

Cómo automatizar el proceso de construcción y configuración de una imagen utilizando un archivo Dockerfile

Este PDF muestra cómo automatizar el proceso de construcción y configuración de una imagen utilizando un archivo Dockerfile en lugar de ejecutar comandos manualmente.

Para este caso se creará una imagen con sistema operativo ubuntu, con un servidor apache funcionando por el puerto 80. En la siguiente imagen se muestra el código de ejemplo usado en el Dockerfile. En el caso de Windows puede utilizar el editor de texto que considere más conveniente pero deberá tener en cuenta la ubicación del archivo, este deberá llamarse Dockerfile y adicionalmente este archivo no puede tener ninguna extensión.

```
C:\Users\ADMIN\test\Dockerfile - Notepad++
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas
Dockerfile
1 FROM ubuntu:latest
2 MAINTAINER JONATHAN jonathanga@mieena.edu.co
3 RUN apt-get update
4 RUN apt-get -y install mysql-server
```

Se puede observar que la imagen se construirá con la última versión del sistema operativo ubuntu, en segundo lugar se indica el nombre y correo electrónico de la persona que genera la imagen nueva, en tercer lugar se ejecuta el comando **apt-get update** para actualizar los repositorios del sistema operativo ubuntu de la imagen, y finalmente se ejecuta el comando **apt-get -y install mysql-server** para ejecutar en la imagen la instalación de mysql, automatizando así todos los pasos ejecutados anteriormente pero desde un archivo Dockerfile.

Para ejecutar el archivo Dockerfile se ejecuta el siguiente comando:

docker build -t nombreNuevaImagen ./directorioContenedor/

```
Windows PowerShell
PS C:\Users\ADMIN> docker build -t mysqlimage .\test\
[+] Building 0.1s (7/7) FINISHED
    docker: Build failed: No such file or directory: /nonexistent: No such file or directory
    docker: Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
    docker: PS C:\Users\ADMIN>
```

