Scenario 4 – Issue with data persistence

You are a DevOps engineer responsible for containerizing services for a startup. The company has multiple databases running on their servers, and you are tasked to make sure the data from the MySQL instance persisted even if the containers are stopped or removed. You've been given a containerized MySQL instance that has some important data that shouldn't be lost. Your task is to migrate that data to a Docker volume and make sure it persists across container restarts.

Requirements

Your goal is to migrate the data currently stored in a running MySQL container to an external Docker volume. This will help the company ensure data persistence and avoid data loss during container redeployments.

Acceptance Criteria

- The MySQL database should persist data even after the container is removed and recreated.
- You should be able to create a new table, add data, and verify that the data remains intact after restarting the container with the Docker volume attached.
- Users should verify data persistence by restarting the container, creating tables, adding rows, and confirming the same data remains intact.

Resources

- Docker Volumes Documentation
- MySQL Docker Official Documentation
- Docker Image: tdevsin/company-mysql-db

Hints

Testing Data Persistence

Step 1: Start by running the provided MySQL container without using a volume:

```
docker run --name company-db -e MYSQL_ROOT_PASSWORD=root -d tdevsin/company-mysql-db
```

Step 2: Connect to the MySQL instance via command line:

```
docker exec -it company-db mysql -u root -p
```

Step 3: Execute MySQL commands to create a table and add data:

```
USE company_db;
CREATE TABLE test_table (
  id INT PRIMARY KEY AUTO_INCREMENT,
  description VARCHAR(255) NOT NULL
);

INSERT INTO test_table (description) VALUES ('Test entry 1'), ('Test entry 2'), ('Test entry 3');
```

Step 4: Use a SELECT statement to check if the data is inserted:

```
SELECT * FROM test_table;
```

Step 5: Stop and start the container again:

```
docker stop company-db
docker rm company-db
docker run --name company-db -e MYSQL_ROOT_PASSWORD=root -d tdevsin/company-mysql-db
```

Step 6: Connect to the MySQL instance again and check if the data persisted. You will notice that the data is lost, demonstrating the need for a volume.

Attach Docker Volume

- Create a Docker volume to store the MySQL data directory (/var/lib/mysql).
- Re-run the container with the Docker volume attached.
- Repeat the steps to create tables and insert rows, and then verify whether data persists after stopping and restarting the container.

Inspecting Data Inside Container

- Use the docker exec command to inspect data inside the MySQL container.
- Understand how Docker volumes link to directories inside the container and ensure the data is being stored externally.