Get the value of a checkbox in Flask

Last Updated : 02 Aug, 2024

The checkbox is a typical widget with selected options within a list by a user. They are employed in web forms whereby they are applied in the selection of preferences, settings, or a survey. Here in this article, we will see how to add the checkboxes in the Flask application right from the installation of Flask to the handling of inputs.

Get the value of a checkbox in Flask

Here is the most basic requirement before we start doing the checkboxes, we require a flask application. Flask is a framework for creating web applications based on Python, it is micro, which is why it doesn't contain a lot of components, but it provides the features needed to start an application. Follow these steps to set up your Flask environment:

Follow these steps to set up your Flask environment:

Step 1: Install Flask

If you haven't already, install Flask using pip. Open your terminal or command prompt and run:

```
pip install Flask
```

Step 2: Create a Flask Application

Create a new directory for your project and navigate to it. Inside this directory, application.

Flask Templates Jinja2 Flask-REST API Python SQLAlchemy Flask Bcrypt Flask Cookies Json Postman

Step 3: Create a Template Directory

Create a folder named templates in the same directory as app.py. Inside the templates folder, create a file named index.html.

Creating Checkboxes

In the **index.html** file, we will create a form with checkboxes. Each checkbox will allow the user to select an option, and when the form is submitted, the selected options will be sent to the server.

Code Implementation

Here is the complete example code for the Flask application with checkboxes:

index.html

In this example, we have created a form with three checkboxes and a submit button. Each checkbox has a unique id and value attribute. The name attribute for all checkboxes is the same (checkbox), which allows us to handle them as a group on the server side.

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Checkbox Example</title>
</head>
<body>
    <form action="/submit" method="post">
        <label for="option1">Option 1</label>
        <input type="checkbox" id="option1" name="checkbox" value="Option 1"><br>
        <label for="option2">Option 2</label>
        <input type="checkbox" id="option2" name="checkbox" value="Option 2"><br>
        <label for="option3">Option 3</label>
        <input type="checkbox" id="option3" name="checkbox" value="Option 3"><br>
        <input type="submit" value="Submit">
```

```
        </form>
</body>
</html>
```

app.py

Python

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

app = Flask(__name__)

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

@app.route('/submit', methods=['POST'])
def submit():
    selected_options = request.form.getlist('checkbox')
    return f'Selected options: {", ".join(selected_options)}'

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

Output

Best Practices to ensure your Application File

When working with checkboxes in Flask, consider the following best practices to ensure your application is robust and user-friendly:

Use Descriptive Labels: Ensure that each checkbox has a clear and descriptive label. This improves the user experience and accessibility, making it easier for users to understand the options available to them.

Validate User Input: Always validate the user input on the server side to ensure that the data received is as expected. This prevents malicious input and

ensures data integrity. For example, you can check if the selected options are within the allowed set of values:

Handle Multiple Checkboxes: Use request.form.getlist('checkbox') to handle multiple checkboxes with the same name attribute. This method returns an array of all selected options, which is convenient on selection of multiple options.

Maintain State: If you need to preserve the state of the checkboxes (e.g., when a form submission fails and needs to be corrected), consider passing the selected values back to the template and pre-selecting the checkboxes based on these values. Here's an example of how you can achieve this:

Use CSRF Protection: When working with form submissions user should take Care against Cross-Site Request Forgery attack. Flask-WTF is an extension that assists Flask applications to add CSRF protection. To use it, install Flask-WTF and configure it in your application:

```
pip install Flask-WTF
```

FAQs

1. How do I pre-select checkboxes based on existing data?

To pre-select checkboxes based on existing data, pass the selected values to the template and use conditional statements in your HTML to check the relevant checkboxes. Here's an example:

In app.py:

Python

```python
@app.route('/')
def index():
    selected_options = ['Option 1', 'Option 3']  # Example selected options
    return render_template('index.html', selected_options=selected_options)
```

In index.html:

HTML

```
<input type="checkbox" id="option1" name="checkbox" value="Option 1" {% if 'Option 1' in selected_options %}checked{% endif %}>
<input type="checkbox" id="option2" name="checkbox" value="Option 2" {% if 'Option 2' in selected_options %}checked{% endif %}>
<input type="checkbox" id="option3" name="checkbox" value="Option 3" {% if 'Option 3' in selected_options %}checked{% endif %}>
```

2. How can I style checkboxes?

You can style checkboxes using CSS. Here's an example:

CSS

```
input[type="checkbox"] {
    width: 20px;
    height: 20px;
    accent-color: #007bff; /* Change the color of the checkbox */
}
```

Add the CSS to a <style> tag in your HTML file or to a separate CSS file linked in your HTML.

3. How can I handle dynamic checkbox lists?

If you need to generate checkboxes dynamically (e.g., from a database), pass the list of options to the template and loop through them in your HTML. Here's an example:

In app.py:

Python

```python
@app.route('/')
def index():
    options = ['Option 1', 'Option 2', 'Option 3', 'Option 4']
    return render_template('index.html', options=options)
```

index.html:

Python

```python
<form action="/submit" method="post">
    {% for option in options %}
        <label for="{{ option }}">{{ option }}</label>
        <input type="checkbox" id="{{ option }}" name="checkbox" value="{{ option }}"><br>
    {% endfor %}
    <input type="submit" value="Submit">
</form>
```


Previous Article

Python | Checkbox widget in Kivy

Next Article

Similar Reads

Documenting Flask Endpoint using Flask-Autodoc

Documentation of endpoints is an essential task in web development and being able to apply it in different frameworks is always a utility. This article discusses...

4 min read

How to use Flask-Session in Python Flask ?

Flask Session - Flask-Session is an extension for Flask that supports Server-side Session to your application.The Session is the time between the client logs in to...

4 min read

How to Integrate Flask-Admin and Flask-Login

In order to merge the admin and login pages, we can utilize a short form or any other login method that only requires the username and password. This is know...

8 min read

Minify HTML in Flask using Flask-Minify

Flask offers HTML rendering as output, it is usually desired that the output HTML should be concise and it serves the purpose as well. In this article, we would...

12 min read

Flask URL Helper Function - Flask url_for()

In this article, we are going to learn about the flask url_for() function of the flask URL helper in Python. Flask is a straightforward, speedy, scalable library, used f...

11 min read

Flask HTTP methods, handle GET & POST requests

In this article, we are going to learn about how to handle GET and POST requests of the flask HTTP methods in Python. HTTP Protocol is necessary for data...

6 min read

GET Request Query Parameters with Flask

In this article, we will learn how we can use the request object in a flask to GET Request Query Parameters with Flask that is passed to your routes using Pytho...

4 min read

How to get data from 'ImmutableMultiDict' in flask

In this article, we will see how to get data from ImmutableMultiDict in the flask. It is a type of Dictionary in which a single key can have different values. It is used...

2 min read

Get the Data Received in a Flask request

In this article, we will learn how we can use the request object in a flask to Get the Data Received that is passed to your routes. and How To Process Get Reque...

6 min read

How to Get Data from API in Python Flask

In modern web development, APIs (Application Programming Interfaces) play a crucial role in enabling the interaction between different software systems. Flas...

2 min read

Article Tags : Python Python Flask Flask Projects

Practice Tags : python

Corporate & Communications Address:- A-143, 9th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305