Laboratorio.

Curso: ST0263
Título: Contenedores - Docker
Objetivo: Desarrollar habilidades en en el proceso de creación, despliegue y gestión de aplicaciones utilizando contenedores, particularmente docker.

Duración: 45 mins.

1. Introducción

Los contenedores puede ser considerados unos de los desarrollos tecnológicos mas interesantes en la industria de TI en los últimos años. En la actualidad, la mayoría de las organizaciones esta migrando sus aplicaciones apalancándose en esta tecnología.

Este laboratorio nos permitirá desarrollar las habilidades básicas que nos permitan el despliegue de contenedores, particularmente Docker.

2. Recursos

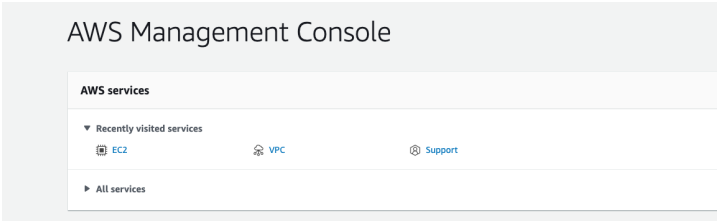
Para el desarrollo de este laboratorio se requiere una cuenta en AWS que le permita desplegar una instancia EC2 para instalar sistema operativo Linux Ubuntu así como

3. Desarrollo

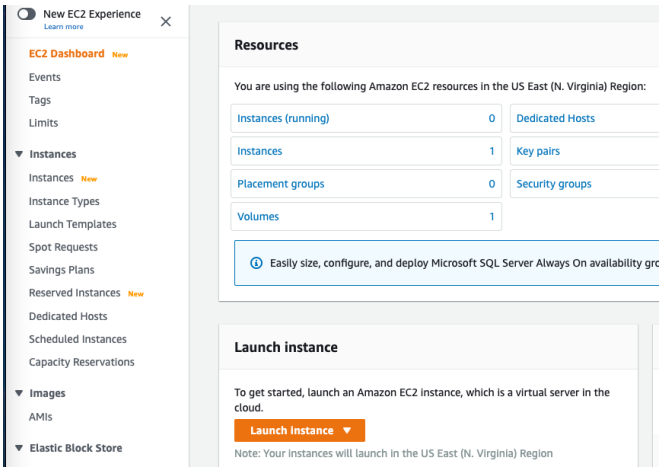
A continuación se presenta el conjunto de pasos para poder desplegar aplicaciones utilizando docker.

3.1. Instancie una máquina EC2 en la consola de AWS.

Para esto en la consola de AWS por favor seleccione el servicio de EC2.



Una vez este aquí, seleccione la opción de “Launch Instances”

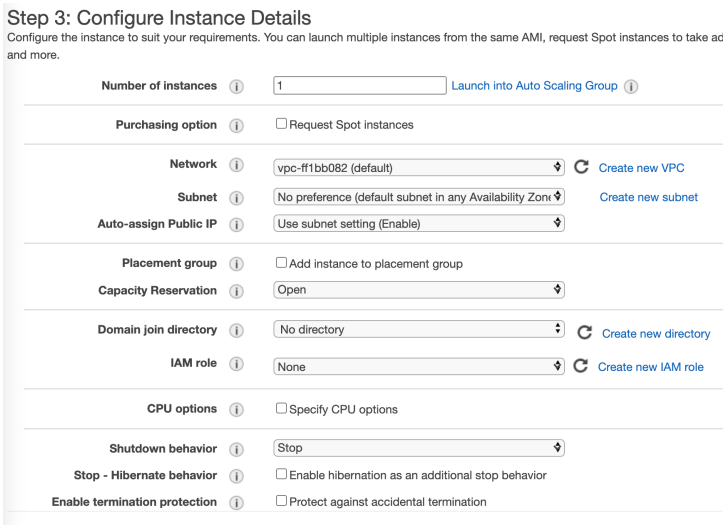


Seleccione el tipo de Amazon Machine Image y proceda a la configuración de los parámetros a través de los siguientes siete (7) pasos.

