Tổng quan đề tài:

* Tên đề tài: Xây dựng website quản lý quy trình thực hiện dự án cho công ty ANNGO bằng Python Django.
* Trong quá trình phát triển dự án công nghệ thông tin thì việc phải quản lý giai đoạn, đội ngũ phát triển, chi phí là chìa khóa để phát triển một dự án thành công. Tuy nhiên nếu thực hiện chúng một cách thủ công thì sẽ vô cùng khó khăn, do đó việc sử dụng một phần mềm quản lý quy trình thực hiện dự án sẽ giúp đỡ rất nhiều cho những người quản lý dự án.
* Lý do chọn Python Django để xây dựng website là vì Django là một framework rất nổi tiếng trong việc phát triển website bằng Python. Django cung cấp sẵn giao diện cho Admin, kiểu dữ liệu trong cơ sở dữ liệu đa dạng, đặc biệt dễ dàng trong việc thiết kế cơ sở dữ liệu.

Mục tiêu đề tài:

* Phát triển website giúp người quản lý theo dõi quá trình thực hiện dự án được giao bởi công ty:
* Phát triển website có đầy đủ chức năng giúp người quản lý dự án quản lý quá trình thực hiện dự án: tạo lịch thực hiện, phân chia nhiệm vụ, theo dõi tiến độ hoàn thành, phân quyền những người tham gia dự án.
* Giao tiếp và trao đổi trong dự án:
* Phát triển kênh giao tiếp riêng giữa những người thực hiện dự án với nhau: nhắn tin, gọi video.
* Cho phép những người thực hiện dự án chia sẻ với nhau tài nguyên dữ liệu phần mềm.
* Xác thực và quản lý người dùng:
* Phát triển chức năng đăng ký, đăng nhập. Phân quyền người dùng theo vai trò khi đăng nhập.

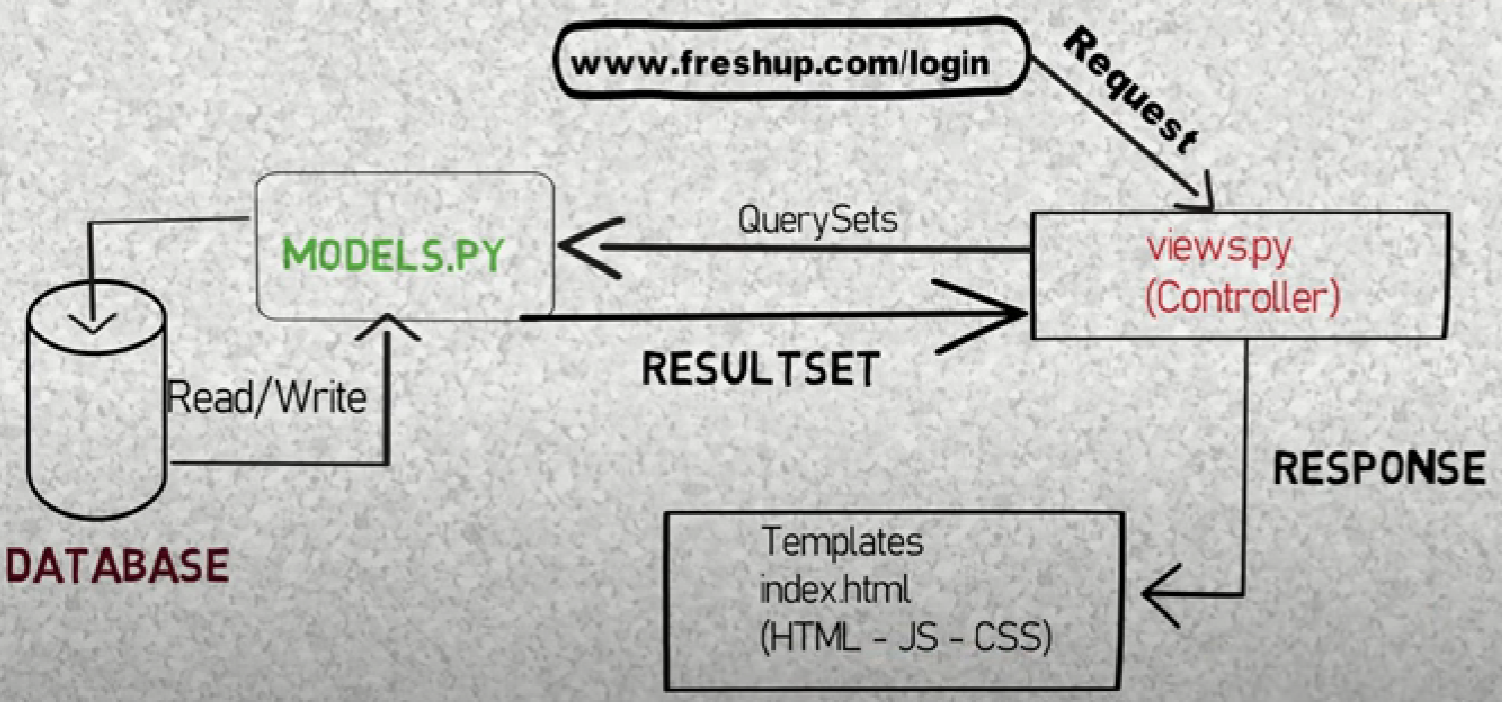
Cơ sở lý thuyết:

* 1. Django

*Django* là một framework web mạnh mẽ của ngôn ngữ lập trình Python, được thiết kế để giúp nhà phát triển xây dựng ứng dụng web nhanh chóng và dễ bảo trì.

