

Homework 4

Exercise 1:

Deploy Back End and Front End with Docker Compose.

Deploy two services (`api` and `fe`) using Docker Compose with the following specifications:

1. Create a compose.yml file.

```
ubuntu@k8s-instance-6:~/training$ cat docker_comp.yml
version: '3.8'

services:
  api:
    image: docker.jala.pro/docker-training/backend:calebespinoza
    container_name: backend
    networks:
      - mynetwork
    volumes:
      - api-data:/app
  fe:
    image: docker.jala.pro/docker-training/frontend:calebespinoza
    container_name: frontend
    ports:
      - "8080:80"
    networks:
      - mynetwork
    volumes:
      - fe-data:/usr/share/nginx/html
networks:
  mynetwork:
    driver: bridge
volumes:
  api-data:
  fe-data:
ubuntu@k8s-instance-6:~/training$ docker volume ls
```

2. Define two services: api and fe , using the private registry images:

docker.jala.pro/docker-training/backend:[TAG]

docker.jala.pro/docker-training/frontend:[TAG]]

```
ubuntu@k8s-instance-6:~/training$ cat docker_comp.yml
version: '3.8'

services:
  api:
    image: docker.jala.pro/docker-training/backend:calebespinoza
    container_name: backend
    networks:
      - mynetwork
    volumes:
      - api-data:/app
  fe:
    image: docker.jala.pro/docker-training/frontend:calebespinoza
    container_name: frontend
    ports:
      - "8080:80"
    networks:
      - mynetwork
    volumes:
      - fe-data:/usr/share/nginx/html
networks:
  mynetwork:
    driver: bridge
volumes:
  api-data:
  fe-data:
```

2. Use a user-defined bridge network to allow inter-container communication.

```
ubuntu@k8s-instance-6:~/training$ cat docker_comp.yml
version: '3.8'

services:
  api:
    image: docker.jala.pro/docker-training/backend:calebespinoza
    container_name: backend
    networks:
      - mynetwork
    volumes:
      - api-data:/app
  fe:
    image: docker.jala.pro/docker-training/frontend:calebespinoza
    container_name: frontend
    ports:
      - "8080:80"
    networks:
      - mynetwork
    volumes:
      - fe-data:/usr/share/nginx/html
networks:
  mynetwork:
    driver: bridge
volumes:
  api-data:
  fe-data:
```

3. Use volumes to persist data of each service.

```
ubuntu@k8s-instance-6:~/training$ cat docker_comp.yml
version: '3.8'

services:
  api:
    image: docker.jala.pro/docker-training/backend:calebespinoza
    container_name: backend
    networks:
      - mynetwork
    volumes:
      - api-data:/app
  fe:
    image: docker.jala.pro/docker-training/frontend:calebespinoza
    container_name: frontend
    ports:
      - "8080:80"
    networks:
      - mynetwork
    volumes:
      - fe-data:/usr/share/nginx/html
networks:
  mynetwork:
    driver: bridge
volumes:
  api-data:
  fe-data:
```

The image shows a terminal window on the left and a web browser on the right. The terminal window displays the contents of a `docker-compose.yml` file and the output of the `docker volume ls` command. The web browser shows the 'Container Info' page for a container named '04-compose - docops - Obsidia', displaying its hostname and IP address.

```
frontend /docker-entrypoint.sh nginx ... Up 0.0.0.0:8080->80/tcp, :::8080->80/tcp
ubuntu@k8s-instance-6:~/training$ cat docker_comp.yml
version: '3.8'

services:
  api:
    image: docker.jala.pro/docker-training/backend:calebospinoza
    container_name: backend
    networks:
      - mynetwork
    volumes:
      - api-data:/app
  fe:
    image: docker.jala.pro/docker-training/frontend:calebospinoza
    container_name: frontend
    ports:
      - "8080:80"
    networks:
      - mynetwork
    volumes:
      - fe-data:/usr/share/nginx/html
networks:
  mynetwork:
    driver: bridge
volumes:
  api-data:
  fe-data:

ubuntu@k8s-instance-6:~/training$ docker volume ls
DRIVER      VOLUME NAME
local       dbdata
local       mariadb_data
local       my_data_volume
local       my_named_vol
local       training_api-data
local       training_fe-data
local       vol_demo_01

timed out waiting for input: auto-logout
Connection to 10.26.32.177 closed.
```

The web browser shows the 'Container Info' page for a container named '04-compose - docops - Obsidia'. The page displays the following information:

- Hostname: 768424ec85ea
- IP Address: 172.18.0.3

Improved index.html from container

```
Starting frontend ... done
ubuntu@k8s-instance-6:~/training$ docker-compose -f docker_comp.yml ps

```

Name	Command	State	Ports
backend	python app.py	Up	
frontend	/docker-entrypoint.sh nginx ...	Up	0.0.0.0:8080->80/tcp, :::8080->80/tcp

```
ubuntu@k8s-instance-6:~/training$ docker exec -it frontend /bin/sh
/usr/share/nginx/html # vi index.html
/usr/share/nginx/html # exit
ubuntu@k8s-instance-6:~/training$ docker-compose -f docker_comp.yml stop
Stopping backend ... done
Stopping frontend ... done
ubuntu@k8s-instance-6:~/training$ docker-compose -f docker_comp.yml up -d
Starting frontend ... done
Starting backend ... done
ubuntu@k8s-instance-6:~/training$ docker-compose -f docker_comp.yml stop
```

Docker :: Container Info

04-compose - docops - Obsidian

Container Info

← → ↺ Not Secure http://10.26.32.177:8080

Homework4-compose

Container Info

Hostname: a965f4181b29

IP Address: 172.18.0.3

Image from docker.jala.pro