

# Persist Data Across Container Restarts Using Docker Volumes

---

## Table of Contents

---

- [Description](#)
- [Problem Statement](#)
- [Prerequisites](#)
  - [Software Requirement](#)
  - [Hardware Requirement](#)
- [Implementation Steps](#)
  - [Step-1: Create a Docker Volume](#)
    - [Step-2: Run MySQL Container with Volume](#)
    - [Step-3: Run the TodoApp Container](#)
    - [Step-4: Test Data Persistence](#)
- [References](#)

## Description

---

This section outlines how to use Docker volumes to persist data in your TodoApp across container restarts. Volumes provide a way to store data independently from the container's lifecycle, ensuring that data remains intact even if the container is stopped or removed.

## Problem Statement

---

By default, any data stored inside a Docker container is ephemeral and will be lost when the container is removed. To ensure that user data (e.g., tasks, user information) persists between container restarts or re-creations, we will use Docker volumes.

## Prerequisites

---

Completion of all previous lab guides (up to Lab Guide-04) is required before proceeding with Lab Guide-05.

### Software Requirement

- **Docker Desktop:** Installed and running on your Windows system.
- **Java JDK 11 or higher:** For building the Java-based TodoApp.
- **MySQL Docker Image:** Official MySQL image pulled from Docker Hub.
- **TodoApp Docker Image:** Make sure [Docker image](#) is present for todoapp.
- **TodoAPP\_MYSQL:** To download the source folder [click here](#)

### Hardware Requirement

- **CPU:** 64-bit processor with virtualization support.
- **RAM:** 4 GB minimum (8 GB recommended).
- **Disk Space:** 1 GB or more for Docker images and containers.

## Implementation Steps

---

### Step-1: Create a Docker Volume

First, we need to create a Docker volume that will hold the data for our MySQL database.

```
docker volume create mysql-data
```

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker volume create mysql-data
mysql-data
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

### Step-2: Run MySQL Container with Volume

Next, we will run the MySQL container using the volume we created. This will ensure that the database data is persisted.

```
docker run -d -p3307:3306 -v mysql-data:/var/lib/mysql --network=todoapp_network -e MYSQL_ROOT_PASSWORD=P@ssw0rd -e MYSQL_DATABASE=tododb --name=mysqlldb mysql
```

- **-v todoapp\_data:/var/lib/mysql:** This flag mounts the **todoapp\_data** volume to the **/var/lib/mysql** directory in the MySQL container, where MySQL stores its data files.

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker run -d -p3307:3306 -v mysql-data:/var/lib/mysql --network=todoapp_network -e MYSQL_ROOT_PASSWORD=P@ssw0rd -e MYSQL_DATABASE=tododb --name=mysqlldb mysql
886a480db2c2415a5c489661a747bccdc37fee9e7a88a6e69197ca8f198b7806
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

### Step-3: Run the TodoApp Container

After setting up the MySQL container with a volume, you can run your TodoApp container. Make sure it connects to the MySQL database correctly.

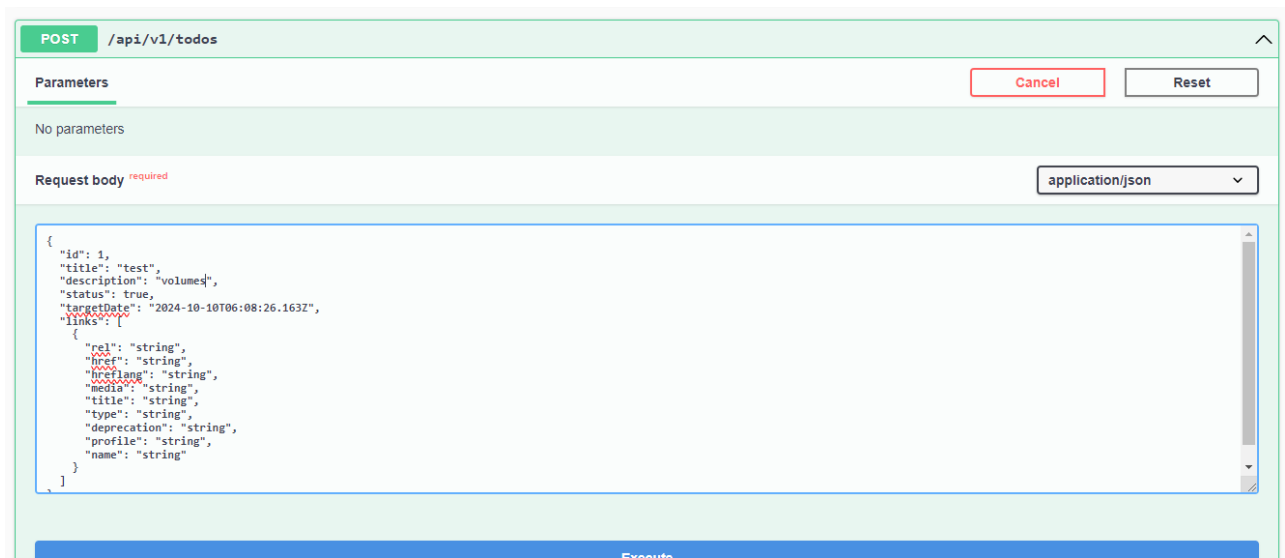
```
docker run -d -p8081:8081 --name todoapp --network=todoapp_network -e MYSQL_HOST=mysqlldb todoapp
```

- Ensure that your TodoApp is configured to connect to the MySQL database using the appropriate credentials and host settings.

```
C:\Users\Administrator\Documents\ToDoApp_MySQL-main\todoapp>docker run -d -p8081:8081 --name todoapp --network=todoapp_network -e MYSQL_HOST=mysqlldb todoapp
aab2bc1b52da9e5dcd7d91e87a5516c2e828e0b3574f81b26682903fc1725455
C:\Users\Administrator\Documents\ToDoApp_MySQL-main\todoapp>
```

## Step-4: Test Data Persistence

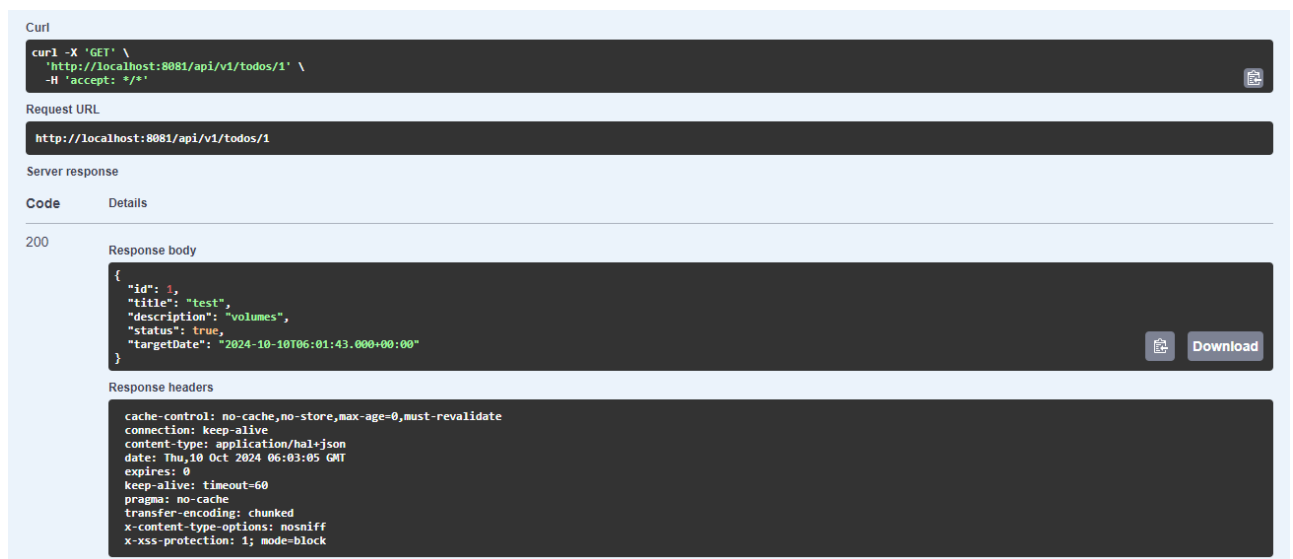
- Open your ToDoApp in a web browser (e.g., <http://localhost:8081/swagger-ui/index.html>).
- Add some tasks to the ToDoApp
  - Using the POST and GET method in **todo-controller**.



The image shows the Swagger UI for the POST endpoint `/api/v1/todos`. The 'Parameters' section is empty. The 'Request body' section is set to 'application/json' and contains a JSON object: 

```
{  "id": 1,  "title": "test",  "description": "volumes",  "status": true,  "targetDate": "2024-10-10T06:08:26.163Z",  "links": [    {      "rel": "string",      "href": "string",      "hreflang": "string",      "media": "string",      "title": "string",      "type": "string",      "deprecation": "string",      "profile": "string",      "name": "string"    }  ]}
```

 The 'Execute' button is at the bottom.



The image shows a curl command and its server response. The curl command is: 

```
curl -X 'GET' \  'http://localhost:8081/api/v1/todos/1' \  -H 'accept: */*'
```

 The request URL is `http://localhost:8081/api/v1/todos/1`. The server response is a 200 status code with the following response body: 

```
{  "id": 1,  "title": "test",  "description": "volumes",  "status": true,  "targetDate": "2024-10-10T06:01:43.000+00:00"}
```

 The response headers are: 

```
cache-control: no-cache,no-store,max-age=0,must-revalidate  connection: keep-alive  content-type: application/hal+json  date: Thu,10 Oct 2024 06:03:05 GMT  expires: 0  keep-alives: timeout=60  pragma: no-cache  transfer-encoding: chunked  x-content-type-options: nosniff  x-xss-protection: 1; mode=block
```

- Stop the MySQL and ToDoApp containers:

```
docker stop mysqlldb
docker stop todoapp
```

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker stop mysqldb
mysqldb

C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker stop todoapp
todoapp

C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

**Note:** You can also delete the containers but make sure that the volumes are added to the running containers

- Start the containers again:

```
docker start mysqldb
docker start todoapp
```

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker start mysqldb
mysqldb

C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

```
C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>docker start todoapp
todoapp

C:\Users\Administrator\Downloads\ToDoApp_MySQL-main>_
```

- Reopen your ToDoApp in the web browser. You should see that the previously added tasks are still present, confirming that the data has been persisted.

## References

---

Refer to Docker's official documentation for more details on Docker volumes:

- [Docker Volumes Documentation](#)