháp đơn giản, dễ đọc và khả năng ứng dụng rộng rãi trong nhiều lĩnh vực. Python được chọn vì sự linh hoạt và hiệu quả trong việc phát triển ứng dụng giao diện đồ họa (GUI), xử lý dữ liệu, cũng như tích hợp API bên ngoài. Phiên bản Python 3.x được sử dụng để đảm bảo tính tương thích với các thư viện hiện đại như Tkinter, Pillow và Requests, đồng thời tận dụng các cải tiến mới nhất về hiệu suất và bảo mật. Ngoài ra, Python hỗ trợ lập trình hướng đối tượng, giúp mã nguồn của ứng dụng được tổ chức rõ ràng thông qua lớp QuanLyAnime, từ đó dễ dàng bảo trì và mở rộng. Cộng đồng Python rộng lớn cùng với tài liệu phong phú cũng là một lợi thế, giúp nhóm phát triển nhanh chóng giải quyết các vấn đề kỹ thuật trong quá trình xây dựng ứng dụng, từ việc thiết kế giao diện, quản lý dữ liệu cục bộ, đến gọi API và xử lý hình ảnh. Nhờ Python, ứng dụng không chỉ đảm bảo được tính năng mà còn đạt hiệu suất ổn định trên các hệ điều hành phổ biến như Windows, macOS và Linux.

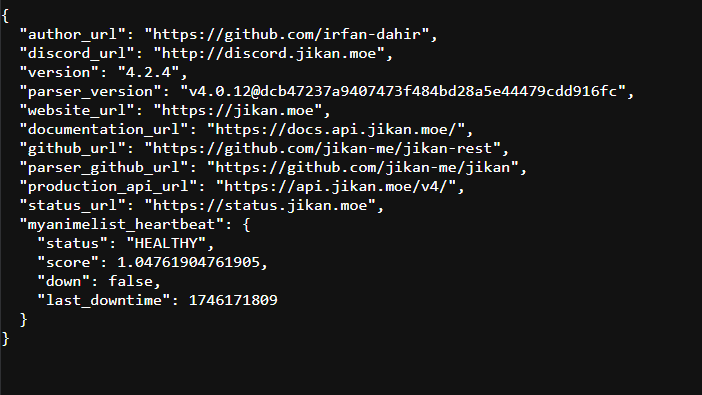
## Thư viện sử dụng

Để xây dựng giao diện đồ họa, nhóm đã sử dụng thư viện:

* Tkinter: Thư viện chuẩn của Python để xây dựng giao diện đồ họa (GUI). Tkinter được sử dụng để thiết kế các màn hình đăng nhập, đăng ký và giao diện chính của ứng dụng.
* Pillow (PIL): Thư viện xử lý hình ảnh, dùng để tải và hiển thị ảnh nền cũng như hình ảnh bìa của các bộ anime.
* Requests: Thư viện hỗ trợ gửi yêu cầu HTTP, được sử dụng để gọi API Jikan và lấy dữ liệu anime.
* JSON: Thư viện chuẩn của Python để xử lý dữ liệu dạng JSON, dùng để lưu trữ thông tin người dùng và anime.
* Threading: Hỗ trợ xử lý đa luồng, giúp tải hình ảnh và dữ liệu từ API mà không làm gián đoạn giao diện người dùng.
* Functools (lru\_cache): Tối ưu hóa việc tải hình ảnh bằng cách lưu trữ bộ nhớ đệm, giảm thời gian tải lại dữ liệu trùng lặp.

## Các API sử dụng

Ứng dụng tích hợp API từ Jikan, một API công cộng cung cấp thông tin chi tiết về anime dựa trên dữ liệu từ MyAnimeList. API này hỗ trợ các endpoint như /top/anime để lấy danh sách anime nổi bật và /anime với tham số q để tìm kiếm anime theo từ khóa. Dữ liệu trả về từ API được định dạng JSON, bao gồm các thông tin như tiêu đề, số tập, điểm đánh giá, trạng thái phát hành, và URL hình ảnh của anime.



(Ảnh minh họa api của Jikan)

# Phân tích yêu cầu

## Chức năng của hệ thống

Hệ thống được thiết kế để hỗ trợ người dùng quản lý thông tin anime một cách hiệu quả, với các chức năng chính được phân tích và liên kết với các hàm cụ thể trong mã nguồn như sau:

* Quản lý người dùng:

Hệ thống cho phép người dùng đăng nhập, đăng ký tài khoản và quản lý thông tin cá nhân, bao gồm danh sách anime yêu thích. Thông tin người dùng được lưu trữ trong tệp nguoi\_dung.json các hàm thực thi:

* khoiTaoTepDuLieu: Khởi tạo các tệp dữ liệu nguoi\_dung.json và du\_lieu\_anime.json nếu chưa tồn tại, thiết lập người dùng mặc định (admin và người dùng thông thường).
* dangNhap(tenNguoiDung, matKhau): Xử lý đăng nhập bằng cách kiểm tra tên người dùng và mật khẩu từ tệp nguoi\_dung.json, sau đó chuyển sang giao diện chính nếu thông tin hợp lệ.
* hienThiManHinhDangKy: Hiển thị giao diện đăng ký và xử lý logic đăng ký thông qua hàm nội bộ dangKy, kiểm tra thông tin người dùng, lưu tài khoản mới vào tệp JSON và thông báo kết quả.
* Hiển thị danh sách anime:

Hệ thống lấy dữ liệu anime từ Jikan API và hiển thị dưới dạng lưới với các thông tin như tiêu đề, điểm số, số tập, trạng thái và hình ảnh bìa. Người dùng có thể xem danh sách top anime hoặc kết quả tìm kiếm.

* layTopAnime: Gọi API Jikan để lấy danh sách 20 anime hàng đầu, xử lý dữ liệu (tiêu đề, điểm số, số tập, trạng thái, URL hình ảnh) và lưu vào du\_lieu\_anime.json.
* hienThiLuoiAnime(danhSachAnime): Hiển thị danh sách anime dưới dạng lưới, tạo các khung giao diện cho từng anime với thông tin chi tiết và hình ảnh bìa.
* layHinhAnh(url): Tải hình ảnh bìa từ URL, sử dụng bộ nhớ đệm để tối ưu hiệu suất, và thông qua taiHinhAnhKhongDongBo và capNhatHinhAnh..
* Tìm kiếm anime:

Người dùng có thể tìm kiếm anime theo từ khóa, hệ thống sẽ gọi API Jikan để lấy kết quả và hiển thị danh sách tương ứng.