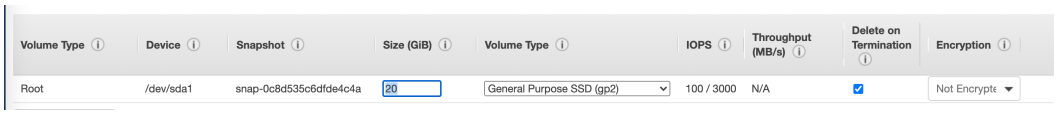
- **Step 1.** Para efectos de este laboratorio utilizaremos una máquina con OS Ubuntu.

- **Step 2.** Esta esta disponible para “free tier.

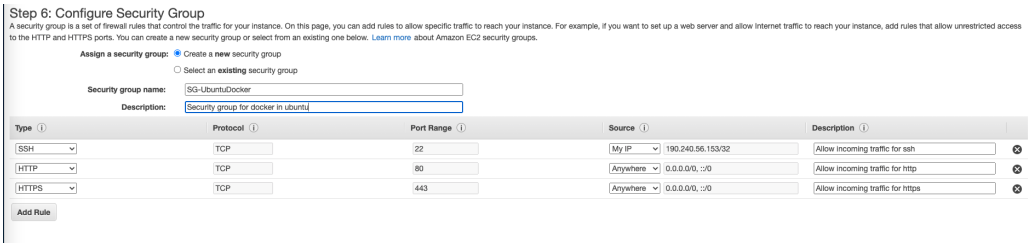
- **Step 3.** Configure los detalles de la instancia. Aquí vamos a dejar los parámetros por defecto. En caso tal desee ubicar la instancia en otra VPC, modifique el valor para el párametro **Network**. Igualmente, dado que necesitamos acceder la máquina, habilite que se le asigne una dirección IP pública. Esto con el fin de poder acceder via ssh a esta y poder configurarla.



- **Step 4.** Configure los parámetros relacionados con el almacenamiento de la EC2. Por favor establezca un valor de 20 GB. Luego de click en **Next: Add Tags.**



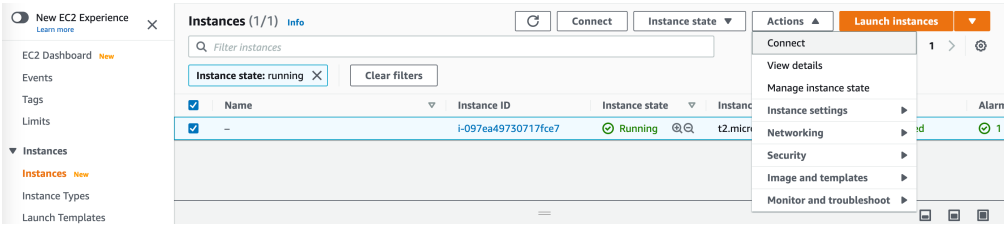
- **Step 5.** Adicione un tag a la instancia que estamos configurando. Para este caso establezca los siguientes valores y luego de click en **Next: Configure Security Group**.
    - **Key:** Server
    - **Value:** Ubuntu 20
- **Step 6.** Configure security group. Por favor defina un nuevo security group para la instancia. Luego de click **Review and Launch**. Tenga presente que debe habilitar el tráfico entrante para SSH (22), HTTP (80) y HTTPS(443).



- **Step 7.** En este punto revise los parámetros de configuración y lance la instancia. En este punto le va a solicitar que cree o seleccione una key existente.

3.2. Sesión con la EC2 instanciada.

- En el menú del servicio EC2, seleccione la instancia con la cual desea establecer la conexión.



