PRÁCTICA 1: Introducción a Docker Compose - Conceptos y Comandos Básicos

Objetivo

Comprender qué es Docker Compose, su estructura básica y los comandos fundamentales para gestionar aplicaciones multi-contenedor.

Conceptos Teóricos

¿Qué es Docker Compose?

Docker Compose es una herramienta que permite definir y ejecutar aplicaciones Docker multi-contenedor. Utiliza un archivo YAML para configurar los servicios de tu aplicación.

Componentes básicos del archivo docker-compose.yml:

- `version`: Versión del formato de Compose
- `services`: Define los contenedores que componen tu aplicación
- `image`: Imagen Docker a utilizar
- `ports`: Mapeo de puertos (host:contenedor)
- `container_name`: Nombre personalizado del contenedor

Pasos de la Práctica

Paso 1: Verificar instalación de Docker y Docker Compose

```
# Verificar Docker
docker --version

# Verificar Docker Compose
docker compose version
```

Paso 2: Crear estructura de directorios

```
# Crear directorio para la práctica
mkdir practica1-docker-compose
```

```
cd practica1-docker-compose
```

Paso 3: Crear archivo docker-compose.yml

Crea un archivo llamado `docker-compose.yml` con el siguiente contenido:

```yaml
version: '3.8'

services:
  web:
    image: nginx:alpine
    container_name: mi-servidor-web
    ports:
      - "8080:80"

  database:
    image: mysql:8.0
    container_name: mi-base-datos
    environment:
      MYSQL_ROOT_PASSWORD: password123
      MYSQL_DATABASE: mi_aplicacion
```

Explicación línea por línea:

- `version: '3.8'` : Especifica la versión de la sintaxis de Docker Compose
- `services:` : Sección donde definimos todos nuestros contenedores
- `web:` : Nombre del primer servicio (puedes elegir cualquier nombre, este parámetro es opcional)
- `image: nginx:alpine` : Imagen Docker que usaremos (Nginx versión Alpine, más liviana)
- `container_name:` : Nombre específico para el contenedor
- `ports:` : Mapeo puerto 8080 del host → puerto 80 del contenedor
- `database:` : Segundo servicio para MySQL
- `environment:` : Variables de entorno necesarias para MySQL

Paso 4: Comandos básicos de Docker Compose

Antes de inicializar la aplicación a continuación se lista los comandos más utilizados en docker compose:

```
# 1. Iniciar todos los servicios (modo detached/background)
docker compose up -d
# 2. Ver el estado de los servicios
docker compose ps
# 3. Ver los logs de todos los servicios
docker compose logs
# 4. Ver logs de un servicio específico
docker compose logs web
# 5. Seguir logs en tiempo real
docker compose logs -f web
# 6. Detener los servicios (sin eliminar contenedores)
docker compose stop
# 7. Iniciar servicios previamente detenidos
docker compose start
# 8. Reiniciar servicios
docker compose restart
# 9. Detener y eliminar contenedores, redes
docker compose down
# 10. Ver información detallada de un servicio
docker compose config
```

Paso 5: Probar la aplicación

Para inicializar la aplicación ejecute los comandos listados a continuación:

```
docker compose up -d
```

Posteriormente abrir su navegador y ejecutar acceder a la siguiente url `http://localhost:8080`

Paso 6: Explorar los contenedores

En ocasiones aun cuando las los contenedores están en ejecución la aplicación puede presentar problemas a continuación se mostrará como es posible ejecutar un comando dentro el contenedor sin acceder directamente a el.

```
# Ejecutar comando dentro del contenedor web
docker compose exec web ls /usr/share/nginx/html
```

Mediante este método es posible acceder a la base de datos desde el contenedor. Una vez conectado al motor de base de datos ejecute el comando "SHOW DATABASES;" este comando permitirá visualizar las base de datos que se encuentran disponibles en el contenedor de BD.

```
# Acceder a shell del contenedor de base de datos
docker compose exec database mysql -uroot -ppassword123

# Dentro de MySQL, ejecutar:
# SHOW DATABASES;
# EXIT;
```

Paso 7: Limpieza

Para eliminar los contenedores ejecute el comando mostrado en la parte inferior:

```
# Detener y eliminar todo
docker compose down

# Verificar que se eliminaron
docker compose ps
```

Ejercicios:

1. Modifica el puerto del servicio web a 9090
2. Agrega un tercer servicio usando la imagen `redis:alpine` y publica este servicio por el puerto 9090.
3. Cambia el nombre del contenedor de la base de datos
4. Investigar cual es la función de `docker compose up --build`
5. ¿Cuál es la diferencia entre docker compose stop y docker compose down?
6. Agregar una nueva variable (DEBUG=True) de entorno al servicio web, posteriormente acceder al contenedor y validar que la variable fue creada con el comando set | grep -i DEBUG.

Puntos Clave

1. ¿Cuál la función de `container_name` y `environment` en el docker compose?