

KHOA KỸ THUẬT VÀ CÔNG NGHỆ  
BỘ MÔN CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH  
HỌC KỲ I, NĂM HỌC 2024-2025

**TÌM HIỂU NGÔN NGỮ LẬP TRÌNH  
PYTHON. XÂY DỰNG PHẦN MỀM  
QUẢN LÝ ĐIỂM ĐỒ ÁN, ĐỀ TÀI CỦA  
SINH VIÊN KHOA KT&CN- TRƯỜNG  
ĐẠI HỌC TRÀ VINH**

*Giáo viên hướng dẫn*

ThS. Trần Văn Nam

*Sinh viên thực hiện:*

Họ tên: Lâm Thanh Đình

Mã số sinh viên: 110122051

Lớp: DA22TTA

*Trà Vinh, Tháng 12 Năm 2024*

KHOA KỸ THUẬT VÀ CÔNG NGHỆ  
BỘ MÔN CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH  
HỌC KỲ I, NĂM HỌC 2024-2025

# **TÌM HIỂU NGÔN NGỮ LẬP TRÌNH PYTHON. XÂY DỰNG PHẦN MỀM**

# QUẢN LÝ ĐIỂM ĐỒ ÁN, ĐỀ TÀI CỦA SINH VIÊN KHOA KT&CN- TRƯỜNG ĐẠI HỌC TRÀ VINH

*Giáo viên hướng dẫn*

ThS. Trần Văn Nam

*Sinh viên thực hiện:*

Họ tên: Lâm Thanh Đình

Mã số sinh viên: 110122051

Lớp: DA22TTA

*Trà Vinh, Tháng 12 Năm 2024*

## NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

[illegible]

*Trà Vinh, ngày ... tháng ... năm 2024*

**Giáo viên hướng dẫn**

*(Ký và ghi rõ họ tên)*

## NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

*Trà Vinh, ngày ... tháng ... năm 2024*

**Thành viên hội đồng**

*(Ký và ghi rõ họ tên)*

## LỜI CẢM ƠN

Trước tiên, em xin chân thành cảm ơn đến Khoa Kỹ Thuật và Công nghệ, Trường Đại học Trà Vinh đã tạo điều kiện cho chúng em thực hiện đồ án cơ sở ngành, cảm ơn những thầy cô trong khoa đã dìu dắt, dạy dỗ em về các kiến thức chuyên môn và tinh thần học tập để em có kiến thức phục vụ cho đồ án của mình.

Tiếp theo, em xin bày tỏ tình cảm và lòng biết ơn đến giảng viên đã hướng dẫn em thực hiện đồ án này – TS. Trần Văn Nam, những lời góp ý đặc biệt của thầy giúp cho em có hướng đi và tầm nhìn đúng đắn. Nhờ đó mà em đã hoàn thành đúng thời hạn quy định và tích lũy được cho bản thân một lượng kiến thức đáng quý.

Bên cạnh đó, em không quên gửi lời cảm ơn đến những thầy cô, những người lập trình viên trên các nền tảng mạng xã hội đã giúp em hiểu rõ hơn về Python.

Trong quá trình thực hiện đồ án, mặc dù đã cố gắng với tất cả nỗ lực của bản thân nhưng do thời gian và kiến thức chuyên ngành còn hạn chế, không tránh được những sai lầm và thiếu sót khi tìm hiểu, đánh giá và trình bày về đề tài của bản thân. Em rất mong nhận được sự góp ý quý báu của tất cả thầy cô giảng viên bộ môn để đồ án của em được hoàn chỉnh và đầy đủ hơn.

Cuối lời, em xin kính chúc Quý Thầy Cô lời chúc sức khỏe và thành công!

Em xin chân thành cảm ơn!



## MỤC LỤC

CHƯƠNG 1. TỔNG QUAN NGHIÊN CỨU.....	11
1.1. Giới thiệu đề tài.....	11
1.2. Mục đích nghiên cứu.....	14
1.3. Đối tượng nghiên cứu .....	14
1.4. Phạm vi nghiên cứu.....	14
CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT .....	15
2.1. Ngôn ngữ Python .....	15
2.1.1 Giới thiệu về Python .....	15
2.1.2 Đặc điểm của Python .....	15
2.1.3 Ứng dụng của Python.....	17
2.1.4 Tải và cài đặt ngôn ngữ Python .....	17
2.2. Framework Django .....	19
2.2.1 Tổng quan về Django.....	19
2.2.2 Các tính năng cơ bản của Django .....	20
2.2.3 Ưu và nhược điểm của Django .....	21
2.2.4 Cài đặt, tạo và chạy dự án.....	22
2.2.4.1 Cài đặt Django .....	22
2.2.4.2 Tạo dự án .....	22
2.2.4.3 Chạy server dự án Django.....	23
2.2.5 Những thành phần chính trong Django.....	24
2.2.5.1 Views .....	24
2.2.5.2 URLS .....	25

2.2.5.3 Models.....	26
2.2.5.4 Hệ thống quản trị viên Admin .....	29
2.3. MySQL.....	30
2.3.1 Khái quát về MySQL .....	30
2.3.2 Ưu và nhược điểm của MySQL .....	31
2.3.3 Kết nối MySQL.....	32
2.4. Mô hình Model-View-Template .....	33
2.4.1 Sơ lược về mô hình .....	33
2.4.2 Ưu và nhược điểm của mô hình.....	35
CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU .....	37
3.1. Mô tả bài toán .....	37
3.2. Đặc tả yêu cầu .....	37
3.2.1 Yêu cầu chức năng.....	37
3.2.2 Yêu cầu phi chức năng.....	38
3.2.3 Sơ đồ Use-case.....	39
3.3. Cơ sở dữ liệu trong MySQL Workbench.....	39
3.3.1 Mô hình MySQL.....	39
3.3.2 Bảng thực thể .....	40
CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU .....	43
4.1. Giao diện người dùng.....	43
4.1.1 Giao diện trang đăng ký .....	43
4.1.2 Giao diện trang đăng nhập .....	44
4.1.3 Giao diện trang chủ.....	45

4.1.4 Giao diện quản lý sinh viên .....	46
4.1.5 Giao diện quản lý giảng viên .....	47
4.1.6 Giao diện quản lý kết quả .....	47
4.2. Giao diện quản trị viên.....	48
4.2.1 Trang chủ quản trị viên .....	48
4.2.2 Chức năng thêm, sửa, xóa.....	49
4.2.3 Quản lý người dung .....	49
4.2.4 Trang báo cáo thống kê.....	50
CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	51
5.1. Kết luận .....	51
5.2. Hướng phát triển .....	51

## DANH MỤC HÌNH ẢNH – BẢNG BIỂU

Hình 2.1 Cài đặt Python.....	18
Hình 2.2 Kiểm tra cài đặt thành công Python .....	19
Hình 2.3 Cài đặt Django .....	22
Hình 2.4 Kiểm tra phiên bản Django .....	22
Hình 2.5 Tạo một dự án Django .....	22
Hình 2.6 Kiểm tra cấu trúc của dự án .....	23
Hình 2.7 Thêm ứng dụng vào settings.py .....	23
Hình 2.8 Chạy server Django.....	24
Hình 2.9 Giao diện khi chạy thành công.....	24
Hình 2.10 Tập views xử lý .....	25
Hình 2.11 Xây dựng models .....	26
Hình 2.12 Tạo form nhập liệu.....	27
Hình 2.13 Chỉnh sửa view xử lý .....	27
Hình 2.14 Chỉnh sửa giao diện trang home .....	28
Hình 2.15 Kết quả .....	29
Hình 2.16 Tạo tài khoản quyền superuser .....	30
Hình 2.17 Giao diện đăng nhập tài khoản superser .....	30
Hình 2.18 Giao diện trang chủ quản trị viên.....	30
Hình 2.19 Tạo kết nối MySQL .....	32
Hình 2.20 Điều chỉnh trong settings.py .....	33
Hình 2.21 Mô hình MVT .....	34
Hình 3.1 Sơ đồ Use-case.....	39

Hình 3.2 Mô hình MySQL .....	39
Hình 3.3 Mô hình MySQL (tt) .....	40
Hình 4.1 Giao diện trang đăng ký .....	44
Hình 4.2 Giao diện trang đăng nhập .....	45
Hình 4.3 Giao diện trang chủ khi đăng nhập với tư cách là giảng viên.....	46
Hình 4.4 Giao diện quản lý sinh viên .....	47
Hình 4.5 Giao diện quản lý giảng viên .....	47
Hình 4.6 Giao diện quản lý kết quả .....	48
Hình 4.7 Giao diện trang chủ quản trị viên.....	49
Hình 4.8 Giao diện của chức năng thêm, sửa, xóa.....	49
Hình 4.9 Giao diện quản lý người dùng.....	50
Hình 4.10 Giao diện trang báo cáo thống kê .....	50
 Bảng 1 So sánh giữa MVC và MVT	 34
Bảng 2 Auth_User	40
Bảng 3 User(Sinh viên)	41
Bảng 4 Adivior (Giảng viên)	42
Bảng 5 Exam (Kết quả)	42

## CHƯƠNG 1. TỔNG QUAN NGHIÊN CỨU

### 1.1. Giới thiệu đề tài

Trong bối cảnh công nghệ số đang phát triển mạnh mẽ, việc ứng dụng công nghệ vào quản lý giáo dục, đặc biệt là quản lý điểm đồ án, đã trở thành một yêu cầu cấp thiết. Sự gia tăng số lượng sinh viên và đồ án tại các trường đại học, cao đẳng, cùng với yêu cầu ngày càng cao về tính minh bạch và hiệu quả trong quản lý, đòi hỏi các phương pháp quản lý truyền thống cần được thay thế bằng các giải pháp công nghệ hiện đại.

Trước đây, việc quản lý điểm đồ án thường được thực hiện thủ công bằng sổ sách, bảng tính hoặc các phần mềm đơn giản, rời rạc. Điều này gây khó khăn trong việc tổng hợp, thống kê, báo cáo và tra cứu thông tin điểm số, tiến độ đồ án của sinh viên. Quá trình này tốn nhiều thời gian, công sức của giảng viên và cán bộ quản lý, đồng thời tiềm ẩn nhiều sai sót do nhập liệu thủ công. Khả năng truy cập và chia sẻ thông tin cũng bị hạn chế, gây khó khăn cho việc theo dõi tiến độ học tập của sinh viên và đánh giá chất lượng đào tạo. Việc thiếu công cụ phân tích dữ liệu cũng hạn chế khả năng đưa ra các quyết định quản lý kịp thời và hiệu quả.

Dựa vào thực tế đó, đề tài "Xây dựng phần mềm quản lý điểm đồ án" được triển khai nhằm cung cấp một giải pháp công nghệ toàn diện cho khoa Kỹ thuật và Công nghệ - Trường Đại học Trà Vinh. Phần mềm này không chỉ tự động hóa quy trình quản lý điểm đồ án mà còn tận dụng các công nghệ hiện đại để nâng cao hiệu quả quản lý và chất lượng đào tạo. Sử dụng các công nghệ như Python, Framework Django, và hệ quản trị cơ sở dữ liệu như MySQL, xây dựng một nền tảng quản lý chuyên nghiệp, đáp ứng đầy đủ các yêu cầu về quản lý thông tin sinh viên, đồ án, điểm số, tiến độ thực hiện và báo cáo thống kê.

Phần mềm được thiết kế với giao diện thân thiện, dễ sử dụng, khả năng xử lý dữ liệu nhanh chóng và tính bảo mật cao, phù hợp với nhu cầu quản lý điểm đồ án của một khoa hoặc bộ môn. Phần mềm tập trung vào các chức năng cốt lõi như nhập điểm, sửa điểm, xem điểm, báo cáo thống kê. Việc tích hợp này giúp đồng bộ dữ liệu, tránh nhập liệu trùng lặp và tạo sự liên mạch trong quy trình quản lý. Hệ thống cũng được thiết kế để dễ

dàng mở rộng thêm các tính năng như phân quyền truy cập chi tiết hơn cho giảng viên và cán bộ quản lý.

Mục tiêu của phần mềm không chỉ dừng lại ở việc tự động hóa quy trình quản lý điểm đồ án mà còn hướng đến việc nâng cao trải nghiệm người dùng, giúp giảng viên và cán bộ quản lý dễ dàng tìm kiếm, nhập, sửa, xem và thống kê điểm một cách nhanh chóng và chính xác. Đồng thời, hỗ trợ quá trình quản lý đào tạo tại khoa Kỹ thuật và Công nghệ - Trường Đại học Trà Vinh trở nên hiệu quả, minh bạch và hiện đại hơn, góp phần nâng cao chất lượng đào tạo và hỗ trợ sinh viên đạt kết quả tốt nhất.

Python là một ngôn ngữ lập trình đa năng, dễ học và sở hữu một cộng đồng hỗ trợ đông đảo. Đặc biệt, Python cung cấp nhiều thư viện và framework mạnh mẽ cho phát triển ứng dụng web, bao gồm xử lý dữ liệu, xây dựng giao diện người dùng và tương tác với cơ sở dữ liệu.

