Containerization with Docker

Course-End Project: **Swarm Microservice Deployment**

Objective

To deploy a scalable, multi-service voting application on a manager node, ensuring efficient orchestration, fault tolerance, and seamless monitoring through Docker visualizer.

Real-time scenario:

John, a DevOps engineer, is tasked with deploying a voting application through multiple microservices. By creating a Docker compose file and deploying it on a manager node in a distributed system, they ensure that each service is efficiently orchestrated and fault-tolerant. To monitor the deployment, John integrates Docker visualizer as a microservice, providing real-time insights. This setup simplifies the deployment process, enhances scalability, and ensures the application runs smoothly in a production environment.

Tasks

The following tasks outline the process of deploying swarm microservice:

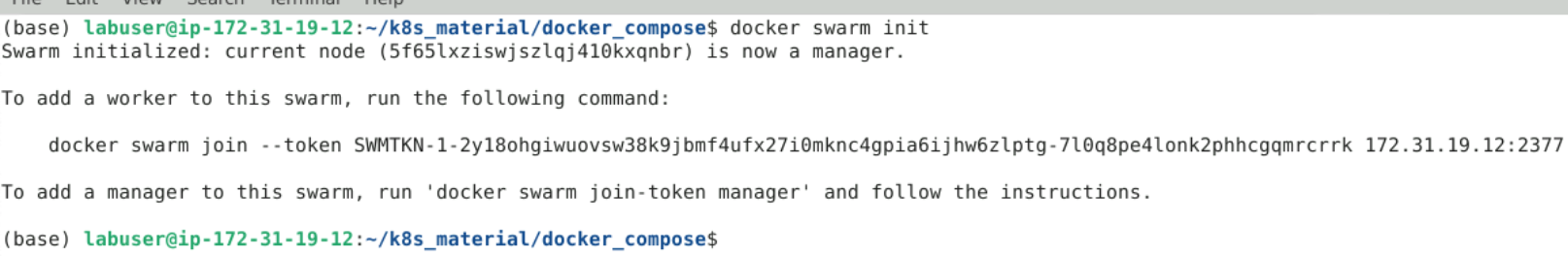
1. Set up the network and storage infrastructure.

2. Define and configure microservices.

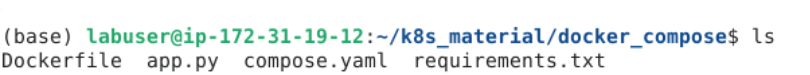
3. Deploy microservices across Docker swarm.

Solution

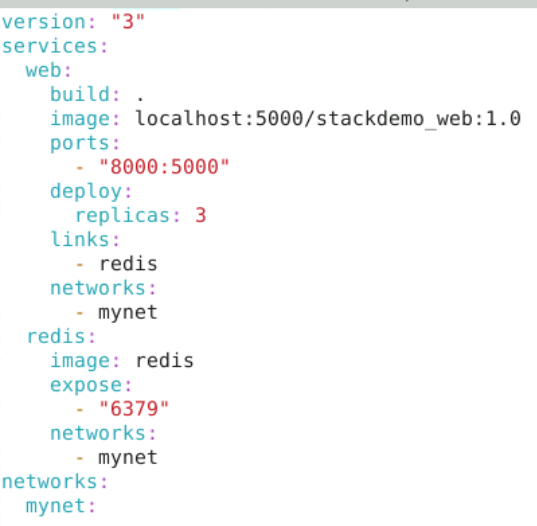
Step 1: Configure docker swarm



Step-2: Create necessary files as below.



Step-3: Define multi-container app structure. (compose.yml)

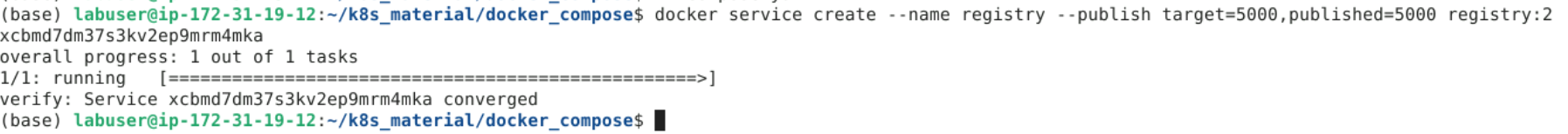


Step4: Install docker compose

Sudo apt update && sudo apt install docker-compose

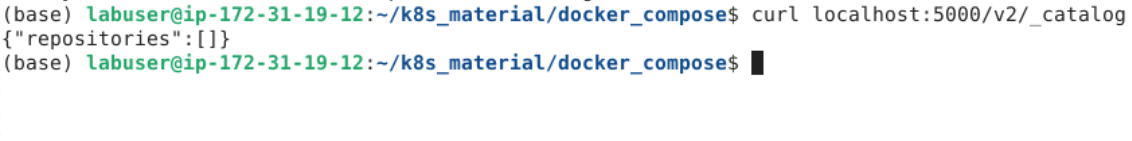
Step5: Creates a **private Docker registry** as a Swarm service.

