Creating a web form to insert records into a MySQL database using Python involves using a web framework. Flask is a popular and lightweight web framework for Python. Here's an example using Flask:

1. Install Flask and MySQL Connector:

pip install Flask

pip install mysql-connector-python

2. Create a Flask App:

Create a file named app.py with the following content:

#python

from flask import Flask, render\_template, request, redirect

import mysql.connector

app = Flask(name)

# MySQL Configuration

db = mysql.connector.connect(

host="localhost",

user="myuser",

password="mypassword",

database="mydatabase"

)

cursor = db.cursor()

# Home route with the form

@app.route('/')

def home():

return render\_template('index.html')

# Form submission route

@app.route('/submit', methods=['POST'])

def submit():

if request.method == 'POST':

name = request.form['name']

email = request.form['email']

# Insert data into MySQL database

sql = "INSERT INTO users (name, email) VALUES (%s, %s)"

values = (name, email)

cursor.execute(sql, values)

db.commit()

return "Record added successfully"

if name == 'main':

app.run(debug=True)

3. Create HTML Templates:

Create a folder named templates in the same directory as your app.py file. Inside the templates folder, create a file named index.html with the following content:

html

Copy code

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Simple Form</title>

</head>

<body>

<h2>Simple Form</h2>

<form action="{{ url\_for('submit') }}" method="post">

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

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

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

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

<input type="submit" value="Submit">

</form>

</body>

</html>

4. Run the Flask App:

Open a terminal in the same directory as your app.py file and run the following command:

bash

Copy code

python app.py

Visit http://localhost:5000 in your web browser to access the simple form. Enter the name and email, click "Submit," and the data will be inserted into the MySQL database.

Notes:

Replace the placeholder credentials (myuser, mypassword, etc.) with your MySQL database credentials.

This example uses the mysql-connector-python library for database connectivity. Adjustments may be needed based on your specific database library.

In a production environment, consider using environment variables for sensitive information.

This is a basic example and doesn't include proper security measures or validation. Always sanitize user input and use secure practices in a production environment.

To Dockerize the Flask application with MySQL, you'll need to create a Dockerfile and a docker-compose.yml file. Here's a basic example:

1. Create a Dockerfile:

Create a file named Dockerfile in the same directory as your app.py file with the following content:

Dockerfile

Copy code

# Use the official Python image

FROM python:3.9

# Set environment variables

ENV PYTHONDONTWRITEBYTECODE 1

ENV PYTHONUNBUFFERED 1

# Create and set the working directory

WORKDIR /app

# Install dependencies

COPY requirements.txt /app/

RUN pip install --upgrade pip

RUN pip install -r requirements.txt

# Copy the application files to the container

COPY . /app/

# Expose the application port

EXPOSE 5000

# Run the application

CMD ["python", "app.py"]

2. Create a docker-compose.yml File:

Create a file named docker-compose.yml in the same directory as your app.py file with the following content:

yaml

Copy code

version: '3'

services:

db:

image: mysql:latest

environment:

MYSQL\_ROOT\_PASSWORD: root\_password

MYSQL\_DATABASE: mydatabase

MYSQL\_USER: myuser

MYSQL\_PASSWORD: mypassword

web:

build:

context: .

ports:

- "5000:5000"

depends\_on:

- db

3. Create a requirements.txt File:

Create a file named requirements.txt in the same directory as your app.py file with the following content:

plaintext

Copy code

Flask==2.0.1

mysql-connector-python==8.0.26

4. Build and Run with Docker Compose:

Open a terminal in the same directory as your app.py file and run the following commands:

bash

Copy code

docker-compose up --build

This will build the Docker image and start the services defined in the docker-compose.yml file.

Visit http://localhost:5000 in your web browser to access the simple form. The application is now running inside a Docker container.

Notes:

Adjust MySQL and Flask configurations in your app.py file if necessary.

Replace the placeholder credentials (myuser, mypassword, etc.) with your MySQL database credentials.

This is a basic setup for demonstration purposes. In production, consider using more secure practices, such as using environment variables for sensitive information.

Ensure Docker is installed on your machine before running the commands.

This example does not include SSL or other security considerations. In a production environment, you should use HTTPS and other security best practices.