### **Lab Exercise 6- Docker-Compose file**

**Objective:**

Set up a WordPress environment using Docker Compose, including a MySQL database as the backend.

**Prerequisites:**

* Docker and Docker Compose installed on your system.

**Step 1: Create a docker-compose.yml File**

1. In the project directory, create a file named docker-compose.yml.
2. Add the following content to docker-compose.yml:

***docker-compose.yml***

version: '3.8'

services:

wordpress:

image: wordpress:latest

ports:

- "8002:80"

environment:

WORDPRESS\_DB\_HOST: db:3306

WORDPRESS\_DB\_USER: wp\_user

WORDPRESS\_DB\_PASSWORD: wp\_pass

WORDPRESS\_DB\_NAME: wp\_database

depends\_on:

- db

db:

image: mysql:latest

environment:

MYSQL\_ROOT\_PASSWORD: root\_password

MYSQL\_DATABASE: wp\_database

MYSQL\_USER: wp\_user

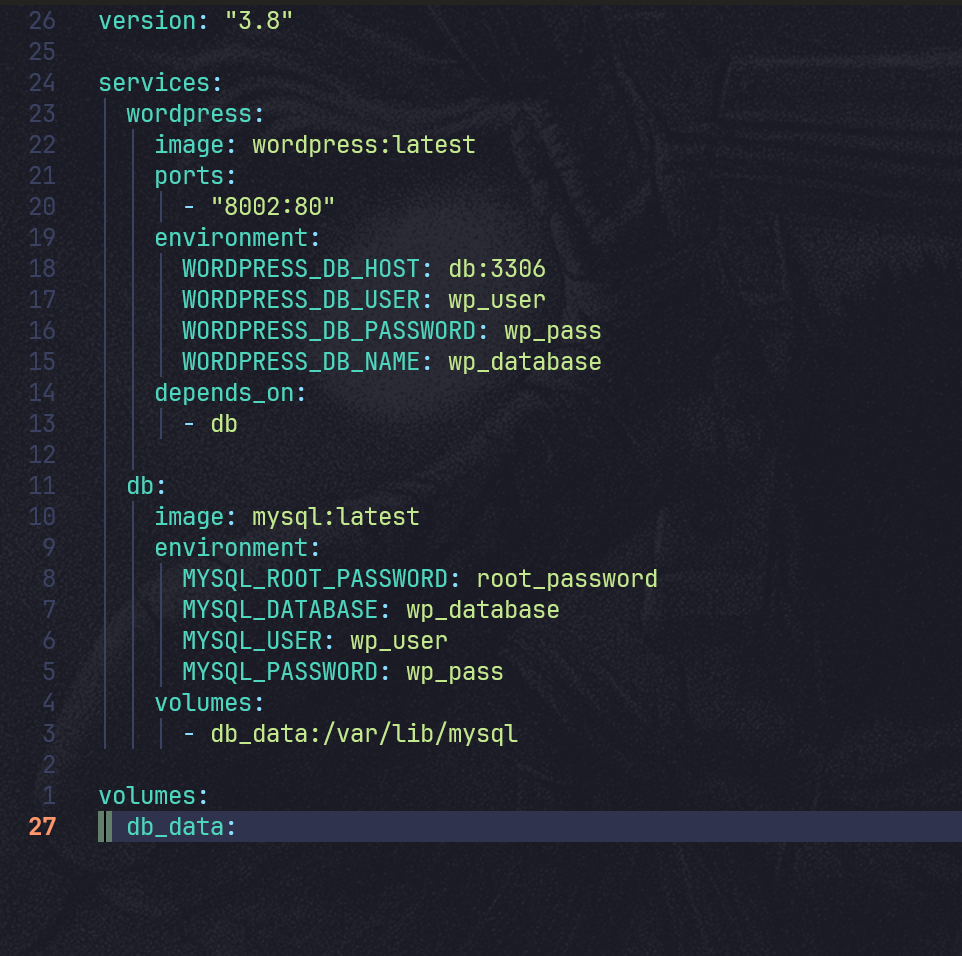
MYSQL\_PASSWORD: wp\_pass

volumes:

- db\_data:/var/lib/mysql

volumes:

