## **Project 02:**

## **Objectives:**

* Deploy an application across multiple Docker Swarm worker nodes.
* Place specific components on designated nodes.
* Monitor and troubleshoot using Docker logs.
* Modify and redeploy the application.

### **Project Outline:**

1. **Initialize Docker Swarm and Join Worker Nodes**
2. **Label Nodes for Specific Component Placement**
3. **Create a Docker Stack File**
4. **Deploy the Application**
5. **Monitor and Troubleshoot Using Docker Logs**
6. **Modify and Redeploy the Application**

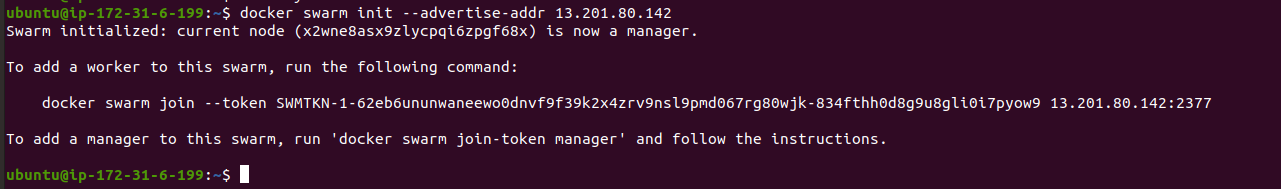
### **Step-by-Step Guide**

#### **1. Initialize Docker Swarm and Join Worker Nodes**

On the manager node, initialize Docker Swarm:

docker swarm init --advertise-addr <MANAGER-IP>

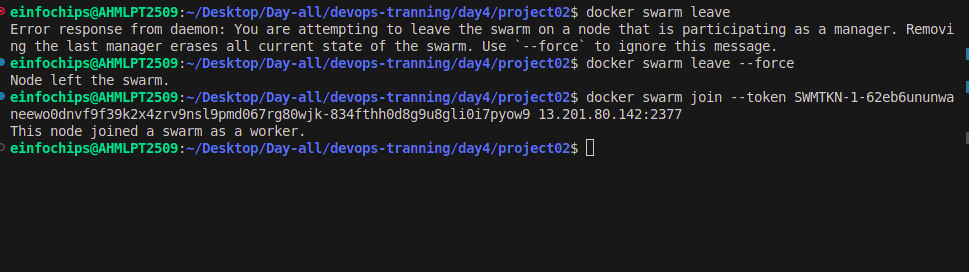
docker swarm init --advertise-addr 13.201.80.142



Join the worker nodes to the swarm. On each worker node, run the command provided by the docker swarm init output:

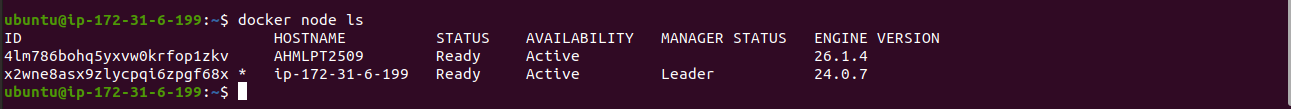
docker swarm join --token <SWARM-TOKEN> <MANAGER-IP>:2377

docker swarm join --token SWMTKN-1-62eb6ununwaneewo0dnvf9f39k2x4zrv9nsl9pmd067rg80wjk-834fthh0d8g9u8gli0i7pyow9 13.201.80.142:2377



Verify the nodes have joined:

docker node ls

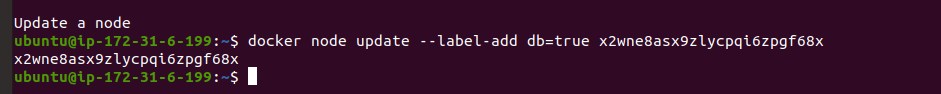


#### **2. Label Nodes for Specific Component Placement**

Label nodes to specify where certain components should run. For example, label a node for the database service:

