Here's a step-by-step guide to create a simple web form with a MySQL database using Docker, Node.js, and JavaScript:

### Step 1: Set Up Docker Containers for MySQL and Nginx

#### 1.1. Create MySQL Container

We will create a MySQL container with the necessary environment variables and bind the ports.

```bash

docker run --name mysql\_mg -e MYSQL\_ROOT\_PASSWORD=your\_password -p 3306:3306 -d mysql

```

- `--name mysql\_mg`: Names the container `mysql\_mg`.

- `-e MYSQL\_ROOT\_PASSWORD=your\_password`: Sets the root password for MySQL.

- `-p 3306:3306`: Maps the host machine's port `3306` to the container's port `3306`.

- `-d mysql`: Runs the MySQL container in detached mode.

#### 1.2. Create Nginx Container (Optional for serving frontend)

To serve your HTML form with Nginx, run the Nginx container:

```bash

docker run --name nginx\_server -p 8080:80 -d nginx

```

This will serve content on `localhost:8080`.

### Step 2: Set Up Node.js Backend with MySQL Database

#### 2.1. Initialize Node.js Project

1. Create a new directory for your project:

```bash

mkdir mysql\_form

cd mysql\_form

```

2. Initialize a new Node.js project:

```bash

npm init -y

```

3. Install necessary dependencies:

```bash

npm install express mysql2

```

#### 2.2. Create Server (Backend)

Create a file `server.js`:

```const express = require('express');

const mysql = require('mysql2');

const bodyParser = require('body-parser');

const app = express();

const port = 3000;

// Middleware to parse JSON and URL-encoded data

app.use(express.json());

app.use(bodyParser.urlencoded({ extended: true }));

app.use(express.static('public'));

// MySQL setup

const db = mysql.createConnection({

host: 'localhost',

user: 'root',

password: 'your\_password',

database: 'mydatabase'

});

db.connect(err => {

if (err) {

console.error('Database connection failed:', err);

return;

}

console.log('Connected to MySQL database.');

});

// Route to handle form submission

app.post('/submit', (req, res) => {

const { name, email } = req.body;

if (!name || !email) {

return res.send('Name and email are required!');

}

const query = 'INSERT INTO users (name, email) VALUES (?, ?)';

db.query(query, [name, email], (err, result) => {

if (err) {

console.error('Error inserting record:', err);

return res.send('Error in inserting record.');

}

res.send('Record inserted successfully!');

});

});

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

```

#### 2.3. Create MySQL Database and Table

To store form data, you need a database and table. Connect to MySQL via CLI or any MySQL client and run the following commands:

```sql

CREATE DATABASE mydatabase;

USE mydatabase;

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL

);

```

#### 2.4. Run the Backend Server

Run your Node.js server:

```bash

node server.js

```

Now your backend server is running at `http://localhost:3000`.

### Step 3: Create Frontend Form (HTML)

Create a folder `public` in your project directory and inside it create `index.html` for the form:

```<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Insert Record</title>

</head>

<body>

<h1>Insert Record into MySQL Database</h1>

<form id="insertForm">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required>

<br><br>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<br><br>

<button type="submit">Submit</button>

</form>

<p id="resultMessage"></p>

<script>

document.getElementById('insertForm').addEventListener('submit', function(event) {

event.preventDefault();

const name = document.getElementById('name').value;

const email = document.getElementById('email').value;

fetch('/submit', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({ name, email })

})

.then(response => response.text())

.then(data => {

document.getElementById('resultMessage').textContent = data;

document.getElementById('insertForm').reset();

})

.catch(error => {

document.getElementById('resultMessage').textContent = 'An error occurred';

console.error('Error:', error);

});

});

</script>

</body>

</html>

```

Now, the form is ready to take input from the user and send it to the backend for insertion into the MySQL database.

### Step 4: Connect Frontend and Backend

- The form sends data to the backend at the `/submit` route, which will insert the record into MySQL.

- The backend uses the MySQL credentials to insert the form data into the `users` table in `testdb`.

### Step 5: Access the Application

1. Open the `index.html` form by navigating to `http://localhost:8080` (if using Nginx).

2. Fill in the form and submit it.

3. After submission, the backend should insert the data into the MySQL database.

You can verify the insertion by querying the `users` table in MySQL.

### Troubleshooting

1. \*\*If MySQL is not connecting\*\*: Ensure the MySQL container is running properly and that the host, username, password, and database name are correct in the backend configuration.

2. \*\*Database Errors\*\*: Double-check the MySQL queries and ensure the database and tables exist.

This should complete your project setup. Let me know if you encounter any issues!