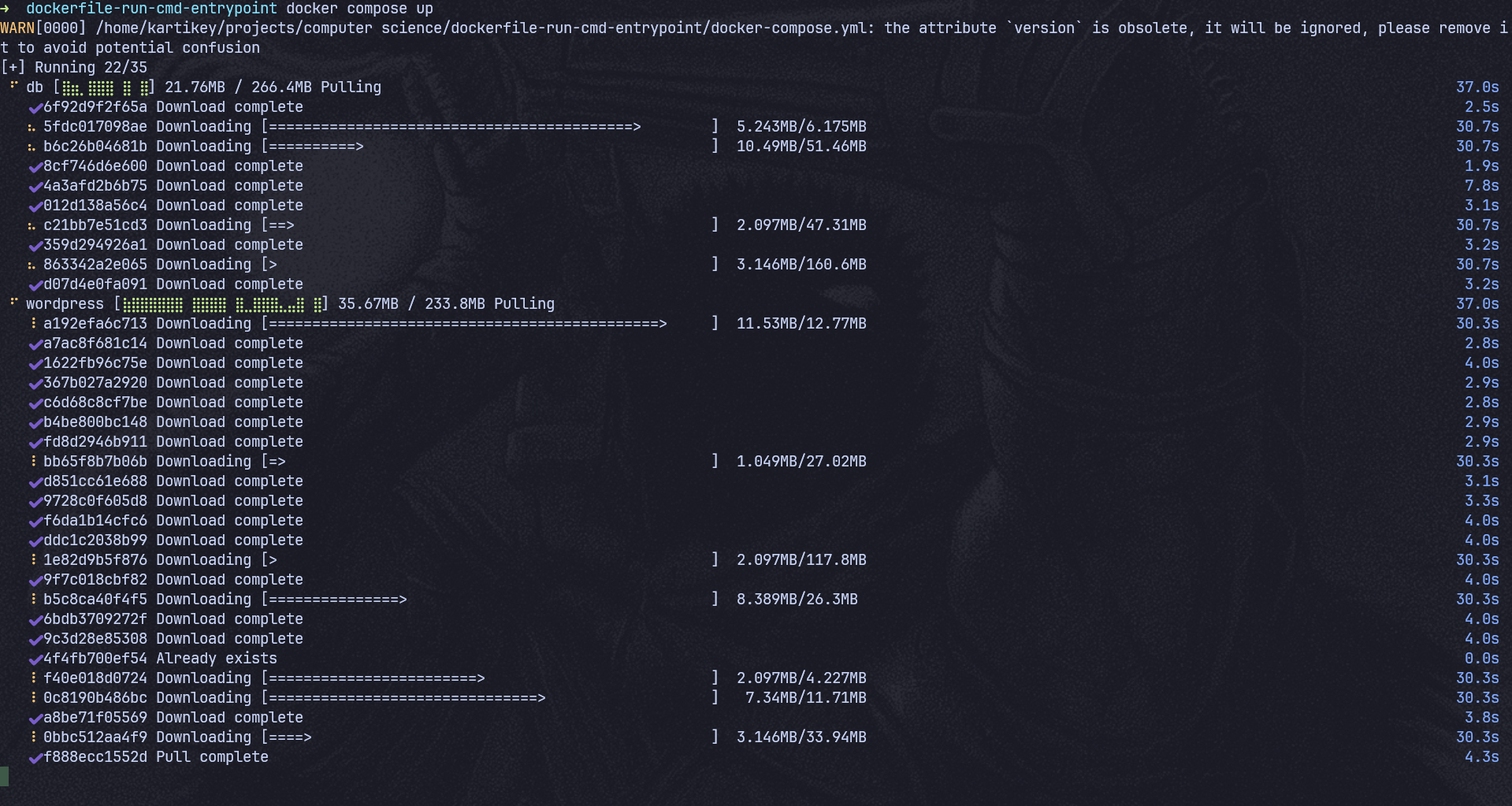
db\_data:

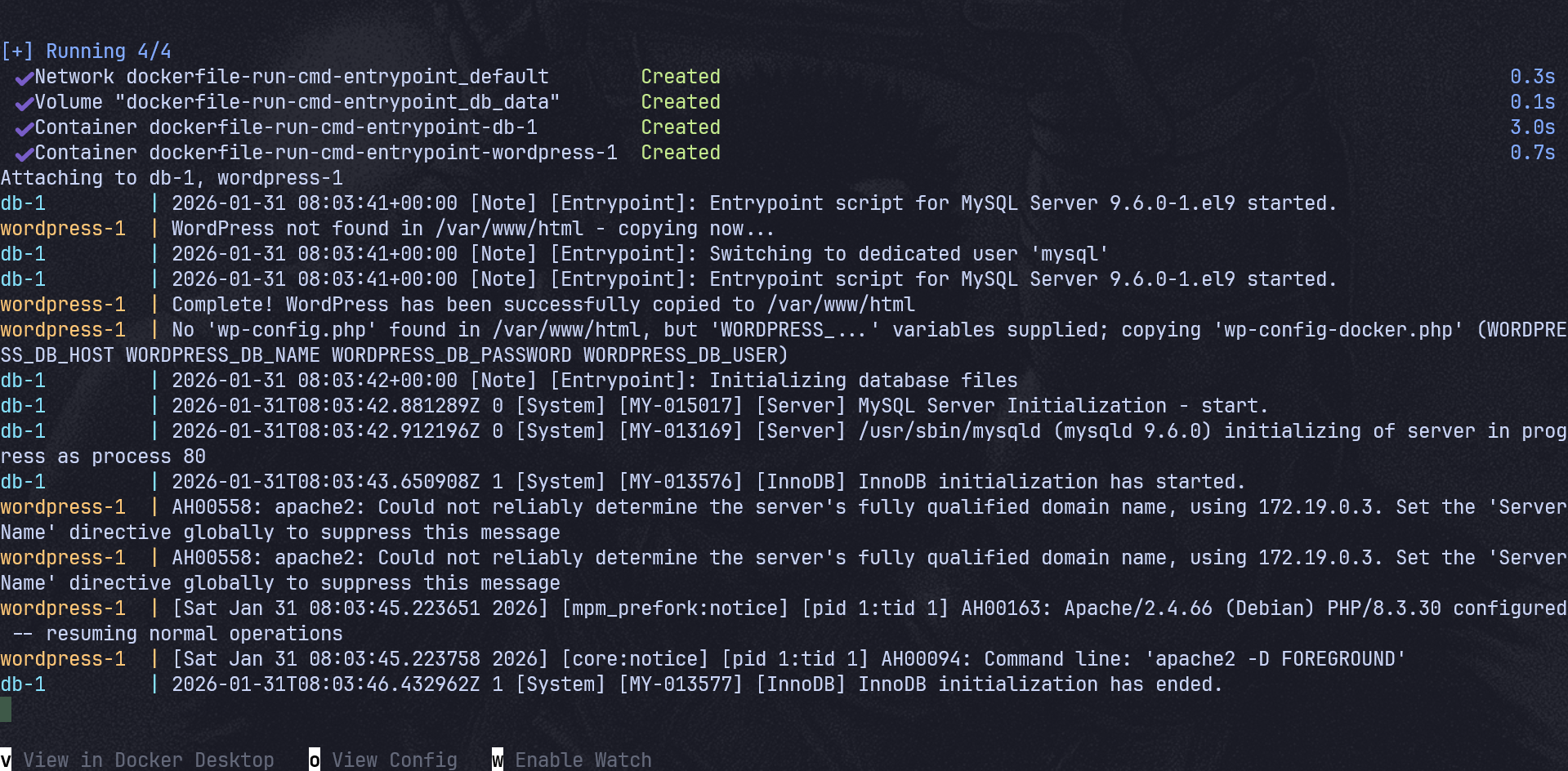


**Step 2: Start the Containers**

1. Run the following command to start the containers:

docker-compose up -d





1. Docker Compose will download the necessary images (WordPress and MySQL) and start both services.

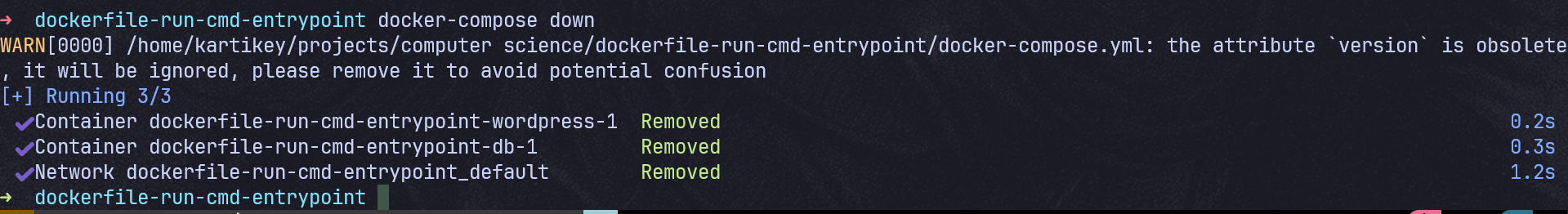
**Step 4: Access WordPress**

1. Open your web browser and go to **http://localhost:8002**
2. Follow the WordPress installation steps to set up your site.

**Step 5: Stop and Remove Containers**

To stop the containers and remove the associated resources, run:

docker-compose down



**Explanation of docker-compose.yml:**

* **wordpress**: Sets up the WordPress container, mapping port 80 inside the container to port 8002 on your local machine.
* **db**: Sets up the MySQL container with a volume (db\_data) for persistent storage.

**Additional Notes:**

* Modify the environment variables as needed for different configurations.
* To view logs, use docker-compose logs -f.

This setup allows you to quickly start a WordPress site locally and experiment with configurations.