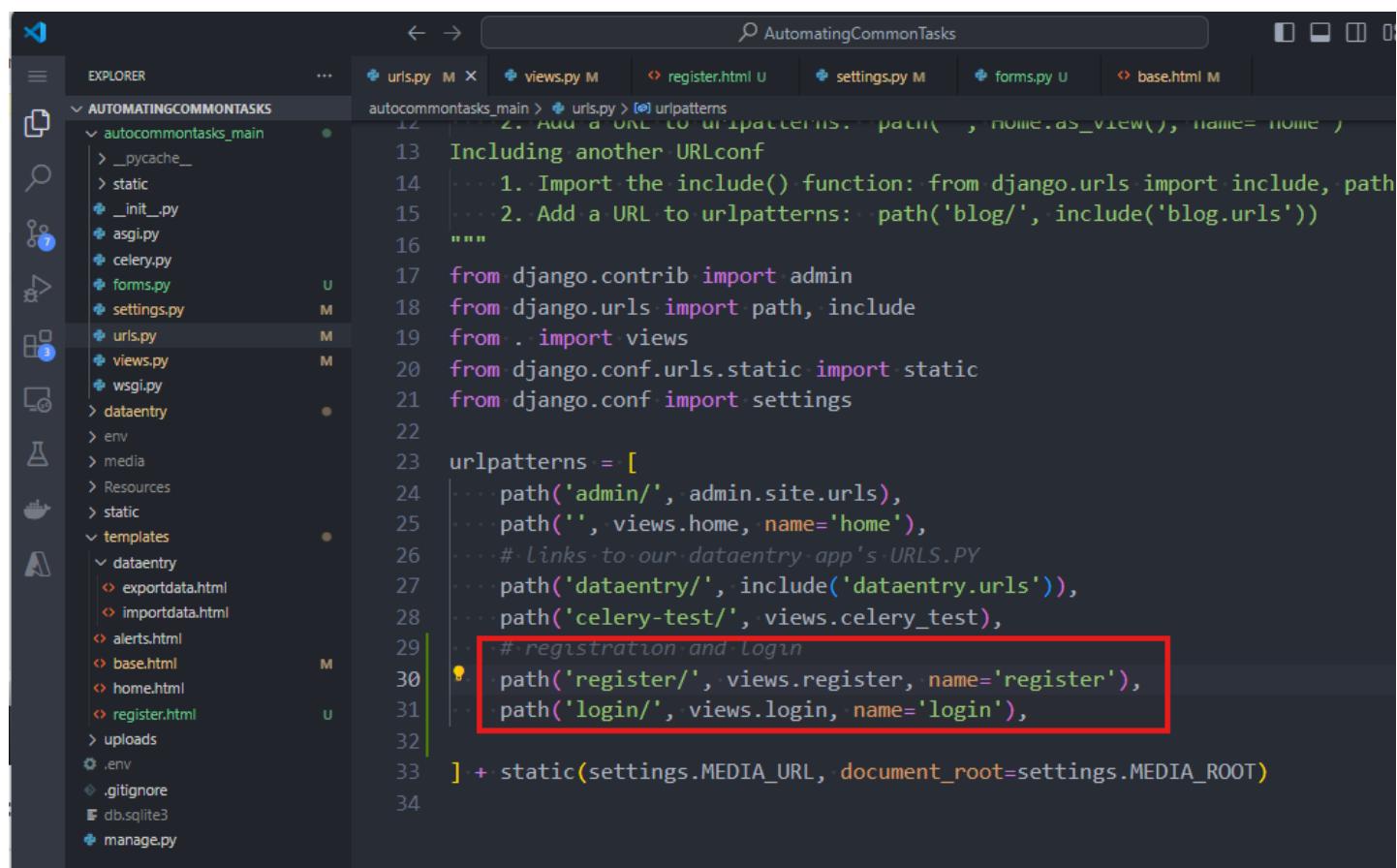


Topic: Login / Logout & User Restrictions Module Part 15

Speaker: Udemy Instructor Rathan Kumar / Notebook: Django: Automating Common Tasks



1. We completed the REGISTER module previously. To make your Login link work, we need to update our URLs.PY



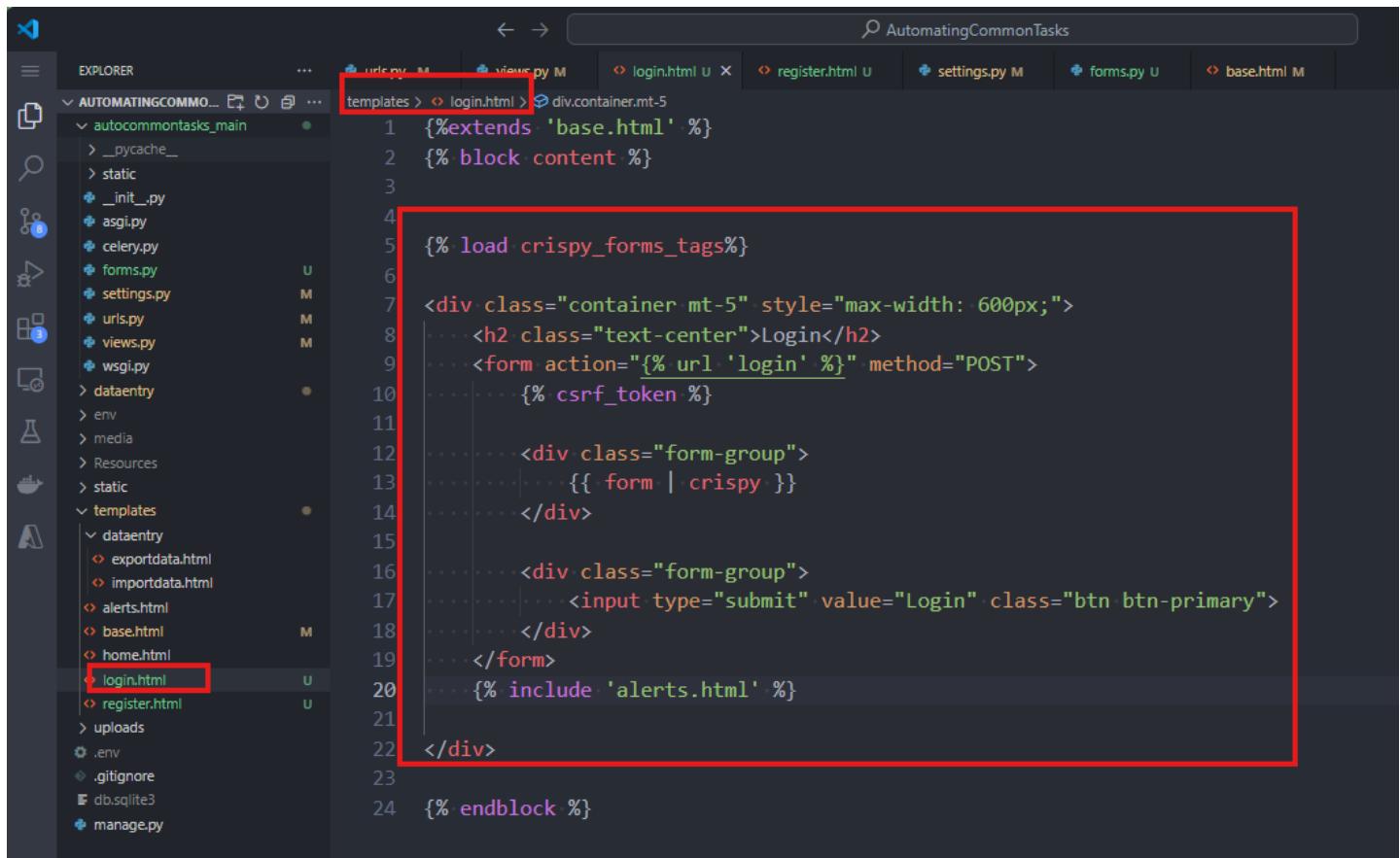
```
12 1. Add a URL to urlpatterns: "path('', home.as_view(), name='home')"
13 Including another URLconf
14 1. Import the include() function: from django.urls import include, path
15 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include
19 from . import views
20 from django.conf.urls.static import static
21 from django.conf import settings
22
23 urlpatterns = [
24     path('admin/', admin.site.urls),
25     path('', views.home, name='home'),
26     # Links to our dataentry app's URLs.PY
27     path('dataentry/', include('dataentry.urls')),
28     path('celery-test/', views.celery_test),
29     # registration and login
30     path('register/', views.register, name='register'),
31     path('login/', views.login, name='login'),
32 ]
33 ] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
34
```

