Week 6 Tutorial 2 (Date: Oct 10, 2024)

How to Build a Web App using Flask and SQLite in Python

Reference link: https://www.geeksforgeeks.org/how-to-build-a-web-app-using-flask-and-sqlite-in-python/

Python-based Flask is a microweb framework. Typically, a micro-framework has little to no dependencies on outside frameworks. Despite being a micro framework, practically everything may be developed when and as needed utilizing Python libraries and other dependencies. In this post, we'll develop a Flask application that collects user input in a form and shows it on an additional web page using SQLite in Python.

Package Required

Install flask to proceed with the Front End of the Web App.

```
pip install flask

pip install db-sqlite3
```

Steps to Build an App Using Flask and SQLite

Step 1: Create Virtual Environment

Step 2: Install the required modules inside Virtual Environment.

Step 3: Build a Front End of the Web App.

- **index.html**

The **index.html** file will contain two buttons, one button to check all the participant's lists (taken from the database). And the other button to create a new entry.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Flask and SQLite </title>
    </head>
    <body>
        <h1>Build Web App Using Flask and SQLite</h1>
        <button class="btn" type="button" onclick="window.location.href='{{ url_for('join') }}';">Fill form to get updates</button><br/>
        <button class="btn" type="button" onclick="window.location.href='{{ url_for('participants') }}';">Check participant list</button>
    </body>
</html>
```

- **join.html**

In the **join.html**, create a simple form that takes Name, Email, City, Country and Phone as the input to store in the database. By the POST method, receive the form request of all the columns and commit the changes in the database after inserting the details in the table.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Flask and SQLite </title>
    </head>
    <body>
        <form method="POST">
            <label>Enter Name:</label>
            <input type="name" name="name" placeholder="Enter your name" required><br/>
            <label>Enter Email:</label>
            <input type="email" name="email" placeholder="Enter your email" required><br/>
            <label>Enter City:</label>
            <input type="name" name="city" placeholder="Enter your City name" required><br/>
            <label>Enter Country:</label>
            <input type="name" name="country" placeholder="Enter the Country name" required><br/>
            <label>Enter phone num:</label>
            <input type="name" name="phone" placeholder="Your Phone Number" required><br/>
            <input type = "submit" value = "submit"/><br/>
        </form>
    </body>
</html>
```

- **participants.html**

Use table tag and assign the heading using <th> tag. To auto increment, the table row on the new entry, use a For loop jinja template. Inside For loop add <tr> and <td> tags.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Flask and SQLite </title>
    </head>
    <style>
        table, th, td {
          border:1px solid black;
        }
        </style>
    <body>
        <table style="width:100%">
            <tr>
              <th>Name</th>
              <th>Email</th>
              <th>City</th>
              <th>Country</th>
              <th>Phone Number</th>
            </tr>
            {%for participant in data%}
               <tr>
                 <td>{{participant[0]}}</td>
                 <td>{{participant[1]}}</td>
                 <td>{{participant[2]}}</td>
                 <td>{{participant[3]}}</td>
                 <td>{{participant[4]}}</td>
                </tr>
            {%endfor%}
          </table>
    </body>
</html>
```

Step 4: Create **app.py**

Create a new file named app.py and build a Front End of the Web App by rendering HTML templates. From here we shall go function by function explanation as in points:

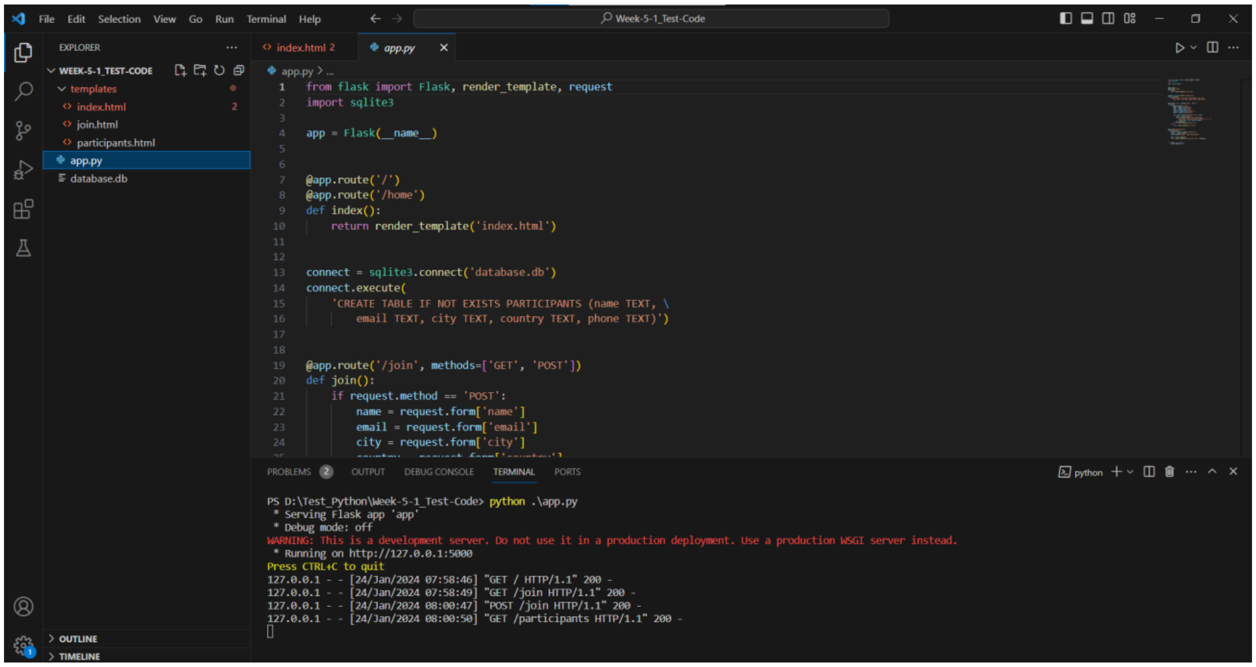
- To insert the data into the database, we first need to create a new database table. The column to be inserted in the database is Name, Email, City, Country, and Phone Number.

- The basic syntax to start with **sqlite3** is to first connect to the database. **sqlite3.connect**(“**database.db**”) will create a new database. The next step is to create a new table, but it will first check if the table already exists or not.
- One button in the **index.html** prompts to the participant’s list, and thus using the existing database select * from the table and display it using a Python template i.e., Jinja template to run through the loop within HTML. In the following code, we have created a table tag, inside the table tag for every new insertion in the database, we add a Loop Jinja Template to auto increment the new table row.
- In the **participants** function, we use select all columns from the table name, we use **fetchall()** method you retrieve the data.

```
from flask import Flask, render_template, request
import sqlite3

app = Flask(__name__)


@app.route('/')
@app.route('/home')
def index():
    return render_template('index.html')


connect = sqlite3.connect('database.db')
connect.execute(
    'CREATE TABLE IF NOT EXISTS PARTICIPANTS (name TEXT, \
    email TEXT, city TEXT, country TEXT, phone TEXT)')


@app.route('/join', methods=['GET', 'POST'])
def join():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        city = request.form['city']
        country = request.form['country']
        phone = request.form['phone']

        with sqlite3.connect("database.db") as users:
            cursor = users.cursor()
            cursor.execute("INSERT INTO PARTICIPANTS \
            (name,email,city,country,phone) VALUES (?,?,?,?,?)",
                           (name, email, city, country, phone))
            users.commit()
        return render_template("index.html")
    else:
        return render_template('join.html')


@app.route('/participants')
def participants():
    connect = sqlite3.connect('database.db')
    cursor = connect.cursor()
    cursor.execute('SELECT * FROM PARTICIPANTS')

    data = cursor.fetchall()
    return render_template("participants.html", data=data)
```

```
if __name__ == '__main__':
    app.run(debug=False)
```

Note: Structure of Files and Folders:

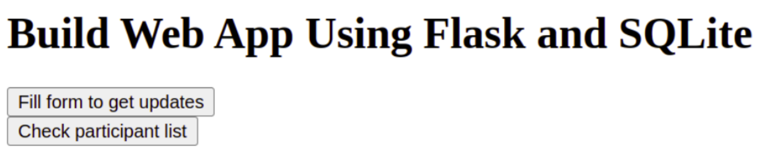


```
WEEK-5-1_TEST-CODE
  templates
    index.html
    join.html
    participants.html
  app.py
  database.db
```

```
from flask import Flask, render_template, request
import sqlite3

app = Flask(__name__)


@app.route('/')
@app.route('/home')
def index():
    return render_template('index.html')


connect = sqlite3.connect('database.db')
connect.execute(
    'CREATE TABLE IF NOT EXISTS PARTICIPANTS (name TEXT, \
        email TEXT, city TEXT, country TEXT, phone TEXT)')


@app.route('/join', methods=['GET', 'POST'])
def join():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        city = request.form['city']
```

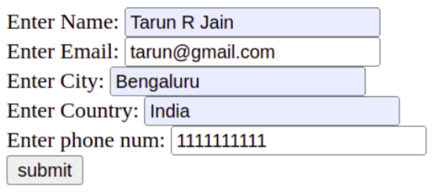
```
PS D:\Test_Python\Week-5-1_Test-Code> python .\app.py
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [24/Jan/2024 07:58:46] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Jan/2024 07:58:49] "GET /join HTTP/1.1" 200 -
127.0.0.1 - - [24/Jan/2024 08:00:47] "POST /join HTTP/1.1" 200 -
127.0.0.1 - - [24/Jan/2024 08:00:50] "GET /participants HTTP/1.1" 200 -
```

Output:
For route: http://127.0.0.1:5000/



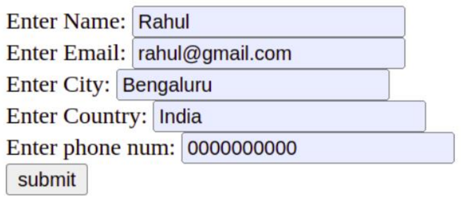
Build Web App Using Flask and SQLite

For route: http://127.0.0.1:5000/join
Here we are adding two new data to the database.



Enter Name:	Tarun R Jain
Enter Email:	tarun@gmail.com
Enter City:	Bengaluru
Enter Country:	India
Enter phone num:	1111111111

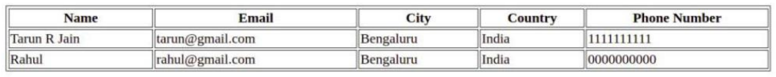
data 1



Enter Name:	Rahul
Enter Email:	rahul@gmail.com
Enter City:	Bengaluru
Enter Country:	India
Enter phone num:	0000000000

data 2

For route: http://127.0.0.1:5000/participants



Name	Email	City	Country	Phone Number
Tarun R Jain	tarun@gmail.com	Bengaluru	India	1111111111
Rahul	rahul@gmail.com	Bengaluru	India	0000000000

Don't miss your chance to ride the wave of the data revolution! Every industry is scaling new heights by tapping into the power of data. Sharpen your skills and become a part of the hottest trend in the 21st century.