

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

XÂY DỰNG HỆ THỐNG BÁN GAME VÀ TRÒ CHƠI ĐIỆN TỬ

Giảng viên hướng dẫn:
ThS. Ngô Thanh Huy

Sinh viên thực hiện:
Họ tên: Nguyễn Nhựt Hóa
MSSV: 110122006
Lớp: DA22TTA

Trà Vinh, tháng 2 năm 2025

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

Xây dựng hệ thống bán game

Giảng viên hướng dẫn:
ThS. Ngô Thanh Huy

Sinh viên thực hiện:
Họ tên: Nguyễn Nhựt Hóa
MSSV: 110122006
Lớp: DA22TTA

Trà Vinh, tháng 2 năm 2025

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

This image shows a full page of primary-ruled paper. It features approximately 28 horizontal dotted lines spaced evenly down the page, providing a guide for handwriting practice. The paper is otherwise blank, with no margins, text, or other markings.

Trà Vinh, ngày tháng năm

Giáo viên hướng dẫn
(Ký tên và ghi rõ họ tên)

[illegible]

Thành viên hội đồng
(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Em xin chân thành gửi lời cảm ơn đến quý thầy cô trường Đại học Trà Vinh nói chung và các thầy cô bộ môn trong Khoa Kỹ thuật và Công nghệ nói riêng đã tạo điều kiện cho chúng em cơ hội thực hành, tiếp xúc để chúng em có thể tránh được những vướng mắc và bế ngõ trong môi trường công việc thời gian tới.

Em xin chân thành cảm ơn Thạc Sĩ Ngô Thanh Huy. Nhờ sự giúp đỡ tận tình và những chỉ bảo của Thầy từ lúc bắt đầu cho tới lúc kết thúc đồ án mà em đã hoàn thành đúng thời hạn quy định và tích lũy được cho mình một lượng nền tảng kiến thức đáng quý.

Mặc dù đã cố gắng hoàn thành đồ án cơ sở ngành một cách tốt nhất nhưng do thời gian và kiến thức còn hạn chế nên em vẫn còn nhiều thiếu sót khi tìm hiểu, đánh giá, và trình bày về đề tài. Rất mong nhận được sự quan tâm, góp ý của các thầy cô giảng viên bộ môn để đề tài 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!

Trà Vinh, ngày ... tháng ... năm

SINH VIÊN

Nguyễn Nhựt Hóa

MỤC LỤC

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH.....	9
MỞ ĐẦU	10
CHƯƠNG 1: TỔNG QUAN	12
CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT.....	14
2.1 Ngôn ngữ Python	14
2.1.1 Giới thiệu về Python.....	14
2.1.2 Ứng dụng.....	14
2.1.3 Tải và cài đặt Python	15
2.2 Framework Django	16
2.2.1 Khái quát về Django	16
2.2.2 Ưu và nhược điểm của Django	16
2.2.3 Các tính năng cơ bản	17
2.2.4 Cài đặt, tạo và chạy một dự án Django.....	17
2.2.5 Các thành phần chính có trong Django	20
2.3 MySQL.....	25
2.3.1 Giới thiệu	26
2.3.2 Một số thế mạnh của MySQL	26
2.3.3 Kết nối MySQL với Django.....	26
2.4 Mô hình MVT	27
2.4.1 Giới thiệu	27
2.4.2 Cách thức mô hình MVT hoạt động.....	28
2.4.3 Ưu và nhược điểm của mô hình MVT.....	29
CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU.....	30
3.1 Mô tả bài toán	30
3.2 Đặc tả yêu cầu hệ thống	30
3.2.1 Yêu cầu chức năng	30

3.2.2	Yêu cầu phi chức năng	31
3.2.3	Sơ đồ Use-case	31
3.3	Thiết kế cơ sở dữ liệu trong MySQL Workbench	32
3.3.1	Mô hình cơ sở dữ liệu	32
3.3.2	Bảng thực thể	32
CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU		36
4.1	Giao diện dành cho người dùng	36
4.1.1	Giao diện trang chủ	36
4.1.2	Giao diện trang đăng nhập	37
4.1.3	Giao diện trang đăng ký	38
4.1.4	Giao diện trang sản phẩm	38
4.1.5	Giao diện trang giỏ hàng	39
4.1.6	Giao diện trang kết quả tìm kiếm	39
4.1.7	Giao diện trang giới thiệu	40
4.2	Giao diện dành cho người quản trị	40
4.2.1	Trang chủ quản trị	40
4.2.2	Danh sách danh mục	41
4.2.3	Thêm, sửa, xóa sản phẩm	41
4.2.4	Quản lý người dùng	42
4.2.5	Quản lý các sản phẩm được đặt hàng	42
4.2.6	Quản lý các đơn hàng	43
4.2.7	Trang báo cáo thống kê	43
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN		44
5.1	Kết quả đạt được	44
5.2	Hạn chế và hướng phát triển	44
DANH MỤC TÀI LIỆU THAM KHẢO		45

DANH MỤC HÌNH ẢNH

Hình 2.1 Python phiên bản mới nhất	15
Hình 2.2 Kiểm tra cài đặt Python	15
Hình 2.3 Cài đặt django trong terminal Visual Studio Code	18
Hình 2.4 Lệnh kiểm tra phiên bản django đã cài đặt	18
Hình 2.5 Lệnh tạo một dự án django	18
Hình 2.6 Cấu trúc các tập tin của một dự án django cơ bản	18
Hình 2.7 Khai báo ứng dụng đã tạo vào project	19
Hình 2.8 Cấu hình lại hệ thống	19
Hình 2.9 Dự án đã hoạt động thành công	20
Hình 2.10 Ví dụ về hàm xử lý trong file views.py	20
Hình 2.11 Ví dụ minh họa cấu hình đường dẫn trong urls.py	21
Hình 2.12 Định nghĩa một tag block trong trang cơ sở	21
Hình 2.13 Template kế thừa từ trang cơ sở	22
Hình 2.14 Tạo model Article trong file models.py	23
Hình 2.15 Tạo form có sẵn từ module django.forms và hàm xử lý dữ liệu của form	23
Hình 2.16 Truyền form vào template	24
Hình 2.17 Giao diện form trên website	24
Hình 2.18 Tạo superuser của django	25
Hình 2.19 Đăng nhập vào trang quản trị	25
Hình 2.20 Giao diện trang quản trị của django	25
Hình 2.21 Tạo MySQL Connections	27
Hình 2.22 Kết nối MySQL với Django	27
Hình 2.23 Cấu trúc mô hình MVT	28
Hình 3.1 Sơ đồ use-case phân quyền chức năng	31
Hình 3.2 Sơ đồ cơ sở dữ liệu trong MySQL Workbench	32

Hình 4.1 Thanh menu và banner	36
Hình 4.2 Giao diện nội dung trang chủ	37
Hình 4.3 Footer	37
Hình 4.4 Giao diện trang đăng nhập	37
Hình 4.5 Giao diện trang đăng ký	38
Hình 4.6 Giao diện trang sản phẩm theo danh mục	39
Hình 4.7 Giao diện trang giỏ hàng	39
Hình 4.8 Giao diện trang kết quả tìm kiếm sản phẩm	40
Hình 4.9 Giao diện trang giới thiệu website	40
Hình 4.10 Giao diện trang chủ quản trị	41
Hình 4.11 Giao diện quản lý danh mục	41
Hình 4.12 Giao diện quản lý sản phẩm	42
Hình 4.13 Giao diện quản lý người dùng	42
Hình 4.14 Giao diện quản lý giỏ hàng	42
Hình 4.15 Giao diện quản lý đơn hàng	43
Hình 4.16 Giao diện trang báo cáo thống kê	43

DANH MỤC BẢNG BIỂU

Bảng 1. So sánh mô hình MVC với MVT	28
Bảng 2. Bảng Auth_User (Bản user của Django).....	32
Bảng 3. Bảng Product	33
Bảng 4. Bảng Order	34
Bảng 5. Bảng Category	34
Bảng 6. Bảng Product_Category	35
Bảng 7. Bảng OrderItem	35

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Vấn đề nghiên cứu

Đồ án "Tìm hiểu ngôn ngữ lập trình Python. Xây dựng hệ thống bán game" nghiên cứu và ứng dụng Python để phát triển hệ thống quản lý bán game đáp ứng các nhu cầu cơ bản của người dùng cũng như của người quản trị hệ thống.

Các hướng tiếp cận

Em đã tiến hành tìm hiểu một số website bán game có nội dung đồng nhất, phù hợp với đề tài nghiên cứu như website divineshop, hacom..., tìm hiểu xem quy trình xây dựng và các thành phần cần có của một trang web. Ngoài ra em cũng tìm hiểu kiến thức về ngôn ngữ lập trình, công nghệ hiện đại như Python, framework Django, MySQL để áp dụng kết hợp chúng vào việc xây dựng một hệ thống với các chứng năng hoàn chỉnh, dễ sử dụng.

Các giải quyết vấn đề

Để giải quyết vấn đề, em sử dụng Django framework để phát triển Backend, nơi quản lý toàn bộ dữ liệu sản phẩm, giỏ hàng, người dùng và báo cáo thống kê. Django giúp quản lý cơ sở dữ liệu trực tiếp thông qua giao diện người dùng, cho phép thao tác dễ dàng mà không cần truy cập vào cơ sở dữ liệu riêng biệt. Đồng thời, Frontend được xây dựng với HTML, CSS/Bootstrap và Django Templates để người dùng có thể thao tác trực quan.

Kết quả đạt được

Kết quả đạt được từ đồ án "Xây dựng hệ thống bán game và trò chơi điện tử" xây dựng được một hệ thống hoàn chỉnh với các chức năng cơ bản về quản lý sản phẩm, giỏ hàng, tìm kiếm và báo cáo thống kê doanh thu, tồn kho. Giao diện thân thiện với người dùng, người quản trị cũng dễ dàng thêm, sửa, xóa sản phẩm, và nhận báo cáo thống kê. Mô hình MVT trong Django giúp em phát triển hệ thống bán sản phẩm hiệu quả bằng cách tách biệt rõ ràng các thành phần: Model quản lý dữ liệu, Template tạo giao diện người dùng dễ dàng tùy chỉnh, và View xử lý logic và tương tác với người dùng. Điều này giúp hệ thống dễ bảo trì và mở rộng.

MỞ ĐẦU

Lý do chọn đề tài

Em lựa chọn đề tài xây dựng hệ thống bán game và trò chơi điện tử và sử dụng ngôn ngữ lập trình Python vì nó là một ngôn ngữ phổ biến, dễ học, và có ứng dụng rộng rãi trong phát triển web, phần mềm. Django, một framework mạnh mẽ của Python, hỗ trợ xây dựng backend hiệu quả với các tính năng bảo mật, xử lý dữ liệu và logic phức tạp. Về mặt thực tiễn, xây dựng hệ thống bán game đáp ứng nhu cầu quản lý sản phẩm, đơn hàng, và báo cáo cho cửa hàng, giúp tự động hóa quy trình kinh doanh. Hệ thống cũng mang lại trải nghiệm tốt hơn cho người dùng nhờ giao diện thân thiện và tính năng tìm kiếm, giỏ hàng tiện ích. Ngoài ra, đề tài này còn giúp em nâng cao kỹ năng lập trình, áp dụng lý thuyết vào thực tế, làm quen với quy trình phát triển phần mềm hiện đại, và chuẩn bị tốt hơn cho các dự án lớn sau này.

Mục đích nghiên cứu

Mục đích chính của đề tài là xây dựng một hệ thống quản lý bán game dựa trên ngôn ngữ lập trình Python và framework Django, nhằm tự động hóa các quy trình như quản lý sản phẩm, giỏ hàng. Hệ thống này không chỉ giúp tự động hóa quy trình kinh doanh, giảm thiểu sai sót và tăng hiệu quả quản lý mà còn cải thiện trải nghiệm người dùng thông qua giao diện trực quan, thân thiện, tích hợp các tính năng như tìm kiếm, phân loại, phân quyền và báo cáo chi tiết. Đồng thời, đề tài giúp em ứng dụng lý thuyết vào thực tiễn, nâng cao kỹ năng lập trình và làm quen với quy trình phát triển phần mềm hiện đại.

Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài bao gồm các công nghệ, công cụ và quy trình liên quan đến việc xây dựng và vận hành hệ thống quản lý bán game. Đề tài nghiên cứu về ngôn ngữ lập trình Python, đặc biệt là các thư viện hỗ trợ phát triển phần backend. Đồng thời, hệ thống được xây dựng trên nền tảng framework Django để quản lý cơ sở dữ liệu, xử lý logic nghiệp vụ và xây dựng API. Cơ sở dữ liệu được quản lý bằng hệ quản trị cơ sở dữ liệu MySQL, và việc thiết kế giao diện người dùng sử dụng HTML, CSS, JavaScript để tạo trải nghiệm dễ sử dụng. Cuối cùng, đề tài còn nghiên cứu các quy trình quản lý bán game như quản lý sản phẩm, đơn hàng, báo cáo doanh thu và tồn kho.

Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài tập trung vào việc tìm hiểu và ứng dụng ngôn ngữ lập trình Python cùng framework Django để xây dựng một hệ thống quản lý bán game cơ bản. Phần backend được triển khai bằng Django để xử lý logic và quản lý cơ sở dữ liệu, sử dụng MySQL làm hệ quản trị cơ sở dữ liệu. Phần frontend được xây dựng với Django Template kết hợp HTML, CSS và JavaScript để tạo giao diện người dùng thân thiện và dễ sử dụng. Những vấn đề như bảo mật nâng cao hay các công nghệ không liên quan trực tiếp đến các công cụ và nền tảng đã được chọn sẽ không được nghiên cứu trong khuôn khổ của đề tài này.

CHƯƠNG 1: TỔNG QUAN

Trong thời đại công nghệ số phát triển mạnh mẽ, việc ứng dụng công nghệ vào quản lý và kinh doanh đã trở thành yếu tố quan trọng giúp doanh nghiệp nâng cao hiệu quả hoạt động. Đặc biệt trong ngành kinh doanh game và trò chơi điện tử, sự cạnh tranh ngày càng gia tăng với sự xuất hiện của các shop bán lẻ trực tuyến và nền tảng thương mại điện tử lớn. Điều này đòi hỏi các cửa hàng game truyền thống phải cải tiến, áp dụng công nghệ để tối ưu hóa quy trình và nâng cao năng lực cạnh tranh.

Trước đây, việc quản lý kinh doanh game thường được thực hiện thủ công, thông qua sổ sách ghi chép hoặc bảng tính Excel đơn giản, chưa có hệ thống tự động hỗ trợ. Điều này gây ra nhiều khó khăn trong việc theo dõi hàng tồn kho, xử lý đơn hàng, và tạo báo cáo doanh thu. Các phương pháp quản lý truyền thống không chỉ mất nhiều thời gian mà còn dễ xảy ra sai sót, ảnh hưởng đến hiệu quả kinh doanh. Khi số lượng sản phẩm tăng lên, việc tìm kiếm và cập nhật thông tin sản phẩm càng trở nên phức tạp, dẫn đến khó khăn trong việc đáp ứng nhu cầu của khách hàng. Ngoài ra, việc thiếu các công cụ phân tích và báo cáo kịp thời còn hạn chế khả năng đưa ra quyết định chiến lược trong kinh doanh.

Xuất phát từ thực tế trên, đề tài "Xây dựng hệ thống quản lý bán game và trò chơi điện tử" được triển khai nhằm mang đến một giải pháp công nghệ toàn diện. Hệ thống này không chỉ giúp tự động hóa quy trình quản lý mà còn tận dụng sức mạnh của công nghệ hiện đại để cải thiện hiệu quả kinh doanh. Sử dụng các công nghệ như Python, Django Framework, và MySQL, hệ thống sẽ xây dựng một nền tảng quản lý chuyên nghiệp, đáp ứng đầy đủ các yêu cầu về quản lý kho, sản phẩm, theo dõi doanh thu và phục vụ khách hàng một cách hiệu quả.

Ngoài ra, hệ thống được thiết kế với khả năng xử lý dữ liệu nhanh chóng, giao diện thân thiện và tính năng bảo mật cao, phù hợp với nhu cầu của các cửa hàng vừa và nhỏ. Không chỉ vậy, hệ thống còn dễ dàng mở rộng và tích hợp thêm nhiều tính năng mới như quản lý khách hàng, tích hợp thanh toán trực tuyến, hay hỗ trợ bán hàng đa kênh.

Mục tiêu của hệ thống không chỉ dừng lại ở việc tự động hóa quy trình mà còn hướng đến việc nâng cao trải nghiệm người dùng, giúp khách hàng dễ dàng tìm kiếm và mua sản phẩm, đồng thời hỗ trợ doanh nghiệp phát triển bền vững và duy trì lợi thế cạnh tranh trên thị trường kinh doanh game hiện nay.

Python là ngôn ngữ lập trình linh hoạt, dễ học và có cộng đồng phát triển mạnh mẽ. Đặc biệt, Python hỗ trợ nhiều thư viện và công cụ mạnh mẽ cho phát triển web, như xử lý dữ liệu, tạo giao diện người dùng, và quản lý cơ sở dữ liệu.

Tuy nhiên, việc phát triển ứng dụng web chỉ với Python thuần có thể mất nhiều thời gian và dễ mắc lỗi, vì lập trình viên phải tự xử lý mọi khía cạnh, từ routing, bảo mật đến quản lý session. Sử dụng một framework như Django giúp tối ưu hóa quy trình phát triển web. Framework cung cấp các công cụ tích hợp sẵn để quản lý cơ sở dữ liệu, bảo mật, xử lý yêu cầu HTTP và tạo giao diện. Hơn nữa, Django còn tuân theo các nguyên tắc tốt nhất như DRY (Don't Repeat Yourself) và mô hình MVT (Model-View-Template), giúp tổ chức mã nguồn gọn gàng và dễ bảo trì. Vì vậy, chọn một framework tốt như Django không chỉ tiết kiệm thời gian mà còn đảm bảo tính ổn định và mở rộng của ứng dụng web.

MySQL được chọn để lưu trữ dữ liệu sản phẩm, khách hàng, và báo cáo, với công cụ MySQL Workbench hỗ trợ trực quan hóa và quản lý cơ sở dữ liệu.

Phần frontend sử dụng HTML, CSS, JavaScript, và Bootstrap để xây dựng giao diện thân thiện, responsive và tương tác tốt. Visual Studio Code được sử dụng làm IDE chính, tối ưu hóa quá trình phát triển nhờ tích hợp kiểm tra lỗi và quản lý mã nguồn. Những công nghệ này đảm bảo hệ thống hoạt động ổn định, dễ mở rộng và đáp ứng nhu cầu thực tế.

\

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 bậc cao được sử dụng rộng rãi trong các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và máy học. Các nhà phát triển sử dụng Python vì nó hiệu quả, dễ học và có thể chạy trên nhiều nền tảng khác nhau. Phần mềm Python được tải xuống miễn phí, tích hợp tốt với tất cả các loại hệ thống và tăng 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. [1]. Phiên bản mới nhất hiện nay của Python là version 3.13.1, nó tập trung cải thiện hiệu suất và thêm tính năng mới trong thư viện chuẩn.

2.1.2 Ứng dụng

Python là ngôn ngữ được ứng dụng đa dạng trong các lĩnh vực.

- Phát triển web phía máy chủ: Bao gồm những hàm backend phức tạp mà các trang web thực hiện để hiển thị thông tin cho người dùng. Ví dụ: các trang web phải tương tác với cơ sở dữ liệu, giao tiếp với các trang web khác và bảo vệ dữ liệu khi truyền qua mạng. Django và Flask là 2 framework phổ biến hiện nay dành cho các lập trình viên Python để tạo ra các website.
- Tự động hóa bằng các tập lệnh Python: Dùng Python để tự động hóa các tác vụ lặp đi lặp lại, như quét dữ liệu với BeautifulSoup, Selenium; tự động hóa hệ thống với os hoặc shutil.
- Khoa học máy tính: Trong Python có rất nhiều thư viện quan trọng phục vụ cho ngành khoa học máy tính như: OpenCV cho xử lý ảnh và Machine Learning; Scipy và Numpys cho lĩnh vực toán học, đại số tuyến tính; Pandas cho việc phân tích dữ liệu, ...
- Lĩnh vực IoT: Python có thể viết được các ứng dụng cho nền tảng nhúng, đồng thời cũng được lựa chọn cho việc xử lý dữ liệu lớn. Vì thế Python là một ngôn ngữ quen thuộc trong lĩnh vực Internet kết nối vạn vật.

- Lập trình game: Pygame là một bộ module Python cross-platform được thiết kế để viết game cho cả máy tính và các thiết bị di động

2.1.3 Tải và cài đặt Python

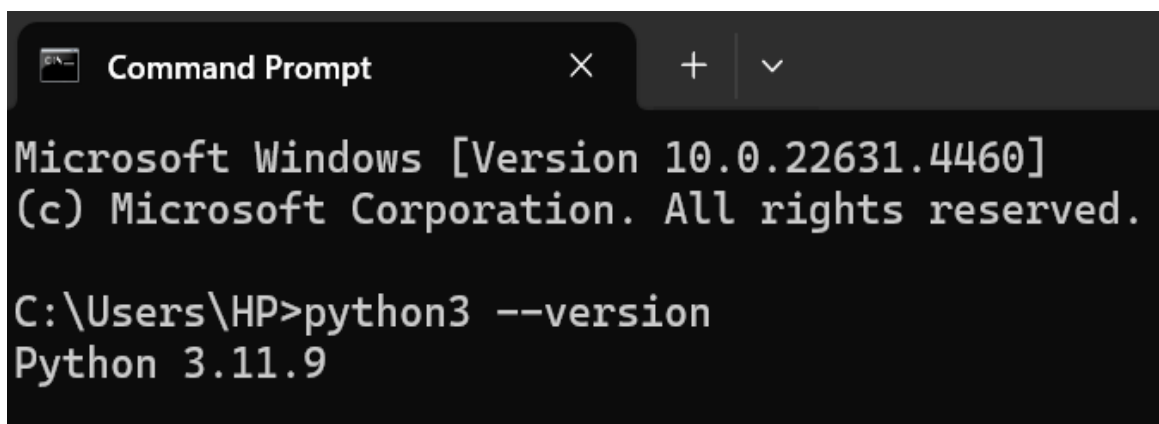
Truy cập trang web chính thức của Python tại python.org, vào mục Downloads, nơi website sẽ tự động đề xuất phiên bản phù hợp với hệ điều hành của bạn (Windows hoặc Mac). Tại đây, bạn có thể chọn tải phiên bản Python mong muốn hoặc phiên bản mới nhất được đề xuất, sau đó nhấp vào nút Download để tải về máy.



Hình 2.1 Python phiên bản mới nhất

Sau khi tải xong, mở tập tin đã tải và cài đặt theo từng bước. Đảm bảo rằng bạn đã chọn tùy chọn Add Python to PATH trong quá trình cài đặt, điều này sẽ giúp bạn có thể chạy Python từ dòng lệnh của máy tính.

Để kiểm tra xem Python đã được cài đặt thành công hay chưa, mở Terminal trên Linux/macOS hoặc Command Prompt trên Windows, sau đó chạy lệnh: **python --version** hoặc **python3 --version**.



Hình 2.2 Kiểm tra cài đặt Python

2.2 Framework Django

2.2.1 Khái quát về Django

Django là một web framework bậc cao, mã nguồn mở, được phát triển bằng Python, giúp đơn giản hóa việc xây dựng ứng dụng web bằng cách cung cấp các module tái sử dụng. Thay vì phải viết mã lặp lại cho các chức năng chung như xác thực người dùng hay truy vấn cơ sở dữ liệu, Django tổ chức các công cụ này thành một khung làm việc mạnh mẽ, giúp tăng hiệu suất và giảm thời gian phát triển ứng dụng. Hiện tại, Django có một cộng đồng đông đảo và tài liệu phong phú, hỗ trợ tốt cho cả người mới bắt đầu.

Django còn nổi bật với triết lý "batteries-included", tích hợp sẵn mọi công cụ cần thiết để xây dựng từ ứng dụng đơn giản đến các hệ thống phức tạp. Đặc biệt, nó đảm bảo bảo mật cao, giảm thiểu các lỗ hổng như SQL Injection hay Cross-Site Scripting. Với Django, lập trình viên có thể tập trung vào logic ứng dụng mà không cần lo ngại về các chi tiết kỹ thuật phức tạp.

2.2.2 Ưu và nhược điểm của Django

- Ưu điểm:
 - + Đơn giản, tiết kiệm thời gian: Django dễ học nhờ sử dụng Python, với code ngắn gọn và thư viện dữ liệu phong phú. Nó tự động loại bỏ mã trùng lặp, giúp tiết kiệm thời gian.
 - + Độ bảo mật cao: Django có hệ thống bảo mật mạnh mẽ, bao gồm mã hóa mật khẩu, phòng tránh lỗi cơ bản và xử lý lỗ hổng bảo mật hiệu quả.
 - + Có khả năng mở rộng: Django phù hợp với các ứng dụng web lớn, như Dropbox, Mozilla, Youtube, cho phép mở rộng dễ dàng và hỗ trợ nhiều máy chủ.
 - + Đa nền tảng: Django hỗ trợ nhiều hệ điều hành (Mac, Linux, Windows) và có thể sử dụng nhiều cơ sở dữ liệu cùng lúc.
 - + Cộng đồng người dùng lớn mạnh: Django có cộng đồng người dùng lớn và tài liệu phong phú, hỗ trợ qua các diễn đàn và website.
- Nhược điểm:
 - + Có thể tồn tại một vài vấn đề khi phát triển các dự án nhỏ.

