Create Contact Us using WTForms in Flask

Last Updated : 26 May, 2022

WTForms is a library designed to make the processing of forms easier to manage. It handles the data submitted by the browser very easily. In this article, we will discuss how to create a contact us form using WTForms.

Advantages of WT-FORM:

1. We don't have to worry about validators.
2. Avoidance of Cross-Site Request Forgery (CSRF).
3. WTForms come as classes, so all the good come's from an object form.
4. No need to create any <label> or <input> elements manually using HTML.

Installation

Use the Terminal to install Flask-WTF.

```
pip install Flask-WTF
```

Stepwise Implementation

Step 1: Create a class having all elements that you want in your Form in the **main.py**.

Python3

```python
from flask_wtf import FlaskForm
from wtforms import StringField, validators, PasswordField, SubmitField
from wtforms.validators import DataRequired, Email
import email_validator


class contactForm(FlaskForm):
    name = StringField(label='Name', validators=[DataRequired()])
    email = StringField(label='Email', validators=[
      DataRequired(), Email(granular_message=True)])
       message= StringField(label='Message')
    submit = SubmitField(label="Log In")
```

Step 2: Create the object of the form and pass the object as a parameter in the render_template

Python3

```
@app.route("/", methods=["GET", "POST"])
def home():
    cform = contactForm()
    return render_template("contact.html", form=cform)
```

Step 3: Add CSRF protection. Add a secret key.

```
app.secret_key = "any-string-you-want-just-keep-it-secret"
```

Step 4: Add the fields in the contact.html HTML FILE.

```
{{ form.csrf_token }} is used to provide csrf protection.
```

HTML

```
<!DOCTYPE HTML>

<html>
    <head>
        <title>Contact</title>
    </head>
    <body>
        <div class="container">
            <h1>Contact Us</h1>
            <form method="POST" action="{{ url_for('home') }}">
                {{ form.csrf_token }}

                <p>
                    {{ form.name.label }}
                    <br>
                    {{ form.name }}
                </p>



                <p>
                    {{ form.email.label }}
                    <br>
                    {{ form.email(size=30) }}
                </p>



                <p>
                    {{ form.message.label }}
                    <br>
                    {{ form.message }}
                </p>

                {{ form.submit }}
            </form>
        </div>
    </body>
</html>
```

Step 5: Validating the Form and receiving the data.

Python3

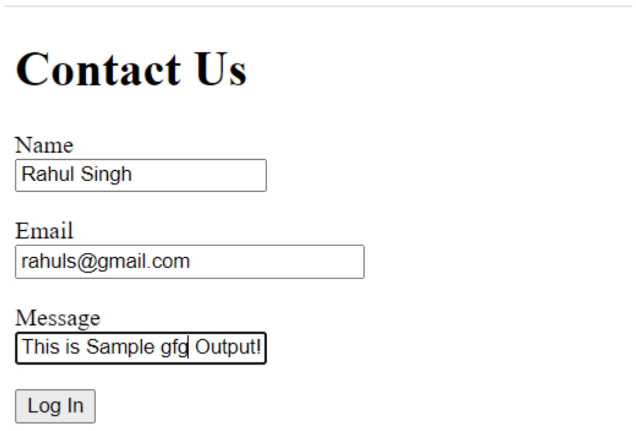
```
@app.route("/", methods=["GET", "POST"])
def home():
    cform = contactForm()
    if cform.validate_on_submit():
        print(f"Name:{cform.name.data},
              E-mail:{cform.email.data},
              message:{cform.message.data}")
    else:
        print("Invalid Credentials")

    return render_template("contact.html", form=cform)
```

Complete Code:

Python3

```
from flask import Flask, render_template, request, redirect, url_for
from flask_wtf import FlaskForm
from wtforms import StringField, validators, PasswordField, SubmitField
from wtforms.validators import DataRequired, Email
import email_validator

app = Flask(__name__)
app.secret_key = "any-string-you-want-just-keep-it-secret"

class contactForm(FlaskForm):
    name = StringField(label='Name', validators=[DataRequired()])
    email = StringField(
      label='Email', validators=[DataRequired(), Email(granular_message=True)])
    message = StringField(label='Message')
    submit = SubmitField(label="Log In")


@app.route("/", methods=["GET", "POST"])
def home():
    cform=contactForm()
    if cform.validate_on_submit():
            print(f"Name:{cform.name.data}, E-mail:{cform.email.data},
                  message:{cform.message.data}")
    return render_template("contact.html",form=cform)


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

Output:



Contact Us

Name
Rahul Singh

Email
rahuls@gmail.com

Message
This is Sample gfg Output!

```
Name:Rahul Singh, E-mail:rahuls@gmail.com, message:This is Sample gfg
Output!!!
```

Adding Bootstrap

We can also add the bootstrap to the above form to make it look interactive. For this, we will use the Flask-Bootstrap library. To install this module type the below command in the terminal.

```
pip install Flask-Bootstrap
```

Step 1: Create base.html

HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
    {% block content %}{% endblock %}
</body>
</html>
```

Step 2: Modify contact.html to

```
with single line {{ wtf.quick_form(form) }}
```

HTML

```html
{% extends 'bootstrap/base.html' %}
{% import "bootstrap/wtf.html" as wtf %}

{% block title %}
Contact Us
{% endblock %}
{% block content %}
        <div class="container">
            <h1>Contact Us</h1>
            {{ wtf.quick_form(form) }}
        </div>
{% endblock %}l>
```

Step 3: MODIFY main.py

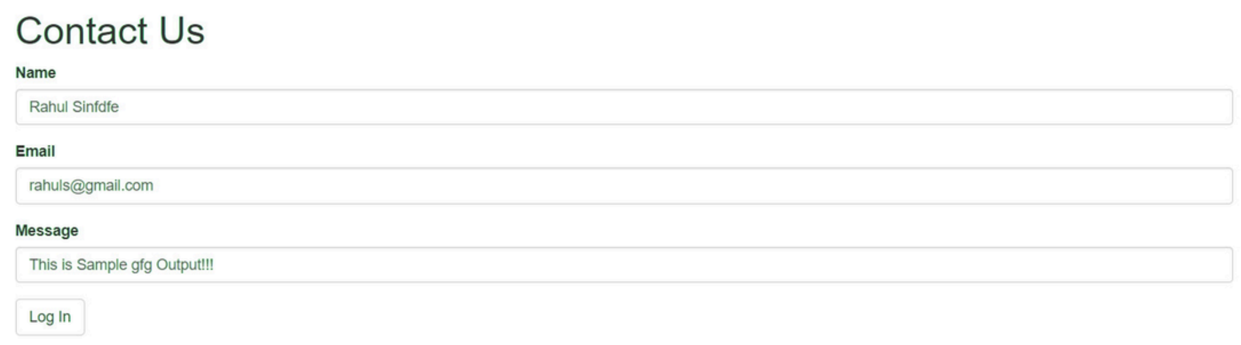
It is very simple to modify the .py file. We just have to import the module and add the below line into the code

```
Bootstrap(app)
```

Python3

```
from flask import Flask, render_template, request, redirect, url_for
from flask_wtf import FlaskForm
from wtforms import StringField, validators, PasswordField, SubmitField
from wtforms.validators import DataRequired, Email
from flask_bootstrap import Bootstrap
import email_validator
app = Flask(__name__)
Bootstrap(app)
app.secret_key = "any-string-you-want-just-keep-it-secret"

class contactForm(FlaskForm):
    name = StringField(label='Name', validators=[DataRequired()])
    email = StringField(label='Email', validators=[DataRequired(), Email(granular_
    message = StringField(label='Message')
    submit = SubmitField(label="Log In")


@app.route("/", methods=["GET", "POST"])
def home():
    cform=contactForm()
    if cform.validate_on_submit():
            print(f"Name:{cform.name.data}, E-mail:{cform.email.data}, message:{c
    return render_template("contact.html",form=cform)


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

Output:


Contact Us

Name

Rahul Sinfdfe

Email

rahuls@gmail.com

Message

This is Sample gfg Output!!!

Looking to dive into the world of programming or sharpen your Python skills? Our [Master Python: Complete Beginner to Advanced Course]() is your ultimate guide to becoming proficient in Python. This course covers everything you need to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!

R rajas... 2

Previous Article
Flask - Message Flashing

Next Article
Sending Emails Using API in Flask-Mail

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

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

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

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

How to Create Contact Chips using CSS?

Contact chips are UI elements that provide user information in a condensed, readable visual format. Usually, these chips have the user's name, avatar, and a...

2 min read

Create a Contact Form using HTML CSS & JavaScript

A contact form is a web form used to collect user information and messages, facilitating communication between visitors and site owners. It is essential for...

3 min read

Create Contact Cards Component Using Next.js and Tailwind CSS

The contact card component can display user information, such as a profile picture, contact details, and links to social media profiles. We will use Tailwind...

4 min read

Create Contact Form Template in Tailwind CSS

A Contact Form Template is a pre-designed layout that simplifies the process of creating a contact form for a website. It typically includes input fields for the...

4 min read

How to Create a Contact Form in WordPress ?

Creating a contact form in WordPress is essential for connecting with your audience. It's a straightforward process that can enhance user interaction on you...

3 min read

Article Tags : Python Web Technologies Flask Projects Python Flask

Practice Tags : python

GeeksforGeeks

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

GET IT ON Google Play
Download on the App Store

Company
About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution

Languages
Python
Java
C++
PHP
GoLang
SQL

Placement Training Program
GeeksforGeeks Community

R Language
Android Tutorial
Tutorials Archive

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

Inteview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

School Subjects

Mathematics
Physics
Chemistry

GeeksforGeeks Videos

DSA
Python
Java

Biology	C++
Social Science	Web Development
English Grammar	Data Science
Commerce	CS Subjects
World GK	

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved