**Day - 01**

**Agenda**

**======**

1. Docker Networking
2. Lab : Wordpress & Mysql
3. Docker Compose
4. Lab: Wordpress & Mysql with Docker Compose

**Docker Installation**

sudo apt update -y

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

sudo systemctl status docker

sudo usermod ubuntu -G docker

sudo su - ubuntu

sudo apt install net-tools -y

**To get details of container:**

docker inspect container\_name

**To Login to the container:**

docker exec -it container\_name bash

**To Install ping command in the container:**

apt update -y

apt install iputils-ping

ping 172.17.0.3

**To Deploy mysql container :**

docker run -d --name=mydbcontainer -e MYSQL\_DATABASE=wordpressdb -e MYSQL\_ROOT\_PASSWORD=dbpassword mysql

**To Deploy wordpress container:**

docker run -d --name mywordpress-container -p 80:80 wordpress

**To list docker network on the docker host:**

docker network ls

**To create a network:**

docker network create --driver bridge network\_name

docker run -d --name=mydbcontainer -e MYSQL\_DATABASE=wordpressdb -e MYSQL\_ROOT\_PASSWORD=dbpassword --network mycustom-network mysql

docker run -d --name mywordpress-container -p 80:80 --network mycustom-network wordpress

**Simulate Failure (Change of db container IP)**

docker stop mydbcontainer

docker run -d --name web01 nginx

docker run -d --name web02 nginx

docker start mydbcontainer

docker stop web01 web01

docker stop mydbcontainer

docker start mydbcontainer

**Day - 02**

**Agenda**

**—--------**

1. Docker compose
2. Lab: mysql & wordpress with docker compose
3. Docker swarm
4. Services in docker swarm
5. Updating service & rollback
6. Docker compose with docker swarm

**Docker Compose Installation**

sudo apt install docker-compose -y

**Download & install Visual Studio Code**

<https://code.visualstudio.com/download>

**VS Code Extensions -** YAML , Docker

**Docker compose file**

**Supported filenames: docker-compose.yml, docker-compose.yaml, compose.yml, compose.yaml**

services:

mydb:

container\_name: mydb

image: mysql

environment:

- MYSQL\_DATABASE=wordpressdb

- MYSQL\_ROOT\_PASSWORD=dbpassword

ports:

- "3306:3306"

myapp:

container\_name: wordpress-container

image: wordpress

ports:

- "80:80"

**To Validate docker compose file**

docker-compose config

**To Apply docker-compose.yaml file**

docker-compose up

**To run container in background/deattached mode**

docker-compose up -d

**To apply docker-compose with a different file name other than docker-compose.yaml**

docker-compose -f filename.yaml up

**To stop & remove the container deployed via docker-compose**

docker-compose down

**To remove container after removing from docker-compose file**

docker-compose up -d --remove-orphans

**To list container deployed through docker-compose file**

docker-compose ps

**To check logs**

docker-compose logs

**Docker Swarm**

**Master Node Setup**

sudo hostnamectl set-hostname master

**To Initialize Master Node**

docker swarm init

Output:

Swarm initialized: current node (degvpriicxm5udqzosf2durhm) is now a manager.

To add a worker to this swarm, run the following command:

docker swarm join --token SWMTKN-1-6djo6qexhmtnj64er7kcmm7bl0qi4jdk6caebpzplyekxjgwq0-3o330l7sf97j6plmu9j5flwn7 192.168.122.1:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

Copy the docker swarm join command from the command output of docker swarm init

Open port 2377 security group of Master Node

**To Initialize Worker Node**

docker swarm join --token SWMTKN-1-6djo6qexhmtnj64er7kcmm7bl0qi4jdk6caebpzplyekxjgwq0-3o330l7sf97j6plmu9j5flwn7 192.168.122.1:2377

**To List nodes connected**

docker node ls

**To deploy container in swarm:**

docker service create --name <name-of-service> --replicas <number-of-replicas> −p 80:80

<image-name>

docker service create --name myapp --replicas 3 −p 80:80 nginx

**To delete service**

docker service rm service\_name

**To list services**

docker service ls

**To list containers running in a service**

docker service ps service\_name

**To scale replicas in service**

docker service scale service\_name=replicas

docker service scale myapp=10

**To update service image:**

docker service update --image=httpd webapp

**To rollback changes made in service**

docker service rollback service\_name

**To check logs of a service**

docker service logs -f service\_name