- + Yêu cầu kiến thức sâu về Python: Django yêu cầu các lập trình viên có kiến thức vững về Python và một số khái niệm nâng cao như ORM, cấu trúc URL, và hệ thống template. Điều này có thể là một trở ngại đối với các lập trình viên mới bắt đầu, đặc biệt là những người không quen với cách làm việc của Python.

2.2.3 Các tính năng cơ bản

Dưới đây là một số tính năng cơ bản và thường sử dụng:

- **Admin Interface tự động:** Django tự động tạo bảng điều khiển quản trị dựa trên các models trong ứng dụng. Nó cung cấp giao diện thân thiện cho quản trị viên để quản lý dữ liệu mà không cần phải lập trình thủ công.
- **Hệ thống quản lý cơ sở dữ liệu với Object-Relational Mapping:** Giúp chuyển đổi dữ liệu giữa mô hình đối tượng Python và cơ sở dữ liệu. Các lập trình viên có thể tương tác với cơ sở dữ liệu mà không cần viết SQL thủ công, giúp tăng hiệu quả và giảm lỗi.
- **Templates:** Django có một hệ thống template đơn giản nhưng linh hoạt, cho phép kết hợp HTML với các biến và logic hiển thị một cách dễ dàng.
- **URL Routing:** Hệ thống định tuyến URL của Django cho phép ánh xạ các URL thân thiện với người dùng đến các chức năng trong ứng dụng mà không cần phụ thuộc vào cấu trúc tệp thực tế.
- **Kiến trúc Model-View-Controller (MVC):** Django thực chất áp dụng mô hình tương tự, được gọi là Model-View-Template (MVT). Cách tổ chức này giúp mã nguồn dễ bảo trì, tách biệt logic xử lý, dữ liệu và giao diện.
- **Authentication và Authorization:** Django cung cấp các công cụ mạnh mẽ để xác thực người dùng và kiểm soát quyền truy cập thông qua hệ thống user và groups.

2.2.4 Cài đặt, tạo và chạy một dự án Django

2.2.4.1 Cài đặt Django

- Django là một framework Python, nên bạn cần có môi trường Python được thiết lập như mục 2.1.3 trước khi cài đặt.
- Sau khi đã cài đặt được Python rồi thì tiến hành cài đặt Django bằng cách gõ lệnh “**pip install django**” trong terminal.

```
PS C:\Users\HP> pip install django
>>
Collecting django
  Downloading Django-5.1.4-py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: asgiref<4,>=3.8.1 in c:\users\hp\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from django) (3.8.1)
Requirement already satisfied: sqlparse>=0.3.1 in c:\users\hp\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from django) (0.5.1)
Requirement already satisfied: tzdata in c:\users\hp\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from django) (2024.2)
Downloading Django-5.1.4-py3-none-any.whl (8.3 MB)
 8.3/8.3 MB 21.4 MB/s eta 0:00:00
Installing collected packages: django
  WARNING: The script django-admin.exe is installed in 'C:\Users\HP\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\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 django-5.1.4
```

Hình 2.3 Cài đặt django trong terminal Visual Studio Code

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

```
PS C:\Users\HP> python -m django --version
>>
5.1.4
```

Hình 2.4 Lệnh kiểm tra phiên bản django đã cài đặt

2.2.4.2 Tạo dự án

- Sử dụng câu lệnh sau để tạo dự án Django: **django-admin -m startproject <tên_dự_án>**. Ví dụ, để tạo một dự án tên **myproject**, gõ : **django-admin -m startproject myproject**.

```
PS F:\> python -m django startproject myproject
>>
```

Hình 2.5 Lệnh tạo một dự án django

- Sau đó vào Visual Studio Code mở thư mục vừa tạo theo đường dẫn để xem cấu trúc thư mục dự án gồm các thành phần chính sau:



Hình 2.6 Cấu trúc các tập tin của một dự án django cơ bản

- Trong Django, mỗi chức năng thường được tổ chức thành một ứng dụng riêng biệt. Để tạo một ứng dụng gõ lệnh: **python manage.py startapp**

<Tên_ứng_dụng>. Ví dụ tạo ứng dụng tên “app”: `python manage.py startapp app`.

- Cần thêm ứng dụng này vào danh sách **INSTALLED_APPS** trong file **settings.py**:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'app',  
]
```

Hình 2.7 Khai báo ứng dụng đã tạo vào project

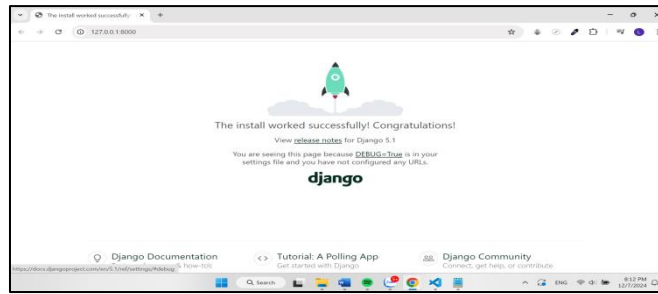
2.2.4.3 Khởi động server của dự án Django

- Di chuyển vào thư mục chứa dự án và gõ lệnh **python manage.py migrate** để cấu hình lại website cập nhật các thay đổi và gõ lệnh **python manage.py runserver** để khởi động server. Server phát triển sẽ chạy tại địa chỉ: <http://127.0.0.1:8000>.

```
PS F:\myproject> python manage.py migrate  
Operations to perform:  
Apply all migrations: admin, auth, contenttypes, sessions  
Running migrations:  
Applying contenttypes.0001_initial... OK  
Applying auth.0001_initial... OK  
Applying admin.0001_initial... OK  
Applying admin.0002_logentry_remove_auto_add... OK  
Applying admin.0003_logentry_add_action_flag_choices... OK  
Applying contenttypes.0002_remove_content_type_name... OK  
Applying auth.0002_alter_permission_name_max_length... OK  
Applying auth.0003_alter_user_email_max_length... OK  
Applying auth.0004_alter_user_username_opts... OK  
Applying auth.0005_alter_user_last_login_null... 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 F:\myproject> python manage.py runserver  
Watching for file changes with StatReloader  
Performing system checks...  
  
System check identified no issues (0 silenced).  
December 07, 2024 - 20:01:04  
Django version 5.1.4, using settings 'myproject.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.
```

Hình 2.8 Cấu hình lại hệ thống

- Truy cập vào trình duyệt web theo địa chỉ localhost trên nếu thấy thông báo "The install worked successfully!" thì dự án đã sẵn sàng để phát triển.



Hình 2.9 Dự án đã hoạt động thành công

- Bước tiếp theo cần làm là định nghĩa các models trong models.py, tạo các view xử lý logic ứng dụng, kết nối views với các URLs trong urls.py, thiết kế giao diện trong templates/.

2.2.5 Các thành phần chính có trong Django

2.2.5.1 Views

Trong Django, **views** là một thành phần chính của kiến trúc MVC và MVT. **Views** chịu trách nhiệm xử lý logic nghiệp vụ và trả về phản hồi cho client (trình duyệt). Đây là cầu nối giữa dữ liệu từ **models** và giao diện người dùng trong **templates**.

```
1 from django.shortcuts import render
2
3 # Create your views here.
4 from django.http import HttpResponse
5
6 def index(request):
7     return HttpResponse("Hello, world. You're at the books index.")
8
9 def home(request):
10    return render(request, 'home.html')
```

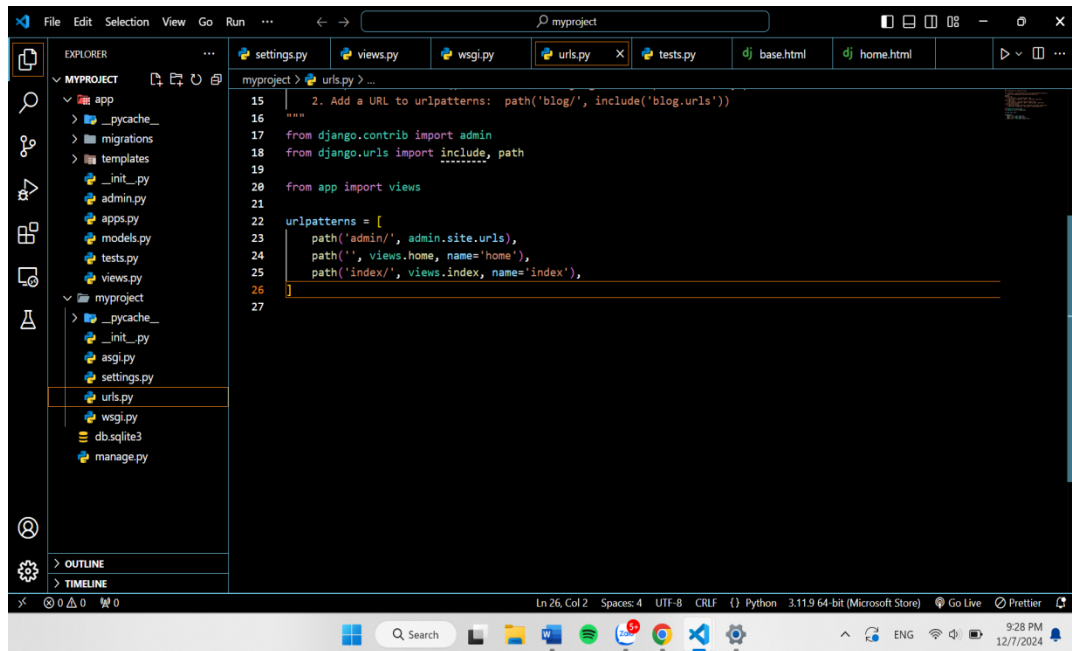
Hình 2.10 Ví dụ về hàm xử lý trong file views.py

2.2.5.2 URLS

- Tập urls.py thường nằm trong thư mục của dự án hoặc ứng dụng, và nó được sử dụng để cấu hình các đường dẫn URL. Cấu trúc tập urls.py gồm đoạn đầu để import các thư viện cần thiết, tiếp theo là phần urlpatterns là nơi chứa các liên kết dẫn đến các trang trong website.

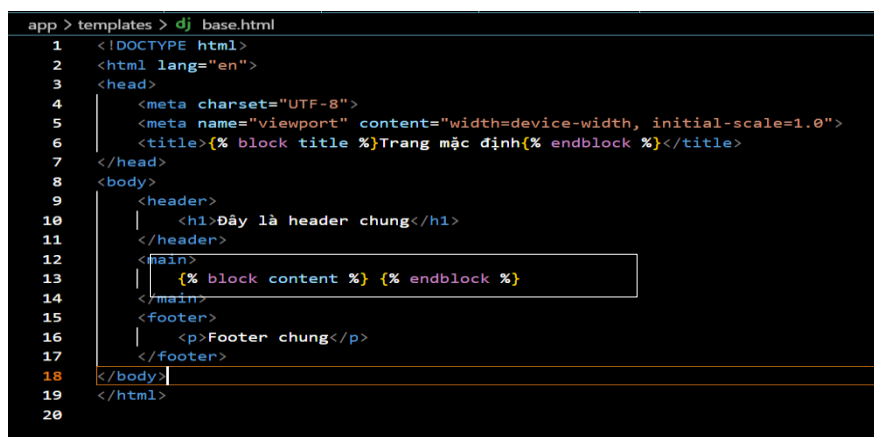
- Cấu trúc khai báo cơ bản: path(route, view, name=None)
 - + route: Chuỗi ký tự đại diện cho URL (ví dụ: **'index/**' hoặc **'** nếu là trang hiện đầu tiên khi mở web lên).
 - + view: Hàm xử lý yêu cầu (ví dụ: **views.index**).

- + name: Tên định danh để tham chiếu URL trong code hoặc template.
- include(): Sử dụng để kết nối urls.py của ứng dụng với dự án. Điều này giúp chia nhỏ các URL khi bạn có nhiều ứng dụng.
- Ví dụ minh họa cấu hình đường dẫn đến **home** và **index** vừa tạo như ở hình 2.10 trong views.py.



Hình 2.11 Ví dụ minh họa cấu hình đường dẫn trong urls.py

- Để hiển thị nội dung trang home.html lên trang base.html:
 - + **Đặt nội dung vào block:** Trong **base.html**, ta sẽ có một phần được gọi là block, ví dụ block tên là **content**, nơi mà nội dung của từng trang con sẽ được chèn vào.



Hình 2.12 Định nghĩa một tag block trong trang cơ sở

+ **Kế thừa từ template base:** Trong file **home.html**, ta sử dụng dòng lệnh `{% extends "base.html" %}` ở đầu file để kế thừa cấu trúc và nội dung chung từ **base.html**. Điều này giúp ta không cần phải viết lại toàn bộ cấu trúc của trang, mà chỉ cần thay thế phần nội dung cụ thể của từng trang. Khi kế thừa từ **base.html**, ta chỉ cần định nghĩa nội dung cho block tên **content** (giống với tên block đã được đặt trong **base.html**) trong **home.html** bằng cách sử dụng `{% block content %} ... {% endblock %}`.

