48. Create a simple web application using LAMP Stack on docker container.

Here's a comprehensive guide to building a simple LAMP stack-based web application using Docker containers. This walkthrough addresses each step from setup to common issues and fixes based on the errors you've encountered.

Step 1: Install Docker and Docker Compose

If Docker and Docker Compose aren't installed, start with the following commands:

bash

Copy code

```
# Update the package index

sudo apt update


# Install Docker

sudo apt install docker.io -y


# Start and enable Docker

sudo systemctl start docker

sudo systemctl enable docker


# Install Docker Compose

sudo apt install docker-compose -y
```

Verify installation with:

bash

Copy code

```
docker --version

docker-compose --version
```

Step 2: Set Up Project Structure

Create a project directory, for example, `my-lamp-app`, and navigate to it:

bash

Copy code

```
mkdir my-lamp-app

cd my-lamp-app
```

Inside this directory, create the following structure:

bash

Copy code

```
# Create necessary directories and files

mkdir src

touch src/index.php

touch docker-compose.yml

touch src/style.css
```

Step 3: Configure `docker-compose.yml`

Here's a basic `docker-compose.yml` file for a LAMP stack setup:

yaml

Copy code

```
version: '3.8'
services:
  apache:
    image: php:8.0-apache
    container_name: apache
    ports:
      - "8081:80"
    volumes:
      - ./src:/var/www/html
    networks:
      - lamp_network
    depends_on:
      - mysql
    environment:
      - MYSQL_ALLOW_EMPTY_PASSWORD=yes
    command: /bin/bash -c "apt-get update && docker-php-ext-install mysqli && apache2-foreground"

  mysql:
    image: mysql:5.7
    container_name: mysql
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: mydb
    networks:
      - lamp_network

networks:
  lamp_network:
    driver: bridge
```

Step 4: Add Sample PHP Code

Edit `src/index.php` with a basic PHP script to connect to MySQL and display user data:

php

Copy code

```
<?php
$servername = "mysql";  // Container name of MySQL in docker-compose
$username = "root";
$password = "rootpassword";
$dbname = "mydb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
```

```
if ($conn->connect_error) {

    die("<div class='error'>Connection failed: " .
$conn->connect_error . "</div>");

}


echo "<h1>Welcome to My PHP Web App</h1>";

echo "<p>Connected successfully to MySQL Database</p>";


// Fetch data from the database

$sql = "SELECT id, name, email FROM users";

$result = $conn->query($sql);

?>


<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width,
initial-scale=1.0">

    <title>PHP Web Application with MySQL</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>
```

```
<div class="container">
    <section>
        <h2>Users List</h2>
        <?php if ($result && $result->num_rows > 0): ?>
            <table>
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Name</th>
                        <th>Email</th>
                    </tr>
                </thead>
                <tbody>
                    <?php while($row = $result->fetch_assoc()): ?>
                        <tr>
                            <td><?php echo $row['id']; ?></td>
                            <td><?php echo $row['name']; ?></td>
                            <td><?php echo $row['email']; ?></td>
                        </tr>
                    <?php endwhile; ?>
                </tbody>
            </table>
        <?php else: ?>
```

```
                <p>No users found.</p>
            <?php endif; ?>
        </section>
    </div>
</body>
</html>

<?php
$conn->close();
?>
```

Step 5: Add Basic CSS

In `src/style.css`, add some styles:

css

Copy code

```
body {
    font-family: Arial, sans-serif;
}
.container {
    width: 80%;
    margin: auto;
    padding: 20px;
}
```

```
table {

    width: 100%;

    border-collapse: collapse;

}

table, th, td {

    border: 1px solid black;

    padding: 8px;

    text-align: left;

}

th {

    background-color: #f2f2f2;

}
```

Step 6: Launch the Docker Containers

Run Docker Compose to build and start the containers:

bash

Copy code

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

To check the status of your containers, use:

bash

Copy code

```
sudo docker-compose ps
```

Step 7: Access the Application

Open a browser and go to `http://localhost:8081`. You should see the "Welcome to My PHP Web App" message with an indication of the database connection status.

Step 8: Initialize the MySQL Database

Access the MySQL container:
bash
Copy code

```
sudo docker exec -it mysql mysql -u root -p
```

1.

Enter the MySQL root password (`rootpassword`), then create a sample table:
sql
Copy code

```
USE mydb;



CREATE TABLE users (

    id INT AUTO_INCREMENT PRIMARY KEY,

    name VARCHAR(100) NOT NULL,

    email VARCHAR(100) NOT NULL

);



INSERT INTO users (name, email) VALUES ('John Doe',
'john@example.com');

INSERT INTO users (name, email) VALUES ('Jane Smith',
'jane@example.com');
```

2.

Step 9: Troubleshooting Common Issues

- Connection Refused on Port 8081: Check if the `8081` port is in use by another service, or try reconfiguring the `docker-compose.yml` file to use a different port.
- MySQL Permissions Error: Restart the MySQL container with `sudo docker-compose restart mysql` and ensure correct credentials.
- Warning: Attempt to read property "num_rows" on bool: This warning appears when the query fails. Ensure the `users` table exists, and check the query syntax.

Step 10: Stop and Clean Up Containers

To stop the containers:

bash

Copy code

```
sudo docker-compose down
```

To remove images and free up space:

bash

Copy code

```
sudo docker system prune -a
```

Additional Enhancements

- Add an HTML form to insert new users directly.
- Add pagination for large data sets.
- Separate PHP and HTML code for maintainability.

With these steps, you should have a functioning LAMP stack web application using Docker containers. Let me know if you need any additional help!