

Python | Using for loop in Flask

Prerequisite: [HTML Basics](#), [Python Basics](#), [Flask](#)

It is not possible to write front-end code every time user makes changes in his/her profile. We use a template and it generates code according to the content.

Flask is one of the web development frameworks written in Python. Through flask, a loop can be run in the HTML code using jinja template and automatically HTML code can be generated using this.

The code will be stored in Directories in the format of Flask. So we will be making two directories,

- **static** – For static Files like images, css, js
- **templates** – For Html templates

app.py file which will contain all the Python file will be stored in the main directory and index.html file will be stored in templates.

app.py

The code of app.py is same for both examples. We will print a Python list with Some names of Pokemons first in the format of a list and then a table.

```

# importing modules
from flask import Flask, render_template

# declaring app name
app = Flask(__name__)

# making list of pokemons
Pokemons = ["Pikachu", "Charizard", "Squirtle", "Jigglyput",
             "Bulbasaur", "Gengar", "Charmander", "Mew", "I

# defining home page
@app.route('/')
def homepage():

# returning index.html and list
# and length of list to html page
    return render_template("index.html", len = len(Pokemons))

# if __name__ == '__main__':

# running app
app.run(use_reloader = True, debug = True)

```

Example #1: Making a List

We will use the argument Pokemons passed from python file here to automatically print a list instead of Writing it everytime.

index.html

```

<!DOCTYPE html>

<html>
<head>
    <title>For loop in Flask</title>
</head>

```

```
<body>
```

```
<ol>
```

```
<!-- For loop logic of jinja template -->
{%for i in range(0, len)%}
```

```
    <li>{{Pokemons[i]}}</li>
{%endfor%}
```

```
</ol>
```

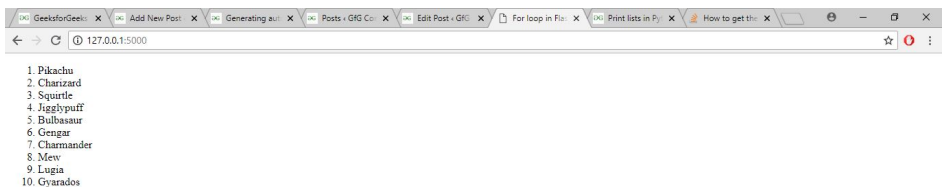
```
</body>
```

```
</html>
```



Output:

Without writing any data of list, the list will be automatically generated. You can use the css and js to make these look beautiful.



Example #2: Making a Table

We will use the argument Pokemons passed from python file here to automatically print a table instead of Writing it our self. Code for app.py for this example is same as the above one.

index.html

```
<!DOCTYPE html>

<html>
<head>
    <title>For loop in Flask</title>
</head>

<!-- Adding some style to table (OPTIONAL) -->
<style type="text/css">

    th:tr{
        color: blue;

    }
    tr:nth-of-type(2n){
        border: 1px solid black;
        background-color: rgba(150, 150, 150, 0.5);

    }

    td{
        padding: 8px 8px;
        border: 1px solid black;
    }
</style>

<body>

<table style="margin-left: 20px;">
<!-- Table headers -->
    <th>
        <tr style="color: green; ">
```

```

        <td>Serial Number</td>
        <td>Pokemon Name</td>
</tr>
</th>

<!-- For loop logic of jinja template -->
{%for i in range(0, len)%}

<!-- table rows -->
<tr>
    <td>{{i}}</td>
    <td>{{Pokemons[i]}}</td>

{%endfor%}

</tr>

</table>

</body>
</html>

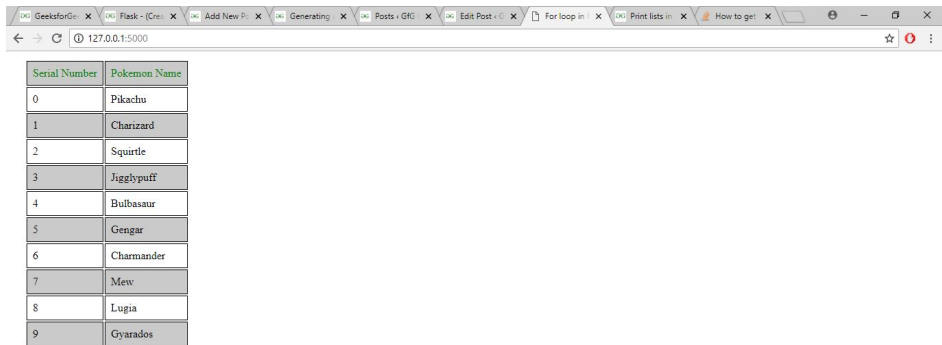
```



Output:

Without writing any data of list, the table will be automatically

generated.



The screenshot shows a web browser window with multiple tabs open. The active tab displays a table with two columns: 'Serial Number' and 'Pokemon Name'. The table contains 10 rows of data, including Pokemon like Pikachu, Charizard, Squirtle, Jigglypuff, Bulbasaur, Gengar, Charmander, Mew, Lugia, and Gyarados. The browser's address bar shows the URL '127.0.0.1:5000'.

Serial Number	Pokemon Name
0	Pikachu
1	Charizard
2	Squirtle
3	Jigglypuff
4	Bulbasaur
5	Gengar
6	Charmander
7	Mew
8	Lugia
9	Gyarados



Instructions to Run code:

- Download the files from link provided above or make and store the code in the same format
- Run the app.py file in root directory
- Go to the local host (<http://127.0.0.1:5000/> in my case) and there you have the website