Bên cạnh đó, việc xây dựng một ứng dụng web phức tạp như phần mềm quản lý điểm đồ án chỉ với Python thuần có thể tốn nhiều thời gian và công sức, dễ dàng mắc các lỗi sai lầm, đòi hỏi lập trình viên phải tự xử lý nhiều khía cạnh như định tuyến (routing), xác thực người dùng, quản lý phiên làm việc (session) và bảo mật. Do đó, việc sử dụng một framework web như Django là một lựa chọn tối ưu. Django cung cấp một bộ công cụ mạnh mẽ và đầy đủ tính năng, giúp đơn giản hóa quá trình phát triển web bằng cách cung cấp các thành phần được xây dựng sẵn để quản lý cơ sở dữ liệu, xử lý yêu cầu HTTP, bảo mật và tạo giao diện người dùng. Hơn nữa, Django tuân theo kiến trúc MVT (Model-View-Template) và các nguyên tắc lập trình tốt như DRY (Don't Repeat Yourself), giúp mã nguồn rõ ràng, dễ bảo trì và mở rộng. Vì vậy, việc lựa chọn Django không chỉ giúp tiết kiệm thời gian phát triển mà còn đảm bảo tính ổn định, bảo mật và khả năng mở rộng của phần mềm quản lý điểm đồ án.

Để lưu trữ dữ liệu về sinh viên, đồ án, điểm số, báo cáo thống kê và các thông tin liên quan, hệ quản trị cơ sở dữ liệu MySQL được lựa chọn. MySQL Workbench, một công cụ giúp đơn giản hóa việc quản lý và truy vấn dữ liệu.

Về phần frontend sử dụng HTML, CSS và JavaScript được sử dụng để xây dựng giao diện thân thiện, dễ sử dụng, đảm bảo giao diện responsive và tương tác khá tốt. Visual Studio Code được lựa chọn làm môi trường phát triển tích hợp (IDE), cung cấp các công cụ hỗ trợ lập trình hiệu quả như gỡ lỗi, quản lý mã nguồn và gợi ý mã.



## **1.2. Mục đích nghiên cứu**

Mục đích chính của đề tài là xây dựng một Phần mềm quản lý điểm đồ án, đề tài của sinh viên khoa Kỹ thuật và Công nghệ - Trường Đại học Trà Vinh, nhằm tự động hóa các quy trình quản lý thông tin sinh viên, giảng viên, đề tài và điểm số dựa trên ngôn ngữ lập trình Python và framework Django. Phần mềm này sẽ giúp giảm thiểu sai sót trong việc ghi chép, lưu trữ và truy xuất dữ liệu, đồng thời cải thiện hiệu quả quản lý điểm đồ án và tiến độ thực hiện các đề tài. Bên cạnh đó, phần mềm sẽ cung cấp giao diện trực quan, dễ sử dụng cho cả giảng viên và sinh viên, tích hợp các tính năng như báo cáo thống kê, khả năng theo dõi tiến độ thực hiện đồ án. Đề tài cũng giúp em áp dụng lý thuyết đã học vào thực tế, nâng cao kỹ năng lập trình, đồng thời làm quen với quy trình phát triển phần mềm hiện đại và chuẩn bị cho những dự án phần mềm lớn trong tương lai.

## **1.3. Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài gồm các khía cạnh công nghệ, công cụ và quy trình liên quan đến việc thiết kế, xây dựng và vận hành một phần mềm quản lý điểm đồ án. Cụ thể, đề tài sẽ nghiên cứu, khảo sát và ứng dụng ngôn ngữ lập trình Python với framework Django chuyên biệt cho phát triển backend. Nền tảng phát triển được lựa chọn là Django nhằm tối ưu hóa việc quản lý dữ liệu, xử lý các quy tắc nghiệp vụ và xây dựng các giao diện lập trình ứng dụng API. Cơ sở dữ liệu sẽ được sử dụng để lưu trữ và quản lý dữ liệu một cách hiệu quả. Về phía giao diện người dùng, đề tài sẽ áp dụng HTML, CSS, JavaScript và Bootstrap để đảm bảo trải nghiệm người dùng thân thiện và trực quan. Bên cạnh đó, đề tài cũng sẽ nghiên cứu các quy trình nghiệp vụ quản lý điểm đồ án, bao gồm các chức năng như nhập điểm, cập nhật điểm, tổng hợp điểm, báo cáo kết quả và quản lý thông tin đồ án, sinh viên.

## **1.4. Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài tập trung vào việc ứng dụng Python và Django để xây dựng phần mềm quản lý điểm đồ án cơ bản. Phần backend sẽ được triển khai

bằng Django để xử lý logic nghiệp vụ và quản lý cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu sẽ sử dụng MySQL. Giao diện người dùng được xây dựng bằng HTML, CSS và JavaScript nhằm đảm bảo tính thân thiện và dễ sử dụng. Đề tài giới hạn trong phạm vi quản lý điểm, đồ án và sinh viên của khoa Kỹ thuật và Công nghệ - Trường Đại học Trà Vinh, không bao gồm các vấn đề bảo mật nâng cao hay các công nghệ ngoài phạm vi đã chọn.

## **CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT**

### **2.1. Ngôn ngữ Python**

#### **2.1.1 Giới thiệu về Python**

Python là một ngôn ngữ lập trình thông dịch và đa mục đích, nổi tiếng với cú pháp đơn giản và dễ đọc. Python được sử dụng phổ biến trong các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và học máy. Các lập trình viên ưa dùng Python vì tính hiệu quả, dễ học cũng như khả năng chạy trên đa nền tảng. Phần mềm Python có sẵn miễn phí, tích hợp mượt mà vào các hệ thống và thúc đẩy tốc độ phát triển.

Guido van Rossum bắt đầu nghiên cứu Python vào cuối những năm 1980 với tư cách là ngôn ngữ kế thừa cho ngôn ngữ lập trình ABC và phát hành nó lần đầu tiên vào năm 1991 với tên gọi Python 0.9.0. Python 2.0 được ra mắt vào năm 2000. Python 3.0 được ra mắt vào năm 2008, là bản sửa đổi lớn không hoàn toàn tương thích ngược với các phiên bản trước đó. Python 2.7.18, được phát hành vào năm 2020, là bản phát hành cuối cùng của Python 2. Phiên bản mới nhất của Python hiện tại là Python 3.12.3, đã được phát hành và đem đến một loạt cải tiến đáng kể cùng nhiều sửa chữa lỗi quan trọng, nhằm tăng cường hiệu quả, tính năng và cấp độ bảo mật của ngôn ngữ lập trình này.

#### **2.1.2 Đặc điểm của Python**

Python có các đặc điểm nổi bật:

Đơn giản và dễ học: Python nổi tiếng với cú pháp rõ ràng, dễ đọc, gần gũi với tiếng Anh thông thường. Điều này giúp người mới bắt đầu dễ dàng tiếp cận và làm quen với lập trình. Khác với nhiều ngôn ngữ khác sử dụng dấu ngoặc nhọn {} để phân tách khối mã, Python sử dụng thụt lề (indentation), giúp code gọn gàng và dễ nhìn hơn.

Miễn phí và mã nguồn mở: Python được phát hành theo giấy phép mã nguồn mở (Open Source), cho phép người dùng tự do sử dụng, sửa đổi và phân phối. Điều này tạo điều kiện cho một cộng đồng phát triển mạnh mẽ, đóng góp vào việc cải tiến và mở rộng ngôn ngữ.

Khả năng di động (Portable): Python có thể chạy trên nhiều nền tảng hệ điều hành khác nhau như Windows, macOS, Linux, Unix, v.v. Điều này giúp cho việc phát triển ứng dụng trở nên linh hoạt hơn, không bị giới hạn bởi một hệ điều hành cụ thể.

Khả năng mở rộng (Extensible) và khả năng nhúng (Embeddable): Python có thể được mở rộng bằng cách sử dụng các module được viết bằng C/C++. Điều này cho phép tận dụng hiệu năng của các ngôn ngữ bậc thấp để tối ưu hóa những phần quan trọng của ứng dụng. Ngược lại, Python cũng có thể được nhúng vào các ứng dụng được viết bằng các ngôn ngữ khác, cho phép tích hợp các tính năng của Python vào các hệ thống hiện có.

Ngôn ngữ thông dịch cấp cao: Python là ngôn ngữ thông dịch (interpreted language), có nghĩa là mã nguồn được thực thi trực tiếp từng dòng một bởi trình thông dịch, không cần phải biên dịch thành mã máy trước khi chạy. Điều này giúp cho quá trình phát triển nhanh chóng hơn, dễ dàng gỡ lỗi. Tuy nhiên, tốc độ thực thi của Python thường chậm hơn so với các ngôn ngữ biên dịch như C/C++.

Thư viện tiêu chuẩn phong phú: Python đi kèm với một thư viện tiêu chuẩn rất lớn (Standard Library), cung cấp nhiều module và gói hỗ trợ cho nhiều lĩnh vực khác nhau như xử lý văn bản, xử lý số, mạng, web, đồ họa, v.v. Ngoài ra, cộng đồng Python còn phát triển một kho tàng các thư viện bên ngoài (third-party libraries) vô cùng đa dạng và mạnh mẽ, giúp giải quyết hầu hết các bài toán trong thực tế.

### 2.1.3 Ứng dụng của Python

Python là ngôn ngữ được ứng dụng trong nhiều lĩnh vực khác nhau.

Về Website: Python là lựa chọn hàng đầu của các lập trình viên khi phát triển Website. Ngôn ngữ Python giúp xây dựng được những web framework linh hoạt, có tính bảo mật cao như Django, Pyramid, Flask hay Plone. Các web framework này đi kèm với thư viện và module tiêu chuẩn, hỗ trợ đơn giản hóa tác vụ như tương tác, quản lý nội dung, liên kết cơ sở dữ liệu, dễ dàng giao tiếp với các giao thức internet như XML, HTTP, SMTP...

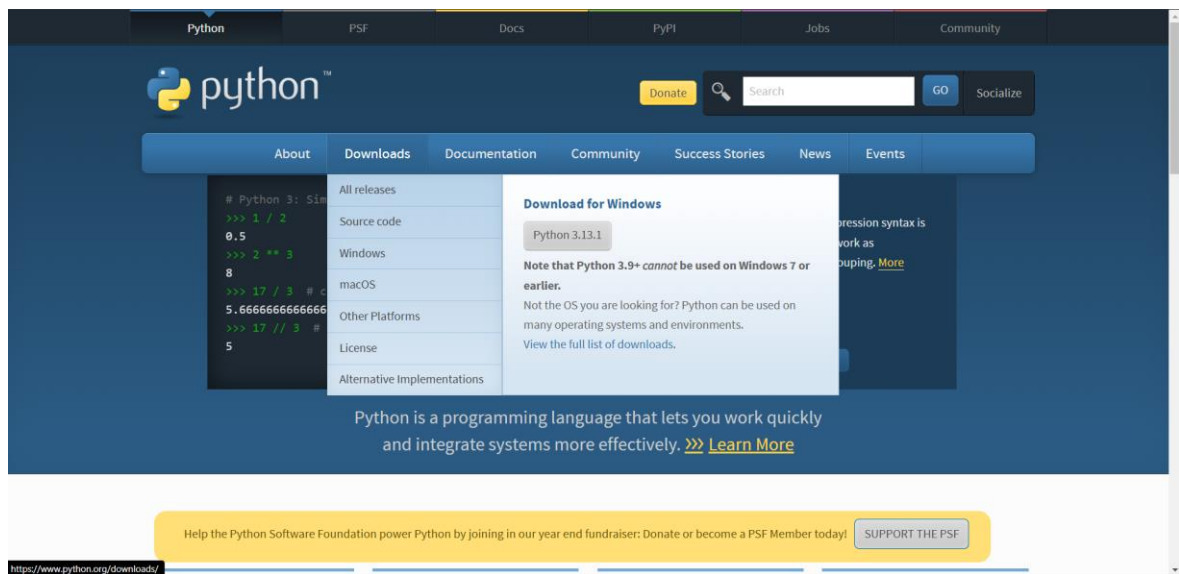
Về ứng dụng khoa học và số: Nếu như bạn chưa biết, Python cung cấp khung xử lý dữ liệu khoa học và tính toán phù hợp để lập trình các ứng dụng khoa học và số. Các ứng dụng lập trình Python phổ biến có thể kể đến như FreeCAD (dựng mô hình 3D), Abaqus (phần mềm phần tử hữu hạn). Các Python packages hữu dụng cho chủ đề tính toán khoa học và số: SciPy, Pandas, IPython, Numeric Python

Về lập trình Game: Python, mặc dù không phải là ngôn ngữ lập trình chính trong ngành công nghiệp phát triển game so với C++ hay C#, tuy nhiên nó vẫn được sử dụng rộng rãi trong các dự án game nhỏ, game giáo dục, hoặc các nguyên mẫu (prototypes) nhờ vào sự hỗ trợ của các thư viện như Pygame, Pyglet, và Arcade.

Có khả năng tự động hóa: Nếu bạn muốn tìm một ngôn ngữ dễ dàng viết chương trình tự động hóa các tác vụ lặp đi lặp lại, giúp tiết kiệm thời gian và giảm thiểu lỗi so với việc thực hiện thủ công, thì Python chính là lựa chọn hoàn hảo dành cho bạn với cú pháp đơn giản, dễ nhớ. Ví dụ: Tự động gửi email thông báo.

