## **Project 01**

### **Objectives:**

* Create and manage Docker volumes for data persistence.
* Set up a Docker network for container communication.
* Use Docker Compose to manage multi-container applications.
* View and manage Docker logs.
* Deploy the application using Docker Swarm.

### **Project Outline:**

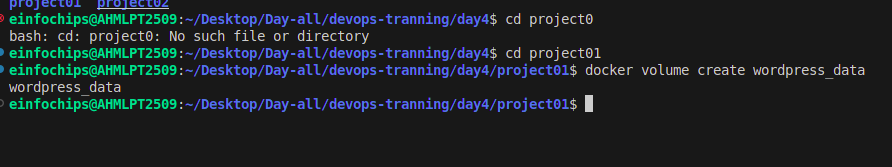
1. **Create Docker Volumes**
2. **Create a Docker Network**
3. **Write a Docker Compose File**
4. **Deploy the Application with Docker Compose**
5. **Manage Docker Logs**
6. **Deploy the Application Using Docker Swarm**

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

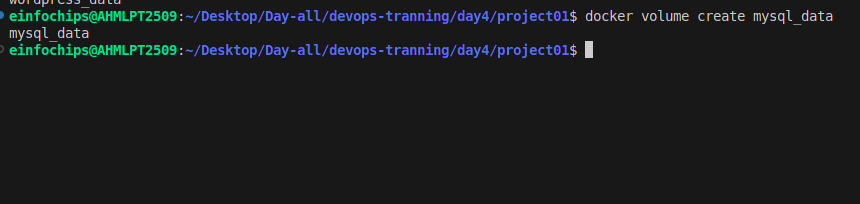
#### **1. Create Docker Volumes**

Docker volumes are used to persist data generated by and used by Docker containers.

docker volume create wordpress\_data



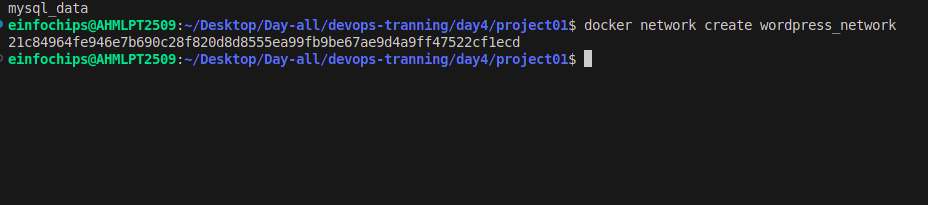
docker volume create mysql\_data



#### **2. Create a Docker Network**

Create a custom network for the containers to communicate.

docker network create wordpress\_network



#### **3. Write a Docker Compose File**

Create a docker-compose.yml file to define and manage the services.

version: '3.3'

services:

db:

image: mysql:5.7

volumes:

- mysql\_data:/var/lib/mysql

networks:

- wordpress\_network

environment:

MYSQL\_ROOT\_PASSWORD: example

MYSQL\_DATABASE: wordpress

MYSQL\_USER: wordpress

MYSQL\_PASSWORD: wordpress

wordpress:

image: wordpress:latest

volumes:

- wordpress\_data:/var/www/html

networks:

- wordpress\_network

ports:

- "8000:80"

environment:

WORDPRESS\_DB\_HOST: db:3306

WORDPRESS\_DB\_USER: wordpress

WORDPRESS\_DB\_PASSWORD: wordpress

WORDPRESS\_DB\_NAME: wordpress

volumes:

mysql\_data:

wordpress\_data:

