# DOCKER

[DOCKER 1](#_Toc16258589)

[I. CÀI ĐẶT PHẦN MỀM CẦN THIẾT 2](#_Toc16258590)

[II. CÀI ĐẶT MÔI TRƯỜNG CHO DỰ ÁN 2](#_Toc16258591)

[CẤU TRÚC THƯ MỤC CHÍNH 2](#_Toc16258592)

[MYSQL 3](#_Toc16258593)

[NGINX 5](#_Toc16258594)

[PHP-FPM 11](#_Toc16258595)

[APACHE: 13](#_Toc16258596)

[DOCKER-COMPOSE 14](#_Toc16258597)

[CÁC CÂU LỆNH TRIỂN KHAI VỚI DOCKER-COMPOSE 17](#_Toc16258598)

[CÁC CÂU LỆNH THAO TÁC VỚI CONTAINER 18](#_Toc16258599)

[III. KẾT QUẢ 20](#_Toc16258600)

[IV. CÁC LỖI CÓ THỂ GẶP 21](#_Toc16258601)

# CÀI ĐẶT PHẦN MỀM CẦN THIẾT

Đối với macOs: Docker desktop for Mac : <https://hub.docker.com/editions/community/docker-ce-desktop-mac>

Đối với Window:

<https://hub.docker.com/editions/community/docker-ce-desktop-windows>

Check kiểm tra việc cài đặt thành công, mở terminal: docker –v, docker-compose -v

# CÀI ĐẶT MÔI TRƯỜNG CHO DỰ ÁN

Hướng dẫn dựa trên yêu cầu cài đặt môi trường như sau : PHP 7.2, NGINX và MYSQL

## CẤU TRÚC THƯ MỤC CHÍNH

Your\_project

docker

nginx

Dockerfile

mysql

Dockerfile

…

docker-compose.yml

…

Thư mục **docker** : chứa thiết lập để giúp Docker tạo môi trường.

Thư mục **nginx, mysql, …**  hoặc các thư mục khác chứa tập tin **Dockerfile :**

Docker dựa vào **Dockerfile** để tạo các image, khi chạy các image tạo ra container tướng ứng. Dockerfile giúp chạy những câu lệnh như : cài đặt module cho PHP, sao chép thư mực hay tập tin vào bên trong container.

Thông tin thêm về Dockerfile: <https://docs.docker.com/engine/reference/builder/>

Tập tin **docker-compose.yml :**  Docker sử dụng để tạo ra service. Giúp kết nối các container lại với nhau.

## MYSQL

Thông tin về mysql trên Docker: <https://hub.docker.com/_/mysql>

Hay <https://dev.mysql.com/doc/mysql-installation-excerpt/8.0/en/linux-installation-docker.html>

**Cấu trúc thư mục để xuất:**

mysql

docker-entrypoint-initdb.d

my.cnf

Dockerfile

* docker-entrypoint-initdb.d : chứa các tập tin .sh, .sql, được thực thi mỗi lần tạo ra container.
* my.cnf : chứa cài đặt mysql
* Dockerfile: chứa thiệt lập container

Ví dụ:

Trong thư mục **docker-entrypoint-initdb.d** tạo ra tập tin **create\_user\_demo.sql**  nội dung:

# If there are any .sh or .sql scripts you want to run on the database immediately after it has been created,

# you can put them into a host directory and then mount the directory at /docker-entrypoint-initdb.d/ inside the container.

# For example, for a MySQL Server container https://dev.mysql.com/doc/mysql-installation-excerpt/8.0/en/docker-mysql-more-topics.html

# file .sh, .sql will be auto run when conatainer be created and started

CREATE USER 'caeruxlab'@'%' IDENTIFIED BY '.caerux';

CREATE DATABASE IF NOT EXISTS `db\_demo\_1` ;

GRANT ALL ON `db\_demo\_1`.\* TO 'caeruxlab'@'%' ;

FLUSH PRIVILEGES;

**My.cnf**

# The MySQL Client configuration file.

#

# For explanations see

# http://dev.mysql.com/doc/mysql/en/server-system-variables.html

[mysql]

[mysqld]

sql-mode="STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION"

character-set-server=utf8

pid-file = /var/run/mysqld/mysqld.pid

socket = /var/run/mysqld/mysqld.sock

datadir = /var/lib/mysql

secure-file-priv= NULL

# Disabling symbolic-links is recommended to prevent assorted security risks

symbolic-links=0

# Custom config should go here

!includedir /etc/mysql/conf.d/

**Dockerfile**

# Xác định base image và phiên bản được sử dụng ở đây là mysql và phiên bản : 8.0.17

FROM mysql:8.0.17

LABEL maintainer="CaeruxLab-TA"

#Sao chép tập tin my.cnf từ bên ngoài vào container thay thế cho my.cnf mặc định của mysql

#COPY my.cnf /etc/mysql/conf.d/my.cnf

CMD ["mysqld"]

# Mở Port 3306

EXPOSE 3306

## NGINX

Thông tin về ngix trên Docker : <https://hub.docker.com/_/nginx>

**Cấu trúc đề xuất:**

nginx

conf

site-available

nginx.conf

Dockerfile

* conf : thư mục này chứa các tập tin thiết lập của nginx.
* sites-available: thư mục này chứa các tập tin thiết lập vitualhost.

Thư mục **conf:**

Ví dụ tạo thêm tập tin **defaul.conf** thay thế tập tin gốc của nginx. Sau đó để cập nhật thay đổi chỉ cần thêm **ADD** vào file Dockerfile và build lại image.

**Nginx.conf**

user nginx;

worker\_processes auto;

pid /run/nginx.pid;

#include /etc/nginx/modules-enabled/\*.conf;

events {

worker\_connections 768;

multi\_accept on;

}

http {

##

# Basic Settings

##

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

keepalive\_timeout 65;

types\_hash\_max\_size 2048;

# server\_tokens off;

server\_names\_hash\_bucket\_size 64;

# server\_name\_in\_redirect off;

include /etc/nginx/mime.types;

default\_type application/octet-stream;

##

# SSL Settings

##

#ssl\_protocols TLSv1 TLSv1.1 TLSv1.2; # Dropping SSLv3, ref: POODLE

#ssl\_prefer\_server\_ciphers on;

##

# Logging Settings

##

access\_log /var/log/nginx/access.log;

error\_log /var/log/nginx/error.log;

##

# Gzip Settings

##

gzip on;

# gzip\_vary on;

# gzip\_proxied any;

# gzip\_comp\_level 6;

# gzip\_buffers 16 8k;

# gzip\_http\_version 1.1;

# gzip\_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss text/javascript;

##

# Virtual Host Configs

##

include /etc/nginx/conf.d/\*.conf;

include /etc/nginx/sites-enabled/\*;

}

#mail {

# # See sample authentication script at:

# # http://wiki.nginx.org/ImapAuthenticateWithApachePhpScript

#

# # auth\_http localhost/auth.php;

# # pop3\_capabilities "TOP" "USER";

# # imap\_capabilities "IMAP4rev1" "UIDPLUS";

#

# server {

# listen localhost:110;

# protocol pop3;

# proxy on;

# }

#

# server {

# listen localhost:143;

# protocol imap;

# proxy on;

# }

#}

**default.conf**

server {

listen 80;

root /home/projects/demo-docker/;

error\_page 500 502 503 504 /50x.html;

gzip on;

gzip\_types image/png image/gif image/jpeg text/javascript text/css;

gzip\_min\_length 1000;

gzip\_proxied any;

gunzip on;

location / {

index index.php index.html index.htm;

try\_files $uri $uri/ /index.php?$query\_string;

proxy\_read\_timeout 3600;

}

location = /50x.html {

root /usr/share/nginx/html;

}

location ~ \.php$ {

fastcgi\_index index.php;

fastcgi\_pass php\_fpm\_caerux:9000;

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param SCRIPT\_FILENAME /var/www/$fastcgi\_script\_name;

}

location ~ .\*\.(jpg|gif|png|css|js|ico|woff) {

expires 10d;

}

}

Thư mục **sites-available:** Thêm một tập tin cấu hình virtualhost như demo\_virtualhost.conf

**demo\_virtualhost.conf :**

server {

listen 80;

server\_name .caeruxlab.test;

root /home/projects/demo-docker/;

index index.html index.htm index.php;

charset utf-8;

location / {

try\_files $uri $uri/ /index.php?$query\_string;

}

location = /favicon.ico { access\_log off; log\_not\_found off; }

location = /robots.txt { access\_log off; log\_not\_found off; }

access\_log off;

error\_log /var/log/nginx/caeruxlab.test-error.log error;

sendfile off;

client\_max\_body\_size 100m;

location ~ \.php$ {

fastcgi\_index index.php;

fastcgi\_split\_path\_info ^(.+\.php)(/.+)$;

fastcgi\_pass php\_fpm\_caerux:9000;

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param SCRIPT\_FILENAME /var/www/html/$fastcgi\_script\_name;

fastcgi\_intercept\_errors off;

fastcgi\_buffer\_size 16k;

fastcgi\_buffers 4 16k;

fastcgi\_connect\_timeout 300;

fastcgi\_send\_timeout 300;

fastcgi\_read\_timeout 300;

}

location ~ /\.ht {

deny all;

}

# ssl\_certificate /etc/nginx/ssl/dengen.test.crt;

# ssl\_certificate\_key /etc/nginx/ssl/dengen.test.key;

}

**Dockerfile:**

FROM nginx:alpine

LABEL maintainer="CaeruxLab-TA"

#RUN giúp thực hiện các câu lệnh trong lúc build image.

RUN apk update \

&& apk upgrade \

&& apk --update add logrotate \

&& apk add --no-cache openssl \

&& apk add --no-cache bash

# Copy our nginx config

RUN rm -Rf /etc/nginx/nginx.conf

RUN rm -Rf /etc/nginx/conf.d/default.conf

#ADD thực hiện công việc chính là sao chép. ở trên ta đã tạo tập tin là nginx.conf

ADD conf/nginx.conf /etc/nginx/nginx.conf

ADD conf/default.conf /etc/nginx/conf.d/default.conf

# nginx site conf

RUN mkdir -p /etc/nginx/sites-available/ && \

mkdir -p /etc/nginx/sites-enabled/ && \

mkdir -p /etc/nginx/ssl/

# Enable site / vitualhost

ADD sites-available/demo\_vitualhost.conf /etc/nginx/sites-available/demo\_vitualhost.conf

RUN ln -s /etc/nginx/sites-available/demo\_vitualhost.conf /etc/nginx/sites-enabled/demo\_vitualhost.conf

EXPOSE 80 443

## PHP-FPM

Thông tin về php: <https://hub.docker.com/_/php>

Chúng ta sử dụng một **image variants** của php: php:<version>-fpm

**Cấu trúc thư mục đề xuất:**

php

Dockerfile

php.ini

Dựa theo cấu trúc thư mục chính, trong thư mục php có: Dockerfile

Chúng ta có thể tạo thêm tập tin **php.ini** để dễ dàng thay đổi thiết lập của php.

**php.ini**

;;;;;;;;;;;;;;;;

; File Uploads ;

;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.

; http://php.net/file-uploads

file\_uploads = On

; Temporary directory for HTTP uploaded files (will use system default if not

; specified).

; http://php.net/upload-tmp-dir

;upload\_tmp\_dir =

; Maximum allowed size for uploaded files.