### 2.1.4 Tải và cài đặt ngôn ngữ Python

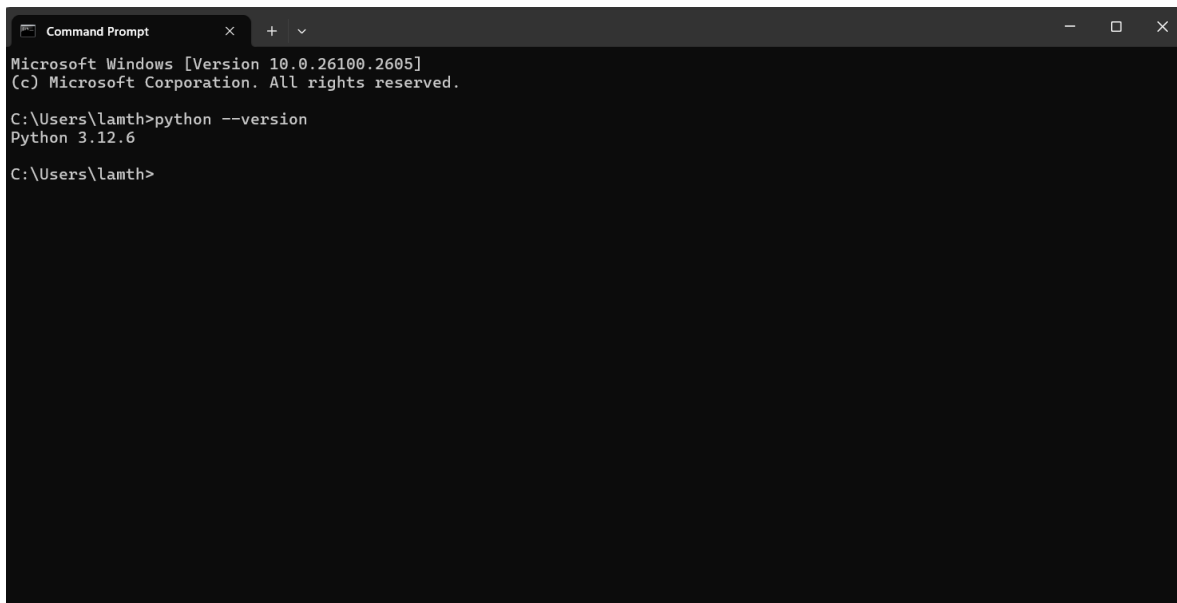
Truy cập vào trang web chính thức của Python tại <https://www.python.org/>, sau đó vào mục Download, chọn Python 3.13.1 ( đây là phiên bản do trang web của Python đề xuất cho hệ điều hành Window hoặc Mac của bạn). Sau đó nhấn vào phiên bản đó và tải về máy.



Hình 2.1 Cài đặt Python

Sau khi đã tải về máy hoàn tất, mở file bạn vừa tải và làm theo hướng dẫn của file để hoàn chỉnh bước cài đặt. Lưu ý, bạn phải chọn tùy chọn Add Python to Path trong quá trình cài đặt để không dẫn đến lỗi sau khi quá trình cài đặt hoàn tất. Điều này sẽ giúp bạn thực hiện được các dòng lệnh của Python.

Để chắc chắn rằng bạn đã tải thành công ngôn ngữ Python hay chưa, hãy mở Terminal trong Visual Code hoặc Command Prompt (cmd) trên Windows, sau đó chạy dòng lệnh: **python --version**.



```
Command Prompt
Microsoft Windows [Version 10.0.26100.2605]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lamth>python --version
Python 3.12.6

C:\Users\lamth>
```

Hình 2.2 Kiểm tra cài đặt thành công Python

## 2.2. Framework Django

### 2.2.1 Tổng quan về Django

Django là một framework lập trình web bậc cao, mã nguồn mở được viết bằng ngôn ngữ lập trình Python, giúp đơn giản hóa việc xây dựng ứng dụng web nhờ vào các mô-đun tập trung vào tính năng “tái sử dụng”, không cần làm lại những việc đã làm. Django là một Framework giúp bạn lập trình web nhanh hơn và tốt hơn so với các framework như Flask mà lại dễ hơn nhiều. Bởi vì Framework này được xây dựng bởi các nhà phát triển có kinh nghiệm nên nó có khả năng xử lý hầu hết những phần phức tạp của việc phát triển web.

Framework này có một cộng đồng người dùng đông đảo và có nhiều tài liệu hỗ trợ.

Django là một framework web Python mạnh mẽ, giúp xây dựng ứng dụng web nhanh chóng và an toàn. Với triết lý "batteries-included", Django cung cấp sẵn nhiều công cụ, từ quản lý người dùng đến ORM, giúp lập trình viên tập trung vào logic ứng dụng. Đặc biệt, Django chú trọng bảo mật, tích hợp sẵn các biện pháp chống lại các lỗ hổng phổ biến như SQL Injection và XSS.

### 2.2.2 Các tính năng cơ bản của Django

Một số tính năng cơ bản:

ORM (Object-Relational Mapping): trong Django cho phép bạn làm việc với cơ sở dữ liệu bằng cách sử dụng các lớp và đối tượng Python thay vì phải viết các câu lệnh SQL trực tiếp. Django ORM giúp bạn truy vấn dữ liệu trong cơ sở dữ liệu bằng cách sử dụng các phương thức Python, giúp việc truy vấn trở nên trực quan hơn so với SQL.

Admin: là một giao diện quản trị tự động được tạo ra từ các models (mô hình) bạn định nghĩa. Nó giúp quản lý và thao tác với dữ liệu ứng dụng web một cách dễ dàng và trực quan, mà không cần viết code HTML cho giao diện quản trị.

URL Routing: trong Django là một cơ chế cho phép bạn định tuyến các yêu cầu HTTP đến các view cụ thể dựa trên URL mà người dùng truy cập. Khi người dùng truy cập một URL trên trang web của bạn, Django sẽ tìm kiếm các mẫu URL (URL patterns) được định nghĩa trong các tệp urls.py để tìm ra view phù hợp để xử lý yêu cầu. Mỗi mẫu URL thường được liên kết với một hàm view hoặc một class view trong Django.

Template Engine: trong Django là một hệ thống giúp bạn xây dựng giao diện người dùng (UI) cho ứng dụng web của mình. Nó giúp bạn tách biệt rõ ràng giữa phần logic xử lý dữ liệu (backend) và phần hiển thị giao diện (frontend). Template Engine của Django giúp bạn tạo các trang HTML động bằng cách sử dụng các biến, logic điều khiển (như vòng lặp và điều kiện), và các template inheritance.

Authentication và Authorization: Django cung cấp hệ thống xác thực (authentication) để quản lý người dùng và hệ thống phân quyền (authorization) nhằm kiểm soát quyền truy cập đến các phần khác nhau của ứng dụng.

Kiến trúc MVT (Model-View-Template): tương tự MVC, nó tách biệt logic xử lý (View), dữ liệu (Model) và giao diện (Template), giúp code dễ bảo trì và quản lý hơn. Điểm khác biệt là Django tự xử lý phần "Controller", giúp lập trình viên tập trung vào Model, View và Template.

### 2.2.3 Ưu và nhược điểm của Django

#### **Ưu điểm:**

**Đảm bảo về tính bảo mật:** Django rất coi trọng vấn đề bảo mật và giúp các nhà phát triển tránh được nhiều lỗi bảo mật phổ biến. Hệ thống xác thực người dùng của Django cung cấp một cách an toàn để quản lý tài khoản và mật khẩu người dùng.

**Đơn giản và nhanh chóng:** Django tối ưu hóa tốc độ phát triển web bằng code ngắn gọn, thư viện mạnh mẽ và khả năng tự động loại bỏ mã trùng lặp. Tiết kiệm thời gian, tập trung vào kết quả.

**Cho khả năng mở rộng ứng dụng:** Khả năng mở rộng của Django cho phép ứng dụng xử lý lượng truy cập và dữ liệu lớn mà không giảm hiệu suất, thông qua mở rộng ngang (phân tán máy chủ), mở rộng dọc (nâng cấp phần cứng), tối ưu cơ sở dữ liệu và caching. Django giúp ứng dụng "lớn lên" theo nhu cầu.

**Cộng đồng người dùng và thiết lập hoàn thiện:** Django set up khá tốt. Điều này đã được chứng nhận bởi thời gian và một số người sử dụng khung công tác này. Nó có một big user cộng đồng, được hỗ trợ truy cập thông qua nhiều diễn đàn, kênh và các chuyên trang web.

#### **Nhược điểm:**

Khi phát triển các dự án nhỏ với Django thì có thể sẽ tồn tại một vài vấn đề.

Chỉ định URL bằng quy tắc biểu thức không dễ thực hiện (đặc biệt là người mới bắt đầu). Nó cũng gây cảm giác cồng kềnh đối với các dự án nhỏ.

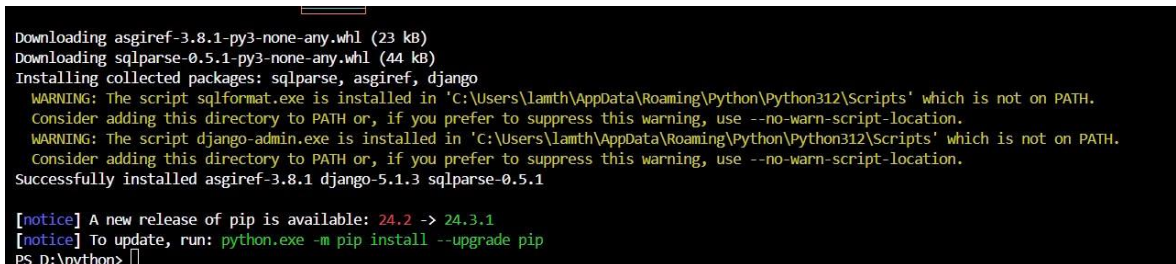
Mẫu lỗi sẽ không phải là lỗi thông báo. Nếu bạn không biết, bạn sẽ mất rất nhiều thời gian để tìm ra vấn đề ở đâu hoặc tệ hơn, bạn sẽ không biết ứng dụng của mình đối với vấn đề.



## 2.2.4 Cài đặt, tạo và chạy dự án

### 2.2.4.1 Cài đặt Django

Để cài đặt Django, ta tiến hành gõ lệnh “pip install django” trong terminal ngay sau khi đã cài đặt xong Python.

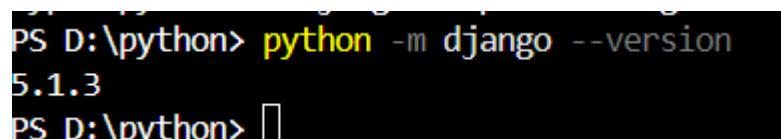


```
Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)
Downloading sqlparse-0.5.1-py3-none-any.whl (44 kB)
Installing collected packages: sqlparse, asgiref, django
WARNING: The script sqlformat.exe is installed in 'C:\Users\lamth\AppData\Roaming\Python\Python312\Scripts' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The script django-admin.exe is installed in 'C:\Users\lamth\AppData\Roaming\Python\Python312\Scripts' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed asgiref-3.8.1 django-5.1.3 sqlparse-0.5.1

[notice] A new release of pip is available: 24.2 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
PS D:\python> |
```

Hình 2.3 Cài đặt Django

Để kiểm tra phiên bản Django đã cài đặt, ta gõ lệnh “python -m django –version”.

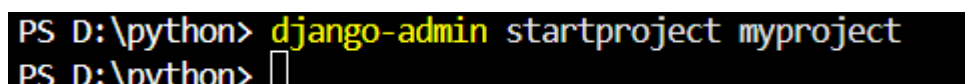


```
PS D:\python> python -m django --version
5.1.3
PS D:\python> |
```

Hình 2.4 Kiểm tra phiên bản Django

### 2.2.4.2 Tạo dự án

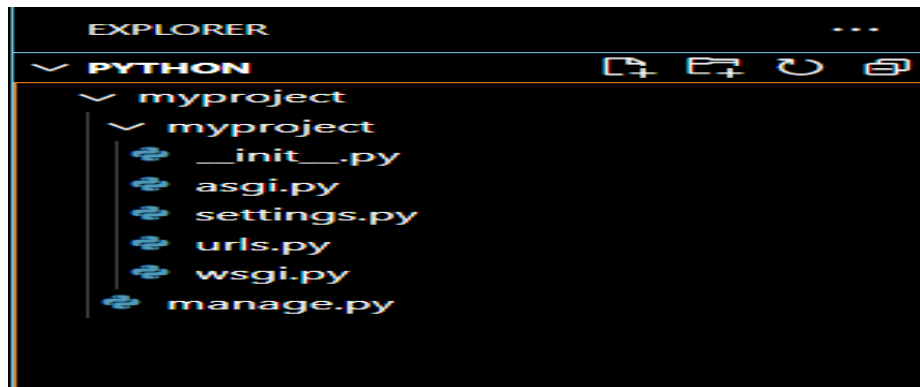
Để tạo 1 dự án trong Django, ta phải gõ câu lệnh vào terminal như sau: **django-admin startproject (tên dự án cần tạo)**. Ví dụ **django-admin startproject myproject**.



```
PS D:\python> django-admin startproject myproject
PS D:\python> |
```

Hình 2.5 Tạo một dự án Django

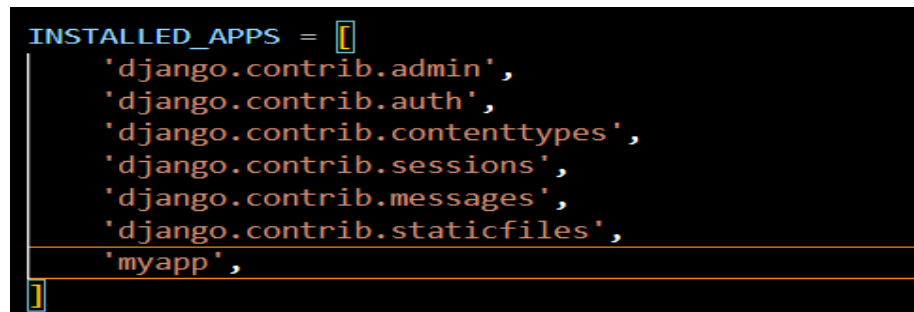
Tiếp theo, mở thư mục dự án vừa tạo để xem cấu trúc và các thành phần chính:



Hình 2.6 Kiểm tra cấu trúc của dự án

Trong Django, kiến trúc được khuyến nghị là chia dự án thành các ứng dụng nhỏ hơn, mỗi ứng dụng chịu trách nhiệm cho một phần chức năng cụ thể. Để tạo một ứng dụng mới, sử dụng lệnh **python manage.py startapp (tên\_ứng\_dụng)**. Ví dụ **python manage.py startapp myapp**. Lệnh này sẽ tạo một thư mục mới với cấu trúc thư mục và các tệp cơ bản cần thiết cho ứng dụng.

