**Mô tả Framework**

**cho**

**<Website nghe nhạc trực tuyến>**

## Phiên bản 1.1

## Người viết <Đàm Quang Đạt>

## <Đại học công nghệ - Đại học quốc gia Hà Nội>

## 30-03-2024

**Table of Contents**

[1 Giới thiệu Framework 3](#_Toc163737210)

[2 Các thành phần 3](#_Toc163737211)

[3 Cách sử dụng 4](#_Toc163737212)

[4 Ưu điểm 4](#_Toc163737213)

[5 Hạn chế 4](#_Toc163737214)

[6 Kết luận 5](#_Toc163737215)

[7 Các công nghệ khác 5](#_Toc163737216)

**Lịch sử sửa đổi**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Ngày** | **Lí do sửa đổi** | **Phiên bản** |
| Quang Đạt | 29/3/2024 | Phiên bản đầu | 1.0 |
| Quang Đạt | 11/4/2024 | Csdl NoSQL không tương thích tốt với Django, thiếu các công nghệ khác. Thay đổi csdl sang SQLite | 1.1 |

**Quy ước tài liệu:**

**Tiêu đề: Cambria /Font cỡ 16/Bold/Underlined**

**Tiêu đề phụ: Cambria/Font cỡ 14/Bold**

Văn bản: Cambria/Font cỡ 14

# Giới thiệu Framework

Django là một framework mã nguồn mở phát triển web mạnh mẽ và linh hoạt được xây dựng bằng Python. Xây dựng từ năm 2003 với nguyên tắc “Don’t repeat yourself (DRY)”, nó được thiết kế để giúp việc đưa ứng dụng từ ý tưởng thành hiện thực càng nhanh càng tốt, và tránh phải lặp lại công việc.

Mục đích của Django trong dự án là cung cấp một cách tiếp cận hiệu quả để xây dựng ứng dụng web nhanh chóng, thực dụng.

# Các thành phần

Django có kiến trúc dựa trên khung MVT (Model – View - Template) vì thế, nó hoàn toàn có thể độc lập cung cấp đầy đủ công cụ để tạo ra một Website từ front-end cho đến back-end.

## Model

* Model đại diện cho lớp dữ liệu. Nó định nghĩa cấu trúc của các bảng cơ sở dữ liệu và xử lý việc thao tác dữ liệu.
* Django cung cấp một tập các APIs để tiện lợi cho các Developer tương tác với cơ sở dữ liệu. Thực hiện các thao tác CRUD (Tạo, Đọc, Cập nhật, Xóa) và bảo đảm tính toàn vẹn dữ liệu.

## View

* View là một file chứa các hàm Python, xử lý logic để xử lý các yêu cầu và trả về các phản hồi từ web một cách phù hợp. Một phản hồi có thể là HTML, XML hoặc một “lỗi 404”.
* View tương tác với các Model để yêu cầu các dữ liệu, chuẩn bị nội dung dữ liệu để hiển thị,...

## Template

* Template định nghĩa cách dữ liệu nên được trình bày cho người dùng, nó là mẫu được thiết kế trước để tạo ra các file HTML, XML.
* Template tách biệt logic trình bày khỏi View, cho phép tạo các bố cục có thể tái sử dụng.
* Template bao gồm các biến (variables), thẻ (tags), điều kiện (filters),...

Tóm lại, kiến trúc MVT của Django thúc đẩy tính mô-đun, khả năng bảo trì và sự tách biệt của các thành phần. Bằng cách hiểu rõ các thành phần này, có thể tạo ra các ứng dụng vững chắc, vừa có chức năng vừa được tổ chức tốt.

# Cách sử dụng

Django được sử dụng để xây dựng 3 thành phần chính: **Cơ sở dữ liệu**, **giao diện người dùng** và một **phương tiện giúp giao tiếp** giữa 2 thành phần trên.

## Model

Định nghĩa và tương tác với các bảng thông tin của người dùng, bài hát,... trong **cơ sở dữ liệu.**

## View

Xử lí logic, nhận và giải quyết các yêu cầu từ **giao diện người dùng** và từ đó tạo ra các thay đổi tương ứng đối với **cơ sở dữ liệu.**

## Template

Giúp tạo ra các files HTML hiển thị bằng cách sử dụng các mẫu, quy tắc có sẵn và cú pháp của cả HTML và Django.

# Ưu điểm

* **Nhanh**: Django được thiết kế để giúp đưa các ứng dụng từ ý tưởng đến hoàn thành càng nhanh càng tốt..
* **Có đầy đủ các thư viện/module cần thiết**: Với cộng đồng phát triển lớn và lâu đời, Django bao gồm hàng tá các tính năng bổ sung mà ta có thể sử dụng để xử lý các tác vụ phát triển Web phổ biến. Django chăm sóc xác thực người dùng, quản trị nội dung, bản đồ trang web, nguồn cấp dữ liệu RSS và nhiều tác vụ khác - ngay lập tức.
* **Đảm bảo về tính bảo mật**: Django rất coi trọng vấn đề bảo mật và giúp tránh được nhiều lỗi bảo mật phổ biến, chẳng hạn như kịch bản chéo trang, giả mạo yêu cầu chéo trang. Hệ thống xác thực người dùng của nó cung cấp một cách an toàn để quản lý tài khoản và mật khẩu người dùng.
* **Khả năng mở rộng tốt**: Do được sử dụng kiến trúc MTV và phong cách xây dựng gọn gàng, tránh lặp lại, Django có khả năng linh hoạt được đánh giá cao.

# Hạn chế

* **Tùy biến hạn chế:** dù có nhiều tính năng, nhưng việc tùy chỉnh theo yêu cầu có thể khó khăn tùy vào độ phức tạp và chuyên sâu
* **Không phù hợp với các dự án đơn giản:**  Đối với các dự án này, Django có thể quá mạnh mẽ và phức tạp, tăng thời gian phát triển không cần thiết.
* **Khó làm quen:** Django có độ dốc học tập tương đối lớn, khó với người mới học lập trình web, cần thời gian để làm quen với cấu trúc và quy tắc.

# Kết luận

Django cung cấp một cách tiếp cận linh hoạt và mạnh mẽ để xây dựng một ứng dụng web full-stack. Với cấu trúc rõ ràng và các tính năng tích hợp sẵn, nó đóng góp vào việc tăng tốc quá trình phát triển và cung cấp một nền tảng ổn định và an toàn cho dự án.

# Các công nghệ khác

## Cơ sở dữ liệu Sqlite

* **Nhẹ nhàng và dễ sử dụng**: SQLite là một cơ sở dữ liệu nhẹ nhàng, không yêu cầu cài đặt hay quản lý riêng biệt. Nó có thể được tích hợp trực tiếp vào ứng dụng Django mà không cần cấu hình phức tạp.
* **Dễ dàng triển khai**: Vì SQLite không yêu cầu máy chủ riêng biệt, nó rất dễ triển khai. Ta có thể đóng gói ứng dụng Django cùng với cơ sở dữ liệu SQLite và triển khai nó một cách dễ dàng trên các môi trường khác nhau.
* **Thích hợp cho các ứng dụng nhỏ và phát triển nhanh**: Trong các dự án Django nhỏ, nơi yêu cầu cơ sở dữ liệu nhẹ và không phức tạp, SQLite là lựa chọn lý tưởng.
* **Tích hợp mạnh mẽ với Django**: Django hỗ trợ SQLite như một backend cơ sở dữ liệu chính thức. Ta có thể định nghĩa các model và tạo csdl một cách dễ dàng thông qua các lệnh quản lý Django như **makemigrations** và **migrate**.
* **Phát triển và kiểm thử**: SQLite là một lựa chọn tốt cho việc phát triển và kiểm thử dự án Django.

## Ngôn ngữ front-end

### **HTML**: Ta có thể tạo các mẫu HTML cho các trang web trong Django bằng cách sử dụng template engine của Django. Để sử dụng HTML trong Django, tạo các file template HTML trong thư mục **templates** của ứng dụng Django. Sau đó, render các template này trong views và trả về chúng như phản hồi từ server.

### **CSS**: Để sử dụng CSS trong Django, Ta có thể tạo các file CSS và lưu chúng trong thư mục **static** của ứng dụng Django. Sau đó, liên kết các file CSS này với các template HTML bằng cách sử dụng các thẻ **<link>** hoặc inline CSS trong các template.

### **JavaScript**: Có thể sử dụng JavaScript để thêm các tính năng tương tác động vào ứng dụng Django. Tương tự như CSS, tạo các file JavaScript và lưu chúng trong thư mục **static** của ứng dụng Django. Sau đó, liên kết các file JavaScript này với các template HTML bằng cách sử dụng các thẻ **<script>** hoặc inline JavaScript trong các template.

## Công cụ hỗ trợ phát triển

### **GitHub**: GitHub là một nền tảng hosting mã nguồn mở cho các dự án phần mềm. Trong dự án Django, GitHub được sử dụng để lưu trữ mã nguồn, quản lý các phiên bản khác nhau của dự án thông qua Git. GitHub cũng cung cấp các tính năng như issue tracking, pull requests, và collaboration tools để hỗ trợ việc làm việc nhóm và phát triển phần mềm một cách hiệu quả.

### **Git**: Git là một hệ thống quản lý phiên bản phổ biến, được sử dụng để theo dõi các thay đổi trong mã nguồn của dự án. Giúp việc quản lí phiên bản dự án cá nhân hiệu quả.

### **PyCharm**: PyCharm là một IDE mạnh mẽ cho Python, cung cấp các công cụ để phát triển, debug và quản lý dự án Python. Trong dự án Django, PyCharm được sử dụng để viết mã nguồn Python, tạo và chỉnh sửa các template HTML và các file CSS, JavaScript. Nó cũng cung cấp tích hợp với Git để quản lý phiên bản của dự án.

### **Visual Studio Code**: Visual Studio Code (VS Code) là một trình biên tập mã nguồn mở và miễn phí, được sử dụng rộng rãi trong cộng đồng phát triển phần mềm. Trong dự án Django, VS Code cung cấp các tính năng như syntax highlighting, code completion, và debugging cho Python, HTML, CSS và JavaScript. Nó cũng hỗ trợ tích hợp với Git và các tiện ích mở rộng giúp tăng cường hiệu suất làm việc.