

[Flask Templates](#) [Jinja2](#) [Flask-REST API](#) [Python SQLAlchemy](#) [Flask Bcrypt](#) [Flask Cookies](#) [Json](#) [Postman](#)

Build Blog website using Flask

Last Updated : 08 Mar, 2024

In this article, we'll explore how to build a dynamic blog website using Flask, a lightweight and versatile [Python](#) web framework. [Flask](#) provides developers with the tools needed to create robust web applications, and its simplicity makes it an excellent choice for beginners and experienced developers alike.

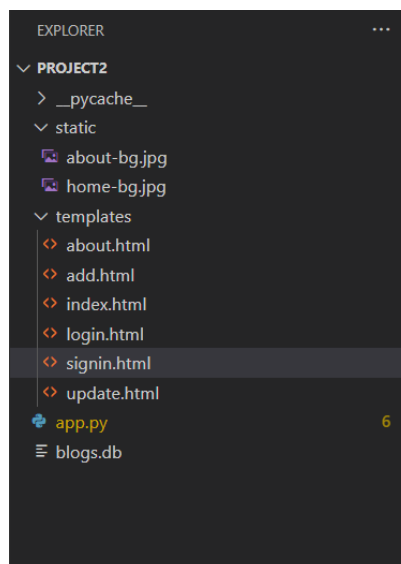
Requirements

- [Basics of HTML & CSS](#)
- [Bootstrap](#)
- [Basic of Python](#)
- [Flask](#)
- [Flask-SQLAlchemy](#)

Packages Required

```
pip install flask  
pip install Flask-SQLAlchemy
```

File Structure



Steps to Build Blog website using Flask

Step 1: Make a folder name as project2.

Step 2: We need to create 2 folders: "templates", and "statics".

Step 3: In templates create a file named an **index.html**.

HTML



```
<!doctype html>
<html lang="en">

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.
css" rel="stylesheet"
  integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfSpd3yD65VohhpUuCOmLASjC"
crossorigin="anonymous">

  <title>Home</title>
</head>

<body>

  <div id="part1">
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">Home</a>
        <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarScroll"
          aria-controls="navbarScroll" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarScroll">
          <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll"
style="--bs-scroll-height: 100px;">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page"
href="/">Home</a>
            </li>
            <li class="nav-item">
              <a class="nav-link active" href="{{ url_for('about')
}}">About</a>
            </li>

            <li class="nav-item">
              <a class="nav-link active" href="{{ url_for('addpost') }}"
tabindex="-1">Add</a>
            </li>

            {% if name != "guest" %}
```

```

        <li class="nav-item">
            <a class="nav-link active" href="{{ url_for('signin') }}"
tabindex="-1">Signin</a>
        </li>
    {% endif %}

</ul>
<div class="mx-3">
    {% if name == "guest" %}
        <a class="btn btn-danger" href="{{ url_for('login')
}}">Login</a>

        {% else %}
            <a class="btn btn-danger" href="#">{{name}}</a>
            <a class="btn btn-danger" href="{{ url_for('logout')
}}">Logout</a>

        {% endif %}

    </div>
</div>
</div>
</nav>
</div>

<div id="part2">
    <div id="carouselExampleCaptions" class="carousel slide" data-bs-
ride="carousel">
        <div class="carousel-indicators">
            <button type="button" data-bs-target="#carouselExampleCaptions"
data-bs-slide-to="0" class="active"
                aria-current="true" aria-label="Slide 1"></button>
            <button type="button" data-bs-target="#carouselExampleCaptions"
data-bs-slide-to="1"
                aria-label="Slide 2"></button>
        </div>
        <div class="carousel-inner">
            <div class="carousel-item active">
                
                <div class="carousel-caption d-none d-md-block">
                    <h5>First slide label</h5>
                    <p>Some representative placeholder content for the first
slide.</p>
                </div>
            </div>
            <div class="carousel-item">
                
                <div class="carousel-caption d-none d-md-block">
                    <h5>Second slide label</h5>
                    <p>Some representative placeholder content for the second
slide.</p>
                </div>
            </div>
        </div>
    </div>
</div>
</div>