```
app > templates > home.html
1  {% extends "base.html" %}
2
3  {% block title %}
4  Trang chủ
5  {% endblock %}
6
7  {% block content %}
8  <h2>Chào mừng đến với trang chủ!</h2>
9  <p>Đây là nội dung chính của trang.</p>
10 {% endblock %}
11
```

Hình 2.13 Template kế thừa từ trang cơ sở

2.2.5.3 Models

Trong Django, Models là thành phần trung tâm của MVT. Nó đại diện cho lớp logic dữ liệu, được sử dụng để quản lý và tương tác với cơ sở dữ liệu. Mỗi lớp tương ứng với một bảng trong cơ sở dữ liệu, còn các trường (fields) và bản ghi (records) bên trong lần lượt tương ứng với các cột và hàng trong cơ sở dữ liệu. Nó hỗ trợ tạo bảng, thiết lập quan hệ, thực hiện các thao tác CRUD (tạo, đọc, cập nhật, xóa), và quản lý dữ liệu dễ dàng mà không cần viết SQL.

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

Để tạo một ví dụ đơn giản với model và view trong Django, chúng ta sẽ thực hiện các bước sau:

- **Bước 1:** Tạo Model cơ bản

Vào file **models.py** để định nghĩa một model đơn giản gọi là **Article**, bao gồm các trường như tiêu đề với kiểu dữ liệu là **Char** và độ dài là 200, làm tương tự với các trường nội dung và ngày tạo.


```

1
2 from django.db import models
3 # Create your models here.
4 class Article(models.Model):
5     title = models.CharField(max_length=200) # Tiêu đề bài viết
6     content = models.TextField() # Nội dung bài viết
7     created_at = models.DateTimeField(auto_now_add=True) # Thời gian tạo bài viết
8
9     def __str__(self):
10         return self.title # Hiển thị tiêu đề của bài viết

```

Hình 2.14 Tạo model Article trong file models.py

Sau khi tạo model, bạn cần áp dụng nó vào cơ sở dữ liệu và cấu hình lại trang web cập nhật các thay đổi. Gõ các lệnh sau vào terminal để thực hiện công việc trên: `python manage.py makemigrations` và `python manage.py migrate`.

- **Bước 2:** Tạo form nhập liệu và cập nhật views.py để xử lý form

Bạn có thể tạo một form đơn giản của Django để người dùng có thể nhập dữ liệu. Bạn sẽ không cần phải tạo file forms.py riêng, chỉ cần tạo form trực tiếp trong views.py. Sau đó viết một hàm view xử lý việc hiển thị và xử lý dữ liệu từ form.

```

1 from django import forms
2 from django.shortcuts import render
3 from django.http import HttpResponseRedirect
4 from .models import *
5 # Create your views here.
6 # Tạo một form đơn giản
7 class SimpleForm(forms.Form):
8     title = forms.CharField(max_length=100, label='Tiêu đề')
9     content = forms.CharField(widget=forms.Textarea, label='Nội dung')
10
11 # View để hiển thị form và xử lý dữ liệu người dùng
12 def home(request):
13     if request.method == 'POST':
14         form = SimpleForm(request.POST)
15         if form.is_valid():
16             title = form.cleaned_data['title']
17             content = form.cleaned_data['content']
18             article = Article.objects.create(title=title, content=content)
19             new_article = article
20         else:
21             form = SimpleForm()
22     return render(request, 'home.html', {'form': form, 'new_article': new_article})

```

Hình 2.15 Tạo form có sẵn từ module django.forms và hàm xử lý dữ liệu của form

SimpleForm: Là một class kế thừa từ **forms.Form** có sẵn trong Django để tạo ra form cho người dùng nhập dữ liệu.

Như chúng ta thấy, hàm **home** sẽ gọi đến phương thức **POST** nếu người dùng gửi form. Hàm sẽ tạo một form mới và điền dữ liệu mà người dùng đã nhập vào. Tiếp theo, hàm kiểm tra xem dữ liệu người dùng đã nhập có hợp lệ không (ví dụ: không rỗng và đúng định dạng) thông qua **is_valid()**. Nếu hợp lệ, nó sẽ lấy **tiêu đề** và **nội dung** từ form, sau đó lưu bài viết vào cơ sở dữ liệu.

Khi bài viết đã được lưu, nó sẽ được hiển thị lại trên **home.html**. Nếu người dùng chưa gửi form, một form trống sẽ được hiển thị. Cuối cùng, dữ liệu form và bài viết mới (nếu có) sẽ được truyền vào template home.html để hiển thị cho người dùng.

- **Bước 3:** Chỉnh sửa trong template home để hiện form và xuất dữ liệu nhập từ form.

```

1  {% extends "base.html" %}
2  {% block title %}
3  Trang chủ
4  {% endblock %}
5  {% block content %}
6  <h2>Chào mừng đến với trang chủ!</h2>
7  <p>Đây là nội dung chính của trang.</p>
8  <!-- Hiển thị danh sách bài viết -->
9  <h2>Thêm bài viết</h2>
10 <form method="post">
11     {% csrf_token %}
12     {{ form.as_p }}
13     <button type="submit">Gửi</button>
14 </form>
15
16 {% if new_article %}
17 <div style="margin-top: 20px;">
18     <h3>Bài viết vừa thêm:</h3>
19     <p><strong>Tiêu đề:</strong> {{ new_article.title }}</p>
20     <p><strong>Nội dung:</strong> {{ new_article.content }}</p>
21     <p><strong>Ngày tạo:</strong> {{ new_article.created_at }}</p>
22 </div>
23 {% endif %}
24 {% endblock %}

```

Hình 2.16 Truyền form vào template

Đoạn mã trên sử dụng Django template để hiển thị form và bài viết mới. Form được hiển thị dưới dạng các thẻ **<p>** cho từng trường dữ liệu với **{{ form.as_p }}**, và mã **{% csrf_token %}** giúp bảo vệ form khỏi các cuộc tấn công **CSRF**. Nếu có bài viết mới được tạo, phân tử **{% if new_article %}** sẽ kiểm tra và hiển thị thông tin bài viết, bao gồm tiêu đề **{{ new_article.title }}**, nội dung **{{ new_article.content }}** và thời gian tạo bài viết **{{ new_article.created_at }}**. Trường **created_at** sẽ tự động được tạo ra do đã định nghĩa trong model Article với **auto_now_add=True**.

- **Bước 4:** Cấu hình URL: Thêm url của trang home trong tập tin urls.py giống như hình 2.11.
- **Bước 5 :** Tiến hành runserver : Truy cập trang web sẽ hiện thị lên form , ta tiến hành nhập form rồi bấm nút “Gửi”

Hình 2.17 Giao diện form trên website

2.2.5.4 Hệ thống Admin

Hệ thống Admin của Django là một công cụ quản trị mạnh mẽ và dễ sử dụng,

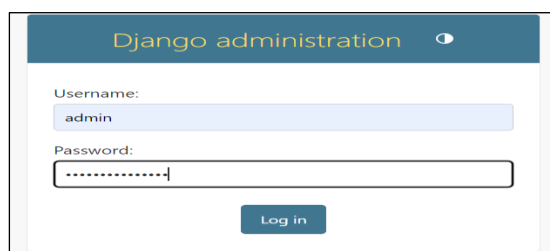
cho phép người dùng (thường là quản trị viên hoặc người phát triển) quản lý dữ liệu trong cơ sở dữ liệu thông qua giao diện web mà không cần phải viết mã SQL hoặc sử dụng dòng lệnh. Django Admin được tích hợp sẵn và có thể tùy chỉnh để phục vụ các nhu cầu quản lý khác nhau.

Để truy cập được vào trang admin có sẵn có Django ta thêm /admin và sao địa chỉ localhost của website, thông thường sẽ là <http://127.0.0.1:8000/admin>. Nhưng trước hết ta cần phải có tài khoản admin. Để có thể tạo được tài khoản ta dùng câu lệnh: **python manage.py createsuperuser**.

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

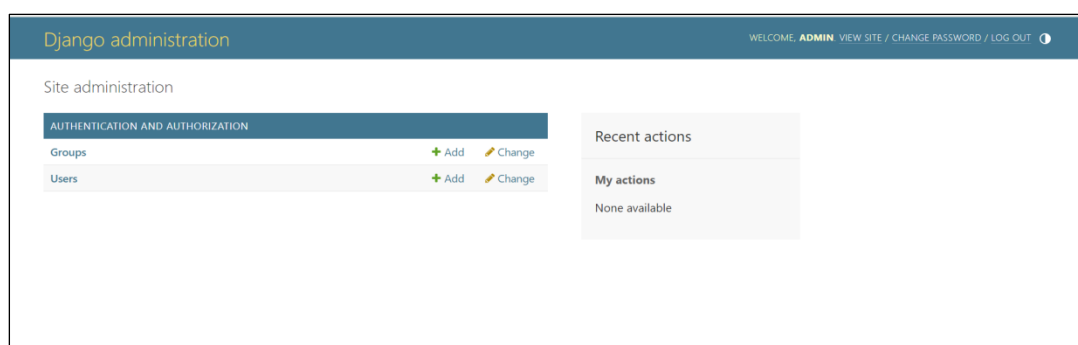
Hình 2.18 Tạo superuser của django

- Khi đã tạo được, ta tiến hành truy cập vào trang admin, và đăng nhập với tài khoản admin vừa tạo.



Hình 2.19 Đăng nhập vào trang quản trị

- Bấm nút “Log in” sẽ hiện ra trang quản trị trông như thế này:



Hình 2.20 Giao diện trang quản trị của django

2.3 MySQL

2.3.1 Giới thiệu

MySQL là một hệ thống quản trị cơ sở dữ liệu (Relational Database Management System - RDBMS) mã nguồn mở hoạt động theo mô hình client - server. MySQL được phát triển bởi Oracle Corporation và được phát hành miễn phí cho cộng đồng người dùng.

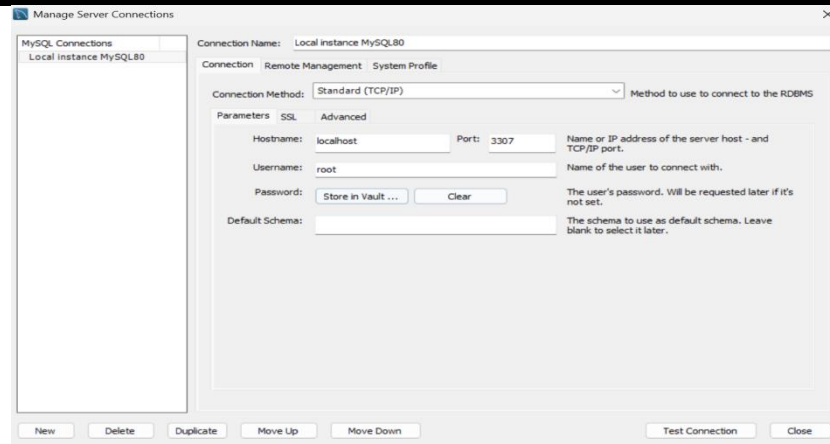
MySQL Workbench là một công cụ để quản lý dữ liệu của MySQL. Nó cung cấp một giao diện đồ họa (GUI) cho phép người dùng thiết kế, quản lý, tối ưu hóa, và thực hiện các thao tác với cơ sở dữ liệu MySQL một cách dễ dàng và hiệu quả.

2.3.2 Một số thế mạnh của MySQL

- Mã nguồn mở và miễn phí: MySQL là phần mềm mã nguồn mở, giúp tiết kiệm chi phí cho các tổ chức và cá nhân.
- Hiệu suất cao: MySQL có khả năng xử lý truy vấn nhanh, đặc biệt trong các ứng dụng web và hệ thống có tải cao.
- Khả năng mở rộng: MySQL hỗ trợ phân tán dữ liệu, giúp dễ dàng mở rộng khi cần thiết.
- Bảo mật mạnh mẽ: MySQL cung cấp các tính năng bảo mật như mã hóa, phân quyền truy cập, và kết nối an toàn qua SSL.
- Đa nền tảng: MySQL có thể chạy trên nhiều hệ điều hành khác nhau như Linux, Windows và MacOS, giúp linh hoạt trong các môi trường phát triển.

2.3.3 Kết nối MySQL với Django

- Trước tiên, ta cần cài đặt MySQL server nếu chưa có và MySQL client để Django có thể giao tiếp với MySQL. Gõ lần lượt các lệnh sau để cài đặt **pip install PyMySQL** và **pip install mysqlclient**.
- Sau khi tải và cài đặt MySQL Workbench cần tạo một kết nối tới MySQL:



Hình 2.21 Tạo MySQL Connections

- Tạo một cơ sở dữ liệu MySQL để lấy nó kết nối với Django. Có thể tạo bằng câu lệnh **CREATE DATABASE <tên_database>;**. Ví dụ **CREATE DATABASE ql_websach;** .
- Tiếp theo mở thư mục dự án Django lên, vào file settings.py để thay đổi cấu hình cơ sở dữ liệu:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql', # Sử dụng MySQL backend
        'NAME': 'qlbangame', # Tên cơ sở dữ liệu MySQL
        'USER': 'root', # Tên người dùng MySQL
        'PASSWORD': '2004', # Mật khẩu MySQL
        'HOST': 'localhost', # Địa chỉ máy chủ (có thể là localhost hoặc IP)
        'PORT': '3307', # Cổng
    }
}
```

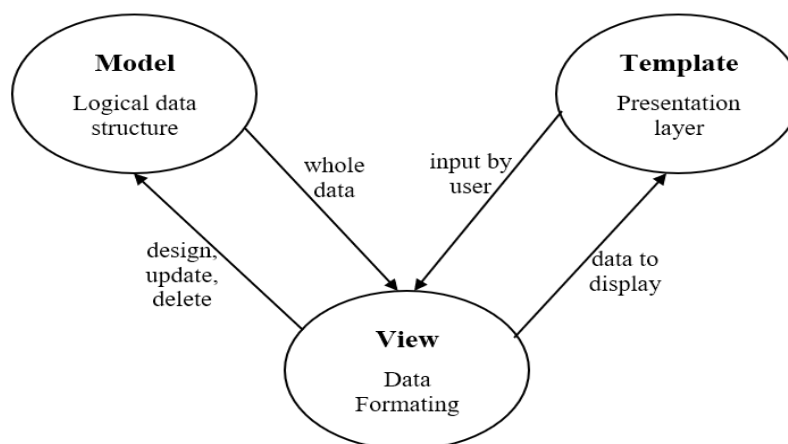
Hình 2.22 Kết nối MySQL với Django

- Cuối cùng tiến hành migrate để cấu hình lại thay đổi và runserver. Nếu sau khi runserver trong terminal không báo lỗi tức là kết nối đã thành công.

2.4 Mô hình MVT

2.4.1 Giới thiệu

MVT là từ viết tắt của Model-View-Template là một mô hình phần mềm được Django sử dụng để phát triển ứng dụng web. MVT giúp phân chia ứng dụng thành các phần riêng biệt, dễ quản lý và phát triển. Nó là biến thể của mô hình MVC nhưng



Hình 2.23 Cấu trúc mô hình MVT

M là Model: Đây là phần chịu trách nhiệm xử lý dữ liệu và logic của ứng dụng. Nó biểu diễn cấu trúc dữ liệu và các chức năng xử lý dữ liệu như truy xuất, cập nhật và xử lý logic. Model có chức năng chuẩn bị dữ liệu để cung cấp cho Controller.

V là View: Là nơi hiển thị dữ liệu cho người dùng theo cách người dùng có thể dễ dàng hiểu và dễ dàng tương tác được. View lấy dữ liệu từ model và gửi dữ liệu đó đến template để hiển thị.

T là Template: Là nơi xử lý giao diện người dùng (UI). Nó là một tệp HTML kết hợp với các thẻ template của Django để hiển thị dữ liệu động từ view. Template nhận dữ liệu từ view và hiển thị chúng cho người dùng.

Bảng 1. So sánh mô hình MVC với MVT

Chức năng	Mô hình MVC	Mô hình MVT
Làm việc với cơ sở dữ liệu	Model	Model
Hiển thị giao diện	View	Template
Xử lý logic	Controller	View

2.4.2 Cách thức mô hình MVT hoạt động

- Khi người dùng truy cập trang web tương tác với hệ thống, trình duyệt sẽ gửi yêu cầu HTTP (GET hoặc POST) tới server.
- Django sẽ nhận yêu cầu và tìm hàm view tương ứng với URL đó để xử lý.
- Hàm view sẽ nhận yêu cầu và thực hiện các bước cần thiết, ví dụ: lấy dữ liệu từ cơ sở dữ liệu thông qua model.

- Model là nơi lưu trữ và quản lý dữ liệu trong cơ sở dữ liệu. View sẽ sử dụng model để lấy hoặc lưu dữ liệu.
- Sau khi xử lý xong, view sẽ gửi dữ liệu đến template. Template chịu trách nhiệm tạo ra giao diện HTML để người dùng thấy.
- Cuối cùng, view sẽ trả lại một trang HTML (tạo từ template) cho trình duyệt, và người dùng sẽ thấy kết quả trên trang web.

2.4.3 Ưu và nhược điểm của mô hình MVT

2.4.3.1 Ưu điểm

- Phân tách rõ ràng: Tách biệt Model, View và Template giúp dễ dàng bảo trì và mở rộng ứng dụng.
- Quản lý dữ liệu tốt: Model giúp quản lý dữ liệu và thao tác với cơ sở dữ liệu hiệu quả mà không cần viết nhiều lệnh SQL.
- Dễ dàng mở rộng và bảo trì: Tách biệt các thành phần giúp dễ dàng phát triển và bảo trì mà không ảnh hưởng tới các phần khác.
- Khả năng tái sử dụng mã: View và Template có thể tái sử dụng trong các phần khác của ứng dụng, giảm mã lặp lại.
- Bảo mật: Django cung cấp các cơ chế bảo mật sẵn có (như chống tấn công CSRF), giảm thiểu rủi ro bảo mật.

2.4.3.2 Nhược điểm

- Khó khăn trong tùy chỉnh giao diện: Tùy chỉnh giao diện phức tạp có thể khó khăn so với các framework front-end khác.
- Cấu trúc phức tạp cho ứng dụng nhỏ: Với ứng dụng nhỏ, MVT có thể gây dư thừa và không cần thiết.
- Hạn chế xử lý UI động: Django không hỗ trợ mạnh mẽ cho các ứng dụng giao diện người dùng động, yêu cầu thêm công cụ hỗ trợ như JavaScript.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1 Mô tả bài toán

Bài toán xây dựng hệ thống bán game và trò chơi điện tử bao gồm việc phát triển một nền tảng trực tuyến cho phép người dùng dễ dàng tìm kiếm, chọn mua và thực hiện thao tác thêm vào giỏ hàng. Hệ thống cần phải có các chức năng cơ bản như quản lý thông tin, giỏ hàng. Admin cần có khả năng quản lý tồn kho, bổ sung hoặc chỉnh sửa sản phẩm trong hệ thống. Hệ thống cũng cần cung cấp báo cáo doanh thu, top sản phẩm bán chạy, và các chỉ số khác để hỗ trợ quyết định kinh doanh. Đồng thời, quản lý người dùng và phân quyền là một yếu tố quan trọng, giúp phân biệt quyền hạn giữa khách hàng và quản trị viên. Mục tiêu của hệ thống là cung cấp trải nghiệm mua sắm dễ dàng, nhanh chóng và bảo mật cho người dùng, đồng thời giúp admin quản lý hiệu quả quá trình kinh doanh game trực tuyến.

3.2 Đặc tả yêu cầu hệ thống

3.2.1 Yêu cầu chức năng

- Người quản trị (admin):
 - + Quản lý sản phẩm: Có thể thêm, sửa, xóa thông tin, danh mục và cung cấp danh sách cho khách hàng với các bộ lọc tìm kiếm theo tên.
 - + Quản lý giỏ hàng: Cho phép khách hàng thêm, sửa, xóa các sản phẩm trong giỏ hàng. Tính toán tổng giá trị giỏ hàng, số lượng, và hiển thị thông tin về các sản phẩm trong giỏ hàng.
 - + Quản lý người dùng: Hệ thống hỗ trợ quản lý tài khoản người dùng với các quyền hạn khác nhau (admin, khách hàng).
 - + Báo cáo thống kê: Hệ thống có các báo cáo về doanh thu, top sản phẩm bán chạy, tồn kho bằng biểu đồ. Chỉ Admin mới có thể xem được báo cáo thống kê này.
 - + Phân quyền: Phân quyền rõ ràng cho admin và người dùng, đảm bảo mỗi người có thể truy cập và thao tác với các tính năng phù hợp.
- Người dùng (khách hàng):
 - + Đăng ký và đăng nhập: Người dùng có thể tạo tài khoản cá nhân, đăng nhập để truy cập các tính năng cá nhân hóa như thêm vào giỏ hàng, xem giỏ hàng.

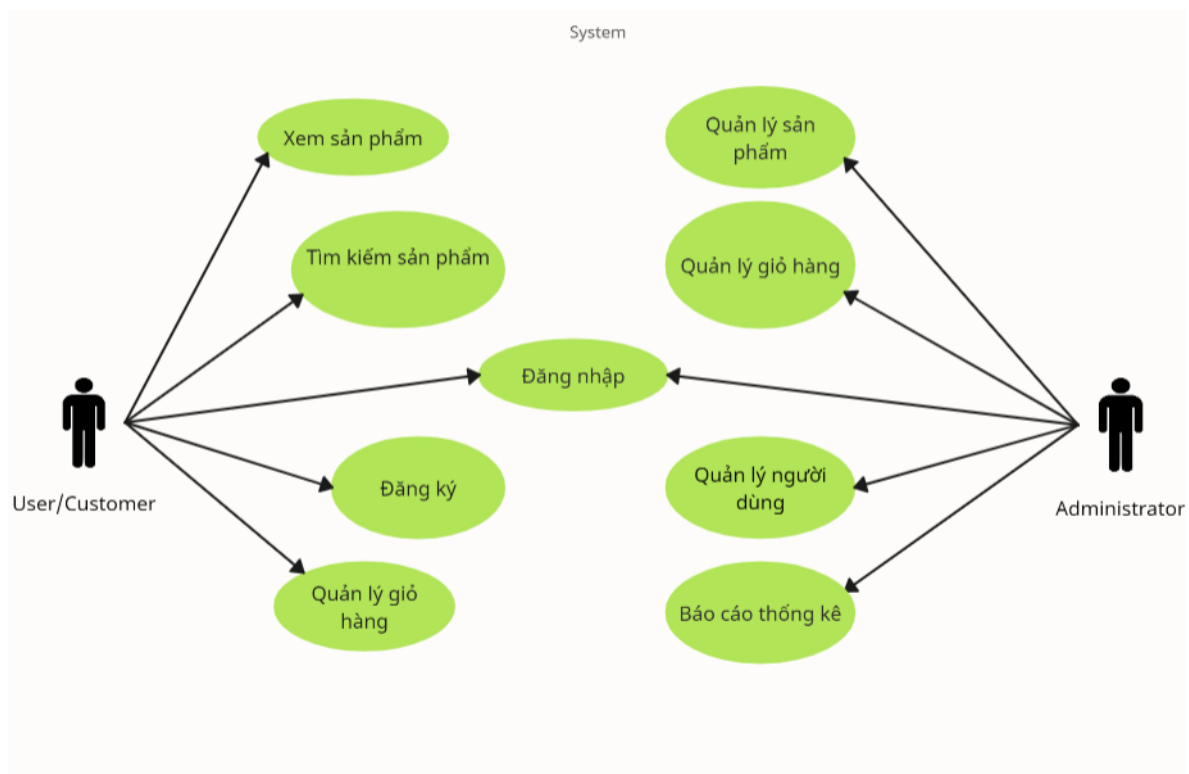
- + Xem và tìm kiếm sản phẩm: Người dùng có thể duyệt qua danh sách các sản phẩm có sẵn, sử dụng bộ lọc để tìm kiếm theo tên.
- + Thêm vào giỏ hàng: Người dùng có thể thêm sản phẩm vào giỏ hàng, thay đổi số lượng hoặc xóa sản phẩm khỏi giỏ hàng. Giỏ hàng sẽ tự động cập nhật khi có sự thay đổi và người dùng có thể xem tổng giá trị đơn hàng trước khi thanh toán.

3.2.2 Yêu cầu phi chức năng

Để nói về một website bán hàng online, đầu tiên phải nói đến giao diện, giao diện phải ưa nhìn, rõ ràng, màu sắc hài hòa, không để mọi thứ quá lộn xộn. Đảm bảo tính thẩm mỹ và gây ấn tượng tốt với người dùng khi họ vừa truy cập vào website.

Các chức năng phải dễ dàng sử dụng, phù hợp với hầu hết độ tuổi của khách hàng, không nên quá phức tạp sẽ gây khó khăn cho người dùng trong quá trình sử dụng.

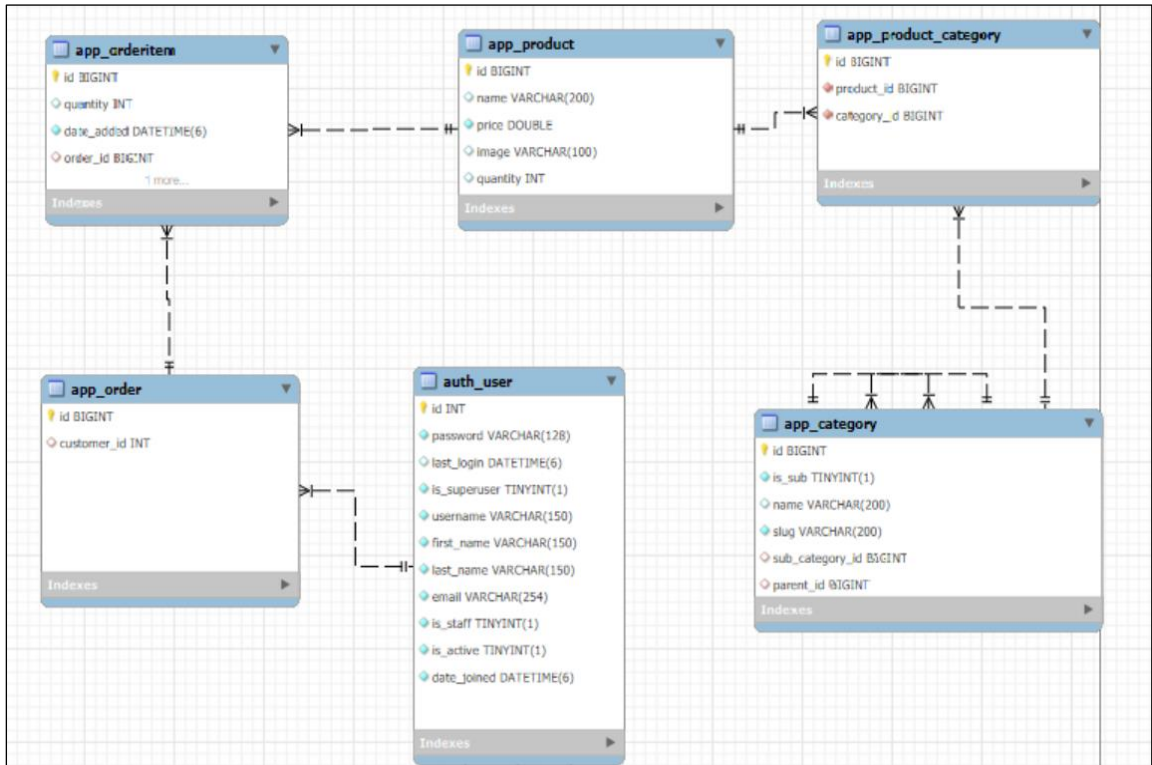
3.2.3 Sơ đồ Use-case



Hình 3.1 Sơ đồ use-case phân quyền chức năng

3.3 Thiết kế cơ sở dữ liệu trong MySQL Workbench

3.3.1 Mô hình cơ sở dữ liệu



Hình 3.2 Sơ đồ cơ sở dữ liệu trong MySQL Workbench

3.3.2 Bảng thực thể

Bảng 2. Bảng Auth_User (Bản user của Django)

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	int	Mã người dùng	- Primary Key - Not null - auto_increment
2	password	varchar	Mật khẩu của người dùng	Not null
3	last_login	datetime	Thời gian đăng nhập lần cuối của người dùng	Có thể null
4	is_superuser	tinyint	Xác định người dùng có phải là quản trị viên siêu quyền	Not null

			(1: có quyền, 0: không)	
5	username	varchar	Tên đăng nhập	Not null
6	first_name	varchar	Tên của người dùng	Có thể null
7	last_name	varchar	Họ của người dùng	Có thể null
8	email	varchar	Địa chỉ email	Có thể null
9	is_staff	tinyint	Xác định người dùng có phải là nhân viên quản lý hay không (1: có quyền, 0: không)	Not null
10	is_active	tinyint	Xác định người dùng có còn hoạt động hay không (1: hoạt động, 0: không)	Not null
11	date_joined	datetime	Thời gian người dùng tham gia hệ thống	auto_now_add=True

Bảng 3. Bảng Product

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	bigint	Mã sản phẩm	- Primary Key - Not null - auto_increment
2	name	varchar	Tên sản phẩm	Có thể null
3	price	double	Giá của sản phẩm	Not null
4	image	varchar	Đường dẫn đến hình ảnh của sản phẩm	Có thể null
5	quantity	int	Số lượng sản phẩm có trong kho	Có thể null, mặc định là 20
6	featured	tinyint	Xác định có phải là sản	Có thể null

			phẩm nổi bật không (1: có, 0: không)	
--	--	--	--------------------------------------	--

Bảng 4. Bảng Order

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	bigint	Mã đơn hàng	-Primary Key -Not null -auto_increment
2	customer_id	int	Mã khách hàng, liên kết với người dùng trong bảng auth_user	- Foreign Key - Not null

Bảng 5. Bảng Category

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	bigint	Mã danh mục	-Primary Key -Not null -auto_increment
2	sub_category	tinyint	Liên kết đến chính bảng Category	-Foreign Key -Có thể null
3	is_sub	tinyint	Xác định danh mục này có phải là danh mục con của một danh mục khác hay không (1: có, 0: không)	- Not null - Mặc định là không
4	name	varchar	Tên của danh mục sản phẩm	Có thể null
5	slug	varchar	Chuỗi định danh thân thiện với URL	- Not null - Duy nhất

6	parent_id	bigint	Mã của danh mục cha (nếu có)	- Foreign Key - Có thể null
---	-----------	--------	---------------------------------	--------------------------------

Bảng 6. Bảng Product_Category

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	bigint	Mã của bản ghi	Primary Key
2	product_id	bigint	Mã sản phẩm, liên kết với id của bảng Product	- Foreign Key - Not null
3	category_id	bigint	Mã danh mục, liên kết với id bảng Category	- Foreign Key - Not null

Bảng 7. Bảng OrderItem

STT	Thuộc tính	Kiểu dữ liệu	Diễn giải	Đặc điểm
1	id	bigint	Mã mục đơn hàng, đại diện cho một sản phẩm trong giỏ hàng	- Primary Key - Not null - auto_increment
2	product_id	bigint	Mã sản phẩm, để liên kết với bảng Product	- Foreign Key - Not null
3	order_id	bigint	Mã đơn hàng, liên kết với bảng Order	- Foreign Key - Not null
4	quantity	int	Số lượng sản phẩm trong giỏ hàng	Có thể null
5	date_added	datetime	Thời gian thêm sản phẩm vào giỏ hàng	auto_now_add=True

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

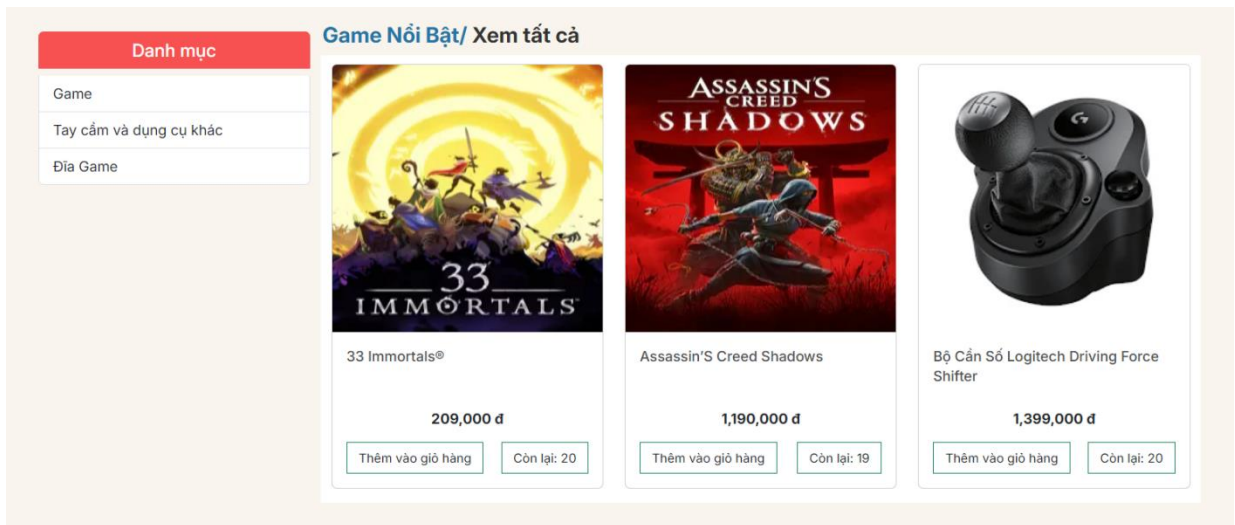
4.1 Giao diện dành cho người dùng

4.1.1 Giao diện trang chủ

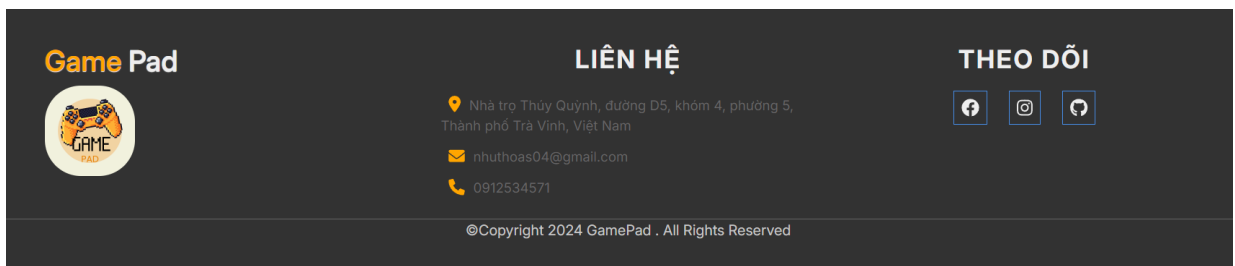
Trang chủ là trang đầu tiên hiển thị khi người dùng truy cập vào website. Giao diện trang chủ bao gồm thanh menu ngang, slide banner, footer, danh mục các sản phẩm hiện có cũng như danh mục các sản phẩm nổi bật. Giao diện tương đối dễ sử dụng và gọn gàng.



Hình 4.1 Thanh menu và banner



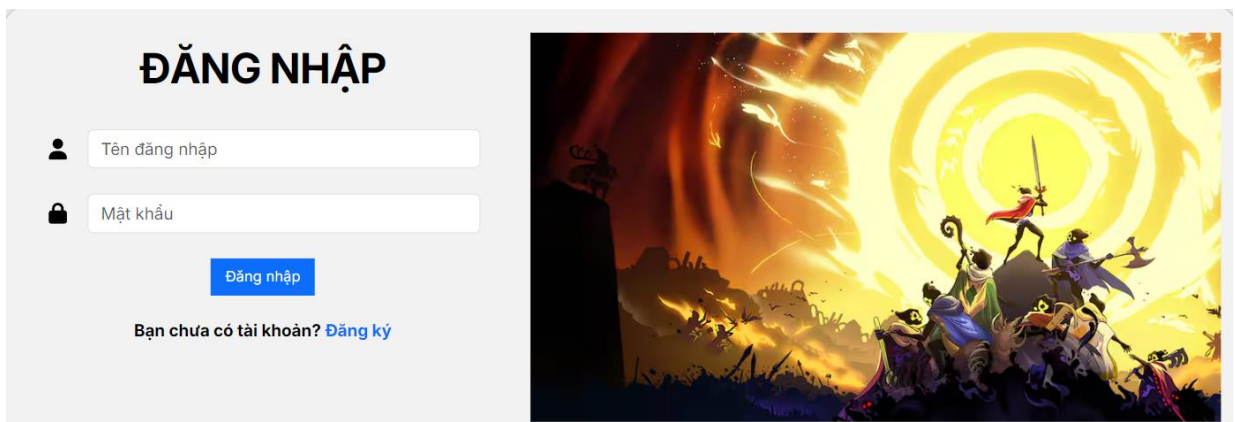
Hình 4.2 Giao diện nội dung trang chủ



Hình 4.3 Footer

4.1.2 Giao diện trang đăng nhập

Khi nhấn vào “Đăng nhập” trên thanh menu thì sẽ dẫn tới trang đăng nhập của trang web, tại đây người dùng có thể đăng nhập với tài khoản đã đăng ký mới có thể thực hiện chức năng thêm vào giỏ hàng cũng như xem giỏ hàng của mình. Nếu chưa có tài khoản thì người dùng có thể chọn vào link đăng ký trên form đăng nhập hoặc trên thanh menu để đăng ký tài khoản mới.



Hình 4.4 Giao diện trang đăng nhập

Nếu chưa đăng nhập mà người dùng bấm vào nút thêm vào giỏ hàng ở mỗi sản phẩm hoặc xem giỏ hàng thì sẽ hiển thị thông báo chuyển đến trang đăng nhập hoặc đăng ký.

4.1.3 Giao diện trang đăng ký

Khi người dùng nhấn chọn vào link đăng ký trên form đăng nhập hoặc trên thanh menu sẽ dẫn đến trang đăng ký. Người dùng có thể nhập các thông tin như tên người dùng, email, mật khẩu và nhấn vào nút đăng kí để tiến hành đăng ký. Khi đăng ký thành công sẽ có thông báo trong form.

ĐĂNG KÝ









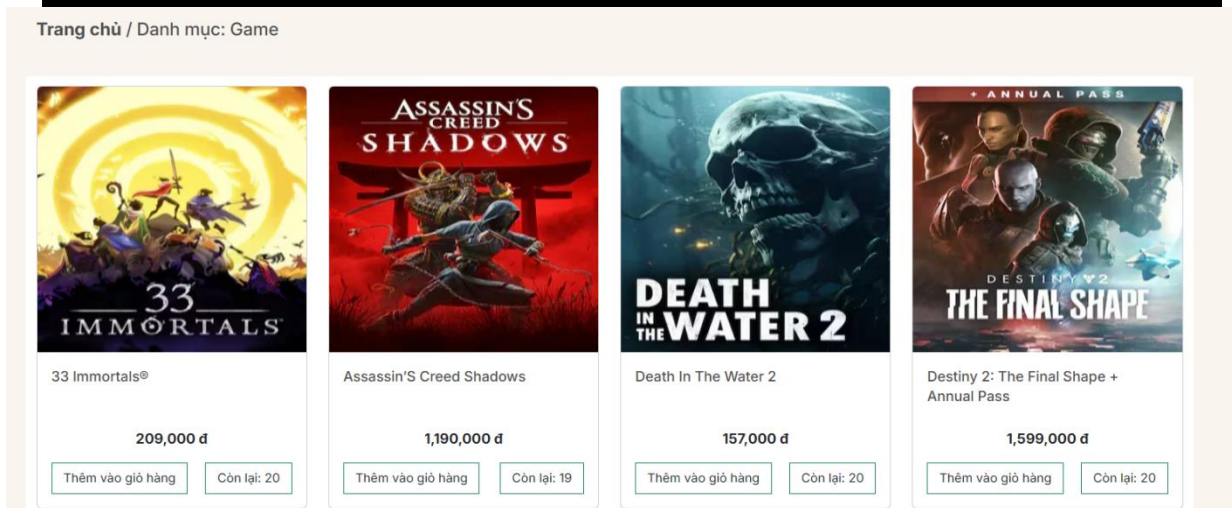
Bạn đã có tài khoản? [Đăng nhập](#)



Hình 4.5 Giao diện trang đăng ký

4.1.4 Giao diện trang sản phẩm

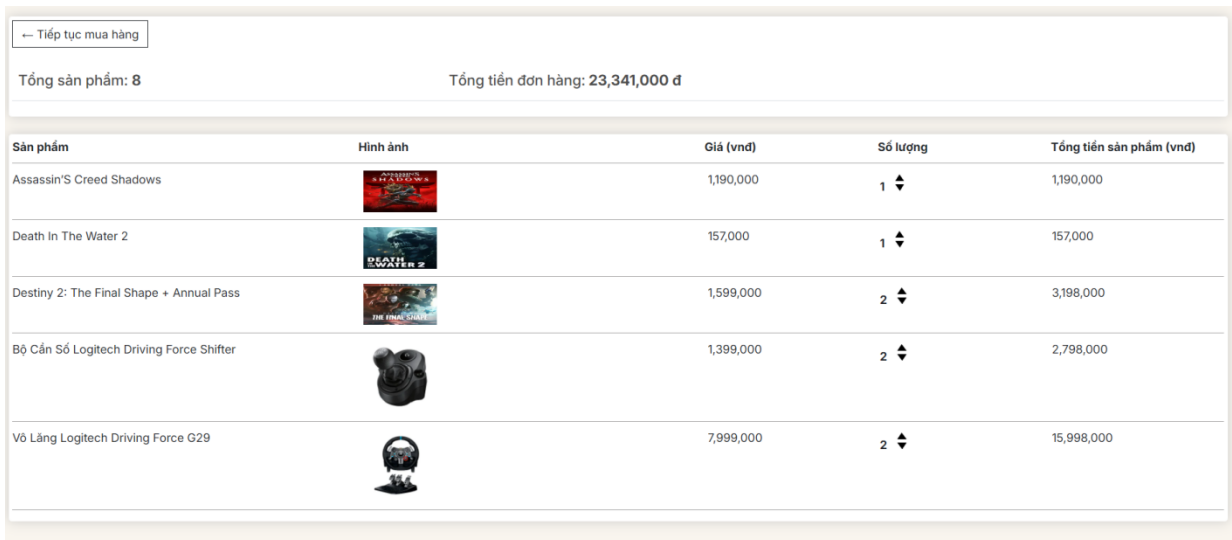
Đây là giao diện trang chứa tất cả sản phẩm của danh mục mà người dùng đã chọn từ thanh menu danh mục ở bên trái hoặc mục “Danh mục sản phẩm” trên thanh menu chính của trang chủ. Tại đây người dùng có thể xem được tên, giá, số lượng còn lại, bấm thêm sản phẩm vào giỏ hàng. Nếu muốn xem sản phẩm của danh mục khác có thể click vào mục “Danh mục sản phẩm” trên thanh menu để xem danh sách các danh mục hiện có.



Hình 4.6 Giao diện trang sản phẩm theo danh mục

4.1.5 Giao diện trang giỏ hàng

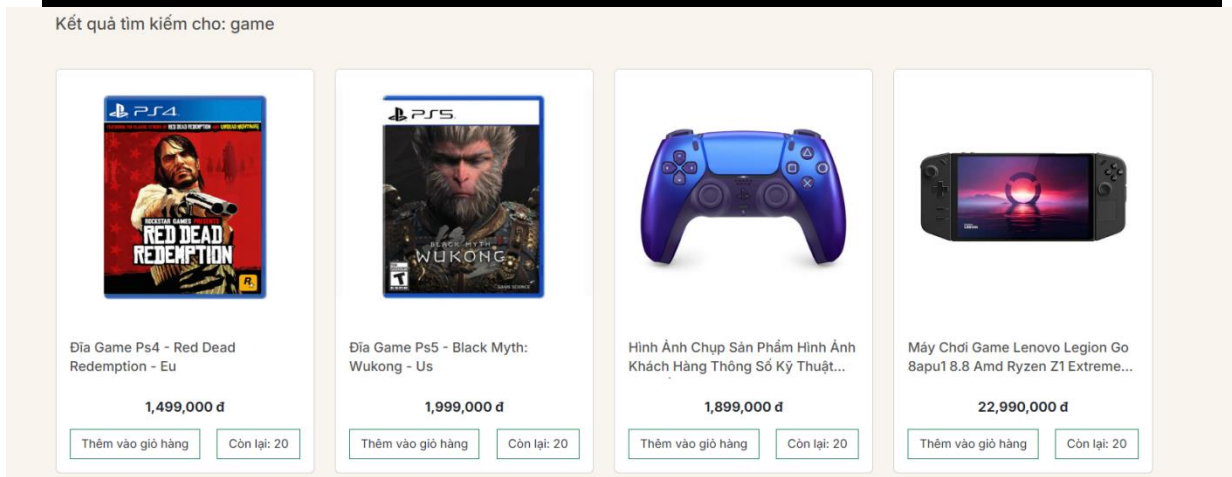
Nếu người dùng đã đăng nhập và chọn vào icon giỏ hàng trên thanh menu sẽ dẫn đến trang giỏ hàng của mình. Trang này hiển thị thông tin về từng sản phẩm bạn đã thêm vào, tổng số sản phẩm đã thêm và tổng giá trị đơn hàng.



Hình 4.7 Giao diện trang giỏ hàng

4.1.6 Giao diện trang kết quả tìm kiếm

Người dùng có thể tìm kiếm sản phẩm theo tên sản phẩm thông qua nút tìm kiếm ở thanh menu.



Hình 4.8 Giao diện trang kết quả tìm kiếm sản phẩm

4.1.7 Giao diện trang giới thiệu

Khi người dùng bấm vào mục Giới thiệu trên thanh menu sẽ dẫn đến trang giới thiệu.



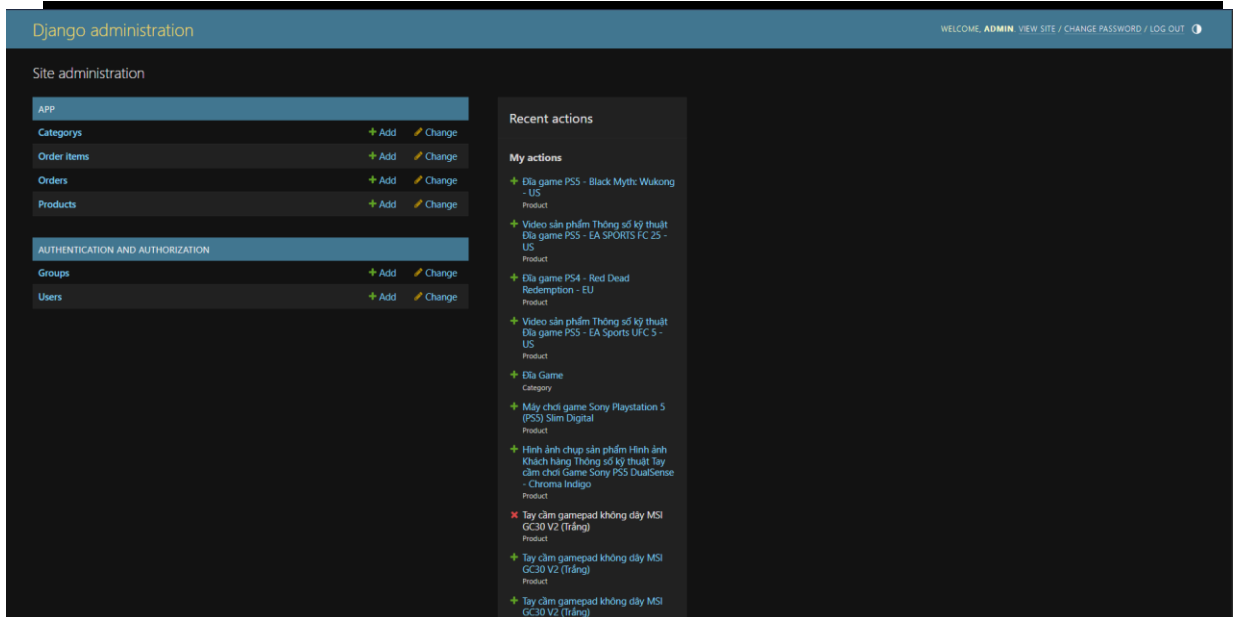
Hình 4.9 Giao diện trang giới thiệu website

4.2 Giao diện dành cho người quản trị

Để truy cập vào trang quản trị cần thêm /admin vào cuối địa chỉ của trang web, thường là <http://127.0.0.1:8000/admin>. Hoặc có thể đăng nhập bằng tài khoản admin với tư cách là superuser để điều hướng tự động vào trang quản trị.