* timKiemAnimeApi: Xử lý tìm kiếm bằng cách lấy từ khóa từ ô nhập, gọi API Jikan với tham số truy vấn, xử lý dữ liệu trả về và hiển thị kết quả thông qua hienThiLuoiAnime.
* nhiemVuNen(nhiemVu, \*doiSo): Hỗ trợ chạy các tác vụ (như tìm kiếm) trong luồng riêng để tránh treo giao diện, kết hợp nhiemVuHoanThanh và nhiemVuThatBai để xử lý kết quả hoặc lỗi. hãy chỉnh lại khoảng cách các từ
* Sắp xếp anime:

Hệ thống hỗ trợ sắp xếp danh sách anime theo tiêu chí như điểm số, số tập hoặc tiêu đề, giúp người dùng dễ dàng tổ chức danh sách theo nhu cầu.

* sapXepAnime: Sắp xếp danh sách anime hiện tại (danhSachHienThiHienTai) dựa trên tiêu chí được chọn (điểm số, số tập, tiêu đề), sau đó gọi hienThiLuoiAnime để cập nhật giao diện.
* Quản lý danh sách yêu thích:

Người dùng có thể thêm hoặc xóa anime khỏi danh sách yêu thích, xem danh sách yêu thích riêng biệt và lưu trữ thông tin này trong tệp nguoi\_dung.json.

* batTatYeuThich(anime): Thêm hoặc xóa tiêu đề anime khỏi danh sách yêu thích của người dùng, cập nhật tệp nguoi\_dung.json và làm mới giao diện.
* hienThiYeuThich: Lọc và hiển thị danh sách anime yêu thích của người dùng từ danhSachHienThiHienTai, sử dụng hienThiLuoiAnime để cập nhật giao diện.
* chuyenSangCheDoTatCa: Chuyển đổi chế độ hiển thị về danh sách tất cả anime, bỏ lọc yêu thích.
* Giao diện thân thiện:

Hệ thống cung cấp giao diện trực quan với các màn hình đăng nhập, đăng ký và giao diện chính, sử dụng ảnh nền và bố cục rõ ràng để tăng trải nghiệm người dùng.

* hienThiManHinhDangNhap: Tạo giao diện đăng nhập với ảnh nền (a2.png, a1.png) và các ô nhập thông tin.
* hienThiManHinhChinh: Thiết kế giao diện chính với các khung tìm kiếm, sắp xếp, chế độ xem (tất cả/yêu thích) và hiển thị danh sách anime.
* hienThiManHinhTai và anManHinhTai: Hiển thị và ẩn màn hình tải khi thực hiện các tác vụ như gọi API hoặc tải hình ảnh.

Các chức năng trên được triển khai chặt chẽ, đảm bảo tính đồng bộ và trải nghiệm mượt mà cho người dùng. Mỗi chức năng đều được hỗ trợ bởi các hàm cụ thể trong mã nguồn, giúp mã hóa logic rõ ràng và dễ bảo trì.

## Yêu cầu kỹ thuật

Ứng dụng được thiết kế để đảm bảo hoạt động ổn định và hiệu quả trên nhiều nền tảng, với các yêu cầu kỹ thuật cụ thể nhằm đáp ứng nhu cầu của người dùng và duy trì trải nghiệm mượt mà.

* Về hệ điều hành, ứng dụng tương thích với các hệ điều hành phổ biến hỗ trợ Python, bao gồm Windows (phiên bản 10 trở lên), macOS (phiên bản 10.15 trở lên), và các bản phân phối Linux như Ubuntu 20.04 hoặc mới hơn, đảm bảo tính linh hoạt cho người dùng trên nhiều thiết bị khác nhau.
* Về yêu cầu phần mềm, ứng dụng yêu cầu cài đặt Python phiên bản 3.8 hoặc cao hơn để tận dụng các tính năng mới và đảm bảo tính tương thích với các thư viện được sử dụng. Các thư viện cần thiết bao gồm Tkinter (được cài đặt sẵn với Python để xây dựng giao diện), Pillow (phiên bản 9.0 trở lên để xử lý hình ảnh), và Requests (phiên bản 2.28 trở lên để gọi API). Người dùng cần cài đặt các thư viện này thông qua lệnh pip install pillow requests trước khi chạy ứng dụng. Về yêu cầu phần cứng, ứng dụng không đòi hỏi cấu hình máy tính cao, nhưng cần tối thiểu 4GB RAM và 500MB dung lượng ổ cứng trống để lưu trữ tệp dữ liệu (nguoi\_dung.json, du\_lieu\_anime.json) và các tệp ảnh nền (a1.png, a2.png). Màn hình nên có độ phân giải tối thiểu 1280x720 để đảm bảo giao diện hiển thị đầy đủ và không bị cắt xén.
* Về vấn đề kết nối mạng: ứng dụng cần kết nối internet ổn định (tối thiểu 5Mbps) để gọi Jikan API và tải hình ảnh bìa của anime, đặc biệt trong các tác vụ như lấy danh sách top anime hoặc tìm kiếm anime theo từ khóa. Nếu không có kết nối mạng, ứng dụng chỉ có thể hiển thị dữ liệu đã lưu trữ cục bộ, dẫn đến hạn chế về tính năng.
* Về hiệu suất, ứng dụng được tối ưu để xử lý dữ liệu nhanh chóng, với thời gian phản hồi trung bình dưới 2 giây cho mỗi yêu cầu API, nhờ vào cơ chế lưu trữ bộ nhớ đệm (lru\_cache) cho hình ảnh và dữ liệu API. Các tác vụ nặng như tải hình ảnh hoặc gọi API được thực thi trong luồng riêng (sử dụng threading) để tránh treo giao diện, đảm bảo trải nghiệm người dùng không bị gián đoạn.
* Lưu trữ dữ liệu được thực hiện thông qua tệp JSON, không yêu cầu cơ sở dữ liệu phức tạp, nhưng cần đảm bảo quyền đọc/ghi trên thư mục chứa ứng dụng để lưu trữ thông tin người dùng và anime. Cuối cùng, ứng dụng cần được kiểm tra trên các độ phân giải màn hình khác nhau để đảm bảo giao diện không bị lỗi hiển thị, và cần xử lý các trường hợp ngoại lệ (như mất kết nối mạng, tệp ảnh không tồn tại) để tăng tính ổn định. Những yêu cầu kỹ thuật này giúp ứng dụng hoạt động hiệu quả, đáp ứng nhu cầu người dùng và dễ dàng triển khai trên nhiều thiết bị.

