

## Week 5 Tutorial 2 (Date: Jan 24, 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']

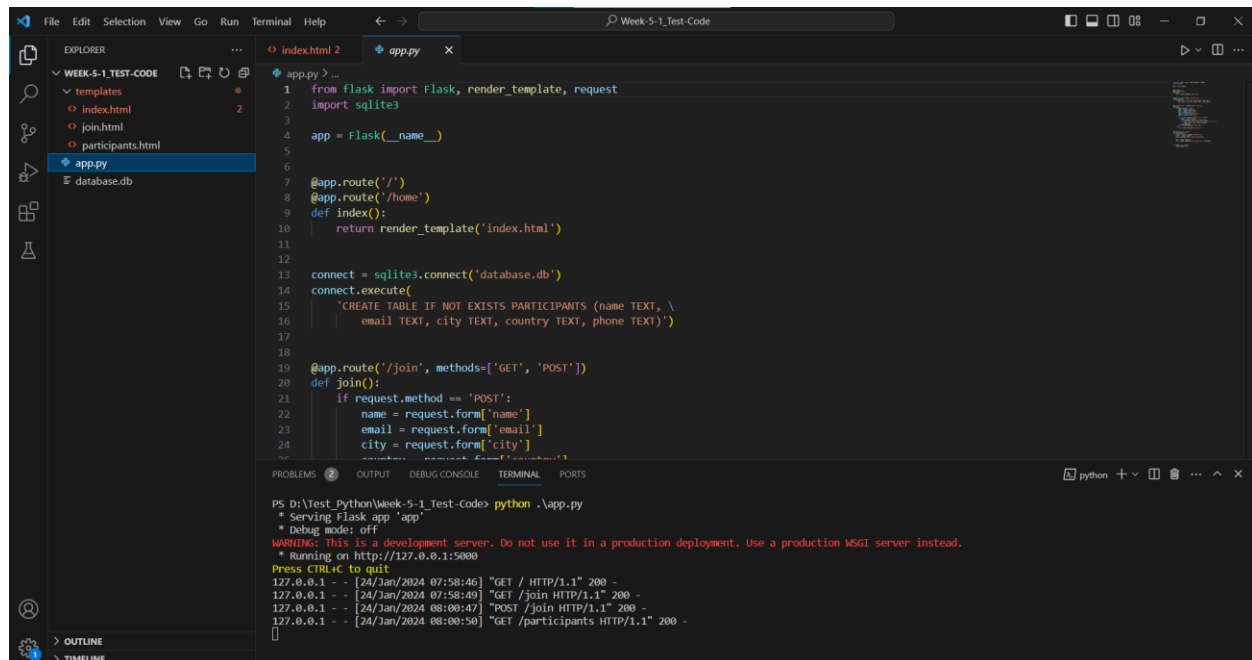
        with sqlite3.connect("database.db") as users:
            cursor = users.cursor()
            cursor.execute("INSERT INTO PARTICIPANTS \
            (name,email,city,country,phone) VALUES (?, ?, ?, ?, ?)",
                (name, email, city, country, phone))
            users.commit()
        return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



The screenshot shows a VS Code editor with a project named 'Week 5-1\_Test-Code'. The Explorer sidebar on the left shows the file structure: 'WEEK-5-1-TEST-CODE' (a folder), 'templates' (a folder containing 'index.html', 'join.html', and 'participants.html'), 'app.py' (a file), and 'database.db' (a file). The main editor area shows the content of 'app.py'. The code imports Flask, render\_template, request, and sqlite3. It creates a Flask app, sets routes for '/', '/home', and '/join', and defines an index() function. It also connects to a SQLite database and creates a table named 'PARTICIPANTS' with columns 'name', 'email', 'city', 'country', and 'phone'. The terminal at the bottom shows the command 'python .\app.py' and the output, including a warning about the development server and a list of HTTP requests.

```
1 from flask import Flask, render_template, request  
2 import sqlite3  
3  
4 app = Flask(__name__)  
5  
6  
7 @app.route('/')  
8 @app.route('/home')  
9 def index():  
10     return render_template('index.html')  
11  
12  
13 connect = sqlite3.connect('database.db')  
14 connect.execute(  
15     'CREATE TABLE IF NOT EXISTS PARTICIPANTS (name TEXT, \  
16         email TEXT, city TEXT, country TEXT, phone TEXT)')  
17  
18  
19 @app.route('/join', methods=['GET', 'POST'])  
20 def join():  
21     if request.method == 'POST':  
22         name = request.form['name']  
23         email = request.form['email']  
24         city = request.form['city']  
25         country = request.form['country']  
26         phone = request.form['phone']  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100
```

**Output:**

For route: <http://127.0.0.1:5000/>

## Build Web App Using Flask and SQLite

Fill form to get updates

Check participant list

**For route: <http://127.0.0.1:5000/join>**

Here we are adding two new data to the database.

Enter Name:	<input type="text" value="Tarun R Jain"/>
Enter Email:	<input type="text" value="tarun@gmail.com"/>
Enter City:	<input type="text" value="Bengaluru"/>
Enter Country:	<input type="text" value="India"/>
Enter phone num:	<input type="text" value="1111111111"/>
	<input type="button" value="submit"/>

*data 1*

Enter Name:

Enter Email:

Enter City:

Enter Country:

Enter phone num:

*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.