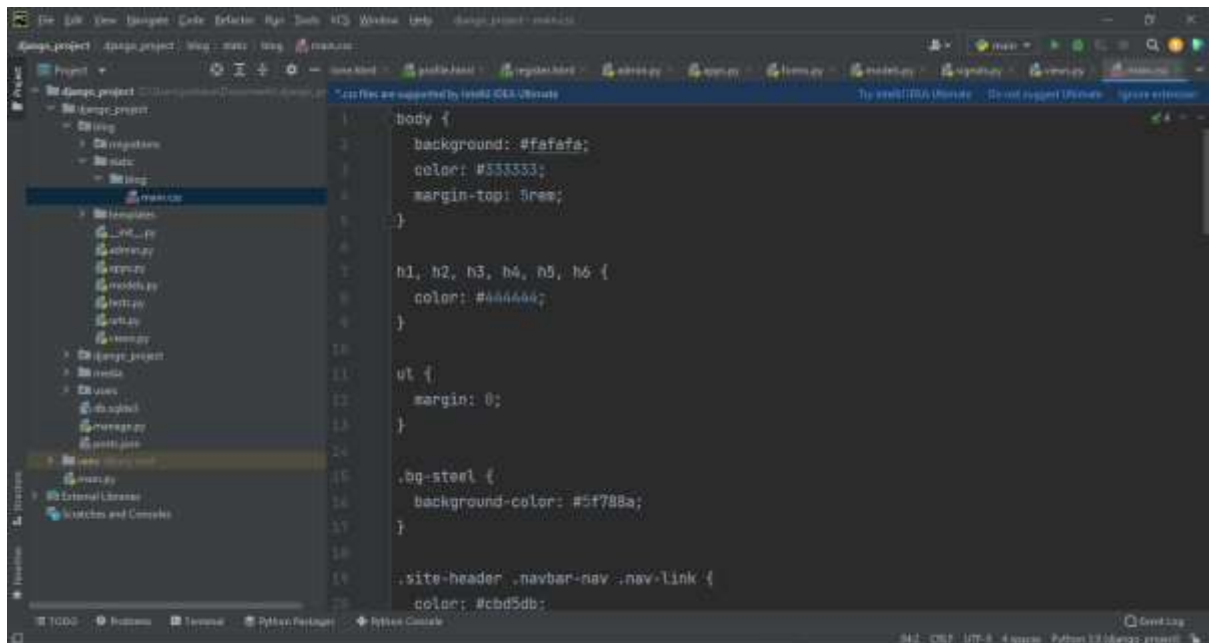


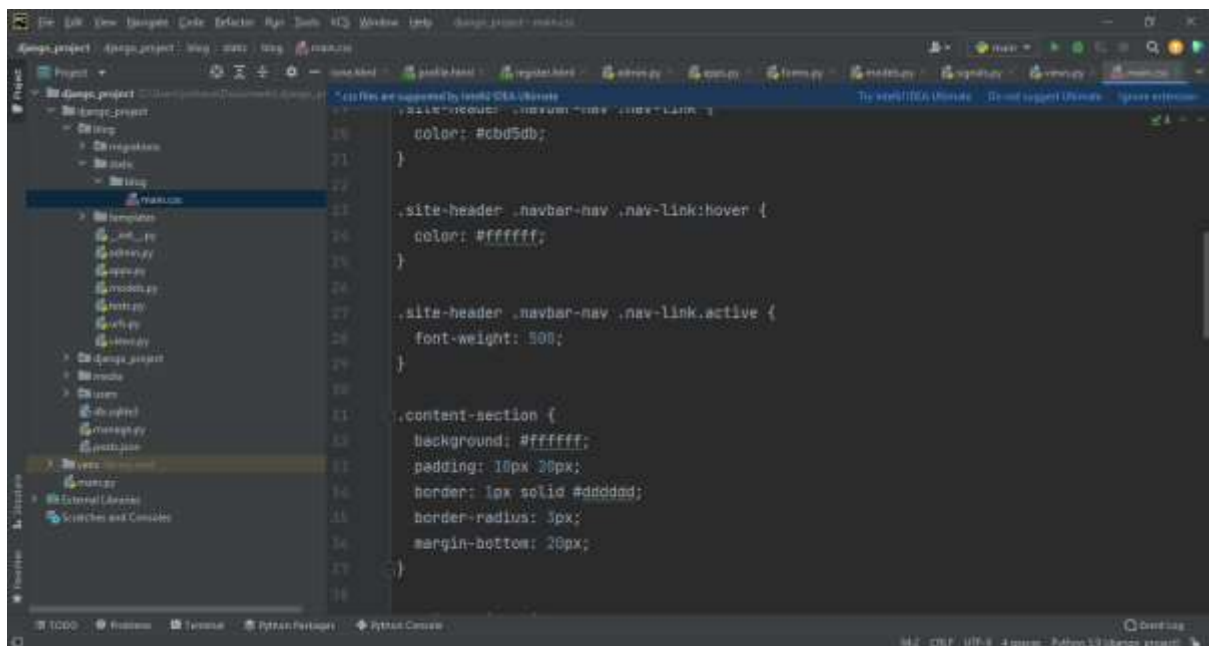
Name: Arapoc, Jumaine Kevin

Year & Section: 3-BSCS-B

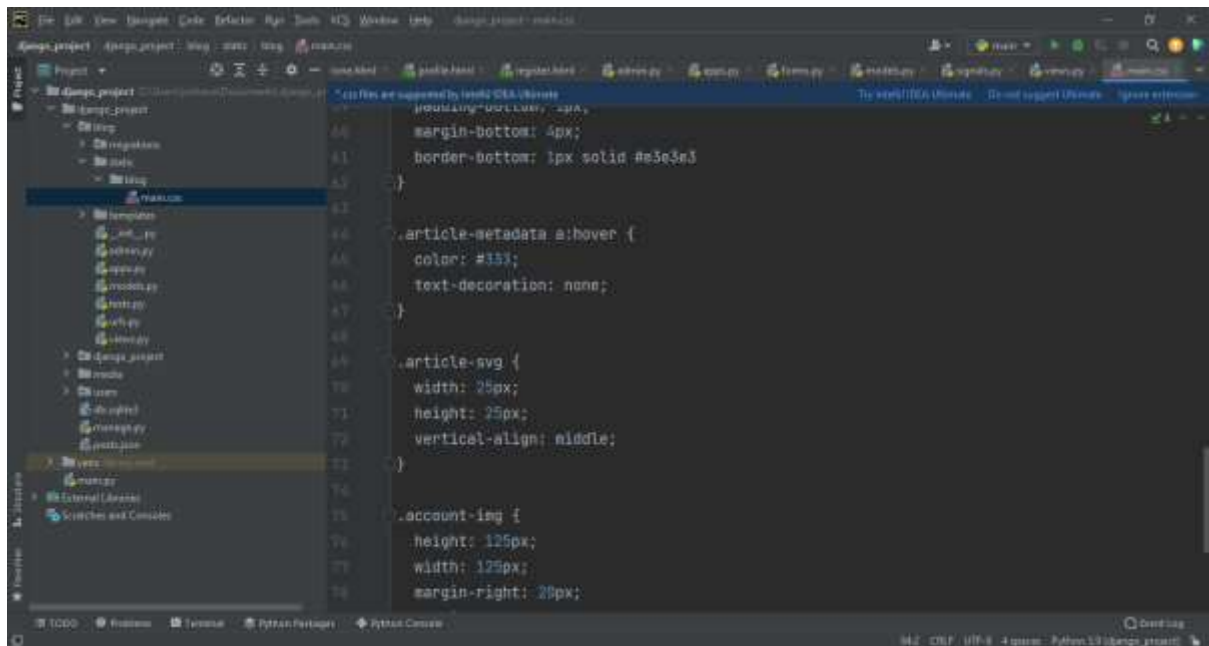
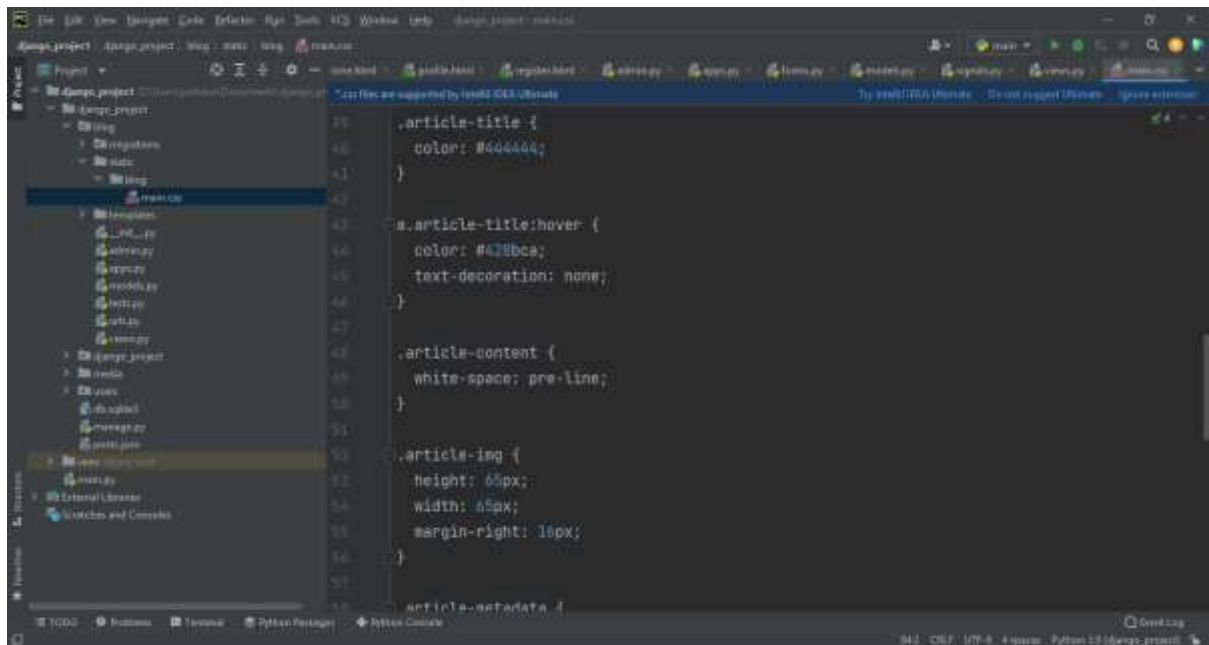
DJANGO BLOG ACTIVITY