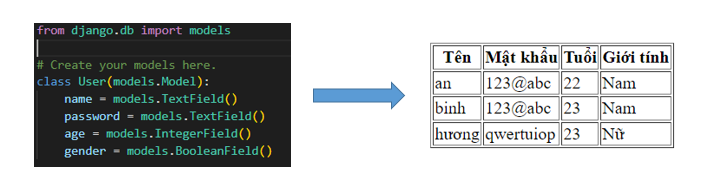
**Một số đặc điểm của Django:**

* *Mô hình MVT (Model-View-Template):* Django sử dụng mô hình MVT, nơi Model đại diện cho dữ liệu và cơ sở dữ liệu, View xử lý logic xử lý và hiển thị, còn Template quản lý giao diện người dùng.



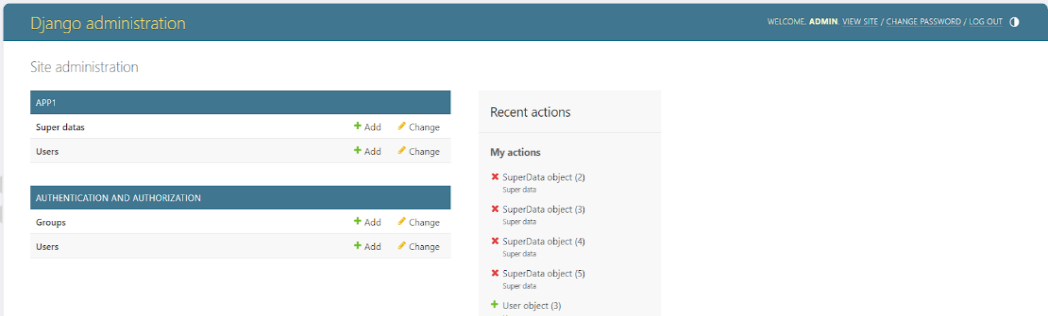
*Mô hình MVT*

* *ORM (Object-Relational Mapping):* Django cung cấp một ORM mạnh mẽ, giúp tương tác với cơ sở dữ liệu mà không cần viết SQL trực tiếp. Điều này giúp làm giảm độ phức tạp và tăng tính di động của ứng dụng.



*Model ORM*

* *Giao diện Admin tích hợp*: Django cung cấp một giao diện quản trị tích hợp sẵn, giúp quản lý dữ liệu của ứng dụng mà không cần phải xây dựng giao diện quản trị riêng.



*Giao diện Admin tích hợp*

* *Định tuyến URL:* Hệ thống định tuyến URL của Django cho phép bạn dễ dàng xác định cách các URL được ánh xạ vào các hàm xử lý (views).
* *Hỗ trợ biểu mẫu (form):* Django cung cấp các lớp biểu mẫu, giúp xử lý dữ liệu đầu vào từ người dùng một cách hiệu quả và an toàn.

**Các bước cơ bản trong Django:**

* *Tạo dự án:* Dùng lệnh “django-admin” trong terminal để tạo 1 dự án mới, nếu chưa cài đặt django thì dùng lệnh “pip install django” để cài đặt.
* *Tạo ứng dụng:* Sử dụng “python manage.py startapp tên\_ứng\_dụng” để tạo một ứng dụng mới trong dự án. Nó sẽ tự tạo 1 folder có tên là tên ứng dụng bạn đặt, nó sẽ có sẵn các file:
  + admin.py: sử dụng để định nghĩa cấu hình liên quan đến giao diện admin của ứng dụng như chỉnh sửa database, tùy chỉnh hiển thị, bộ lọc, tìm kiếm,…
  + models.py: sử dụng để định nghĩa cấu trúc dữ liệu của ứng dụng.
  + views.py: sử dụng để viết các hàm xử lý logic và trả lại kết quả.
* *Xây dựng mô hình (Models*): Xác định các lớp mô hình để đại diện cho dữ liệu và cấu trúc cơ sở dữ liệu.
* *Tạo Views:* Viết hàm xử lý trong views để xử lý logic và trả về kết quả cho người dùng.
* *Xây dựng Templates*: Tạo folder đặt tên là Templates (lưu ý: phải đặt tên đúng là Templates) bên trong folder ứng dụng. Folder này sẽ chứa các file HTML, CSS được dùng trong ứng dụng.
* *Xác định URL:* Định nghĩa các mẫu URL để ánh xạ đến các views tương ứng.
* *Chạy Migrations:* Sử dụng “python manage.py makemigrations” và “python manage.py migrations” để áp dụng các thay đổi vào cơ sở dữ liệu. Lặp lại 2 lệnh này bất cứ khi nào có thay đổi về  cấu trúc cơ sở dữ liệu.
* *Chạy Server:* Sử dụng “python manage.py runserver” để chạy máy chủ và kiểm tra ứng dụng

**Cơ sở dữ liệu của Django**

Cơ sở dữ liệu của Django được xây dựng dựa trên ORM (Object-Relational Mapping), giúp tương tác với cơ sở dữ liệu bằng cách sử dụng đối tượng Python thay vì truy vấn SQL trực tiếp.