docker node update --label-add db=true <NODE-ID>

docker node update --label-add db=true x2wne8asx9zlycpqi6zpgf68x

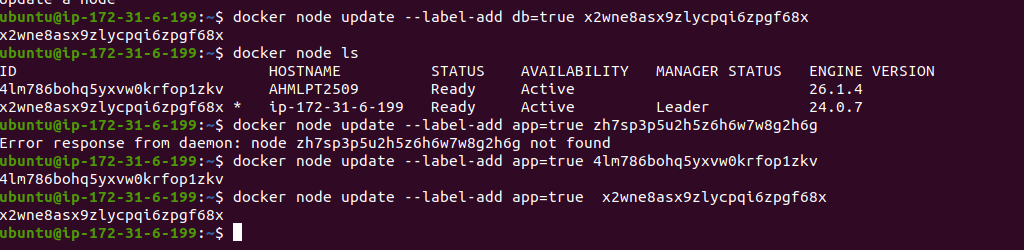


Label another node for the application service:

docker node update --label-add app=true <NODE-ID>

docker node update --label-add app=true 4lm786bohq5yxvw0krfop1zkv

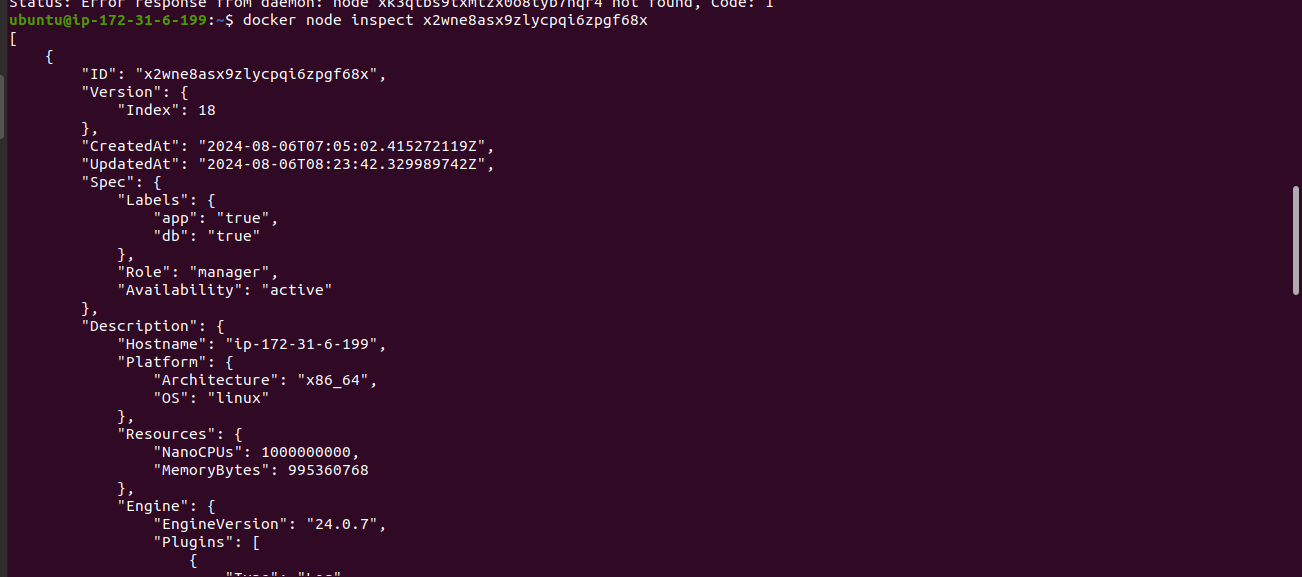
docker node update --label-add app=true x2wne8asx9zlycpqi6zpgf68x



Verify the labels:

docker node inspect <NODE-ID>

docker node inspect x2wne8asx9zlycpqi6zpgf68x



#### **3. Create a Docker Stack File**

Create a docker-stack.yml file to define the services and node placement constraints:

version: '3.8'

services:

db:

image: mysql:5.7

volumes:

- mysql\_data:/var/lib/mysql

networks:

- app\_network

environment:

MYSQL\_ROOT\_PASSWORD: example

MYSQL\_DATABASE: appdb

MYSQL\_USER: user

MYSQL\_PASSWORD: password

deploy:

placement:

constraints:

- node.labels.db == true

app:

image: your-app-image

networks:

- app\_network

ports:

- "8000:80"

environment:

DB\_HOST: db

deploy:

replicas: 2

placement:

constraints:

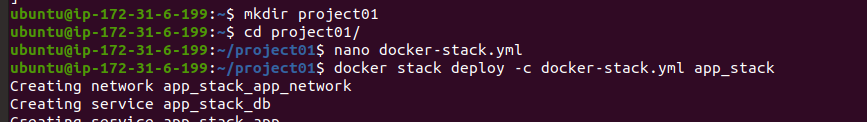
- node.labels.app == true

volumes:

mysql\_data:

networks:

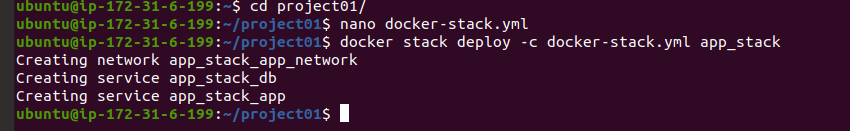
app\_network:



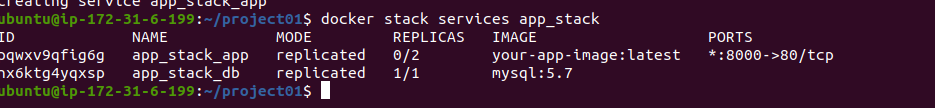
#### **4. Deploy the Application**

Deploy the stack using Docker Swarm:

docker stack deploy -c docker-stack.yml app\_stack



docker stack services app\_stack

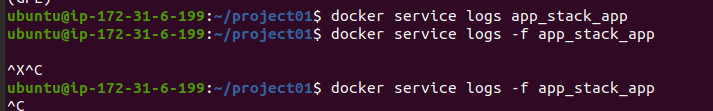


#### **5. Monitor and Troubleshoot Using Docker Logs**

Check the logs for the services:

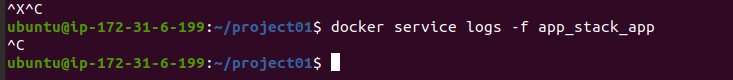
docker service logs app\_stack\_db

docker service logs app\_stack\_app



Follow the logs in real-time to monitor issues:

docker service logs -f app\_stack\_app



#### **6. Modify and Redeploy the Application**

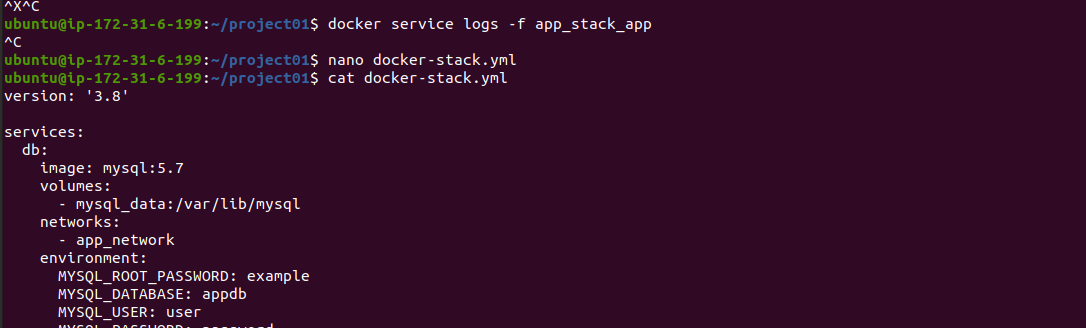
Make modifications to the application or the stack file as needed. For example, change the number of replicas:

services:

app:

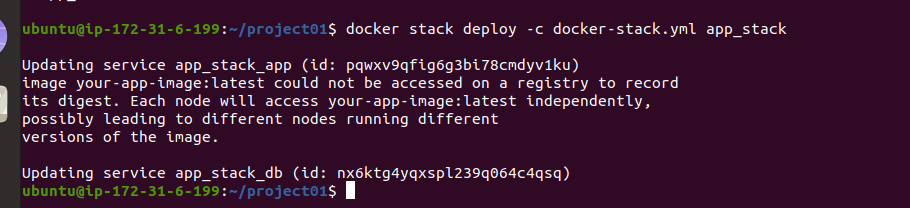
deploy:

replicas: 3



Update the stack with the new configuration:

docker stack deploy -c docker-stack.yml app\_stack



Verify the changes:

docker stack services app\_stack

