Actividad 2

Fundamentos de docker

Mikroways

Construcción de imágenes

1. Generar una imagen basada en **httpd** (apache) mediante la definición de un Dockerfile. Tendrá que modificar el mensaje **It works!** por el mensaje **¡ Hola Mundo !**

> Puede leer la documentación de la imagen para poder cambiar el contenido del `DocumentRoot`. El que ya viene en la configuración es **/usr/local/apache2/htdocs** .

- La imagen debe llamarse **mikroways/hola-mundo-httpd**
- Verificar que la imagen se creó correctamente listando las imágenes guardadas en su host.
- Verifique el funcionamiento instanciando la imagen y accediendo al servicio usando curl o un navegador.

2. Push de imagen: cree una imagen a su gusto y luego realice un push en docker hub y gitlab. La idea es disponibilizar la nueva imagen públicamente indicando el repositorio y comandos para que podamos descargarlas.

- Si no tiene una cuenta en Docker Hub, registrarse. Una vez creada la cuenta, iniciar sesión y crear un nuevo repositorio en el cual subirá su imagen docker. Para ello deberá iniciar sesión desde la cli de docker usando: `docker login`.
- Si no tienen una cuenta en GitLab, registrarse. Una vez creada la cuenta, iniciar sesión y crear un nuevo repositorio en el cual subirá su imagen Docker. Al crear un nuevo repositorio, en el panel de la izquierda, se tiene un menú **Packages & Registries > Container Registry**. Ingresando a este item se indican las acciones necesarias para subir la imagen a la registry de GitLab.

3. Crear una imagen en donde al instanciar el contenedor debe imprimir **"Hola Mundo"**. Para esto deberá utilizar un ENTRYPOINT en el Dockerfile y **CMD** para parametrizar el mensaje que por defecto será **Hola mundo**.
4. Cree un Dockerfile para generar una imagen llamada **mikroways/file-creator**.
    - Utilice como base la imagen alpine:latest.
    - Investigue el comando WORKDIR dentro de un Dockerfile.
    - Dentro del Dockerfile cambie el WORKDIR a `/tmp`.
    - Utilizando un ENTRYPOINT Y CMD, al invocar el contenedor debe crear un archivo de 10M o del espacio recibido como argumento.

> El comando `dd` copia archivos. El siguiente ejemplo muestra como crear un archivo de 1MB lleno de ceros.

```
dd if=/dev/zero bs=1M count=1 of=created-file
```

Ejemplo de uso de la imagen mikroways/file-creator:

```
docker run mikroways/file-creator # crea un archivo de 10MB en /tmp
docker run mikroways/file-creator 100M # creará un archivo de 100 MB en /tmp
```

Puede verificar el tamaño del contenedor resultante usando:

```
docker ps -asn 2
```

Indique qué interpreta en la salida del tamaño del contenedor lanzado.

5. Modifique la imagen anterior para que se fuerce a correr con un usuario diferente a ROOT.
6. Utilizando el concepto de multistage builds, genere una imagen de docker que contenga el sitio web de mikroways. Los fuentes del sitio, están disponibles en un repositorio en github. Para poder construir el sitio, es necesario partir de una imagen de ruby versión 2.7, clonar el repositorio usando git, instalar las dependencias y luego generar el sitio estático usando jekyll. A continuación mostramos la secuencia de comandos para construir el sitio y obtener en la carpeta `_site/` el sitio estático. Ese contenido deberá incluirlo en un web server de contenidos estáticos como ser **nginx**.

```
git clone https://github.com/Mikroways/mikroways.net.git /tmp/site
cd /tmp/site
bundle install
jekyll build
```

7. Usando el mismo proceso antes mencionado, es posible generar un sitio con su currículum vitae. Si ha tomado con Mikroways el curso de *Workflow Contínuo de Desarrollo, Testing y Producción*, seguramente haya desarrollado utilizando despliegue continuo su CV. En caso que no lo haya realizado, puede tomar como referencia el repositorio https://github.com/chrodriguez/chrodriguez.github.io.

   > Es importante destacar que con el comando `npm run resume export index.html` Es el index.html resultante del comando anterior el único archivo estático que debe utilizarse en un web server para servir este contenido.

8. Analice cómo funciona `.dockerignore`. Analice el contexto de un repositorio git que en su raíz tenga un Dockerfile. ¿Qué sucede con el directorio `.git` si se encuentra dentro del contexto del docker build?
9. Cree un directorio en su filesystem `/tmp/docker-image-size` e ingrese al mismo. Luego cree dos archivos de 1MB cada uno usando:

```
dd if=/dev/zero bs=1M count=1 of=file-1
dd if=/dev/zero bs=1M count=1 of=file-2
```

En ese mismo directorio cree un `Dockerfile` con el siguiente contenido:

```
FROM busybox
ADD . /app
RUN chmod 400 /app/*
```

Construya una imagen llamada mikroways/image-size con el dockerfile mencionado y analice por qué la imagen base se incrementa en 4MB en vez de 2MB. ¿Podría proponer una solución para incrementar únicamente 2MB?

10. Es una buena práctica escribir entrypoints como shell scripts bash o bourn shell. Esos scripts en sus primeras líneas suelen utilizar la sentencia `set -eo pipefail` Analice por qué es tan importante utilizar esas opciones.

    > Para entender como funciona puede probar instanciar un contenedor busybox y probar en él qué sucede si setea por un lado `set -e` y luego la instrucción completa. Para cada caso, probar luego de setear las opciones comandos como: `ls /not-exist` y `ls /not-exist | echo No problem`

Entregables

Los ejercicios a entregar son 1, 2, 3, 4, 5, 6, 9. Las entregas deben responder a las preguntas de cada ejercicio, los comandos o el Dockerfile usados para completarlos.