2. Create our LOGIN function in the VIEWS.PY

```
def login(request):
    return render(request, 'login.html')

def logout(request):
    return render(request, 'home.html')
```

3. Create our LOGIN.HTML

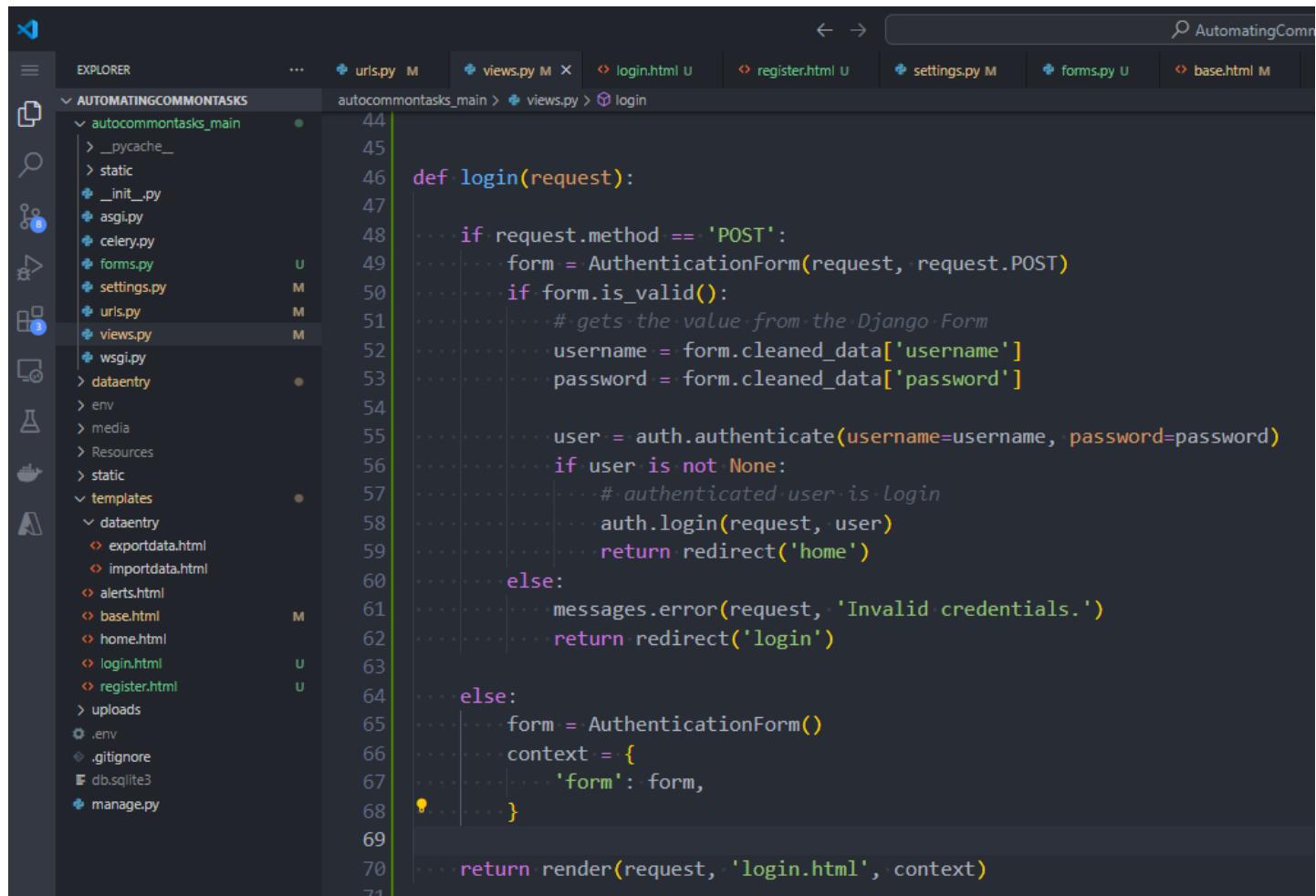


```
1  {%extends 'base.html'%}
2  {%block content%}
3
4
5  {%load crispy_forms_tags%}
6
7  <div class="container mt-5" style="max-width: 600px;">
8      <h2 class="text-center">Login</h2>
9      <form action="{% url 'login' %}" method="POST">
10         {% csrf_token %}
11
12         <div class="form-group">
13             {{ form | crispy }}
14         </div>
15
16         <div class="form-group">
17             <input type="submit" value="Login" class="btn btn-primary">
18         </div>
19     </form>
20     {% include 'alerts.html' %}
21
22 </div>
23
24  {%endblock%}
```

4. Update our BASE.HTML

```
<div>
    <a href="{% url 'login' %}" class="float-right text-decoration-none text-dark">Login</a>
    &nbsp;<a href="{% url 'register' %}" class="float-right btn btn-primary">Register</a>
</div>
```

5. Update our VIEWS.PY

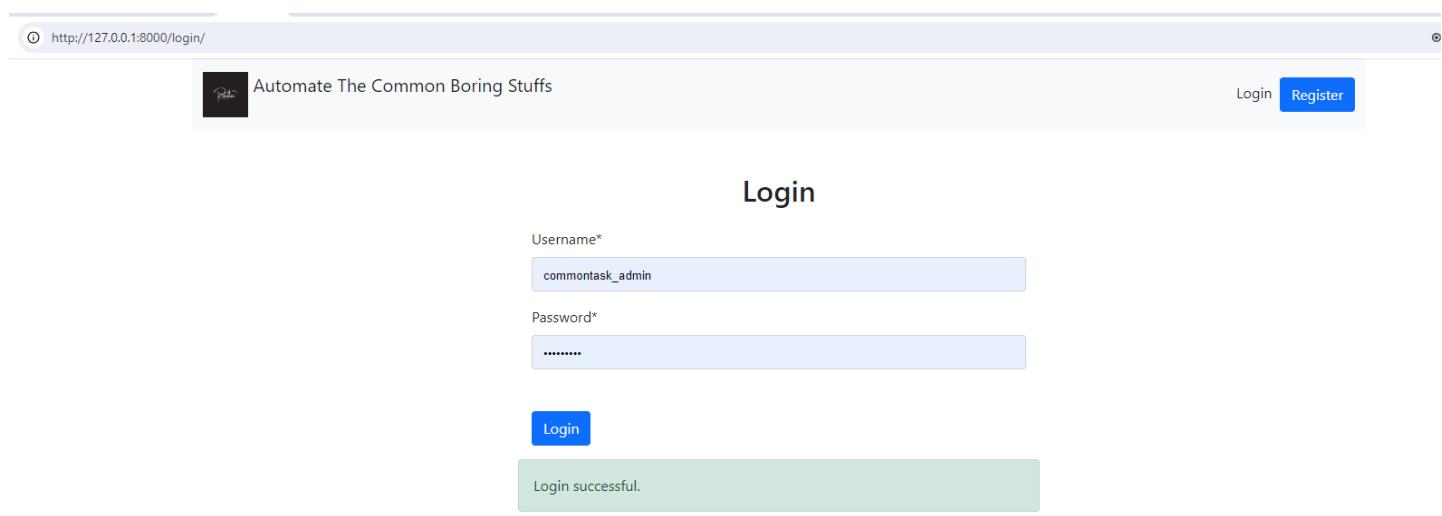


```

44
45
46 def login(request):
47
48     if request.method == 'POST':
49         form = AuthenticationForm(request, request.POST)
50         if form.is_valid():
51             # gets the value from the Django Form
52             username = form.cleaned_data['username']
53             password = form.cleaned_data['password']
54
55             user = auth.authenticate(username=username, password=password)
56             if user is not None:
57                 # authenticated user is login
58                 auth.login(request, user)
59                 return redirect('home')
60             else:
61                 messages.error(request, 'Invalid credentials.')
62                 return redirect('login')
63
64     else:
65         form = AuthenticationForm()
66         context = {
67             'form': form,
68         }
69
70     return render(request, 'login.html', context)
71

```

Run the code:



Automate The Common Boring Stuffs

Login

Username*

commenttask_admin

Password*

.....

Login

Login successful.

Copyright © 2024 Automate The Common Boring Stuffs | By [Rosolie Lim](#)

6. We should not show REGISTER button if we are logged in, so we update our BASE.HTML AS:

```

<nav class="navbar navbar-light bg-light">
  <div class="container">
    <a class="navbar-brand" href="{% url 'home' %}">
      
      Automate The Common Boring Stuffs
    </a>
    {% if user.is_authenticated %}
      <!-- show Logout button -->
      <div>
        Logged in as: {{user}}
        <a href="{% url 'logout' %}" class="float-right btn btn-outline-danger">Logout</a>
      </div>
    {% else %}
      <div>
        <a href="{% url 'login' %}" class="float-right text-decoration-none text-dark">Login</a>
        &nbsp;<a href="{% url 'register' %}" class="float-right btn btn-primary">Register</a>
      </div>
    {% endif %}
  </div>
</nav>

```

7. To make our logout button work, we update our VIEWS.PY AS:

```

def logout(request):
    auth.logout(request)
    return redirect('home')

```

8. Test our code:



Automate The Boring Stuff With Django

Import Data

Imports data from CSV file to any table.

Export Data

Exports data from a specific table to a CSV file.

Bulk Emails

Sends bulk emails to users

Email Tracking

Tracks emails and shows open and click rate

Compress Images

Compresses Image files and reduces the size

Web Scraping

Scrapes websites for useful information

Copyright © 2024 Automate The Common Boring Stuffs | By [Rosilie Lim](#)

9. To make see only the tools when they log in, we must update our HOME.HTML. We add the IF ELSE statement so that the tools would only appear if we log in otherwise we see the homepage with some information instead.

AutomatingCommonTasks

EXPLORER

- AUTOMATINGCOMMONTASKS
 - autocommontasks_main
 - static
 - css
 - # custom.css
 - images
 - js
 - __init__.py
 - asgi.py
 - celery.py
 - forms.py
 - settings.py
 - urls.py
 - views.py
 - wsgi.py
 - dataentry
 - env
 - media
 - Resources
 - static
 - templates
 - dataentry
 - exportdata.html
 - importdata.html
 - alerts.html
 - base.html
 - home.html
 - login.html
 - register.html
 - uploads
 - .env
 - .gitignore
 - db.sqlite3
 - manage.py

templates > home.html > div.container

```

1  {% extends 'base.html' %} 
2  {% block content %} 
3  <div class="container" style="padding: 50px; margin-top: 5%;> 
4  ....!<!-- Only logged-in user can see the tools--> 
5  <% if user.is_authenticated %> 
6  <h2 class="text-center">Automate The Common Boring Stuffs with Django</h2> 
7  <div class="row pt-5"> 
8  ....!<!-- Import data --> 
9  <div class="col-md-4"> 
10 <a href="{% url 'import_data' %}" class="text-decoration-none text-dark"> 
11 <div class="card alert-primary shadow"> 
12 <div class="card-header"> 
13 <h4>Import Data</h4> 
14 </div> 
15 <div class="card-body"> 
16 <p class="card-text">Imports data from CSV file to any table.</p> 
17 </div> 
18 </div> 
19 </a> 
20 </div> 
21 </div> 
22 <!-- Export Data --> 
23 <div class="col-md-4"> 
24 <a href="{% url 'export_data' %}" class="text-decoration-none text-dark"> 
25 <div class="card alert-success shadow"> 
26 <div class="card-header"> 
27 <h4>Export Data</h4> 
28 </div>

```

AutomatingCommonTasks

EXPLORER

- AUTOMATINGCOMMONTASKS
 - autocommontasks_main
 - static
 - css
 - # custom.css
 - images
 - js
 - __init__.py
 - asgi.py
 - celery.py
 - forms.py
 - settings.py
 - urls.py
 - views.py
 - wsgi.py
 - dataentry
 - env
 - media
 - Resources
 - static
 - templates
 - dataentry
 - exportdata.html
 - importdata.html
 - alerts.html
 - base.html
 - home.html
 - login.html
 - register.html
 - uploads
 - .env
 - .gitignore
 - db.sqlite3
 - manage.py

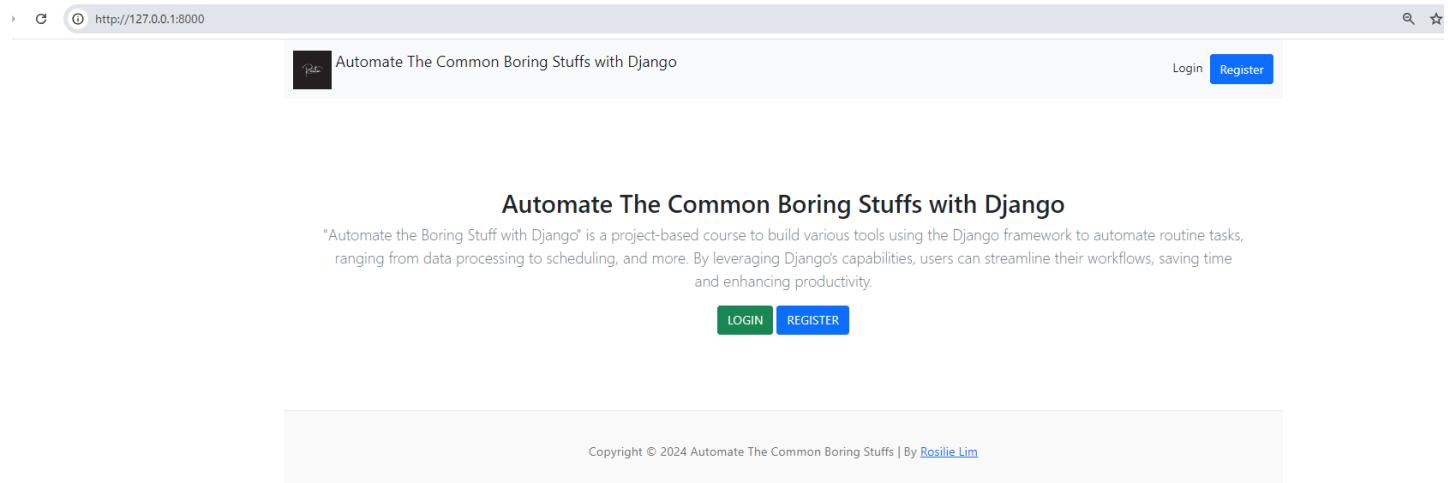
templates > home.html > div.container > div.row.pt-5 > div.col-md-4 > a.text-decoration-none.text-dark > div.card.alert-primary.shadow

```

3  <div class="container" style="padding: 50px; margin-top: 5%;> 
52 <div class="row pt-5"> 
81 ....!<!-- Web Scraping --> 
82 <div class="col-md-4"> 
83 <a href="#" class="text-decoration-none text-dark"> 
84 <div class="card alert-warning shadow"> 
85 <div class="card-header"> 
86 <h4>Web Scraping</h4> 
87 </div> 
88 <div class="card-body"> 
89 <p class="card-text">Scrapes websites for useful information</p> 
90 </div> 
91 </div> 
92 </a> 
93 </div> 
94 </div> 
95 &lt;% else %&gt; 
96 <!-- Home Page content if user is not logged-in--> 
97 <div class="home-info"> 
98 <h2 class="text-center">Automate The Common Boring Stuffs with Django</h2> 
99 <p class="lead text-center text-muted"> 
100 "Automate the Boring Stuff with Django" is a project-based course to build various tools using the Django framework to automate routine tasks, ranging from data processing to scheduling, and more. 
101 By leveraging Django's capabilities, users can streamline their workflows, saving time and enhancing productivity. 
102 </p> 
103 <div style="display: flex; justify-content: center;"> 
104 <a href="{% url 'login' %}" class="btn btn-success">LOGIN</a> 
105 &nbsp;<a href="{% url 'register' %}" class="btn btn-primary">REGISTER</a> 
106 </div> 
107 </div> 
108 <% endif %> 
109 </div> 
110 </div> 
111 </div> 
112 <% endif %> 
113 </div> 
114 <% endif %> 
115 </div> 
116 &lt;% endblock %&gt;

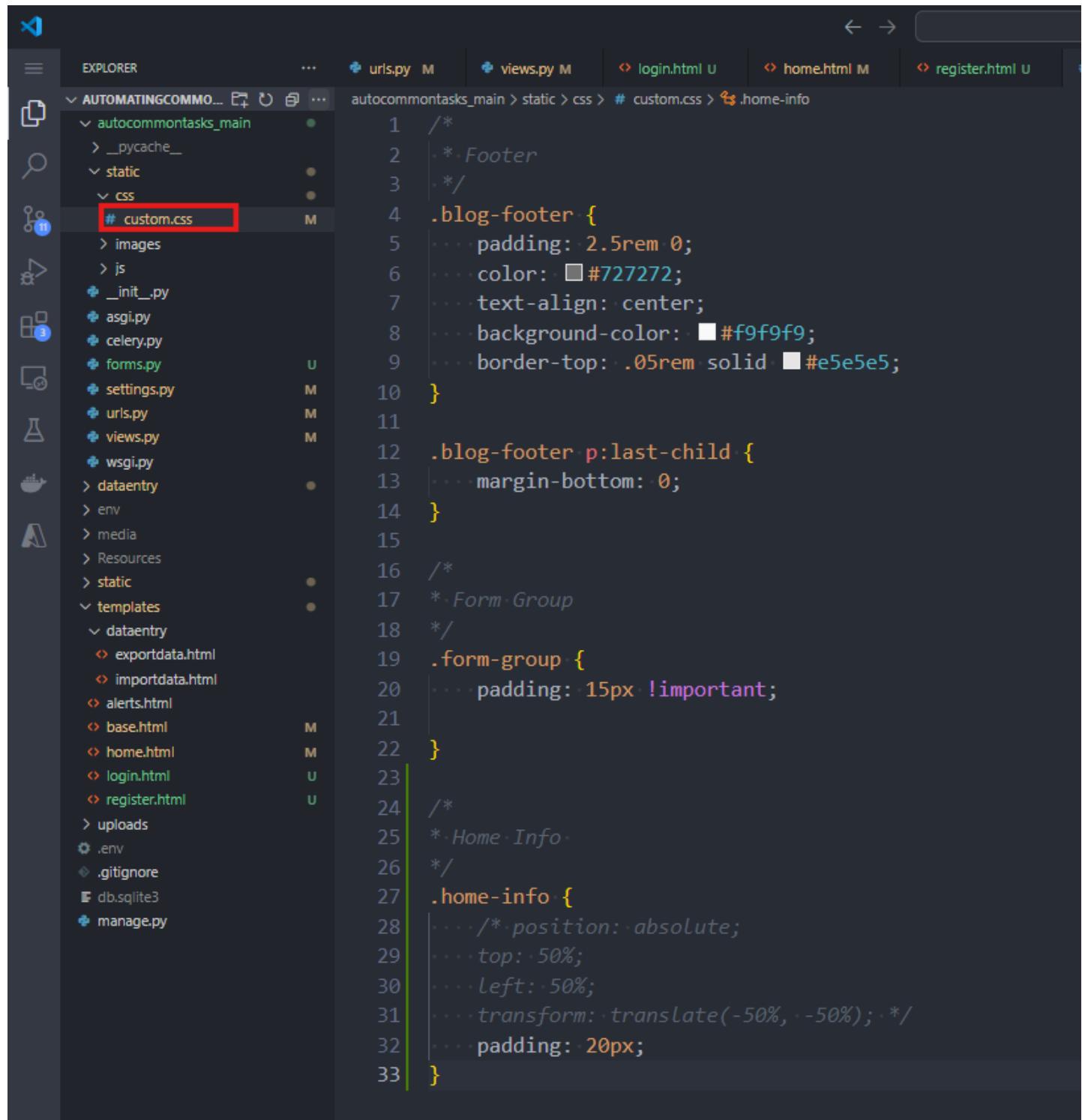
```

9. Load the page:



From here, you can login or register to see the Django tools.

We have made some custom CSS for our footer and home-info: We run COLLECTSTATIC to make sure that this external CSS is added to our root folder for deployment.



The image shows a screenshot of a code editor, likely Visual Studio Code, displaying a file structure and a CSS file content. The file structure on the left shows a project named 'AUTOMATINGCOMMONTASKS' with various files and folders. The 'static/css' folder contains a file named '# custom.css', which is highlighted with a red box. The content of this file is a CSS code snippet:

```
1  /*
2  * .Footer
3  */
4 .blog-footer {
5   padding: 2.5rem 0;
6   color: #727272;
7   text-align: center;
8   background-color: #f9f9f9;
9   border-top: .05rem solid #e5e5e5;
10 }
11
12 .blog-footer p:last-child {
13   margin-bottom: 0;
14 }
15
16 /*
17 * .Form-Group
18 */
19 .form-group {
20   padding: 15px !important;
21 }
22
23 /*
24 * .Home-Info
25 */
26
27 .home-info {
28   /*position: absolute;
29   top: 50%;
30   left: 50%;
31   transform: translate(-50%, -50%); */
32   padding: 20px;
33 }
```