; http://php.net/upload-max-filesize

upload\_max\_filesize = 20M ;value default 2M

; Maximum number of files that can be uploaded via a single request

max\_file\_uploads = 200 ;value default 20

**Dockerfile:**

FROM php:7.2-fpm

LABEL maintainer="CaeruxLab-TA"

RUN apt-get update && apt-get install -y \

libfreetype6-dev \

libjpeg62-turbo-dev \

libpng-dev \

&& docker-php-ext-install -j$(nproc) bcmath \

&& docker-php-ext-configure gd --with-freetype-dir=/usr/include/ --with-jpeg-dir=/usr/include/ \

&& docker-php-ext-install -j$(nproc) gd

ADD php.ini /usr/local/etc/php/

## APACHE:

Thông tin về apche container: <https://hub.docker.com/_/php>

Chúng ta sẽ sử dụng : **php:<version>-apache** là **image variants** (biến thể của PHP)

**Cấu trúc thư mục đề xuất:**

apache

conf-available: thư mục chứa các tập tin config của apche

sites-available: thư mục chứa các tập tin config virtualhost của apche

Dockerfile: chứa các thông tin config container

**Thư mục** conf-available:  
Tạo ra một tập tin demo-virtualhost.conf, nội dung :

<VirtualHost \*:80>

DocumentRoot "/var/www/html"

ServerName www.caeruxlab-apche.test

# Other directives here

</VirtualHost>

Để những tập tin trong hai thư mục **conf-available, sites-available**  có hiệu lực trong **apache**  cần thực hiện 2 câu lệnh lần lượt RUN a2enconf name\_file\_conf,

RUN a2ensite name\_file\_virtualhost việc này được thực hiện tập tin **Dockerfile** bên dưới:

**Dockerfile:**

FROM php:7.2-apache

# a2enconf, a2disconf - Enable or disable an apache2 configuration file

ADD /conf-available/demo-config.conf /etc/apache2/conf-available

RUN a2enconf demo-config

# a2ensite, a2dissite - Enable or disable an apache2 site / virtual host

ADD /sites-available/demo-vitualhost.conf /etc/apache2/sites-available

RUN a2ensite demo-vitualhost

# a2enmod, a2dismod - Enable or disable an apache2 module

RUN a2enmod ssl

## DOCKER-COMPOSE

Thông tin docker-compose: <https://docs.docker.com/compose/compose-file/>

**docker-compose:**

version: "3.7"

# Tạo một network để các container thuộc network này có thể giao tiếp với nhau.

networks:

mynet:

driver: bridge

services:

### NGINX Server #########################################

nginx:

build:

#Xác định đường dẫn tới tập tin Dockerfile, tạo ra container cũng như image

context: ./docker/nginx

container\_name: nginx\_caerux

#Chia sẻ thư mục máy thực(host) với thư mục trong container

volumes:

- ./:/home/projects/demo-docker

depends\_on:

- php-fpm

# Xác định PORT lắng nghe trên HOST:CONTAINER.

ports:

- "80:80"

- "443:443"

# Xác định container này kết nối với mạng mynet, các container cần thuộc chung một network để có thể giao tiếp được với nhau.

networks:

- mynet

### PHP-FPM ##############################################

php-fpm:

build:

context: ./docker/php

volumes:

- ./:/var/www

container\_name: php\_fpm\_caerux

#Mở port của service này là 9000

expose:

- "9000"

networks:

- mynet

### MySQL ################################################

mysql:

build:

context: ./docker/mysql

container\_name: mysql\_caerux

environment:

- MYSQL\_ALLOW\_EMPTY\_PASSWORD=true

volumes:

- ${MYSQL\_ENTRYPOINT\_INITDB}:/docker-entrypoint-initdb.d

- ./docker/mysql/my.cnf:/etc/mysql/my.cnf

ports:

- "3306:3306"

networks:

- mynet

## IF YOU WANT TO USE APCHE FOR WEB PROXY, YOU WILL REMOVE COMMENT

## APACHE ################################################

# apache:

# build:

# context: ./docker/apache

# container\_name: apache\_caerux

# volumes:

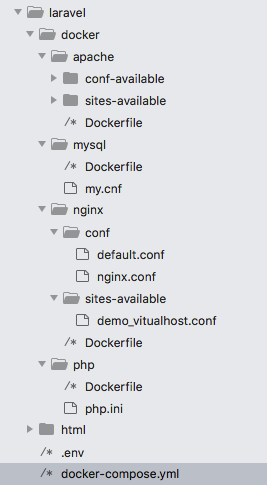
# - ./:/var/www/html

# ports:

# - "8000:80"

# networks:

# - mynet

Chúng ta có câu trúc thư mục như sau :  


## CÁC CÂU LỆNH TRIỂN KHAI VỚI DOCKER-COMPOSE

Tại thư mục chứa tập tin **docker-compose.yml**:  
**docker-compose up --build** : Build các image và container cho service.

**docker-compose up -h :** Để biết thêm về các option của câu lệnh docker-compose up

**Lưu ý :** khi sử dụng option **–d (--detach)** :giúp ẩn đi quá trình build ,nó cũng ẩn cả những lỗi khi build. Để dễ dàng cho quá trình debug chúng ta không sử dụng option –d này.Cách khác , có thể dụng dụng **docker-compose logs**  để xem lại log.

**docker-compose start :** Sau khi build thành công, để khởi động container cho service

**docker-compose stop:** Dừng các container của service

**docker stop container\_name :** Dừng container bất kỳ

**docker-compose down :** Dừng các containers và xoá containers, networks, volumes, and images do **docker-compose up**  tạo ra

**docker-compose ps :** Hiển thị danh sách các container

**docker-compose build --no-cache : Cài mới lại, xoá cache cũ**

## CÁC CÂU LỆNH THAO TÁC VỚI CONTAINER

**docker –h :** Hiển thị toàn bộ câu lệnh của **docker**.

**docker ps :** Hiển thị danh sách containers

Thực hiện câu lệnh với container đang chạy: **docker exec**

**docker exec -it name\_container /bin/bash**

Ví dụ : Sử dụng câu lệnh này để ssh vào container và chạy **bash**.

**docker  network ls** : Hiển thị danh sách các network

**docker network inspect name\_network :** Xem chi tiết một network cụ thể

(Dùng câu lệnh này để kiểm tra  xem các container có thuộc vào network không vì các container muốn giao tiếp được với nhau cần thuộc network)

**docker start name\_container :** Bật một container hoặc nhiều container

**docker stop name\_container:** Tắt một container hoặc nhiều container

**GIẢI THÍCH CƠ CHẾ CÁC CONTAINER GIAO TIẾP VỚI NHAU**  
Như đã nhắc đến ở trên, các container kết nối chung vào một network để có thể giao tiếp được với nhau.

Bài toán ở trên:

Các container trên đều thuộc về một network là : **mynet**

Để nginx(container nginx) có thể chạy được các tập tin **.php**, nginx cần gửi một yêu cầu (request) sang php-fpm (container php-fpm) bằng giao thức tcp.

Giải quyết : Trong config server của nginx:   
server {

…

location ~ \.php$ {

fastcgi\_pass php\_fpm\_caerux:9000; #name\_container:port\_container

}

…

}

Với php**\_fpm\_caerux:9000**  là tên và port được mở của container, dựa vào nó docker có thể xác định tới php-fpm container và thực hiện việc chạy tập tin **.php**.

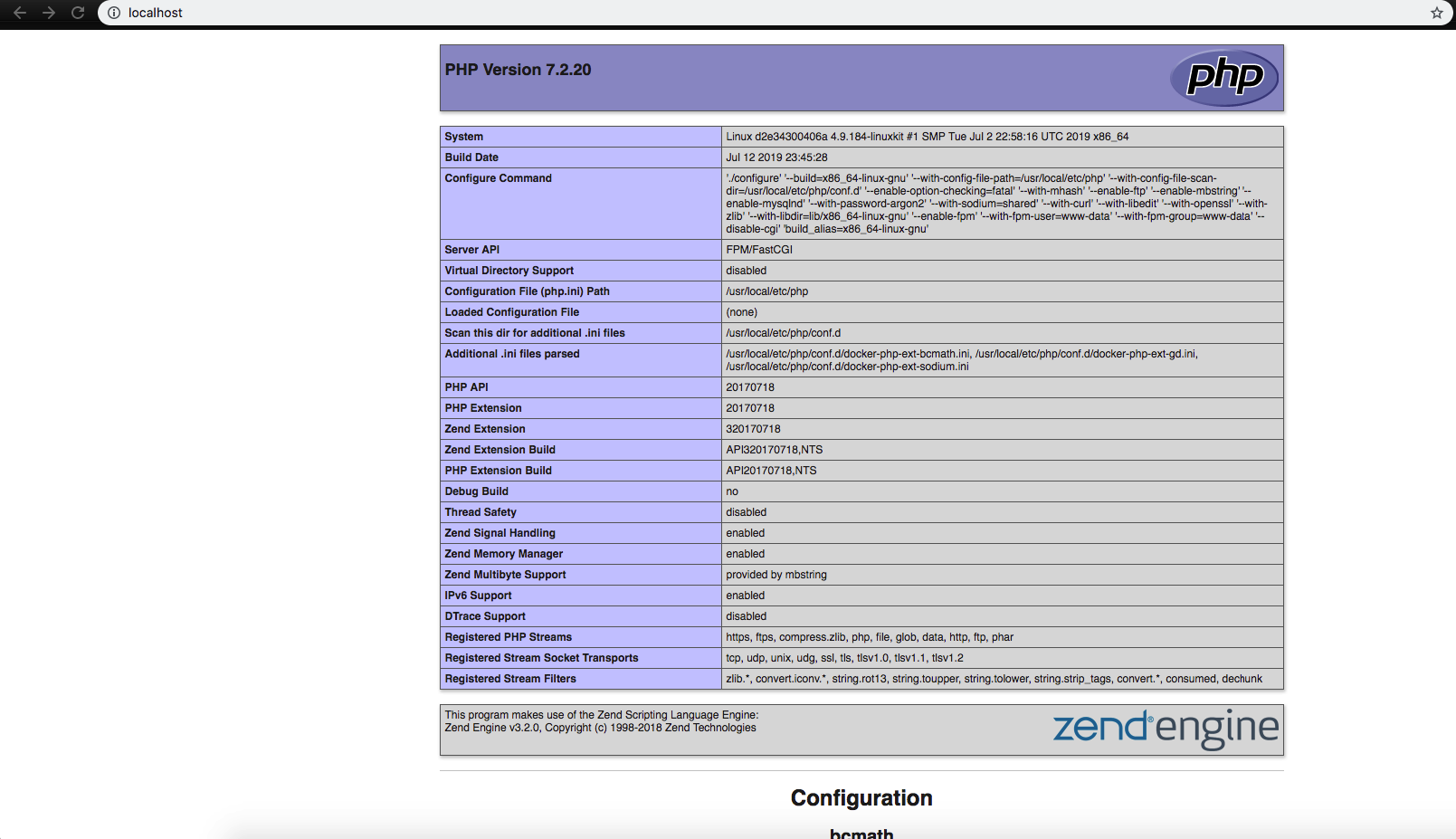
# KẾT QUẢ

Demo: <https://github.com/tanhit243/docker>

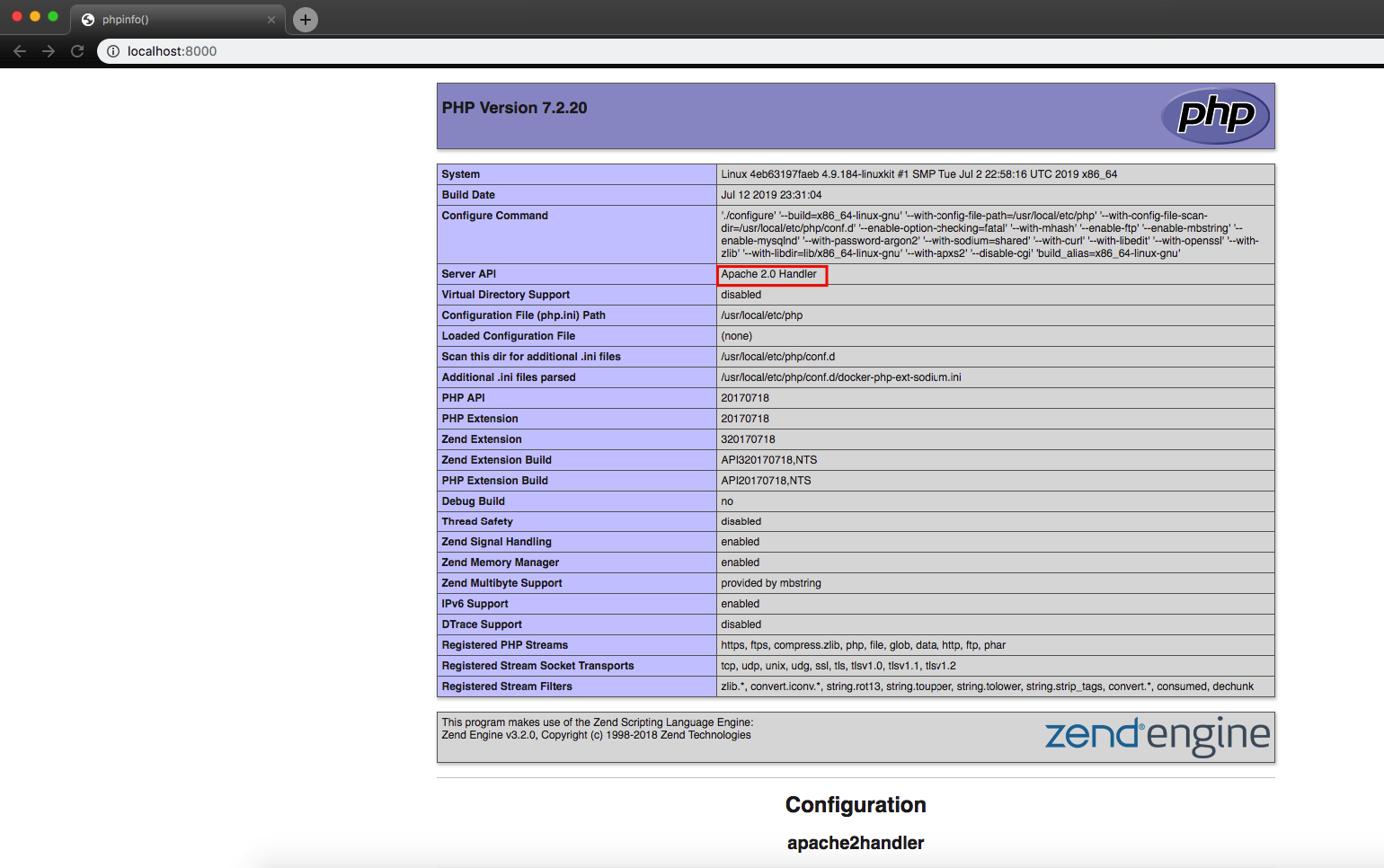
Để kiểm tra : chúng ta tạo một file index.php với nội dung: <?php phpinfo(); ?>

Mở trình duyệt, truy cập: localhost:80

Vì nginx được thiết lập port là 80:80 trong tập tin docker-compose.yml



Đối với apache: vì chúng ta đã thiết lập trong tập tin docker-compose.yml : 8000:80. Nên khi sử dụng để truy cập vào apache cần sử dụng port 8000.



# CÁC LỖI CÓ THỂ GẶP

**Khi gặp lỗi khi build**

1. **Lỗi StoreError**

dockerpycreds.errors.StoreError: Credentials store docker-credential-osxkeychain exited with "The specified

Tham khảo:

Bỏ chọn Securely store Docker logins in macOS keychain và sửa file

vi ~/.docker/config.json xoá "credsStore": "osxkeychain"

<https://qiita.com/ryosukehujisawa/items/3b6ad2124d166b3d052c>

1. **Lỗi mysql login trên sequel pro**

Khi sử dụng mysql > 8.0 , mysql thay đổi cơ chế xác thực từ mysql\_native\_password thành caching\_sha2\_password . Lỗi phát sinh do sequel pro chưa cập nhật plugin mới để hỗ trợ.

Tham khảo:

1. caching\_sha2\_password đổi thành mysql\_native\_password.

<https://qiita.com/yusuke_dev/items/7f0ca12ced72363f9448>

1. Sử dụng navicat thay cho sequel pro.