Tarea Curso Docker

Nombre:

Apellidos:

Laboratorio 1: Comandos básicos

1. Comando para parar todos los contenedores

```

```

2. Comando para eliminar todos los contenedores

```

```

3. Lanzar un contenedor llamado `web1` con la imagen `agarciaf/intranet`

```

```

4. Lanzar un contenedor llamado `bd1` con la imagen `mariadb`

```

```

5. Lanzar un contenedor llamado `bd2` con la imagen `postgres`

```

```

6. Lanzar un contenedor llamado `web2` que exponga el puerto en nuestra máquina `81` basado en la imagen `nginx`, y que se reinicie siempre

7. ¿Qué IP tienen los contenedores `web1` y `web2`?

8. Comando para ver las estadísticas del contenedor `web1` y `web2`

Laboratorio 2: Publicar una imagen en Docker Hub

En este laboratorio seguiremos este procedimiento para lanzan un contenedor y convertirlo a imagen, tras la conversión a imagen subiremos dicha imagen a nuestro Docker hub, en el cual tendremos que tener creada una cuenta para poder realizar dicho laboratorio.

docker

Partimos de la imagen httpd que tiene como document-root el directorio: `/usr/local/apache2/htdocs`

1. Lanzamos contenedor intranet

2. Entramos en el contenedor y creamos un index.html

3. Nos salimos del contenedor y visualizamos los puertos que tenemos nateados

4. Salvamos el contenedor a imagen

5. Eliminamos el contenedor base

6. Lanzamos un nuevo contenedor basado en la imagen creada del contenedor anteriormente:

7. Publicar una imagen en Docker Hub:

Laboratorio 3: Explique la siguiente configuración de este archivo docker compose, para desplegar un entorno de Mysql y WordPress

```
version: '3.8'  #

services:
  db:  #
    image: mysql:5.7  #
    container_name: mysql_db  #
    restart: always  #
    environment:  #
      MYSQL_ROOT_PASSWORD: root_password  #
      MYSQL_DATABASE: wordpress  #
      MYSQL_USER: wp_user  #
      MYSQL_PASSWORD: wp_password  #
    volumes:
      - db_data:/var/lib/mysql  #
    networks:
      - wp_network  #

  wordpress:  #
    image: wordpress:5.6.2-php7.3  #
    container_name: wordpress_app  #
    depends_on:
      - db  #
    ports:
      - "8080:80"  #
```

```
    restart: always  #
    environment:  #
      WORDPRESS_DB_HOST: db:3306  #
      WORDPRESS_DB_USER: wp_user  #
      WORDPRESS_DB_PASSWORD: wp_password  #
      WORDPRESS_DB_NAME: wordpress  #
    volumes:
      - wp_data:/var/www/html  #
    networks:
      - wp_network  #

volumes:  #
  db_data:  #
  wp_data:  #

networks:  #
  wp_network:  #
```

```

```

Laboratorio 4: Explique los siguientes comandos de docker compose

Comando	Descripción
`docker compose build`	
`docker compose up --build`	
`docker compose logs`	
`docker compose logs -f`	
`docker compose exec <servicio>`	
`docker compose run <servicio>`	
`docker compose scale <servicio>=N`	
`docker compose rm`	
`docker compose kill`	
`docker compose top`	
`docker compose pull`	
`docker compose push`	

Laboratorio 5: Explique las siguientes estructuras de los campos de docker compose, que faltan por comentar

```

```

```yaml
version: '3.8'  #

services:  #
  db:  #
    image: mysql:5.7  #
    container_name: mysql_db  #
    restart: always  #
    environment:  #
      MYSQL_ROOT_PASSWORD: root_password  #
      MYSQL_DATABASE: example_db  #
      MYSQL_USER: example_user  #
      MYSQL_PASSWORD: example_password  #
    volumes:  #
      - db_data:/var/lib/mysql  #
    networks:  #
      - example_network  #

  app:  #
    image: wordpress:5.8  #
    container_name: wordpress_app  #
    depends_on:  #
      - db  #
    ports:  #
      - "8080:80"  #
    restart: always  #
    environment:  #
      WORDPRESS_DB_HOST: db:3306  #
      WORDPRESS_DB_USER: example_user  #
      WORDPRESS_DB_PASSWORD: example_password  #
      WORDPRESS_DB_NAME: example_db  #
    volumes:  #
      - app_data:/var/www/html  #
    networks:  #
      - example_network  #

volumes:  #
  db_data:  #
  app_data:  #

networks:  #
  example_network:  #
```