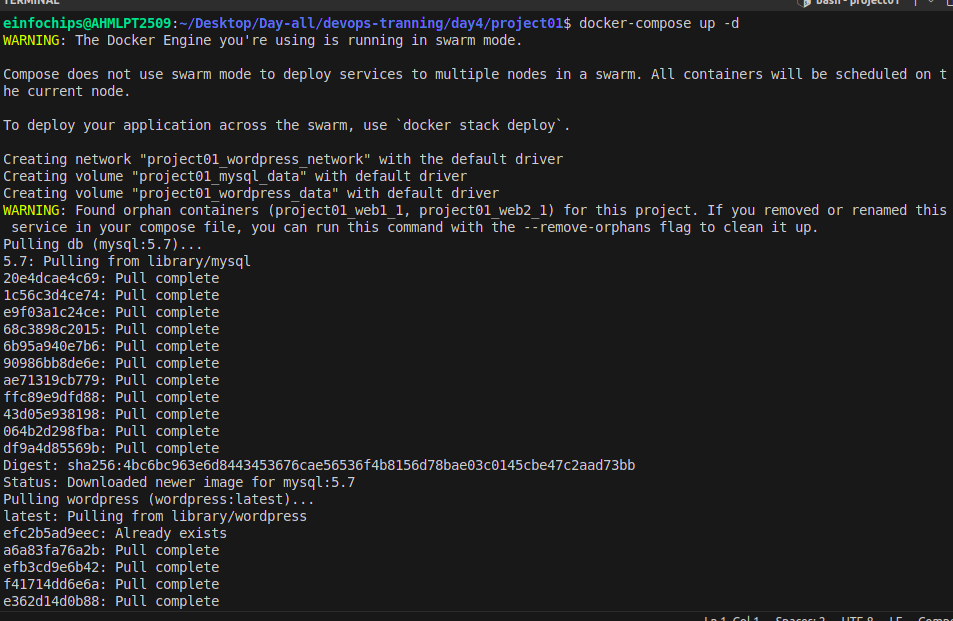
networks:

wordpress\_network:

#### **4. Deploy the Application with Docker Compose**

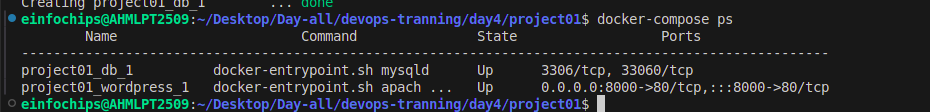
Run the following command to start the services defined in the docker-compose.yml file.