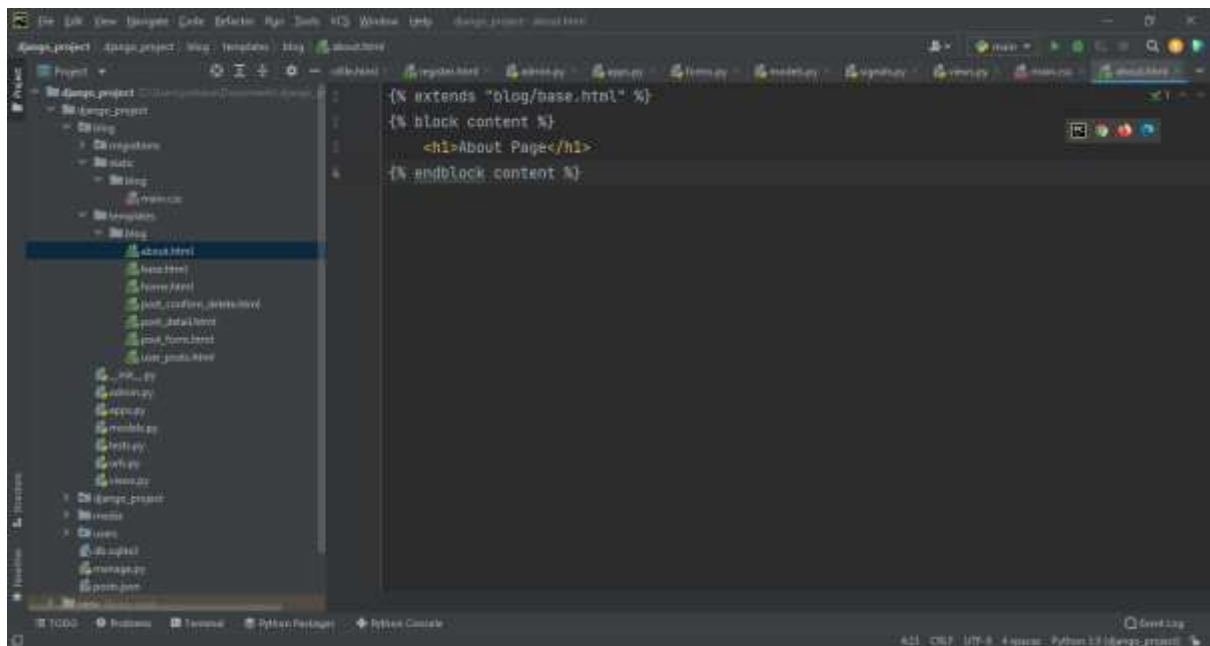
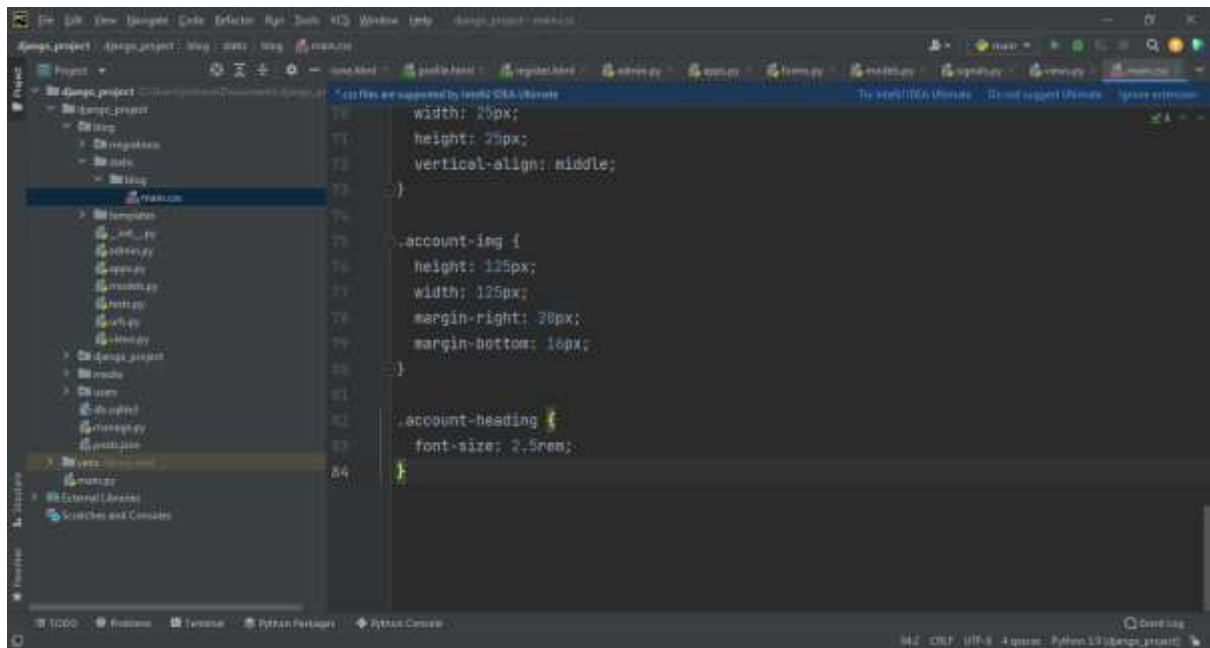


```
1 body {  
2     background: #fafafa;  
3     color: #333333;  
4     margin-top: 5px;  
5 }  
6  
7 h1, h2, h3, h4, h5, h6 {  
8     color: #444444;  
9 }  
10  
11 ul {  
12     margin: 0;  
13 }  
14  
15 .bg-steel {  
16     background-color: #5f788a;  
17 }  
18  
19 .site-header .navbar-nav .nav-link {  
20     color: #cbd5db;
```



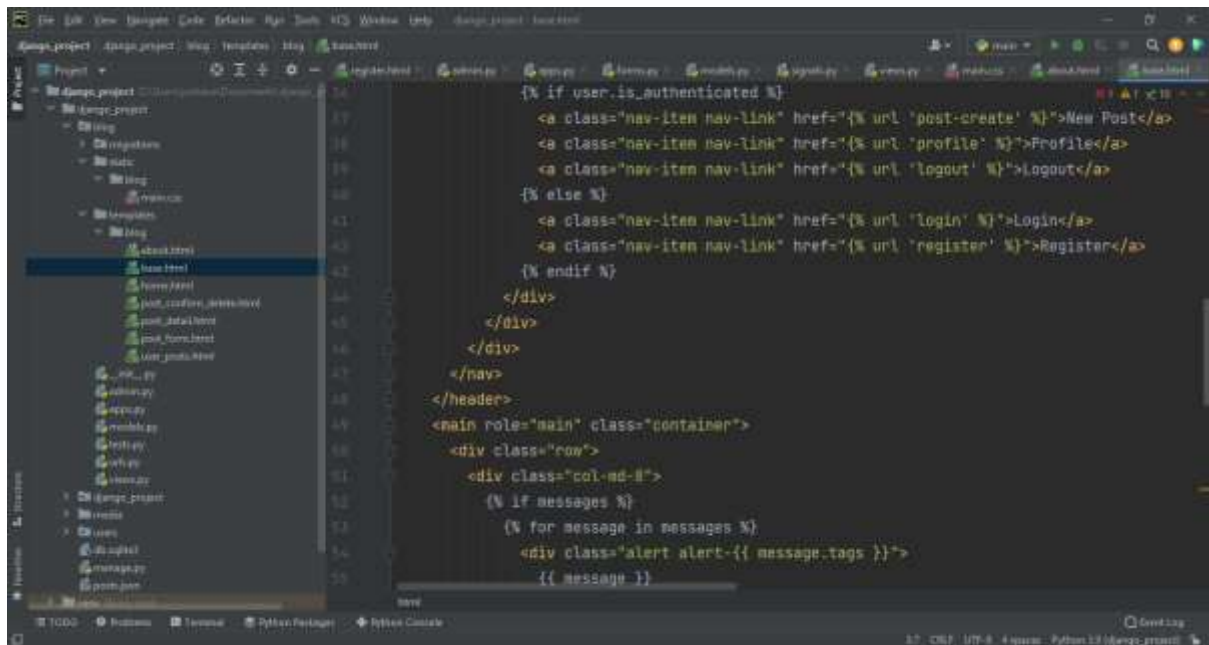
```
21     color: #cbd5db;  
22 }  
23  
24 .site-header .navbar-nav .nav-link:hover {  
25     color: #ffffff;  
26 }  
27  
28 .site-header .navbar-nav .nav-link.active {  
29     font-weight: 500;  
30 }  
31  
32 .content-section {  
33     background: #ffffff;  
34     padding: 10px 30px;  
35     border: 1px solid #d9d9d9;  
36     border-radius: 3px;  
37     margin-bottom: 20px;  
38 }
```



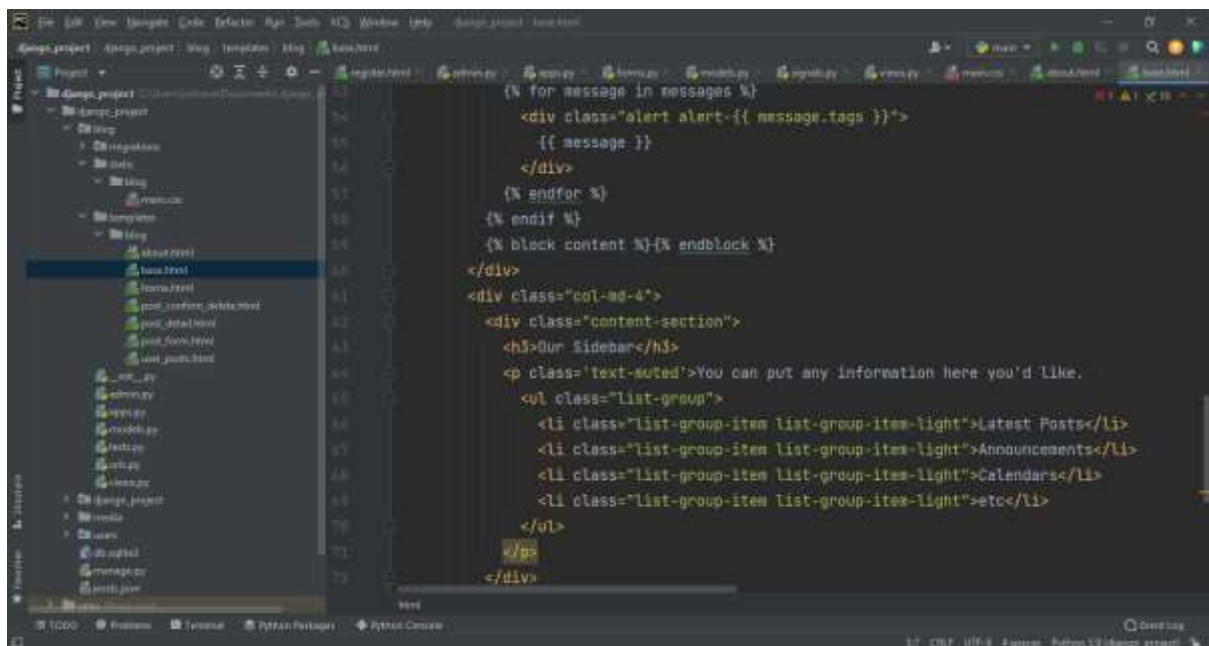


```
1 {% load static %}
2 <!DOCTYPE html>
3 <html>
4 <head>
5
6     <!-- Required meta tags -->
7     <meta charset="utf-8">
8     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
9
10
11     <!-- Bootstrap CSS -->
12     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
13
14     <link rel="stylesheet" type="text/css" href="{% static 'blog/main.css' %}">
15
16     {% if title %}
17     <title>Django Blog - {{ title }}</title>
18     {% else %}
19     <title>Django Blog</title>
20     {% endif %}
21 </head>
22
23 <body>
```

```
24 <header class="site-header">
25     <nav class="navbar navbar-expand-md navbar-dark bg-steal fixed-top">
26         <div class="container">
27             <a class="navbar-brand mr-4" href="{% url 'blog-home' %}">Django Blog</a>
28             <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarToggler"
29                 <span class="navbar-toggler-icon"></span>
30             </button>
31             <div class="collapse navbar-collapse" id="navbarToggler">
32                 <div class="navbar-nav mr-auto">
33                     <a class="nav-item nav-link" href="{% url 'blog-home' %}">Home</a>
34                     <a class="nav-item nav-link" href="{% url 'blog-about' %}">About</a>
35                 </div>
36                 <!-- Navbar Right Side -->
37                 <div class="navbar-nav">
38                     {% if user.is_authenticated %}
39                     <a class="nav-item nav-link" href="{% url 'post-create' %}">New Post</a>
40                     <a class="nav-item nav-link" href="{% url 'profile' %}">Profile</a>
41                     </div>
42             </div>
43         </div>
44     </nav>
45 </header>
```



```
1  {% if user.is_authenticated %}
2
3  <a class="nav-item nav-link" href="{% url 'post-create' %}">New Post</a>
4  <a class="nav-item nav-link" href="{% url 'profile' %}">Profile</a>
5  <a class="nav-item nav-link" href="{% url 'logout' %}">Logout</a>
6  {% else %}
7
8  <a class="nav-item nav-link" href="{% url 'login' %}">Login</a>
9  <a class="nav-item nav-link" href="{% url 'register' %}">Register</a>
10 {% endif %}
11 </div>
12 </div>
13 </nav>
14 </header>
15 <main role="main" class="container">
16 <div class="row">
17 <div class="col-md-8">
18 {% if messages %}
19 {% for message in messages %}
20 <div class="alert alert-{{ message.tags }}">
21 {{ message }}
22
```



```
23 </div>
24 {% endfor %}
25 {% endif %}
26 {% block content %}{% endblock %}
27 </div>
28 <div class="col-md-4">
29 <div class="content-section">
30 <h3>Our Sidebar</h3>
31 <p class="text-muted">You can put any information here you'd like.</p>
32 <ul class="list-group">
33 <li class="list-group-item list-group-item-light">Latest Posts</li>
34 <li class="list-group-item list-group-item-light">Announcements</li>
35 <li class="list-group-item list-group-item-light">Calendars</li>
36 <li class="list-group-item list-group-item-light">etc</li>
37 </ul>
38 </div>
39 </div>
```