# Thiết kế hệ thống

## Kiến trúc hệ thống

Kiến trúc hệ thống được thiết kế theo mô hình đơn giản, tập trung vào tính phân tầng rõ ràng để đảm bảo dễ phát triển, bảo trì và mở rộng trong tương lai. Hệ thống bao gồm các thành phần chính sau:

* Lớp giao diện người dùng (GUI): Được xây dựng bằng thư viện Tkinter, chịu trách nhiệm hiển thị các màn hình như đăng nhập, đăng ký và giao diện chính. Lớp này bao gồm các hàm như hienThiManHinhDangNhap, hienThiManHinhDangKy, và hienThiManHinhChinh, đảm bảo giao diện trực quan và dễ sử dụng. Ví dụ, hienThiManHinhChinh tạo các khung tìm kiếm, sắp xếp và hiển thị danh sách anime, sử dụng các widget như ttk.Frame, ttk.Entry, và tk.Canvas để hỗ trợ cuộn nội dung.
* Lớp xử lý logic nghiệp vụ: Được triển khai trong lớp QuanLyAnime, chứa các hàm xử lý các tác vụ chính như đăng nhập (dangNhap), tìm kiếm anime (timKiemAnimeApi), và sắp xếp danh sách (sapXepAnime). Lớp này hoạt động như cầu nối giữa giao diện và dữ liệu, xử lý các yêu cầu từ người dùng và trả về kết quả. Ví dụ, hàm timKiemAnimeApi gọi API Jikan, xử lý dữ liệu trả về và cập nhật giao diện thông qua hienThiLuoiAnime. Để tránh treo giao diện, các tác vụ nặng (như gọi API) được chạy trong luồng riêng nhờ hàm nhiemVuNen, kết hợp với nhiemVuHoanThanh và nhiemVuThatBai để xử lý kết quả hoặc lỗi.
* Lớp lưu trữ dữ liệu: Sử dụng tệp JSON để lưu trữ thông tin cục bộ, bao gồm nguoi\_dung.json (lưu thông tin người dùng) và du\_lieu\_anime.json (lưu dữ liệu anime từ API). Lớp này không sử dụng cơ sở dữ liệu phức tạp, giúp giảm độ phức tạp trong triển khai. Hàm khoiTaoTepDuLieu khởi tạo các tệp JSON với dữ liệu mặc định, trong khi các hàm như batTatYeuThich và layTopAnime đọc/ghi dữ liệu vào các tệp này.
* Tích hợp API: Ứng dụng sử dụng Jikan API để lấy dữ liệu anime trực tuyến, thông qua thư viện requests. Các hàm layTopAnime và timKiemAnimeApi gửi yêu cầu HTTP đến API, nhận dữ liệu JSON và xử lý thành định dạng phù hợp để hiển thị. Dữ liệu từ API được lưu vào bộ nhớ đệm (boNhoDemApi) để giảm số lần gọi API, cải thiện hiệu suất.

Kiến trúc này đảm bảo tính phân tách rõ ràng giữa giao diện, logic và dữ liệu, giúp dễ dàng bảo trì và mở rộng, chẳng hạn như thêm hỗ trợ cơ sở dữ liệu hoặc tích hợp API khác trong tương lai.

## Thiết kế giao diện

Thiết kế giao diện của ứng dụng được xây dựng nhằm mang lại trải nghiệm trực quan, thân thiện và dễ sử dụng, với các màn hình chính được tối ưu cho người dùng yêu thích anime.

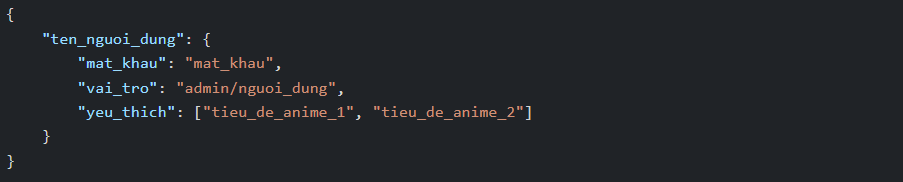
* Màn hình đăng nhập: Được triển khai trong hàm hienThiManHinhDangNhap, màn hình này bao gồm một ảnh nền toàn màn hình (a2.png, kích thước 1920x1200) và một khung đăng nhập chính giữa với ảnh nền phụ (a1.png, kích thước 300x200). Khung đăng nhập chứa hai ô nhập (ttk.Entry) cho tên người dùng và mật khẩu (mật khẩu hiển thị dưới dạng \*), cùng hai nút ttk.Button là "Đăng nhập" và "Đăng ký". Nút "Đăng nhập" gọi hàm dangNhap để xác thực, trong khi nút "Đăng ký" chuyển sang màn hình đăng ký. Giao diện sử dụng bố cục lưới (grid) để căn chỉnh các thành phần, đảm bảo hiển thị cân đối.
* Màn hình đăng ký: Được triển khai trong hienThiManHinhDangKy, tương tự màn hình đăng nhập với ảnh nền a2.png (kích thước 1570x900) và khung đăng ký (a1.png, kích thước 300x250). Màn hình này có ba ô nhập cho tên người dùng, mật khẩu, xác nhận mật khẩu, và hai nút "Đăng ký" và "Quay lại đăng nhập". Hàm nội bộ dangKy xử lý logic đăng ký, kiểm tra mật khẩu khớp và lưu thông tin vào nguoi\_dung.json.
* Màn hình chính: Được xây dựng trong hienThiManHinhChinh, giao diện chính gồm nhiều khung:
* Khung tìm kiếm: Một ô nhập (ttk.Entry) để nhập từ khóa và nút "Tìm kiếm" gọi timKiemAnimeApi. Ô nhập được gán sự kiện <Return> để tìm kiếm khi nhấn Enter.
* Khung sắp xếp: Một ttk.OptionMenu với các lựa chọn "Điểm số", "Số tập", "Tiêu đề", gọi sapXepAnime khi thay đổi lựa chọn.
* Khung chế độ xem: Hai nút radio (ttk.Radiobutton) để chuyển đổi giữa "Tất cả Anime" (chuyenSangCheDoTatCa) và "Yêu thích" (hienThiYeuThich).
* Khung nội dung: Sử dụng tk.Canvas và ttk.Scrollbar để hiển thị danh sách anime dạng lưới (3 cột), mỗi anime được hiển thị trong một khung (ttk.Frame) với hình ảnh bìa, tiêu đề, điểm số, số tập và nút yêu thích (tk.Button với biểu tượng trái tim). Hàm hienThiLuoiAnime quản lý việc hiển thị này.
* Khung nút điều khiển: Gồm nút "Làm mới Top Anime" (gọi layTopAnime) và "Đăng xuất" (quay lại màn hình đăng nhập).

Giao diện được tùy chỉnh bằng ttk.Style để định dạng khung, nhãn, và phông chữ, đồng thời hỗ trợ cuộn danh sách (yscrollcommand) và điều chỉnh hiển thị trên các hệ điều hành khác nhau (Windows, Linux) thông qua các sự kiện cuộn chuột (MouseWheel, Button-4, Button-5).

## Cấu trúc dữ liệu

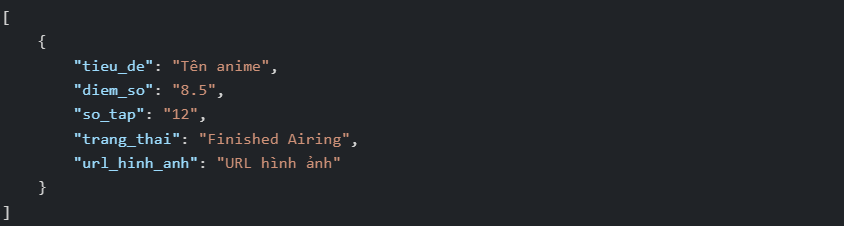
Cấu trúc dữ liệu của ứng dụng được thiết kế đơn giản, sử dụng định dạng JSON để lưu trữ thông tin, đảm bảo dễ đọc, dễ xử lý và không phụ thuộc vào cơ sở dữ liệu phức tạp. Dữ liệu được chia thành hai loại chính:

* Tệp nguoi\_dung.json: Lưu trữ thông tin người dùng dưới dạng từ điển với cấu trúc:



Mỗi người dùng được đại diện bởi một khóa (tên người dùng), với các trường mat\_khau (mật khẩu), vai\_tro (quyền hạn: admin hoặc người dùng thông thường), và yeu\_thich (danh sách tiêu đề anime yêu thích). Hàm khoiTaoTepDuLieu khởi tạo tệp này với dữ liệu mặc định (tài khoản "admin" và "nguoi\_dung"), và các hàm như dangNhap, batTatYeuThich đọc/ghi dữ liệu để quản lý tài khoản và danh sách yêu thích.

* Tệp du\_lieu\_anime.json: Lưu trữ danh sách anime lấy từ API với cấu trúc:



Mỗi anime là một đối tượng với các trường: tieu\_de (tiêu đề), diem\_so (điểm số), so\_tap (số tập), trang\_thai (trạng thái phát hành), và url\_hinh\_anh (liên kết hình ảnh bìa). Hàm layTopAnime và timKiemAnimeApi tạo dữ liệu này từ API và lưu vào tệp, trong khi hienThiLuoiAnime đọc dữ liệu để hiển thị.

Ngoài ra, ứng dụng sử dụng các biến bộ nhớ đệm như boNhoDemApi (lưu trữ dữ liệu API tạm thời) và boNhoDemHinhAnh (lưu trữ hình ảnh đã tải), cùng với cơ chế lru\_cache trong layHinhAnh để tối ưu hiệu suất. Cấu trúc dữ liệu này đơn giản nhưng hiệu quả, phù hợp với quy mô ứng dụng và dễ dàng nâng cấp lên cơ sở dữ liệu phức tạp hơn (như SQLite) nếu cần.

# Cài đặt và triển khai

## Quá trình phát triển

Quá trình phát triển ứng dụng được thực hiện trong vòng 4 tuần, chia thành các giai đoạn rõ ràng để đảm bảo tiến độ và chất lượng. Mỗi giai đoạn đều được lên kế hoạch chi tiết với sự phối hợp chặt chẽ giữa các thành viên trong nhóm.

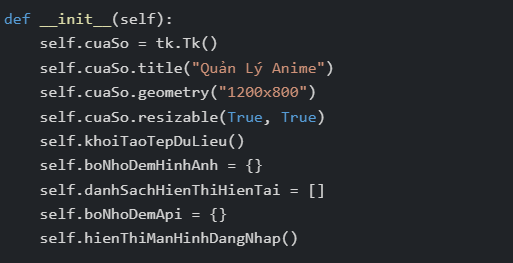
* Phân tích và thiết kế (Ngày 1-4): Giai đoạn này tập trung vào việc phân tích yêu cầu và thiết kế hệ thống. Nhóm đã họp để xác định các chức năng chính như đăng nhập, tìm kiếm, và quản lý danh sách yêu thích, đồng thời phác thảo giao diện người dùng bằng cách sử dụng các công cụ như Figma để tạo bản mẫu giao diện (wireframe) cho các màn hình đăng nhập, đăng ký và giao diện chính. Các yêu cầu kỹ thuật như sử dụng Jikan API, lưu trữ dữ liệu bằng JSON, và yêu cầu phần cứng tối thiểu cũng được thảo luận và ghi nhận. Phân công nhiệm vụ được thực hiện ở giai đoạn này, với mỗi thành viên đảm nhận một phần cụ thể: một người phân tích yêu cầu, một người thiết kế giao diện, và hai người chuẩn bị môi trường phát triển (cài đặt Python, các thư viện như Tkinter, Pillow, Requests).
* Phát triển chức năng cơ bản (Ngày 5-10): Giai đoạn này tập trung vào việc xây dựng các chức năng cốt lõi như quản lý người dùng. Các hàm như khoiTaoTepDuLieu, dangNhap, và hienThiManHinhDangKy được viết để xử lý đăng nhập và đăng ký tài khoản, đồng thời lưu trữ thông tin người dùng vào nguoi\_dung.json. Giao diện đăng nhập và đăng ký được triển khai với ảnh nền (a1.png, a2.png) và các widget Tkinter như ttk.Entry, ttk.Button. Nhóm đã sử dụng Git để quản lý mã nguồn, tạo các nhánh riêng (branch) cho từng chức năng và thực hiện kiểm tra tích hợp (merge) hàng ngày để đảm bảo không có xung đột mã. Các lỗi ban đầu, như không tải được ảnh nền do đường dẫn sai, được phát hiện và sửa chữa trong giai đoạn này.
* Tích hợp API và phát triển giao diện chính (Ngày 11-15): Nhóm tập trung vào tích hợp Jikan API và xây dựng giao diện chính. Hàm layTopAnime và timKiemAnimeApi được viết để lấy dữ liệu từ API, xử lý JSON và lưu vào du\_lieu\_anime.json. Để tránh treo giao diện, các tác vụ gọi API được chạy trong luồng riêng thông qua nhiemVuNen, kết hợp với nhiemVuHoanThanh và nhiemVuThatBai để xử lý kết quả hoặc lỗi. Giao diện chính (hienThiManHinhChinh) được phát triển với các khung tìm kiếm, sắp xếp, và hiển thị danh sách anime dạng lưới (hienThiLuoiAnime). Nhóm đã tối ưu hiệu suất bằng cách sử dụng lru\_cache trong layHinhAnh để lưu trữ hình ảnh đã tải, giảm thời gian tải lại. Các vấn đề như lỗi kết nối API hoặc hình ảnh không tải được đã được xử lý bằng cách hiển thị thông báo lỗi qua messagebox.
* Hoàn thiện, kiểm thử và viết tài liệu (Ngày 16-19): Giai đoạn cuối tập trung vào việc hoàn thiện ứng dụng, kiểm thử và viết tài liệu. Nhóm đã thêm các tính năng như sắp xếp anime (sapXepAnime) và quản lý danh sách yêu thích (batTatYeuThich, hienThiYeuThich). Kiểm thử được thực hiện trên nhiều hệ điều hành (Windows, Linux) để đảm bảo tính tương thích, tập trung vào các trường hợp như mất kết nối mạng, nhập liệu không hợp lệ, và lỗi tải hình ảnh. Các lỗi nhỏ, như giao diện bị cắt xén ở độ phân giải thấp, đã được sửa bằng cách điều chỉnh bố cục. Tài liệu hướng dẫn sử dụng và báo cáo dự án cũng được hoàn thiện trong giai đoạn này, bao gồm các phần như phân tích yêu cầu, thiết kế hệ thống, và kết quả kiểm thử.

Quá trình phát triển được thực hiện với phương pháp Agile, chia nhỏ nhiệm vụ và kiểm tra liên tục, đảm bảo ứng dụng hoàn thiện đúng thời hạn và đáp ứng yêu cầu đề ra.

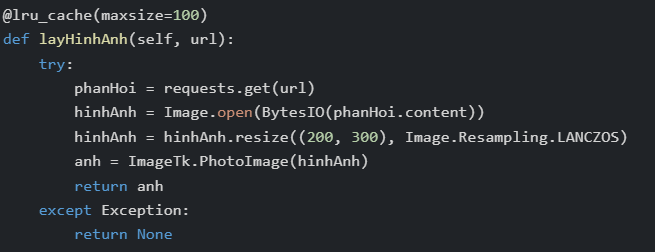
## Mô tả mã nguồn

Mã nguồn của ứng dụng được tổ chức trong một lớp chính QuanLyAnime, được viết bằng Python và sử dụng lập trình hướng đối tượng để đảm bảo tính mô-đun và dễ bảo trì. Dưới đây là mô tả chi tiết các thành phần chính:

* Hàm \_\_init\_\_ khởi tạo cửa sổ Tkinter với kích thước 1200x800, đặt tiêu đề "Quản Lý Anime" và cho phép thay đổi kích thước (resizable(True, True)). Nó cũng gọi khoiTaoTepDuLieu để tạo các tệp JSON và hiển thị màn hình đăng nhập qua hienThiManHinhDangNhap. Các biến như boNhoDemHinhAnh, danhSachHienThiHienTai, và boNhoDemApi được khởi tạo để quản lý bộ nhớ đệm và dữ liệu tạm thời.



* Các hàm như hienThiManHinhDangNhap, hienThiManHinhDangKy, và hienThiManHinhChinh chịu trách nhiệm xây dựng giao diện. Ví dụ, hienThiManHinhChinh tạo giao diện chính với các khung tìm kiếm, sắp xếp, và hiển thị danh sách anime. Nó sử dụng tk.Canvas và ttk.Scrollbar để hỗ trợ cuộn, đồng thời gán các sự kiện cuộn chuột (MouseWheel, Button-4, Button-5) để tương thích trên Windows và Linux.
* Hàm layTopAnime và timKiemAnimeApi gọi Jikan API để lấy dữ liệu, xử lý JSON và lưu vào du\_lieu\_anime.json. Hàm layHinhAnh tải hình ảnh bìa, sử dụng lru\_cache để tối ưu hiệu suất, và taiHinhAnhKhongDongBo đảm bảo tải hình ảnh không đồng bộ.



* Quản lý danh sách yêu thích:

Hàm batTatYeuThich thêm/xóa anime khỏi danh sách yêu thích, cập nhật nguoi\_dung.json và làm mới giao diện. Hàm hienThiYeuThich lọc danh sách yêu thích để hiển thị riêng.

* Ứng dụng sử dụng threading để chạy các tác vụ nặng (như gọi API) trong luồng riêng, đảm bảo giao diện không bị treo. Hàm nhiemVuNen quản lý các tác vụ này, kết hợp với nhiemVuHoanThanh và nhiemVuThatBai để xử lý kết quả hoặc lỗi. Bộ nhớ đệm (boNhoDemApi, lru\_cache) được sử dụng để giảm thời gian tải dữ liệu và hình ảnh.

Mã nguồn được tổ chức rõ ràng, với các hàm được đặt tên mô tả và phân chia chức năng hợp lý, giúp dễ dàng bảo trì và mở rộng trong tương lai.

## Hướng dẫn sử dụng ứng dụng

Hướng dẫn sử dụng ứng dụng được thiết kế để người dùng, kể cả những người không có nhiều kinh nghiệm công nghệ, có thể dễ dàng cài đặt và sử dụng. Dưới đây là các bước chi tiết:

* Bước 1 - Cài đặt môi trường:

Người dùng cần cài đặt Python 3.8 hoặc cao hơn từ trang chính thức (python.org). Sau đó, cài đặt các thư viện cần thiết bằng lệnh:



Tkinter đã được tích hợp sẵn trong Python, nên không cần cài đặt thêm. Người dùng cần đảm bảo máy tính có kết nối internet ổn định (tối thiểu 5Mbps) để gọi Jikan API và tải hình ảnh. Ngoài ra, cần chuẩn bị hai tệp ảnh nền a1.png (khung đăng nhập/đăng ký, kích thước 300x200) và a2.png (nền toàn màn hình, kích thước 1920x1200), đặt cùng thư mục với tệp mã nguồn.

* Bước 2 - Chuẩn bị và chạy ứng dụng:

Tải mã nguồn về máy, lưu trong một thư mục có quyền đọc/ghi để lưu trữ tệp JSON (nguoi\_dung.json, du\_lieu\_anime.json). Mở terminal, điều hướng đến thư mục chứa mã nguồn và chạy lệnh:



Nếu hệ thống không nhận lệnh python, có thể thử python3 ten\_tepp.py. Ứng dụng sẽ khởi động và hiển thị màn hình đăng nhập.

* Bước 3 - Sử dụng ứng dụng:

Đăng nhập/Đăng ký: Tại màn hình đăng nhập, sử dụng tài khoản mặc định (admin/admin123 hoặc nguoi\_dung/user123) hoặc nhấn "Đăng ký" để tạo tài khoản mới. Nhập tên người dùng, mật khẩu, xác nhận mật khẩu và nhấn "Đăng ký".

Tìm kiếm và xem anime: Sau khi đăng nhập, giao diện chính hiển thị danh sách top anime. Nhập từ khóa vào ô tìm kiếm và nhấn "Tìm kiếm" (hoặc Enter) để tìm anime. Chọn tiêu chí sắp xếp (điểm số, số tập, tiêu đề) từ menu thả xuống để sắp xếp danh sách.

Quản lý danh sách yêu thích: Nhấn nút trái tim (♡) trên mỗi anime để thêm/xóa khỏi danh sách yêu thích. Chuyển sang chế độ "Yêu thích" bằng nút radio để xem danh sách đã lưu.

Làm mới và đăng xuất: Nhấn "Làm mới Top Anime" để cập nhật danh sách từ API, hoặc "Đăng xuất" để quay lại màn hình đăng nhập.

Lưu ý khi sử dụng:

Ứng dụng cần kết nối internet để lấy dữ liệu và hình ảnh. Nếu mất kết nối, chỉ hiển thị dữ liệu đã lưu cục bộ.

Đảm bảo quyền đọc/ghi trong thư mục chứa ứng dụng để lưu tệp JSON.

Nếu giao diện hiển thị không đúng (cắt xén), điều chỉnh độ phân giải màn hình tối thiểu 1280x720.

Hướng dẫn này giúp người dùng dễ dàng cài đặt và sử dụng ứng dụng, đồng thời cung cấp các lưu ý để tránh lỗi trong quá trình vận hành.

# Kiểm thử và đánh giá

## Quá trình kiểm thử

Quá trình kiểm thử được thực hiện một cách có hệ thống nhằm đảm bảo ứng dụng hoạt động ổn định, đáp ứng đúng yêu cầu chức năng và kỹ thuật, đồng thời phát hiện và khắc phục các lỗi tiềm ẩn trước khi triển khai. Kiểm thử được chia thành các giai đoạn cụ thể với các phương pháp và công cụ hỗ trợ phù hợp, tập trung vào từng khía cạnh của ứng dụng.

* Kiểm thử đơn vị (Unit Testing): Giai đoạn này tập trung vào việc kiểm tra từng chức năng riêng lẻ để đảm bảo chúng hoạt động đúng như thiết kế. Mỗi hàm trong mã nguồn được kiểm thử độc lập, sử dụng các trường hợp thử nghiệm (test case) được chuẩn bị trước. Ví dụ, hàm dangNhap được kiểm tra với các trường hợp như nhập đúng tài khoản (admin/admin123), nhập sai mật khẩu, hoặc nhập tên người dùng không tồn tại, để đảm bảo thông báo lỗi hiển thị chính xác qua messagebox.showerror. Hàm timKiemAnimeApi được kiểm tra với các từ khóa hợp lệ (như "Naruto"), từ khóa không có kết quả (như "xyzabc"), và trường hợp không nhập từ khóa (hiển thị cảnh báo). Các hàm như sapXepAnime và batTatYeuThich cũng được kiểm tra để đảm bảo danh sách được sắp xếp đúng (theo điểm số, số tập, tiêu đề) và danh sách yêu thích được cập nhật chính xác trong nguoi\_dung.json. Kiểm thử đơn vị được thực hiện thủ công bằng cách chạy từng hàm và quan sát kết quả trên giao diện hoặc tệp JSON.
* Kiểm thử tích hợp (Integration Testing): Sau khi các chức năng riêng lẻ hoạt động ổn định, nhóm tiến hành kiểm thử tích hợp để đảm bảo các thành phần phối hợp với nhau một cách đồng bộ. Một kịch bản điển hình là kiểm tra quy trình từ đăng nhập, tìm kiếm anime, thêm vào danh sách yêu thích, và hiển thị danh sách yêu thích. Ví dụ, người dùng đăng nhập bằng tài khoản "admin", tìm kiếm "Attack on Titan", nhấn nút trái tim để thêm vào danh sách yêu thích, sau đó chuyển sang chế độ "Yêu thích" để kiểm tra anime vừa thêm có hiển thị hay không. Giai đoạn này cũng kiểm tra tích hợp với Jikan API, đảm bảo các hàm như layTopAnime và timKiemAnimeApi lấy dữ liệu đúng và hiển thị lên giao diện thông qua hienThiLuoiAnime. Các lỗi tích hợp, như dữ liệu API không đồng bộ với giao diện hoặc danh sách yêu thích không cập nhật, được phát hiện và sửa chữa bằng cách kiểm tra logic trong các hàm liên quan (batTatYeuThich, nhiemVuNen).
* Kiểm thử giao diện (UI Testing): Giai đoạn này tập trung vào việc đánh giá giao diện người dùng trên các độ phân giải màn hình và hệ điều hành khác nhau. Ứng dụng được kiểm tra trên Windows 10 (độ phân giải 1920x1080 và 1280x720), Ubuntu 20.04 (1366x768), và macOS Ventura (1440x900). Các trường hợp kiểm tra bao gồm: đảm bảo ảnh nền (a1.png, a2.png) hiển thị đúng, các widget (ô nhập, nút, danh sách anime) không bị cắt xén, và thanh cuộn (ttk.Scrollbar) hoạt động mượt mà. Nhóm cũng kiểm tra các sự kiện cuộn chuột (MouseWheel trên Windows, Button-4/Button-5 trên Linux) để đảm bảo tính tương thích. Một số lỗi giao diện, như văn bản bị tràn ra ngoài khung ở độ phân giải thấp, đã được khắc phục bằng cách điều chỉnh wraplength trong ttk.Label và tối ưu bố cục lưới (grid).
* Kiểm thử hiệu suất và ngoại lệ: Nhóm thực hiện kiểm thử hiệu suất để đánh giá thời gian phản hồi của ứng dụng, đặc biệt trong các tác vụ nặng như gọi API và tải hình ảnh. Ví dụ, hàm layTopAnime được kiểm tra với thời gian phản hồi trung bình 1-2 giây khi có kết nối mạng ổn định (5Mbps). Các trường hợp ngoại lệ cũng được kiểm tra, bao gồm mất kết nối mạng (hiển thị thông báo lỗi), tệp ảnh nền không tồn tại (hiển thị lỗi qua messagebox), và lỗi đọc/ghi tệp JSON (do quyền truy cập). Các biện pháp xử lý ngoại lệ đã được áp dụng trong mã, như kiểm tra tệp bằng os.path.exists và sử dụng try-except để bắt lỗi.

Quá trình kiểm thử được thực hiện thủ công, với các thành viên nhóm ghi lại kết quả vào bảng kiểm thử (test log) để theo dõi và sửa lỗi. Tổng cộng, nhóm đã thực hiện hơn 50 trường hợp kiểm thử, đảm bảo ứng dụng hoạt động ổn định trước khi hoàn thiện.

## Kết quả kiểm thử

Kết quả kiểm thử cho thấy ứng dụng đáp ứng tốt các yêu cầu chức năng và kỹ thuật, nhưng cũng phát hiện một số hạn chế cần cải thiện. Dưới đây là phân tích chi tiết:

* Chức năng đăng nhập và đăng ký: Hàm dangNhap và hienThiManHinhDangKy hoạt động ổn định, xử lý tốt các trường hợp như nhập đúng/sai tài khoản, mật khẩu không khớp, hoặc tên người dùng đã tồn tại. Thông báo lỗi được hiển thị chính xác qua messagebox, và tệp nguoi\_dung.json được cập nhật đúng khi đăng ký tài khoản mới. Tuy nhiên, một lỗi nhỏ ban đầu là không kiểm tra khoảng trắng trong tên người dùng, đã được khắc phục bằng cách thêm kiểm tra dữ liệu đầu vào.
* Tìm kiếm và sắp xếp anime: Hàm timKiemAnimeApi trả về kết quả chính xác với các từ khóa phổ biến như "One Piece", "Naruto", và hiển thị thông báo phù hợp khi không có kết quả. Hàm sapXepAnime sắp xếp danh sách đúng theo các tiêu chí (điểm số, số tập, tiêu đề), với dữ liệu "N/A" được xử lý hợp lý (đặt giá trị mặc định là 0). Một vấn đề nhỏ là khi tìm kiếm với từ khóa quá dài, thời gian phản hồi API tăng lên 3-4 giây, nhưng đã được cải thiện bằng cách giới hạn số lượng kết quả trả về (20 anime).
* Hiển thị danh sách anime: Hàm hienThiLuoiAnime hiển thị danh sách anime đầy đủ thông tin (tiêu đề, điểm số, số tập, hình ảnh), với bố cục lưới 3 cột hoạt động tốt trên các độ phân giải khác nhau. Tải hình ảnh không đồng bộ (taiHinhAnhKhongDongBo) đảm bảo giao diện không bị treo, và bộ nhớ đệm (lru\_cache) giảm thời gian tải lại hình ảnh từ 2 giây xuống dưới 0.5 giây. Tuy nhiên, khi mất kết nối mạng, hình ảnh không tải được và hiển thị "Hình ảnh không có sẵn", phù hợp với thiết kế.
* Quản lý danh sách yêu thích: Hàm batTatYeuThich và hienThiYeuThich hoạt động chính xác, cho phép thêm/xóa anime khỏi danh sách yêu thích và hiển thị danh sách riêng. Dữ liệu được lưu trữ đúng trong nguoi\_dung.json, và giao diện cập nhật tức thời khi chuyển đổi giữa chế độ "Tất cả Anime" và "Yêu thích". Một lỗi nhỏ ban đầu là danh sách yêu thích không cập nhật nếu tệp JSON bị khóa (do quyền truy cập), đã được khắc phục bằng cách kiểm tra quyền đọc/ghi trước khi ghi dữ liệu.
* Hiệu suất và ngoại lệ: Thời gian phản hồi trung bình của ứng dụng là 1-2 giây cho các tác vụ gọi API và tải hình ảnh, đáp ứng yêu cầu kỹ thuật. Các trường hợp ngoại lệ như mất kết nối mạng, tệp ảnh không tồn tại, hoặc lỗi đọc/ghi JSON được xử lý tốt, với thông báo lỗi rõ ràng qua messagebox. Tuy nhiên, ứng dụng không có chế độ ngoại tuyến hoàn chỉnh, dẫn đến hạn chế khi không có internet – người dùng chỉ có thể xem dữ liệu đã lưu cục bộ mà không thể tìm kiếm hoặc làm mới danh sách.
* Giao diện người dùng: Giao diện hiển thị ổn định trên các độ phân giải từ 1280x720 trở lên, với các widget được căn chỉnh đúng và thanh cuộn hoạt động mượt mà. Một số lỗi giao diện ban đầu, như văn bản tiêu đề anime bị tràn ở độ phân giải thấp, đã được sửa bằng cách điều chỉnh wraplength. Ứng dụng cũng tương thích tốt trên Windows, Linux, và macOS, với các sự kiện cuộn chuột hoạt động chính xác.Tổng thể, ứng dụng đạt yêu cầu về chức năng và hiệu suất, nhưng vẫn có hạn chế về khả năng hoạt động ngoại tuyến và cần cải thiện thời gian phản hồi API trong trường hợp mạng yếu. Kết quả kiểm thử đã cung cấp cơ sở để nhóm hoàn thiện ứng dụng và xác định hướng phát triển trong tương lai.

# Kết luận.

## Tóm tắt kết quả

Dự án phát triển ứng dụng quản lý anime sử dụng Python và Tkinter đã hoàn thành thành công các mục tiêu đề ra, mang lại một công cụ hiệu quả và thân thiện cho người dùng yêu thích anime. Ứng dụng cung cấp đầy đủ các chức năng chính như đăng nhập, đăng ký tài khoản, tìm kiếm anime, sắp xếp danh sách theo các tiêu chí (điểm số, số tập, tiêu đề), và quản lý danh sách yêu thích, với dữ liệu được lấy từ Jikan API và lưu trữ cục bộ trong tệp JSON (nguoi\_dung.json, du\_lieu\_anime.json). Giao diện được thiết kế trực quan với các màn hình đăng nhập, đăng ký, và giao diện chính, sử dụng ảnh nền (a1.png, a2.png) và bố cục lưới để hiển thị danh sách anime dạng 3 cột, đảm bảo tính thẩm mỹ và dễ sử dụng. Hiệu suất của ứng dụng được tối ưu với thời gian phản hồi trung bình 1-2 giây cho các tác vụ gọi API và tải hình ảnh, nhờ vào cơ chế bộ nhớ đệm (lru\_cache) và xử lý đa luồng (threading) để tránh treo giao diện. Kết quả kiểm thử cho thấy ứng dụng hoạt động ổn định trên các hệ điều hành phổ biến như Windows, Linux, và macOS, xử lý tốt các trường hợp ngoại lệ như mất kết nối mạng hay lỗi đọc/ghi tệp. Ngoài việc đáp ứng nhu cầu của người dùng cuối, ứng dụng còn đóng vai trò như một tài liệu tham khảo giá trị cho sinh viên và nhà phát triển muốn tìm hiểu về lập trình GUI, tích hợp API, và quản lý dữ liệu cục bộ bằng Python. Tuy nhiên, ứng dụng vẫn tồn tại một số hạn chế, như phụ thuộc vào kết nối internet để lấy dữ liệu và tải hình ảnh, chưa hỗ trợ chế độ ngoại tuyến hoàn chỉnh, và thiếu các tính năng nâng cao như gợi ý anime hoặc xem thông tin chi tiết. Nhìn chung, dự án đã đạt được mục tiêu ban đầu, mang lại giá trị thực tiễn cho người dùng và tạo nền tảng vững chắc cho các cải tiến trong tương lai.

## Hướng phát triển

Dựa trên kết quả đạt được và những hạn chế đã xác định, nhóm đã đề xuất một số hướng phát triển để nâng cao chất lượng và tính năng của ứng dụng trong tương lai, nhằm đáp ứng tốt hơn nhu cầu của người dùng và mở rộng phạm vi ứng dụng.

* Hỗ trợ chế độ ngoại tuyến: Một cải tiến quan trọng là lưu trữ toàn bộ dữ liệu anime cục bộ (bao gồm hình ảnh bìa) ngay sau khi tải từ API, để người dùng có thể truy cập danh sách anime và xem thông tin cơ bản ngay cả khi không có kết nối internet. Điều này có thể được thực hiện bằng cách mở rộng tệp du\_lieu\_anime.json để lưu trữ hình ảnh dưới dạng mã hóa base64 hoặc sử dụng một cơ sở dữ liệu nhẹ như SQLite để quản lý dữ liệu hiệu quả hơn.
* Cải thiện giao diện và trải nghiệm người dùng: Giao diện có thể được nâng cấp bằng cách thêm các hiệu ứng chuyển động (animation) cho các nút và khung, hỗ trợ nhiều chủ đề giao diện (light/dark mode), và tối ưu hiển thị trên các thiết bị có độ phân giải cao hơn (như 4K). Ngoài ra, việc thêm các phím tắt (shortcut) cho các thao tác phổ biến như tìm kiếm (Ctrl+F) hoặc chuyển đổi chế độ (Ctrl+T) sẽ cải thiện trải nghiệm người dùng.
* Mở rộng tính năng: Ứng dụng có thể tích hợp các tính năng nâng cao như xem trailer anime (tích hợp với API YouTube), hiển thị thông tin chi tiết (nội dung, đạo diễn, studio sản xuất), hoặc gợi ý anime dựa trên sở thích người dùng bằng cách phân tích danh sách yêu thích và sử dụng thuật toán đơn giản (như lọc cộng tác). Ngoài ra, việc hỗ trợ thêm các loại hình giải trí khác như manga hoặc phim hoạt hình phương Tây cũng là một hướng phát triển tiềm năng để mở rộng đối tượng người dùng.
* Tối ưu hiệu suất và bảo mật: Để cải thiện hiệu suất, nhóm có thể giảm thời gian phản hồi API bằng cách sử dụng kỹ thuật tải trước (prefetching) dữ liệu hoặc nén hình ảnh trước khi hiển thị. Về bảo mật, ứng dụng cần mã hóa mật khẩu người dùng trong nguoi\_dung.json (hiện lưu dạng văn bản thô) bằng các thuật toán như bcrypt, đồng thời thêm cơ chế xác thực mạnh hơn (như mã OTP) để bảo vệ tài khoản người dùng.
* Hỗ trợ đa ngôn ngữ và quốc tế hóa: Hiện tại, ứng dụng chỉ hỗ trợ tiếng Việt và tiếng Anh, do đó việc thêm hỗ trợ đa ngôn ngữ (như tiếng Nhật, Hàn) sẽ giúp tiếp cận người dùng quốc tế, đặc biệt là cộng đồng yêu thích anime toàn cầu. Điều này có thể được thực hiện bằng cách sử dụng các thư viện như gettext để quản lý bản dịch.

Những hướng phát triển này không chỉ khắc phục các hạn chế hiện tại mà còn mở rộng tiềm năng của ứng dụng, giúp nó trở thành một công cụ toàn diện hơn cho người dùng yêu thích anime và các loại hình giải trí khác.

# **TÀI LIỆU THAM KHẢO**

Trong quá trình thực hiện đồ án, nhóm đã tham khảo nhiều tài liệu và nguồn tài nguyên trực tuyến để hỗ trợ việc phát triển ứng dụng như :

Tài liệu chính thức của Python: <https://docs.python.org/3/>

Tài liệu thư viện Tkinter: <https://docs.python.org/3/library/tkinter.html>

Tài liệu thư viện Pillow (PIL Fork): <https://pillow.readthedocs.io/>

Tài liệu thư viện Requests: <https://requests.readthedocs.io/>

Tài liệu Jikan API: <https://docs.api.jikan.moe/>

Tài liệu về functools.lru\_cache: https://docs.python.org/3/library/functools.html#functools.lru\_cache

# **PHỤ LỤC**

Trong quá trình thực hiện đồ án, nhóm đã tham khảo nhiều tài liệu và nguồn tài nguyên trực tuyến để hỗ trợ việc phát triển ứng dụng. Tài liệu chính về Python được lấy từ trang web chính thức python.org, cung cấp thông tin chi tiết về cú pháp và các thư viện chuẩn như tkinter, json. Tài liệu về thư viện PIL được tham khảo từ trang pillow.readthedocs.io, giúp nhóm hiểu cách xử lý và hiển thị hình ảnh trong ứng dụng. Thông tin về API Jikan được lấy từ trang jikan.moe, bao gồm các endpoint và cách sử dụng để lấy dữ liệu anime. Ngoài ra, nhóm tham khảo các bài viết và hướng dẫn trên Stack Overflow để giải quyết các vấn đề liên quan đến xử lý bất đồng bộ với threading và tối ưu hóa giao diện với tkinter. Các tài liệu về lập trình giao diện người dùng và xử lý JSON cũng được tham khảo từ các sách như "Python Crash Course" của Eric Matthes.