Donde **nombreNuevaImagen** corresponde al nombre de la nueva imagen a generar a partir de las sentencias del archivo Dockerfile y **./directorioContenedor/** corresponde al directorio de la máquina host donde se encuentra el archivo Dockerfile.

```
root@b0d8c6ec283a:/# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mysqlimage           latest              3c7403871170       14 minutes ago     689MB
jonathanga/ubuntu_mysql latest             6ddc7f0c2639       2 hours ago        699MB
jonathanga/docker10tutorial latest            c143c547c834       7 hours ago        28MB
docker10tutorial    latest            c143c547c834       7 hours ago        28MB
alpine/git           latest            b0f176fa3f8d       2 weeks ago        25.1MB
ubuntu               latest            7d8da2d69a15       7 weeks ago        72.7MB

PS C:\Users\ADMIN> docker run -it mysqlimage
root@b0d8c6ec283a:/# service mysql start
* Starting MySQL database server mysqld
su: warning: cannot change directory to /nonexistent: No such file or directory

root@b0d8c6ec283a:/# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.25-0ubuntu20.04.1 (Ubuntu)

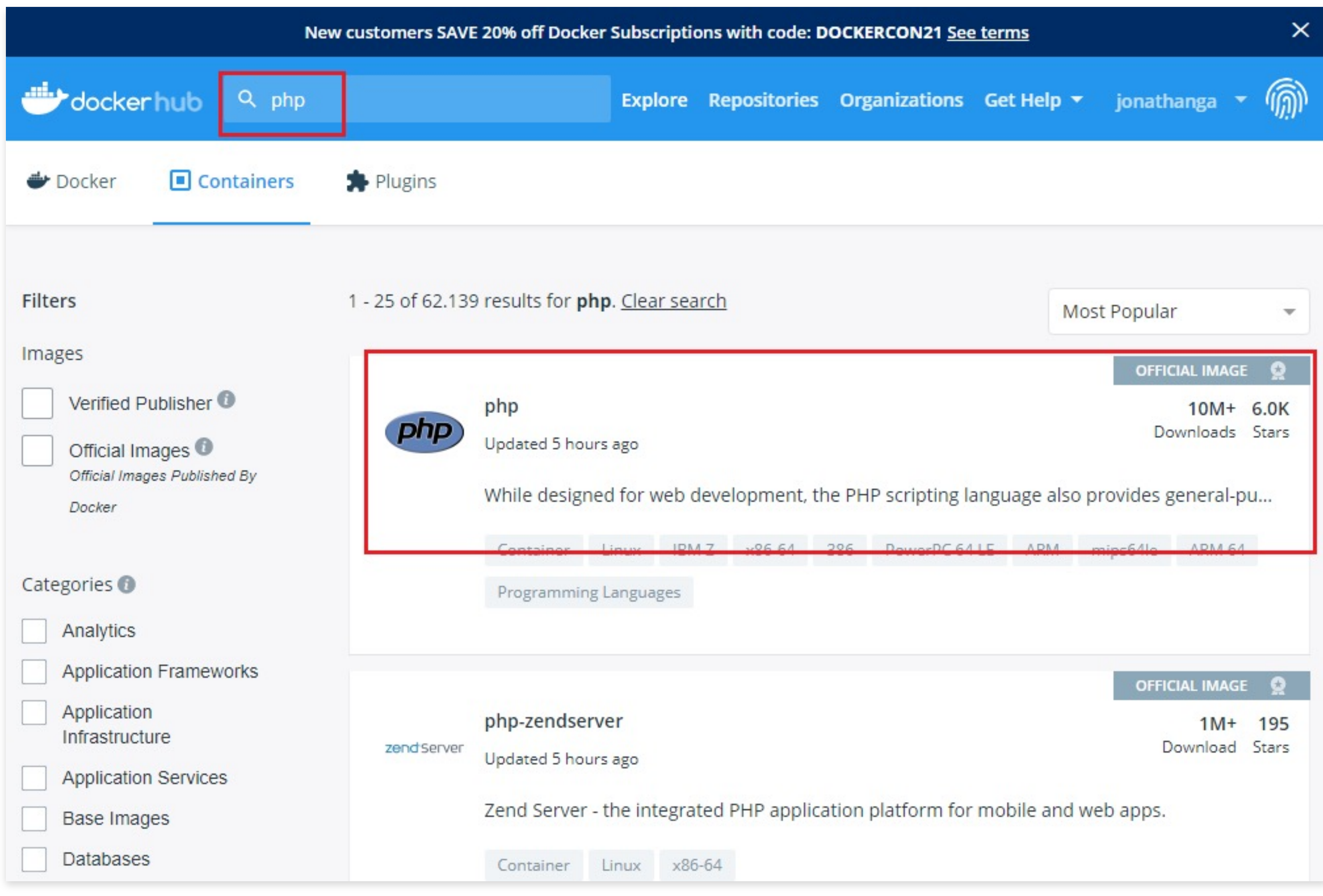
Copyright (c) 2000, 2021, Oracle and/or its affiliates.

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>
```