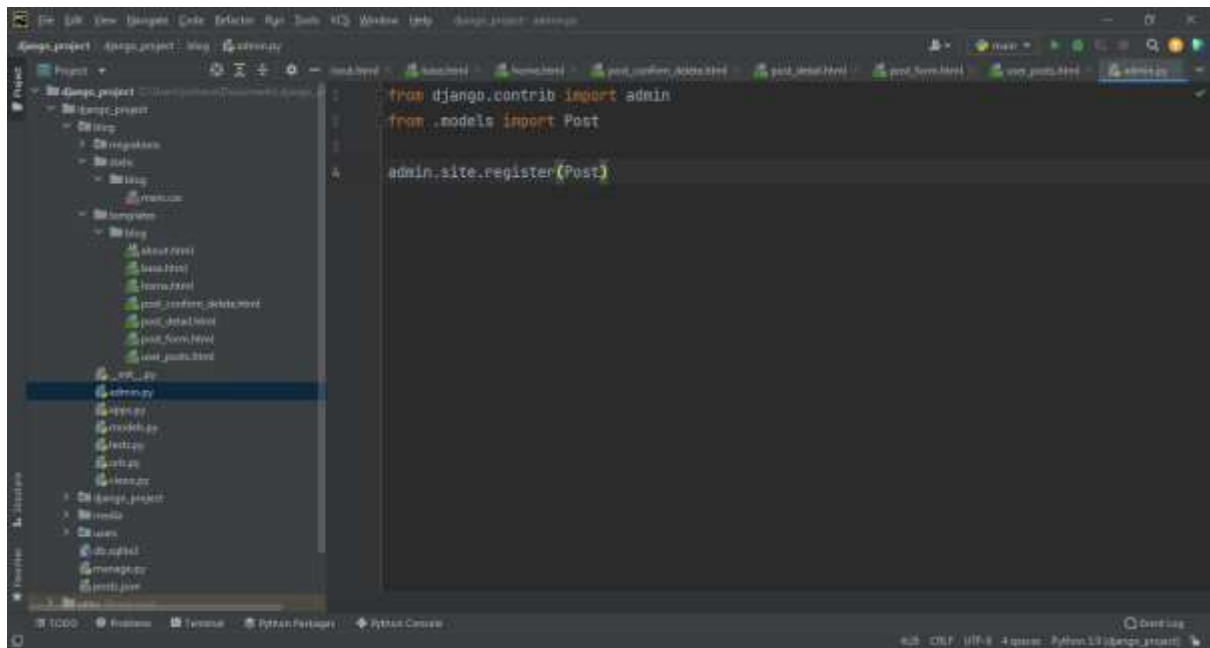
```
1 {% extends "blog/base.html" %}
2 {% block content %}
3
4 <article class="media content-section">
5   
6   <div class="media-body">
7     <div class="article-metadata">
8       <a class="mr-2" href="{% url 'user-posts' object.author.username %}">{{ object.author.username }}</a>
9       <small class="text-muted">{{ object.date_posted|date:'F d, Y' }}</small>
10       {% if object.author == user %}
11         <div>
12           <a class="btn btn-secondary btn-sm mt-1 mb-1" href="{% url 'post-update' object.id %}">Update</a>
13           <a class="btn btn-danger btn-sm mt-1 mb-1" href="{% url 'post-delete' object.id %}">Delete</a>
14         </div>
15       {% endif %}
16     </div>
17     <h2 class="article-title">{{ object.title }}</h2>
18     <p class="article-content">{{ object.content }}</p>
19   </div>
20 </article>
21 {% endblock content %}
```

```
1 {% extends "blog/base.html" %}
2 {% load crispy_forms_tags %}
3 {% block content %}
4   <div class="content-section">
5     <form method="POST">
6       {% csrf_token %}
7       <fieldset class="form-group">
8         <legend class="border-bottom mb-4">Blog Post</legend>
9         {{ form|crispy }}
10       </fieldset>
11       <div class="form-group">
12         <button class="btn btn-outline-info" type="submit">Post</button>
13       </div>
14     </form>
15   </div>
16 {% endblock content %}
```



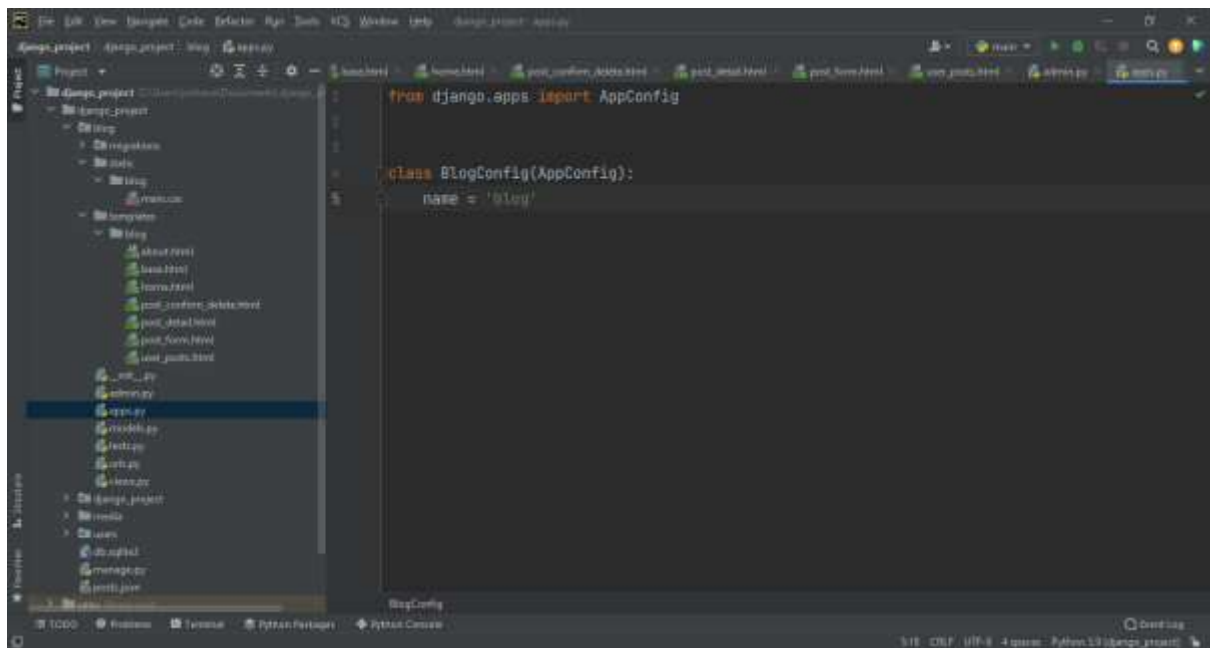
```
1 {% extends "blog/base.html" %}
2 {% block content %}
3     <h1 class="mb-3">Posts by {{ view.kwargs.username }} ({{ page_obj.paginator.count }})
4     {% for post in posts %}
5         <article class="media content-section">
6             
8                 <div class="article-metadata">
9                     <a class="mr-2" href="{% url 'user-posts' post.author.username %}">{{ post.author.username }}
10                     <small class="text-muted">{{ post.data_posted|date:"F d, Y" }}</small>
11                 </div>
12                 <h2><a class="article-title" href="{% url 'post-detail' post.id %}">{{ post.title }}
13                 <p class="article-content">{{ post.content }}</p>
14             </div>
15         </article>
16     {% endfor %}
17     {% if is_paginated %}
18
19     {% if page_obj.has_previous %}
20         <a class="btn btn-outline-info mb-4" href="/page1">First</a>
```

```
21
22         <a class="btn btn-outline-info mb-4" href="/page1">First</a>
23         <a class="btn btn-outline-info mb-4" href="/page={{ page_obj.previous_page_number }}">
24     {% endif %}
25
26     {% for num in page_obj.paginator.page_range %}
27     {% if page_obj.number == num %}
28         <a class="btn btn-info mb-4" href="/page={{ num }}">{{ num }}</a>
29     {% elif num > page_obj.number|add:'-3' and num < page_obj.number|add:'3' %}
30         <a class="btn btn-outline-info mb-4" href="/page={{ num }}">{{ num }}</a>
31     {% endif %}
32     {% endfor %}
33
34     {% if page_obj.has_next %}
35         <a class="btn btn-outline-info mb-4" href="/page={{ page_obj.next_page_number }}">Next</a>
36         <a class="btn btn-outline-info mb-4" href="/page={{ page_obj.paginator.num_pages }}">
37     {% endif %}
38
39     {% endif %}
40 {% endblock content %}
```



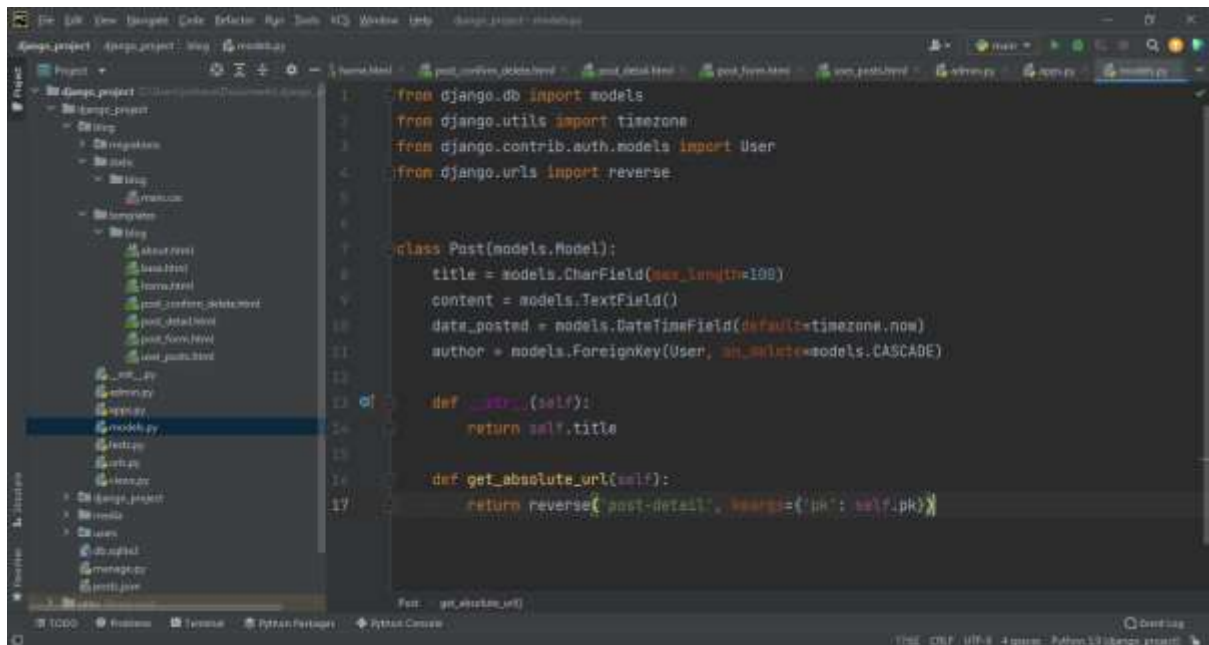
```
from django.contrib import admin
from .models import Post

admin.site.register(Post)
```

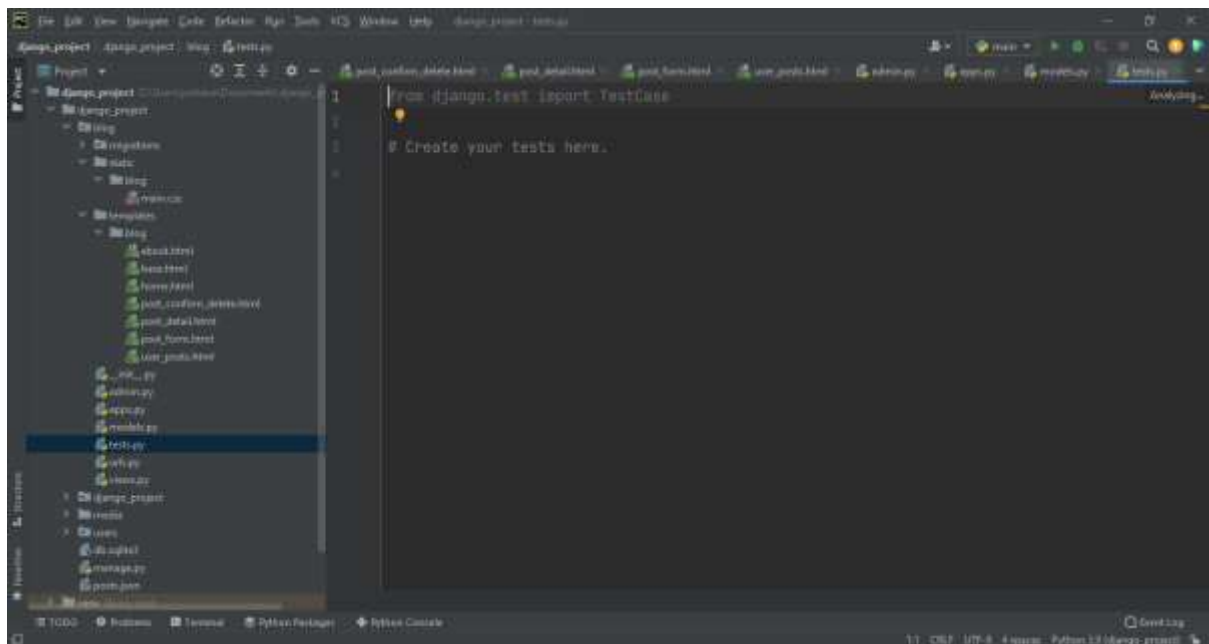


```
from django.apps import AppConfig

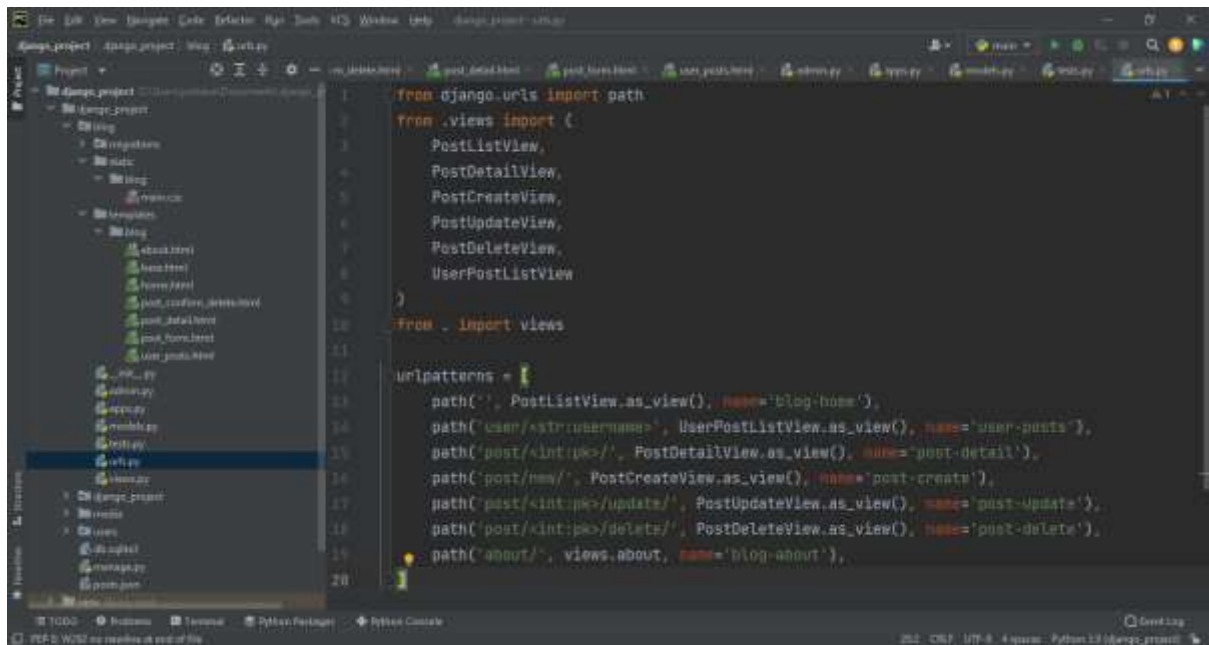
class BlogConfig(AppConfig):
    name = 'Blog'
```