**Một số khái niệm về cơ sở dữ liệu của Django:**

* *Models:* Models là các lớp Python đại diện cho các bảng trong cơ sở dữ liệu. Mỗi trường của mô hình tương ứng với một cột trong bảng.
* *Migration:* Migration là quá trình chuyển đổi các thay đổi trong models thành các bảng cơ sở dữ liệu thực tế.
* *Quan hệ:* Django hỗ trợ nhiều loại quan hệ giữa các models, chẳng hạn như ForeignKey (một-đến-nhiều), OneToOneField (một-một) và ManyToManyField (nhiều-đến-nhiều).
* *Django ORM:* Django ORM giúp thực hiện các thao tác cơ sở dữ liệu mà không cần phải viết SQL trực tiếp. Các thao tác tạo, sửa đổi, xóa và truy vấn dữ liệu được thực hiện một cách đơn giản dễ dàng.

**Các giá trị dùng để định nghĩa các trường:**

* *primary\_key:* dùng để biết có gán khóa chính cho trường này không.
* *null:* cho phép trường có nhận được giá trị null không.
* *blank:* cho phép trường được bỏ trống khi nhập form.
* *default:* gán giá trị mặc định cho trường.
* *unique:* đảm bảo giá trị của trường là duy nhất trong toàn bộ bảng.
* *max\_lenghth:* gán độ dài tối đa của chuỗi ký tự cho phép lưu trữ, chỉ sử dụng cho CharField.
* *choices:* gán trường chỉ nhận các giá trị đã được định nghĩa sẵn.
* *upload\_to:* gán vị trí lưu trữ file, chỉ dùng cho các trường như file, hình ảnh.
* *auto\_now* và *auto\_now\_add*: dùng để tự động cập nhật thời gian cho trường, chỉ dùng cho DateTimeField.
  + *auto\_now*: sẽ cập nhật thời gian khi bản ghi được tạo hoặc mỗi khi được cập nhật.
  + *auto\_now\_add*: chỉ cập nhật thời gian khi bản ghi được tạo.
* *on\_delete:* thường được dùng cho trường ForeignKey, cho biết hành động nào sẽ được thực hiện khi bản ghi liên kết với trường (đối tượng mẹ) bị xóa. Các giá trị mà *on\_delete* có thể nhận:
  + *models.CASCADE*: khi đối tượng mẹ bị xóa, tất cả các đối tượng con sẽ bị xóa.
  + *models.PROTECT*: ngăn việc xóa đối tượng  mẹ nếu còn tồn tại bất kỳ đối tượng con nào.
  + *models.SET\_NULL*: khi đối tượng mẹ bị xóa, tất cả các đối tượng con sẽ có giá trị là null, yêu cầu phải có null=True.
  + *models.SET\_DEFAULT*: khi đối tượng mẹ bị xóa, tất cả các đối tượng con sẽ có giá trị là giá trị default, yêu cầu gán giá trị default.
  + *models.SET*(set\_value): khi đối tượng mẹ bị xóa, tất cả các đối tượng con sẽ có giá trị là giá trị set\_value.
  + *models.DO\_NOTHING:* khi đối tượng mẹ bị xóa thì không làm gì cả. Lưu ý: chỉ sử dụng khi đảm bảo sẽ không phát sinh bất kỳ lỗi ràng buộc nào.

**Các trường và model dữ liệu được sử dụng trong dự án:**

* *AutoField*: được sử dụng để đại diện cho một cột cơ sở dữ liệu tự động tăng, thường được sử dụng để làm khóa chính, định danh duy nhất cho mỗi bản ghi trong một bảng.
* *TextField*: được sử dụng để lưu trữ dữ liệu văn bản dài, không giới hạn.
* *CharField*: được sử dụng để lưu trữ dữ liệu dạng chuỗi ký tự, có giới hạn.
* *EmailField*: được sử dụng để lưu trữ dữ liệu kiểu chuỗi địa chỉ email, tự động kiểm tra xem địa chỉ có đúng định dạng không khi nhập, lưu trữ.
* *DateTimeField*: được sử dụng để lưu trữ dữ liệu kiểu ngày, thời gian được lấy theo múi giờ (timezone) được định nghĩa trong setting.py.
* *BooleanField*: được sử dụng để lưu trữ dữ liệu kiểu logic, True hoặc False.
* *ImageField*: được sử dụng để lưu trữ dữ liệu kiểu file hình ảnh.
* *FileField*: được sử dụng để lưu trữ dữ liệu kiểu file bất kỳ.
* *ForeignKey*: được dùng để gán liên kết khóa ngoài.
* *models.TextChoices:* là một lớp model cho phép định nghĩa các lựa chọn cho các trường CharField hoặc TextField, giúp mã hóa một số lựa chọn cụ thể. Một thuộc tính của lớp TextChoices có 2 giá trị là giá trị (value) và nhãn (label). Giá trị (value) sẽ là giá trị thật được lưu trong bản ghi cơ sở dữ liệu còn nhãn (label) thường sẽ được dùng để hiển thị trong giao diện.
  1. Jinja

*Jinja* là một hệ thống template engine cho Python. Nó được sử dụng để tách logic xử lý và hiển thị trong ứng dụng web, giúp code trở nên sáng sủa và dễ hiểu. Jinja được sử dụng rộng rãi trong các framework web như Flask và Django.

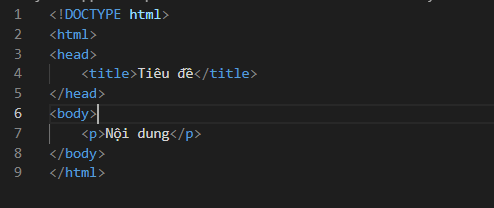
Khi cài đặt Django, Jinja cũng sẽ được cài đặt theo.

**Một số điều cơ bản trong Jinja:**

* *Giá trị và câu lệnh:* Các biểu thức được bao bọc trong cặp dấu {{ … }} để hiển thị giá trị, và các câu lệnh điều khiển như vòng lặp và điều kiện được sử dụng với cặp {% … %}.
  + {{ variable }} : dùng để hiển thị giá trị biểu thức
  + {% for giá\_trị in chuỗi\_giá\_trị %} code\_html {% endfor %} : vòng lặp
  + {% if điều\_kiện %} code\_html {% else %} code\_html {% endif %} : điều kiện
* *Kế thừa và mở rộng:* Jinja hỗ trợ cơ chế kế thừa, cho phép tái sử dụng các mẫu và mở rộng chúng để tạo ra các mẫu lớn hơn.
  + {% extends “tên file html dùng để mở rộng”%}
  + {% block tên\_block %} code\_html {% endblock %} : trong file html dùng để mở rộng cũng phải có block trùng tên thì mới truyền code html sang được.
* *Bảo mật:* Jinja cũng chú trọng đến bảo mật. Nó giúp ngăn chặn các tấn công injection và cung cấp các cơ chế để làm cho template an toàn.
  1. HTML

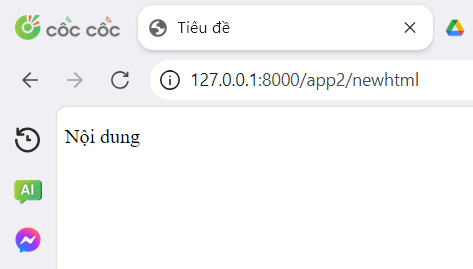
*HTML (HyperText Markup Language):* HTML là một ngôn ngữ đánh dấu được sử dụng để tạo và định nghĩa cấu trúc của các trang web. Nó sử dụng các thẻ (tags) để mô tả các phần khác nhau của trang web, chẳng hạn như tiêu đề, đoạn văn bản, hình ảnh, liên kết, bảng, biểu mẫu, v.v. Các trình duyệt web sử dụng HTML để hiển thị nội dung của trang web một cách có tổ chức và có ý nghĩa.

Mỗi thẻ HTML được bắt đầu bằng một dấu nhấn mở (<) và kết thúc bằng một dấu nhấn đóng (>). Nội dung của thẻ nằm giữa dấu nhấn mở và dấu nhấn đóng, và thường mô tả loại dữ liệu hoặc chức năng của phần tử đó.



1. Cấu trúc cơ bản của 1 file HTML

Kết quả khi hiển thị trên browser:

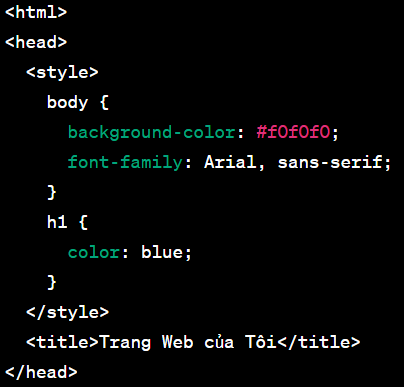


1. Kết quả file HTML trên
   1. CSS

*CSS (Cascading Style Sheets):* CSS là một ngôn ngữ được sử dụng để mô tả cách trang web được hiển thị. Nó định nghĩa các quy tắc và kiểu dáng cho các phần tử HTML, giúp điều chỉnh màu sắc, font chữ, kích thước, khoảng cách, vị trí và nhiều thuộc tính khác của các phần tử trang web. CSS cho phép bạn tách rời phần nội dung (HTML) và phần kiểu dáng, giúp làm cho mã nguồn trang web trở nên dễ bảo trì và tái sử dụng.

