##Docker for DevOps Engineers – Day 18

* **Docker Volume:**
    1. Docker volume is simply a directory.
    2. We need to declare directory as a volume first, then we ca share it with containers.
    3. If we have stopped the container, still we can access the volume.
    4. We can declare a directory as a volume only while creating a container.
    5. We can't create a volume from existing container.
    6. We can share one volume across any number of containers.
    7. Volume will not be included when we will update an image.
    8. We can map volume in two ways:
        * Container ←→ Container
        * Host ←→ Container

* **Docker Network:**
    1. Docker allows you to create virtual spaces called networks, where you can connect multiple containers (small packages that hold all the necessary files for a specific application to run) together.
    2. This way, the containers can communicate with each other and with the host machine (the computer on which the Docker is installed).
    3. When we run a container, it has its own storage space that is only accessible by that specific container. If we want to share that storage space with other containers, we can't do that.

* **Task-1**
    1. **- Create a multi-container docker-compose file which will bring *UP* and bring *DOWN* containers in a single shot (Example - Create application and database container)**
        ➢ Here we are creating a docker-compose.yml file for one application and database container.

        version : "3.0"
        services :
        web:
        image : nginx
        database :
        image : mysql
        environment :

MYSQL_ROOT_PASSWORD : Secrete@123

```
vagrant@vagrant:~/dockerCompose$ cat docker-compose.yml
version : "3.0"
services :
  web:
    image : nginx
  database :
   image : mysql
   environment :
     MYSQL_ROOT_PASSWORD : Secret@123
```

2. - Use the `docker-compose up` command with the `-d` flag to start a multi-container application in detached mode.
   ➢ We have used below command to start the application.

   docker-compose up -d

```
vagrant@vagrant:~/dockerCompose$ docker-compose up -d
Starting dockercompose_web_1 ...
Starting dockercompose_database_1 ...
Starting dockercompose_web_1
Starting dockercompose_database_1 ... done
```

3. - Use the `docker-compose scale` command to increase or decrease the number of replicas for a specific service. You can also add 'replicas' in deployment file for *auto-scaling*.

   docker-compose up -d --scale database=3

```
vagrant@vagrant:~/dockerCompose$ docker-compose up -d --scale database=3
dockercompose_web_1 is up-to-date
Recreating dockercompose_database_1 ...
Recreating dockercompose_database_1 ... done
Creating dockercompose_database_2 ...
Creating dockercompose_database_3 ...
Creating dockercompose_database_2 ... done
Creating dockercompose_database_3 ... done
vagrant@vagrant:~/dockerCompose$
```

4. - Use the 'docker-compose ps' command to view the status of all containers, and `docker-compose logs` to view the logs of a specific service.

   docker-compose ps

```
vagrant@vagrant:~/dockerCompose$ docker-compose ps
            Name                    Command              State           Ports
---------------------------------------------------------------------------------------
dockercompose_database_1    docker-entrypoint.sh mysqld    Up      3306/tcp, 33060/tcp
dockercompose_database_2    docker-entrypoint.sh mysqld    Up      3306/tcp, 33060/tcp
dockercompose_database_3    docker-entrypoint.sh mysqld    Up      3306/tcp, 33060/tcp
dockercompose_web_1         /docker-entrypoint.sh ngin ...  Up      80/tcp
vagrant@vagrant:~/dockerCompose$
```

docker-compose logs web

```
vagrant@vagrant:~/dockerCompose$ docker-compose logs web
Attaching to dockercompose_web_1
web_1     | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
web_1     | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
web_1     | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
web_1     | 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
web_1     | /docker-entrypoint.sh: Configuration complete; ready for start up
web_1     | 2023/03/24 08:51:47 [notice] 1#1: using the "epoll" event method
web_1     | 2023/03/24 08:51:47 [notice] 1#1: nginx/1.23.3
web_1     | 2023/03/24 08:51:47 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
web_1     | 2023/03/24 08:51:47 [notice] 1#1: OS: Linux 4.15.0-58-generic
web_1     | 2023/03/24 08:51:47 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
web_1     | 2023/03/24 08:51:47 [notice] 1#1: start worker processes
web_1     | 2023/03/24 08:51:47 [notice] 1#1: start worker process 28
web_1     | 2023/03/24 12:12:38 [notice] 1#1: signal 3 (SIGQUIT) received, shutting down
web_1     | 2023/03/24 12:12:38 [notice] 28#28: gracefully shutting down
web_1     | 2023/03/24 12:12:38 [notice] 28#28: exiting
web_1     | 2023/03/24 12:12:38 [notice] 28#28: exit
web_1     | 2023/03/24 12:12:41 [notice] 1#1: signal 17 (SIGCHLD) received from 28
web_1     | 2023/03/24 12:12:41 [notice] 1#1: worker process 28 exited with code 0
web_1     | 2023/03/24 12:12:41 [notice] 1#1: exit
web_1     | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
web_1     | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
web_1     | 10-listen-on-ipv6-by-default.sh: info: IPv6 listen already enabled
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
web_1     | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
web_1     | /docker-entrypoint.sh: Configuration complete; ready for start up
web_1     | 2023/03/27 06:30:40 [notice] 1#1: using the "epoll" event method
web_1     | 2023/03/27 06:30:40 [notice] 1#1: nginx/1.23.3
web_1     | 2023/03/27 06:30:40 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
web_1     | 2023/03/27 06:30:40 [notice] 1#1: OS: Linux 4.15.0-58-generic
web_1     | 2023/03/27 06:30:40 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
web_1     | 2023/03/27 06:30:40 [notice] 1#1: start worker processes
web_1     | 2023/03/27 06:30:40 [notice] 1#1: start worker process 21
```

