Create and configure a Docker Compose file to manage multi-container applications

Table of Contents

Description

This guide demonstrates how to use Docker Compose to define a multi-container application. We'll set up two containers: one for a Java-based TodoApp and another for a MySQL database. Using Docker Compose simplifies managing multiple containers and their communication.

Problem Statement

You need to run a Java-based TodoApp that requires a MySQL database. Instead of manually configuring and linking containers, you want to automate this process using a **docker-compose.yml** file.

Prerequisites

Completion of all previous lab guides (up to Lab Guide-02) is required before proceeding with Lab Guide-03.

Software Required

- **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 :: Create Standard Folder Structure

1. Create a project folder:

```
todoapp-docker
├── docker-compose.yml
├── todoapp/
│   ├── Dockerfile
│   ├── src/
│   └── target/
└── db/
```

2. **Dockerfile** for your Java-based TodoApp:

   Inside the `app/` directory, create a `Dockerfile` to build the Java TodoApp.

```
FROM openjdk:11.0.15-jre
ADD target/*.jar app.jar
ENTRYPOINT ["java","-jar","app.jar"]
```

Step-2 :: Create docker-compose.yml

In the root directory of your project, create a **docker-compose.yml** file:

```
services:
  # MySQL Database Service
  db:
    image: mysql
    container_name: mysql_db
    environment:
      MYSQL_ROOT_PASSWORD: P@ssw0rd
      MYSQL_DATABASE: tododb
      MYSQL_PASSWORD: P@ssw0rd
    ports:
      - "3306:3306"
    networks:
      - todoapp_network
    volumes:
      - db_data:/var/lib/mysql

  # Java TodoApp Service
```

```
    todoapp:
      build:
        context: ./todoapp
      container_name: todoapp_container
      ports:
        - "8081:8081"
      depends_on:
        - db
      environment:
        SPRING_DATASOURCE_URL: jdbc:mysql://db:3306/tododb
        SPRING_DATASOURCE_USERNAME: root
        SPRING_DATASOURCE_PASSWORD: P@ssw0rd
      networks:
        - todoapp_network
      entrypoint: sh -c "sleep 30 && java -jar /app.jar"
  networks:
    todoapp_network:
      driver: bridge

  volumes:
    db_data:
```

- **Services Section**: This section defines the different containers (services) that will be run as part of the Docker Compose setup.
- **image: mysql**: This specifies the Docker image to use. In this case, it uses the official MySQL image from Docker Hub.
- **container_name: mysql_db**: This gives the container a specific name (**mysql_db**).
- **depends_on**: Ensures that the **todoapp** service starts after the **db** service (MySQL) is up and running. However, this does not wait for the database to be fully initialized.
- **environment**: Defines the environment variables to configure MySQL:
  - **MYSQL_ROOT_PASSWORD**: Sets the root user password (**P@ssw0rd**).
  - **MYSQL_DATABASE**: Creates a database named **tododb**.
  - **MYSQL_PASSWORD**: Sets the password for the default MySQL user (root).
  - **SPRING_DATASOURCE_URL**: Defines the JDBC connection URL to the MySQL database (**db** is the hostname for the database within the Docker network).
  - **SPRING_DATASOURCE_USERNAME**: Sets the username to connect to the MySQL database (**root**).
  - **SPRING_DATASOURCE_PASSWORD**: Defines the password to connect to the MySQL database.
- **ports**: Exposes port **3306** (MySQL's default port) on the host system, allowing external connections to the MySQL database.
  - **"3306:3306"**: The format is **host_port:container_port**.
- **entrypoint: sh -c "sleep 30 && java -jar /app.jar"**: This introduces a delay of 30 seconds before starting the application, allowing MySQL to be ready.

- **networks**: Adds this service to the custom network **todoapp_network** so that the other services can communicate with it.
- **volumes**: Uses a Docker volume (**db_data**) to persist the database data so that it won't be lost when the container is restarted. The database files are stored in **/var/lib/mysql** inside the container.