```

```

</div>

<div id="part3">
  <div class="album py-5 bg-light">
    <div class="container">

      <div class="row row-cols-1 row-cols-sm-2 row-cols-md-3 g-3">

        <!-- <div class="col">
          <div class="card shadow-sm">
            <svg class="bd-placeholder-img card-img-top" width="100%"
height="225"
              xmlns="http://www.w3.org/2000/svg" role="img" aria-
label="Placeholder: Thumbnail"
              preserveAspectRatio="xMidYMid slice" focusable="false">
                <title>Placeholder</title>
                <rect width="100%" height="100%" fill="#55595c"></rect>
            <text x="50%" y="50%" fill="#eceeef"
              dy=".3em">Thumbnail</text>
            </svg>

            <div class="card-body">
              <h2 class="post-title">
                GFG Blog
              </h2>
              <p class="card-text">This is a wider card with
supporting text below as a natural lead-in to
              additional content. This content is a little bit
longer.</p>
              <div class="d-flex justify-content-between align-items-
center">
                <div class="btn-group">
                  <button type="button" class="btn btn-sm btn-outline-
secondary">Edit</button>
                  <button type="button" class="btn btn-sm btn-outline-
secondary">Delete</button>
                </div>
                <small class="text-muted">9 mins</small>
              </div>
            </div>
          </div>
        <!-- Loop to show all blogs that are present in the database -->
        {% for articles in article %}
          <div class="col">
            <div class="card shadow-sm">
              <svg class="bd-placeholder-img card-img-top" width="100%"
height="225"
                xmlns="http://www.w3.org/2000/svg" role="img" aria-
label="Placeholder: Thumbnail"
                preserveAspectRatio="xMidYMid slice" focusable="false">
                <title>Placeholder</title>
                <rect width="100%" height="100%" fill="#55595c"></rect>
              <text x="50%" y="50%" fill="#eceeef"
                dy=".3em">Thumbnail</text>
              </svg>

              <div class="card-body">

```

```

<h2 class="post-title">
  {{ articles.title }}
</h2>
<p class="card-text">{{articles.content}}</p>
<p class="post-meta">Posted by {{ articles.author }}

</p>

<div class="d-flex justify-content-between align-items-
center">
  <div class="btn-group">
    <!-- Check if the user is not guest, if guest "edit
and delete button will be hidden" -->
    {% if name != "guest" %}
      <a href="/update/{{articles.id}}" type="button"
class="btn btn-sm btn-outline-secondary">Edit</a>
      <a href="/delete/{{articles.id}}" type="button"
class="btn btn-sm btn-outline-secondary">Delete</a>
    {% endif %}
  </div>
  <small class="text-muted">{{
articles.post_date.strftime('%B %d, %Y') }}</small>
  </div>
</div>
</div>
</div>
</div>
</div>

<div>

</div>

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: Bootstrap Bundle with Popper -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle
.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>

</body>

</html>

```

Step 4: Create an about.html page.

HTML



```

1 <!doctype html>
2 <html lang="en">
3

```

```

4  <head>
5    <!-- Required meta tags -->
6    <meta charset="utf-8">
7    <meta name="viewport" content="width=device-width, initial-scale=1">
8
9    <!-- Bootstrap CSS -->
10   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/boot
11     integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfSpd3yD65V
12
13   <title>About</title>
14 </head>
15
16 <body>
17
18   <div id="part1">
19     <nav class="navbar navbar-expand-lg navbar-light bg-light">
20       <div class="container-fluid">
21         <a class="navbar-brand" href="#">GFG Blog</a>
22         <button class="navbar-toggler" type="button" data-bs-toggle="col
23           aria-controls="navbarScroll" aria-expanded="false" aria-label=
24           <span class="navbar-toggler-icon"></span>
25         </button>
26         <div class="collapse navbar-collapse" id="navbarScroll">
27           <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll"
28             <li class="nav-item">
29               <a class="nav-link active" aria-current="page" href="/">Hc
30             </li>
31             <li class="nav-item">
32               <a class="nav-link" href="/about.html">About</a>
33             </li>
34
35             <li class="nav-item">
36               <a class="nav-link" href="/add.html" tabindex="-1">Add</a>
37             </li>
38           </ul>
39           <form class="d-flex">
40             <input class="form-control me-2" type="search" placeholder="
41             <button class="btn btn-outline-success" type="submit">Search
42           </form>
43         </div>
44       </div>
45     </nav>
46   </div>
47
48   <div id="part2">
49     <div id="carouselExampleCaptions" class="carousel slide" data-bs-ric
50       <div class="carousel-indicators">
51         <button type="button" data-bs-target="#carouselExampleCaptions"
52           aria-current="true" aria-label="Slide 1"></button>
53       </div>
54       <div class="carousel-inner">
55         <div class="carousel-item active">
56           
58             <h5>About Me</h5>
59             <p>Some representative placeholder content for the first sli
60           </div>

```

```

61         </div>
62
63     </div>
64
65 </div>
66 </div>
67
68 <div class="container">
69     <p class="lead">Lorem ipsum, dolor sit amet consectetur adipisicing
70     Ipsam ea, quam tempora quibusdam, ullam ex consequuntur illum
71     dolor cum, aperiam illo obcaecati facere voluptatibus? Voluptas
72     deserunt accusamus eius! Quo velit culpa obcaecati.</p>
73 </div>
74
75 <div>
76
77 </div>
78
79 <!-- Optional JavaScript; choose one of the two! -->
80
81 <!-- Option 1: Bootstrap Bundle with Popper -->
82 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/boot
83     integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+
84     crossorigin="anonymous"></script>
85
86 <!-- Option 2: Separate Popper and Bootstrap JS -->
87 <!--
88     <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/
89     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bc
90     -->
91 </body>
92
93 </html>

```

Step 5: Create an `add.html` page.