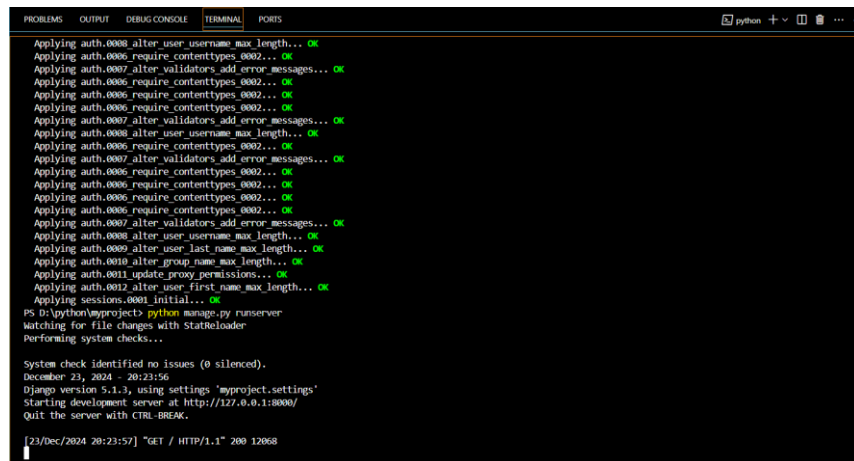
Sau khi tạo ứng dụng, bạn cần thêm nó vào INSTALLED\_APPS trong tệp settings.py của dự án để Django nhận diện ứng dụng.



Hình 2.7 Thêm ứng dụng vào settings.py

#### 2.2.4.3 Chạy server dự án Django

Sau khi đã thêm ứng dụng vào trong tệp settings.py, bạn cần phải gõ lệnh **python manage.py migrate** để cập nhật các thay đổi và cấu hình lại cho website. Tiếp đó, bạn phải gõ lệnh **python manage.py runserver** để chạy server. Khi đó server sẽ đề địa chỉ trong terminal để ta click vào ( Ctrl + click ).



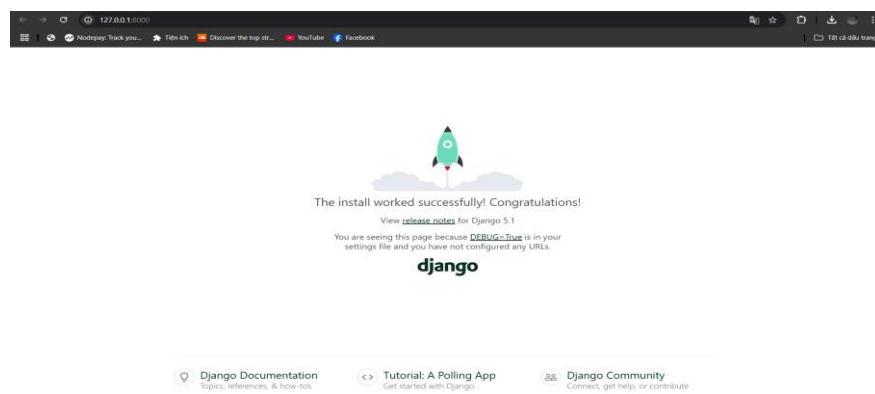
```
Applying auth.0006_alter_user_username_max_length... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0008_alter_user_username_max_length... OK
Applying auth.0009_alter_user_last_name_max_length... OK
Applying auth.0010_alter_group_name_max_length... OK
Applying auth.0011_update_proxy_permissions... OK
Applying auth.0012_alter_user_first_name_max_length... OK
Applying sessions.0001_initial... OK
PS D:\python\myproject> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
December 23, 2024 - 20:23:56
Django version 5.1.3, using settings 'myproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

[23/Dec/2024 20:23:57] "GET / HTTP/1.1" 200 12068
```

Hình 2.8 Chạy server Django

Khi đã truy cập vào địa chỉ <http://127.0.0.1:8000/>, nếu thấy thông báo The install worked successfully! Congratulations!, thì đã thành công. Đến đây dự án đang sẵn sàng để phát triển.



Hình 2.9 Giao diện khi chạy thành công

Cuối cùng là cần triển khai định nghĩa của các models trong models.py, tạo các views để xử lý logic, kết nối giữa views và urls trong urls.py và thiết kế giao diện bằng cách tạo file templates/.

## 2.2.5 Những thành phần chính trong Django

### 2.2.5.1 Views

Trong Django, thuật ngữ "**Views**" đề cập đến một phần quan trọng trong kiến trúc của framework, cụ thể là trong mẫu MVT (Model-View-Template) mà Django sử dụng (một biến thể của MVC - Model-View-Controller). Mặc dù tên gọi có vẻ đơn

giản, "**Views**" đóng vai trò trung tâm trong việc xử lý logic ứng dụng và tương tác giữa dữ liệu models và giao diện người dùng templates.

```
myproject > myapp > views.py > ...
1  from django.shortcuts import render
2  from django.http import HttpResponse
3  -----
4
5  def home(request):
6  |   return render(request, "home.html")
7  def base(request):
8  |   return render(request, "base.html")
```

Hình 2.10 Tập views xử lý

### 2.2.5.2 URLS

Tập urls.py thường sẽ nằm trong thư mục dự án và cả ứng dụng, đóng vai trò quan trọng trong việc định tuyến URL của một dự án Django. Nó như một "bản đồ" giúp Django hiểu được khi người dùng truy cập một địa chỉ (URL) cụ thể, thì cần phải thực hiện hành động gì, cụ thể là gọi view (hàm xử lý) nào để trả về nội dung tương ứng. Và phần urlpatterns sẽ là nơi để chứa liên kết chỉ dẫn đến các trang trong website.

Cấu trúc khai báo cơ bản: path(gồm có route, views, name)

Route: Chuỗi ký tự, là đường dẫn (URL path) mà Django sử dụng để khớp yêu cầu từ người dùng. Đây là phần xác định URL mà bạn muốn ánh xạ đến một hàm xử lý (view). (Ví dụ '' hoặc 'home/' là trang đầu tiên khi truy cập vào web).

Views: Hàm hoặc lớp view, chịu trách nhiệm xử lý logic khi một URL được khớp. Đây là nơi bạn viết mã xử lý dữ liệu, truy vấn cơ sở dữ liệu, và trả về phản hồi HTTP. (Ví dụ views.home)

Name: Chuỗi ký tự, là một tên duy nhất được đặt cho URL pattern. Đây là cách bạn tham chiếu đến URL trong các template, view hoặc các nơi khác trong ứng dụng Django.

Phần include(): Giúp chia nhỏ urlpatterns theo từng ứng dụng hoặc chức năng, giữ file urls.py chính ngắn gọn, dễ quản lý. Mỗi ứng dụng tự quản lý các URL của mình qua urls.py, để mở rộng và chỉnh sửa mà không ảnh hưởng đến ứng dụng khác.

Cho phép các ứng dụng có thể được sử dụng lại trong các dự án khác chỉ bằng cách include.

### 2.2.5.3 Models

Trong framework Django, Model giữ một vị trí then chốt trong kiến trúc MVT (Model-View-Template), đóng vai trò như một lớp trừu tượng mạnh mẽ để quản lý và tương tác với dữ liệu được lưu trữ trong cơ sở dữ liệu. Nó không chỉ đơn thuần là một bản sao của cấu trúc bảng biểu mà còn cung cấp một loạt các tính năng giúp đơn giản hóa quá trình phát triển ứng dụng web.

Models sẽ giúp bảo trì dữ liệu, tự động hóa các thay đổi, đảm bảo an toàn khi thao tác với cơ sở dữ liệu.

Để tạo models với views một cách đơn giản, ta sẽ thực hiện các bước sau trong Django:

#### Bước 1: Tạo Models

Đầu tiên, vào file models.py để định nghĩa một models đơn giản ( gọi là Student ), trong đó bao gồm các kiểu dữ liệu là Char với độ dài 200. unique=true (chỉ duy nhất, không trùng lặp), làm tương tự với những phần còn lại.

```
myproject > myapp > models.py > ...
1  from django.db import models
2
3  # Create your models here.
4  class Student(models.Model):
5      full_name = models.CharField(max_length=200) #Họ tên
6      date_of_birth = models.DateField() #Ngày sinh
7      email = models.EmailField(max_length=200, unique=True)#Email
8      student_id = models.CharField(max_length=20, unique=True)#Mã số sinh viên
9
10     def __str__(self):
11         return f"{self.full_name}"#Hiển thị họ tên
```

Hình 2.11 Xây dựng models

Đến đây, khi đã tạo được một models, áp dụng vào cơ sở dữ liệu bằng cách phải lưu, cập nhật lại thay đổi và cấu hình lại cho trang web. Gõ các dòng lệnh sau lần lượt là : **python manage.py makemigrations** và **python manage.py migrate**.

#### Bước 2: Tạo form nhập và cập nhật views để xử lý form đã tạo

Để tạo được form nhập cho người dùng nhập dữ liệu vào trong, bạn có thể tạo một thư mục forms.py riêng biệt để không bị lẫn lộn hoặc có thể tạo trực tiếp vào trong views.py.

```
myproject > myapp > forms.py > ...
1 from django import forms
2 from .models import Student
3
4 class StudentForm(forms.ModelForm):
5     class Meta:
6         model = Student
7         fields = ['full_name', 'date_of_birth', 'email', 'student_id'] # Danh sách các trường của model Student
8         widgets = {
9             'date_of_birth': forms.DateInput(attrs={'type': 'date'}), #Kiểu chọn hiển thị lịch
10        }
11
```

Hình 2.12 Tạo form nhập liệu

Sau khi đã tạo thành công một forms.py để có form nhập liệu, sang bên file views.py để viết một hàm views xử lý từ việc hiển thị và dữ liệu đã nhập vào form.

```
myproject > myapp > views.py > ...
1 from django.shortcuts import render, redirect
2 from django.http import HttpResponseRedirect
3 from django import forms
4 from .models import Student
5 from .forms import StudentForm
6
7
8 #views xử lý để hiển thị forms và dữ liệu nhập từ người dùng
9 def home(request):
10     if request.method == "POST":
11         form = StudentForm(request.POST)
12         if form.is_valid():
13             form.save()
14             return redirect('home')
15     else:
16         form = StudentForm()
17         students = Student.objects.all()
18         return render(request, "home.html", {"form": form, "student_list": students})
19
```

Hình 2.13 Chỉnh sửa view xử lý

Như ta thấy, hàm **home** sẽ xử lý yêu cầu từ người dùng khi họ gửi form qua phương thức **POST**. Nếu form được gửi đi, hàm sẽ tạo một đối tượng **StudentForm** với dữ liệu mà người dùng đã nhập vào từ **request.POST**. Sau đó, hàm sẽ kiểm tra tính hợp lệ của form bằng cách gọi **is\_valid()**. Nếu form hợp lệ (tức là tất cả các trường bắt buộc đều có dữ liệu hợp lệ), hàm sẽ lưu thông tin sinh viên vào cơ sở dữ liệu bằng phương thức **form.save()**.

Sau khi lưu thành công, người dùng sẽ được chuyển hướng về trang home thông qua **redirect('home')**, làm mới lại trang và hiển thị danh sách các sinh viên mới nhất. Nếu người dùng chưa gửi form, một form trống sẽ được khởi tạo và hiển thị. Cuối cùng, tất cả các đối tượng form và danh sách sinh viên (**students**) sẽ được truyền vào template **home.html** để hiển thị thông tin lên giao diện người dùng.

Bước 3: Chỉnh sửa giao diện trang home để hiện form và xuất dữ liệu nhập từ form.

```
myproject > myapp > templates > > home.html
4  {% block %}
5  {% block content %}
6  <body>
7  <h1>Chào mừng đến với trang chủ</h1>
8  <!-- Hiển thị nội dung -->
9  <h2>Thêm sinh viên</h2>
10 <form method="post">
11   {% csrf_token %}
12   {{ form.as_p }}
13   <button type="submit">Thêm</button>
14 </form>
15 <h2>Danh sách sinh viên</h2>
16 <table border="1">
17   <thead>
18     <tr>
19       <th>Họ và tên</th>
20       <th>Ngày sinh</th>
21       <th>Email</th>
22       <th>Mã số sinh viên</th>
23     </tr>
24   </thead>
25   <tbody>
26     {% for student in student_list %}
27     <tr>
28       <td>{{ student.full_name }}</td>
29       <td>{{ student.date_of_birth }}</td>
30       <td>{{ student.email }}</td>
31       <td>{{ student.student_id }}</td>
32     </tr>
33   </tbody>
34   <tr>
35     <td colspan="4">Chưa có sinh viên nào</td>
36   </tr>
37 </table>
38 </body>
39 </html>
```

Hình 2.14 Chỉnh sửa giao diện trang home

Đoạn mã trên sử dụng Django template để tạo giao diện trang chủ hiển thị danh sách sinh viên và cho phép người dùng thêm sinh viên mới. Trong đó, form để thêm sinh viên được hiển thị dưới dạng các thẻ **<p>** thông qua cú pháp **{{ form.as\_p }}**, giúp trình bày các trường thông tin trong form một cách dễ nhìn và rõ ràng. Đặc biệt, mã **{% csrf\_token %}** được sử dụng để bảo vệ form khỏi các tấn công CSRF, đảm bảo an toàn khi người dùng gửi dữ liệu lên server. Sau khi người dùng gửi form, thông tin sinh viên mới sẽ được lưu vào cơ sở dữ liệu và hiển thị trong danh sách sinh viên.