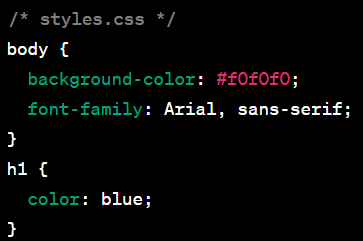
**Các cách để tích hợp CSS vào HTML:**

* Sử dụng thẻ style trong phần head



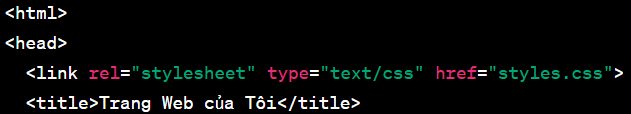
1. Ví dụ sử dụng thẻ style

* Sử dụng thẻ link để liên kết với CSS tạo riêng
* Tạo 1 file có tên là *style.css* như sau



1. File styles.css

* Thêm thẻ <link> dẫn đến file css trên



1. Tích hợp file vào HTML

* Sử dụng thẻ style trực tiếp lên từng phần tử



1. Ví dụ dùng thẻ style trực tiếp
   1. Java script

*JavaScript* là một ngôn ngữ lập trình thông dịch (interpreted programming language) được sử dụng chủ yếu để tạo ra các trang web tương tác và động. JavaScript được phát triển ban đầu bởi Netscape và sau đó được tiêu chuẩn hóa bởi Ecma International trong tiêu chuẩn ECMA-262. JavaScript là một phần của các trình duyệt web phổ biến như Chrome, Firefox, Safari, và Edge, và được sử dụng để thêm các tính năng động vào các trang web, như hiển thị thông báo, thay đổi nội dung của trang web mà không cần tải lại, và xử lý các sự kiện như nhấp chuột và gõ phím. JavaScript cũng có thể được sử dụng để tạo ra các ứng dụng web độc lập, nhưng nó thường được sử dụng kết hợp với HTML và CSS để tạo ra các trang web.

**Các kiểu dữ liệu trong Java script:**

* *Number*: Kiểu dữ liệu này biểu diễn các số, bao gồm cả số nguyên và số thập phân.
* *String*: Kiểu dữ liệu này biểu diễn các chuỗi ký tự. Chuỗi được bao quanh bởi dấu nháy đơn hoặc dấu nháy kép.
* *Boolean*: Kiểu dữ liệu này chỉ có hai giá trị: true và false
* *Undefined*: Kiểu dữ liệu này đại diện cho một biến chưa được gán giá trị.
* *Null*: Kiểu dữ liệu này đại diện cho một giá trị không tồn tại hoặc không hợp lệ.
* *Object*: Kiểu dữ liệu này đại diện cho một đối tượng, một nhóm các thuộc tính và phương thức.
* *Array*: Kiểu dữ liệu này đại diện cho một mảng các giá trị. Mảng có thể chứa bất kỳ kiểu dữ liệu nào.
* *Function*: Kiểu dữ liệu này đại diện cho một hàm.
* *Symbol*: Kiểu dữ liệu này đại diện cho một giá trị duy nhất không thể thay đổi được.
* *BigInt*: Kiểu dữ liệu này cho phép bạn làm việc với số nguyên lớn hơn 2^53 - 1.
* *Date*: Kiểu dữ liệu này đại diện cho một ngày và thời gian cụ thể.
* *RegExp*: Kiểu dữ liệu này đại diện cho một biểu thức chính quy.

Khi được sử dụng trong HTML, Java script được viết trong thẻ <script>…</script>

* 1. WebSocket

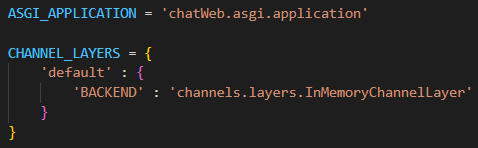
*WebSocket* là một công nghệ được sử dụng để thiết lập một kết nối liên tục, hai chiều giữa máy khách và máy chủ qua mạng Internet. Nó cho phép truyền tải dữ liệu trực tiếp và mạnh mẽ giữa các ứng dụng web.

**Các đặc điểm của WebSocket**

* *Kết nối liên tục*: WebSocket thiết lập một kết nối TCP hai chiều giữa máy khách và máy chủ, giúp truyền tải dữ liệu theo thời gian thực.
* *Giao thức dựa trên TCP*: WebSocket sử dụng giao thức TCP để thiết lập kết nối, giúp tránh được các vấn đề liên quan đến HTTP như các tiêu đề lặp đi lặp lại và chi phí liên kết.
* *Giao thức mở và tiêu chuẩn*: WebSocket là một tiêu chuẩn được định rõ bởi IETF (Internet Engineering Task Force), giúp đảm bảo tính tương thích giữa các ứng dụng trên các nền tảng và trình duyệt khác nhau.
* *Khả năng mở rộng*: WebSocket có khả năng mở rộng và tương thích tốt với các công nghệ và thư viện khác nhau như Node.js, Python, Ruby, và nhiều ngôn ngữ lập trình web khác.
* *Hiệu suất cao*: Do không cần thiết lập lại kết nối mỗi khi cần truyền tải dữ liệu, WebSocket giúp cải thiện hiệu suất và giảm độ trễ so với các phương pháp truyền tải truyền thống như long polling hay AJAX.
* *Bảo mật*: WebSocket hỗ trợ các lớp bảo mật như SSL/TLS để bảo vệ dữ liệu trong quá trình truyền tải, giúp ngăn chặn các cuộc tấn công như giả mạo hoặc đánh cắp thông tin.

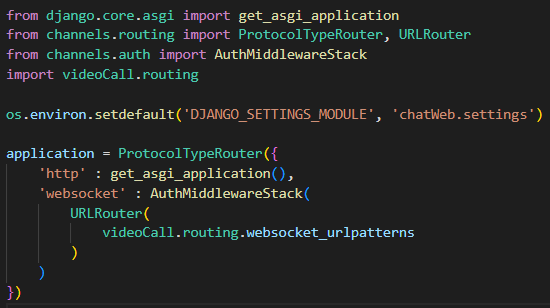
**Tích hợp WebSocket vào dự án Django**

* *Cài đặt channels và daphne*: channels và daphne là 2 thư viện hỗ trợ WebSocket của Python.
* Cài đặt chúng bằng lệnh pip installer channels và pip installer daphne trong terminal.
* Sau khi cài đặt thì ta cần vào file setting.py của dự án để khai báo sử dụng chúng. Thêm ‘channels’, ‘daphne’ và INSTALLED\_APPS. Ngoài ra cần thêm 1 số biến vào file setting. Sau khi thêm, thực hiện chạy python manage.py runserver mà thấy có thông báo Starting ASGI/Daphne thì đã cài đặt thành công.



Biến cần thêm

* Ngoài ra còn phải cấu hình lại file asgi.py để server có thể tiếp nhận tín hiệu từ WebSocket



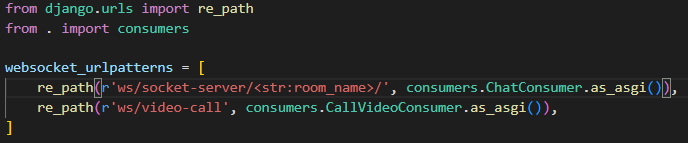
Ví dụ cấu hình

* *Cách sử dụng bên server*: ta cần tạo 2 file consumers.py và routing.py.
* File consumers.py sẽ chứa các lớp AsyncWebsocketConsumer hoặc WebsocketConsumer, để thực hiện các tác vụ khi có tín hiệu WebSocket từ client. Lớp AsyncWebsocketConsumer và WebsocketConsumer khá giống nhau, nhưng lớp AsyncWebsocketConsumer có thể thực hiện các hàm bất đồng bộ ‘async def’ giúp tăng hiệu suất so với việc sử dụng các hàm đồng bộ ‘def’ cổ điển của WebsocketConsumer.



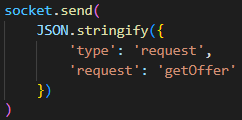
Ví dụ AsyncWebsocketConsumer

* File routing.py sẽ chứa các đường dẫn, mỗi đường dẫn sẽ kích hoạt 1 consumer cụ thể khi có tín hiệu từ client.



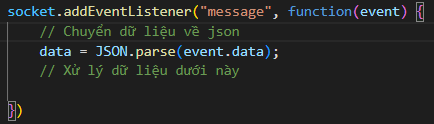
Ví dụ file routing

* *Cách sử dụng bên client*: trong JavaScript có sẵn lớp WebSocket hỗ trợ thực hiện gửi và nhận từ server.
* Tạo 1 đối tương WebSocket bằng câu lệnh <tên đối tượng > = new WebSocket(<chuỗi URL kết nối đến máy chủ>); . Chuỗi URL kết nối gồm 3 thành phần: Giao thức WebSocket, URL của trang web, đường dẫn endpoint.
* Giao thức WebSocket: có 2 loại giao thức không bảo mật ‘ws://’ và giao thức có bảo mật ‘wss://’. Khi dùng giao thức bảo mật, dữ liệu được mã hóa trước khi được gửi đi và giải mã sau khi đến máy chủ, đảm bảo tính riêng tư và bảo mật của dữ liệu.
* URL của trang web: bao gồm tên miền và cổng (nếu có) của trang web, có thể dễ dàng lấy nó thông qua đối tượng window, câu lệnh ‘window.location.host’.
* Đường dẫn endpoint: là đường dẫn đã khai báo trong routing.py
* Sau khi đã tạo xong đối tượng ta có thể gửi và nhận dữ liệu với server mà không phải tải lại trang.
* Gửi dữ liệu: sử dụng hàm send của WebSocket



Ví dụ gửi dữ liệu

* Nhận dữ liệu: sử dụng hàm addEventListener của WebSocket



Ví dụ nhận dữ liệu

* 1. WebRTC

WebRTC, viết tắt của Web Real-Time Communication, là một công nghệ cho phép trình duyệt web và ứng dụng truyền thông giao tiếp trực tuyến một cách trực quan và tiện lợi mà không cần cài đặt phần mềm bổ sung hay plugin. Được hỗ trợ bởi một số trình duyệt phổ biến như Google Chrome, Mozilla Firefox, và Opera, WebRTC cho phép truyền dữ liệu âm thanh, video, và dữ liệu trực tiếp giữa các máy tính mà không cần qua máy chủ trung gian.

Công nghệ này cung cấp các API (Application Programming Interfaces) để phát triển ứng dụng giao tiếp trực tuyến như video call, voice call, và chia sẻ dữ liệu trong thời gian thực. Một số tính năng chính của WebRTC bao gồm:

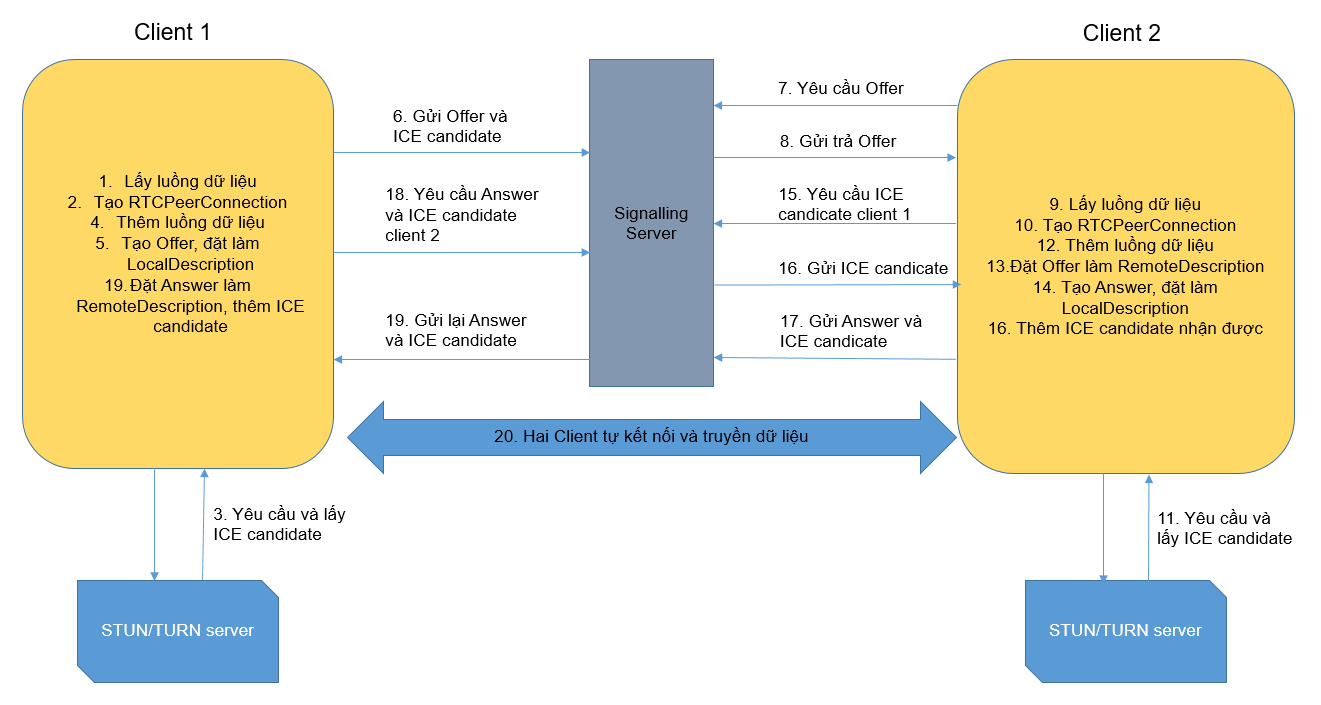
* Giao tiếp ngang hàng: WebRTC cho phép truyền dữ liệu trực tiếp giữa các máy tính, giảm thiểu độ trễ và tăng tính bảo mật bằng cách loại bỏ sự phụ thuộc vào máy chủ trung gian.
* Điều khiển truy vấn phần cứng: WebRTC cung cấp khả năng truy cập camera và microphone của người dùng thông qua trình duyệt, cho phép chúng ta tạo các ứng dụng giao tiếp trực tuyến mà không cần cài đặt phần mềm bổ sung.
* Mã hóa và bảo mật: WebRTC sử dụng các giao thức mã hóa tiêu chuẩn như DTLS (Datagram Transport Layer Security) và SRTP (Secure Real-time Transport Protocol) để bảo vệ dữ liệu trong quá trình truyền.
* Tương thích đa nền tảng: WebRTC được hỗ trợ trên nhiều nền tảng và trình duyệt, giúp ứng dụng phát triển trên các thiết bị và hệ điều hành khác nhau.

Một số thuật ngữ trong WebRTC

* ICE (Interactive Connectivity Establishment): là một giao thức được sử dụng để thiết lập kết nối trực tiếp giữa các máy tính thông qua mạng Internet, bao gồm việc xác định địa chỉ IP và cổng phù hợp.
* ICE candidate: là một thành phần của giao thức ICE, mỗi máy tính trong một kết nối WebRTC sẽ tạo ra các ICE candidate, đại diện cho các cách tiếp cận khác nhau để truy cập địa chỉ IP và cổng của nó thông qua NAT hoặc firewall. Cụ thể, mỗi ICE candidate bao gồm một địa chỉ IP, một cổng và một giao thức (UDP, TCP hoặc SCTP).
* NAT (Network Address Translation): là một công nghệ được sử dụng trong mạng máy tính để chuyển đổi các địa chỉ IP của các thiết bị trong mạng nội bộ thành địa chỉ IP công cộng.
* STUN (Session Traversal Utilities for NAT): Là một dịch vụ hoặc máy chủ được sử dụng để tìm kiếm địa chỉ IP công cộng của một máy tính nằm sau NAT.
* TURN (Traversal Using Relays around NAT): Là một dịch vụ hoặc máy chủ được sử dụng khi các kỹ thuật khi STUN không thể thực hiện. Thay vì cố gắng thiết lập kết nối trực tiếp giữa hai máy tính, TURN sẽ tạo ra một "trạm" (relay) hoạt động trung gian. Dữ liệu sẽ được truyền qua trạm này, giúp vượt qua rào cản NAT.
* SDP (Session Description Protocol): Là một định dạng dữ liệu được sử dụng để mô tả thông tin về phiên trò chuyện hoặc kết nối giao tiếp, bao gồm các thông tin về các luồng phương tiện và các tham số kết nối.
* Offer: là 1 loại SDP mà các đối tác trao đổi với nhau, offer được tạo ra bởi bên yêu cầu kết nối để gửi đến bên muốn được kết nối.
* Answer: tương tự offer nhưng được tạo ra bởi bên được kết nối để trả lời lại cho bên yêu cầu.
* Peer Connection (Kết nối đồng đối tác): Đại diện cho kết nối trực tiếp giữa hai máy tính trong một ứng dụng WebRTC.
* MediaStream (Luồng phương tiện): Là một chuỗi dữ liệu liên tục của dữ liệu phương tiện như âm thanh hoặc video được tạo ra từ các nguồn như camera hoặc microphone.
* DataChannel (Kênh dữ liệu): Là một phần của API WebRTC cho phép truyền dữ liệu trực tiếp giữa các peer connection, không chỉ là dữ liệu phương tiện mà còn là dữ liệu văn bản hoặc nhị phân.
* Signalling: Là quá trình trao đổi thông tin giữa các máy tính để thiết lập một kết nối WebRTC, bao gồm việc truyền tải thông điệp như địa chỉ IP, cổng, và các tham số kết nối. Thường được thực hiện qua 1 server trung gian, gọi là Signalling Server.

Quy trình thiết lập kết nối giữa 2 máy Client 1 và Client 2 trong WebRTC

1. Client 1 yêu cầu lấy luồng dữ liệu dữ liệu từ camera, microphone.
2. Client 1 tạo 1 đối tượng RTCPeerConnection
3. Kết nối đến STUN hoặc TURN để lấy các ICE candidate
4. Thêm luồng dữ liệu đã lấy vào RTCPeerConnection để có thể truyền đi
5. Tạo 1 Offer thông qua đối tượng RTCPeerConnection. Đặt Offer đã tạo làm LocalDescription của RTCPeerConnection.
6. Gửi Offer đã tạo và các ICE candidate nhận được tới Signalling Server.
7. Client 2 yêu cầu lấy Offer của Client 1 từ Signalling Server.
8. Signalling Server gửi trả Offer
9. Client 2 yêu cầu lấy luồng dữ liệu dữ liệu từ camera, microphone.
10. Client 2 tạo 1 đối tượng RTCPeerConnection
11. Kết nối đến STUN hoặc TURN để lấy các ICE candidate
12. Thêm luồng dữ liệu đã lấy vào RTCPeerConnection để có thể truyền đi.
13. Client 2 đặt Offer Client 1 làm RemoteDescription của RTCPeerConnection.
14. Client 2 tạo Answer thông qua đối tượng RTCPeerConnection và đặt Answer đã tạo làm LocalDescription của RTCPeerConnection.
15. Client 2 yêu cầu các ICE candidate của Client 1 từ Signalling Server.
16. Signalling Server gửi lại các ICE candidate, thêm các ICE candidate đó vào RTCPeerConnection của Client 2.
17. Gửi Answer đã tạo và các ICE candidate nhận được tới Signalling Server.
18. Client 1 yêu cầu Answer và các ICE candidate của Client 2 từ Signalling Server.
19. Signalling Server gửi lại Answer và các ICE candidate, đặt Answer đã tạo làm RemoteDescription và thêm các ICE candidate vào RTCPeerConnection
20. Client 1 và Client 2 bắt đầu tự kết nối và truyền dữ liệu cho với nhau.



Biểu đồ quy trình thiết lập kết nối