EXPERIMENT 6

AIM: Working with Docker Compose File to Control Multiple Containers

Steps to Complete:

Creating compose files

- Create a directory named nginx in your root.

```
mkdir nginx
```

- Switch to that directory and create a file named docker-compose.yaml

```
cd nginx
```

```
vi docker-compose.yml
```

- Use docker-compose version 2 to create docker-compose.yaml file.
  Create a service named "databases". Use image named "mysql"

  Map container 3306 port to host machine 3306 port.

  Add environment variables named "MYSQL_ROOT_PASSWORD", "MYSQL_DATABASE", "MYSQL_USER" and "MYSQL_PASSWORD" along with corresponding values for all.

```
cat evs.env
```

```
MYSQL_ROOT_PASSWORD=redhat08
MYSQL_DATABASE=nginxdb
MYSQL_USER=root
```

Add another service named "web"
Use image "nginx"

```
cat docker-compose.yml
```

```
version: '3'
services:
  databases:
    image: mysql
    ports:
      - "3307:3306"
    env_file:
      - evs.env
  web:
    image: nginx
```

```
    ports:
      - "80:80"
    depends_on:
      - databases
```

Running images using docker-compose

- Save docker-compose.yaml file and do docker-compose up.

```
docker-compose up -d
```

- Verify nginx service is up and is accessible on machine.

```
curl localhost:80
```

Stop and remove your docker container using docker-compose.

```
docker-compose down
```