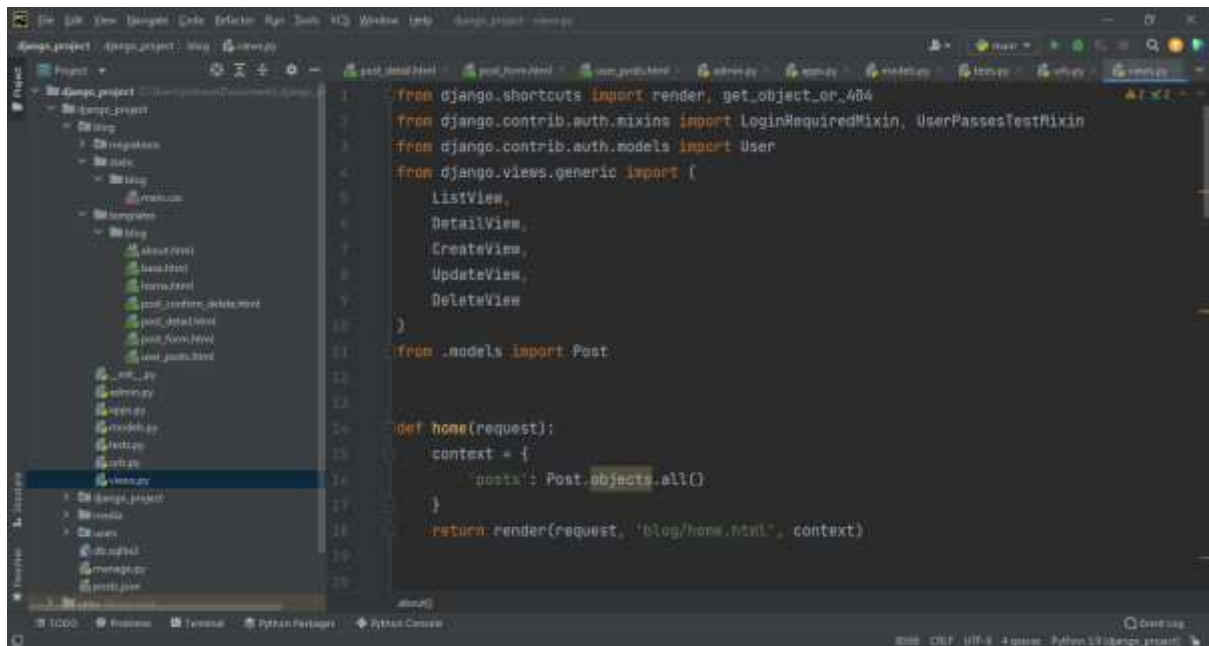
```
1 from django.db import models
2 from django.utils import timezone
3 from django.contrib.auth.models import User
4 from django.urls import reverse
5
6
7 class Post(models.Model):
8     title = models.CharField(max_length=100)
9     content = models.TextField()
10     date_posted = models.DateTimeField(default=timezone.now)
11     author = models.ForeignKey(User, on_delete=models.CASCADE)
12
13     def __str__(self):
14         return self.title
15
16     def get_absolute_url(self):
17         return reverse('post-detail', kwargs={'pk': self.pk})
```



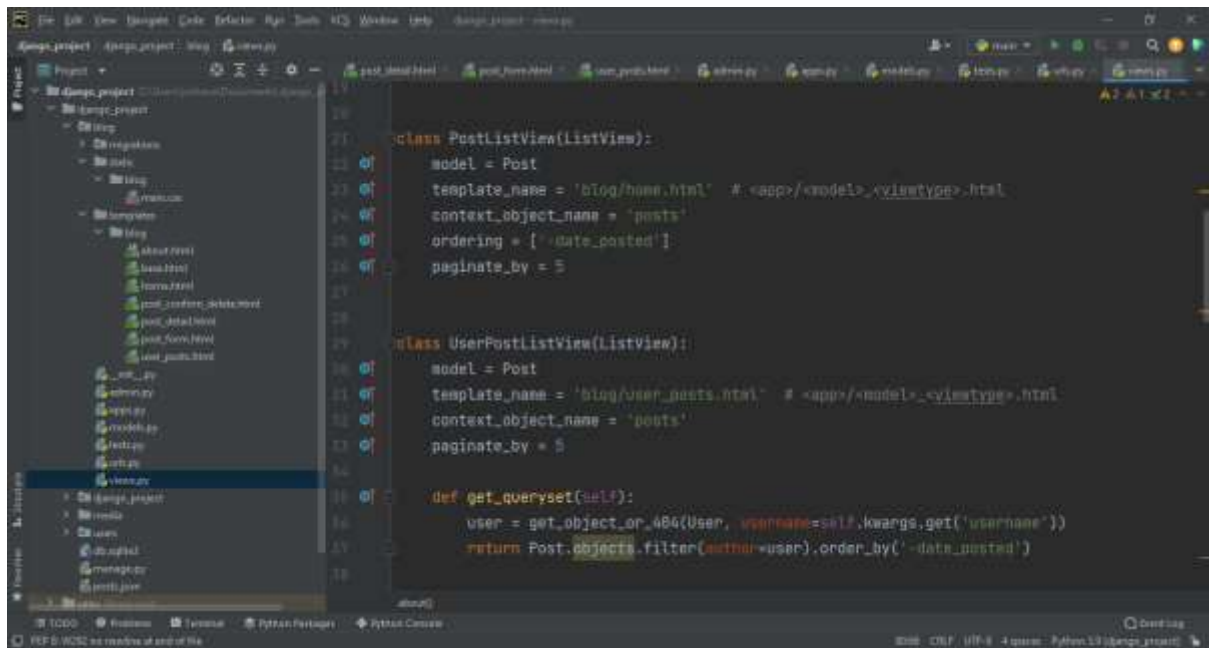
```
1 from django.test import TestCase
2
3 # Create your tests here.
```



```
1 from django.urls import path
2 from .views import (
3     PostListView,
4     PostDetailView,
5     PostCreateView,
6     PostUpdateView,
7     PostDeleteView,
8     UserPostListView
9 )
10 from . import views
11
12 urlpatterns = [
13     path('', PostListView.as_view(), name='blog-home'),
14     path('user/<int:pk>', UserPostListView.as_view(), name='user-posts'),
15     path('post/<int:pk>', PostDetailView.as_view(), name='post-detail'),
16     path('post/new/', PostCreateView.as_view(), name='post-create'),
17     path('post/<int:pk>/update/', PostUpdateView.as_view(), name='post-update'),
18     path('post/<int:pk>/delete/', PostDeleteView.as_view(), name='post-delete'),
19     path('about/', views.about, name='blog-about'),
20 ]
```

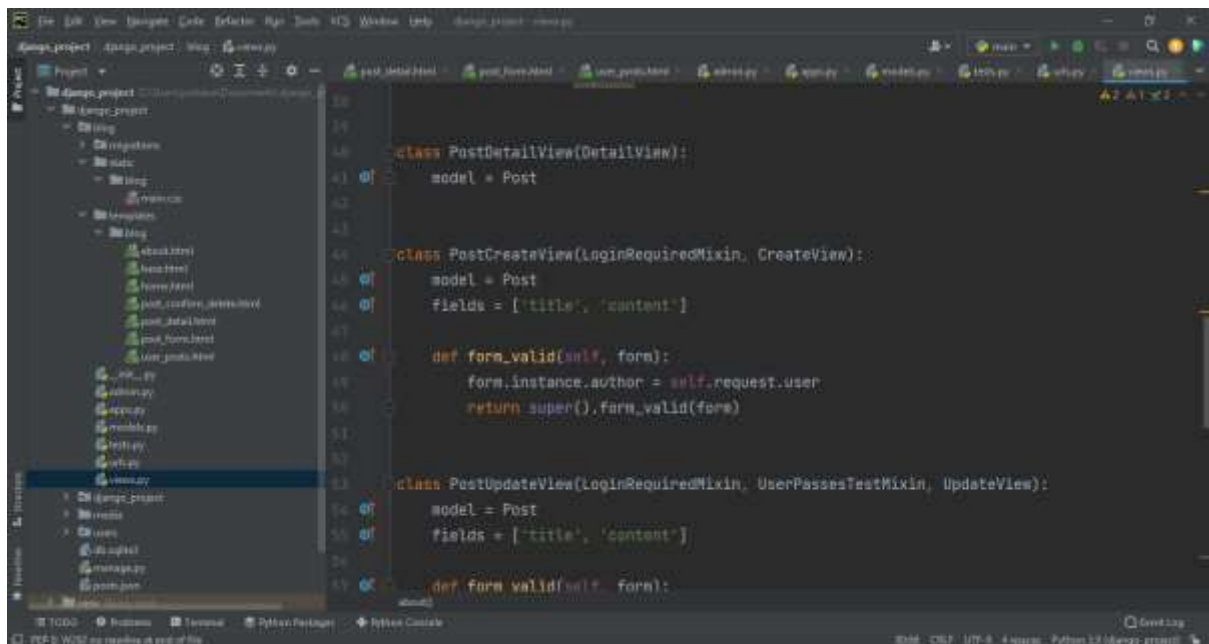


```
1 from django.shortcuts import render, get_object_or_404
2 from django.contrib.auth.mixins import LoginRequiredMixin, UserPassesTestMixin
3 from django.contrib.auth.models import User
4 from django.views.generic import (
5     ListView,
6     DetailView,
7     CreateView,
8     UpdateView,
9     DeleteView
10 )
11 from .models import Post
12
13 def home(request):
14     context = {
15         'posts': Post.objects.all()
16     }
17     return render(request, 'blog/home.html', context)
```



The screenshot shows the PyCharm IDE with the `views.py` file open. The left sidebar displays the project structure, including `blog` and `blog.views`. The main editor area contains the following Python code:

```
11
12
13 class PostListView(ListView):
14     model = Post
15     template_name = 'blog/home.html' # <app>/<model>_<viewtype>.html
16     context_object_name = 'posts'
17     ordering = ['-date_posted']
18     paginate_by = 5
19
20
21 class UserPostListView(ListView):
22     model = Post
23     template_name = 'blog/user_posts.html' # <app>/<model>_<viewtype>.html
24     context_object_name = 'posts'
25     paginate_by = 5
26
27     def get_queryset(self):
28         user = get_object_or_404(User, username=self.kwargs.get('username'))
29         return Post.objects.filter(author=user).order_by('-date_posted')
```



The screenshot shows the PyCharm IDE with the `views.py` file open. The left sidebar displays the project structure, including `blog` and `blog.views`. The main editor area contains the following Python code:

```
30
31
32 class PostDetailView(DetailView):
33     model = Post
34
35
36 class PostCreateView(LoginRequiredMixin, CreateView):
37     model = Post
38     fields = ('title', 'content')
39
40     def form_valid(self, form):
41         form.instance.author = self.request.user
42         return super().form_valid(form)
43
44
45 class PostUpdateView(LoginRequiredMixin, UserPassesTestMixin, UpdateView):
46     model = Post
47     fields = ('title', 'content')
48
49     def form_valid(self, form):
```


The screenshot shows the PyCharm IDE with the 'views.py' file open. The left sidebar displays the project structure, including 'blog' and 'blog.urls'. The main editor area shows the following Python code:

```
class PostUpdateView(LoginRequiredMixin, UserPassesTestMixin, UpdateView):
    model = Post
    fields = ['title', 'content']

    def form_valid(self, form):
        form.instance.author = self.request.user
        return super().form_valid(form)

    def test_func(self):
        post = self.get_object()
        if self.request.user == post.author:
            return True
        return False

class PostDeleteView(LoginRequiredMixin, UserPassesTestMixin, DeleteView):
    model = Post
    success_url = '/'
```

The status bar at the bottom indicates the file is 'UTF-8' and 'Python 3.7'.