Step-3 :: Run the Program

To start the multi-container application:

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

```
cd Docker
```

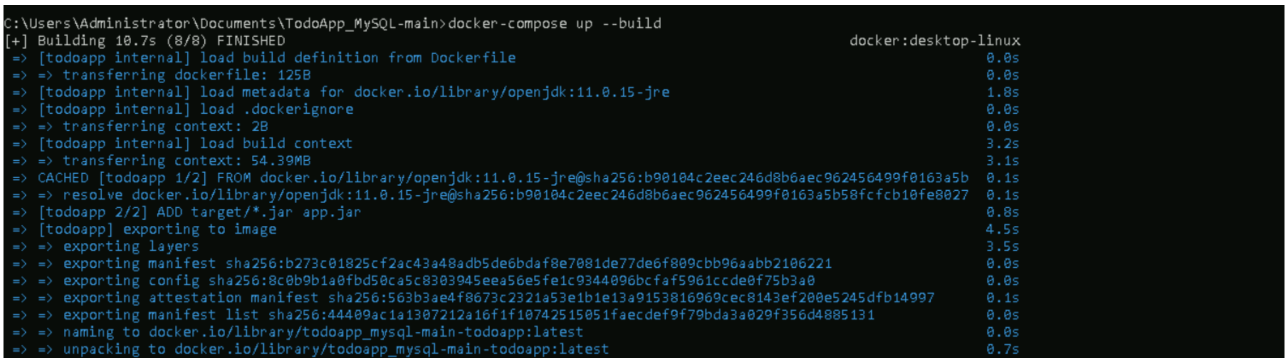
2. Run Docker Compose to build and start the containers:

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

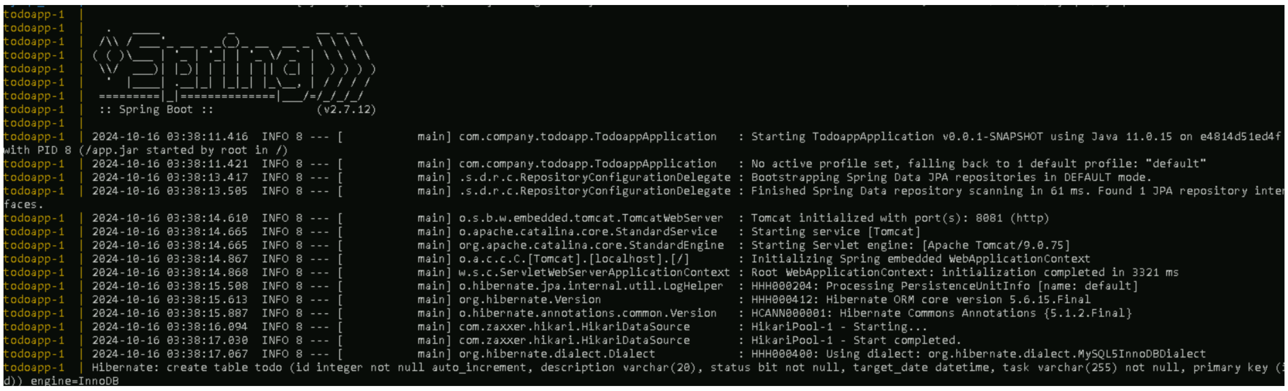
This command:

- **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.



```
C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose up --build
[+] Building 10.7s (8/8) FINISHED                                              docker:desktop-linux
 => [todoapp internal] load build definition from Dockerfile                                   0.0s
 => => transferring dockerfile: 125B                                                           0.0s
 => [todoapp internal] load metadata for docker.io/library/openjdk:11.0.15-jre                 1.8s
 => [todoapp internal] load .dockerignore                                                      0.0s
 => => transferring context: 2B                                                                0.0s
 => [todoapp internal] load build context                                                      3.2s
 => => transferring context: 54.39MB                                                           3.1s
 => CACHED [todoapp 1/2] FROM docker.io/library/openjdk:11.0.15-jre@sha256:b90104c2eec246d8b6aec962456499f0163a5b  0.1s
 => => resolve docker.io/library/openjdk:11.0.15-jre@sha256:b90104c2eec246d8b6aec962456499f0163a5b58fcfcb10fe8027  0.1s
 => [todoapp 2/2] ADD target/*.jar app.jar                                                     0.8s
 => [todoapp] exporting to image                                                               4.5s
 => => exporting layers                                                                        3.5s
 => => exporting manifest sha256:b273c01825cf2ac43a48adb5de6bdaf8e7081de77de6f809cbb96aabb2106221   0.0s
 => => exporting config sha256:8c0b9b1a0fbd50ca5c8303945eea56e5fe1c9344096bcfaf5961ccde0f75b3a0     0.0s
 => => exporting attestation manifest sha256:563b3ae4f8673c2321a53e1b1e13a9153816969cec8143ef200e5245dfb14997  0.1s
 => => exporting manifest list sha256:44409ac1a1307212a16f1f10742515051faecdef9f79bda3a029f356d4885131  0.0s
 => => naming to docker.io/library/todoapp_mysql-main-todoapp:latest                           0.0s
 => => unpacking to docker.io/library/todoapp_mysql-main-todoapp:latest                        0.7s
```



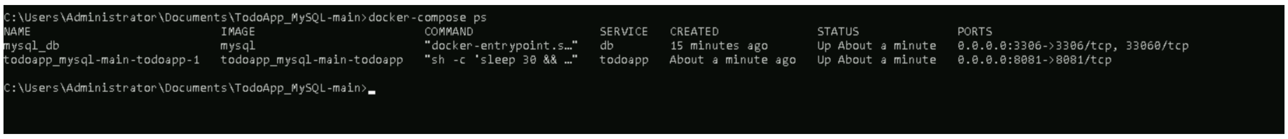
```
todoapp-1  |
todoapp-1  |   .   ____          _            __ _ _
todoapp-1  |  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
todoapp-1  | ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
todoapp-1  |  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
todoapp-1  |   '  |____| .__|_| |_|_| |_\__, | / / / /
todoapp-1  |  =========|_|==============|___/=/_/_/_/
todoapp-1  |  :: Spring Boot ::               (v2.7.12)
todoapp-1  |
todoapp-1  | 2024-10-16 03:38:11.416  INFO 8 --- [           main] com.company.todoapp.TodoappApplication   : Starting TodoappApplication v0.0.1-SNAPSHOT using Java 11.0.15 on e4814d51ed4f
with PID 8 (/app.jar started by root in /)
todoapp-1  | 2024-10-16 03:38:11.421  INFO 8 --- [           main] com.company.todoapp.TodoappApplication   : No active profile set, falling back to 1 default profile: "default"
todoapp-1  | 2024-10-16 03:38:13.417  INFO 8 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
todoapp-1  | 2024-10-16 03:38:13.505  INFO 8 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 61 ms. Found 1 JPA repository inter
faces.
todoapp-1  | 2024-10-16 03:38:14.610  INFO 8 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8081 (http)
todoapp-1  | 2024-10-16 03:38:14.665  INFO 8 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
todoapp-1  | 2024-10-16 03:38:14.665  INFO 8 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.75]
todoapp-1  | 2024-10-16 03:38:14.867  INFO 8 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
todoapp-1  | 2024-10-16 03:38:14.868  INFO 8 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 3321 ms
todoapp-1  | 2024-10-16 03:38:15.508  INFO 8 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
todoapp-1  | 2024-10-16 03:38:15.613  INFO 8 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.15.Final
todoapp-1  | 2024-10-16 03:38:15.887  INFO 8 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
todoapp-1  | 2024-10-16 03:38:16.094  INFO 8 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
todoapp-1  | 2024-10-16 03:38:17.030  INFO 8 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
todoapp-1  | 2024-10-16 03:38:17.067  INFO 8 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.MySQL5InnoDBDialect
todoapp-1  | Hibernate: create table todo (id integer not null auto_increment, description varchar(20), status bit not null, target_date datetime, task varchar(255) not null, primary key (i
d)) engine=InnoDB
```

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-4 :: Manage the Containers

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

   ```
   docker-compose ps
   ```

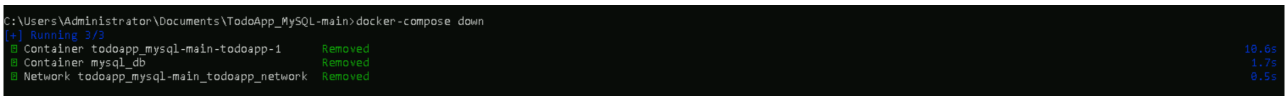
   ```
   C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose ps
   NAME                          IMAGE                          COMMAND                    SERVICE   CREATED              STATUS             PORTS
   mysql_db                      mysql                          "docker-entrypoint.s…"     db        15 minutes ago       Up About a minute  0.0.0.0:3306->3306/tcp, 33060/tcp
   todoapp_mysql-main-todoapp-1  todoapp_mysql-main-todoapp     "sh -c 'sleep 30 && …"     todoapp   About a minute ago   Up About a minute  0.0.0.0:8081->8081/tcp

   C:\Users\Administrator\Documents\TodoApp_MySQL-main>
   ```

   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
   ```

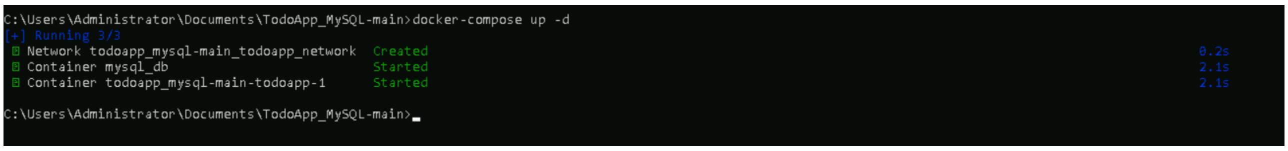
   

   ```
   C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose down
   [+] Running 3/3
    ✔ Container todoapp_mysql-main-todoapp-1      Removed                                   10.6s
    ✔ Container mysql_db                          Removed                                    1.7s
    ✔ Network todoapp_mysql-main_todoapp_network  Removed                                    0.5s
   ```

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:

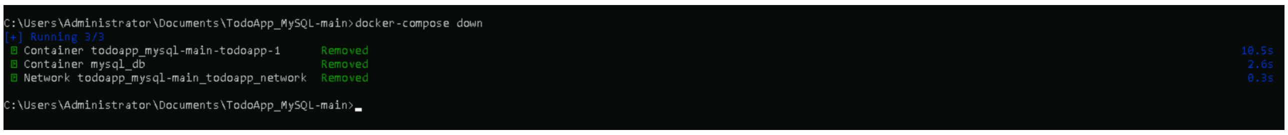
   ```
   docker-compose up -d
   ```

   ```
   C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose up -d
   [+] Running 3/3
    ✔ Network todoapp_mysql-main_todoapp_network  Created                                    0.2s
    ✔ Container mysql_db                          Started                                    2.1s
    ✔ Container todoapp_mysql-main-todoapp-1      Started                                    2.1s

   C:\Users\Administrator\Documents\TodoApp_MySQL-main>
   ```

   To stop the detached containers:

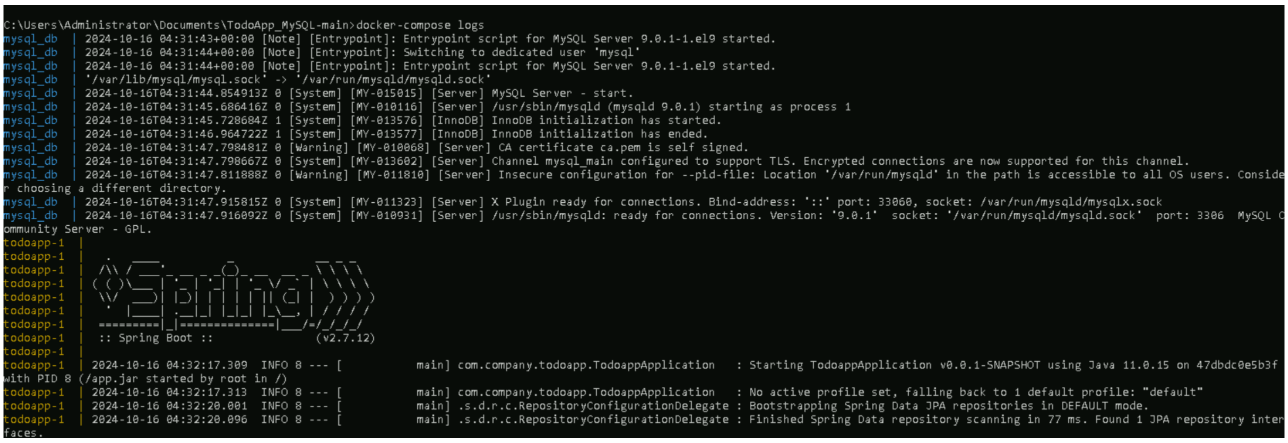
   ```
   docker-compose down
   ```



```
C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose down
[+] Running 3/3
 ✔ Container todoapp_mysql-main-todoapp-1      Removed                                                        10.5s
 ✔ Container mysql_db                          Removed                                                         2.6s
 ✔ Network todoapp_mysql-main_todoapp_network  Removed                                                         0.3s

C:\Users\Administrator\Documents\TodoApp_MySQL-main>
```

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

```
docker-compose logs
```



```
C:\Users\Administrator\Documents\TodoApp_MySQL-main>docker-compose logs
mysql_db  | 2024-10-16 04:31:43+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 9.0.1-1.el9 started.
mysql_db  | 2024-10-16 04:31:44+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
mysql_db  | 2024-10-16 04:31:44+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 9.0.1-1.el9 started.
mysql_db  | '/var/lib/mysql/mysql.sock' -> '/var/run/mysqld/mysqld.sock'
mysql_db  | 2024-10-16T04:31:44.854913Z 0 [System] [MY-015015] [Server] MySQL Server - start.
mysql_db  | 2024-10-16T04:31:45.686416Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 9.0.1) starting as process 1
mysql_db  | 2024-10-16T04:31:45.728684Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
mysql_db  | 2024-10-16T04:31:46.964722Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
mysql_db  | 2024-10-16T04:31:47.798481Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
mysql_db  | 2024-10-16T04:31:47.798667Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.
mysql_db  | 2024-10-16T04:31:47.811888Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file: Location '/var/run/mysqld' in the path is accessible to all OS users. Conside
r choosing a different directory.
mysql_db  | 2024-10-16T04:31:47.915815Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqlx.sock
mysql_db  | 2024-10-16T04:31:47.916092Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '9.0.1'  socket: '/var/run/mysqld/mysqld.sock'  port: 3306  MySQL C
ommunity Server - GPL.
todoapp-1  |
todoapp-1  |   .   ____          _            __ _ _
todoapp-1  |  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
todoapp-1  | ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
todoapp-1  |  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
todoapp-1  |   '  |____| .__|_| |_|_| |_\__, | / / / /
todoapp-1  |  =========|_|==============|___/=/_/_/_/
todoapp-1  |  :: Spring Boot ::               (v2.7.12)
todoapp-1  |
todoapp-1  | 2024-10-16 04:32:17.309  INFO 8 --- [           main] com.company.todoapp.TodoappApplication   : Starting TodoappApplication v0.0.1-SNAPSHOT using Java 11.0.15 on 47dbdc0e5b3f
with PID 8 (/app.jar started by root in /)
todoapp-1  | 2024-10-16 04:32:17.313  INFO 8 --- [           main] com.company.todoapp.TodoappApplication   : No active profile set, falling back to 1 default profile: "default"
todoapp-1  | 2024-10-16 04:32:20.001  INFO 8 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
todoapp-1  | 2024-10-16 04:32:20.096  INFO 8 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 77 ms. Found 1 JPA repository inter
faces.
```

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