HTML

```

1  <!doctype html>
2  <html lang="en">
3
4  <head>
5      <!-- Required meta tags -->
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1">
8
9      <!-- Bootstrap CSS -->
10     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
11         integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfspd3yD65VohhpuuCOMLASjC"
12
13     <title>Add Post</title>
14 </head>
15

```

```

16 <body>
17
18 <div id="part1">
19 <nav class="navbar navbar-expand-lg navbar-light bg-light">
20 <div class="container-fluid">
21 <a class="navbar-brand" href="#">GFG Blog</a>
22 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
23 aria-controls="navbarScroll" aria-expanded="false" aria-label="Toggle navigat
24 <span class="navbar-toggler-icon"></span>
25 </button>
26 <div class="collapse navbar-collapse" id="navbarScroll">
27 <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll" style="--bs-scr
28 <li class="nav-item">
29 <a class="nav-link active" aria-current="page" href="/">Home</a>
30 </li>
31 <li class="nav-item">
32 <a class="nav-link" href="/about.html">About</a>
33 </li>
34
35 <li class="nav-item">
36 <a class="nav-link" href="/add.html" tabindex="-1">Add</a>
37 </li>
38 </ul>
39 <form class="d-flex">
40 <input class="form-control me-2" type="search" placeholder="Search" aria-lal
41 <button class="btn btn-outline-success" type="submit">Search</button>
42 </form>
43 </div>
44 </div>
45 </nav>
46 </div>
47
48 <div id="part3">
49 <div class="container">
50 <div class="row">
51 <div class="col-lg-8 col-md-10 mx-auto">
52 <form name="addForm" id="addForm" method="POST" action="{ { url_for('addpost')
53 <div class="control-group">
54 <div class="form-group floating-label-form-group controls">
55 <label>Title</label>
56 <input type="text" class="form-control" placeholder="Title" name="title"
57 data-validation-required-message="Please enter a title.">
58 <p class="help-block text-danger"></p>
59 </div>
60 </div>
61
62 <div class="control-group">
63 <div class="form-group col-xs-12 floating-label-form-group controls">
64 <label>Author</label>
65 <input type="text" class="form-control" placeholder="Author" name="autho
66 data-validation-required-message="Please enter your phone number.">
67 <p class="help-block text-danger"></p>
68 </div>
69 </div>
70 <div class="control-group">
71 <div class="form-group floating-label-form-group controls">
72 <label>Blog Content</label>

```



```

73         <textarea rows="5" class="form-control" placeholder="Blog content" name="
74             data-validation-required-message="Please enter a message."></textarea>
75         <p class="help-block text-danger"></p>
76     </div>
77 </div>
78 <div class="mb-3">
79     <label for="formFileSm" class="form-label">Small file input example</label>
80     <input class="form-control form-control-sm" id="formFileSm" type="file">
81 </div>
82 <br>
83 <div id="success"></div>
84 <div class="form-group">
85     <button type="submit" class="btn btn-secondary" id="sendMessageButton">Se
86 </div>
87 </form>
88 </div>
89 </div>
90 </div>
91 </div>
92
93 <div>
94
95 </div>
96
97 <!-- Optional JavaScript; choose one of the two! -->
98
99 <!-- Option 1: Bootstrap Bundle with Popper -->
100 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min
101     integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
102     crossorigin="anonymous"></script>
103
104 <!-- Option 2: Separate Popper and Bootstrap JS -->
105 <!--
106     <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
107     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
108     -->
109 </body>
110
111 </html>

```

Step 6: Create a `update.html` page.