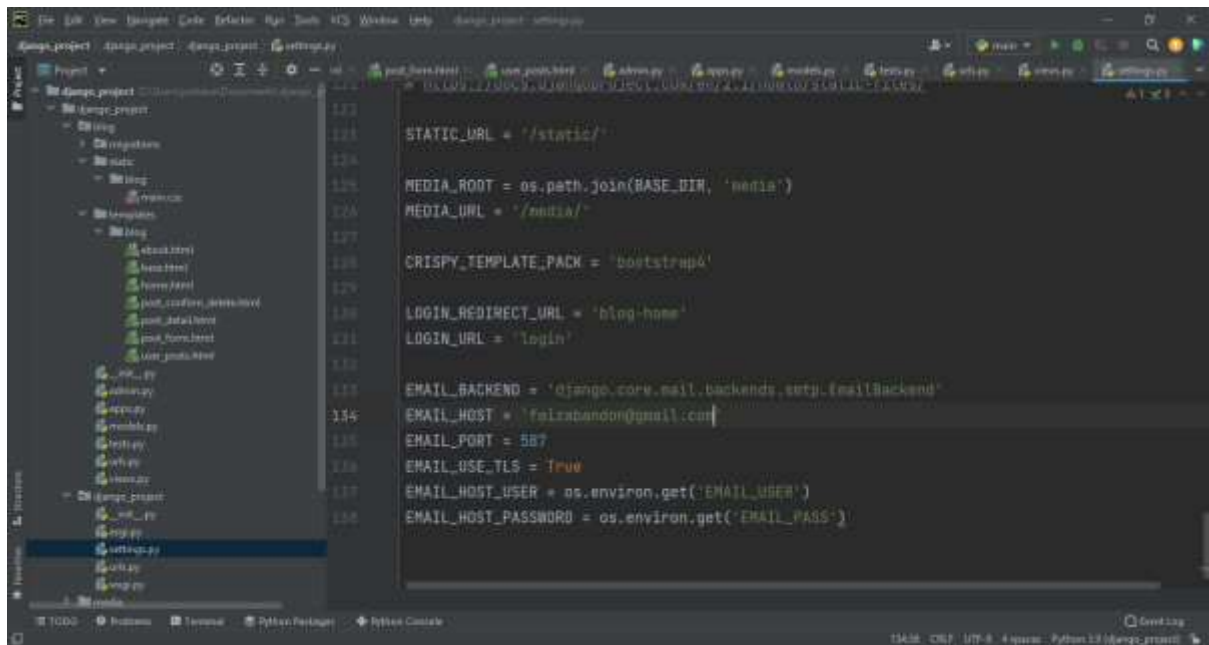
The screenshot shows the PyCharm IDE with the 'views.py' file open. The left sidebar displays the project structure, including 'blog' and 'blog.urls'. The main editor area shows the following Python code:

```
class PostDeleteView(LoginRequiredMixin, UserPassesTestMixin, DeleteView):
    model = Post
    success_url = '/'

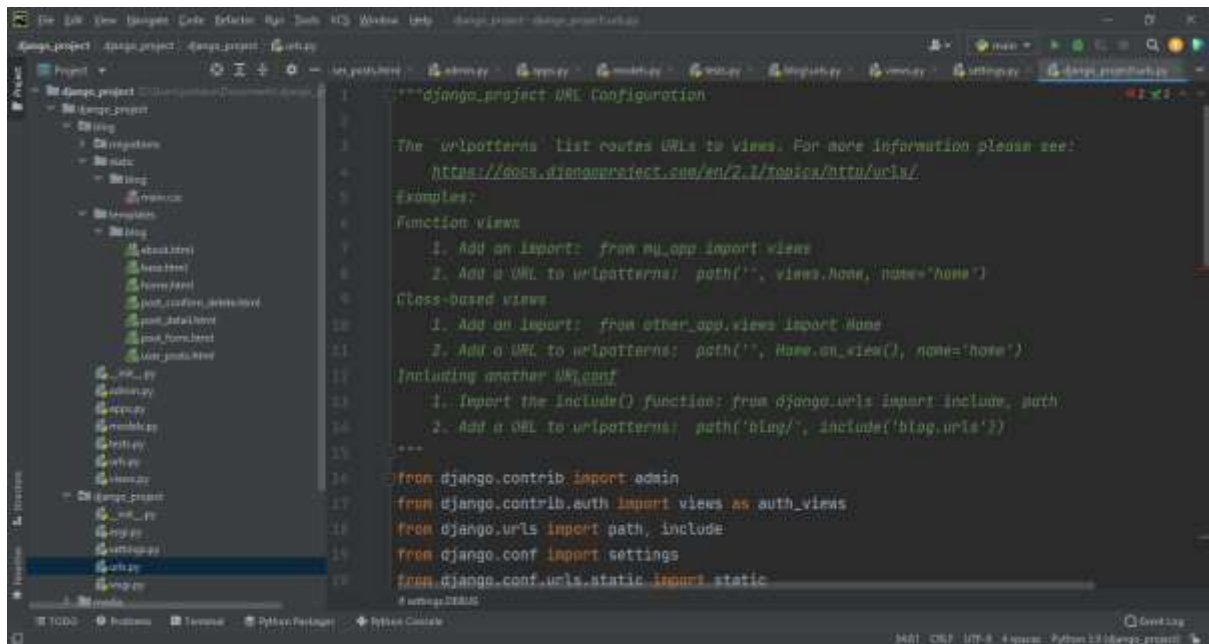
    def test_func(self):
        post = self.get_object()
        if self.request.user == post.author:
            return True
        return False

def about(request):
    return render(request, 'blog/about.html', {'title': 'About'})
```

The status bar at the bottom indicates the file is 'UTF-8' and 'Python 3.7'.



```
123 STATIC_URL = '/static/'
124
125 MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
126 MEDIA_URL = '/media/'
127
128 CRISPY_TEMPLATE_PACK = 'bootstrap4'
129
130 LOGIN_REDIRECT_URL = 'blog-home'
131 LOGIN_URL = 'login'
132
133 EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
134 EMAIL_HOST = 'faizabandon@gmail.com'
135 EMAIL_PORT = 587
136 EMAIL_USE_TLS = True
137 EMAIL_HOST_USER = os.environ.get('EMAIL_USER')
138 EMAIL_HOST_PASSWORD = os.environ.get('EMAIL_PASS')
```



```
1 """django_project URL Configuration
2
3 The urlpatterns list routes URLs to views. For more information please see:
4     https://docs.djangoproject.com/en/2.1/topics/http/urls/
5 #examples:
6 function views
7 1. Add an import: from my_app import views
8 2. Add a URL to urlpatterns: path('', views.home, name='home')
9 class-based views
10 1. Add an import: from other_app.views import Home
11 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13 1. Import the include() function: from django.urls import include, path
14 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.contrib.auth import views as auth_views
18 from django.urls import path, include
19 from django.conf import settings
20 from django.conf.urls.static import static
21 urlpatterns = [
22     path('admin/', admin.site.urls),
23     path('login/', auth_views.LoginView.as_view(), name='login'),
24     path('logout/', auth_views.LogoutView.as_view(), name='logout'),
25     path('home/', views.home, name='home'),
26     path('blog/', include('blog.urls')),
27 ]
28 if settings.DEBUG:
29     urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
30 
```

```
from django.conf import settings
from django.conf.urls.static import static
from users import views as user_views

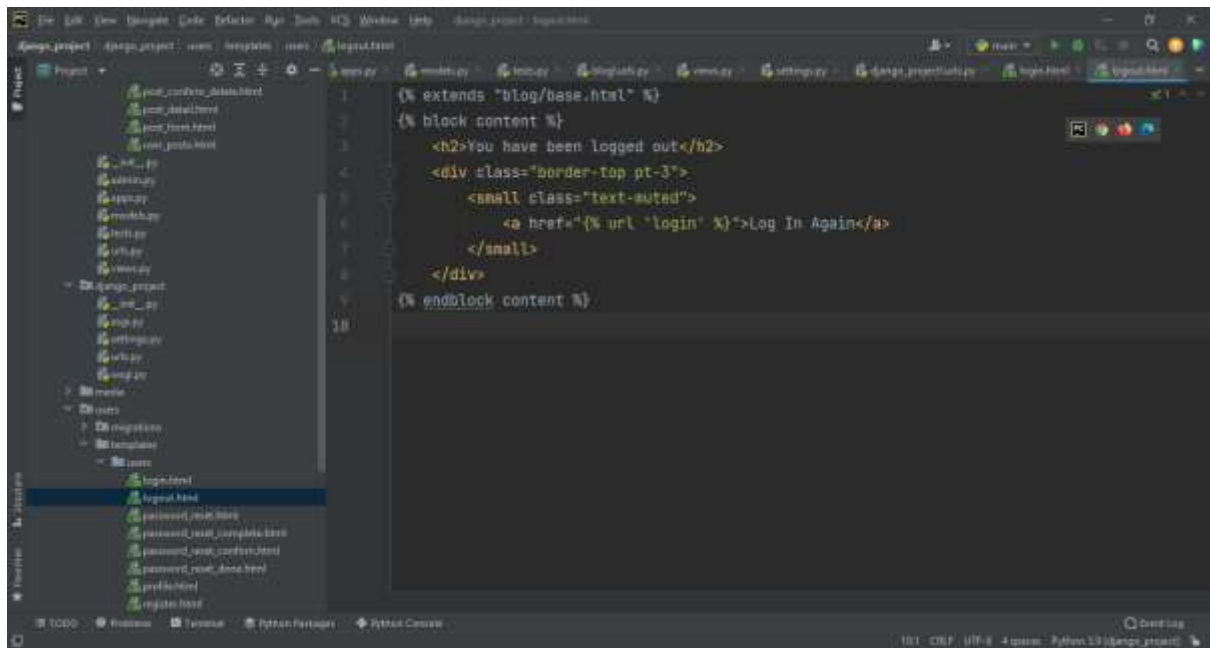
urlpatterns = [
    path('admin/', admin.site.urls),
    path('register/', user_views.register, name='register'),
    path('profile/', user_views.profile, name='profile'),
    path('login/', auth_views.LoginView.as_view(template_name='users/login.html'), name='login'),
    path('logout/', auth_views.LogoutView.as_view(template_name='users/logout.html'), name='logout'),
    path('password-reset/',
        auth_views.PasswordResetView.as_view(
            template_name='users/password_reset.html'
        ),
        name='password_reset'),
    path('password-reset/done/',
        auth_views.PasswordResetDoneView.as_view(
            template_name='users/password_reset_done.html'
        ),
        name='password_reset_done'),
]
```

```
    ),
    name='password_reset_done'),
    path('password-reset-confirm/<uidb64>/<token>/',
        auth_views.PasswordResetConfirmView.as_view(
            template_name='users/password_reset_confirm.html'
        ),
        name='password_reset_confirm'),
    path('password-reset-complete/',
        auth_views.PasswordResetCompleteView.as_view(
            template_name='users/password_reset_complete.html'
        ),
        name='password_reset_complete'),
    path('', include('blog.urls')),
]

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

```
1 {% extends "blog/base.html" %}
2 {% load crispy_forms_tags %}
3 {% block content %}
4     <div class="content-section">
5         <form method="POST">
6             {% csrf_token %}
7             <fieldset class="form-group">
8                 <legend class="border-bottom mb-4">Log In</legend>
9                 {{ form|crispy }}
10            </fieldset>
11            <div class="form-group">
12                <button class="btn btn-outline-info" type="submit">Login</button>
13                <small class="text-muted ml-2">
14                    <a href="{% url 'password_reset' %}">Forgot Password?</a>
15                </small>
16            </div>
17        </form>
18        <div class="border-top pt-3">
19            <small class="text-muted">
20                Need An Account? <a class="ml-2" href="{% url 'register' %}">Sign Up Now</a>
21            </small>
22        </div>
23    </div>
24{% endblock content %}
```

```
8     <legend class="border-bottom mb-4">Log In</legend>
9     {{ form|crispy }}
10 </fieldset>
11 <div class="form-group">
12     <button class="btn btn-outline-info" type="submit">Login</button>
13     <small class="text-muted ml-2">
14         <a href="{% url 'password_reset' %}">Forgot Password?</a>
15     </small>
16 </div>
17 </form>
18 <div class="border-top pt-3">
19     <small class="text-muted">
20         Need An Account? <a class="ml-2" href="{% url 'register' %}">Sign Up Now</a>
21     </small>
22 </div>
23 </div>
24 {% endblock content %}
```

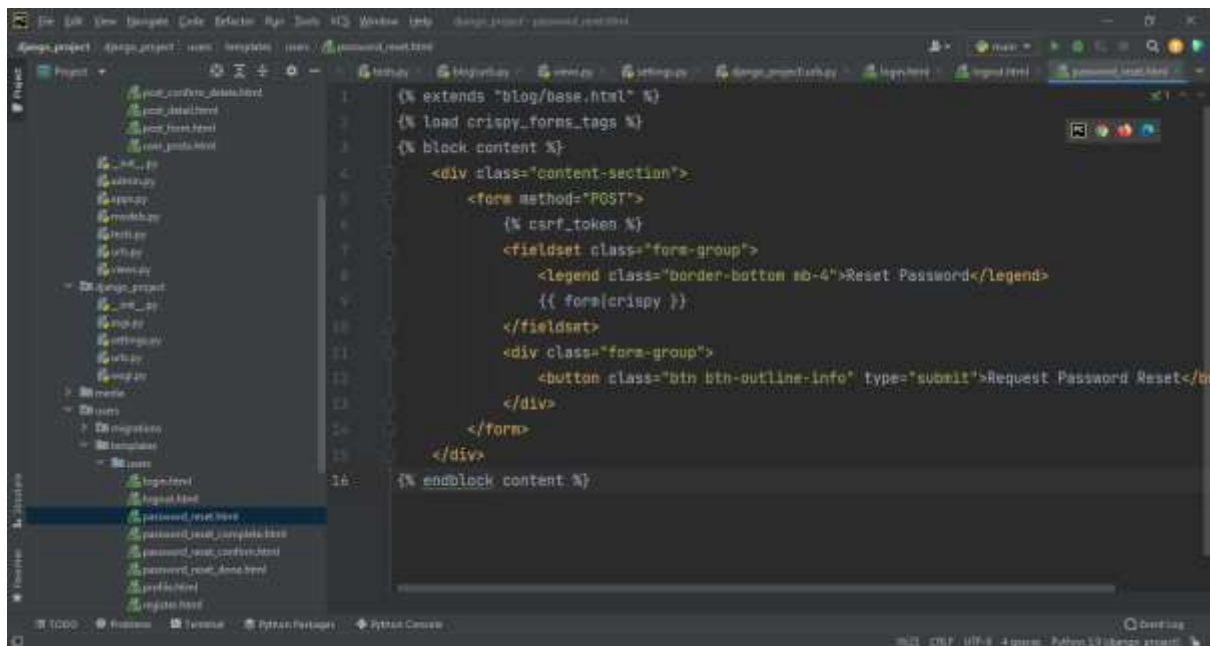


The screenshot shows a web browser window with the address bar displaying 'http://127.0.0.1:8000/login.html'. The page content is as follows:

```
{% extends "blog/base.html" %}
{% block content %}
    <h2>You have been logged out</h2>
    <div class="border-top pt-3">
        <small class="text-muted">
            <a href="{% url 'login' %}">Log In Again</a>
        </small>
    </div>
{% endblock content %}
```

The browser's developer tools are open, showing the 'Elements' panel on the left and the 'Console' panel on the right. The 'Elements' panel shows the document structure, including the 'body' and 'main' elements. The 'Console' panel shows the following log messages:

```
181 CSP: Violation: A non-whitelisted source was loaded.
182 CSP: Violation: A non-whitelisted source was loaded.
```

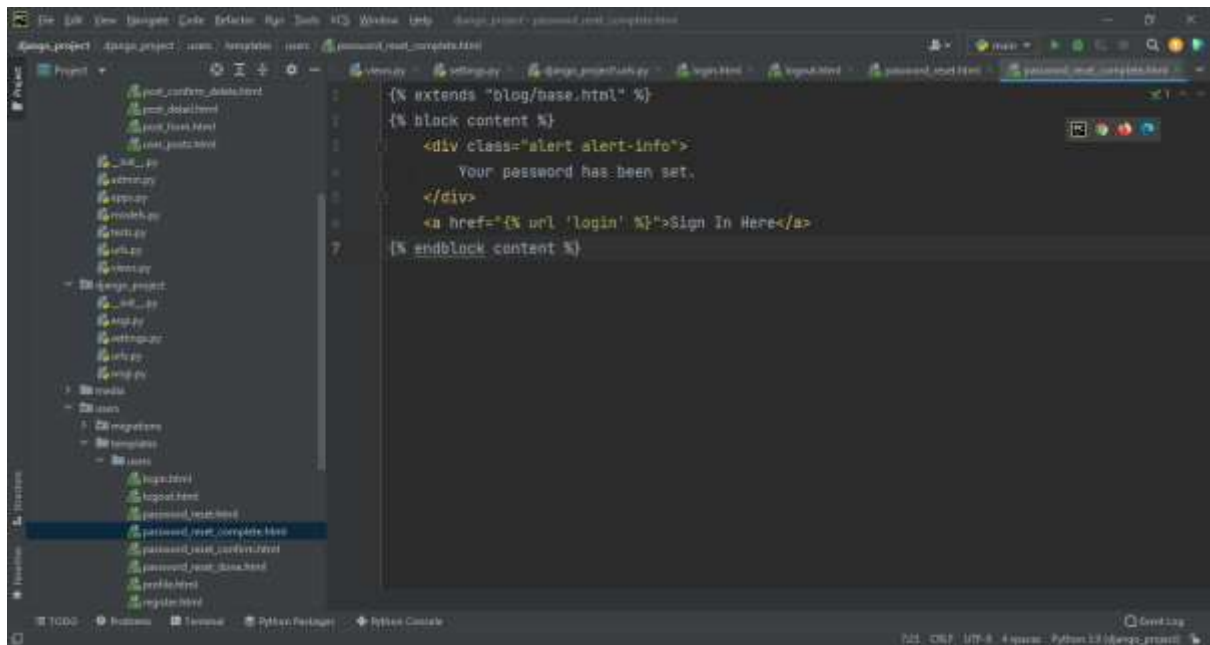


The screenshot shows a web browser window with the address bar displaying 'http://127.0.0.1:8000/password/reset.html'. The page content is as follows:

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
    <div class="content-section">
        <form method="POST">
            <div class="form-group">
                <legend class="border-bottom mb-4">Reset Password</legend>
                {{ form.crispy }}
            </div>
            <div class="form-group">
                <button class="btn btn-outline-info" type="submit">Request Password Reset</button>
            </div>
        </form>
    </div>
{% endblock content %}
```

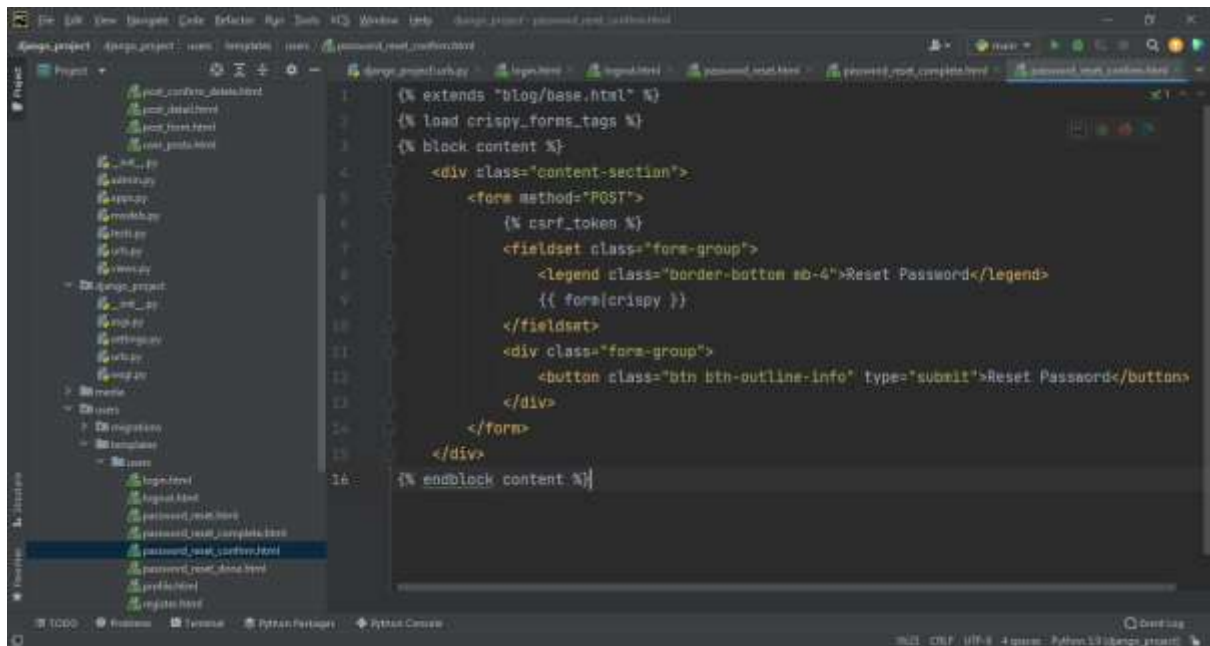
The browser's developer tools are open, showing the 'Elements' panel on the left and the 'Console' panel on the right. The 'Elements' panel shows the document structure, including the 'body' and 'main' elements. The 'Console' panel shows the following log messages:

```
181 CSP: Violation: A non-whitelisted source was loaded.
182 CSP: Violation: A non-whitelisted source was loaded.
```

The screenshot shows a web browser window with the URL `localhost:5000/password/reset/complete.html`. The page displays a confirmation message: "Your password has been set." followed by a link "Sign In Here" with a href attribute pointing to `login.html`. The browser's developer tools are open, showing the HTML structure of the page.

```
{% extends "blog/base.html" %}
{% block content %}
    <div class="alert alert-info">
        Your password has been set.
    </div>
    <a href="{% url 'login' %}">Sign In Here</a>
{% endblock content %}
```



The screenshot shows a web browser window with the URL `localhost:5000/password/reset/confirm.html`. The page displays a password reset form. The form includes a CSRF token, a legend "Reset Password", and a "Reset Password" button. The browser's developer tools are open, showing the HTML structure of the page.

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
    <div class="content-section">
        <form method="POST">
            {% csrf_token %}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Reset Password</legend>
                {{ form|crispy }}
            </fieldset>
            <div class="form-group">
                <button class="btn btn-outline-info" type="submit">Reset Password</button>
            </div>
        </form>
    </div>
{% endblock content %}
```

The screenshot shows the VS Code editor with the file `password_reset_done.html` open. The left sidebar displays a project tree with folders like `blog`, `users`, and `templates`. The main editor area contains the following HTML code:

```
{% extends "blog/base.html" %}

{% block content %}

<div class="alert alert-info">
    An email has been sent with instructions to reset your password
</div>

{% endblock content %}
```

The status bar at the bottom indicates the file is encoded in UTF-8 and uses the Python 3.9 interpreter.

The screenshot shows the VS Code editor with the file `profile.html` open. The left sidebar displays the same project tree. The main editor area contains the following HTML code:

```
{% extends "blog/base.html" %}

{% load crispy_forms_tags %}

{% block content %}

<div class="content-section">
    <div class="media">
        
        <div class="media-body">
            <h2 class="account-heading">{{ user.username }}</h2>
            <p class="text-secondary">{{ user.email }}</p>
        </div>
    </div>
    <form method="POST" enctype="multipart/form-data">
        {% csrf_token %}
        <fieldset class="form-group">
            <legend class="border-bottom mb-4">Profile Info</legend>
            {{ u_form|crispy }}
            {{ p_form|crispy }}
        </fieldset>
        <div class="form-group">
            <button class="btn btn-outline-info" type="submit">Update</button>
        </div>
    </form>
</div>

{% endblock content %}
```

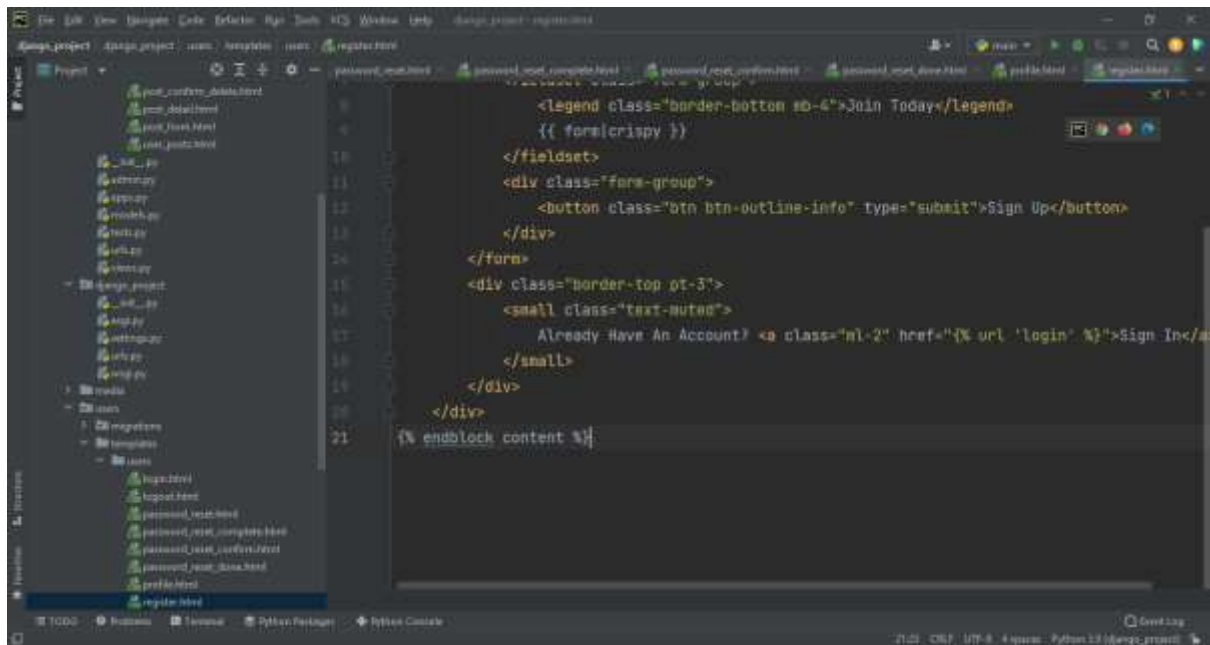
The status bar at the bottom indicates the file is encoded in UTF-8 and uses the Python 3.9 interpreter.