4.2.1 Trang chủ quản trị

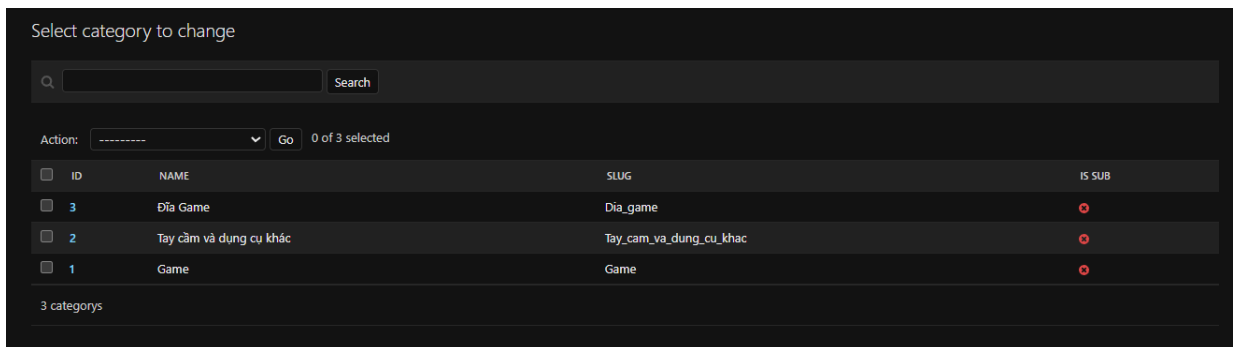
Là trang đầu tiên khi truy cập vào trang quản trị, gồm các models đã tạo như Category, Product, Order, OrderItem, User và Group được django tự động tạo. Phía bên phải là các hành động quản trị gần đây.



Hình 4.10 Giao diện trang chủ quản trị

4.2.2 Danh sách danh mục

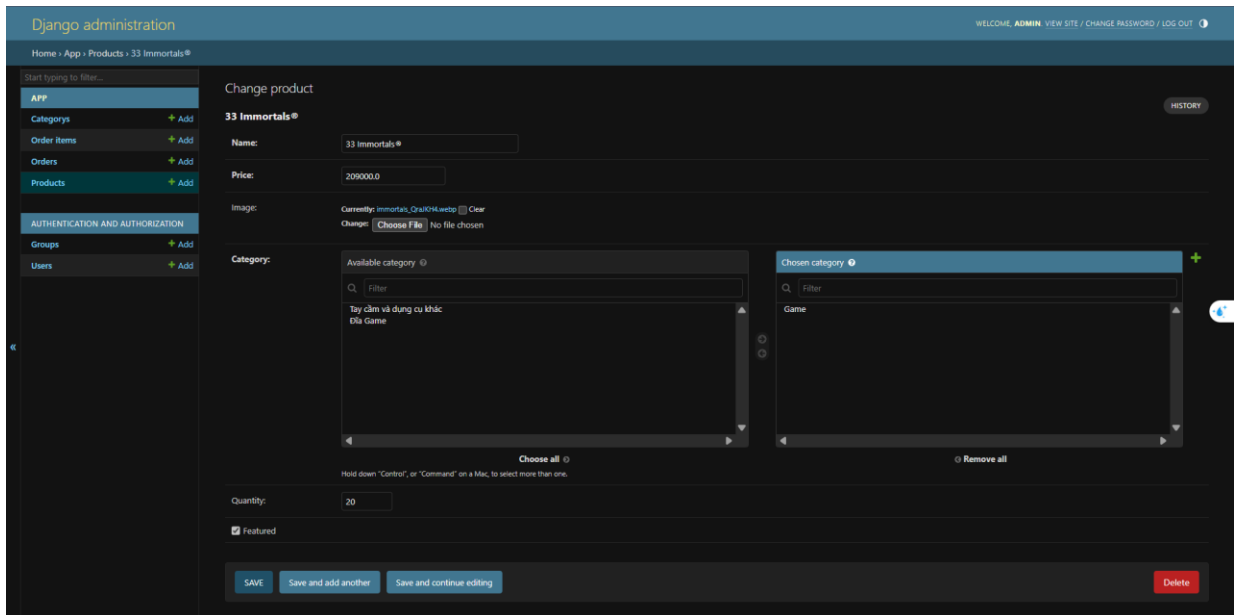
Chứa tất cả danh mục và bấm vào danh mục nào sẽ hiện sản phẩm danh mục đó.



Hình 4.11 Giao diện quản lý danh mục

4.2.3 Thêm, sửa, xóa sản phẩm

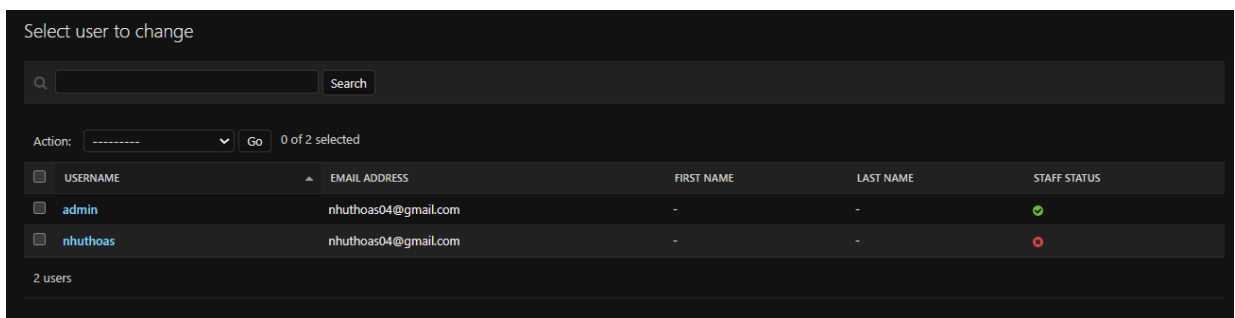
Người quản trị có thể thêm, sửa thông tin về sản phẩm nào đó. Nếu muốn xóa có thể chọn Delete để xóa sản phẩm. Khi một sản phẩm được xóa thì nó cũng tự được xóa trong phần quản lý danh mục của nó.



Hình 4.12 Giao diện quản lý sản phẩm

4.2.4 Quản lý người dùng

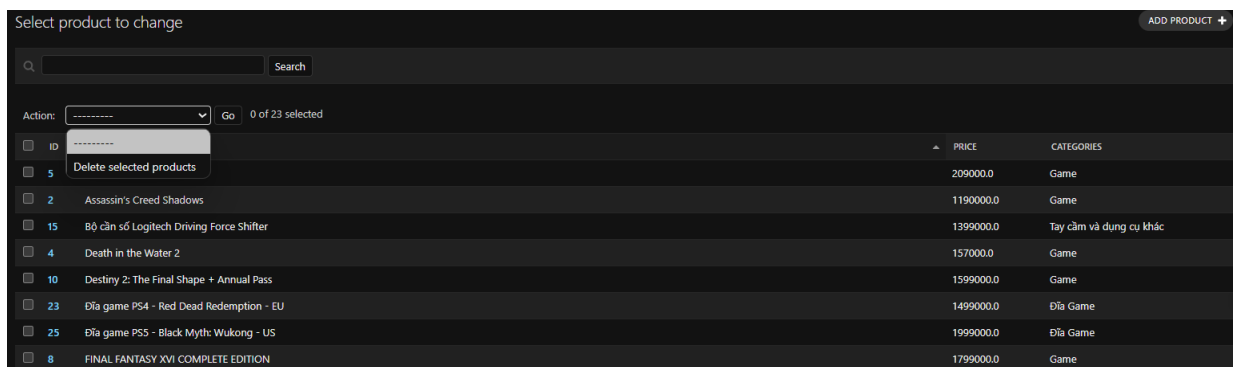
Người quản trị có thể thêm, sửa, xóa thông tin của người dùng.



Hình 4.13 Giao diện quản lý người dùng

4.2.5 Quản lý các sản phẩm được đặt hàng

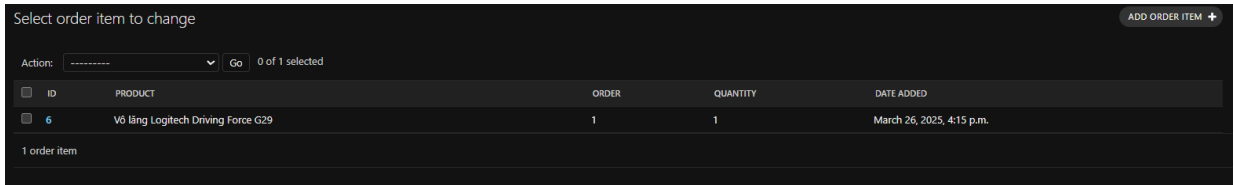
Người quản trị có thể thêm, sửa, xóa các sản phẩm đã được người dùng đặt.



Hình 4.14 Giao diện quản lý giỏ hàng

4.2.6 Quản lí các đơn hàng

Người quản trị có quyền thêm, sửa thông tin đơn hàng như đổi user đặt đơn đó và xóa đơn hàng của user.



Hình 4.15 Giao diện quản lí đơn hàng

4.2.7 Trang báo cáo thống kê

Chỉ khi đăng nhập với tài khoản quản trị thì trên thanh menu mới xuất hiện mục “Báo cáo thống kê”. Trang này sẽ cho người quản trị biết các thông tin như tổng doanh thu, tổng tồn kho, top các sản phẩm bán chạy và tồn kho nhiều nhất. Có các biểu đồ được xuất dưới dạng hình ảnh sau khi đã xử lí dữ liệu và vẽ trong View.



Hình 4.16 Giao diện trang báo cáo thống kê

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết quả đạt được

Đề tài "Tìm hiểu ngôn ngữ lập trình Python. Xây dựng hệ thống bán game và trò chơi điện tử" đã xây dựng thành công hệ thống quản lý bán game với Python và framework Django. Hệ thống đáp ứng các yêu cầu cơ bản của người dùng và quản trị viên như quản lý sản phẩm, giỏ hàng, đơn hàng và báo cáo thống kê. Giao diện thân thiện, dễ sử dụng, và hệ thống dễ bảo trì nhờ mô hình MTV của Django.

Đề tài giúp em làm quen và sử dụng Django để xây dựng phần backend, quản lý cơ sở dữ liệu, xử lý các yêu cầu HTTP và tạo giao diện người dùng bằng Django Templates. Từ đó hiểu rõ hơn về quy trình phát triển ứng dụng web, từ việc thiết kế hệ thống đến triển khai và vận hành.

5.2 Hạn chế và hướng phát triển

- Hiện tại, người dùng chỉ mới có thể thêm được sản phẩm vào giỏ nên cần phát triển thêm chức năng thanh toán online và đặt hàng, tích hợp các cổng thanh toán như PayPal, VNPay, hoặc MOMO.
- Người dùng chưa thể chỉnh sửa thông tin cá nhân. Điều này có thể làm giảm khả năng tương tác và tùy chỉnh của người dùng. Cần thêm trang chỉnh sửa thông tin cá nhân và đổi mật khẩu, đồng thời bổ sung tính năng xem lịch sử mua hàng.
- Báo cáo thống kê còn cơ bản có thể làm cho việc phân tích và ra quyết định kinh doanh chưa được chính xác, hợp lý. Cần có thêm báo cáo doanh thu theo thời gian (ngày, tháng, quý), phân tích xu hướng bán hàng và thống kê sản phẩm.
- Tính năng tìm kiếm cần được tối ưu hóa để giúp người dùng dễ dàng tìm thấy sản phẩm mong muốn. Điều này có thể bao gồm cải thiện thuật toán tìm kiếm và thêm các tiện ích như lọc, sắp xếp để tối ưu hóa trải nghiệm tìm kiếm.
- Cần gia tăng số lượng sản phẩm trong hệ thống để tăng sự đa dạng về mặt sản phẩm cho người dùng có nhiều sự lựa chọn mua sắm hơn.
- Do đề tài phần lớn tập trung vào phần back-end của hệ thống nên phần giao diện front-end còn đơn giản, chưa quá bắt mắt, sinh động. Có thể nâng cấp giao diện người dùng với thiết kế hiện đại, sử dụng Bootstrap hoặc CSS để tạo trải nghiệm trực quan và sinh động hơn.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Rossum, Guido Van, "The History of Python: A Brief Timeline of Python", (20/01/2009). Có tại: <https://pythonhistory.blogspot.com/2009/01/brief-timeline-of-python.html>.
- [2] Cao Lê Viết Tiến, "Django là gì? Hướng dẫn cài đặt Django dành cho người mới," Vietnix. Có tại: <https://vietnix.vn/django-la-gi/>.
- [3] Hưng Phạm, "Báo cáo môn Python Django," Studocu, (2023). Có tại: <https://www.studocu.vn/vn/document/truong-dai-hoc-cong-nghe-thong-tin-va-truyen-thong/phan-tich-thiet-ke-huong-doi-tuong/bao-cao-mon-python-django/92330996>.
- [4] Django Software Foundation, "The web framework for perfectionists with deadlines." Django. Có tại: <https://www.djangoproject.com/>.