Deploy the multi-container application using docker-compose up

Table of Contents

- Description
- Problem Statement
- Prerequisites
  - Software Requirement
  - Hardware Requirement
- Implementation Steps
  - Step-1 :: Run the Program
  - Step-2 :: Manage the Containers
- References

Description

This guide walks you through deploying a multi-container application using Docker Compose. We will set up a **Java-based TodoApp** and a **MySQL database**. With **docker-compose**, you can manage and deploy both services at once, ensuring they are connected and operational.

Problem Statement

The objective is to deploy the **Java-based TodoApp** that relies on a **MySQL** database, using Docker Compose to manage and link both services efficiently without manual configuration.

Prerequisites

Completion of all previous lab guides (up to Lab Guide-06) is required before proceeding with Lab Guide-07.

Software Requirement

- **Docker Desktop**: Ensure Docker and Docker Compose are installed on your Windows machine.
- **Java SDK**: If you're working with Java.
- **Maven/Gradle**: For building your Java app.
- **MySQL Database**: To store todo application data.
- **TodoAPP_MYSQl**: To download the source folder **click here**

Hardware Requirement

- Minimum of 4 GB RAM
- At least 2 cores in the processor
- 5 GB of free storage space for Docker images and containers

Implementation Steps

Step-1 :: Run the Program

To start the multi-container application:

1. Navigate to the project folder containing the **docker-compose.yml** file.

```
cd Docker
```

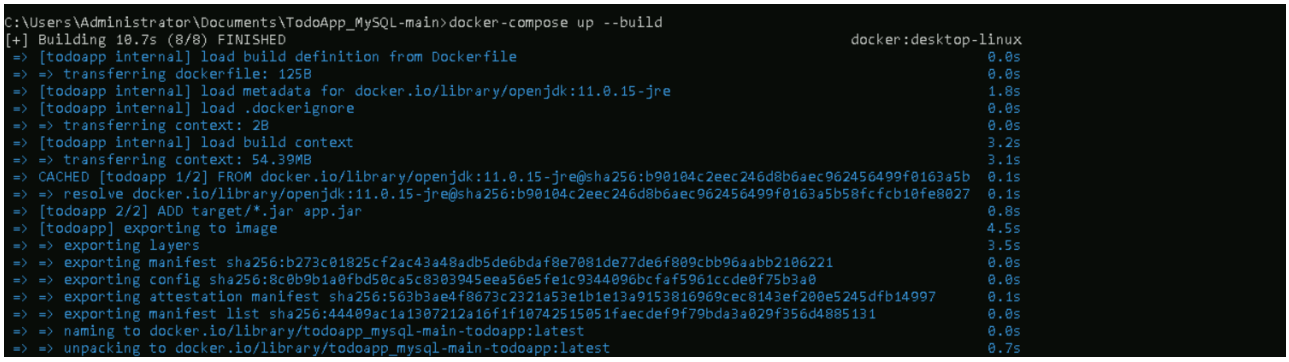
2. Run Docker Compose to build and start the containers:

```
docker-compose up --build
```

This command:

- **Builds** the Java TodoApp container.
- **Pulls** the MySQL image if it's not available locally.
- **Creates** and **starts** the containers for both services.
- **Establishes** a network (todoapp_network) allowing them to communicate.

3. Check the logs to ensure everything is running properly. Once both containers start, you should see output from both services.



4. Once the deployment is successful:

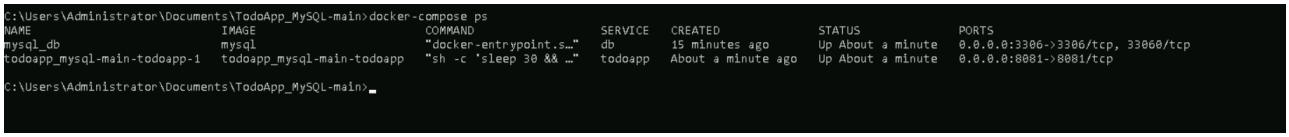
    - **Access** the Java TodoApp on your browser at **http://localhost:8081/swagger-ui/index.html**

- The MySQL database will be running on port **3306**, and the TodoApp will communicate with it.

Step-2 :: Manage the Containers

1. **Check the status of the containers**:

```
docker-compose ps
```



This will display the list of running containers along with their status and ports.

2. **Stop the running containers**:

To stop and remove all the containers, networks, and volumes created by `docker-compose`:

```
docker-compose down
```

3. **Run the containers in detached mode**:

If you want the containers to run in the background (without displaying logs in the terminal), you can run the following command:

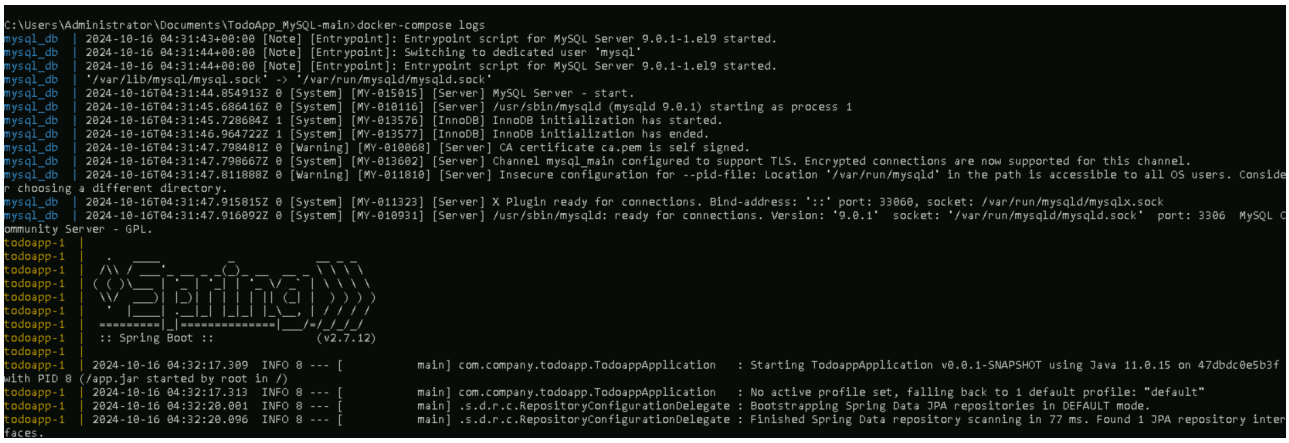
```
docker-compose up -d
```

To stop the detached containers:

```
docker-compose down
```

4. **View container logs** (for troubleshooting):

```
docker-compose logs
```



References

- Docker documentation: https://docs.docker.com/
- Docker Compose official guide: https://docs.docker.com/compose/
- MySQL Docker Hub page: https://hub.docker.com/_/mysql
- Java and Spring Boot examples: Spring Boot with Docker