Bước 4: Cập nhật lại URLs: Thêm url trang home trong urls.py.

Bước 5: Khởi động server: Tiến hành truy cập vào trang web, khi trang web hiển thị form nhập, tiến hành nhập dữ liệu vào form và bấm nút “Thêm”:

### Thêm sinh viên

Full name:

Date of birth:

Email:

Student id:

### Danh sách sinh viên

#	Họ và tên	Ngày sinh	Email	Mã số sinh viên
1	Đạt	Dec. 24, 2024	lamthanhdinh123@gmail.com	11011
2	Tuấn	Jan. 19, 2004	tuan123@gmail.com	11012
3	Thiên	Sept. 10, 2004	thien@gmail.com	110
4	Thiên	Jan. 1, 2004	thientv@gmail.com	111
5	Tuấn	Nov. 11, 2004	tuan12223@gmail.com	123
6	Phong	Dec. 29, 2005	phong2005@gmail.com	112

Hình 2.15 Kết quả

#### 2.2.5.4 Hệ thống quản trị viên Admin

Hệ thống Admin của Django là một công cụ quản trị cực kỳ hữu ích và mạnh mẽ, được tích hợp sẵn trong Django framework. Nó cho phép người quản trị (thường là quản trị viên hệ thống hoặc các nhà phát triển) dễ dàng quản lý và thao tác với dữ liệu trong cơ sở dữ liệu thông qua giao diện web trực quan, mà không cần phải sử dụng câu lệnh SQL phức tạp hay công cụ dòng lệnh. Django Admin không chỉ giúp tiết kiệm thời gian mà còn cung cấp một phương pháp trực quan để theo dõi và quản lý các mô hình dữ liệu trong ứng dụng.

Một trong những điểm mạnh của Django Admin là khả năng tùy chỉnh linh hoạt. Người dùng có thể thay đổi cách thức hiển thị dữ liệu, thêm các tính năng như tìm kiếm, lọc, phân trang, và nhiều tính năng khác để phù hợp với các nhu cầu cụ thể của từng dự án. Điều này giúp các quản trị viên có thể dễ dàng kiểm soát và thao tác với các bản ghi trong cơ sở dữ liệu mà không cần phải viết mã phức tạp.

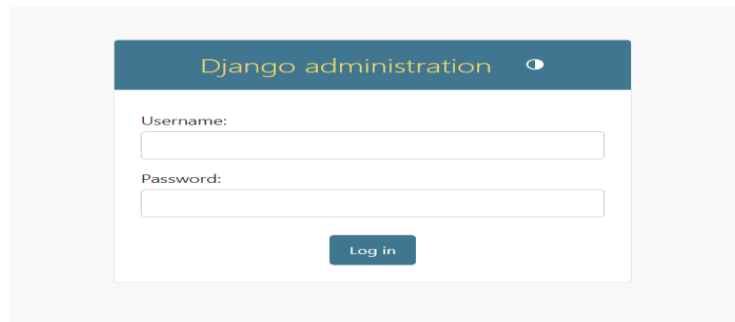
Để truy cập trang admin của Django, bạn chỉ cần thêm /admin vào địa chỉ URL, thường là **http://127.0.0.1:8000/admin** khi chạy trên localhost. Tuy nhiên, trước tiên bạn cần tạo tài khoản quản trị viên bằng lệnh **python manage.py createsuperuser**. Sau khi cung cấp tên người dùng, email và mật khẩu, bạn có thể đăng nhập vào giao diện quản trị và sử dụng Django Admin để quản lý ứng dụng của mình.



```
PS D:\python\myproject> python manage.py createsuperuser
Username (leave blank to use 'lamth'): thdinh
Email address: thdinh@gmail.com
Password:
Password (again):
Superuser created successfully.
```

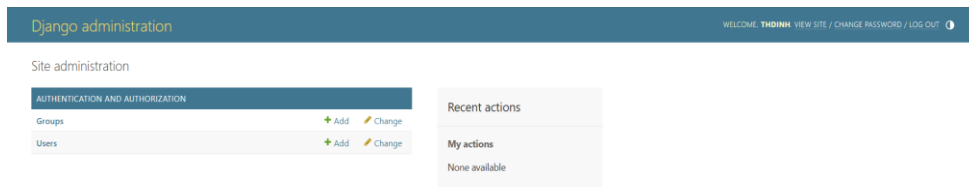
Hình 2.16 Tạo tài khoản quyền superuser

Sau khi đã tạo thành công tài khoản, truy cập vào ngay trang quản trị admin, đăng nhập tài khoản vừa tạo.



Hình 2.17 Giao diện đăng nhập tài khoản superser

Đăng nhập với tài khoản đã tạo, sẽ hiện ra giao diện của trang admin như sau:



Hình 2.18 Giao diện trang chủ quản trị viên

## 2.3. MySQL

### 2.3.1 Khái quát về MySQL

MySQL là một hệ thống quản lý cơ sở dữ liệu quan hệ mã nguồn mở (Relational Database Management System - RDBMS) dựa trên ngôn ngữ truy vấn có cấu trúc (SQL) được phát triển, phân phối và hỗ trợ bởi tập đoàn Oracle. MySQL chạy trên hầu hết tất cả các nền tảng, bao gồm cả Linux , UNIX và Windows. MySQL thường được kết hợp với các ứng dụng web.

Để quản lý cơ sở dữ liệu của MySQL trong đề tài này, em sử dụng công cụ MySQL Workbench. Công cụ này sẽ cung cấp giao diện người dùng trực quan, hỗ trợ người dùng tương tác với hệ quản trị cơ sở dữ liệu MySQL một cách mạnh mẽ. Nó cung cấp các công cụ để thiết kế cấu trúc dữ liệu, quản lý các đối tượng cơ sở dữ liệu, tối ưu hiệu suất truy vấn và thực thi các tác vụ quản trị một cách thuận tiện và hiệu quả, thay vì phải sử dụng dòng lệnh phức tạp.

### 2.3.2 Ưu và nhược điểm của MySQL

Ưu điểm:

Dễ dàng sử dụng: là hệ thống khá dễ sử dụng, khiến nó trở thành sự chọn lựa khá phổ biến dành cho các nhà phát triển ở tất cả cấp độ kỹ năng. Sử dụng ngôn ngữ truy vấn đơn giản sẽ giúp dễ dàng truy xuất và thao tác dữ liệu.

Khả năng tương thích rộng: MySQL tương thích với nhiều nền tảng, bao gồm Windows, Linux và macOS. Điều này có nghĩa là nó có thể được sử dụng trên nhiều loại thiết bị, từ máy tính để bàn đến thiết bị di động.

Hiệu suất cao: được thiết kế cho hiệu suất cao, có nghĩa là nó có thể xử lý lượng lớn dữ liệu và lưu lượng truy cập mà không bị chậm. Nó sử dụng các kỹ thuật lập chỉ mục nâng cao và cơ chế lưu vào bộ nhớ đệm để tối ưu hóa hiệu suất

Các tính năng bảo mật mạnh mẽ: cung cấp các tính năng bảo mật mạnh mẽ, bao gồm xác thực người dùng, mã hóa và kiểm soát truy cập. Điều này có nghĩa là dữ liệu nhạy cảm có thể được bảo vệ khỏi truy cập trái phép, giúp ngăn chặn vi phạm dữ liệu và các sự cố bảo mật khác.

Hỗ trợ từ cộng đồng: có một cộng đồng lớn và tích cực bao gồm các nhà phát triển và người dùng, những người đóng góp cho sự phát triển của nó và cung cấp hỗ trợ cho những người khác sử dụng nó.

Nhược điểm:

Hạn chế dung lượng: Trong trường hợp số lượng bản ghi ngày càng gia tăng, quá trình truy xuất dữ liệu sẽ trở nên khó khăn. Để giải quyết vấn đề này, bạn có thể

áp dụng nhiều biện pháp như tạo cache MySQL hoặc phân tải cơ sở dữ liệu ra nhiều máy chủ.

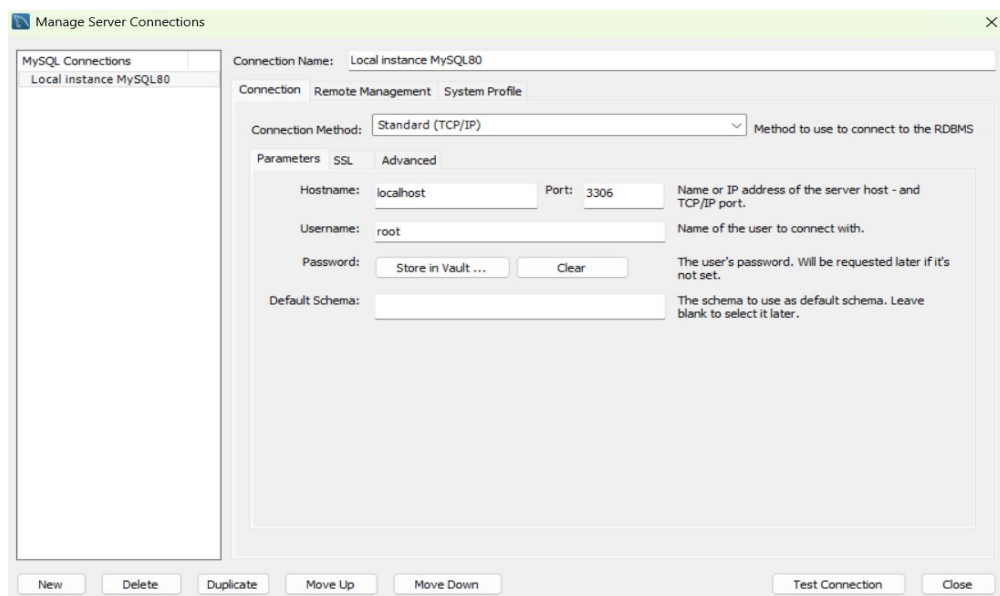
**Độ tin cậy thấp:** Một nhược điểm khác của MySQL là trong quá trình thực hiện các chức năng cụ thể như kiểm toán, giao dịch và quản lý tài liệu tham khảo, MySQL có thể trở nên kém tin cậy hơn một số hệ quản trị cơ sở dữ liệu quan hệ khác.

**Giới hạn chức năng:** MySQL không thiết kế để thực hiện toàn bộ các chức năng và nó đi kèm với những hạn chế về chức năng mà một số ứng dụng có thể cần.

### 2.3.3 Kết nối MySQL

Để kết nối MySQL với Django, trước tiên ta cần phải cài đặt MySQL client và MySQL server nếu chưa có, để Django có thể giao tiếp với MySQL. Để cài đặt ta gõ lệnh như sau: `pip install mysqlclient` và `pip install PyMySQL`.

Sau đó, tải MySQL Workbench và MySQL Installer Web Community. Cả 2 phải cùng một phiên bản ( ví dụ 8.0.40 ) hoặc chênh lệch nhau 1 đến 2 phiên bản. Khi đã cài thành công, tạo một kết nối tới MySQL.



Hình 2.19 Tạo kết nối MySQL

Để thiết lập kết nối giữa ứng dụng Django và cơ sở dữ liệu MySQL, bước đầu tiên là tạo một cơ sở dữ liệu trên máy chủ MySQL. Bạn có thể thực hiện việc này

bằng lệnh `CREATE DATABASE (tên database);`. Ví dụ, để tạo một cơ sở dữ liệu có tên `mydatabase`, bạn sẽ sử dụng lệnh `CREATE DATABASE ql_diemdoan;`.