docker-compose up –d

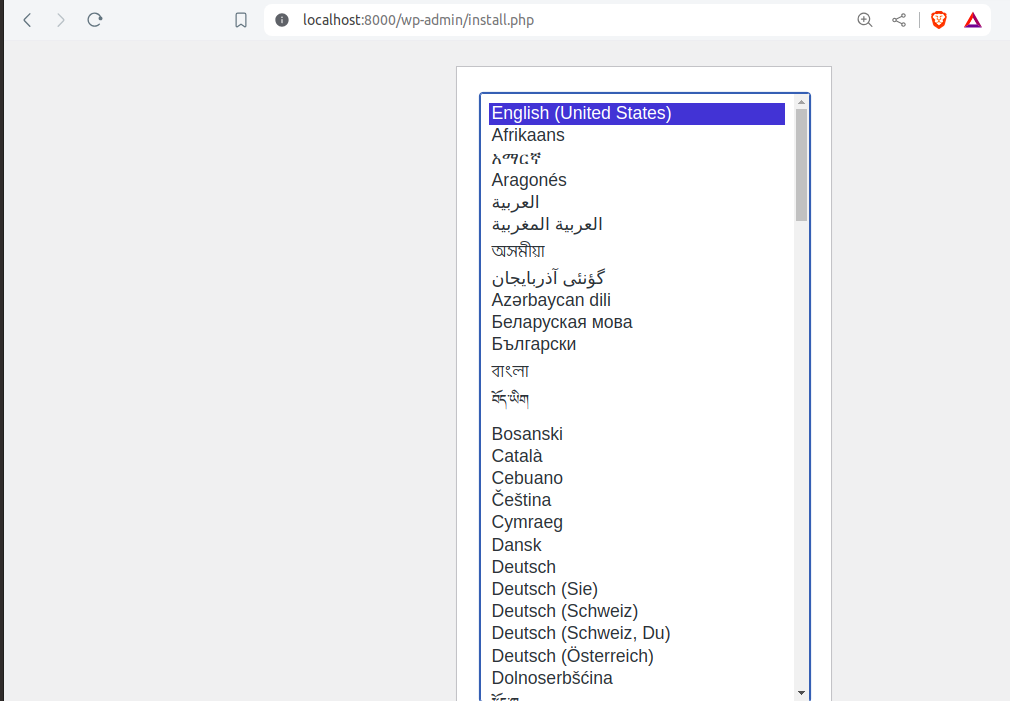


* Verify that the containers are running.

docker-compose ps



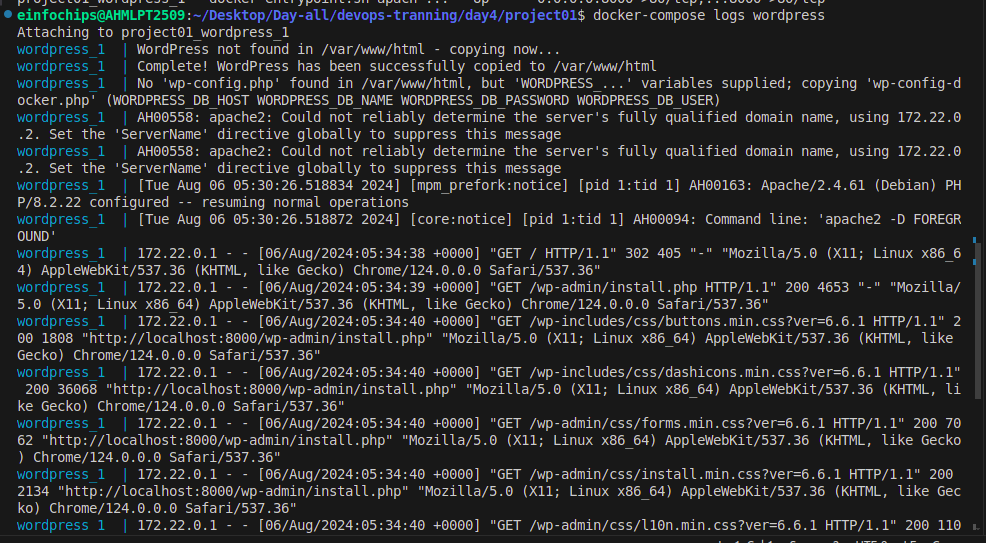
* Access the WordPress setup by navigating to <http://localhost:8000>.



#### **5. Manage Docker Logs**

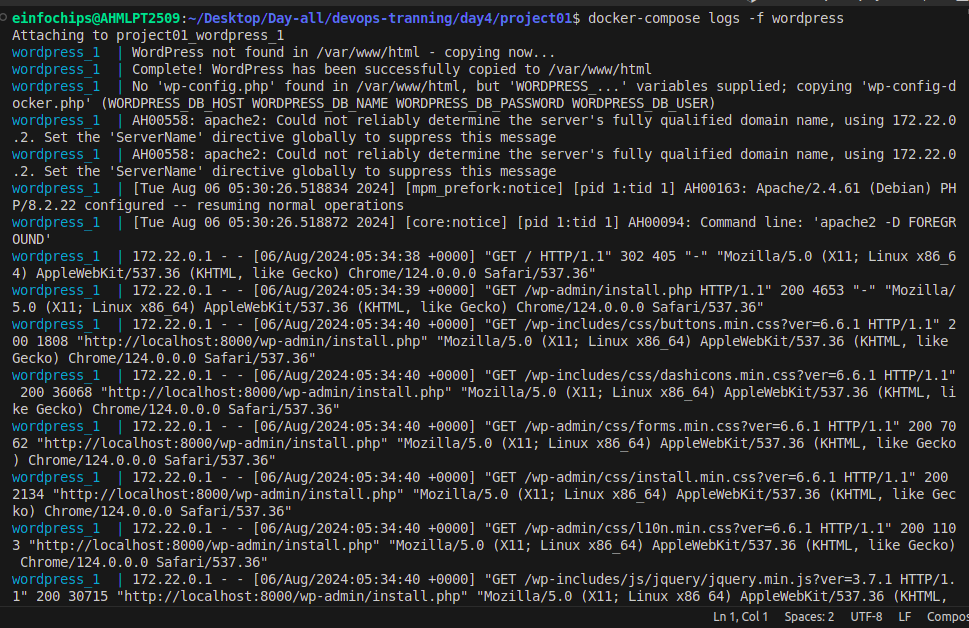
* View logs for a specific service.