5. - Use the `docker-compose down` command to stop and remove all containers, networks, and volumes associated with the application.

docker-compose down

```
vagrant@vagrant:~/dockerCompose$ docker-compose down
Stopping dockercompose_database_3 ... done
Stopping dockercompose_database_2 ... done
Stopping dockercompose_database_1 ... done
Stopping dockercompose_web_1      ... done
Removing dockercompose_database_3 ... done
Removing dockercompose_database_2 ... done
Removing dockercompose_database_1 ... done
Removing dockercompose_web_1      ... done
Removing network dockercompose_default
vagrant@vagrant:~/dockerCompose$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS   PORTS   NAMES
vagrant@vagrant:~/dockerCompose$
```

* **Task-2**

    1. **Learn how to use Docker Volumes and Named Volumes to share files and directories between multiple containers.**
        > We have created docker container and docker volume using pre-existing image.

docker run -it --name cntr_1 -v /volume1 ac232364af84 /bin/bash

```
vagrant@vagrant:~/Volume$ docker images
REPOSITORY                          TAG       IMAGE ID       CREATED        SIZE
nginx                               latest    ac232364af84   3 days ago     142MB
python                              latest    df3e9d105d6c   4 days ago     921MB
mysql                               latest    483a8bc460a9   4 days ago     530MB
trainwithshubham/react-django-app   <none>    7619048f7856   5 months ago   962MB
vagrant@vagrant:~/Volume$ docker run -it --name cntr_1 -v /volume1 ac232364af84 /bin/bash
root@5b58ae2e28cb:/# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var  volume1
root@5b58ae2e28cb:/#
```

    2. **Create two or more containers that read and write data to the same volume using the `docker run --mount` command.**

docker run -it --name cntr_2 --privileged=true --volumes-from 5b58ae2e28cb nginx /bin/bash

```
vagrant@vagrant:~/Volume$ docker run -it --name cntr_2 --privileged=true --volumes-from 5b58ae2e28cb nginx /bin/bash
root@f5ff64196b82:/# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var  volume1
root@f5ff64196b82:/# cd volume1/
root@f5ff64196b82:/volume1# ls
cntr_1_File.txt
root@f5ff64196b82:/volume1# cat cntr_1_File.txt
This is from cntr_1.
root@f5ff64196b82:/volume1#
```

    3. **Verify that the data is the same in all containers by using the docker exec command to run commands inside each container.**

docker exec -it <cntr_1_name> /bin/bash

```
vagrant@vagrant:~/Volume$ docker exec -it cntr_1 /bin/bash
root@5b58ae2e28cb:/# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var  volume1
root@5b58ae2e28cb:/# cd volume1/
root@5b58ae2e28cb:/volume1# ls
cntr_1_File.txt
root@5b58ae2e28cb:/volume1# cat cntr_1_File.txt
This is from cntr_1.
root@5b58ae2e28cb:/volume1#
```

docker exec -it <cntr_2_name> /bin/bash

```
vagrant@vagrant:~/Volume$ docker exec -it cntr_2 /bin/bash
root@f5ff64196b82:/# ls
bin  boot  dev  docker-entrypoint.d  docker-entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var  volume1
root@f5ff64196b82:/# cd volume1/
root@f5ff64196b82:/volume1# ls
cntr_1_File.txt
root@f5ff64196b82:/volume1# cat cntr_1_File.txt
This is from cntr_1.
root@f5ff64196b82:/volume1#
```

    4. **Use the docker volume ls command to list all volumes and docker volume rm command to remove the volume when you're done.**

docker volume ls

docker volume rm <volumeame>