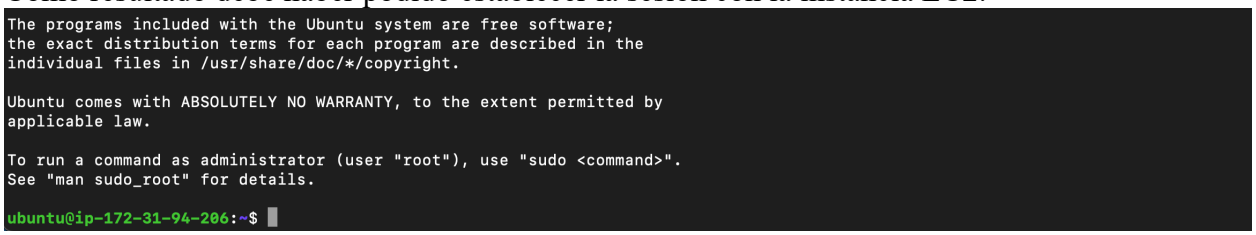
- Cuando seleccione la opción de conectarse, le aparecerá un cuadro parecido. Tenga presente que debe cambiar los permisos de el archivo key (.pem).

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ST0263-TET.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 ST0263-TET.pem
4. Connect to your instance using its Public DNS:
   ec2-3-88-84-127.compute-1.amazonaws.com

Example:

ssh -i "ST0263-TET.pem" ubuntu@ec2-3-88-84-127.compute-1.amazonaws.com

- En una terminal de consola desde su máquina, inicie una sesión ssh contra el servidor que configuro tal como lo indica la sesión de "example" en la sesión anterior.
- Como resultado debe haber podido establecer la sesión con la instancia EC2.



3.3. Instalación de Docker.

Una vez tenemos ya nuestra máquina Linux instalada vamos a proceder a configurar Docker en ésta.

- Antes que cualquier cosa, procedamos a verificar que la máquina no tenga ninguna versión de Docker instalada, para esto, por favor, ejecute el siguiente comando:

```
$ sudo apt-get remove docker docker-engine docker.io
```

- Actualice el index del páquete apt-get:

```
$ sudo apt-get update
```

- Instale los paquetes necesarios para permitir a apt el utilizar repositorios sobre https:

```
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

- Se hace necesario agregar la clase oficial GPG de Docker:

```
$curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor
        -o /usr/share/keyrings/docker-archive-keyring.gpg
```

- Aplique el siguiente comando para establecer el repositorio "stable".

```
$ echo \
"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
```

- Ahora procedamos a instalar el docker engine.

```
$ sudo apt-get update
$ sudo apt-get install docker-ce
```

- Verifique la instalación de docker con el siguiente comando:

```
$ sudo docker container run hello-world
```

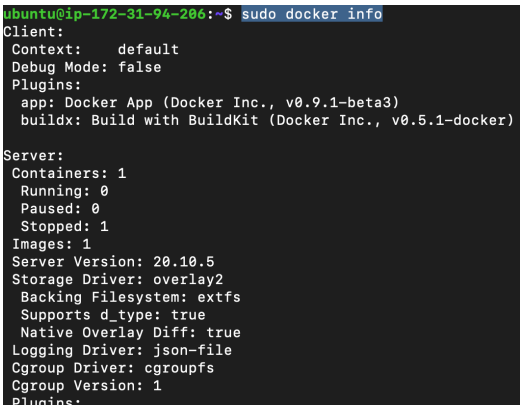
```
ubuntu@ip-172-31-94-206:~$ sudo docker container run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b8dfde127a29: Pull complete
Digest: sha256:308866a43596e83578c7dfa15e27a73011bdd402185a84c5cd7f32a88b501a24
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

- Para obtener la información del docker instalado, digite el siguiente comando:

```
$ sudo docker info
```

Podrá observar una salida parecida a esta:



3.4. Iniciando el servicio

- Para inicializar el servicio, por favor digite el siguiente comando

```
$ sudo systemctl start docker
```

- Para habilitar el inicio del servicio de Docker al momento del inicio del sistema:

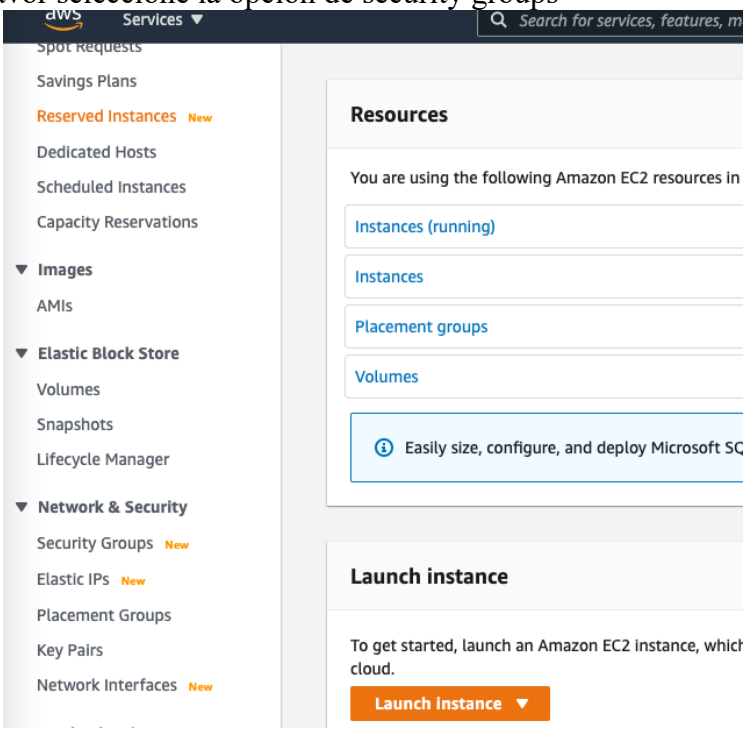
```
$ sudo systemctl enable docker
```

3.5. Desplegando y gestionando imágenes así como contenedores.

En esta sección vamos a desplegar un servidor web apache utilizando contenedores. Todo esto lo vamos a realizar a partir de un archvo que se denomina dockerfile. Al final vamos a poder acceder a la siguiente URL:

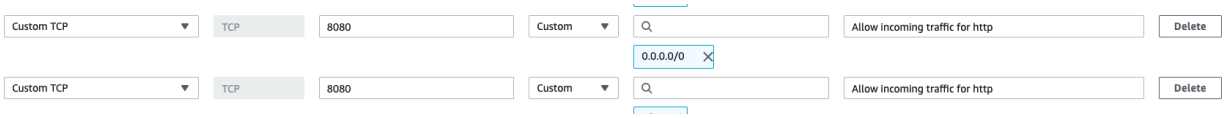
http://PublicIPAddress:8080/test.html

Como pueden observar estamos utilizando en este ejercicio el puerto 8080, por esta razón se hace necesario que usted agregue en el security group de su máquina los permisos para que el tráfico de entrada a este puerto se permita. Para esto en el menú del servicio de EC2 de AWS, por favor seleccione la opción de security groups



En esta vista seleccione el security group que definio para su instancia (p.ej., SG-UbuntuDocker) y en la pestaña de abajo seleccione la opción de **Inbound rules y de click en Edit Inbound Rules**

Ahora agregue la regla para permitir el ingreso de tráfico http por el puerto 8080 y guarde los cambios.



Nota: Recuerde que no se hace necesario reiniciar la instancia una vez modifique las directivas del security group.

Ahora, se va a proceder a crear un directorio en el home de su sistema de la siguiente forma:

```
$ sudo mkdir mywebserver
$cd mywebserver
```

Una vez en este directorio vamos a crear un archivo test.html con el siguiente código ejemplo:

```
$ nano test.html
```

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Docker Apache</title>
</head>
<body>
  <h2>Hello from Apache Web Server container...</h2>
</body>
</html>
```

Ya que tenemos una página personalizada para el ejemplo, vamos a crear un archivo Dockerfile para la instalación del servidor web apache.

```
FROM httpd:2.4
COPY ./test.html /usr/local/apache2/htdocs/
```

Recuerde que la directiva **FROM** nos indica la imagen base que estamos tomando, en este caso la del servidor apache versión 2.4. El comando **COPY** nos permite copiar nuestro archivo test.html en el directorio de htdocs de apache como objeto/recurso web a ser accedido.

Ahora vamos a proceder a construir nuestra imagen a partir de este archivo Dockerfile. Para esto utilizamos el siguiente comando:

```
$ sudo docker build -t my-apache2 .
```

```
ubuntu@ip-172-31-28-223:~/my_website$ sudo docker build -t my-apache2 .
Sending build context to Docker daemon  3.072kB
Step 1/2 : FROM httpd:2.4
 ---> ae15ff2bdcb4
Step 2/2 : COPY ./test.html /usr/local/apache2/htdocs/
 ---> Using cache
 ---> cf5942ffa94b
Successfully built cf5942ffa94b
Successfully tagged my-apache2:latest
```

Una vez creada, vamos a proceder a ejecutar nuestro contenedor con el servidor web apache.

```
$ docker run -dit --name my-running-app -p 8080:80 my-apache2
```

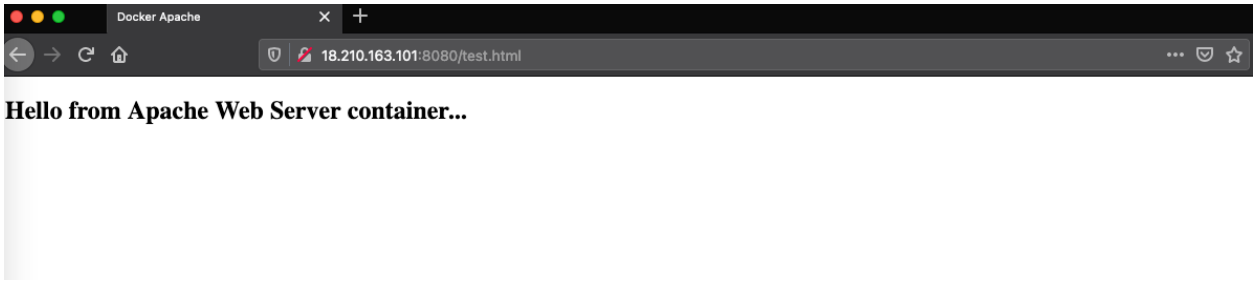
Este comando permite ejecutar el contenedor en backgroud (-d). El nombre va a ser **my-running-app (--name)** el cual se ejecuta desde la instancia **my-apache2**. El servidor web estará escuchando peticiones en el puerto (-p) 8080 de la máquina el cual se redirige al puerto 80 del contenedor.

Con el siguiente comando podemos verificar el funcionamiento de nuestro contenedor:

```
$ sudo docker container ps
```

```
ubuntu@ip-172-31-28-223:~/my_website$ sudo docker container ps
CONTAINER ID   IMAGE         COMMAND              CREATED          STATUS          PORTS                    NAMES
f1e7f9bc69e6   my-apache2    "httpd-foreground"   41 minutes ago   Up 41 minutes   0.0.0.0:8080->80/tcp     my-running-app
```

Finalmente, para verificar el funcionamiento de nuestro servidor, vamos a ejecutar un browser con la dirección pública de nuestro servidor web en el puerto 8080 accediendo el recurso test.html. El resultado debe ser similar al observado en la figura.



3.6. Desplegando aplicaciones con múltiples contenedores con Docker Compose.

Para poder ejecutar aplicaciones con múltiples contenedores, una forma de hacerlo es utilizar la herramienta docker compose. Para esto se hace necesario instalar la herramienta. Recuerde por favor tener instalado antes el docker engine.

```
$ sudo curl -L
"https://github.com/docker/compose/releases/download/1.28.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

```
ubuntu@ip-172-31-94-206:~$ sudo curl -L "https://github.com/docker/compose/releases/download/1.28.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   633  100   633    0     0   6525      0 --:--:-- --:--:-- --:--:--  6525
100 11.6M  100 11.6M    0     0  45.4M      0 --:--:-- --:--:-- --:--:-- 45.4M
```

Ahora proceda a cambiar los permisos al ejecutable con el siguiente comando:

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

Verifique la versión a través del comando:

```
$docker-compose --version
```

```
ubuntu@ip-172-31-94-206:~$ docker-compose --version
docker-compose version 1.28.5, build c4eb3a1f
```

A partir de este momento se puede utilizar docker compose. De esta forma vamos a desplegar un CMS (wordpress) el cual requiere dos contenedores: uno para el CMS como tal y el otro para la persistencia de datos.

Para poder desplegar el wordpress vamos a crear un directorio para el proyecto

```
$ sudo mkdir dk_wordpress
    $cd dk_wordpress/
```

Ahora, en este directorio vamos escribir un archivo el cual tiene la estructura definida a continuación:

```
$ nano docker-compose.yml
```

docker-compose.yml

```
version: '3.1'

services:

  wordpress:
    image: wordpress
    restart: always
    ports:
      - 80:80
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: exampleuser
      WORDPRESS_DB_PASSWORD: examplepass
      WORDPRESS_DB_NAME: exampledb
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: exampledb
      MYSQL_USER: exampleuser
      MYSQL_PASSWORD: examplepass
```

```
      MYSQL_RANDOM_ROOT_PASSWORD: '1'
    volumes:
      - db:/var/lib/mysql

volumes:
  wordpress:
  db:
```

Como se puede observar, se muestra un archivo simple Compose que define una aplicación Multiservicio. Como se menciono anteriormente, el nombre por defecto para un archico compose es *"docker-compose.yml"*. En este caso el archivo define la sección para el wordpress así como la sección para la bases de datos. Se puede observar que el archivo tiene tres (3) secciones:

- *version*: Esta sección es de carácter obligatoria y siempre debe ser la primera línea del archivo Compose. Indica la versión de formato del archivo compose y no se debe confundir con la versión del docker engine así como del docker compose.
- *services*: En esta sección es donde se definen los diferentes servicios que componen la aplicación que se desea desplegar. Para este caso, el wordpress y la base de datos. De esta forma compose va a desplegar cada uno de estos servicios en un contenedor propio.
- *Volumes:* Esta sección le indica al docker donde se le indica al docker para crear nuevos volúmenes

Ahora vamos a proceder a desplegar los contenedores a través del siguiente comando:

```
$ sudo docker-compose up -d
```

El flag -d indica que el proceso se estará ejecutando en background.

```
ubuntu@ip-172-31-28-223:~$ sudo docker-compose up -d
Creating network "ubuntu_default" with the default driver
Creating ubuntu_wordpress_1 ... done
Creating ubuntu_db_1        ... done
```

Nota: Normalmente el archivo se denomina docker-compose.yml o docker-compose.yaml. Ambas extensiones las puede utilizar. Tenga presente que usted también puede nombrar el archivo de otra forma. En ese caso debe ejecutar el comando docker-compose con la bandera (flag) -f con el fin de especificar el nombre del archivo.

En este punto vamos a verificar las imágenes que se encuentran en la máquina actualmente así como los contenedores que se encuentran en ejecución. Para observar la información de loas imágenes digite el comando:

```
$ sudo docker-compose ps
```

```
[ubuntu@ip-172-31-28-223:~$ sudo docker-compose ps
       Name                     Command               State          Ports
-------------------------------------------------------------------------------
ubuntu_db_1          docker-entrypoint.sh mysqld      Up      3306/tcp, 33060/tcp
ubuntu_wordpress_1   docker-entrypoint.sh apach ...   Up      0.0.0.0:80->80/tcp
```

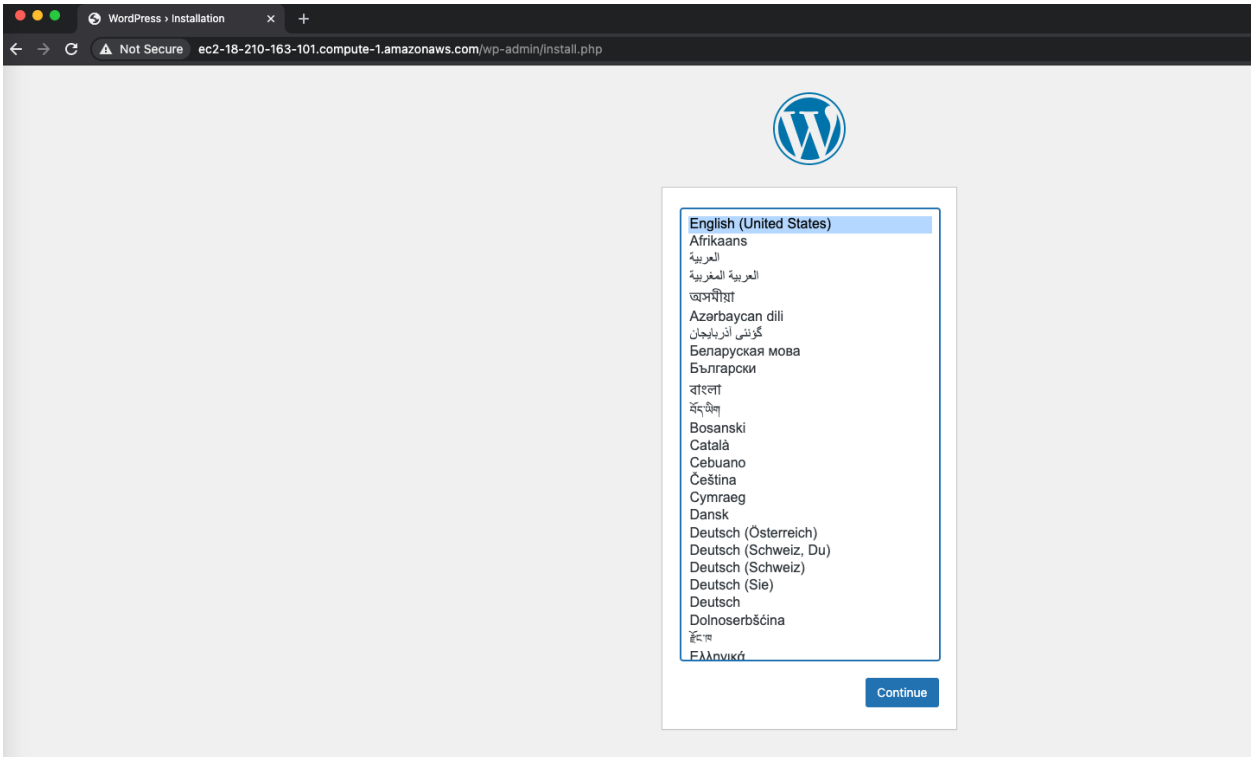
Como se puede observar se ven los dos contenedores que están en ejecución: bases de datos así como CMS. Ambos en estatus arriba (up) y con la información de los puertos que están abiertos para la operación. Esta misma información puede ser verificada con el siguiente comando:

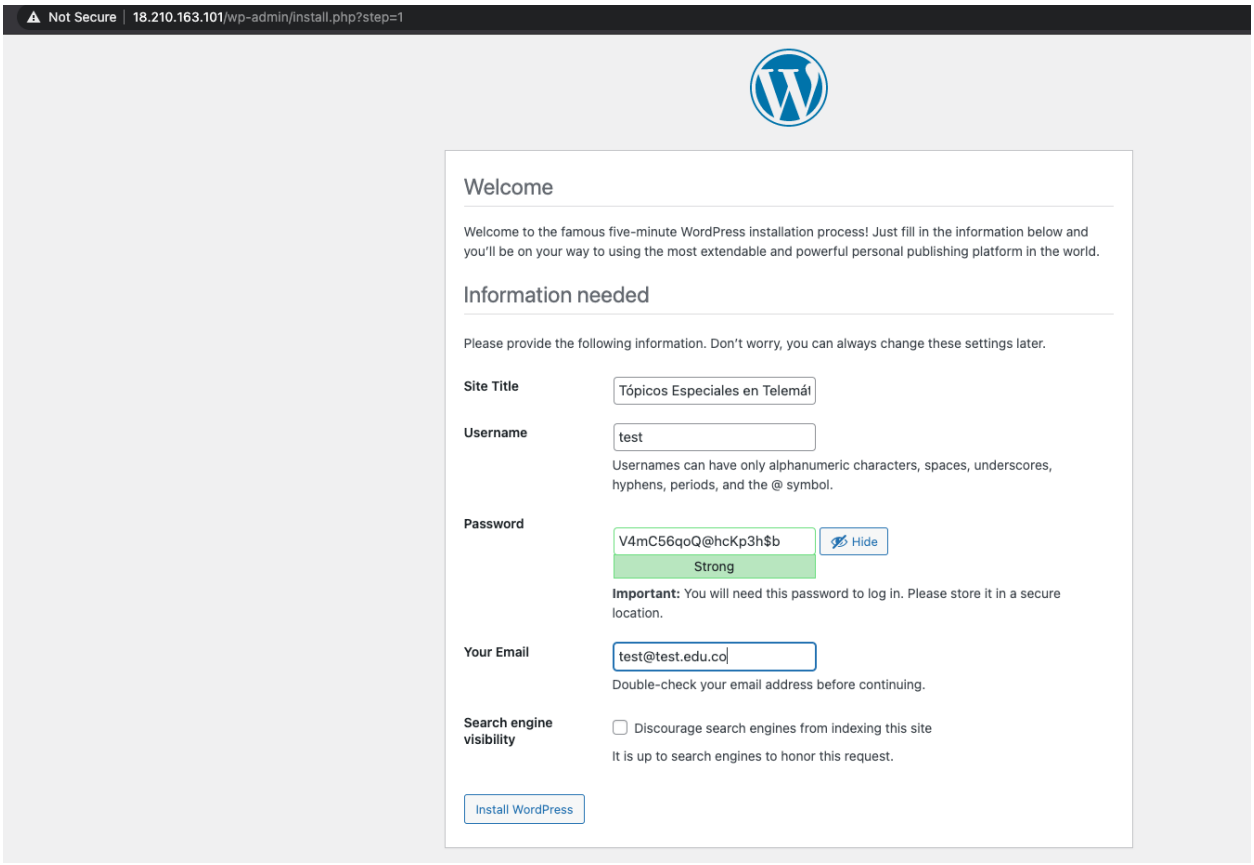
$ sudo docker ps

```
ubuntu@ip-172-31-28-223:~$ sudo docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED          STATUS          PORTS                   NAMES
d9f3a1c9db7c   mysql:5.7   "docker-entrypoint.s…"   17 minutes ago   Up 17 minutes   3306/tcp, 33060/tcp     ubuntu_db_1
1c4e519b1122   wordpress   "docker-entrypoint.s…"   17 minutes ago   Up 17 minutes   0.0.0.0:80->80/tcp      ubuntu_wordpress_1
```

Como puede observar, este comando nos proporciona información similar y un poco mas detallada sobre los contenedores que están en ejecución en la instancia.

Finalmente, para verificar e interactuar con el servicio/aplicación desplegada, inicie una nueva ventana/pestaña de un navegador (browser) y digite en el campo URL bien sea el nombre de la máquina en AWS (Public IPv4 DNs) o dirección IP pública (Public IPv4 address). A continuación debe observar la pantalla de bienvenida para la configuración del sitio Wordpress con lo cual puede proceder a la configuración del mismo.




Not Secure | 18.210.163.101/wp-admin/install.php?step=1
Welcome
Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.
Information needed
Please provide the following information. Don't worry, you can always change these settings later.
Site Title
Tópicos Especiales en Telemát
Username
test
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password
V4mC56qoQ@hcKp3h$b
Hide
Strong
Important: You will need this password to log in. Please store it in a secure location.
Your Email
test@test.edu.co
Double-check your email address before continuing.
Search engine visibility
Discourage search engines from indexing this site
It is up to search engines to honor this request.
Install WordPress