**Dockerized Web Server with MySQL Integration: A Step-by-Step Guide**

This guide outlines the process of setting up a web server with a MySQL database in Docker, using Flask for the web application. The data entered through the web form is stored in the MySQL database.

**1. Creating a Docker Network**

We start by creating a custom Docker network to ensure that both the web server and MySQL database containers can communicate with each other.

docker network create project1

* docker network create project1: This command creates a new Docker network named project1. Docker containers running in the same network can communicate with each other using container names as hostnames.

**2. Running MySQL Container**

Next, we run a MySQL container with the specified root password and the database name where data will be stored.

docker run --name sqlserver --network project1 -e MYSQL\_ROOT\_PASSWORD=password -e MYSQL\_DATABASE=mydb -d mysql:latest

* --name sqlserver: Specifies the name of the container as sqlserver.
* --network project1: Connects the container to the project1 network so it can communicate with the web server.
* -e MYSQL\_ROOT\_PASSWORD=password: Sets the root password of the MySQL database to password.
* -e MYSQL\_DATABASE=mydb: Creates a new database named mydb for storing form data.
* -d mysql:latest: Runs the container in detached mode using the latest version of the MySQL image.

**3. Setting Up the Web Server**

Create a folder called web-server to store all the necessary scripts for the web server.

mkdir web-server && cd web-server

* mkdir web-server && cd web-server: Creates a folder named web-server and navigates into it. This folder will contain the necessary files for your web server.

**4. Creating the Web Page and Flask Application**

Inside the web-server directory, create the index.html and app.py files.

**index.html**: This is the HTML page that will contain the form where users can submit their data.

html

* This file will contain the structure of the web page, including a form to gather input from users.

**app.py**: This Python script will handle the backend logic of the application, including connecting to the MySQL database, receiving data from the HTML form, and inserting it into the database.

python

* This file will contain the Flask application that listens for requests, processes form data, and stores it in the MySQL database.

**5. Creating the Dockerfile**

Create a Dockerfile in the same folder (web-server) to automate the building and running of the Flask web server.

dockerfile

* The Dockerfile will include the necessary instructions to set up the environment for Flask, install dependencies, and run the app.

**6. Building the Docker Image**

Build the Docker image for the web server from the web-server directory.

docker build -t web-server .

* docker build -t web-server .: This command builds the Docker image for the web server using the current directory (.) where the Dockerfile is located. The -t flag is used to tag the image with the name web-server.

**7. Running the Web Server Container**

Run the web-server container and link it to the previously created project1 network, while exposing port 5000 for local access.

docker run --name web-server --network project1 -p 5000:5000 -d web-server

* --name web-server: Names the container as web-server.
* --network project1: Connects the container to the project1 network, enabling communication with the MySQL container.
* -p 5000:5000: Exposes port 5000 on the container and maps it to port 5000 on the host machine.
* -d web-server: Runs the web server container in detached mode.

**8. Checking the Web Server**

To verify that the web server is running correctly, open your browser and go to:

http://localhost:5000

* This URL should display the web form you created in index.html. You can enter data into the form, and it will be submitted to the MySQL database.

**9. Checking Submitted Data in MySQL Database**

After submitting the form data, check the MySQL container to ensure that the data was inserted correctly into the database.

1. Access the MySQL container:

docker exec -it sql-server bash

1. Log in to MySQL:

mysql -u root -p

Enter the password password when prompted.

1. Select the database:

USE mydb;

1. Show the tables in the database:

SHOW TABLES;

1. View the data in the relevant table (e.g., entries):

SELECT \* FROM entries;

**Conclusion**

This guide covers the complete process of setting up a Flask web server with a MySQL database in Docker. By following the steps, you can easily create a web app that submits data to a MySQL database. All of the code and necessary configurations are documented here to ensure reproducibility.

Feel free to modify the code and expand upon this project to fit your needs!