docker service create --name registry --publish target=5000,published=5000 registry:2



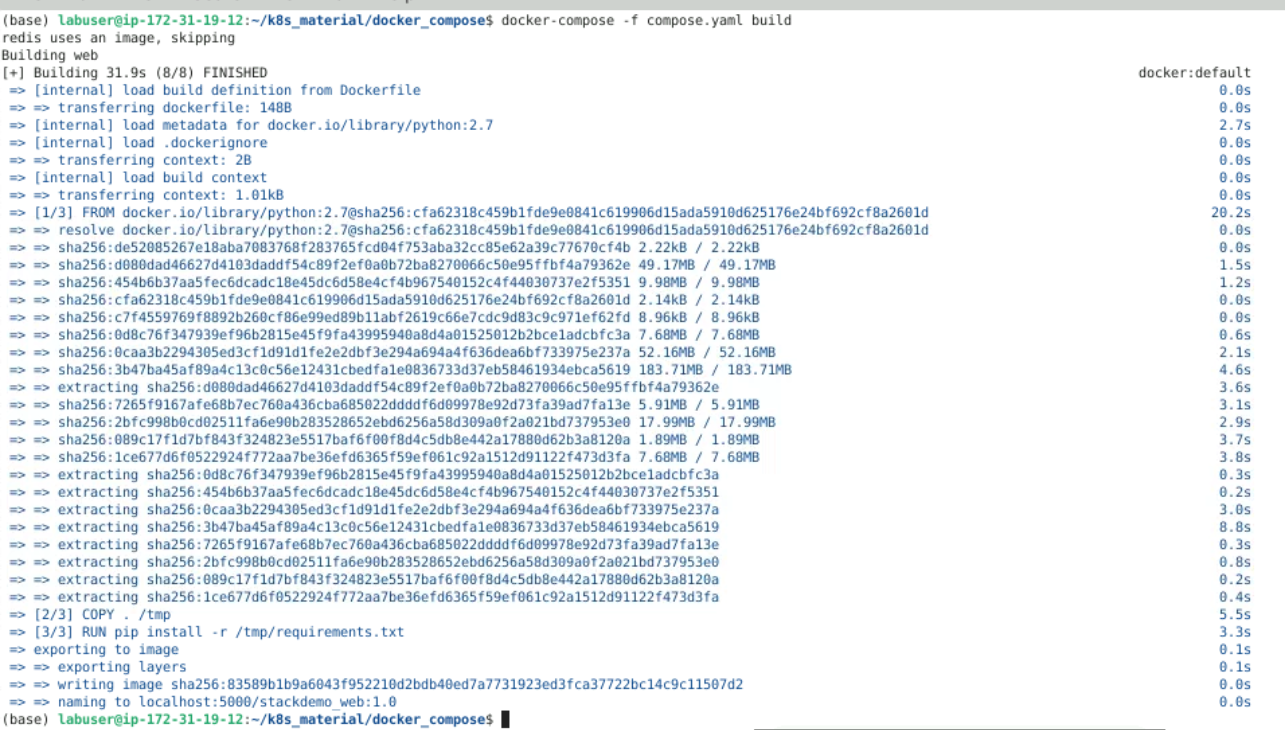
Step6: Verifies that the local registry is running.

curl localhost:5000/v2/\_catalog



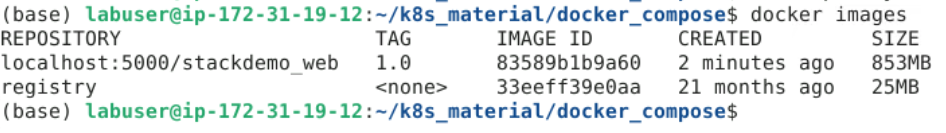
Step7: Building image using the local Dockerfile and tags it as stackdemo\_web.

docker-compose -f compose.yaml build



Step8: Verifies the image was built successfully.

docker images



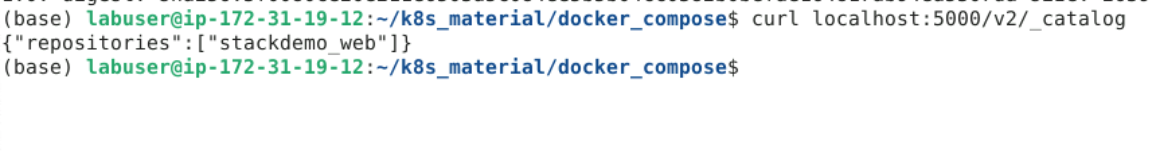
Step9: Pushes your built image to the **local registry at localhost:5000**

