**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ**

**TRUYỀN THÔNG VIỆT – HÀN**

**KHOA KHOA HỌC MÁY TÍNH**



**BÁO CÁO MÔN HỌC**

**LINUX VÀ PHẦN MỀM NGUỒN MỞ**

**ĐỀ TÀI: THIẾT LẬP BỘ DỊCH VỤ LEMP ĐỂ TRIỂN KHAI DỊCH VỤ WEB CHO NGƯỜI DÙNG**

Sinh viên thực hiện : **NGUYỄN VIẾT TÂN**  Mã: **20IT513**

**ĐINH HỮU ĐỨC** Mã:

**LƯU MINH KHA** Mã:

Giảng viên hướng dẫn : **TS. TRẦN THẾ SƠN**

***Đà nẵng, tháng 12 năm 2022***

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ**

**TRUYỀN THÔNG VIỆT – HÀN**

**KHOA KHOA HỌC MÁY TÍNH**



**BÁO CÁO MÔN HỌC**

**LINUX VÀ PHẦN MỀM NGUỒN MỞ**

**ĐỀ TÀI: THIẾT LẬP BỘ DỊCH VỤ LEMP ĐỂ TRIỂN KHAI DỊCH VỤ WEB CHO NGƯỜI DÙNG**

***Đà Nẵng, tháng 12 năm 2022***

**MỞ ĐẦU**

Hiện nay Công nghệ thông tin vô cùng phát triển thì mọi người đều sử dụng máy vi tính để làm việc. Công nghệ thông tin cũng được áp dụng rất nhiều vào các lĩnh vực mà điển hình là lĩnh vực quản lý, làm việc đội nhóm.

Những năm gần đây dịch bệnh bùng phát đã gây ra tác động nghiêm trọng đến hoạt động sản xuất kinh doanh của các doanh nghiệp, tổ chức. Khiến cho các tổ chức, doanh nghiệp chuyển dần sang xu hướng làm việc trực tuyến. Tuy nhiên việc chuyển đổi này đòi hỏi phải có những giải pháp, công nghệ đúng đắn để không gây mất thời gian, khó khăn trong việc theo dõi và quản lý công việc.

Từ thực tế đó, việc xây dựng được chương trình hỗ trợ làm việc tương tác với nhau trên không gian mạng là rất cần thiết. Vì vậy em đã chọn đề tài “ Xây dựng hệ thống mạng ngân hàng AB Bank chi nhánh Đà Nẵng”.

Với mục đích nghiên cứu, tìm hiểu về và xây dựng website để có thể đáp ứng được nhu cầu của các tổ chức, doanh nghiệp làm việc trực tuyến, giúp cho mọi người có thể tiết kiệm công sức, thời gian và giúp việc quản lý dễ dàng, tiện lợi hơn.

**LỜI CẢM ƠN**

Để thực hiện và hoàn thành tốt đồ án này, chúng em đã nhận được sự giúp đỡ và hướng dẫn rất tận tình của các thầy cô giảng viên Trường Đại Học Công Nghệ Thông Tin và Truyền Thông Việt-Hàn. Chúng em xin cảm ơn các thầy cô thuộc bộ môn chuyên ngành đã cung cấp cho chúng em các thông tin, kiến thức vô cùng quý báu và cần thiết trong suốt thời gian quá để chúng em có thể thực hiện và hoàn thành đồ án của mình. Đặc biệt chúng em xin chân thành cảm ơn đến ThS. Ninh Khánh Chi, cô đã tận tình góp ý, hướng dẫn để chúng em có những định hướng tốt trong việc triển khai và đáp ứng các yêu cầu trong quá trình thực hiện đồ án.

Cuối cùng, xin chân thành cảm ơn các bạn trong ngành công nghệ thông tin đã ủng hộ, giúp đỡ, chia sẻ kiến thức, kinh nghiệm giúp chúng em trong quá trình nghiên cứu và thực hiện đề tài.

Vì lí do giới hạn về mặt thời gian và kiến thức cũng như kinh nghiệm thực tiễn nên đề tài không tránh khỏi những sai xót. Chúng em rất mong nhận được sự thông cảm cùng những góp ý của quý thầy cô và các bạn.

Nhóm chúng em xin chân thành cảm ơn!

Đà Nẵng, ngày 01 tháng 12 năm 2022

Thành viên nhóm

Nguyễn Viết Tân

Đinh Hữu Đức

Lưu Minh Kha

**NHẬN XÉT**

Đà Nẵng, ngày …, tháng …, năm 2022

Giảng viên hướng dẫn

TS. Trần Thế Sơn

MỤC LỤC

[Chương 1 CƠ SỞ LÝ THUYẾT 1](#_Toc123890352)

[1.1 Hệ điều hành Linux là gì: 1](#_Toc123890353)

[1.1.1 Định nghĩa: 1](#_Toc123890354)

[1.1.2 Cấu trúc hệ điều hành Linux: 1](#_Toc123890355)

[1.1.3 Công dụng của hệ điều hành Linux: 2](#_Toc123890356)

[1.1.4 Ưu điểm: 2](#_Toc123890357)

[1.1.5 Nhược điểm: 2](#_Toc123890358)

[1.1.6 Các phiên bản của hệ điều hành Linux: 2](#_Toc123890359)

[1.2 Ubuntu 3](#_Toc123890360)

[1.2.1 Khái niệm: 3](#_Toc123890361)

[1.2.2 Một số tính năng của Ubuntu: 3](#_Toc123890362)

[1.2.3 Ưu điểm của Ubuntu: 6](#_Toc123890363)

[1.3 LEMP 7](#_Toc123890364)

[1.3.1 Khái niệm: 7](#_Toc123890365)

[1.3.2 Ưu điểm: 8](#_Toc123890366)

[Chương 2 PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 9](#_Toc123890367)

[2.1 Mô hình thiết kế hệ thống mạng: 9](#_Toc123890368)

[Chương 3 TRIỂN KHAI VÀ ĐÁNH GIÁ 10](#_Toc123890369)

[3.1 Bảng thống kê thiết bị chính: 10](#_Toc123890370)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 11](#_Toc123890371)

[1. Kết quả đạt được: 12](#_Toc123890372)

[2. Phương hướng phát triển: 12](#_Toc123890373)

[DANH MỤC TÀI LIỆU THAM KHẢO 12](#_Toc123890374)

**DANH MỤC CÁC HÌNH**

No table of figures entries found.

**DANH MỤC BẢNG**

**DANH SÁCH VIẾT TẮT**

# CƠ SỞ LÝ THUYẾT

## Hệ điều hành Linux là gì:

### Định nghĩa:

Linux là một hệ điều hành máy tính được phát triển từ năm 1991 dựa trên hệ điều hành Unix và bằng viết bằng ngôn ngữ C.

Do Linux được phát hành miễn phí và có nhiều ưu điểm vượt trội nên Linux vẫn giữ được một chỗ đứng vững chắc trong lòng người dùng trước các ông lớn như Windows hay macOS.



### Cấu trúc hệ điều hành Linux:

#### Kernel

Hay được gọi là phần Nhân vì đây là phần quan trọng nhất trong máy tính bởi chứa đựng các module hay các thư viện để quản lý, giao tiếp giữa phần cứng máy tính và các ứng dụng.

#### Shell

Shell là phần có chức năng thực thi các lệnh (command) từ người dùng hoặc từ các ứng dụng yêu cầu, chuyển đến cho Kernel xử lý. Shell chính là cầu nối để kết nối Kernel và Application, phiên dịch các lệnh từ Application gửi đến Kernel để thực thi.

Có các loại Shell như sau: sh (the Bourne Shell), bash(Bourne-again shell), csh (C shell), ash (Almquist shell), tsh (TENEX C shell), zsh (Z shell).

#### Application

Đây là phần quen thuộc với chúng ta nhất, phần để người dùng cài đặt ứng dụng, chạy ứng dụng để người dùng có thể phục vụ cho nhu cầu của mình.

### Công dụng của hệ điều hành Linux:

Tương tự như các hệ điều hành khác, Linux cũng cấp môi trường trung gian để người dùng có thể giao tiếp với phần cứng máy tính, thực hiện các công việc của mình.

Bên cạnh đó, nhờ ứng dụng mã nguồn mở mà Linux đem lại nhiều sự thoải mái hơn cho người dùng, đặc biệt các lập trình viên, nhà phát triển.

### Ưu điểm:

* Miễn phí và được hỗ trợ các ứng dụng văn phòng OpenOffice và LibreOffice.
* Tính bảo mật cao.
* Tính linh hoạt, người dùng có thể chỉnh sửa hệ điều hành để phù hợp với nhu cầu sử dụng của mình.
* Không lo sợ giật, lag, không chạy nổi,…trên các máy tính có cấu hình không cao.

### Nhược điểm:

* Các nhà phát triển phần mềm vẫn chưa để tâm đến hệ điều hành tiềm năng này nên số lượng phần mềm được hỗ trợ vẫn còn hạn chế.
* Một số nhà sản xuất không phát triển driver hỗ trợ nền tảng Linux.

### Các phiên bản của hệ điều hành Linux:

* Ubuntu
* Linux Mint
* Debian
* Fedora
* CentOS Linux
* OpenSUSE/SUSE Linux Enterprise
* Mageia Linux
* Slackware Linux
* Puppy Linux

## Ubuntu

### Khái niệm:

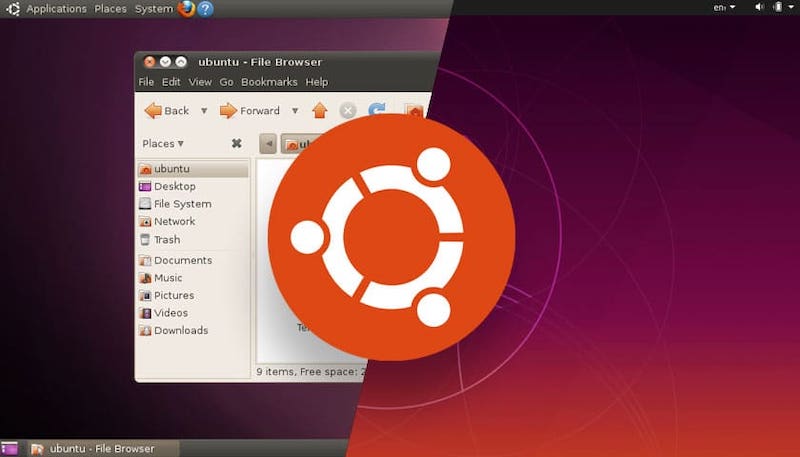
Ubuntu là một hệ điều hành máy tính dựa trên Debian GNU/Linux, một bản phân phối Linux thông dụng. Mục đích của Ubuntu bao gồm việc cung cấp một hệ điều hành ổn định, cập nhật cho người dùng thường, và tập trung vào sự tiện dụng và dễ dàng cài đặt. Ubuntu đã được đánh xếp hạng là bản phân phối Linux thông dụng nhất cho máy tính để bàn, chiếm khoảng 30% số bản Linux được cài đặt trên máy tính để bàn năm 2007.

Ubuntu là phần mềm mã nguồn mở tự do, có nghĩa là người dùng được tự do chạy, sao chép, phân phối, nghiên cứu, thay đổi và cải tiến phần mềm theo điều khoản của giấy phép GNU GPL. Ubuntu được tài trợ bởi Canonical Ltd (chủ sở hữu là một người Nam Phi Mark Shuttleworth). Thay vì bán Ubuntu, Canonical tạo ra doanh thu bằng cách bán hỗ trợ kĩ thuật. Bằng việc để cho Ubuntu tự do và mở mã nguồn, Canonical có thể tận dụng tài năng của những nhà phát triển ở bên ngoài trong các thành phần cấu tạo của Ubuntu mà không cần phải tự mình phát triển.

### Một số tính năng của Ubuntu:

#### Thừa hưởng tính năng nổi bật của Linux:

Vì đây là phiên bản hệ điều hành phát triển dựa trên Linux, vậy nên sẽ thừa hưởng được các tính năng đặc biệt của Linux. Chẳng hạn như khả năng tùy biến hiệu suất làm việc, tốc độ, cũng như khả năng bảo mật trước sự tấn công của malware hay virus.



