Lab 5

1. Preparación del Entorno

Asegúrate de tener Docker Desktop para Windows instalado y ejecutándose. Puedes verificar la instalación de Docker en CMD:

```
docker --version
```

2. Listar Puertos de un Contenedor

a. **Iniciar un Contenedor con Puertos Externos**: Ejecuta un contenedor de prueba y mapea los puertos. Usaremos un servidor NGINX como ejemplo:

```
docker run -d -p 8080:80 --name mi-nginx nginx
```

Este comando descarga la imagen de NGINX (si aún no la tienes), inicia un contenedor llamado `mi-nginx` y mapea el puerto 80 del contenedor al puerto 8080 de tu máquina host.

b. **Listar los Puertos del Contenedor**: Utiliza el siguiente comando para ver los puertos mapeados:

```
docker port mi-nginx
```

3. Crear una Red con Docker (Network Create)

a. **Crear una Red de Tipo Bridge**: Crea una red personalizada en Docker:

```
docker network create --driver bridge mi-red-bridge
```

4. Port Forwarding en una Red Personalizada

a. **Iniciar un Contenedor en la Red Personalizada**: Ejecuta un contenedor y únete a la red `mi-red-bridge`:

```
docker run -d -p 9090:80 --name mi-nginx-bridge --network mi-red-bridge nginx
```

Aquí, estás mapeando el puerto 80 del contenedor al puerto 9090 de tu host y conectando el contenedor a `mi-red-bridge`.

5. Trabajar con Modo de Red Bridge

a. **Crear y Conectar otro Contenedor a la Misma Red**: Inicia otro contenedor y conéctalo a la misma red:

```
docker run -d --name otro-nginx --network mi-red-bridge nginx
```

Este contenedor también estará en la red `mi-red-bridge`, permitiendo la comunicación entre `mi-nginx-bridge` y `otro-nginx`.

6. Trabajar con Modo de Red Host

Nota: El modo de red 'host' en Docker Desktop para Windows funciona de manera diferente que en Linux debido a las diferencias en la forma en que las redes son manejadas en Windows.

a. **Iniciar un Contenedor con Modo de Red Host**: En Windows, el modo host no aisla el contenedor de la red del host de la misma manera que en Linux. Sin embargo, puedes probarlo con:

```
docker run -d --name mi-nginx-host --network host nginx
```

En Windows, este contenedor no estará aislado del host en términos de red, pero no tendrá acceso directo a las interfaces de red del host como en Linux.

7. Comunicación entre Contenedores en la Misma Red

a. **Instalar Herramientas de Red en uno de los Contenedores**: Primero, necesitarás herramientas para probar la conectividad, como `curl`. Puedes instalar `curl` en uno de los contenedores. Vamos a instalarlo en `mi-nginx-bridge`. Para hacer esto, primero necesitas acceder al contenedor:

```
docker exec -it mi-nginx-bridge /bin/bash
```

Dentro del contenedor, si es necesario, actualiza los paquetes e instala `curl` (Nota: NGINX no tiene un gestor de paquetes por defecto, por lo que este paso podría no ser aplicable directamente en un contenedor NGINX. Aquí se asume una imagen que tiene un gestor de paquetes como `apt`):

```
apt-get update
apt-get install curl
```

b. **Probar la Conectividad**: Ahora, intenta conectar desde `mi-nginx-bridge` a `otro-nginx` usando el nombre del contenedor, que Docker resuelve a su dirección IP en la red:

```
curl http://otro-nginx
```

Esto debería devolver el HTML predeterminado de la página de NGINX del contenedor `otro-nginx`. Esto demuestra que los contenedores en la misma red pueden comunicarse entre sí usando sus nombres de contenedor, que funcionan como nombres de host dentro de la red de Docker.

c. **Salir del Contenedor**: Una vez que hayas terminado, puedes salir del contenedor `mi-nginx-bridge`:

```
exit
```

Nota Importante

- En Docker, los contenedores en la misma red pueden comunicarse entre sí utilizando los nombres de los contenedores como nombres de host. Docker internamente maneja la resolución de nombres y el enrutamiento de red.
- Si utilizas imágenes de Docker que no tienen un shell o herramientas comunes como `curl` o `apt`, considera usar imágenes diferentes para la práctica o imágenes personalizadas que incluyan las herramientas necesarias.

Con este paso adicional, habrás demostrado cómo los contenedores en la misma red Docker pueden comunicarse entre sí.