Python3

```

1  <!doctype html>
2  <html lang="en">
3
4  <head>
5      <!-- Required meta tags -->
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1">
8
9      <!-- Bootstrap CSS -->
10     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
11         integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfspd3yD65VohhpuuCOMLA'

```

```

12
13     <title>Update Post</title>
14 </head>
15
16 <body>
17
18     <div id="part1">
19         <nav class="navbar navbar-expand-lg navbar-light bg-light">
20             <div class="container-fluid">
21                 <a class="navbar-brand" href="#">GFG Blog</a>
22                 <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
23                     aria-controls="navbarScroll" aria-expanded="false" aria-label="Toggle
24                     <span class="navbar-toggler-icon"></span>
25                 </button>
26                 <div class="collapse navbar-collapse" id="navbarScroll">
27                     <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll" style:
28                         <li class="nav-item">
29                             <a class="nav-link active" aria-current="page" href="/">Home
30                         </li>
31                         <li class="nav-item">
32                             <a class="nav-link" href="/about.html">About</a>
33                         </li>
34
35                         <li class="nav-item">
36                             <a class="nav-link" href="/add.html" tabindex="-1">Add</a>
37                         </li>
38                     </ul>
39                     <form class="d-flex">
40                         <input class="form-control me-2" type="search" placeholder="Sea
41                         <button class="btn btn-outline-success" type="submit">Search</b
42                     </form>
43                 </div>
44             </div>
45         </nav>
46     </div>
47
48     <div id="part2">
49         <div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="carousel"
50             <div class="carousel-indicators">
51                 <button type="button" data-bs-target="#carouselExampleCaptions" data-bs
52                     aria-current="true" aria-label="Slide 1"></button>
53             </div>
54             <div class="carousel-inner">
55                 <div class="carousel-item active">
56                     
57                     <div class="carousel-caption d-none d-md-block">
58                         <h5>About Me</h5>
59                         <p>Some representative placeholder content for the first slide.
60                     </div>
61                 </div>
62
63             </div>
64
65         </div>
66     </div>
67
68     <div id="part3">

```

```

69     <div class="container">
70         <div class="row">
71             <div class="col-lg-8 col-md-10 mx-auto">
72                 <form name="addForm" id="addForm" method="POST" action="/update/{{e
73                     <div class="control-group">
74                         <div class="form-group floating-label-form-group controls">
75                             <label>Update Title</label>
76                             <input type="text" class="form-control" value="{{edit.t
77                                 required data-validation-required-message="Please en
78                             <p class="help-block text-danger"></p>
79                         </div>
80                     </div>
81
82                     <div class="control-group">
83                         <div class="form-group col-xs-12 floating-label-form-group">
84                             <label>Author</label>
85                             <input type="text" class="form-control" value="{{edit.a
86                                 id="author" required
87                                 data-validation-required-message="Please enter your
88                             <p class="help-block text-danger"></p>
89                         </div>
90                     </div>
91                     <div class="control-group">
92                         <div class="form-group floating-label-form-group controls">
93                             <label>Blog Content</label>
94
95
96                             <!-- <textarea rows="5" class="form-control"
97                                 placeholder="{{edit.content}}" name="content" id="c
98                                 data-validation-required-message="Please enter a me
99                                 Geeksforgeeks
100                             </textarea> -->
101
102                             <input type="text" value="{{edit.content}}" name="con
103                                 class="form-control">
104                             <p class="help-block text-danger"></p>
105                         </div>
106                     </div>
107                     <br>
108                     <div id="success"></div>
109                     <div class="form-group">
110                         <button type="submit" class="btn btn-secondary" id="sendMes
111                     </div>
112                 </form>
113             </div>
114         </div>
115     </div>
116 </div>
117
118 <div>
119
120 </div>
121
122 <!-- Optional JavaScript; choose one of the two! -->
123
124 <!-- Option 1: Bootstrap Bundle with Popper -->
125 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.js">

```

```

126         integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIax\
127         crossorigin="anonymous"></script>
128
129     <!-- Option 2: Separate Popper and Bootstrap JS -->
130     <!--
131     <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.
132     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
133     -->
134 </body>
135
136 </html>

```

Step 7: Create a login.html page.

HTML



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta name="viewport" content="width=device-width, initial-scale=1">
5 <title>Login</title>
6 <style>
7 body {font-family: Arial, Helvetica, sans-serif;}
8
9 /* Full-width input fields */
10 input[type=text], input[type=password] {
11     width: 100%;
12     padding: 12px 20px;
13     margin: 8px 0;
14     display: inline-block;
15     border: 1px solid #ccc;
16     box-sizing: border-box;
17 }
18
19 /* Set a style for all buttons */
20 button {
21     background-color: #04AA6D;
22     color: white;
23     padding: 14px 20px;
24     margin: 8px 0;
25     border: none;
26     cursor: pointer;
27     width: 100%;
28 }
29
30 button:hover {
31     opacity: 0.8;
32 }
33
34 /* Extra styles for the cancel button */
35 .cancelbtn {
36     width: auto;
37     padding: 10px 18px;
38     background-color: #f44336;
39 }

```