#### Hỗ trợ người dùng trong việc cài đặt:

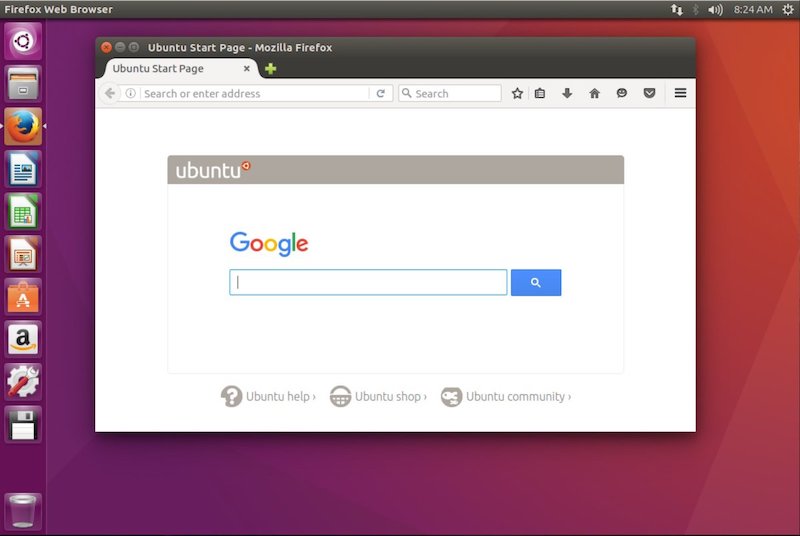
Để kiểm tra sự tương thích của các phiên bản với máy tính, bạn có thể sử dụng đĩa chạy trực tiếp phiên bản đó của hệ điều hành, từ đó người dùng sẽ quyết định xem có thể cài đặt trên thiết bị của họ hay không thông qua phần mềm Ubiquity.

Ngoài ra, người dùng sử dụng Windows có thể phân vùng lại đĩa cứng để cài đặt Ubuntu, và bạn cũng có thể gỡ bỏ nó khá dễ dàng.ngay cả khi đang sử dụng Windows.



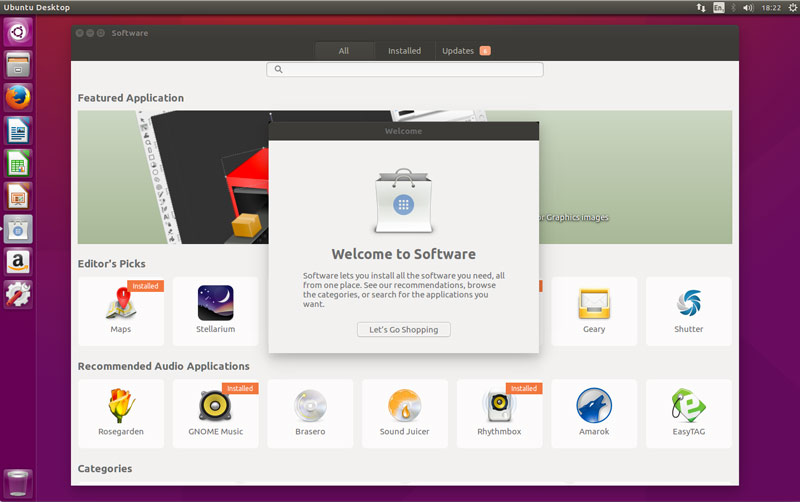
#### Giao diện:

Ubuntu hiện đang sử dụng giao diện đồ hoạ Unity, và giao diện này được thiết kế ra với mục đích tối ưu diện tích màn hình sử dụng, đồng thời cũng mang lại trải nghiệm dùng thân thiện.



#### Ứng dụng:

Hệ điều hành Ubuntu đã được cài sẵn các phần mềm mã nguồn mở, chẳng hạn như trình duyệt Internet Firefox, bộ ứng dụng văn phòng đến từ LibreOffice (phiên bản Ubuntu 10.04 trở đi) hay trình tải file torrent Tranmission. Ngoài ra, hệ điều hành này cũng có một kho các ứng dụng miễn phí có tên là Ubuntu Software Center.



### Ưu điểm của Ubuntu:

#### Không yêu cầu cấu hình mạnh:

Thông thường, quá trình cài đặt Ubuntu trên máy tính sẽ giao động từ 20 - 30 phút, tuỳ theo cấu hình của máy, và bạn chỉ cần có thiết bị có dung lượng RAM từ 256MB trở lên. Và cấu hình khuyên dùng thì chỉ cần RAM trên 384MB, và CPU là 700 MHz (x86).

#### Có nhiều ứng dụng miễn phí:

Bạn không cần tốn quá nhiều chi phí cho các ứng dụng, và chúng đều có mặt trên Ubuntu Software Center. Chẳng hạn như bộ ứng dụng văn phòng LibreOffice miễn phí cũng đã được cài sẵn trên hệ điều hành Ubuntu.

#### Khả năng tương tác sâu

Mặc dù đôi khi phải sử dụng các câu lệnh để tương tác với thiết bị, nhưng nhờ vậy mà người dùng có thể điều khiển phần mềm một cách sâu hơn, từ đó hiệu quả công việc sẽ tốt hơn, nhất là khi so sánh với các giao diện đồ họa. Đồng thời, đây là hệ điều hành mã nguồn mở, thế nên Ubuntu có độ tùy biến cao.

#### Có tính bảo mật cao

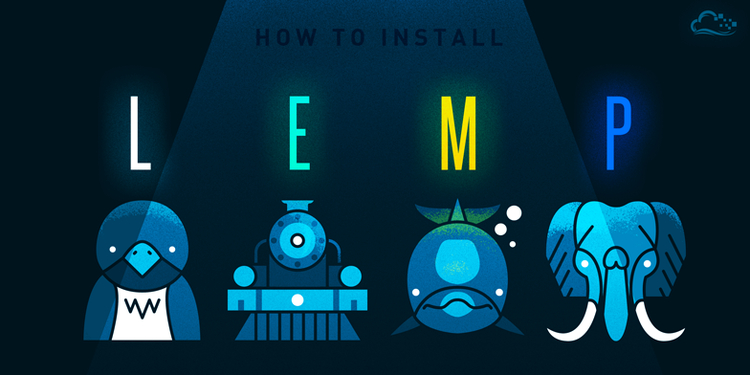
Không cần lo lắng quá nhiều về việc có cần cài đặt các phần mềm diệt virus như trên Windows bởi vì số lượng malware hay virus khá ít trên hệ điều hành Ubuntu.

## LEMP

### Khái niệm:

Các thành phần cấu thành LEMP stack cũng gần tương tự với LAMP, chỉ khác là Apache sẽ được thay thế bởi nginx. Nginx được đọc là "engine-x", giải thích cho chữ E trong "LEPM", nginx cũng là một ứng dụng HTTP proxy nhưng không có được danh tiếng ấn tượng như Apache, tuy nhiên, nó có ưu điểm là cho phép xử lý tốc độ tải cao hơn đối với các HTTP request.

Nginx giờ đây, đã đạt được sự thu hút đáng kể đối với người dùng khi nó bắt đầu được nhiều người sử dụng từ năm 2008 và hiện trở thành ứng dụng web server tiếng tăm thứ 2 sau Apache khi đề cập các active site theo báo cáo của Netcraft.



### Ưu điểm:

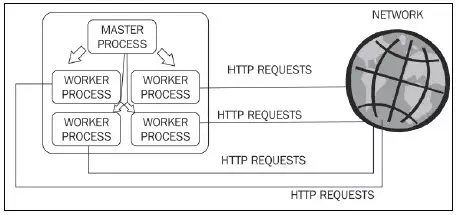
* Ứng dụng web server mã nguồn mở được viết để giải quyết các vấn đề về hiệu suất và khả năng mở rộng có liên quan đến Apache.
* Phương pháp Event-driven, không đồng bộ và không bị chặn, không tạo các process mới cho mỗi request từ web.
* Đặt số lượng cho các worker process và mỗi worker có thể xử lý hàng nghìn kết nối đồng thời
* Các module sẽ được chèn vào trong thời gian biên dịch, có trình biên dịch mã PHP bên trong (không cần đến module PHP).

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Cấu trúc HTTP trong nginx

### Kiến trúc của Nginx

Khi được khởi chạy service, nginx khởi tạo mọt tiến trình chủ và cũng là tiến trình duy nhất tồn tại trong bộ nhớ ***Master Process***. Tiến trình này không chịu trách nhiệm tự xử lý bất kỳ request nào từ phía client mà thay vào đó nó sinh ra các tiến trình con gọi là ***Worker Process*** để xử lý các request này.



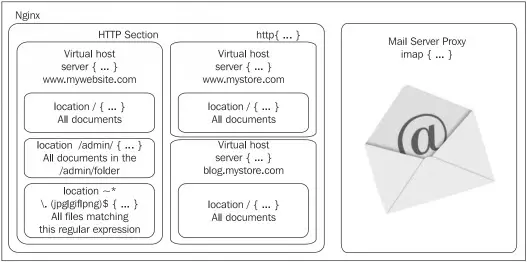
Để định nghĩa cho các ***Worker Process*** này, chúng ta cần sử dụng tệp tin cấu hình để xác định số tiến trình, số lượng kết nối , tài khoản và nhóm tài khoản mà mỗi ***Worker Process*** chạy.

### Giới thiệu Module Http trong Nginx

Module HTTP Core là thành phần chứa tất cả các khối, chỉ thị và các biến cơ bản của máy chủ HTTP. Mặc định module này được cài đặt trong khi biên dịch, nhưng không được bật lên khi Nginx chạy, việc sử dụng module này là không bắt buộc.

Module này là 1 trong những module tiêu chuẩn lớn nhất của Nginx – nó cung cấp 1 số lượng lớn các chỉ thị và biến. Để có thể hiểu được tất cả các yếu tố này và vai trò của chúng, chúng ta sẽ bắt tay vào tìm hiểu 3 khối chỉ thị chính – ***http***, ***server*** và ***location***.

* ***http*** : được khai báo ở phần đầu của tập tin cấu hình. Nó cho phép chúng ta định nghĩa các chỉ thị và các khối từ tất cả các module liên quan đến khía cạnh HTTP của Nginx. Khối chỉ thị này có thể được khai báo nhiều lần trong tập tin cấu hình, và các giá trị chỉ thị được chèn trong khối http sau sẽ ghi đè lên các chỉ thị nằm trong khối http trước đó.
* ***server*** : khối này cho phép chúng ta khai báo 1 website. Nói cách khác, 1 website cụ thể (được nhận diện bởi 1 hoặc nhiều hostname) được thừa nhận bới Nginx và nhận cấu hình của chính nó. Khối này chỉ có thể được dùng bên trong khối http.
* ***location*** : cho phép chúng ta định nghĩa 1 nhóm các thiết lập được áp dụng cho 1 vị trí cụ thể trên website (thể hiện qua URL của website đó). Khối location có thể được dùng bên trong 1 khối server hoặc nằm chồng bên trong 1 khối location khác.



Trong biểu đồ trên, khu vực HTTP, được định nghĩa bởi khối http, bao quanh toàn bộ các cấu hình liên quan đến web. Nó cũng chứa 1 hoặc nhiều khối server, định nghĩa các tên miền của các website mà chúng ta có. Với mỗi website này, chúng ta có thể định nghĩa nhiều khối location mà cho phép chúng ta áp dụng các thiết lập bổ sung đến 1 URI yêu cầu cụ thể của website hoặc các URI yêu cầu khớp 1 mẫu nào đó.

### Các chỉ thị

#### Các chỉ thị về cấu hình HOST và SOCKET

* listen :
* Sử dụng trong khối : server
* Chỉ rõ địa chỉ IP và/hoặc port được dùng bởi socket phục vụ website. Các website thường được phục vụ trên port 80 (giá trị mặc định) qua HTTP, hoặc 443 qua HTTPS.
* Cú pháp: listen [address] [:port] [additional options];
* Các tùy chọn bổ sung:
* default hoặc default\_server : Chỉ rõ khối server này được dùng như website mặc định cho bất kỳ yêu cầu nhận được tại địa chỉ IP và port được chỉ rõ.
* ssl: Chỉ rõ website sẽ sử dụng SSL.
* Các tùy chọn khác liên quan đến các lời gọi hệ thống bind và listen gồm: backlog=num, rcvbuf=size, sndbuf=size, accept\_filter=filter, deferred, setfib=number, và bind.
* Ví dụ:

*Listen 192.168.1.1:80;*

*Listen 127.0.0.1;*

*Listen 80 default;*

*Listen 443 ssl;*

* server\_name :
* Sử dụng trong khối : server
* Đăng ký 1 hoặc nhiều hostname cho khối server. Khi Nginx nhận 1 yêu cầu HTTP, nó so sánh giá trị Host trong phần header của yêu cầu với tất cả các khối server đang có. Khối server đầu tiên khớp với hostname này sẽ được chọn.
* Nếu không có khối server nào khớp với hostname trên, Nginx chọn khối server đầu tiên khớp với các thông số của chỉ thị listen (ví dụ như listen \*:80 sẽ bắt tất cả các yêu cầu nhận được trên port 80), ưu tiên khối đầu tiên có tùy chọn mặc định được cho phép trên chỉ thị listen.
* Cú pháp: server\_name hostname1 [hostname2…];
* Ví dụ :

*server\_name www.acb.com;*

*server\_name www.abc.com abc.com;*

*server\_name \*.website.com; # nhận tất cả các domain có đuôi là .website.com*

*server\_name .website.com; # Kết hợp cả \*.website.com và website.com*

*server\_name \*.website.\*;*

*server\_name ~^\.example\.com$;*

Lưu ý rằng chúng ta có thể sử dụng chuỗi rỗng như 1 giá trị của chỉ thị để bắt tất cả các yêu cầu không có giá trị Host trong phần header, nhưng chỉ sau ít nhất 1 tên thông thường (hoặc “\_”).

*server\_name abc.com “”;*

*server\_name \_ “”;*

* server\_name\_in\_redirect :
* Sử dụng trong khối : server, http, location
* Chỉ thị này áp dụng cho trường hợp của các chuyển hướng nội bộ (internal redirect). Nếu thiết lập thành on, Nginx sẽ sử dụng hostname đầu tiên được chỉ rõ trong chỉ thị server\_name. Nếu thiết lập thành off, Nginx sẽ sử dụng giá trị Host trong phần header của yêu cầu HTTP.
* Cú pháp: server\_name\_in\_redirect [on | off];
* Giá trị mặc định: off
* server\_names\_hash\_max\_size :
* Sử dụng trong khối : http
* Nginx sử dụng các bảng hash cho các tập hợp dữ liệu khác nhau để tăng tốc việc xử lý yêu cầu. Chỉ thị này định nghĩa kích thước tối đa của bảng hash chứa các server\_name. Giá trị mặc định phù hợp với đa số cấu hình. Nếu giá trị này cần thay đổi, Nginx sẽ tự động thông báo cho chúng ta khi khởi động, hay khi chúng ta tải lại cấu hình.
* Cú pháp: server\_names\_hash\_max\_size 512;
* Giá trị mặc định: 512 bytes
* server\_names\_hash\_bucket\_size :
* Sử dụng trong khối : http
* Thiết lập kích thước vùng chứa các bảng hash server\_name. Tương tự, chúng ta chỉ nên thay đổi giá trị này khi Nginx yêu cầu.
* Cú pháp: server\_names\_hash\_bucket\_size 32;
* Giá trị mặc định: 32 byte (hoặc 64, hay 128, dựa vào bộ nhớ cache của CPU).
* port\_in\_redirect :
* Sử dụng trong khối : server, http, location
* Chỉ thị này định nghĩa việc Nginx có hoặc không thêm giá trị port vào URL chuyển hướng.
* Cú pháp: port\_in\_redirect [on or off];
* Giá trị mặc định: on
* tcp\_nodelay :
* Sử dụng trong khối : server, http, location
* Cho phép hoặc vô hiệu hóa tùy chọn socket TCP\_NODELAY cho chỉ các kết nối keep-alive.
* Cú pháp: tcp\_nodelay [on or off];
* Giá trị mặc định: on
* NOTE : Lưu ý: Keep-alive (hoặc Keepalive) là 1 thông điệp được gửi bởi 1 thiết bị đến các thiết bị khác để kiểm tra rằng liên kết giữa 2 thiết bị đang hoạt động, hoặc để ngăn ngừa việc liên kết này bị mất.
* tcp\_nopush :
* Sử dụng trong khối : server, http, location
* Cho phép hoặc vô hiệu hóa tùy chọn socket TCP\_NOPUSH (FreeBSD) hoặc TCP\_CORK (Linux). Chỉ thị này chỉ áp dụng khi chỉ thị sendfile được cho phép. Nếu chỉ thị tcp\_nopush được thiết lập là on, Nginx sẽ cố gắng truyền toàn bộ phần header phản hồi HTTP trong 1 gói tin TCP.
* Cú pháp: tcp\_nopush [on or off];
* Giá trị mặc định: off
* sendfile\_max\_chunk :
* Sử dụng trong khối : server, http
* Chỉ thị này định nghĩa 1 kích thước tối đa của dữ liệu được dùng cho mỗi lời gọi sendfile.
* Cú pháp: sendfile\_max\_chunk 0;
* Giá trị mặc định: 0
* reset\_timedout\_connection :
* Sử dụng trong khối : server, http, location
* Khi 1 kết nối hết thời hạn, thông tin tương ứng của nó có thể được giữ trong bộ nhớ dựa vào trạng thái của kết nối đó. Việc sử dụng chỉ thị này sẽ xóa tất cả bộ nhớ tương ứng đến kết nối sau khi nó hết thời hạn.
* Cú pháp: reset\_timedout\_connection off;
* Giá trị mặc định: off

### Cấu hình đường dẫn và tài liệu

* root :
* Sử dụng trong khối : server, http, location, if Các biến được chấp nhận.
* Định nghĩa tài liệu gốc, chứa các tập tin mà bạn muốn phục vụ cho khách.
* Cú pháp: root /path/resource/;
* Giá trị mặc định: html
* alias :
* Sử dụng trong khối :location Các biến được chấp nhận.
* 1 chỉ thị mà chúng ta chỉ có thể đặt trong khối location. Nó đăng ký 1 đường dẫn khác cho Nginx lấy các tài liệu cho 1 yêu cầu cụ thể :

*http {*

*server {*

*server\_name abc.com;*

*root /var/www/abc.com/;*

*location /admin/ {*

*alias /var/www/abc.net/;*

*}*

}

*Chú thích* : Khi 1 yêu cầu cho http://abc.com/ được nhận, các tập tin được phục vụ từ thư mục /var/www/abc.com. Tuy nhiên, nếu Nginx nhận 1 yêu cầu cho http://abc.com/admin/, đường dẫn được dùng để lấy tập tin là /var/www/abc.net/. Hơn thế nữa, giá trị của chỉ thị root không được thay đổi. Quá trình này vô hình trong mắt của các script động.

* error\_page :
* Sử dụng trong khối : server, http, location, if Các biến được chấp nhận.
* Cho phép chúng ta ảnh hưởng các URI đến mã phản hồi HTTP và tùy chọn để thay thế code với cái khác.
* Cú pháp: error\_page code1 [code2…] [=replacement code] [=@block | URI]
* Ví dụ :

*error\_page 404 /not\_found.html;*

*error\_page 500 501 502 503 504 /server\_error.html;*

*error\_page 403 http://website.com/;*

*error\_page 404 @notfound; #nhảy đến 1 khối location được đặt tên là notfound*

*error\_page 404 =200 /index.html; # trong trường hợp lỗi 404, chuyển hướng đến index.html với mã phản hồi là 200 OK.*

* if\_modified\_since :
* Sử dụng trong khối : server, http, location
* Định nghĩa cách Nginx xử lý header If-Modified-Since trong gói tin HTTP. Header này được sử dụng đa số bởi các bot của công cụ tìm kiếm (như Google bot). Các bot này chỉ ra ngày tháng và thời gian của lần truyền cuối cùng. Nếu tập tin được yêu cầu không được sửa đổi từ thời gian đó, máy chủ sẽ trả về 1 mã phản hồi 304 Not Modified.
* Chỉ thị này chấp nhận 3 giá trị:
* off: bỏ qua header If-Modified-Since
* exact : trả về mã 304 Not Modified nếu ngày tháng và thời gian được chỉ rõ trong phần header chính xác khớp với ngày tháng sửa đổi thật sự của tập tin được yêu cầu. Nếu ngày tháng sửa đổi tập tin là trước hoặc sau, tập tin được phục vụ bình thường. (mã phản hồi 200 OK).
* before : trả về mã 304 Not Modified nếu ngày tháng và thời gian được chỉ rõ trong phần header là trước hoặc bằng với ngày tháng sửa đổi tập tin được yêu cầu.
* Cú pháp: if\_modified\_since off | exact | before
* Giá trị mặc định: exact
* index :
* Sử dụng trong khối : server, http, location. Các biến được chấp nhận.
* Định nghĩa trang mặc định mà Nginx sẽ phục vụ nếu không có tên tập tin được chỉ rõ trong yêu cầu (nói cách khác, trang chỉ mục). Chúng ta có thể chỉ rõ nhiều tên tập tin và tập tin đầu tiên được tìm thấy sẽ được sử dụng. Nếu không có tập tin cụ thể nào được tìm thấy, Nginx sẽ hoặc là cố gắng phát sinh 1 chỉ mục tự động của các tập tin, nếu chỉ mục autoindex được cho phép hoặc trả về 1 trang lỗi 403 Forbidden. Tùy chọn, chúng ta có thể nhập 1 tên tập tin tuyệt đối (như là /page.html, tính từ thư mục gốc của website) nhưng đây chỉ có thể là tham số cuối cùng của chỉ thị này..
* Cú pháp: ndex file1 [file2…] [absolute\_file];
* Giá trị mặc định: index.php index.html index.htm;
* recursive\_error\_pages :
* Sử dụng trong khối : server, http, location,.
* Khi 1 trang lỗi (được khai báo trong chỉ thị error\_page) gặp lỗi, nó sử dụng giá trị của chỉ thị error\_page (tức là chính trang lỗi đó) để xử lý lỗi trên, và điều này lặp đi lặp lại, tạo ra vòng lặp vô hạn với lỗi trên (tình trạng này được gọi là đệ quy). Chỉ thị này cho phép hoặc vô hiệu hóa các trang lỗi đệ quy như trên.
* Cú pháp: recursive\_error\_pages off/on;
* Giá trị mặc định: off
* try\_files :
* Sử dụng trong khối : server, location Các biến được chấp nhận.
* Cố gắng phục vụ các tập tin được chỉ rõ (các tham số từ 1 đến N-1 trong chỉ thị), nếu không có tập tin nào tồn tại, nhảy đến khối location được khai báo (tham số cuối cùng trong chỉ thị) hoặc phục vụ 1 URI được chỉ định.
* Cú pháp: Nhiều đường dẫn tập tin, theo sau bởi 1 khối location được đặt tên hoặc 1 URI.

*location / {*

*try\_files $uri $uri.html $uri.php $uri.xml @proxy;*

*}*

*# the following is a "named location block"*

*location @proxy {*

*proxy\_pass 127.0.0.1:8080;*

}

* Trong ví dụ trên, Nginx sẽ cố gắng phục vụ các tập tin 1 cách bình thường. Nếu URI của yêu cầu không tương ứng với bất kỳ tập tin đang có nào, Nginx sẽ thêm .html vào URI và cố gắng phục vụ tập tin này 1 lần nữa. Nếu nó vẫn tiếp tục thất bại, Nginx thử với .php, sau đó là .xml. Nếu tất cả đều thất bại, 1 khối location khác **(@proxy)** xử lý yêu cầu này.

### Cấu hình các request từ client:

* keepalive\_requests :
* Sử dụng trong khối : server, http, location
* Xác định số lượng tối đa các yêu cầu được phục vụ trên 1 kết nối keep-alive.
* Cú pháp: keepalive\_requests 100;
* Giá trị mặc định: 100
* keepalive\_timeout :
* Sử dụng trong khối : server, http, location
* Định nghĩa số giây mà máy chủ sẽ chờ trước khi đóng 1 kết nối keep-alive.
* Cú pháp: keepalive\_timeout time1 [time2];
* Giá trị mặc định: 75

*keepalive\_timeout 75;*

*keepalive\_timeout 75 60;*

* keepalive\_disable :
* Sử dụng trong khối : server, http, location
* Tùy chọn này cho phép chúng ta vô hiệu hóa chức năng keepalive cho các trình duyệt web.
* Cú pháp: keepalive\_disable browser1 browser2;
* Giá trị mặc định: msie6
* send\_timeout :
* Sử dụng trong khối : server, http, location
* Số thời gian sau đó Nginx đóng 1 kết nối không hoạt động. 1 kết nối trở thành không hoạt động khi client ngừng truyền dữ liệu.
* Cú pháp: send\_timeout [giá trị thời gian tính bằng giây];
* Giá trị mặc định: 60
* client\_body\_in\_file\_only :
* Sử dụng trong khối : server, http, location
* Nếu chỉ thị này được cho phép, phần nội dung của 1 yêu cầu HTTP sẽ được lưu trong các tập tin thực sự trên đĩa cứng. Phần client\_body tương ứng với dữ liệu thô trong yêu cầu HTTP của client, trừ phần header (nói cách khác, nội dung được truyền trong các yêu cầu POST). Các tập tin được lưu như các tập tin text.
* Chỉ thị này chấp nhận 3 giá trị:
* off: không lưu dữ liệu yêu cầu trong 1 tập tin.
* clean: Lưu dữ liệu yêu cầu trong 1 tập tin và xóa tập tin đó sau khi yêu cầu được xử lý.
* on: Lưu dữ liệu yêu cầu trong 1 tập tin, nhưng không xóa tập tin đó sau khi yêu cầu đó được xử lý (tùy chọn này chỉ nên được xử lý cho những mục đích kiểm tra lỗi).
* Cú pháp: client\_body\_in\_file\_only on | clean | off;
* Giá trị mặc định: off
* client\_body\_in\_single\_buffer :
* Sử dụng trong khối : server, http, location
* Định nghĩa Nginx lưu trữ/hoặc không lưu trữ dữ liệu của yêu cầu trong 1 vùng đệm trong bộ nhớ.
* Cú pháp: client\_body\_in\_single\_buffer [on / off];
* Giá trị mặc định: off
* client\_body\_buffer\_size :
* Sử dụng trong khối : server, http, location
* Chỉ rõ kích thước vùng bộ nhớ đệm lưu giữ dữ liệu của các yêu cầu. Nếu kích thước này bị vượt qua, dữ liệu (hoặc ít nhất là 1 phần của nó) của yêu cầu sẽ được ghi trên đĩa. Lưu ý rằng nếu chỉ thị client\_body\_in\_file\_only được cho phép, dữ liệu của yêu cầu sẽ luôn được lưu thành 1 tập tin trên đĩa cứng, mà không quan tâm đến kích thước của chúng (dù chúng có phù hợp với kích thước bộ nhớ đệm hay không).
* Cú pháp: client\_body\_buffer\_size [giá trị của bộ nhớ];
* Giá trị mặc định: 8k hoặc 16k
* client\_body\_temp\_path :
* Sử dụng trong khối : server, http, location
* Cho phép chúng ta định nghĩa đường dẫn đến thư viện chứa các tập tin dữ liệu yêu cầu từ client. 1 tùy chọn bổ sung cho phép chúng ta tách riêng các tập tin này vào 1 cây thư mục lên đến 3 cấp.
* Cú pháp: client\_body\_temp\_path path [level1] [level2] [level3];
* Giá trị mặc định:

*client\_body\_temp\_path /tmp/nginx\_rbf;*

*client\_body\_temp\_path temp 2; # Nginx sẽ tạo các thư mục có tên 2 chữ số để giữ các tập tin dữ liệu yêu cầu.*

*client\_body\_temp\_path temp 1 2 4; # Nginx sẽ tạo 3 cấp thư mục (cấp đầu tiên: tên có 1 chữ số, cấp thứ 2: tên có 2 chữ số, cấp thứ 3: tên có 4 chữ số)*

* client\_body\_timeout :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa thời gian chờ không hoạt động trong khi đọc 1 dữ liệu yêu cầu từ client. 1 kết nối trở nên không hoạt động khi client ngừng truyền dữ liệu. Nếu đạt đến độ chậm trễ (delay), Nginx trả về 1 lỗi HTTP 408 Request timeout.
  + Cú pháp: client\_body\_timeout [giá trị thời gian tính bằng giây]
  + Giá trị mặc định: 60
* client\_header\_buffer\_size :
  + Sử dụng trong khối : server, http, location, if Các biến được chấp nhận.
  + Chỉ thị này cho phép chúng ta định nghĩa kích thước vùng đệm mà Nginx phân bổ cho các tiêu đề (header) của yêu cầu. Thông thường, 1 KB là đủ. Tuy nhiên, trong vài trường hợp, các tiêu đề chứa nhiều dữ liệu cookie hay độ dài của URI quá lớn. Nếu gặp những trường hợp đó, Nginx sẽ phân bổ 1 hoặc nhiều vùng đệm lớn hơn (kích thước của vùng đệm lớn hơn được định nghĩa trong chỉ thị large\_client\_header\_buffers)..
  + Cú pháp: client\_header\_buffer\_size [giá trị dung lượng byte]
  + Giá trị mặc định: 1k
* client\_header\_timeout :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa thời gian chờ không hoạt động trong khi đọc 1 tiêu đề yêu cầu từ client. 1 kết nối trở nên không hoạt động khi client ngừng truyền dữ liệu. Nếu độ chậm trễ (delay) đạt đến, Nginx trả về 1 lỗi HTTP 408 Request timeout.
  + Cú pháp: client\_header\_timeout [giá trị thời gian tính bằng giây]
  + Giá trị mặc định: 60
* client\_max\_body\_size :
  + Sử dụng trong khối : server, http, location
  + Nó là kích thước tối đa của dữ liệu yêu cầu từ client. Nếu kích thước này bị vượt qua, Nginx trả về 1 lỗi HTTP 413 Request entity too large. Thiết lập này đặc biệt quan trọng nếu chúng ta cho phép người dùng tải các tập tin lên máy chủ qua HTTP.
  + Cú pháp: client\_max\_body\_size [Giá trị dung lượng bộ nhớ trong Nginx.]
  + Giá trị mặc định: 1m (MB)
* large\_client\_header\_buffers :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa số lượng và kích thước của các vùng đệm lớn hơn được dùng cho việc lưu trữ các yêu cầu từ client, trong trường hợp vùng đệm mặc định (client\_header\_buffer\_size) không hiệu quả. Mỗi dòng của tiêu đề phải phù hợp với kích thước của 1 vùng đệm. Nếu dòng URI lớn hơn kích thước của 1 vòng đệm, Nginx trả về 1 lỗi 414 Request URI too large. Nếu 1 dòng tiêu đề khác vượt qua kích thước của 1 vùng đệm, Nginx trả về lỗi 400 Bad request.
  + Cú pháp: large\_client\_header\_buffers [Giá trị dung lượng bộ nhớ .]
  + Giá trị mặc định: 32 KB
* lingering\_time :
  + Sử dụng trong khối : server, http, location
  + Chỉ thị này áp dụng cho các yêu cầu từ client với dữ liệu của yêu cầu đó. Ngay khi lượng dữ liệu được tải lên vượt quá giá trị max\_client\_body\_size, Nginx sẽ ngay lập tức gửi 1 phản hồi lỗi HTTP 413 Request entity too large. Tuy nhiên, phần lớn các trình duyệt tiếp tục tải dữ liệu lên bỏ qua thông báo này. Chỉ thị này định nghĩa lượng thời gian Nginx sẽ đợi sau khi gửi phản hồi lỗi trên trước khi đóng kết nối.
  + Cú pháp: lingering\_time [Giá trị thời gian tính bằng giây]
  + Giá trị mặc định: 30
* lingering\_timeout :
  + Sử dụng trong khối : server, http, location
  + Chỉ thị này định nghĩa lượng thời gian Nginx sẽ đợi giữa 2 tiến trình đọc trước khi đóng kết nối từ client.
  + Cú pháp: lingering\_timeout [Giá trị thời gian tính bằng giây]
  + Giá trị mặc định: 5
* lingering\_close :
  + Sử dụng trong khối : server, http, location
  + Điều khiển cách Nginx đóng các kết nối từ client. Giá trị off sẽ ngay tức khắc đóng các kết nối sau khi tất cả dữ liệu từ các yêu cầu được nhận. Giá trị mặc định (on) cho phép đợi và xử lý dữ liệu bổ sung nếu cần thiết. Nếu thiết lập là always, Nginx sẽ luôn luôn đợi để đóng kết nối. Lượng thời gian đợi được định nghĩa bởi chỉ thị lingering\_timeout.
  + Cú pháp: lingering\_close [on/ off/ always]
  + Giá trị mặc định: on
* ignore\_invalid\_headers :
  + Sử dụng trong khối : server, http
  + Nếu chỉ thị này bị vô hiệu hóa, Nginx sẽ trả về 1 lỗi HTTP 400 Bad Request trong trường hợp tiêu đề của yêu cầu bị lỗi.
  + Cú pháp: ignore\_invalid\_headers [on/ off]
  + Giá trị mặc định: on
* chunked\_transfer\_encoding :
  + Sử dụng trong khối : server, http, location
  + Cho phép hoặc vô hiệu hóa việc mã hóa truyền theo khối cho các yêu cầu HTTP 1.1.
  + Cú pháp: chunked\_transfer\_encoding [on/ off]
  + Giá trị mặc định: on
* max\_ranges :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa số lượng các vùng byte Nginx sẽ chấp nhận để phục vụ khi client yêu cầu 1 phần nội dung từ 1 tập tin. Nếu chúng ta không chỉ rõ giá trị, sẽ không có giới hạn cho giá trị của chỉ thị này. Nếu chúng ta thiết lập giá trị này là 0, chức năng này sẽ bị vô hiệu hóa.
  + Cú pháp: max\_ranges [Giá trị dung lượng bộ nhớ trong Nginx.]

### MIME TYPES

Multipurpose Internet Mail Extensions (MIME) là 1 chuẩn Internet mở rộng định dạng của thư điện tử (email) để hỗ trợ:

* Văn bản trong các tập ký tự khác ASCII.
* Các phần đính kèm không phải văn bản
* Dữ liệu của thông điệp với nhiều phần.
* Thông tin tiêu đề trong các tập ký tự khác ASCII.

Mặc dù MIME đươc thiết kế chủ yếu cho giao thức SMTP, ngày nay nó được sử dụng trong chỉ trong việc mô tả nội dung của email và mà còn bao gồm cả các mô tả của các loại nội dung nói chung.

Tất cả các email trên Internet được truyền qua SMTP trong định dạng MIME.

Các loại nội dung định nghĩa trong các chuẩn MIME cũng quan trọng bên ngoài phạm vi của thư điện tử, như trong các giao thức giao tiếp như HTTP. HTTP đòi hỏi rằng dữ liệu được truyền trong bối cảnh của các thông điệp giống như email, mặc dù dữ liệu thường không phải là những email thực sự.

* types :
* Sử dụng trong khối : server, http, location
* Chỉ thị này cho phép chúng ta tạo các mối tương quan giữa các loại MIME và các phần mở rộng tập tin. Nó thực sự là 1 khối chấp nhận 1 cú pháp đặc biệt:

*types {*

*mimetype1 extension1;*

*mimetype2 extension2 [extension3…];*

*[…]*

}

Khi Nginx phục vụ 1 tập tin, nó kiểm tra phẩn mở rộng của tập tin để quyết định loại MIME. Loại MIME này sau đó được gửi như giá trị của trường Content-Type trong tiêu đề HTTP trong phản hồi của máy chủ với yêu cầu từ client. Tiêu đề này có thể ảnh hưởng đến cách các trình duyệt xử lý các tập tin.

Ví dụ: Nếu loại MIME của tập tin là chúng ta yêu cầu là application/pdf, trình duyệt của chúng ta sẽ cố gắng đọc tập tin đó bằng việc sử dụng 1 plugin tương ứng với loại MIME đó thay cho việc tải tập tin đó về. Nói cách khác, chúng ta sẽ có thể đọc nội dung của tập tin pdf trên trình duyệt, thay cho việc trình duyệt sẽ tải tập tin pdf đó về máy của chúng ta.

Nginx bao gồm 1 tập cơ bản các loại MIME trong 1 tập tin riêng biệt (mime.types), chúng ta có thể bao hàm nội dung của tập tin này vào cấu hình của Nginx bằng việc sử dụng chỉ thị include.

*include mime.types;*

Tập tin này đã bao gồm các phần mở rộng tập tin quan trong nhất, vì vậy chúng ta có lẽ không cần phải điều chỉnh nó. Nếu phần mở rộng của tập tin được phục vụ không được tìm thấy trong các loại đã được liệt kê, loại mặc định được sử dụng, định nghĩa trong chỉ thỉ default\_type.

Giá trị mặc định, nếu tập tin mime.types không được đính kèm, là:

*types {*

*text/html html;*

*image/gif gif;*

*image/jpeg jpg;*

*}*

* default\_type :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa loại MIME mặc định. Khi Nginx phục vụ 1 tập tin, phần mở rộng của tập tin này được đối chiếu với các loại đã biết được khai báo bên trong các khối types để trả về loại MIME chính xác như giá trị của trường Content-Type trong tiêu đề HTTP. Nếu phần mở rộng không khớp với bất kỳ loại nào, giá trị của chỉ thị default\_type được dùng.
  + Cú pháp: default\_type [loại MIME];
  + Giá trị mặc định: text/plain
* types\_hash\_max\_size :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa kích thước tối đa của 1 entry trong bảng băm các loại MIME.
  + Cú pháp: types\_hash\_max\_size [giá trị số ];
  + Giá trị mặc định: 4k hoặc 8k (1 dòng của CPU cache)

### Các chỉ thị về giới hạn

* limit\_except :
  + Sử dụng trong khối : location
  + Chỉ thị này cho phép chúng ta ngăn chặn việc sử dụng của tất cả các phương thức HTTP, ngoại trừ những phương thức mà chúng ta cho phép 1 cách tường minh. Bên trong 1 khối location, chúng ta có thể giới hạn việc sử dụng 1 vài phương thức HTTP, như cấm client gửi các yêu cầu POST. Chúng ta cần định nghĩa 2 thành phần – đầu tiên, những phương thức không bị cấm, và thứ 2, những client được ảnh hưởng bởi việc giới hạn này:

*location /admin/ {*

*limit\_except GET {*

*allow 192.168.1.0/24;*

*deny all;*

*}*

*}*

* Cú pháp:

*limit\_except METHOD1 [METHOD2…] {*

*allow | deny | auth\_basic | auth\_basic\_user\_file | proxy\_pass | perl;*

*}*

* limit\_rate :
  + Sử dụng trong khối : server, http, location, if
  + Cho phép chúng ta giới hạn tỉ lệ truyền của các kết nối khách hàng cá nhân. Tỉ lệ này được thể hiện ở dạng số byte trên mỗi giây (bytes per second):

*limit\_rate 500k;*

* Điều này sẽ giới hạn tỉ lệ truyền kết nối là 500 KB/second. Nếu 1 khách hàng mở 2 kết nối, khách hàng sẽ được cho phép 2\*500 KB
* Cú pháp: limit\_rate [Giá trị dung lượng trong Nginx.]
* Giá trị mặc định: Không có
* limit\_rate\_after :
  + Sử dụng trong khối : server, http, location, if
  + Định nghĩa số lượng dữ liệu được truyền trước khi chỉ thị limit\_rate có tác dụng.:

*limit\_rate\_after 10m;*

* Nginx sẽ gửi 10 MB đầu tiên với tốc độ tối đa. Qua kích thước này, tỉ lệ truyền được giới hạn bởi giá trị được chỉ rõ trong khai báo limit\_rate. Tương tự với chỉ thị limit\_rate, thiết lập này chỉ áp dụng cho 1 kết nối.
* Cú pháp: limit\_rate\_after [Giá trị dung lượng trong Nginx.]
* Giá trị mặc định: Không có
* satisfy :
  + Sử dụng trong khối : location
  + Chỉ thị satisfy định nghĩa việc khách hàng được yêu cầu đáp ứng tất cả các điều kiện truy cập (satisfy all) hoặc ít nhất 1 điều kiện (satisfy any) để là khách hàng hợp lệ.:

*location /admin/ {*

*allow 192.168.1.0/24;*

*deny all;*

*auth\_basic “Authentication required”;*

*auth\_basic\_user\_file conf/htpasswd;*

*}*

* Trong ví dụ trên, có 2 điều kiện cho khách hàng để có thể truy cập:
  + Qua chỉ thị allow và deny, chúng ta chỉ cho phép các khách hàng có địa chỉ IP cục bộ, tất cả các khách hàng khác sẽ bị từ chối.
  + Qua chỉ thị auth\_basic và auth\_basic\_user\_file, chúng ta chỉ cho phép các khách hàng cung cấp 1 username và password hợp lệ
* Với satisfy all, khách hàng phải thoả mãn cả 2 điều kiện trên.
* Với satisfy any, khách hàng chỉ cần thoả mãn 1 trong 2 điều kiện trên
* Cú pháp: satisfy any | all;
* Giá trị mặc định: all
* internal :
  + Sử dụng trong khối : location
  + Chỉ thị này chỉ rõ rằng khối location này là nội bộ. Nói cách khác, các tài nguyên được chỉ rõ không thể được truy cập từ các yêu cầu bên ngoài.

*server {*

*location /admin/ {*

*internal;*

*}*

*}*

* Với cấu hình trên, khách hàng sẽ không thể truy cập vào đường dẫn http://abc.com/admin/. Những yêu cầu như vậy sẽ gặp lỗi HTTP 404 Not Found. Cách duy nhất để truy cập vào các tài nguyên này là qua các chuyển hướng nội bộ.

### Các chỉ thị về xử lý tệp tin và bộ nhớ đệm:

* disable\_symlinks :
  + Chỉ thị này cho phép chúng ta điều khiển cách Nginx xử lý các liên kết tượng trưng khi chúng được phục vụ. Mặc định (giá trị mặc định là off) các liên kết này được cho phép và Nginx sẽ xử lý chúng. Chúng ta có thể quyết định vô hiệu hóa việc cho phép các liên kết tượng trưng dưới những điều kiện khác nhau bằng việc chỉ rõ 1 trong những giá trị sau:
    - on: Nếu bất kỳ phần nào của URI là 1 liên kết tượng trưng, truy cập đến nó sẽ bị từ chối và Nginx trả về 1 trang lỗi HTTP 403.
    - if\_not\_owner: Tương tự với giá trị on ở trên, nhưng truy cập bị từ chối chỉ khi liên kết và đối tượng nó trỏ đến có owner khác nhau.
    - Thông số tùy chọn từ from = cho phép chúng ta chỉ rõ 1 phần của URL mà sẽ không được kiểm tra liên kết tượng trưng. Ví dụ, nếu ta khai báo là disable\_symlinks from=$document\_root; sẽ yêu cầu Nginx theo các liên kết tượng trưng trong URI đến thư mục $document\_root. Nếu 1 liên kết tượng trưng được tìm thấy trong các phần URI sau phần $document\_root đó, truy cập đến tập tin được yêu cầu sẽ bị từ chối.
  + Note : Symbolic Link – liên kết tượng trưng (hoặc symlink hay soft link) là 1 loại tập tin đặc biệt chứa thông tin tham chiếu đến 1 tập tin hoặc thư mục khác (targer) trong hệ thống dưới dạng 1 đường dẫn tuyệt đối hoặc tương đối.
* directio :
  + Sử dụng trong khối : server, http, location
  + Nếu chỉ thị này được cho phép, các tập tin với kích thước lớn hơn giá trị được chỉ rõ sẽ được đọc với cơ chế hệ thống Direct I/O. Điều này cho phép Nginx đọc dữ liệu từ các thiết bị lưu trữ và đặt chúng trực tiếp trong bộ nhớ mà không cần tiến trình đưa vào bộ nhớ đệm trung gian.
  + Cú pháp: directio [Giá trị dung lượng trong Nginx, hoặc là off];
  + Giá trị mặc định: off
* directio\_alignment :
  + Sử dụng trong khối : server, http, location
  + Thiết lập liên kết byte khi sử dụng directio. Thiết lập giá trị này là 4k nếu sử dụng XFS trên Linux.
  + Cú pháp: directio [Giá trị dung lượng trong Nginx ];
  + Giá trị mặc định: 512
* open\_file\_cache :
  + Sử dụng trong khối : server, http, location
  + Chỉ thị này cho phép chúng ta cho phép chúng ta sử dụng bộ nhớ cache lưu trữ thông tin về các tập tin được mở. Nó thực sự không lưu trữ nội dung của chính tập tin đó mà chỉ lưu trữ thông tin như:
    - Mô tả tập tin (kích thước, thời gian điều chỉnh, …).
    - Sự tồn tại của tập tin và thư mục.
    - Các lỗi của tập tin, như là Permission denied, File not found, … Lưu ý rằng chỉ thị này có thể bị vô hiệu hóa với chỉ thị open\_file\_cache\_errors.
  + Chỉ thị này chấp nhận 2 tham số:
    - max=X, với X là số mục mà bộ nhớ cache có thể lưu trữ. Nếu số lượng này bị vượt qua, các mục cũ hơn sẽ được xóa để lấy không gian cho các mục mới hơn.
    - inactive=Y (tùy chọn, không bắt buộc) với Y là số giây mà 1 mục trong bộ nhớ cache sẽ được lưu trữ. Mặc định, Nginx sẽ đợi 60 giây trước khi xóa 1 mục trong bộ nhớ cache. Nếu mục trong bộ nhớ cache được truy cập, bộ đếm thời gian được khởi động lại. Nếu mục trong bộ nhớ cache được truy cập nhiều hơn giá trị định nghĩa bởi chỉ thị open\_file\_cache\_min\_users, mục đó sẽ không bị xóa (đến khi Nginx hết bộ nhớ và quyết định xóa các mục cũ hơn).
  + Cú pháp: open\_file\_cache max=X [inactive=Y] | off

*open\_file\_cache max=5000 inactive=180;*

* Giá trị mặc định: off
* open\_file\_cache\_min\_uses :
  + Sử dụng trong khối : server, http, location
  + Mặc định: các mục trong chỉ thị open\_file\_cache được xóa sau 1 khoảng thời gian không được sử dụng (mặc định là 60 giây). Chúng ta có thể ngăn Nginx xóa các mục này. Chỉ thị này định nghĩa số lần 1 mục trong bộ nhớ cache phải được truy cập để có đủ điều để bảo vệ.

*open\_file\_cache\_min\_uses 3;*

Nếu 1 mục trong bộ nhớ cache được truy cập nhiều hơn 3 lần, nó chuyển trạng thái thành hoạt động vĩnh viễn và sẽ không bị xóa đến khi Nginx quyết định xóa các mục cũ hơn để giải phóng vùng nhớ.

Cú pháp: open\_file\_cache\_min\_uses [giá trị số];

Giá trị mặc định: 1

* open\_file\_cache\_valid :
  + Sử dụng trong khối : server, http, location
  + Cơ chế lưu trong bộ nhớ cache các tập tin mở là quan trọng, nhưng thông tin được lưu trong bộ nhớ cache nhanh chóng trở thành lỗi thời. Về phương diện này, thông tin cần được tái kiểm tra sau 1 khoảng thời gian ngắn. Chỉ thị này chỉ rõ số giây mà Nginx sẽ đợi trước khi tái xác nhận 1 mục trong bộ nhớ cache.
  + Cú pháp: open\_file\_cache\_valid [giá trị thời gian tính bằng giây]
  + Giá trị mặc định: 60s
* read\_ahead :
  + Sử dụng trong khối : server, http, location
  + Định nghĩa số byte sẽ được đọc trước từ các tập tin. Dưới các hệ điều hành dựa trên Linux, việc thiết lập chỉ thị này này 1 giá trị lớn hơn 0 sẽ cho phép việc đọc trước, nhưng giá trị thực sự chúng ta chị định không có tác dụng. Thiết lập chỉ thị này là 0 để vô hiệu hóa việc đọc trước.
  + Cú pháp: read\_ahead [Giá trị dung lượng trong Nginx.]
  + Giá trị mặc định: 0

### Các chỉ thị khác

* log\_not\_found :
  + Sử dụng trong khối : server, http, location
  + Cho phép hoặc vô hiệu hóa việc ghi nhận các lỗi HTTP 404 Not Found.
  + Cú pháp: log\_not\_found [on | off]
  + Giá trị mặc định: on
* log\_subrequest :
  + Sử dụng trong khối : server, http, location
  + Cho phép hoặc vô hiệu hóa việc ghi nhận các yêu cầu con được kích hoạt bởi các chuyển hướng nội bộ hoặc các yêu cầu SSI.
  + Cú pháp: log\_subrequest [on | off]
  + Giá trị mặc định: on
* merge\_slashes
  + Sử dụng trong khối : server, http, location
  + Cho phép chỉ thị này sẽ có ảnh hưởng đến việc sát nhập nhiều ký tự “/” liên tiếp trong URI.