docker-compose -f compose.yaml push



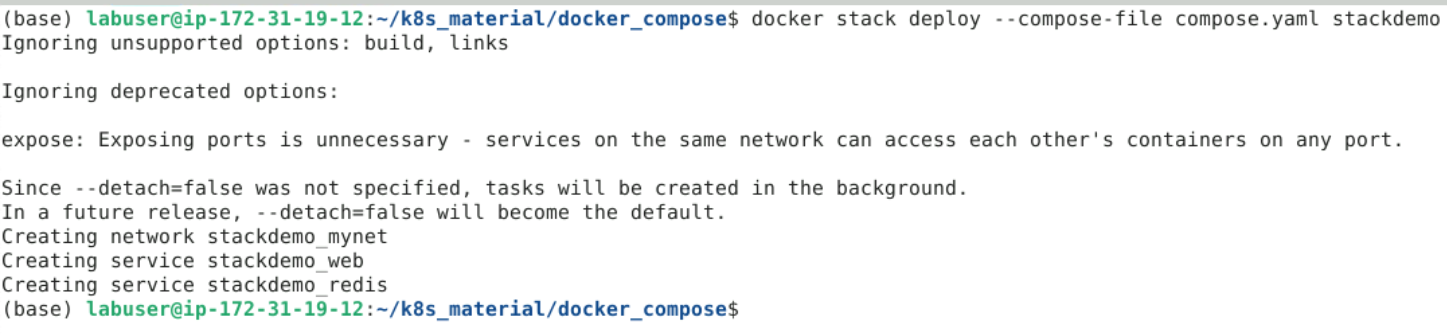
Step10: Now should show your pushed image

curl localhost:5000/v2/\_catalog



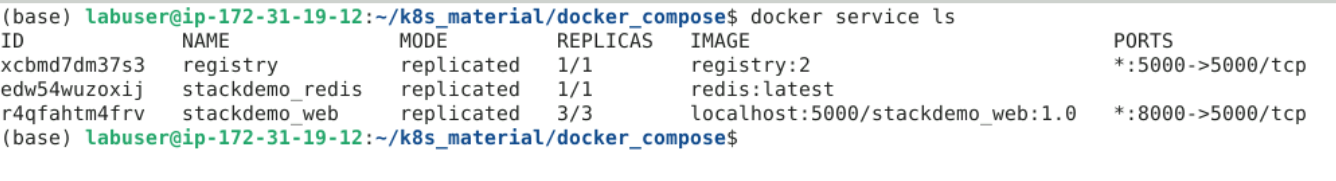
Step11: Deploys the app as a Swarm stack named stackdemo.

docker stack deploy --compose-file compose.yaml stackdemo



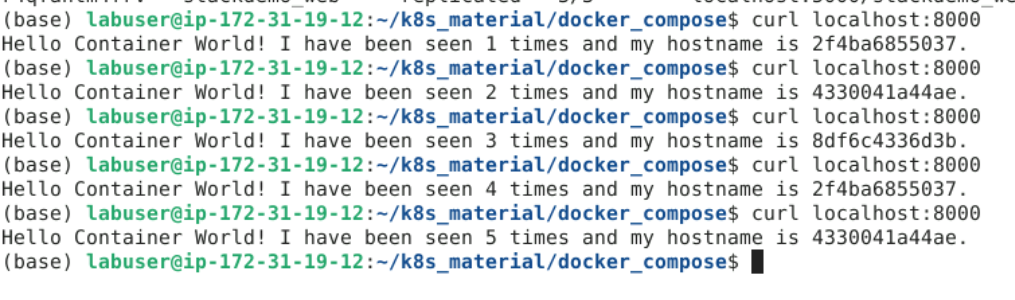
Step12: Lists services created by the stack.

docker service ls



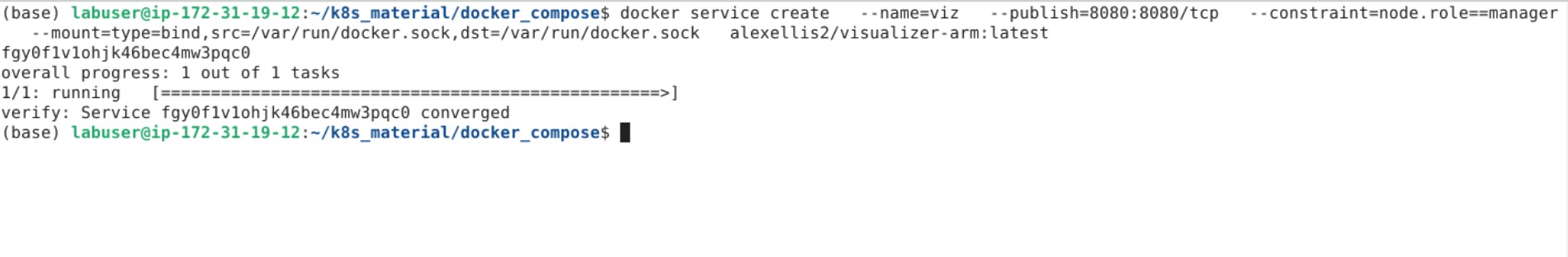
Step13: Accesses the running container via published port 8000

curl localhost:8000



Step14: Deploying Visualizer

**docker service create   --name=viz   --publish=8080:8080/tcp   --constraint=node.role==manager   --mount=type=bind,src=/var/run/docker.sock,dst=/var/run/docker.sock   alexellis2/visualizer-arm:latest**



Step15: Open the browser to see the containers.