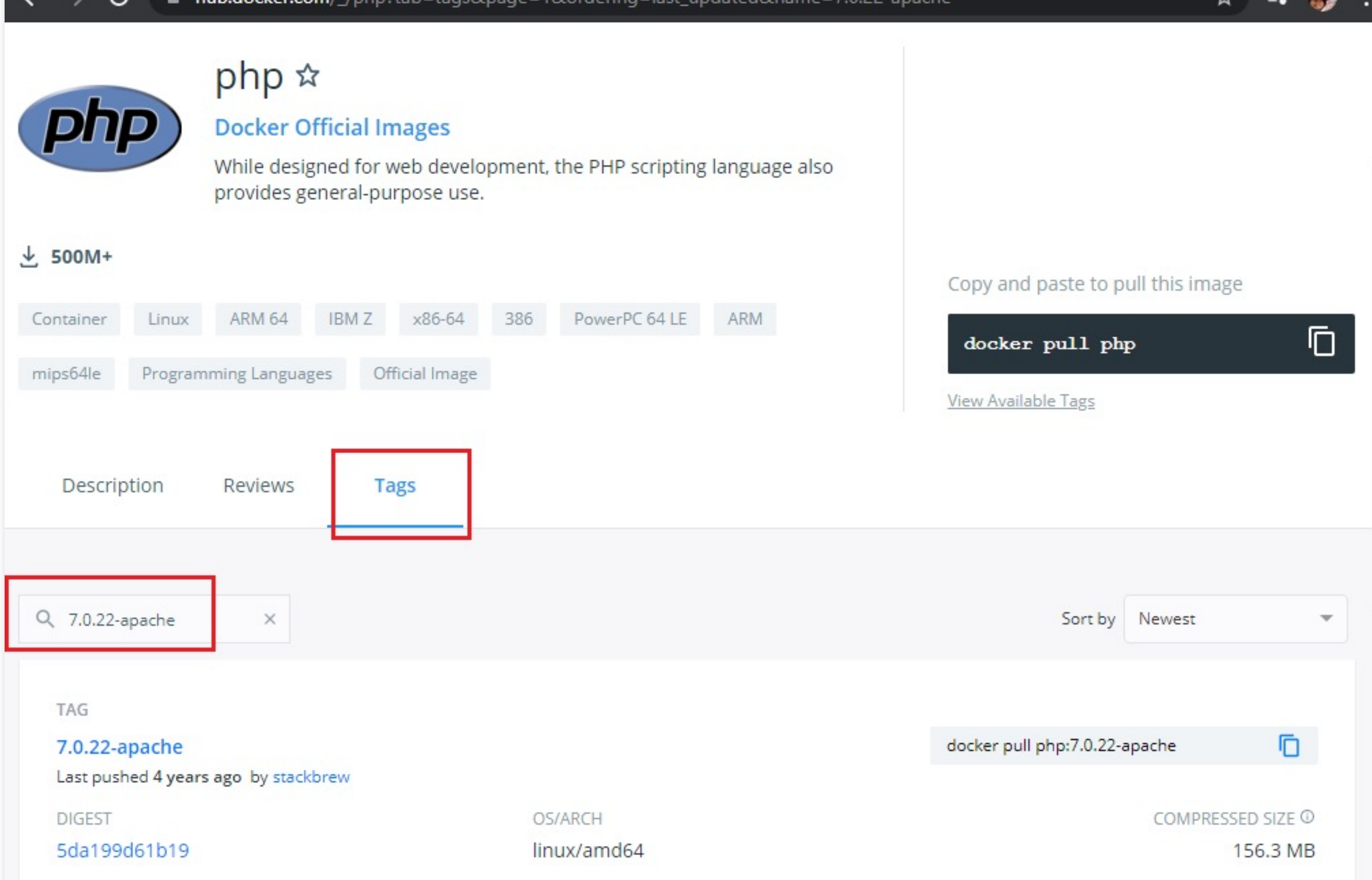
El proceso de construcción de la imagen de acuerdo con las sentencias definidas en el dockerfile se realiza de forma lineal de arriba hacia abajo, docker dividirá esta construcción en pasos de acuerdo al número de comandos definidos en el archivo Dockerfile, el tamaño final de la imagen, la cantidad de comandos a ejecutar en la imagen construida y la velocidad de conexión a internet definen el tiempo que se puede demorar el sistema en la construcción de la imagen.



Así como se pueden crear imágenes propias iniciando desde lo básico e instalando de acuerdo a las necesidades, es posible encontrar imágenes públicas con todos los elementos ya pre-configurados listos para la creación de nuevos contenedores.

Para el siguiente ejemplo se ejecutará un servidor web que permita archivos de extensión php usando un contenedor a partir de una imagen publicada en Docker Hub.

Para esto debe ingresar a Docker hub, buscar php y verificar los detalles de la imagen oficial.



Se puede verificar en la pestaña de tags todas las versiones disponibles de imágenes de php, en este caso requerimos una versión específica de php y el servidor apache.

```
Windows PowerShell
PS C:\Users\ADMIN\web> docker pull php:7.0.22-apache
7.0.22-apache: Pulling from library/php
a174f08f5922: Pull complete
a1e755572441: Pull complete
6ab4772a86ad: Pull complete
55e3b08042ca: Pull complete
88792c88e1bc: Pull complete
1d8a48c4fe59: Pull complete
0c30cf90e233: Pull complete
37ec3cd3c9fb: Pull complete
90b543ec2d81: Pull complete
e31603a3190b: Pull complete
aa853f8165fa: Pull complete
c728020d0d41: Pull complete
a7a1c939c9f4: Pull complete
Digest: sha256:5da199d61b19df8414c96327f3f869d9c3f1eb87140091209ca7614cf653
Status: Downloaded newer image for php:7.0.22-apache
docker.io/library/php:7.0.22-apache
PS C:\Users\ADMIN\web>
```

```
Windows PowerShell
PS C:\Users\ADMIN\web> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mysqlimage           latest              3c7403871170       58 minutes ago     689MB
jonathanga/ubuntu_mysql latest             6ddc7f0c2639       3 hours ago        699MB
jonathanga/docker10tutorial latest            c143c547c834       7 hours ago        28MB
docker10tutorial    latest            c143c547c834       7 hours ago        28MB
alpine/git           latest            b0f176fa3f8d       2 weeks ago        25.1MB
ubuntu               latest            7d8da2d69a15       7 weeks ago        72.7MB
php                  7.0.22-apache     23ed8a9a03f2       3 years ago        389MB
```

Una vez seleccionada la imagen y la versión procedemos a descargar la imagen usando el siguiente comando: **docker pull imagen:versión**, para el ejemplo la imagen se llama **php** y la versión es la **7.0.22-apache**. Además, para asegurarse de hacer las pruebas, debe ubicar la terminal de ejecución de docker sobre un directorio donde contendrá los archivos a publicar en el servidor que se va a montar desde el contenedor.

```
Windows PowerShell
PS C:\Users\ADMIN\web> docker run -p 8081:80 -v C:\Users\ADMIN\web\:/var/www/html -d php:7.0.22-apache
82e19eba49e9  php:7.0.22-apache
PS C:\Users\ADMIN\web>
```

```
Windows PowerShell
PS C:\Users\ADMIN\web> docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED          STATUS          PORTS          NAMES
82e19eba49e9   php:7.0.22-apache  "docker-php-entrypoi..." About a minute ago Up 55 seconds   0.0.0.0:8081->80/tcp, :::8081->80/tcp   elated
```

A continuación, se creará el contenedor desde la máquina teniendo en cuenta que en el contenedor el servidor de php estará escuchando solicitudes desde el puerto 80 y debo indicar qué puerto en la máquina local será asignado para realizar el proceso de escucha en la ejecución del contenedor, para esto se usa el comando de ejecución del contenedor con la especificación de los puertos de la siguiente forma:

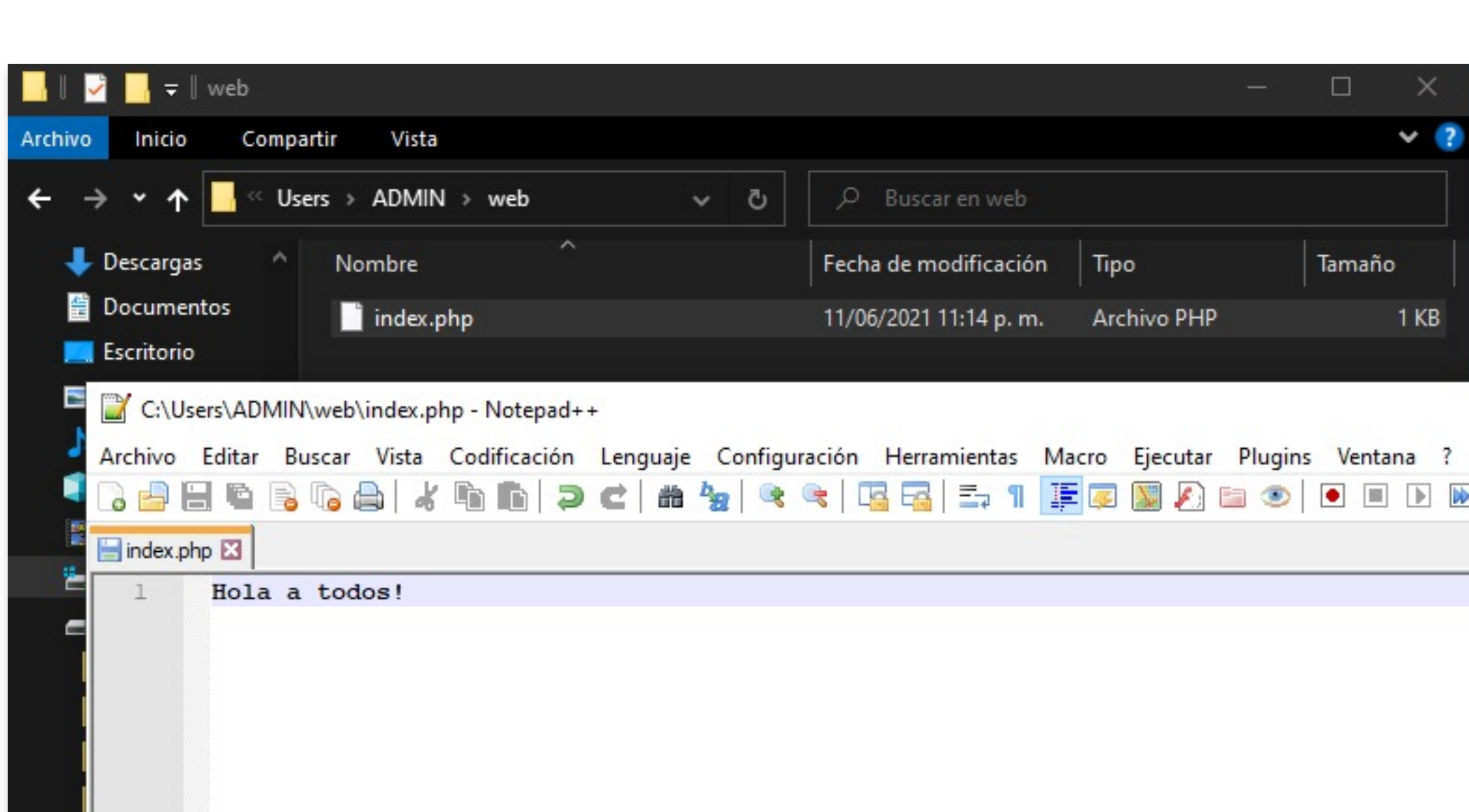
docker run -p puertoLocal:puertoContenedor

Donde **puertoLocal** corresponde al puerto donde escuchará el servicio en mi máquina local y **puertoContenedor** corresponde al puerto con el que se equipará en el contenedor.

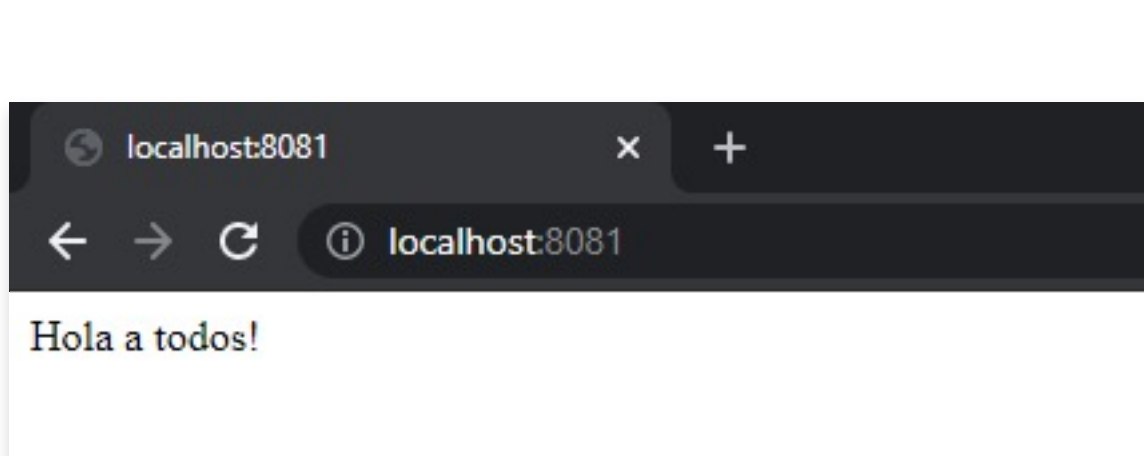
Como este contenedor va a ser utilizado para ir probando cambios en los archivos php que se van generando en el equipo local, es necesario indicar a docker para que haga un mapeo del directorio en la máquina local con el directorio donde se deben alojar los archivos que deben ser ejecutados por apache en el contenedor con lo cual el comando de ejecución puede tomar la siguiente forma:

docker run -p puertoLocal:puertoContenedor -v directorioMaquinaLocal:DirectorioEnContenedor Imagen

Donde **-v** es la directriz para hacer mapeo y **directorioMaquinaLocal** corresponde a una ruta donde se aloja mi directorio de desarrollo local y **DirectorioEnContenedor** representa el directorio donde será mapeado toda la información en el contenedor. **Imagen** corresponde al nombre de la imagen previamente descargada. También se puede agregar la bandera **-d** para hacer que el contenedor que se ejecute lo haga en segundo plano.



En la siguiente imagen se puede observar que la terminal de comandos está ubicada sobre un directorio llamado **web** el cual corresponde con la ruta que se indica en el mapeo **C:\Users\ADMIN\web** y el directorio mapeado del lado del contenedor corresponde a **/var/www/html** el cual es el sitio que utilizará el servidor Apache para la publicación de aplicaciones web.



Con este mapeo se logra que todos los cambios que haga en el directorio de la máquina local se verá publicado en el servidor de apache que está siendo ejecutado por el contenedor y que también está mapeado al puerto local 8081 en mi máquina local.