```
django_project > django_project > users > templates > users > profile.html

10 </div>
11 </div>
12 <form method="POST" enctype="multipart/form-data">
13     {% csrf_token %}
14     <fieldset class="form-group">
15         <legend class="border-bottom mb-4">Profile Info</legend>
16         {{ u_form|crispy }}
17         {{ p_form|crispy }}
18     </fieldset>
19     <div class="form-group">
20         <button class="btn btn-outline-info" type="submit">Update</button>
21     </div>
22 </form>
23 </div>
24 {% endblock content %}
```

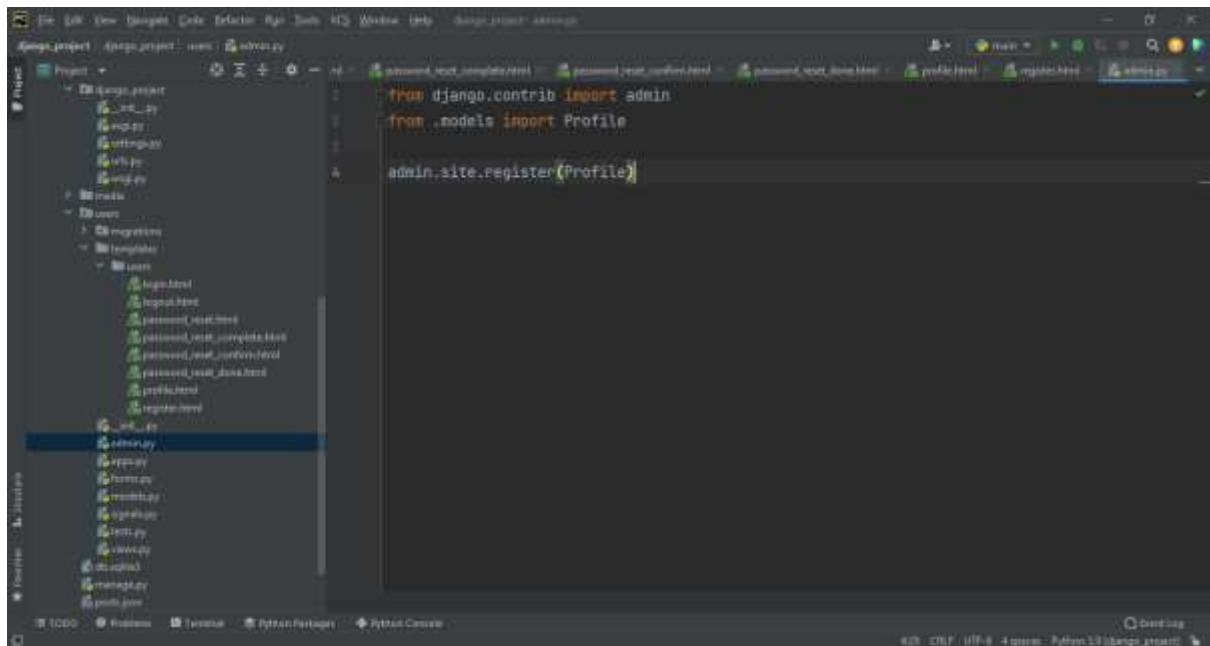
```
django_project > django_project > users > templates > users > register.html

1 {% extends "blog/base.html" %}
2 {% load crispy_forms_tags %}
3 {% block content %}
4     <div class="content-section">
5         <form method="POST">
6             {% csrf_token %}
7             <fieldset class="form-group">
8                 <legend class="border-bottom mb-4">Join Today</legend>
9                 {{ form|crispy }}
10            </fieldset>
11            <div class="form-group">
12                <button class="btn btn-outline-info" type="submit">Sign Up</button>
13            </div>
14        </form>
15        <div class="border-top pt-3">
16            <small class="text-muted">
17                Already Have An Account? <a class="ml-2" href="{% url 'login' %}">Sign In</a>
18            </small>
19        </div>
20    </div>
```



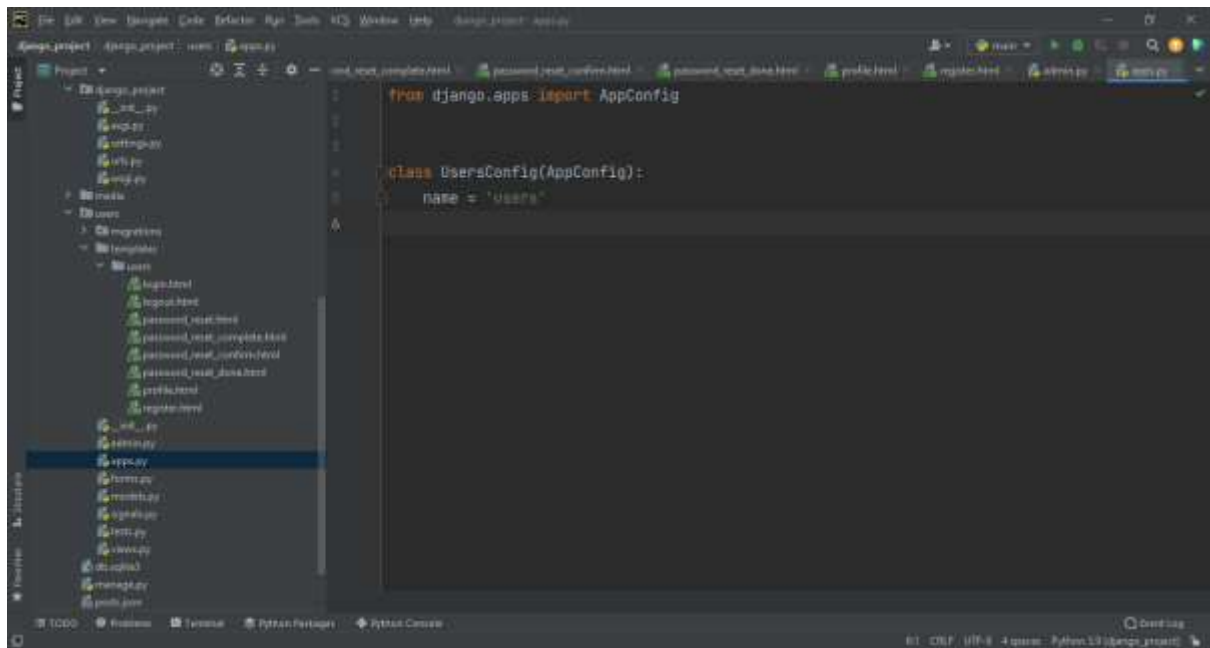
The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'django_project', 'users', and 'templates'. The code editor displays HTML template code for a registration form. The code includes a legend for 'Join Today', a form group with a 'Sign Up' button, and a link for 'Already Have An Account?'.

```
1 <legend class="border-bottom mb-4">Join Today</legend>
2 {{ form|crispy }}
3
4 </fieldset>
5 <div class="form-group">
6   <button class="btn btn-outline-info" type="submit">Sign Up</button>
7 </div>
8 </form>
9
10 <div class="border-top pt-3">
11   <small class="text-muted">
12     Already Have An Account? <a class="ml-2" href="{% url 'login' %}">Sign In</a>
13   </small>
14 </div>
15 </div>
16 {% endblock content %}
```



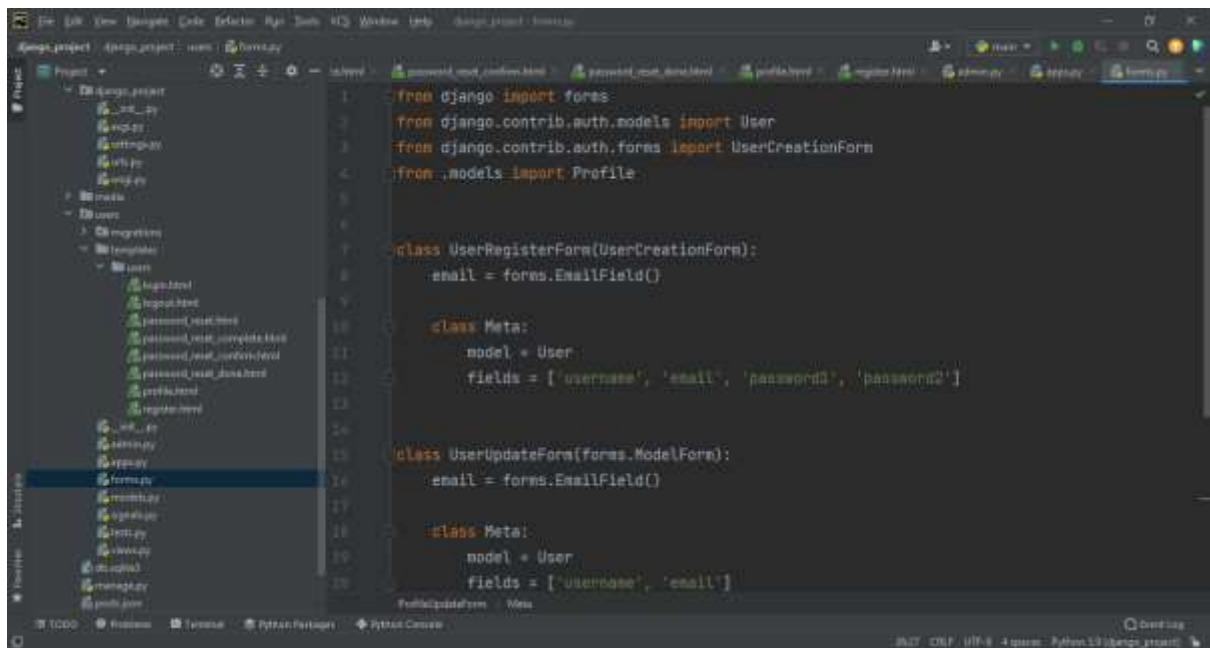
The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'django_project', 'users', and 'templates'. The code editor displays Python code for registering a 'Profile' model in the Django admin.

```
1 from django.contrib import admin
2 from .models import Profile
3
4 admin.site.register(Profile)
```



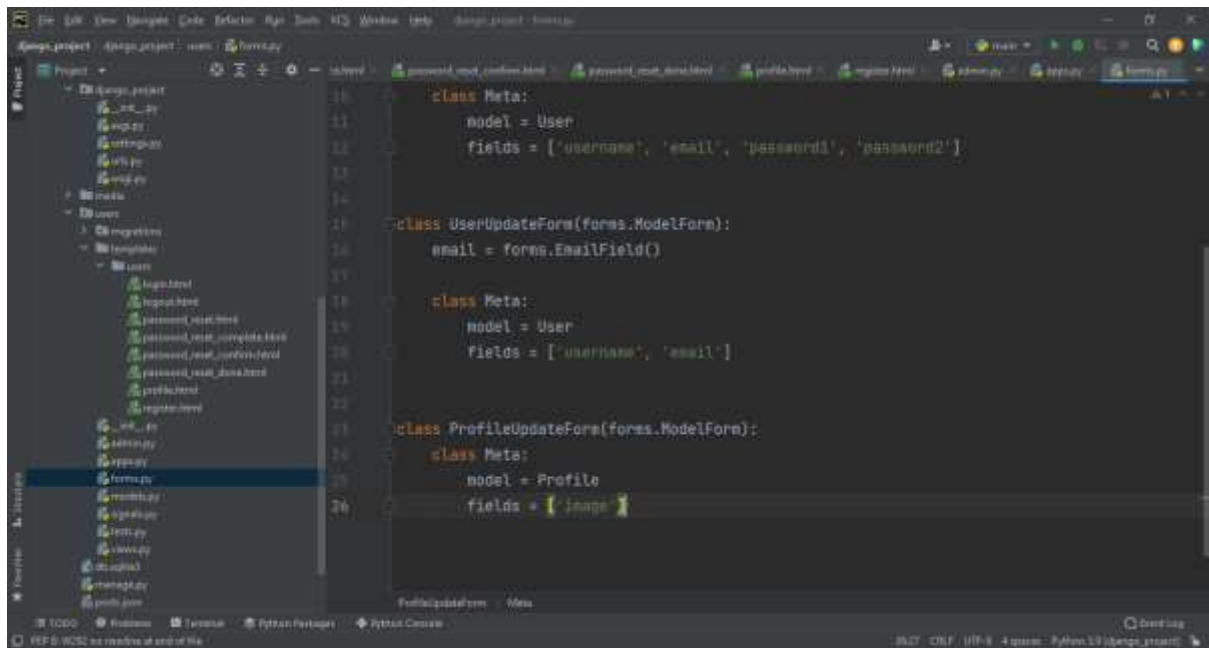
The screenshot shows the PyCharm IDE with the 'UsersConfig' class defined in 'app.py'. The file explorer on the left shows the project structure, including 'django_project', 'users', and 'templates'. The main editor displays the following code:

```
1 from django.apps import AppConfig
2
3
4 class UsersConfig(AppConfig):
5     name = 'users'
```

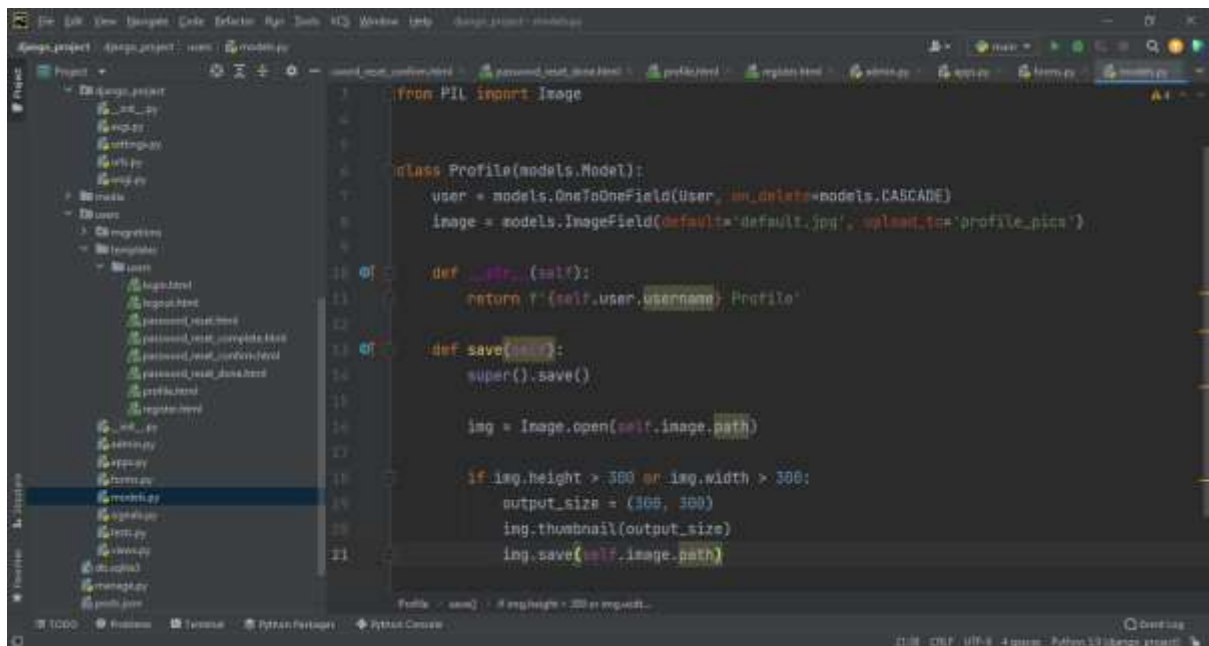


The screenshot shows the PyCharm IDE with the 'UserRegisterForm' and 'UserUpdateForm' classes defined in 'forms.py'. The file explorer on the left shows the project structure, including 'django_project', 'users', and 'templates'. The main editor displays the following code:

```
1 from django import forms
2 from django.contrib.auth.models import User
3 from django.contrib.auth.forms import UserCreationForm
4 from .models import Profile
5
6
7 class UserRegisterForm(UserCreationForm):
8     email = forms.EmailField()
9
10     class Meta:
11         model = User
12         fields = ['username', 'email', 'password1', 'password2']
13
14 class UserUpdateForm(forms.ModelForm):
15     email = forms.EmailField()
16
17     class Meta:
18         model = User
19         fields = ['username', 'email']
```

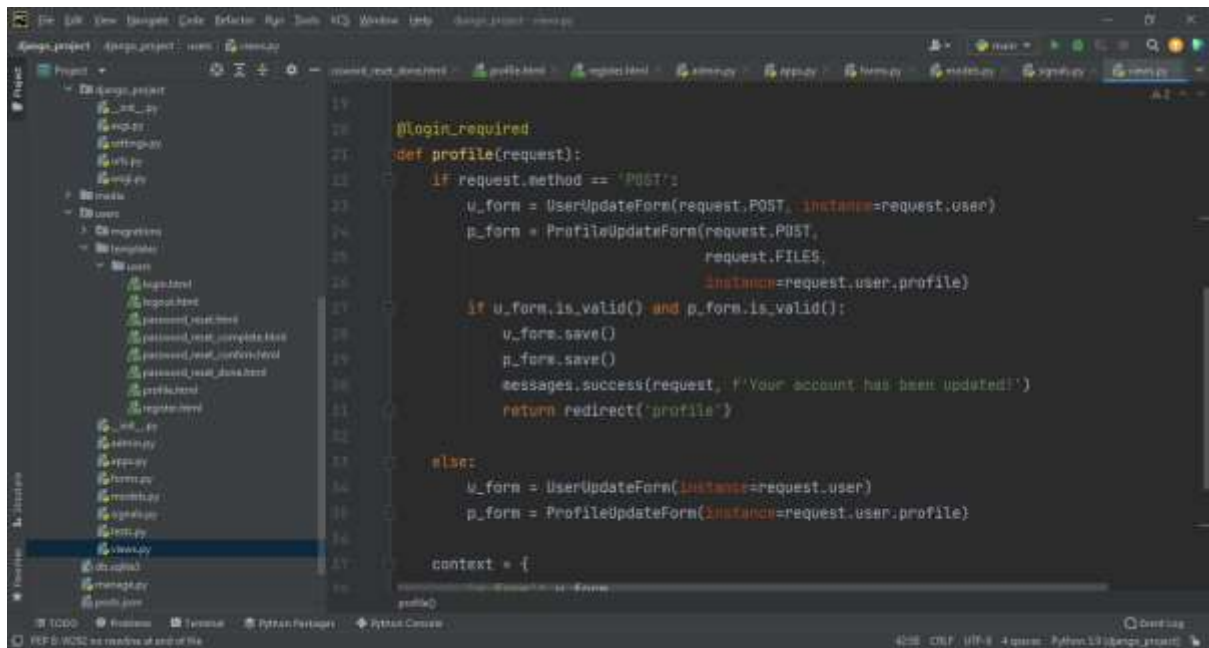
```
10 class Meta:
11     model = User
12     fields = ['username', 'email', 'password1', 'password2']
13
14
15 class UserUpdateForm(forms.ModelForm):
16     email = forms.EmailField()
17
18     class Meta:
19         model = User
20         fields = ['username', 'email']
21
22
23 class ProfileUpdateForm(forms.ModelForm):
24     class Meta:
25         model = Profile
26         fields = ['image']
```



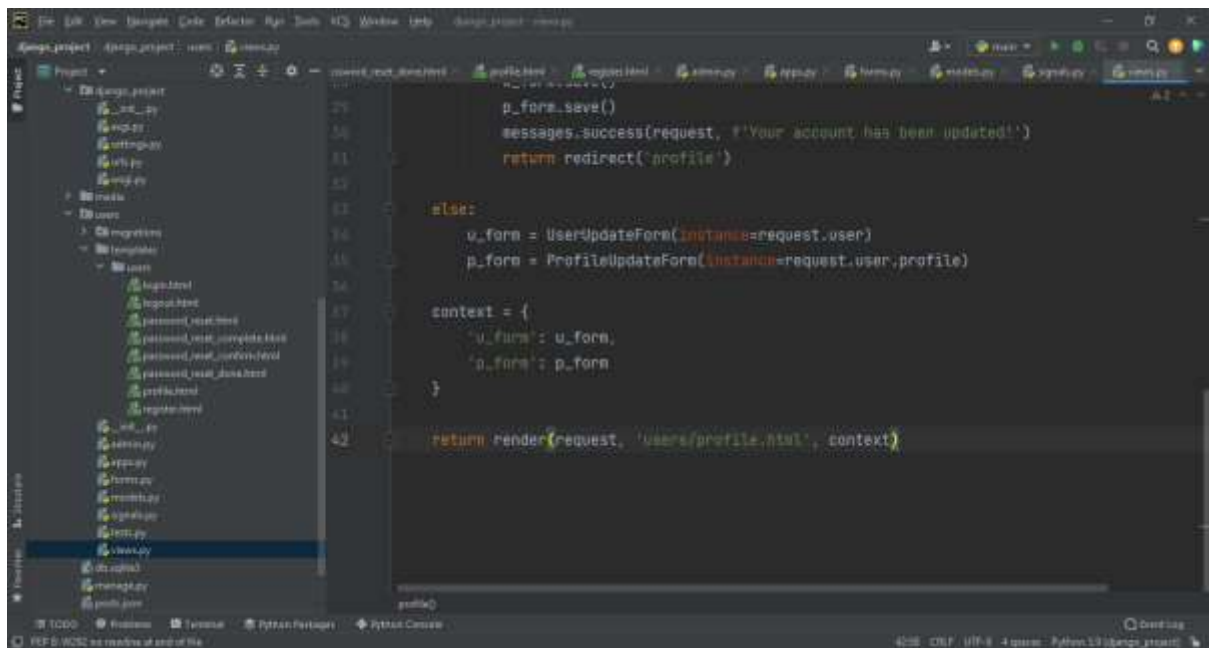
```
1 from PIL import Image
2
3
4 class Profile(models.Model):
5     user = models.OneToOneField(User, on_delete=models.CASCADE)
6     image = models.ImageField(default='default.jpg', upload_to='profile_pics')
7
8     def __str__(self):
9         return f'{self.user.username} Profile'
10
11     def save(self):
12         super().save()
13
14         img = Image.open(self.image.path)
15
16         if img.height > 300 or img.width > 300:
17             output_size = (300, 300)
18             img.thumbnail(output_size)
19             img.save(self.image.path)
```

```
1 from django.db.models.signals import post_save
2 from django.contrib.auth.models import User
3 from django.dispatch import receiver
4 from .models import Profile
5
6
7 @receiver(post_save, sender=User)
8 def create_profile(sender, instance, created, **kwargs):
9     if created:
10         Profile.objects.create(user=instance)
11
12
13 @receiver(post_save, sender=User)
14 def save_profile(sender, instance, **kwargs):
15     instance.profile.save()
```

```
1 from django.shortcuts import render, redirect
2 from django.contrib import messages
3 from django.contrib.auth.decorators import login_required
4 from .forms import UserRegisterForm, UserUpdateForm, ProfileUpdateForm
5
6
7 def register(request):
8     if request.method == 'POST':
9         form = UserRegisterForm(request.POST)
10         if form.is_valid():
11             form.save()
12             username = form.cleaned_data.get('username')
13             messages.success(request, f'Your account has been created! You are now able to login')
14             return redirect('login')
15         else:
16             form = UserRegisterForm()
17     return render(request, 'users/register.html', {'form': form})
18
19
20 @login_required
```



```
19 @login_required
20 def profile(request):
21     if request.method == 'POST':
22         u_form = UserUpdateForm(request.POST, instance=request.user)
23         p_form = ProfileUpdateForm(request.POST,
24                                   request.FILES,
25                                   instance=request.user.profile)
26         if u_form.is_valid() and p_form.is_valid():
27             u_form.save()
28             p_form.save()
29             messages.success(request, f'Your account has been updated!')
30             return redirect('profile')
31
32     else:
33         u_form = UserUpdateForm(instance=request.user)
34         p_form = ProfileUpdateForm(instance=request.user.profile)
35
36     context = {
37         'u_form': u_form,
38         'p_form': p_form
39     }
40
41     return render(request, 'users/profile.html', context)
```



```
19 @login_required
20 def profile(request):
21     if request.method == 'POST':
22         u_form = UserUpdateForm(request.POST, instance=request.user)
23         p_form = ProfileUpdateForm(request.POST,
24                                   request.FILES,
25                                   instance=request.user.profile)
26         if u_form.is_valid() and p_form.is_valid():
27             u_form.save()
28             p_form.save()
29             messages.success(request, f'Your account has been updated!')
30             return redirect('profile')
31
32     else:
33         u_form = UserUpdateForm(instance=request.user)
34         p_form = ProfileUpdateForm(instance=request.user.profile)
35
36     context = {
37         'u_form': u_form,
38         'p_form': p_form
39     }
40
41     return render(request, 'users/profile.html', context)
```