docker-compose logs wordpress



* Follow logs for real-time updates.

docker-compose logs -f wordpress

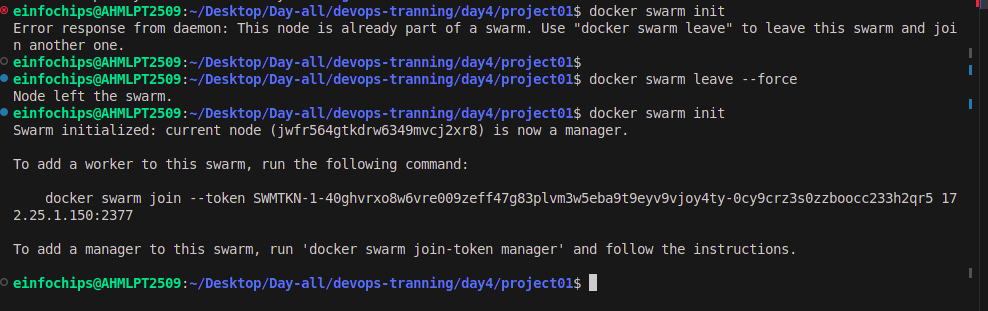


#### **6. Deploy the Application Using Docker Swarm**

Docker Swarm is a native clustering and orchestration tool for Docker.

* Initialize Docker Swarm.

docker swarm init



* Convert the Docker Compose file to a Docker Stack file, docker-stack.yml.

version: '3.3'

services:

db:

image: mysql:5.7

volumes:

- mysql\_data:/var/lib/mysql

networks:

- wordpress\_network

environment:

MYSQL\_ROOT\_PASSWORD: example

MYSQL\_DATABASE: wordpress

MYSQL\_USER: wordpress

MYSQL\_PASSWORD: wordpress

deploy:

replicas: 1

wordpress:

image: wordpress:latest

volumes:

- wordpress\_data:/var/www/html

networks:

- wordpress\_network

ports:

- "8000:80"

environment:

WORDPRESS\_DB\_HOST: db:3306

WORDPRESS\_DB\_USER: wordpress

WORDPRESS\_DB\_PASSWORD: wordpress

WORDPRESS\_DB\_NAME: wordpress

deploy:

replicas: 1

volumes:

mysql\_data:

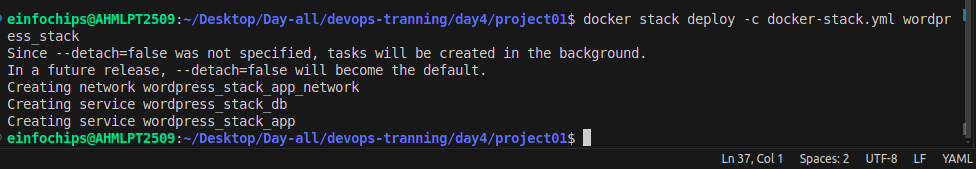
wordpress\_data:

networks:

wordpress\_network:

* Deploy the stack using Docker Swarm.

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



* Verify the stack is running.

docker stack services wordpress\_stack

