Guion de exposición: Introducción a Docker.

Buenas tardes compañeros, profesor, como parte de nuestra exposición sobre contenedores les traemos un guía y primeros pasos utilizando la herramienta Docker, esta es la herramienta más comúnmente utilizada en el manejo y orquestación de contenedores como muchos sabrán.

Paso 1.

Lo primero que haremos será instalar docker desktop ya que nos encontramos en Windows sin embargo utilizaremos una distribución Ubuntu que tengo instalada junto con mi sistema operativo para utilizar los mismos comandos que usaríamos en Windows.

Solamente seguimos el asistente de instalación y tendríamos todo listo.

Paso 2.

Verificamos la versión y correcta instalación de docker ejecutando el comando:

docker –version

Después verificamos que todo este corriendo de manera correcta con:

docker info

Paso 3.

Vamos a correr el contenedor de prueba hello world con el comando:

docker run hello-world

Docker verifica si existe la imagen localmente, si no la baja, nos arroja un resultado correcto y confirma que todo está funcionando.

Paso 4.

Para visualizar los contenedores que tenemos activos actualmente, corremos el comando:

docker ps

Actualmente no tenemos ningún contenedor activo, podemos ver los contenedores inactivos ejecutando:

docker ps -a

Aquí podemos ver que el contenedor hello-world esta inactivo.

Paso 5. Remover un contenedor.

Tomamos el id que visualizamos con el comando anterior y ejecutamos la siguiente línea:

docker rm <id del contenedor>

En este caso borraremos el contenedor hello-world con su id.

Volvemos a ejecutar:

docker ps -a

Y vemos que ya no se encuentra en la lista.

Paso 7.

Ahora vamos a crear un contenedor, crearemos un contenedor con distribución fedora para esto utilizamos el comando:

docker run -it –name fedora_base fedora

-it Indica a que el contenedor será interactivo por medio de consola, --name indica el nombre del contenedor, fedora indica la imagen a partir de la cual se creará el contenedor en este caso fedora.

Docker reconocerá que no existe el contenedor y hará pull de la imagen.

Paso 8.

Ahora mismo nos encontramos dentro del contenedor fedora_base, salimos utilizando:

exit

Verificamos ahora los contenedores existentes con:

docker ps -a

Vemos que ahora se encuentra entre los valores nuestro contenedor nuevo.

Paso 9.

Iniciamos el contenedor con el comando:

docker start fedora_base

Y entramos con el comando:

docker attach fedora_base

Paso 10.

Vamos a actualizar nuestra distribución de fedora, actualizamos el instalador de paquetes de fedora con el comando:

dnf update -y

E instalamos Python con el comando:

dnf install -y python3

Paso 11.

Verificamos la instalación de Python con el comando:

python3 –version

Con lo que nuestro contenedor de fedora ya debería poder ejecutar códigos, creamos un script de Python:

echo 'print("Hola desde Docker Fedora!")' > script.py

Ejecutamos con:

python3 script.py

Y obtenemos la salida esperada.

Paso 10.

Ahora, no queremos tener que hacer el update e instalación de Python cada que queremos un nuevo contenedor de fedora por lo que creamos la imagen con Python instalado, salimos del contenedor y ejecutamos:

docker commit fedora_base fedora_python

Revisamos las imágenes existentes con el comando:

docker images

Vemos que ahora contamos con la imagen fedora_python

Paso 11.

Creamos entonces un nuevo contenedor con la imagen fedora_python:

docker run -it –name python_instalado fedora_python

Revisamos que python siga instalado con python3 –version y vemos que este nuevo contenedor ya puede ejecutar python.

python3 script.py

El script también se guardo puesto que lo creamos antes de hacer commit a la imagen.

Paso 13.

Ahora vamos a mostrar distintas distribuciones en contenedores, limpiamos la consola y vamos a crear distintos contenedores con distribuciones distintas:

docker run -it –name u ubuntu

docker run -it –name a alpine

docker run -it –name d debian

Dentro de cada una utilizamos el comando:

cat /etc/os-release

Para asegurarnos que estamos en el SO especificado.

Paso 14.

Ahora, utilizando:

docker images

podemos comparar los tamaños de las imágenes de cada imagen equivalente a cada sistema operativo.

Paso 15.

También podemos monitorear el consumo de recursos de todos los contenedores para eso los iniciaremos todos Ver el consumo de recursos de los contenedores iniciamos cada uno con:

docker start <nombre contenedor>

Y utilizamos el comando:

docker stats

esto nos permite ver el consumo de RAM y CPU de cada una de las distribuciones.

Paso 16.

Finalmente detenemos los contenedores, visualizamos sus ids con:

docker ps

y los detenemos con la línea:

docker stop <id del contenedor>

Eso fue todo por nuestra parte, gracias por su atención y esperamos la presentación les sea útil, gracias.