```
40
41  /* Center the image and position the close button */
42  .imgcontainer {
43      text-align: center;
44      margin: 24px 0 12px 0;
45      position: relative;
46  }
47
48  img.avatar {
49      width: 40%;
50      border-radius: 50%;
51  }
52
53  .container {
54      padding: 16px;
55  }
56
57  span.psw {
58      float: right;
59      padding-top: 16px;
60  }
61
62  /* The Modal (background) */
63  .modal {
64      display: none; /* Hidden by default */
65      position: fixed; /* Stay in place */
66      z-index: 1; /* Sit on top */
67      left: 0;
68      top: 0;
69      width: 100%; /* Full width */
70      height: 100%; /* Full height */
71      overflow: auto; /* Enable scroll if needed */
72      background-color: rgb(0,0,0); /* Fallback color */
73      background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
74      padding-top: 60px;
75  }
76
77  /* Modal Content/Box */
78  .modal-content {
79      background-color: #fefefe;
80      margin: 5% auto 15% auto; /* 5% from the top, 15% from
81      the bottom and centered */
82      border: 1px solid #888;
83      width: 80%; /* Could be more or less, depending on screen size */
84  }
85
86  /* The Close Button (x) */
87  .close {
88      position: absolute;
89      right: 25px;
90      top: 0;
91      color: #000;
92      font-size: 35px;
93      font-weight: bold;
94  }
95
96  .close:hover,
```

```

97     .close:focus {
98         color: red;
99         cursor: pointer;
100     }
101
102     /* Add Zoom Animation */
103     .animate {
104         -webkit-animation: animatezoom 0.6s;
105         animation: animatezoom 0.6s
106     }
107
108     @-webkit-keyframes animatezoom {
109         from {-webkit-transform: scale(0)}
110         to {-webkit-transform: scale(1)}
111     }
112
113     @keyframes animatezoom {
114         from {transform: scale(0)}
115         to {transform: scale(1)}
116     }
117
118     /* Change styles for span and cancel button on extra small screens */
119     @media screen and (max-width: 300px) {
120         span.psw {
121             display: block;
122             float: none;
123         }
124         .cancelbtn {
125             width: 100%;
126         }
127     }
128 </style>
129 </head>
130 <body>
131
132
133     <form class="modal-content animate" action="{ { url_for('login') }}"
134         method="POST">
135
136         <div class="container">
137             <label for="uname"><b>Username</b></label>
138             <input type="text" placeholder="Enter Username"
139                 name="username" id="username" required>
140
141             <label for="psw"><b>Password</b></label>
142             <input type="password" placeholder="Enter Password"
143                 name="password" id="password" required>
144
145             <button type="submit">Login</button>
146             <label>
147                 <input type="checkbox" checked="checked"
148                     name="remember"> Remember me
149             </label>
150         </div>
151
152         <div class="container" style="background-color:#f1f1f1">
153             <button type="button">

```

```

154         onclick="document.getElementById('id01').style.display='none'"
155         class="cancelbtn">Cancel</button>
156     <span class="psw">Forgot <a href="#">password?</a></span>
157 </div>
158 </form>
159
160
161
162 </body>
163 </html>

```

Step 8: Create a **signin.html** page.

HTML



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta name="viewport" content="width=device-width, initial-scale=1">
5 <title>Signup</title>
6
7 <style>
8 body {font-family: Arial, Helvetica, sans-serif;}
9
10 /* Full-width input fields */
11 input[type=text], input[type=password] {
12     width: 100%;
13     padding: 12px 20px;
14     margin: 8px 0;
15     display: inline-block;
16     border: 1px solid #ccc;
17     box-sizing: border-box;
18 }
19
20 /* Set a style for all buttons */
21 button {
22     background-color: #04AA6D;
23     color: white;
24     padding: 14px 20px;
25     margin: 8px 0;
26     border: none;
27     cursor: pointer;
28     width: 100%;
29 }
30
31 button:hover {
32     opacity: 0.8;
33 }
34
35 /* Extra styles for the cancel button */
36 .cancelbtn {
37     width: auto;
38     padding: 10px 18px;
39     background-color: #f44336;
40 }
41