*server {*

*[…]*

*server\_name abc.com;*

*location /documents/ {*

*type {}*

*default\_type text/plain;*

*}*

*}*

* Mặc định, nếu khách hàng cố gắng truy cập http://abc.com//documents/ (lưu ý // ở giữa URI), Nginx sẽ trả về lỗi HTTP 404 Not Found. Nếu cho phép chỉ thị này, 2 ký hiệu / sẽ được hợp thành 1 và mẫu location sẽ khớ
* Cú pháp: merge\_slashes [on | off]
* Giá trị mặc định: on
* msie\_padding :
  + Sử dụng trong khối : server, http, location
  + Chỉ thị này được dùng với các trình duyệt Microsoft Internet Explorer (MSIE) và Google Chrome. Trong trường hợp các trang lỗi (với mã lỗi là 400 hoặc cao hơn), nếu kích thước của dữ liệu phản hồi nhỏ hơn 512 Byte, những trình duyệt này sẽ hiển thị trang lỗi của chính những trình duyệt đó. Nếu cho phép chỉ thị này, phần thân của phản hồi với 1 mã trạng thái 400 hoặc cao hơn sẽ được đệm thêm đến 512 Byte.
  + Cú pháp: msie\_padding [on | off]
  + Giá trị mặc định: off
* resolver :
  + Sử dụng trong khối : server, http, location
  + Chỉ rõ các máy chủ phân giải tên miền (DNS) được sử dụng bởi Nginx để phân giải hostname của các địa chỉ IP và ngược lại. Các kết quả truy vấn DNS được lưu trong bộ nhớ cache trong 1 thời gian, hoặc được chỉ rõ bởi giá trị TTL (Time-to-live) của máy chủ DNS, hoặc được chỉ rõ bởi giá trị thời gian cho đối số hợp lệ.
  + Cú pháp: resolver [địa chỉ id], [valid=Time value]
  + Ví dụ :

*resolver 8.8.8.8 8.8.4.4 valid=1h; # use Google DNS and cache results for 1 hour*

* resolver\_timeout :
  + Sử dụng trong khối : server, http, location
  + Giới hạn thời gian cho 1 truy vấn phân giải hostname.
  + Cú pháp: resolver\_timeout [Giá trị thời gian trong Nginx (tính bằng giây)]
  + Giá trị mặc định: 30s
* server\_tokens :
  + Sử dụng trong khối : server, http, location
  + Chỉ thị này cho phép chúng ta định nghĩa việc cho phép/không cho phép Nginx thông báo cho khách hàng biết về phiên bản Nginx đang chạy tại máy chủ. Có 2 tình huống Nginx chỉ số phiên bản của nó là:
    - Trong phần server header của những phản hồi HTTP ( như là nginx/1.2.9). Nếu chúng ta thiết lập server\_tokens là off, phần server header sẽ chỉ có giá trị là Nginx.
    - Trên các trang lỗi, Nginx chỉ số phiên bản trong phần footer. Nếu chúng ta thiết lập server\_tokens là off, phần footer của trang lỗi sẽ chỉ là Nginx.
  + Cú pháp: server\_tokens [on | off]
  + Giá trị mặc định: on
* underscores\_in\_headers :
  + Sử dụng trong khối : server, http, location
  + Cho phép hoặc không cho phép các dấu gạch dưới (\_) trong các tên header HTTP tùy chọn. Nếu chỉ thị này được để là on, phần header ví dụ sau sẽ được xem là hợp lệ bởi Nginx: test\_header: value
  + Cú pháp: underscores\_in\_headers [on | off]
  + Giá trị mặc định: off
* variables\_hash\_max\_size :
  + Sử dụng trong khối : http
  + Cho phép hoặc không cho phép các dấu gạch dưới (\_) trong các tên header HTTP tùy chọn. Nếu chỉ thị này được để là on, phần header ví dụ sau sẽ được xem là hợp lệ bởi Nginx: test\_header: value
  + Cú pháp: variables\_hash\_max\_size [Giá trị số]
  + Giá trị mặc định: 512
* variables\_hash\_bucket\_size :
  + Sử dụng trong khối : http
  + Chỉ thị này cho phép chúng ta thiết lập kích thước vùng chứa cho các bảng băm chứa các biến.
  + Cú pháp: variables\_hash\_bucket\_size [Giá trị số]
  + Giá trị mặc định: 64 (hoặc 32, hoặc 128, phụ thuộc vào bộ nhớ cache của bộ xử lý).
* post\_action :
  + Sử dụng trong khối : http, server, location, if
  + Định nghĩa 1 hành động “sau khi hoàn thành”, tức 1 URI sẽ được gọi bởi Nginx sau khi yêu cầu đã được hoàn thành.
  + Cú pháp: URI hoặc 1 khối location được đặt tên.

*location /payment/ {*

*post\_action /scripts/done.php;}*

## Triển khai dịch vụ

### Cài đặt Nginx Web Server

* + - Để hiển thị các trang web cho khách truy cập trang web của chúng tôi, chúng tôi sẽ sử dụng Nginx, một máy chủ web hiệu suất cao. Chúng tôi sẽ sử dụng apttrình quản lý gói để có được phần mềm này.
    - Vì đây là lần đầu tiên chúng tôi sử dụng aptcho phiên này, hãy bắt đầu bằng cách cập nhật chỉ mục gói máy chủ của bạn. Sau đó, bạn có thể sử dụng apt installđể cài đặt Nginx:

sudo apt update

sudo apt install nginx

* + - Khi được nhắc, hãy nhập Yđể xác nhận rằng bạn muốn cài đặt Nginx. Sau khi cài đặt xong, máy chủ web Nginx sẽ hoạt động và chạy trên máy chủ Ubuntu 20.04 của bạn.
    - Nếu bạn đã ufwbật tường lửa, như được đề xuất trong hướng dẫn thiết lập máy chủ ban đầu của chúng tôi, bạn sẽ cần cho phép kết nối với Nginx. Nginx đăng ký một vài cấu hình ứng dụng UFW khác nhau khi cài đặt. Để kiểm tra cấu hình UFW nào khả dụng, hãy chạy:

sudo ufw app list

Kích hoạt tính năng này bằng cách gõ:

sudo ufw allow 'Nginx HTTP'

* + - Nhập địa chỉ mà bạn nhận được trong trình duyệt web của mình và nó sẽ đưa bạn đến trang đích mặc định của Nginx:



### Cài đặt MySQL

Bây giờ bạn đã có một máy chủ web đang hoạt động, bạn cần cài đặt hệ thống cơ sở dữ liệu để có thể lưu trữ và quản lý dữ liệu cho trang web của mình. MySQL là một hệ thống quản lý cơ sở dữ liệu phổ biến được sử dụng trong môi trường PHP.

Một lần nữa, sử dụng aptđể tải và cài đặt phần mềm này:

#Install MySQL and set a strong root password

apt-get install -y mysql-client mysql-server;

#Secure your Mysql installation

MYSQL\_ROOT\_PASSWORD=$1

while [ "$MYSQL\_ROOT\_PASSWORD" = "" ]

do

echo "Please provide your MYSQL\_ROOT\_PASSWORD:"

read MYSQL\_ROOT\_PASSWORD

done

if [ "$WP\_PASSWORD" = "" ]; then

WP\_PASSWORD=$MYSQL\_ROOT\_PASSWORD

fi

mysql -uroot <<MYSQL\_SCRIPT

CREATE DATABASE dblinux;

DELETE FROM mysql.user WHERE User='';

DELETE FROM mysql.user WHERE User='root' AND Host NOT IN ('localhost', '127.0.0.1', '::1');

DROP DATABASE IF EXISTS test;

DELETE FROM mysql.db WHERE Db='test' OR Db='test\\\_%';

FLUSH PRIVILEGES;

MYSQL\_SCRIPT

mysql -e "ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY '${MYSQL\_ROOT\_PASSWORD}';"

mysql -u root -p${MYSQL\_ROOT\_PASSWORD} dblinux < CodeWeb/php\_mvc\_pdo\_techstore.sql

echo $MYSQL\_ROOT\_PASSWORD

Khi bạn hoàn tất, hãy kiểm tra xem bạn có thể đăng nhập vào bảng điều khiển MySQL hay không bằng cách nhập:

Sudo mysql –u root –p(your\_pass)

Output

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 22

Server version: 8.0.19-0ubuntu5 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>

### Cài đặt PHP & PHP myadmin

Để cài đặt các gói php-fpmvà php-mysql, hãy chạy:

#install php7.4

apt install -y php7.4 php7.4-fpm php7.4-mysql php-common php7.4-cli php7.4-common php7.4-json php7.4-opcache php7.4-readline php7.4-mbstring php7.4-xml php7.4-gd php7.4-curl

systemctl enable php7.4-fpm

systemctl start php7.4-fpm

#install phpmyadmin

apt install phpmyadmin -y

### Định cấu hình nginx để sử dụng bộ xử lý PHP

Tạo thư mục web gốc cho **your\_domain** như sau:

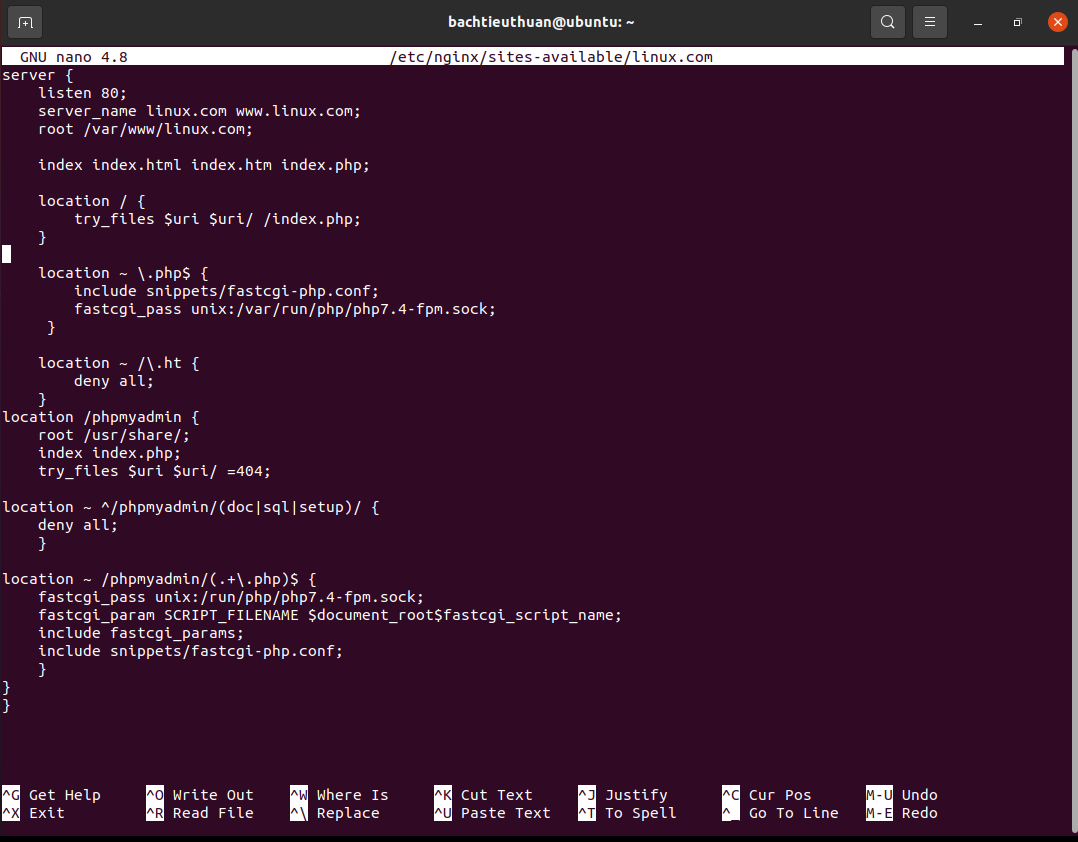
sudo mkdir /var/www/linux.com

Tiếp theo, chỉ định quyền sở hữu thư mục với biến môi trường $USER, sẽ tham chiếu người dùng hệ thống hiện tại của bạn:

sudo chown -R $USER:$USER /var/www/linux.com

Sau đó, mở tệp cấu hình mới trong thư mục của Nginx sites-availablebằng trình chỉnh sửa dòng lệnh ưa thích của bạn. Ở đây, chúng tôi sẽ sử dụng nano:

sudo nano /etc/nginx/sites-available/linux.com



Hình 2. : Set available/linux.com

Đây là chức năng của từng chỉ thị và khối vị trí này:

* listen— Xác định cổng Nginx sẽ lắng nghe. Trong trường hợp này, nó sẽ lắng nghe trên cổng 80, cổng mặc định cho HTTP.
* root— Xác định gốc tài liệu nơi lưu trữ các tệp do trang web này cung cấp.
* index— Xác định thứ tự Nginx sẽ ưu tiên các tệp chỉ mục cho trang web này. Thực tế phổ biến là liệt kê index.htmlcác tệp có quyền ưu tiên cao hơn index.phpcác tệp để cho phép nhanh chóng thiết lập trang đích bảo trì trong các ứng dụng PHP. Bạn có thể điều chỉnh các cài đặt này để phù hợp hơn với nhu cầu ứng dụng của mình.
* server\_name— Xác định những tên miền và/hoặc địa chỉ IP mà khối máy chủ này sẽ phản hồi. **Trỏ lệnh này tới tên miền hoặc địa chỉ IP công cộng của máy chủ của bạn.**
* location /— Khối vị trí đầu tiên bao gồm một lệnh try\_fileskiểm tra sự tồn tại của các tệp hoặc thư mục khớp với yêu cầu URI. Nếu Nginx không thể tìm thấy tài nguyên phù hợp, nó sẽ trả về lỗi 404.
* location ~ \.php$— Khối vị trí này xử lý quá trình xử lý PHP thực tế bằng cách trỏ Nginx tới fastcgi-php.conftệp cấu hình và php7.4-fpm.socktệp này khai báo ổ cắm nào được liên kết với php-fpm.
* location ~ /\.ht— Khối vị trí cuối cùng xử lý .htaccesscác tệp mà Nginx không xử lý. Bằng cách thêm deny allchỉ thị, nếu bất kỳ .htaccesstệp nào tình cờ tìm được đường vào thư mục gốc của tài liệu, chúng sẽ không được phục vụ cho khách truy cập.

Kích hoạt cấu hình của bạn bằng cách liên kết đến tệp cấu hình từ thư mục của Nginx sites-enabled:

sudo ln -s /etc/nginx/sites-available/linux.com /etc/nginx/sites-enabled/

Sau đó, hủy liên kết tệp cấu hình mặc định khỏi thư mục /sites-enabled/:

sudo unlink /etc/nginx/sites-enabled/default

Bạn có thể kiểm tra cấu hình của mình để tìm lỗi cú pháp bằng cách nhập:

sudo nginx –t

Khi bạn đã sẵn sàng, hãy tải lại Nginx để áp dụng các thay đổi:

sudo systemctl reload nginx

Trang web mới của hiện đang hoạt động nhưng thư mục gốc của trang web vẫn trống. Đưa code website vào và trang web sẽ hoạt động bình thường nếu không lỗi.

### Kiểm tra PHP với nginx

Bạn có thể làm điều này bằng cách tạo một tệp PHP thử nghiệm trong thư mục gốc của tài liệu. Mở một tệp mới có tên info.phptrong thư mục gốc của tài liệu trong trình soạn thảo văn bản của bạn:

nano /var/www/linux.com/info.php

Nhập hoặc dán các dòng sau vào tệp mới. Đây là mã PHP hợp lệ sẽ trả về thông tin về máy chủ của bạn:

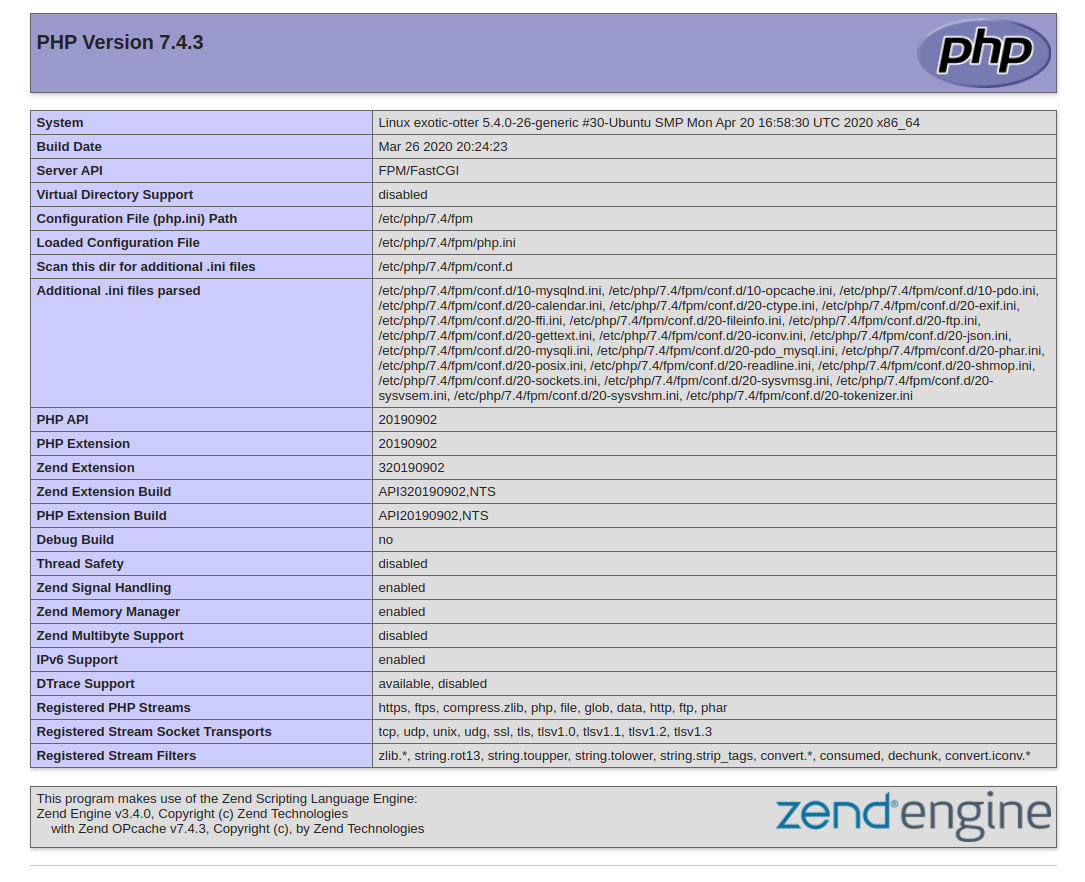
<?php

Phpinfo();

Khi bạn hoàn tất, hãy lưu và đóng tệp bằng cách nhập CTRL+ Xvà sau đó yvà ENTERđể xác nhận.

Bây giờ bạn có thể truy cập trang này trong trình duyệt web của mình bằng cách truy cập tên miền hoặc địa chỉ IP công khai mà bạn đã thiết lập trong tệp cấu hình Nginx của mình, sau đó /info.php

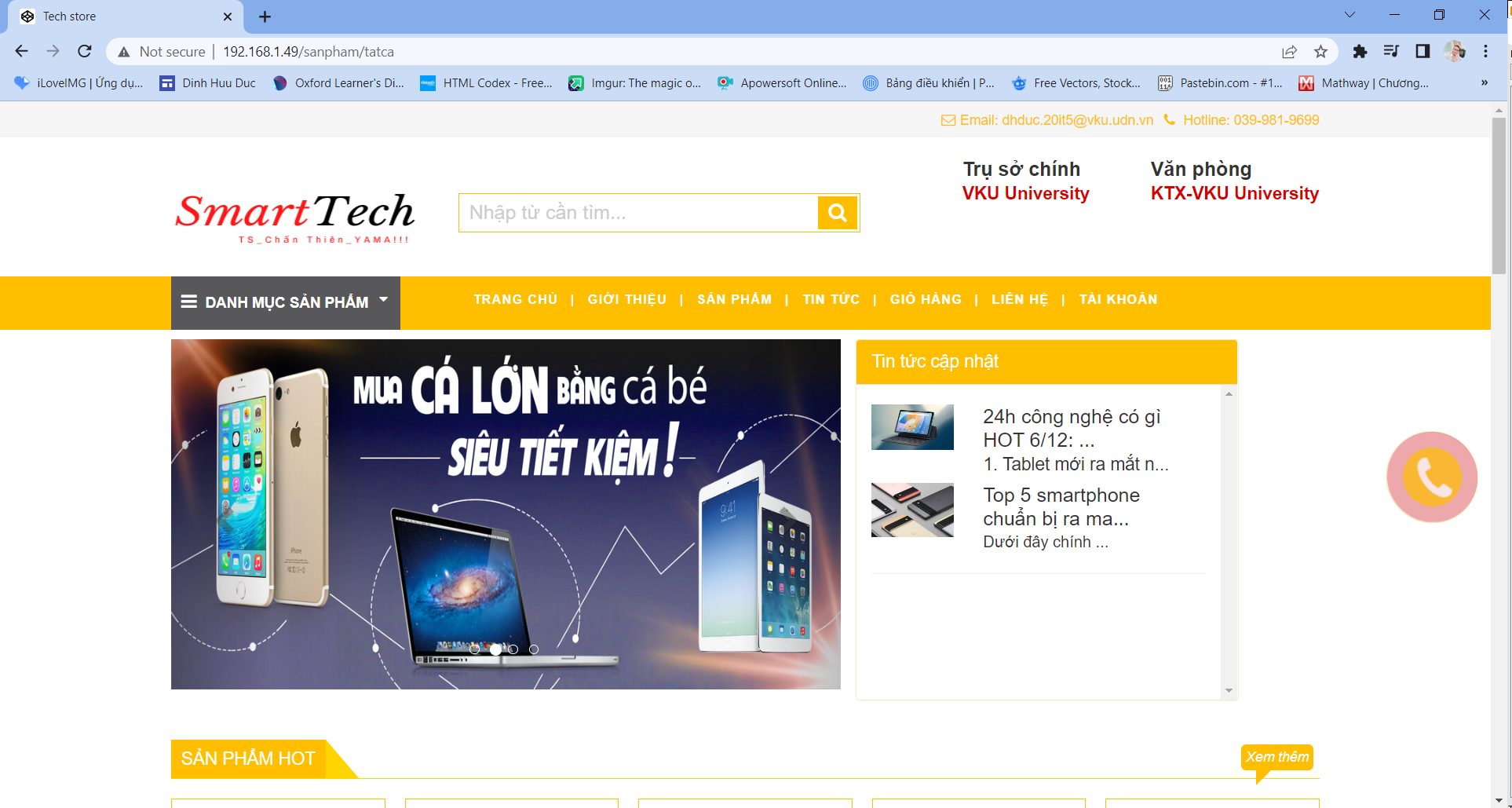
Bạn sẽ thấy một trang web chứa thông tin chi tiết về máy chủ của bạn:



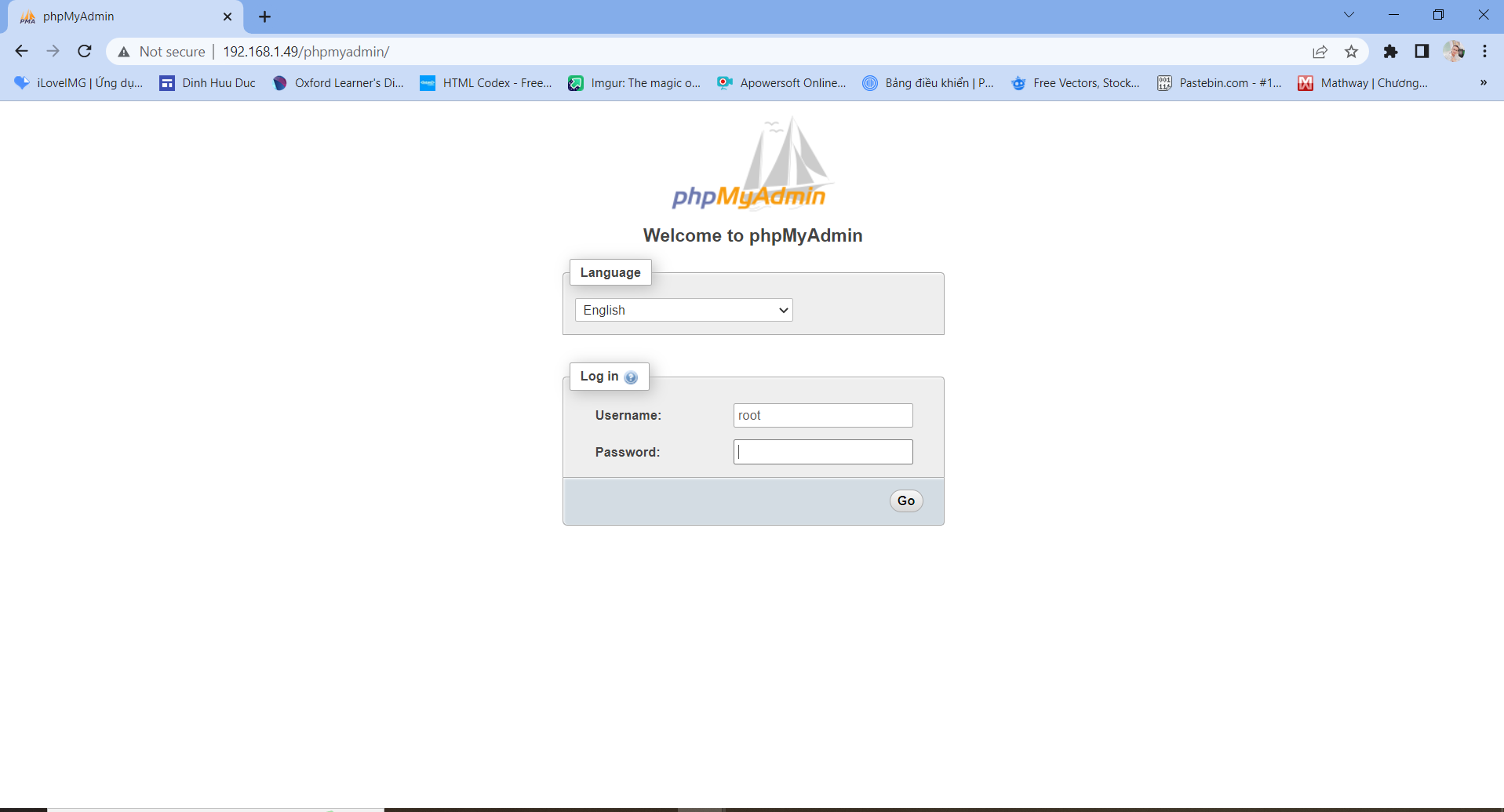
Hình 2. : PHPinfo

# TRIỂN KHAI VÀ ĐÁNH GIÁ

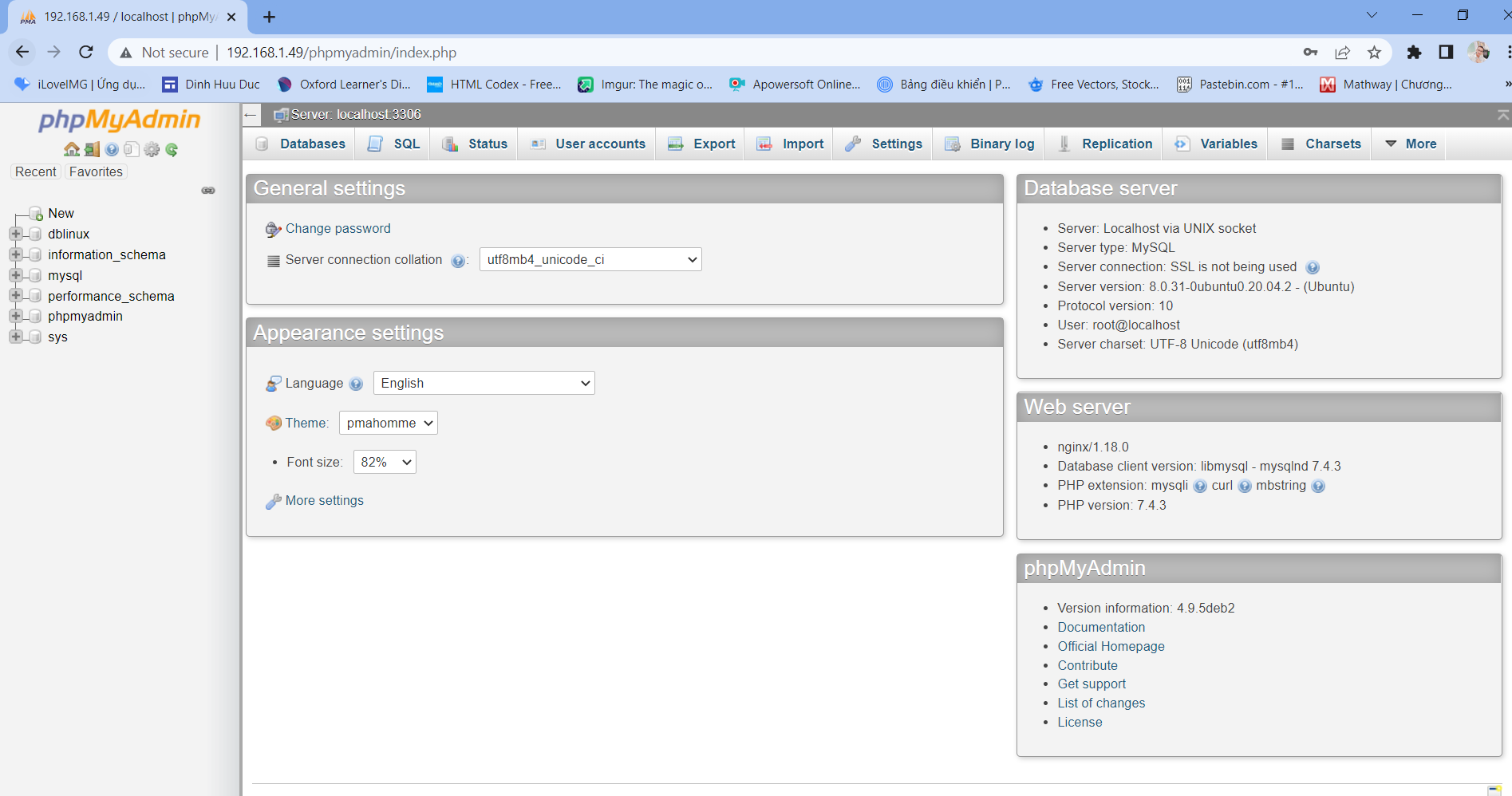
## Kết quả triển khai Web Server



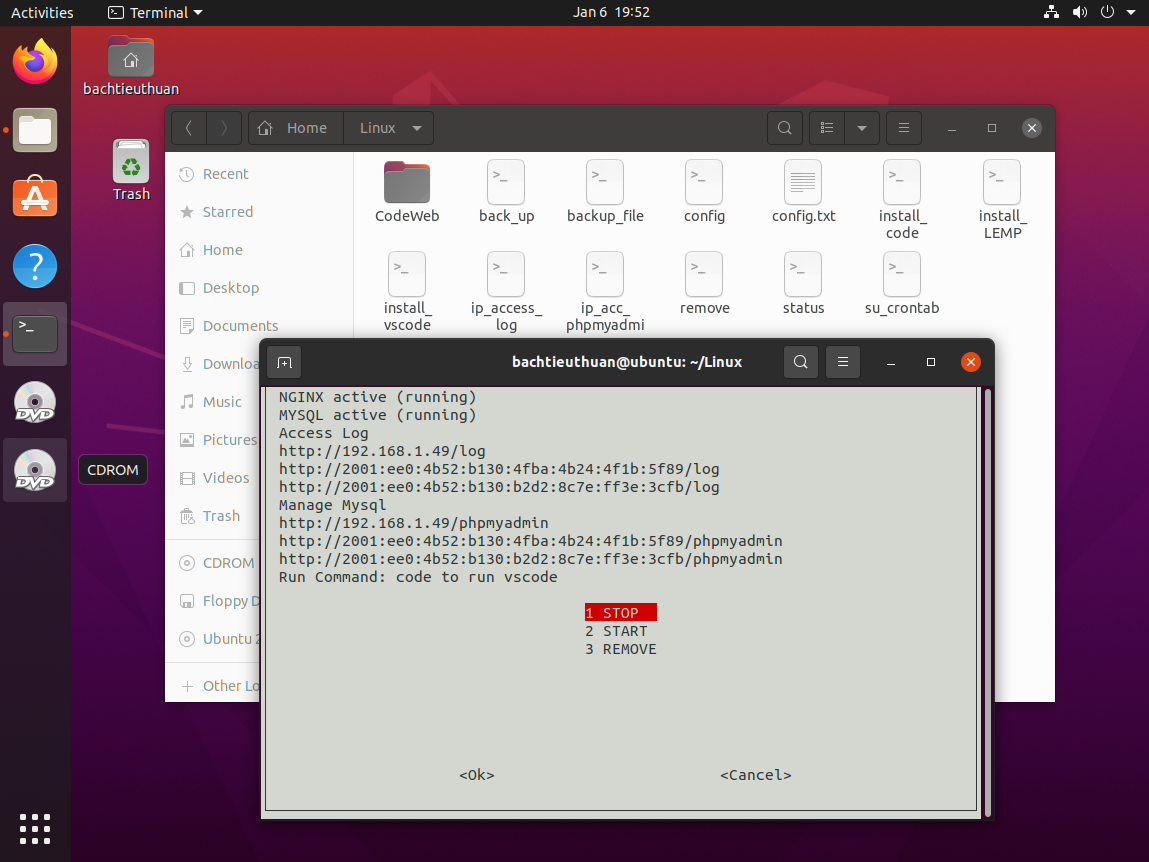
Hình 3.1: Giao diện web khi đã triển khai



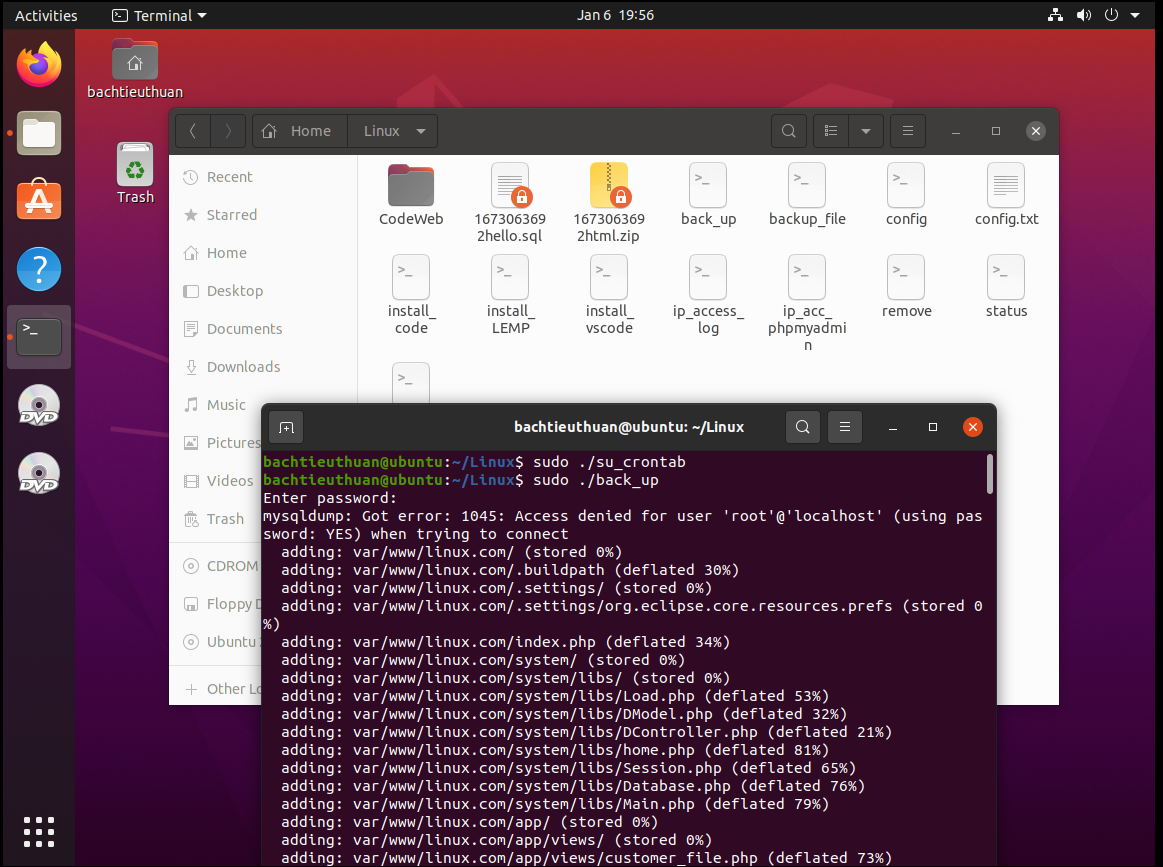
Hình 3.2: Giao diện web phpmyadmin



Hình 3.3: Giao diện web phpmyadmin khi đăng nhập



Hình 3.3: Giao diện status LEMP



Hình 3.4: Back up file

## Kết quả và đánh giá

### Kết quả đạt được

* Hoàn thiện được tính năng cơ bản của một chương trình quản lí Web Server.
* Giao diện được thiết kế đẹp mắt đơn giản và dễ sử dụng.
* Người dung đã có thể dễ dàng cài đặt một cách đơn giản nhất
* Có thể đáp ưng được các nhu cầu cần thiết của một dịch vụ Web Server Nginx

### Hướng phát triển

Mặc dù đã cố gắng, tìm hiểu các kiến thức đã học, kết hợp với thực tế và tra cứu các tài liệu liên quan đến đề tài “**THIẾT LẬP BỘ DỊCH VỤ LEMP ĐỂ TRIỂN KHAI DỊCH VỤ WEB CHO NGƯỜI DÙNG**” do hạn chế về thời gian, khả năng và kinh nghiệm nên đề tài không tránh khỏi những sai sót. Sau đây là hướng phát triển của em trong tương lai:

* Tìm hiểu sâu hơn về LEMP để có thể đáp ứng nhiều hơn nữa nhu cầu của người sử dụng, phát triển và tối ưu hóa hệ thống.
* Tìm hiểu thêm một số ngôn ngữ, các phần mềm ứng dụng để nâng cao giao diện đồ họa đẹp mắt, thân thiện hơn…
* Xây dựng ứng dụng quy mô lớn hơn với nhiều ứng dụng, chức năng…
* Thường xuyên cập nhật theo từng sự kiện lớn.
* Luôn cập nhật sản phẩm công nghệ mới và chất lượng, bắt kịp thời đại.
* Luôn theo dõi, chăm sóc khách hàng nhằm tăng độ uy tín.

# DANH MỤC TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | [https://viblo.asia/p/tim-hieu-va-huong-dan-setup-web-server-nginx-OREGwBwlvlN](https://viblo.asia/p/tim-hieu-va-huong-dan-setup-web-server-nginx-OREGwBwlvlN%20) |
| [2] | [https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-ubuntu-20-04](https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-ubuntu-20-04%20) |
| [3] | <https://dev.mysql.com/doc> |
| [4] | <https://github.com/lelong1210/lelong1210-script_lampp/blob/main/Config/apache2.conf> |
| [5] | <https://github.com/noamanahmed/LEMP/blob/master/templates/nginx/php/vhost.conf> |