Sau khi cơ sở dữ liệu được tạo, bạn cần cấu hình Django để kết nối với nó. Điều này được thực hiện bằng cách chỉnh sửa tệp `settings.py` trong thư mục dự án Django của bạn. Trong tệp này, bạn sẽ cần cung cấp thông tin kết nối đến cơ sở dữ liệu MySQL vừa tạo.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql', # Sử dụng MySQL backend
        'NAME': 'ql_diemdoan', # Tên cơ sở dữ liệu MySQL
        'USER': 'root', # Tên người dùng MySQL
        'PASSWORD': '12345678', # Mật khẩu MySQL
        'HOST': '127.0.0.1', # Địa chỉ máy chủ (có thể là localhost hoặc IP)
        'PORT': '3306', # Cổng
    }
}
```

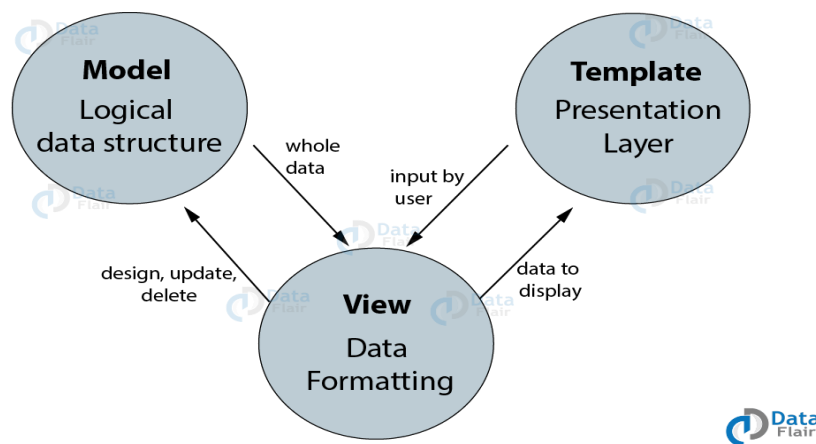
Hình 2.20 Điều chỉnh trong `settings.py`

Để hoàn tất, hãy chạy lệnh `migrate` để áp dụng các thay đổi vào cơ sở dữ liệu và sau đó chạy `runserver`. Nếu quá trình `runserver` không gặp lỗi nào trong terminal, điều đó xác nhận kết nối đến cơ sở dữ liệu đã thành công.

## 2.4. Mô hình Model-View-Template

### 2.4.1 Sơ lược về mô hình

Mô hình MTV (Model-Template-View): Django sử dụng mô hình này để phân chia logic dữ liệu, giao diện người dùng và logic xử lý trong ứng dụng web. Bộ công cụ tích hợp: Django cung cấp các công cụ tích hợp mạnh mẽ cho việc xử lý đăng nhập người dùng, quản lý admin, tự động tạo giao diện quản trị và nhiều tính năng khác.



Hình 2.21 Mô hình MVT

Model (M): trong Django được xây dựng trên ORM (Object-Relational Mapping), cho phép lập trình viên tương tác với cơ sở dữ liệu thông qua các đối tượng Python thay vì viết trực tiếp các truy vấn SQL. Điều này giúp cho việc phát triển ứng dụng trở nên dễ dàng và hiệu quả hơn, đồng thời cũng giảm thiểu nguy cơ phát sinh lỗi từ các truy vấn SQL phức tạp.

View (V): trong Django, không giống như controller trong MVC, không chỉ xử lý logic mà còn chịu trách nhiệm phản hồi lại người dùng. Các view có thể là function-based hoặc class-based, tùy thuộc vào yêu cầu và phong cách lập trình của từng nhà phát triển. View lấy dữ liệu từ model, xử lý chúng và gửi chúng đến template để hiển thị.

Template (T): là lớp cuối cùng trong mô hình MVT, nơi định nghĩa cách dữ liệu được trình bày. Templates sử dụng ngôn ngữ đánh dấu (thường là HTML) và có khả năng nhúng mã Python để tạo ra các trang web động. Cơ chế kế thừa template của Django cung cấp khả năng tái sử dụng và quản lý mã hiệu quả, giúp duy trì sự nhất quán trong giao diện người dùng trên toàn bộ ứng dụng.

Bảng 1 So sánh giữa MVC và MVT

Mô hình MVC	Mô hình MVT	Chức năng
Model	Model	Làm việc với cơ sở dữ liệu

View	Template	Hiển thị giao diện
Controller	View	Xử lý logic

#### 2.4.2 Ưu và nhược điểm của mô hình

Ưu điểm:

Cấu trúc phân lớp mạch lạc: Việc phân chia ứng dụng thành ba lớp riêng biệt – Model, View và Template – tạo ra một cấu trúc rõ ràng và dễ quản lý. Mỗi lớp đảm nhận một nhiệm vụ cụ thể, giúp lập trình viên dễ dàng hiểu, sửa đổi và bảo trì từng phần của ứng dụng mà không bị rối bởi sự phức tạp của toàn bộ hệ thống.

Quản lý dữ liệu hiệu quả: Model sẽ tương tác với cơ sở dữ liệu trở nên đơn giản như làm việc với các đối tượng Python. Bạn không cần phải viết SQL, nhờ đó giảm thiểu lỗi và tăng tốc độ phát triển.

Khả năng mở rộng và bảo trì linh hoạt: Khi cần thêm chức năng mới hoặc sửa đổi chức năng hiện có, chỉ cần tập trung vào lớp liên quan mà không cần phải lo lắng về việc ảnh hưởng đến các phần khác.

Tối ưu hóa khả năng tái sử dụng mã: View và Template được thiết kế để có thể tái sử dụng trong nhiều phần khác nhau. Điều này giúp giảm thiểu sự trùng lặp code, tiết kiệm thời gian phát triển và đảm bảo tính nhất quán.

Bảo mật được ưu tiên hàng đầu: Django được tích hợp nhiều tính năng bảo mật mạnh mẽ, giúp bảo vệ ứng dụng khỏi các cuộc tấn công phổ biến như CSRF.

Nhược điểm:

Độ khó trong việc tùy biến giao diện phức tạp: Mặc dù Django cung cấp hệ thống template mạnh mẽ, việc tùy chỉnh giao diện người dùng phức tạp, đặc biệt là khi yêu cầu hiệu ứng động và tương tác người dùng cao.

**Khả năng tách biệt:** Việc tách biệt các thành phần yêu cầu khả năng hiểu biết chuyên sâu về cả 3 lĩnh vực (Model-Views-Template) để có thể hiệu quả.

**Hạn chế trong việc xây dựng giao diện người dùng động(SPA):** Django chủ yếu tập trung vào việc xử lý phía server và không được thiết kế tối ưu cho việc xây dựng các ứng dụng một trang với giao diện người dùng động và tương tác cao.

**Xử lý logic:** Việc đồng bộ và xử lý các logic cũng trở nên phức tạp và khó khăn hơn trong quá trình phát triển.

## **CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU**

### **3.1. Mô tả bài toán**

Bài toán xây dựng phần mềm quản lý điểm đồ án cho sinh viên khoa KT&CN Trường Đại học Trà Vinh xuất phát từ nhu cầu số hóa quy trình quản lý điểm, thay thế các phương pháp thủ công bằng một công cụ hiệu quả, chính xác và minh bạch. Hệ thống cần có các chức năng cơ bản như quản lý giảng viên (tên, chuyên ngành, đề tài hướng dẫn), quản lý thông tin sinh viên (mã số, họ tên, lớp, đề tài đã chọn) và quản lý điểm số (điểm thi lần 1, lần 2, điểm trung bình). Giảng viên cần có khả năng nhập, chỉnh sửa và xem điểm, cũng như xem danh sách sinh viên theo từng đồ án. Hệ thống cũng cần cung cấp báo cáo tổng kết điểm, thống kê số lượng sinh viên đã đăng ký các đồ án và các chỉ số khác hỗ trợ đánh giá kết quả học tập. Bên cạnh đó, quản lý người dùng và phân quyền là yếu tố quan trọng, phân biệt quyền hạn giữa giảng viên (nhập/sửa điểm, xem báo cáo), cán bộ quản lý khoa (xem toàn bộ dữ liệu). Mục tiêu là cung cấp trải nghiệm quản lý điểm dễ dàng, nhanh chóng và chính xác cho giảng viên, đồng thời hỗ trợ cán bộ quản lý khoa theo dõi hiệu quả quá trình đào tạo và giúp sinh viên dễ dàng theo dõi kết quả học tập.

### **3.2. Đặc tả yêu cầu**

#### **3.2.1 Yêu cầu chức năng**

Người quản trị (Admin):

Quản lý sinh viên: Có thể thêm mới sinh viên (bao gồm các thông tin như mã số sinh viên, họ tên, mã lớp, ...), chỉnh sửa thông tin sinh viên đã có và xóa sinh viên.

Quản lý giảng viên: Tương tự như quản lý sinh viên, có thể thêm mới giảng viên (bao gồm các thông tin như họ tên, chuyên ngành, đề tài hướng dẫn,...), chỉnh sửa thông tin giảng viên đã có và xóa giảng viên.

Quản lý kết quả: Có thể thêm, sửa, xóa sinh viên đã có kết quả, tự động tính điểm trung bình (điểm lần 1 và điểm lần 2).



Báo cáo thống kê: Chỉ có người quản trị mới xem được số lượng sinh viên đã đăng ký, số lượng sinh viên đã có kết quả và số lượng sinh viên trên 4.0.

Phân quyền: Phân quyền rõ ràng giữa người quản trị và giảng viên, đảm bảo mỗi người truy cập đều có thể thao tác với các tính năng phù hợp.

Người dùng (Giảng viên):

Đăng ký và đăng nhập: Giảng viên có thể đăng ký tài khoản cho cá nhân, đăng nhập để xem, thêm, chỉnh sửa hoặc xóa đề tài của mình, thông tin của sinh viên, nhập điểm cho sinh viên.

Thêm thông tin: Giảng viên có thể thêm thông tin và đề tài của bản thân, thay đổi đề tài hoặc xóa. Trang quản lý sẽ tự động cập nhật ngay sau khi có sự thay đổi từ phía giảng viên.

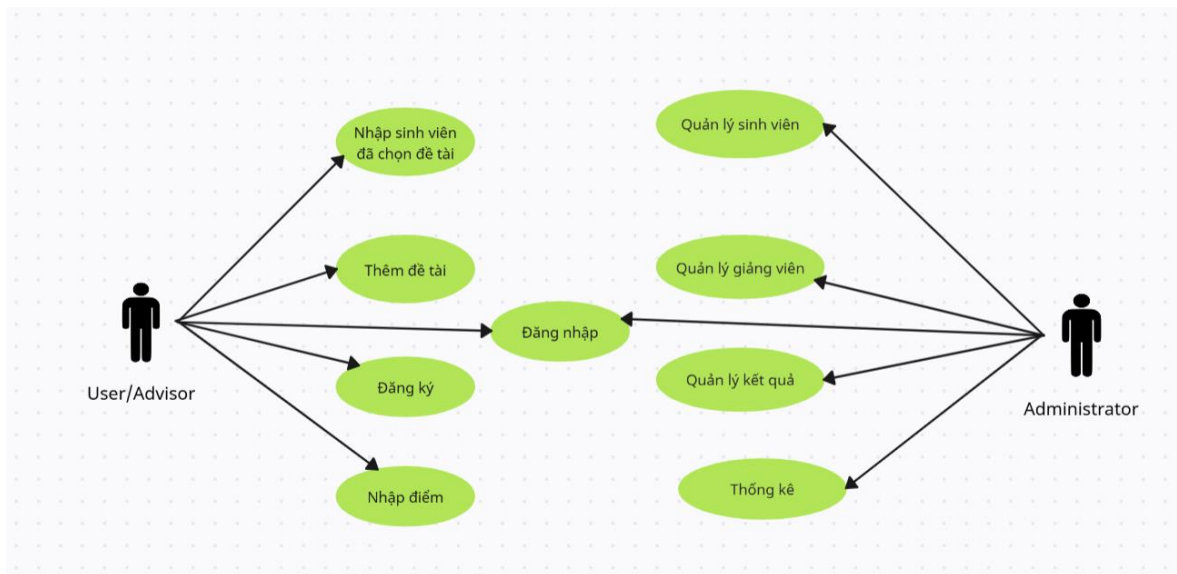
Xem và tìm kiếm thông tin: Giảng viên có tìm kiếm thông tin sinh viên bằng cách ghi mã số sinh viên hoặc họ tên của sinh viên trên thanh tìm kiếm.

### **3.2.2 Yêu cầu phi chức năng**

Khi nói về một phần mềm quản lý điểm đồ án, trước hết cần đề cập đến giao diện người dùng. Giao diện phải thân thiện, rõ ràng và dễ dàng sử dụng, đảm bảo các thông tin được hiển thị một cách logic và trực quan. Màu sắc nên được lựa chọn sao cho hài hòa, tránh sự rối mắt, đồng thời tạo cảm giác chuyên nghiệp và hiện đại.

Chức năng của phần mềm cần phải dễ dàng sử dụng, phần mềm cần tập trung vào việc tối ưu hóa các tính năng cơ bản như nhập điểm, tra cứu thông tin, và tổng hợp báo cáo. Không nên quá phức tạp sẽ gây khó khăn cho người dùng.

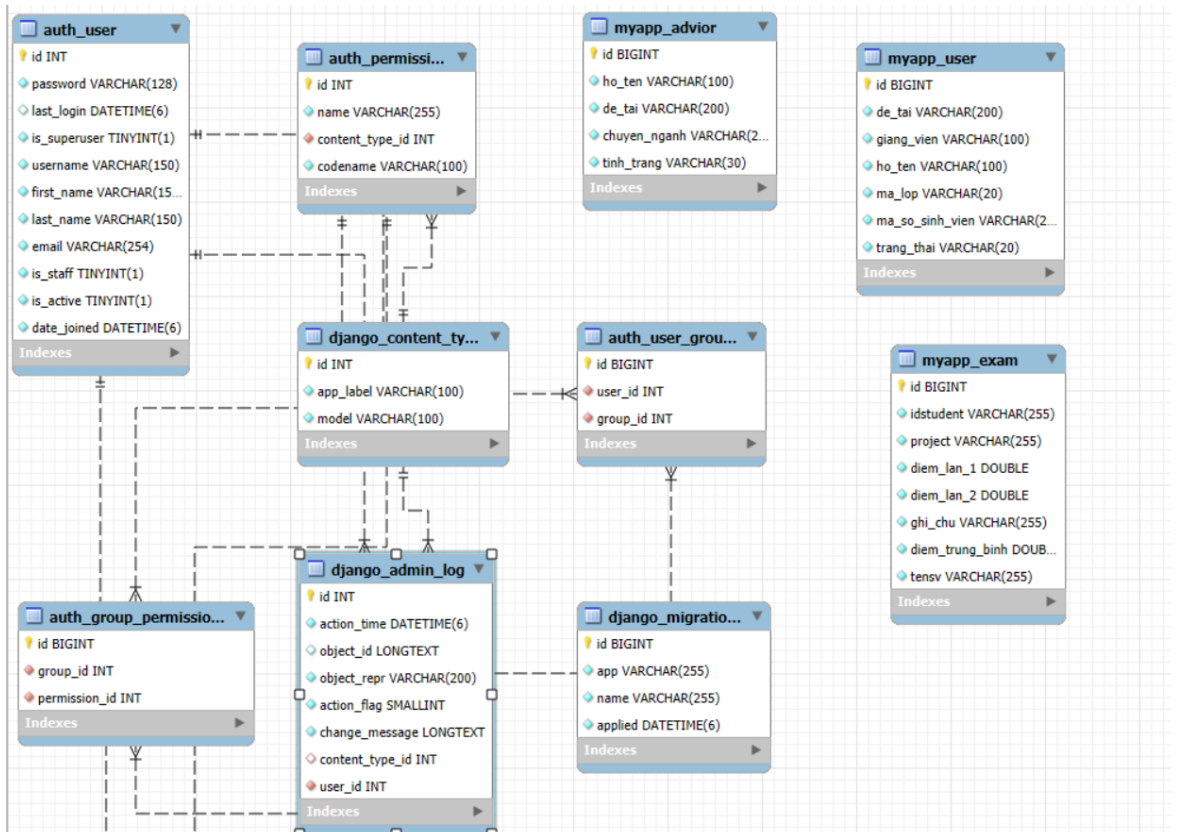
### 3.2.3 Sơ đồ Use-case



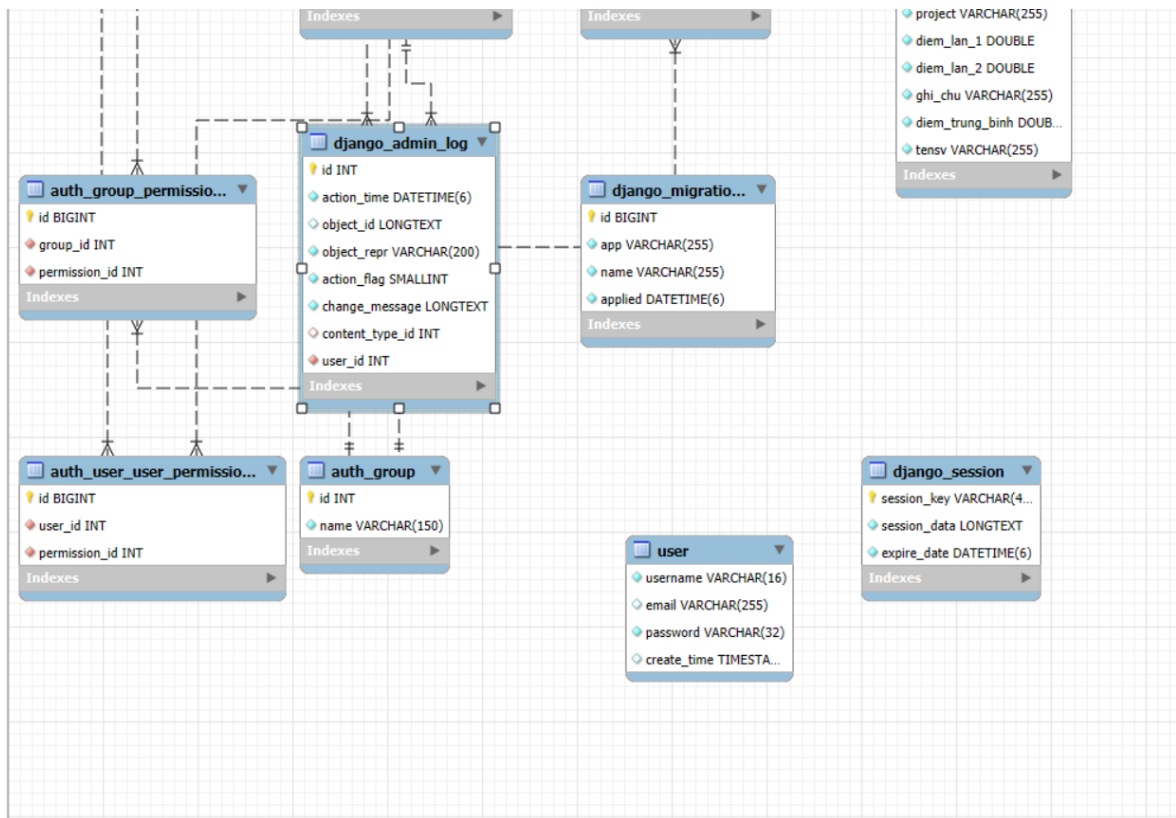
Hình 3.1 Sơ đồ Use-case

### 3.3. Cơ sở dữ liệu trong MySQL Workbench

#### 3.3.1 Mô hình MySQL



Hình 3.2 Mô hình MySQL



Hình 3.3 Mô hình MySQL (tt)

### 3.3.2 Bảng thực thể

Bảng 2 Auth\_User

STT	Thuộc tính	Mô tả	Kiểu dữ liệu
1	id	Mã người dùng	int
2	password	Mật khẩu	varchar
3	last_login	Thời gian đăng nhập lần cuối	datetime
4	is_superuser	Xác định người quản trị (có :1, không :0 )	tinyint
5	username	Tên đăng nhập	varchar

6	first_name	Tên người dùng	varchar
7	last_name	Họ người dùng	varchar
8	email	Email người dùng	varchar
9	is_staff	Xác định người dùng có phải giảng viên (1: có, 0: không )	tinyint
10	is_active	Xác định người dùng có còn hoạt động không (1: có, 0: không)	tinyint
11	data_joined	Thời gian người dùng tham gia hệ thống	datetime

Bảng 3 User(Sinh viên)

STT	Thuộc tính	Mô tả	Kiểu dữ liệu
1	id	Mã đại diện sinh viên	int
2	de_tai	Đề tài đăng ký	varchar
3	giang_vien	Giảng viên hướng dẫn	varchar

4	ho_ten	Họ tên sinh viên	varchar
5	ma_lop	Mã lớp	varchar
6	ma_so_sinh_vien	Mã số sinh viên	varchar
7	trang_thai	Trạng thái sinh viên đã đăng ký	varchar

Bảng 4 Adivior (Giảng viên)

STT	Thuộc tính	Mô tả	Kiểu dữ liệu
1	id	Mã đại diện giảng viên	int
2	ho_ten	Họ tên giảng viên	varchar
3	de_tai	Đề tài giảng viên đưa ra	varchar
4	chuyen_nganh	Chuyên ngành giảng viên	varchar
5	ting_trang	Tình trạng nhận sinh viên đăng ký đề tài	varchar

Bảng 5 Exam (Kết quả)

STT	Thuộc tính	Mô tả	Kiểu dữ liệu
1	id	Mã đại diện kết quả	int
2	idstudent	Mã số sinh viên	varchar

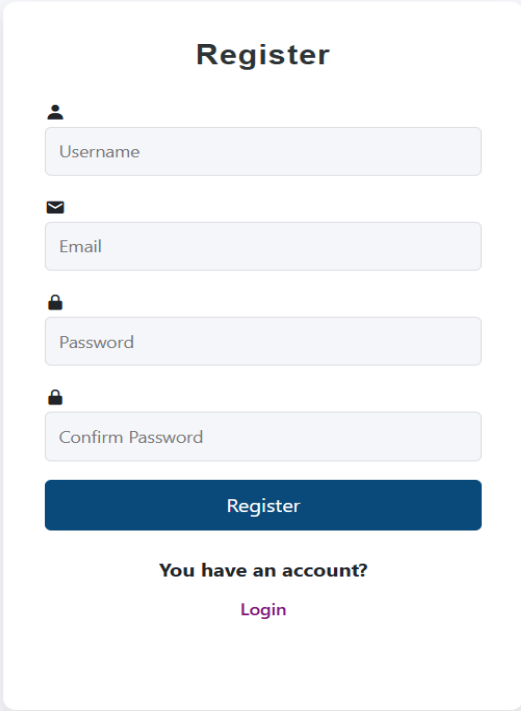
3	project	Đề tài của sinh viên	varchar
4	diem_lan_1	Điểm thi lần 1	double
5	diem_lan_2	Điểm thi lần 2 (Nếu trượt lần 1)	double
6	ghi_chu	Đánh giá kết quả học tập, theo dõi sinh viên	varchar
7	diem_trung_binh	Điểm trung bình (Cả 2 lần thi)	varchar
8	tensv	Tên của sinh viên	varchar

## CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU

### 4.1. Giao diện người dùng

#### 4.1.1 Giao diện trang đăng ký

Khi người dùng truy cập vào trang web, bắt buộc phải tạo một tài khoản cá nhân. Người dùng sẽ phải nhập thông tin như tên người dùng, email, mật khẩu. Sau khi đã nhập đầy đủ thông tin, nhấn Register để tiến hành đăng ký. Khi đăng ký thành công sẽ lập tức chuyển sang đăng nhập.

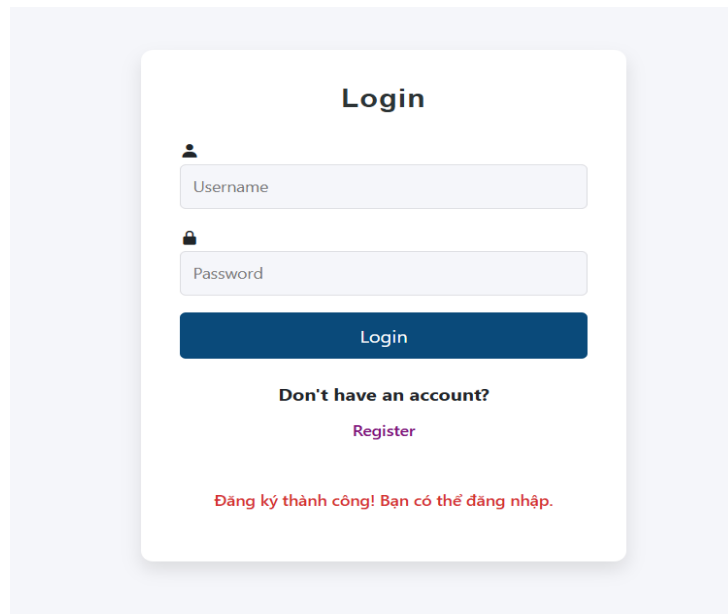


The image shows a registration form titled "Register". It contains four input fields: "Username" (with a person icon), "Email" (with an envelope icon), "Password" (with a lock icon), and "Confirm Password" (with a lock icon). Below these fields is a dark blue "Register" button. Under the button, it says "You have an account?" followed by a purple "Login" link.

Hình 4.1 Giao diện trang đăng ký

#### 4.1.2 Giao diện trang đăng nhập

Ngay sau khi đăng ký thành công, người dùng sẽ nhận được thông báo trong form, tại đây người dùng sẽ đăng nhập bằng thông tin vừa đăng ký để có thể tiến hành thực hiện các chức năng thêm, sửa, xóa.

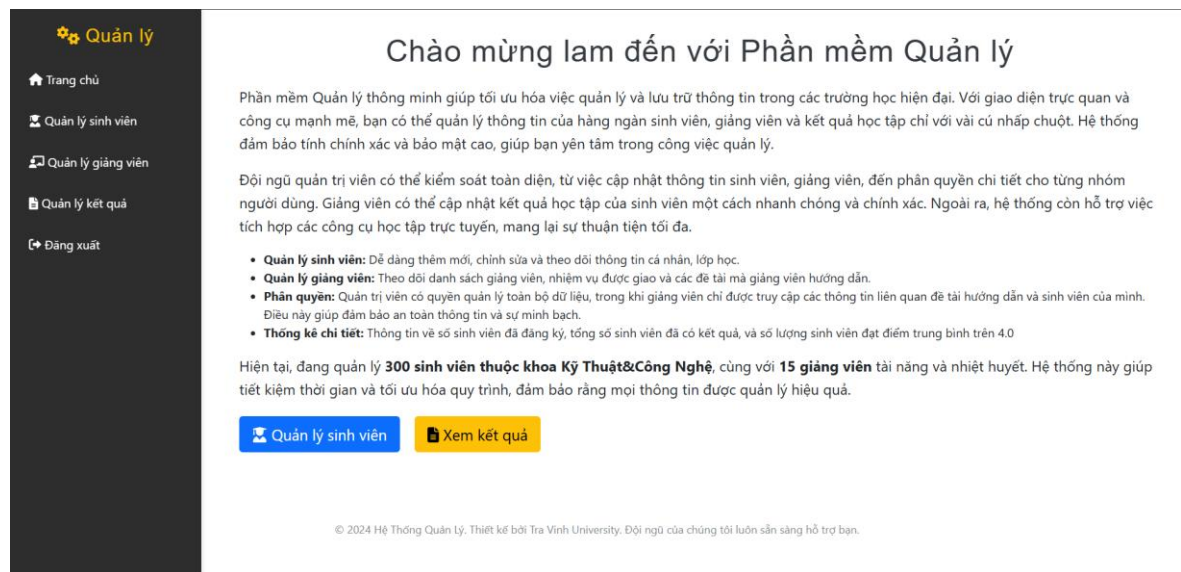


Hình 4.2 Giao diện trang đăng nhập

#### 4.1.3 Giao diện trang chủ

Trang chủ sẽ là trang hiện ra đầu tiên ngay sau khi đăng nhập thành công với quyền giảng viên. Giao diện của trang chủ bao gồm sidebar, các danh mục quản lý. Ở đây khi đăng nhập với tư cách là giảng viên, sẽ không thể thấy danh mục thống kê, chỉ khi đăng nhập với tư cách là quản trị viên thì mới thấy được.





Hình 4.3 Giao diện trang chủ khi đăng nhập với tư cách là giảng viên

#### 4.1.4 Giao diện quản lý sinh viên

Trên giao diện quản lý sinh viên, giảng viên có thể dễ dàng tra cứu danh sách sinh viên đã đăng ký đề tài. Chức năng như tìm kiếm sinh viên bằng mã số hoặc tên, giúp giảng viên dễ dàng tìm ra sinh viên cần quản lý. Bên cạnh đó, giảng viên cũng có thể chỉnh sửa thông tin sinh viên và theo dõi tiến độ thực hiện đề tài.

**Thêm mới**

Mã số sinh viên:

Họ tên:

Mã lớp:

Đề tài:

Giảng viên:

Trạng thái:

**Thêm mới**

**Danh sách**

Tìm kiếm theo MSSV hoặc họ tên... **Tìm**

#	Mã số sinh viên	Họ tên	Mã lớp	Đề tài	Giảng viên	Trạng thái	Hành động
1	101	Trần Quốc Toán	DA22TTA	Tìm hiểu Python	Trần Quốc Toán	Đã hoàn thành	<a href="#">Sửa</a> <a href="#">Xóa</a>
2	123	Đạt	da	Python123	Trần Quốc	Đã đăng ký	<a href="#">Sửa</a> <a href="#">Xóa</a>

Hình 4.4 Giao diện quản lý sinh viên

#### 4.1.5 Giao diện quản lý giảng viên

Tại giao diện quản lý giảng viên, giảng viên và người quản lý có thể cập nhật thông tin cơ bản của giảng viên như họ tên, chuyên ngành, đề tài hướng dẫn, trạng thái nhận đề tài. Chức năng tìm kiếm giảng viên cũng tương tự như quản lý sinh viên

**Thêm mới**

Họ tên:

Chuyên ngành:

Đề tài:

Tình trạng:

**Thêm mới**

**Danh sách**

Tìm giảng viên có tên là ... **Tìm**

#	Họ tên	Chuyên ngành	Đề tài	Tình trạng	Hành động
---	--------	--------------	--------	------------	-----------

Hình 4.5 Giao diện quản lý giảng viên

#### 4.1.6 Giao diện quản lý kết quả

Ở giao diện quản lý kết quả, người quản trị và giảng viên có thể theo dõi và cập nhật kết quả của sinh viên khi đã hoàn thành đồ án, bao gồm điểm lần 1 và điểm lần 2, tổng điểm trung bình. Chức năng tìm kiếm cũng sẽ tương tự như quản lý sinh viên

và giảng viên. Bên cạnh đó, thêm ghi chú cho những trường hợp đặc biệt như sinh viên cần được theo dõi thêm.

**Thêm mới**

MSSV:

Họ tên sinh viên:

Đề án đề tài:

Điểm lần 1:

Điểm lần 2:

Điểm trung bình:

Ghi chú:

**Thêm mới**

**Danh sách kết quả thi**

Tim kiếm sinh viên có mã số hoặc tên l **Tim**

#	Mã số sinh viên	Họ tên	Đề án	Điểm lần 1	Điểm lần 2	Điểm trung bình	Ghi chú	Hành động
1	101	Trần Quốc Toàn	Tìm hiểu Python	0.0	0.0	0.0		<a href="#">Sửa</a> <a href="#">Xóa</a>

Hình 4.6 Giao diện quản lý kết quả

## 4.2. Giao diện quản trị viên

Để truy cập vào trang quản trị viên, người dùng có thể thực hiện bằng cách thêm **/admin** vào cuối URL của trang web, chẳng hạn **http://127.0.0.1:8000/admin**. Nếu sử dụng tài khoản với quyền superuser, sau khi đăng nhập, sẽ tự động chuyển đến trang quản trị mà không cần thao tác thủ công.

### 4.2.1 Trang chủ quản trị viên

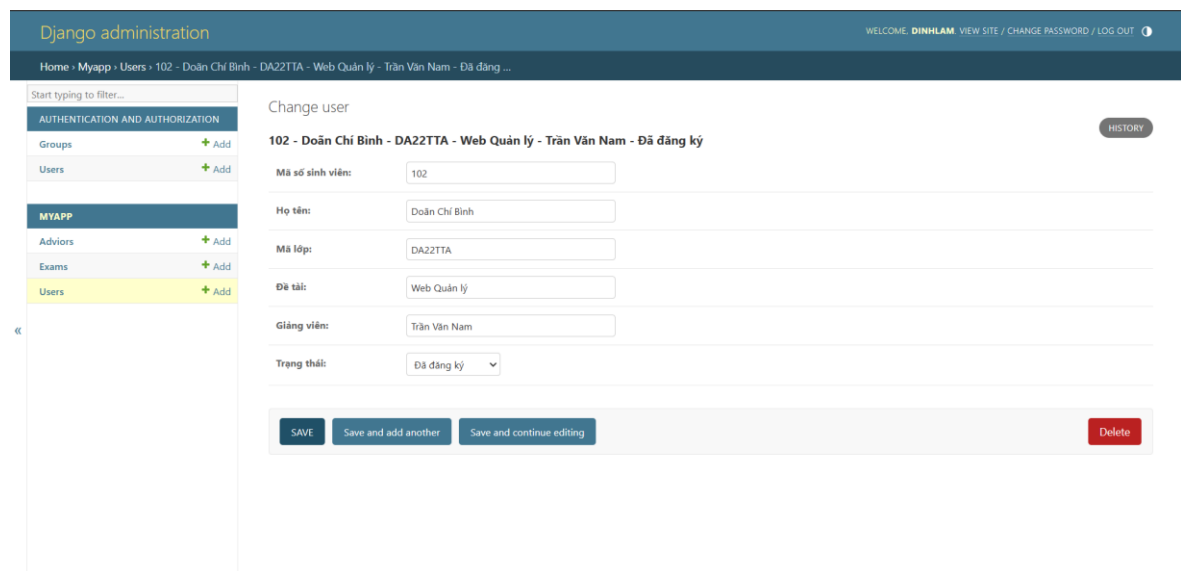
Ngay sau khi truy cập vào trang quản trị, giao diện đầu tiên hiển thị danh sách các models đã được tạo, bao gồm: User(Sinh viên), Advior(Giảng viên), Exam(Kết quả). Ở trên User và Group được Django tự động sinh ra. Ở phía bên phải, đây là nơi hiện ra các hành động của quản trị viên gần nhất.



Hình 4.7 Giao diện trang chủ quản trị viên

#### 4.2.2 Chức năng thêm, sửa, xóa

Người quản trị có thể thêm mới hoặc chỉnh sửa thông tin chi tiết của một sinh viên bao gồm tên, mã số, lớp học và các thông tin liên quan khác. Nếu cần xóa, chỉ cần chọn **Delete** để loại bỏ sinh viên khỏi hệ thống. Khi một sinh viên bị xóa, các dữ liệu liên quan đến sinh viên đó, như đăng ký đề tài hay kết quả học tập, cũng sẽ được tự động xóa để đảm bảo tính nhất quán trong hệ thống.



Hình 4.8 Giao diện của chức năng thêm, sửa, xóa

#### 4.2.3 Quản lý người dùng

Người quản trị có thể thêm mới chỉnh sửa, hoặc xóa thông tin chi tiết của người dùng.

Select user to change

Q  Search

Action:  Go 0 of 4 selected

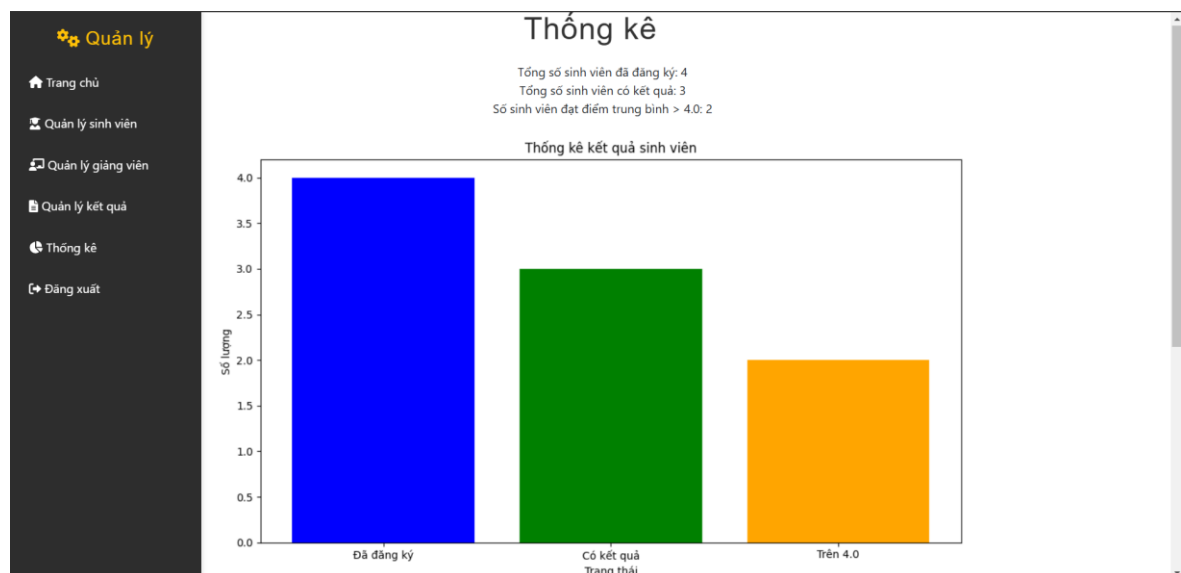
<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	a	aaaa@gmail.com	-	-	✗
<input type="checkbox"/>	dinhlam	dinhlam@gmail.com	-	-	✓
<input type="checkbox"/>	lam	lam@gmail.com	-	-	✗
<input type="checkbox"/>	thdinh	123@gmail.com	-	-	✗

4 users

Hình 4.9 Giao diện quản lý người dùng

#### 4.2.4 Trang báo cáo thống kê

Mục “Báo cáo thống kê” chỉ hiển thị trên thanh sidebar khi đăng nhập bằng tài khoản quản trị viên. Tại đây, người quản trị viên có thể xem các thông tin quan trọng như tổng số sinh viên đã đăng ký, số sinh viên đã có kết quả, số lượng sinh viên có điểm trung bình trên 4.0. Ngoài ra, trang cũng cung cấp lại bảng kết quả từ quản lý kết quả để người quản trị có thể dễ dàng quản lý.



Hình 4.10 Giao diện trang báo cáo thống kê

## CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1. Kết luận

Đề tài “Tìm hiểu ngôn ngữ lập trình Python. Xây dựng phần mềm quản lý điểm đồ án, đề tài của sinh viên khoa KT&CN- Trường Đại học Trà Vinh” đã xây dựng thành công phần mềm quản lý điểm đồ án, đề tài của sinh viên với ngôn ngữ Python và framework Django. Phần mềm hỗ trợ được các yêu cầu cơ bản cho người dùng và quản trị viên, bao gồm quản lý sinh viên, quản lý giảng viên, nhập điểm và xem điểm cũng như tạo ra được báo cáo thống kê. Giao diện thân thiện, người dùng dễ dàng sử dụng, cấu trúc tuân theo mô hình MVT của Django giúp cho phần mềm dễ nâng cấp và bảo trì.

Quá trình thực hiện đề tài đã giúp em có cơ hội làm quen và sử dụng Django để phát triển backend, tương tác với cơ sở dữ liệu, xử lý các yêu cầu HTTP và xây dựng giao diện cho người dùng bằng Django Templates. Nhờ đó mà hiểu rõ hơn về quy trình phát triển ứng dụng, từ việc thiết kế đến triển khai và vận hành.

### 5.2. Hướng phát triển

Hiện tại, giảng viên chỉ có thể thêm, sửa, xóa, kiểm soát thông tin sinh viên đã đăng ký đề tài, nhập điểm nên cần phát triển thêm số lượng sinh viên đăng ký đề tài, quản lý quy trình duyệt đề tài do sinh viên đề xuất hoặc sinh viên đăng ký. Theo dõi tiến độ làm đồ án của sinh viên.

Phân quyền chỉ mới có người dùng là giảng viên và người quản trị, cần phát triển thêm phân quyền cho sinh viên để sinh viên có thể theo dõi kết quả của mình.

Phát triển hệ thống đánh giá tiêu chí cho từng loại đồ án như đánh giá giữa kỳ, cuối kỳ, đánh giá của giảng viên hướng dẫn hoặc giảng viên phản biện.

Giảng viên chưa thể chỉnh sửa thông tin cá nhân. Việc này sẽ làm giảm khả năng tương tác và tùy chỉnh cho giảng viên. Cần thêm trang chỉnh sửa thông tin cá nhân và bảo mật mật khẩu.

Báo cáo thống kê còn cơ bản, cần thêm sự tăng giảm về số lượng sinh viên hoàn thành đồ án, điểm trung bình sinh viên đạt được năm nay so với năm vừa rồi.

Do trọng tâm chính của đề tài đặt vào việc xây dựng phần backend nên ở giao diện người dùng hiện tại còn ở mức cơ bản, chưa được đầu tư nhiều về mặt thẩm mỹ. Để giao diện trực quan, sinh động và hiện đại hơn, có thể sử dụng các framework CSS hiện đại như Bootstrap hoặc Tailwind CSS, giúp thân thiện, dễ sử dụng và nâng cao hiệu quả quản lý điểm đồ án.

## TÀI LIỆU THAM KHẢO

- [1] W3Schools, "Python Tutorial," [Online]. Available: <https://www.w3schools.com/python/>. [Accessed 19 11 2024].
- [2] W3School, "Django Tutorial," [Online]. Available: <https://www.w3schools.com/django/>. [Accessed 19 11 2024].
- [3] W3School, "MySQL Tutorial," [Online]. Available: <https://www.w3schools.com/mysql/default.asp>. [Accessed 19 11 2024].
- [4] V.Tiến, "Cài đặt MySQL Workbench," 4 1 2024. [Online]. Available: <https://vietnix.vn/mysql-workbench/>.
- [5] W3School, "Bootstrap," [Online]. Available: <https://www.w3schools.com/bootstrap5/index.php>. [Accessed 25 11 2024].
- [6] "Lịch sử của Python" Update:29 12 2024. [Online]. Available: <https://imic.edu.vn/tin-tuc-cong-nghe/31984/lich-su-phat-trien-cua-ngon-ngu-lap-trinh-python.html>.