```

```
42  /* Center the image and position the close button */
43  .imgcontainer {
44      text-align: center;
45      margin: 24px 0 12px 0;
46      position: relative;
47  }
48
49  img.avatar {
50      width: 40%;
51      border-radius: 50%;
52  }
53
54  .container {
55      padding: 16px;
56  }
57
58  span.psw {
59      float: right;
60      padding-top: 16px;
61  }
62
63  /* The Modal (background) */
64  .modal {
65      display: none; /* Hidden by default */
66      position: fixed; /* Stay in place */
67      z-index: 1; /* Sit on top */
68      left: 0;
69      top: 0;
70      width: 100%; /* Full width */
71      height: 100%; /* Full height */
72      overflow: auto; /* Enable scroll if needed */
73      background-color: rgb(0,0,0); /* Fallback color */
74      background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
75      padding-top: 60px;
76  }
77
78  /* Modal Content/Box */
79  .modal-content {
80      background-color: #fefefe;
81      margin: 5% auto 15% auto; /* 5% from the top, 15%
82      from the bottom and centered */
83      border: 1px solid #888;
84      width: 80%; /* Could be more or less, depending on screen size */
85  }
86
87  /* The Close Button (x) */
88  .close {
89      position: absolute;
90      right: 25px;
91      top: 0;
92      color: #000;
93      font-size: 35px;
94      font-weight: bold;
95  }
96
97  .close:hover,
98  .close:focus {
```



```

99     color: red;
100    cursor: pointer;
101  }
102
103  /* Add Zoom Animation */
104  .animate {
105    -webkit-animation: animatezoom 0.6s;
106    animation: animatezoom 0.6s
107  }
108
109  @-webkit-keyframes animatezoom {
110    from {-webkit-transform: scale(0)}
111    to {-webkit-transform: scale(1)}
112  }
113
114  @keyframes animatezoom {
115    from {transform: scale(0)}
116    to {transform: scale(1)}
117  }
118
119  /* Change styles for span and cancel button on extra small screens */
120  @media screen and (max-width: 300px) {
121    span.psw {
122      display: block;
123      float: none;
124    }
125    .cancelbtn {
126      width: 100%;
127    }
128  }
129  </style>
130  </head>
131  <body>
132
133
134    <form class="modal-content animate"
135          action="{ { url_for('signin') } }" method="POST">
136
137      <div class="container">
138        <label for="uname"><b>Username</b></label>
139        <input type="text" placeholder="Enter Username"
140              name="username" id="username" required>
141
142        <label for="psw"><b>Password</b></label>
143        <input type="password" placeholder="Enter Password"
144              name="password" id="password" required>
145
146        <button type="submit">Sign In</button>
147
148      </div>
149
150      <div class="container" style="background-color:#f1f1f1">
151        <button type="button"
152              onclick="document.getElementById('id01').style.display='none'"
153              class="cancelbtn">Cancel</button>
154        <span class="psw">Forgot <a href="#">password?</a></span>
155      </div>

```

```

156     </form>
157
158 </body>
159 </html>

```

Step 9: Create an app.py

Import all libraries.

```

from distutils.log import debug
from flask import Flask, render_template, request, redirect, url_for
from flask_login import UserMixin, login_user, login_required, logout_user, current_user
from flask_sqlalchemy import SQLAlchemy
from flask_login import login_user, logout_user, login_required, LoginManager
from werkzeug.security import generate_password_hash, check_password_hash
from datetime import datetime
from flask import flash

```

Create a Flask application object and set the URI for the database to use.

```

app = Flask(__name__)
db = SQLAlchemy(app)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///blogs.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
app.config['SECRET_KEY'] = 'thisisasecretkey'

```

Create a class for our blog. Then use the application object as a parameter to create a GFGBLOG object of class SQLAlchemy.

```

class GFGBLOG(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(50))
    author = db.Column(db.String(20))
    post_date = db.Column(db.DateTime)
    content = db.Column(db.Text)

```

Create a class for users, and use the application object as a parameter to create a User object of class SQLAlchemy.

```

class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20), nullable=False, unique=True)
    password = db.Column(db.String(80), nullable=False)

```

Here we will return the index.html whenever the route is called. The show all () function, which is tied to the '/' URL, is the application's entrance point. The HTML template receives the GFGBLOG table's record set as an input. The record is rendered as an HTML table by the template's server-side code.

```

@app.route("/")
def hello_world():
    article = GFGBLOG.query.order_by(GFGBLOG.post_date.desc()).all()
    print(current_user.is_anonymous)
    if current_user.is_anonymous:
        name = "guest"
    else:
        name = current_user.username
    print("bye")
    return render_template('index.html', article=article, name=name)

```

Here we will return the about.html whenever the **/about** is called.

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

Here we will return the about.html whenever the /about is called. If there is any POST request then the 'title', 'author', and 'content' will be saved in the database and it will redirect to the home page.

```
@app.route('/addpost', methods=['POST', 'GET'])
def addpost():
    if request.method == 'POST':
        title = request.form['title']
        author = request.form['author']
        content = request.form['content']
        post = GFGBLOG(title=title, author=author, content=content,
            post_date=datetime.now())
        db.session.add(post)
        db.session.commit()
        print("Done")
        return redirect(url_for('hello_world'))
    return render_template('add.html')
```

In the update, we will return the about.html whenever the /update is called. If there is any POST request for editing the blog then it will query the blog id from the database and then the 'title', 'author', and 'content' will be updated in the database and it will redirect to the home page.

Note: This can be only edited by a registered user, if a user is not registered then the edit button will be not shown to the Guest user.

```
@app.route('/update/<int:id>', methods=['POST', 'GET'])
@login_required
def update(id):
    if request.method == 'POST':
        title = request.form['title']
        author = request.form['author']
        content = request.form['content']
        print(content)
        post = GFGBLOG.query.filter_by(id=id).first()
        post.title = title
        post.author = author
        post.content = content
        db.session.add(post)
        db.session.commit()
        return redirect("/")
    edit = GFGBLOG.query.filter_by(id=id).first()
    return render_template('update.html', edit=edit)
```

In the delete, If there is any request for deleting the blog then it will query the blog id from the database then it will filter the first occurrence from the database and redirect to the home page.

Note: This can be only deleted by a registered user, if a user is not registered then the delete button will be not shown to the Guest user.

```
@app.route('/delete/<int:id>')
@login_required
def delete(id):
```

```
d = GFGBLOG.query.filter_by(id=id).first()
db.session.delete(d)
db.session.commit()
return redirect(url_for('hello_world'))
```

Here we will return the login.html whenever the login is called. If there is any POST request for logging a user then it will query the username from the database and if the user exists then it will log the user otherwise a message will flash to check user credentials.

```
@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        # print("hello")
        username = request.form['username']
        password = request.form['password']
        user = User.query.filter_by(username=username).first()
        if not user and not check_password_hash(user.password, password):
            flash('Please check your login details and try again.')
            # return redirect(url_for('auth.login'))
            return render_template('not.html')
        else:
            login_user(user)
            print("yes")
            return redirect(url_for('hello_world'))
    return render_template('login.html')
```

In the sign-in, we will return the signin.html whenever the **/signin** is called. If there is any POST request for registering a user for the blog then it will be saved in the User class database and it will redirect to the home page.

```
@app.route('/signin', methods=['POST', 'GET'])
def signin():
    if request.method == 'POST':
        print("hello")
        username = request.form['username']
        password = request.form['password']
        user = User(username=username, password=password)
        db.session.add(user)
        db.session.commit()
        return redirect(url_for('hello_world'))
    return render_template('signin.html')
```

Whenever a user clicks on a logout button then the app will log out that current user.

```
@app.route('/logout', methods=['GET', 'POST'])
@login_required
def logout():
    logout_user()
    return redirect(url_for('hello_world'))
```

Code Implementation:

Python3



```

1  from distutils.log import debug
2  from flask import Flask, render_template, request, redirect, url_for
3  from flask_login import UserMixin, login_user,
4      login_required, logout_user, current_user
5  from flask_sqlalchemy import SQLAlchemy
6  from flask_login import login_user, logout_user, login_required, LoginManager
7  from werkzeug.security import generate_password_hash, check_password_hash
8  from datetime import datetime
9  from flask import flash
10
11
12  app = Flask(__name__)
13  db = SQLAlchemy(app)
14  app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///blogs.db'
15  app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
16  app.config['SECRET_KEY'] = 'thisisasecretkey'
17
18  login_manager = LoginManager()
19  login_manager.init_app(app)
20  login_manager.login_view = 'login'
21
22
23  @login_manager.user_loader
24  def load_user(user_id):
25      return User.query.get(int(user_id))
26
27  class GFGBLOG(db.Model):
28      id = db.Column(db.Integer, primary_key=True)
29      title = db.Column(db.String(50))
30      author = db.Column(db.String(20))
31      post_date = db.Column(db.DateTime)
32      content = db.Column(db.Text)
33
34  class User(db.Model, UserMixin):
35      id = db.Column(db.Integer, primary_key=True)
36      username = db.Column(db.String(20), nullable=False, unique=True)
37      password = db.Column(db.String(80), nullable=False)
38
39
40
41  @app.route("/")
42  def hello_world():
43      article = GFGBLOG.query.order_by(GFGBLOG.post_date.desc()).all()
44      print(current_user.is_anonymous)
45      if current_user.is_anonymous:
46          name = "guest"
47      else:
48          name = current_user.username
49      print("bye")
50
51      return render_template('index.html', article=article, name=name)
52
53
54  @app.route('/about')
55  def about():
56      return render_template('about.html')
57

```

```
58
59 @app.route('/addpost', methods=['POST', 'GET'])
60 def addpost():
61     if request.method == 'POST':
62         title = request.form['title']
63         author = request.form['author']
64         content = request.form['content']
65
66         post = GFGBLOG(title=title, author=author,
67                        content=content, post_date=datetime.now())
68
69         db.session.add(post)
70         db.session.commit()
71         print("Done")
72         return redirect(url_for('hello_world'))
73     return render_template('add.html')
74
75 @app.route('/update/<int:id>', methods=['POST', 'GET'])
76 @login_required
77 def update(id):
78     if request.method == 'POST':
79         title = request.form['title']
80         author = request.form['author']
81         content = request.form['content']
82         print(content)
83
84         post = GFGBLOG.query.filter_by(id=id).first()
85
86         post.title = title
87         post.author = author
88         post.content = content
89
90         db.session.add(post)
91         db.session.commit()
92         return redirect("/")
93
94         edit = GFGBLOG.query.filter_by(id=id).first()
95         return render_template('update.html', edit=edit)
96
97 @app.route('/delete/<int:id>')
98 @login_required
99 def delete(id):
100     d = GFGBLOG.query.filter_by(id=id).first()
101     db.session.delete(d)
102     db.session.commit()
103     return redirect(url_for('hello_world'))
104
105 @app.route('/login', methods=['POST', 'GET'])
106 def login():
107     if request.method == 'POST':
108         # print("hello")
109         username = request.form['username']
110         password = request.form['password']
111         user = User.query.filter_by(username=username).first()
112
113         if not user and not check_password_hash(user.password, password):
114             flash('Please check your login details and try again.')
```

```

115         # return redirect(url_for('auth.login'))
116         return render_template('not.html')
117     else:
118         login_user(user)
119         print("yes")
120         return redirect(url_for('hello_world'))
121
122     return render_template('login.html')
123
124 @app.route('/signin', methods=['POST', 'GET'])
125 def signin():
126     if request.method == 'POST':
127         print("hello")
128         username = request.form['username']
129         password = request.form['password']
130
131         user = User(username=username, password=password)
132         db.session.add(user)
133         db.session.commit()
134         return redirect(url_for('hello_world'))
135
136     return render_template('signin.html')
137
138
139 @app.route('/logout', methods=['GET', 'POST'])
140 @login_required
141 def logout():
142     logout_user()
143     return redirect(url_for('hello_world'))
144
145 # main driver function
146 if __name__ == '__main__':
147
148     # run() method of Flask class runs the application
149     # on the local development server.
150     app.run(debug=True)

```

Step 10: Create your database

Select cmd.

Type "**python**", to open the python console.

Type "**from <app_name> from db**".

Then type "**db.create_all()**", and you will see a database is created named **blogs.db** in the root directory.

Then you can type **exit()** to exit from the python console.

Output:

Note: For a first-time user you can register yourself by going directly to the link "<http://127.0.0.1:5000/signin>" which will show you to the sign-in page to register yourself in the database.

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 mailf...



1

[Next Article](#)

[E-commerce Website using Django](#)

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 how endpoints ...

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 display minificatio...

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 the server and...

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 known as...

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 for building,...

11 min read

Create A Bookmark Organizer For Website Using Flask

This article explains how to make a tool called a Bookmark Organizer with Flask for a website. With this tool, we can add, create, update, delete, and edit bookmarks easily...

5 min read

Python | Build a REST API using Flask

Prerequisite: Introduction to Rest API REST stands for REpresentational State Transfer and is an architectural style used in modern web development. It defines a set or...

3 min read

Build a Text Translator Web App using Flask and Azure Cognitive Services

We're going to create a website that will translate text into multiple languages using Artificial Intelligence(AI). We'll use Flask framework for the Front end and Azure...

7 min read

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...

4 min read

How to Build a Simple Android App with Flask Backend?

Flask is an API of Python that allows us to build up web applications. It was developed by Armin Ronacher. Flask's framework is more explicit than Django's framework and is...

8 min read

Article Tags :

[Geeks Premier League](#)[Python](#)[Geeks Premier League 2023](#)[Python Flask](#)[+1 More](#)

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



Company

About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

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

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Computer Science

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

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

Data Science & ML

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

Python Tutorial

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

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

Interview Preparation

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

OOAD

Puzzles

System Design Bootcamp

Interview Questions

School Subjects

Mathematics

Physics

Chemistry

Biology

Social Science

English Grammar

Commerce

World GK

GeeksforGeeks Videos

DSA

Python

Java

C++

Web Development

Data Science

CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved