TRAINING DAY12 REPORT

07 JULY 2025

DJANGO FORMS

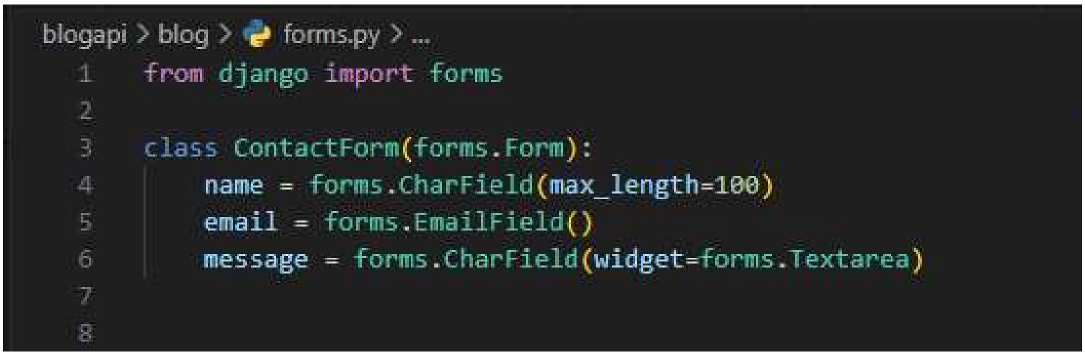
Django Forms provide a powerful way to **handle user input**, **render HTML forms**, **validate data**, and **save to the database** — all in a clean and secure way.

Types of Forms in Django

Type	Description
forms.Form	Used for creating forms **not directly tied** to models (manual field definition).
forms.ModelForm	Tied to a Django model — automatically creates form fields based on the model.

Example 1: Using forms.Form

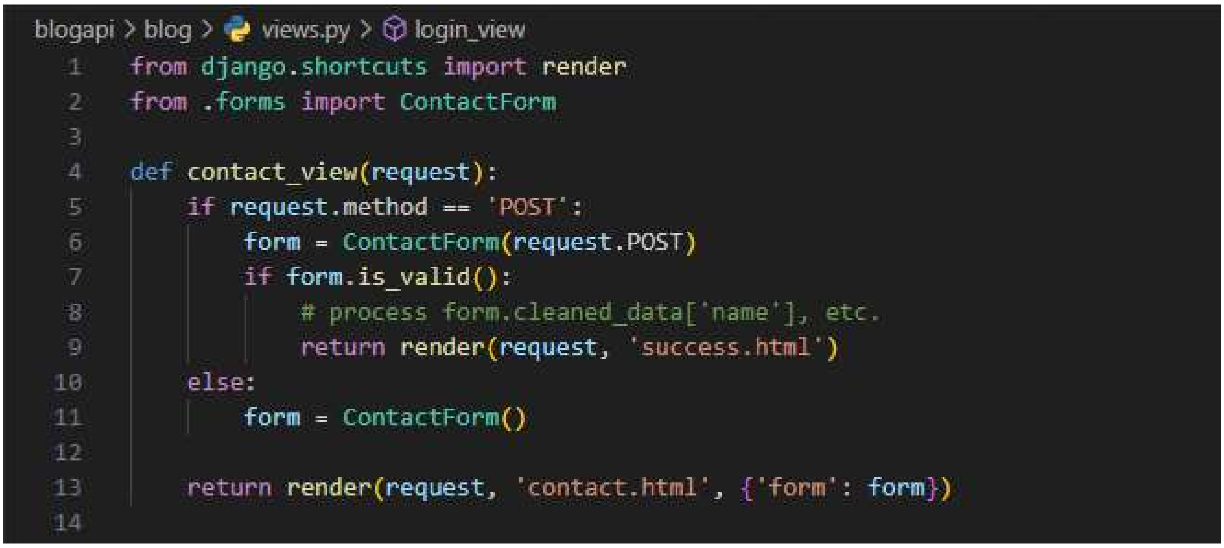
1. forms.py



blogapi > blog > forms.py > ...

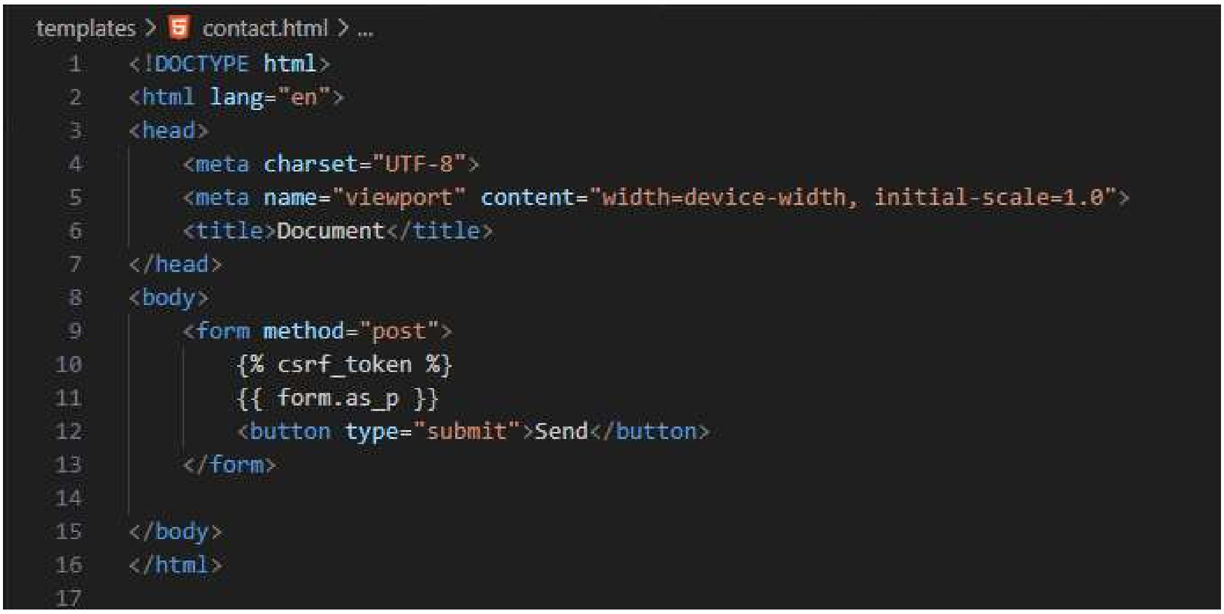
```python
from django import forms

class ContactForm(forms.Form):
    name = forms.CharField(max_length=100)
    email = forms.EmailField()
    message = forms.CharField(widget=forms.Textarea)
```

2. views.py



```python
from django.shortcuts import render
from .forms import ContactForm

def contact_view(request):
    if request.method == 'POST':
        form = ContactForm(request.POST)
        if form.is_valid():
            # process form.cleaned_data['name'], etc.
            return render(request, 'success.html')
    else:
        form = ContactForm()

    return render(request, 'contact.html', {'form': form})

```

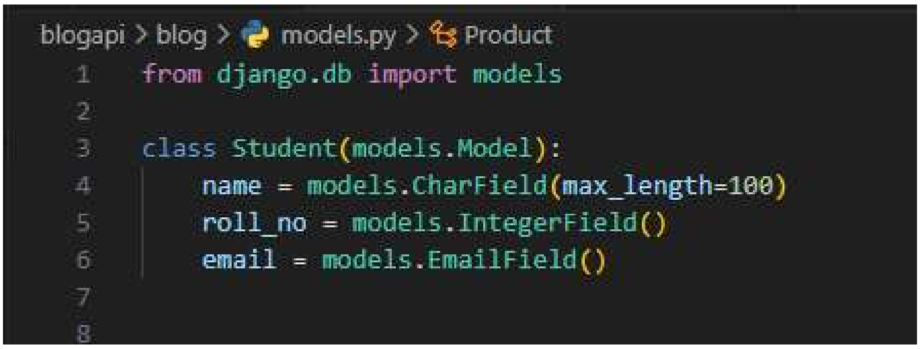
3. contact.html



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form method="post">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Send</button>
    </form>

</body>
</html>

```

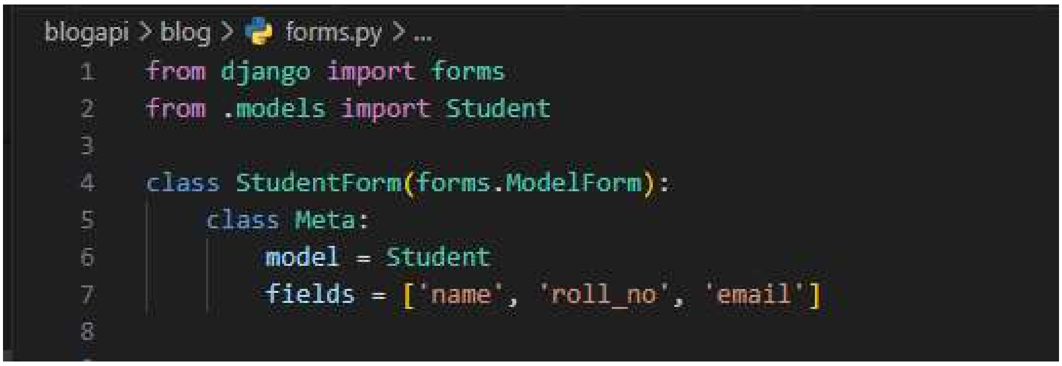
Example 2: Using forms.ModelForm

1. models.py



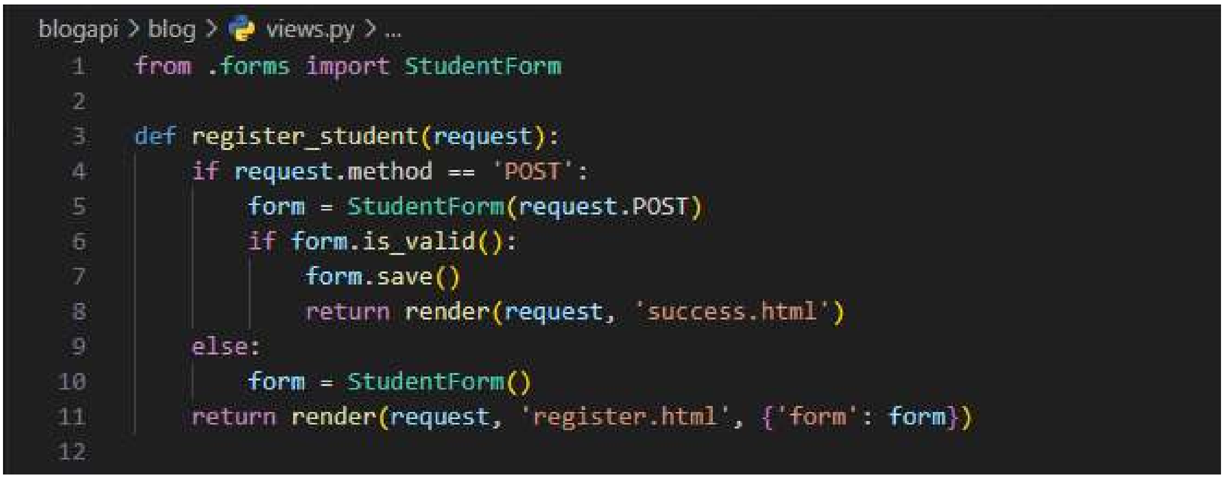
```python
from django.db import models

class Student(models.Model):
    name = models.CharField(max_length=100)
    roll_no = models.IntegerField()
    email = models.EmailField()

```

2. forms.py



```python
from django import forms
from .models import Student

class StudentForm(forms.ModelForm):
    class Meta:
        model = Student
        fields = ['name', 'roll_no', 'email']

```

3. views.py



```python
from .forms import StudentForm

def register_student(request):
    if request.method == 'POST':
        form = StudentForm(request.POST)
        if form.is_valid():
            form.save()
            return render(request, 'success.html')
    else:
        form = StudentForm()
    return render(request, 'register.html', {'form': form})

```

Useful Form Methods & Attributes

Feature	Description
form.is_valid()	Checks if submitted data meets field requirements
form.cleaned_data	Returns a dictionary of validated data
form.errors	Displays validation errors
form.save()	Saves the form (ModelForm only)
form.as_p, as_table, as_ul	Renders form fields in different HTML formats

CSRF Protection

Always include:

{% csrf_token %}

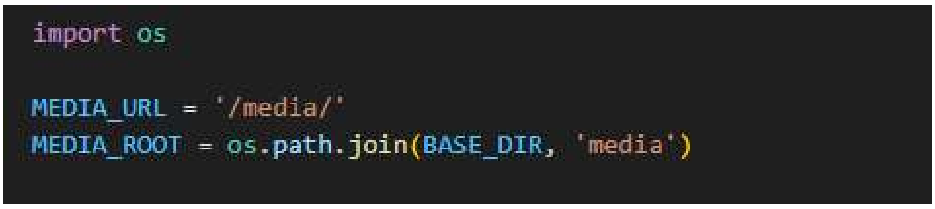
inside your <form> tag to protect against CSRF attacks.

FILE AND IMAGE UPLOADS

How Django Handles Uploads

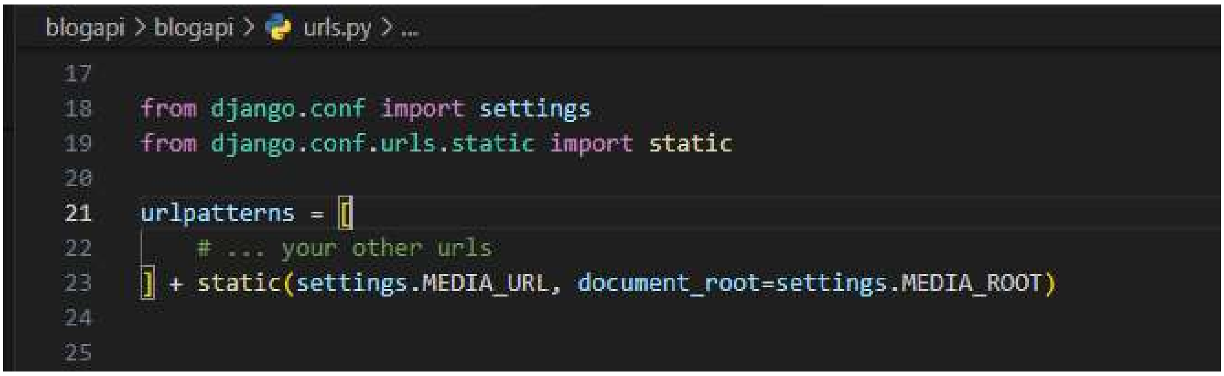
- Django stores uploaded files in a location defined in settings.py (MEDIA_ROOT).
- The **file's path** is stored in the database, **not the file's content**.
- Requires enctype="multipart/form-data" in form tag.

1. **In settings.py:**



```python
import os

MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

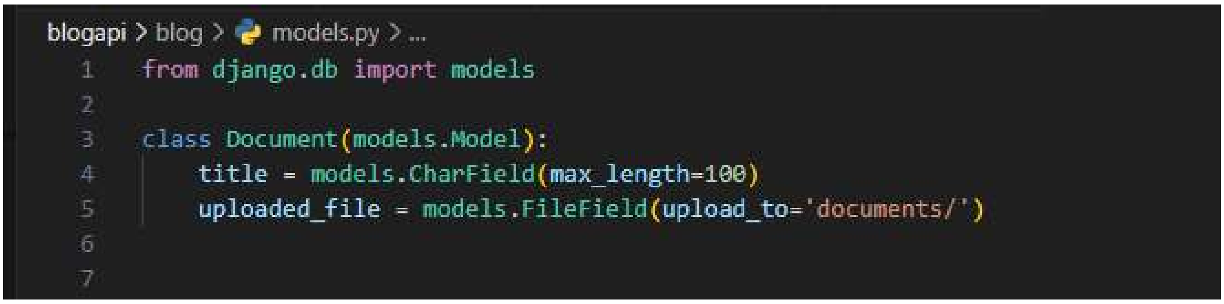
2. **In urls.py:**



```python

from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    # ... your other urls
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)


```

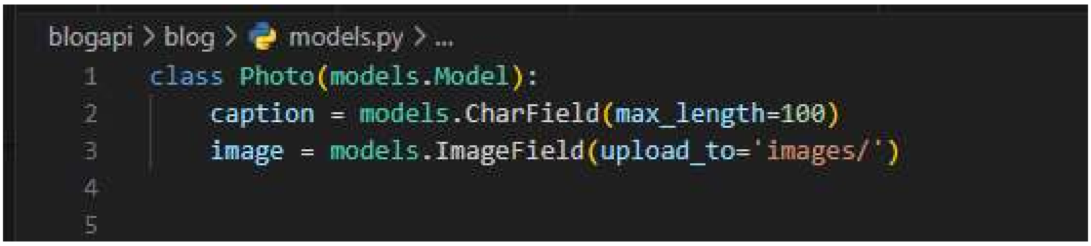
3. **Models for File & Image Upload**

File Upload Model



```python
from django.db import models

class Document(models.Model):
    title = models.CharField(max_length=100)
    uploaded_file = models.FileField(upload_to='documents/')


```

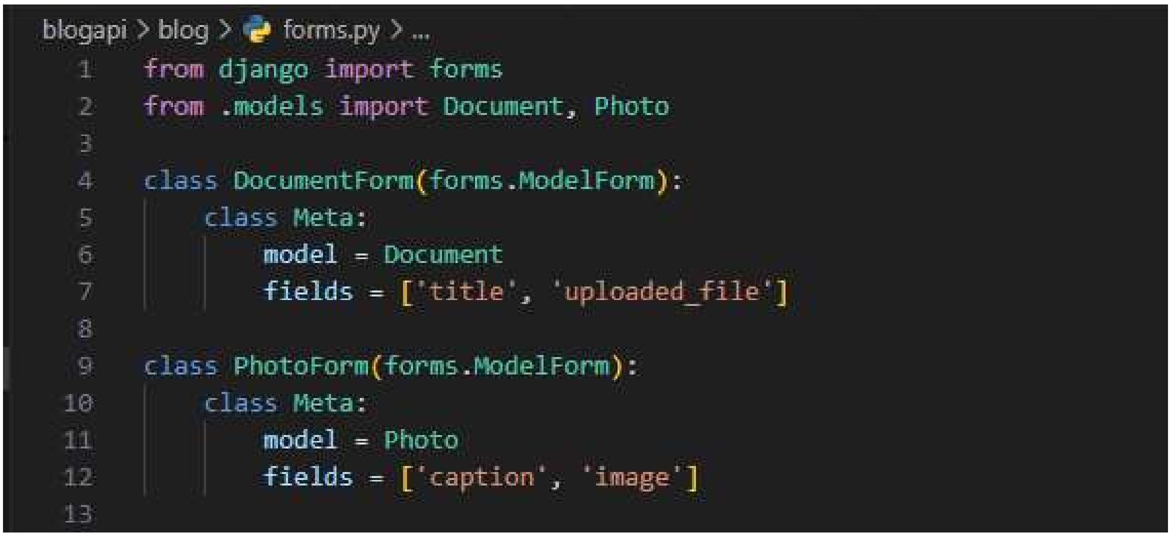
Image Upload Model



```
class Photo(models.Model):
    caption = models.CharField(max_length=100)
    image = models.ImageField(upload_to='images/')
```

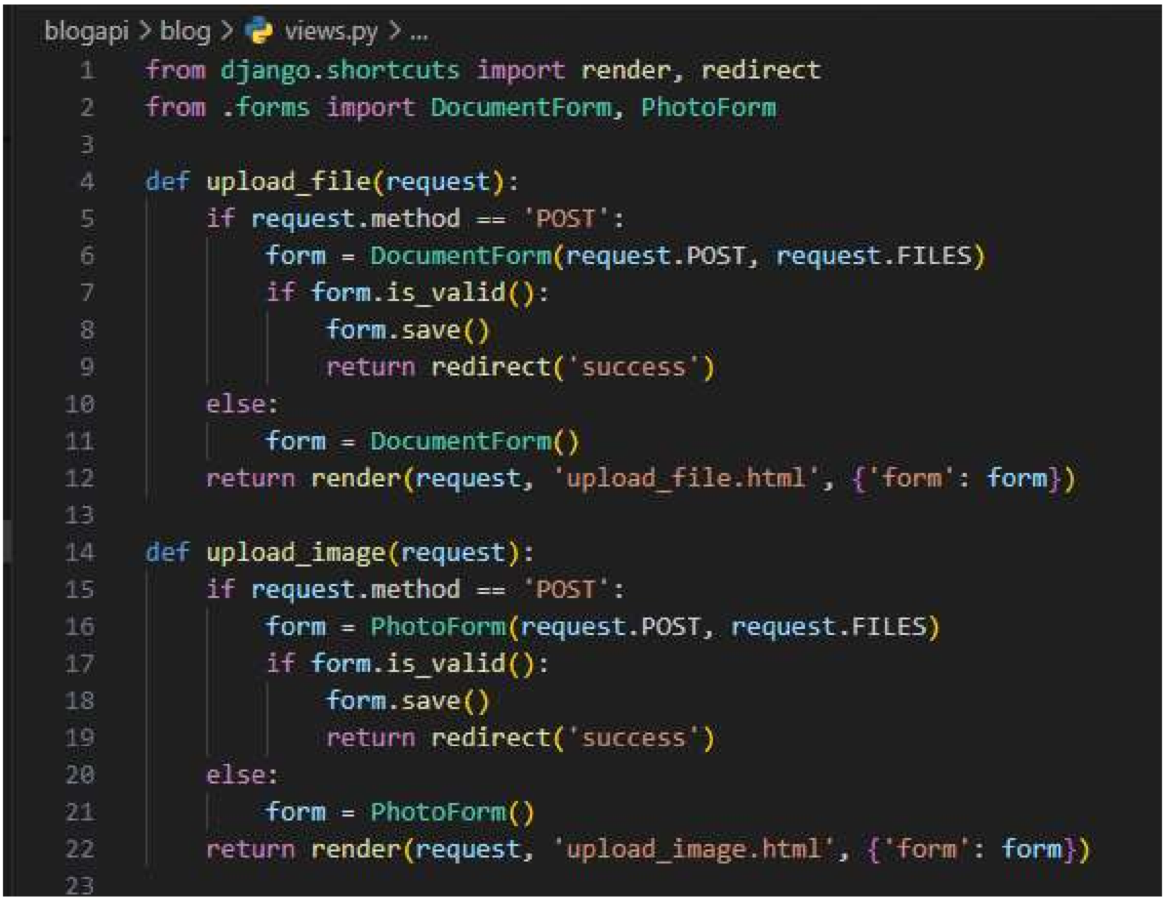
NOTE: For imageField , install pillow.

4. Forms



```
from django import forms
from .models import Document, Photo

class DocumentForm(forms.ModelForm):
    class Meta:
        model = Document
        fields = ['title', 'uploaded_file']

class PhotoForm(forms.ModelForm):
    class Meta:
        model = Photo
        fields = ['caption', 'image']
```

5. Views

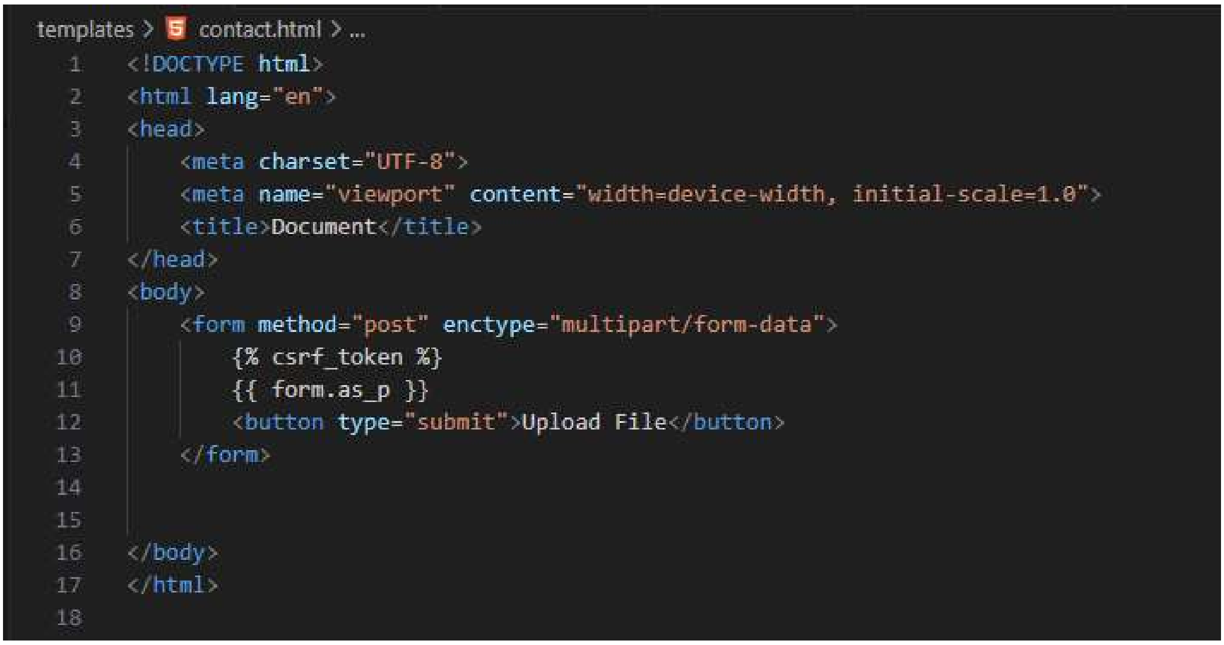


blogapi > blog > views.py > ...

```python
from django.shortcuts import render, redirect
from .forms import DocumentForm, PhotoForm

def upload_file(request):
    if request.method == 'POST':
        form = DocumentForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('success')
    else:
        form = DocumentForm()
    return render(request, 'upload_file.html', {'form': form})

def upload_image(request):
    if request.method == 'POST':
        form = PhotoForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('success')
    else:
        form = PhotoForm()
    return render(request, 'upload_image.html', {'form': form})

```

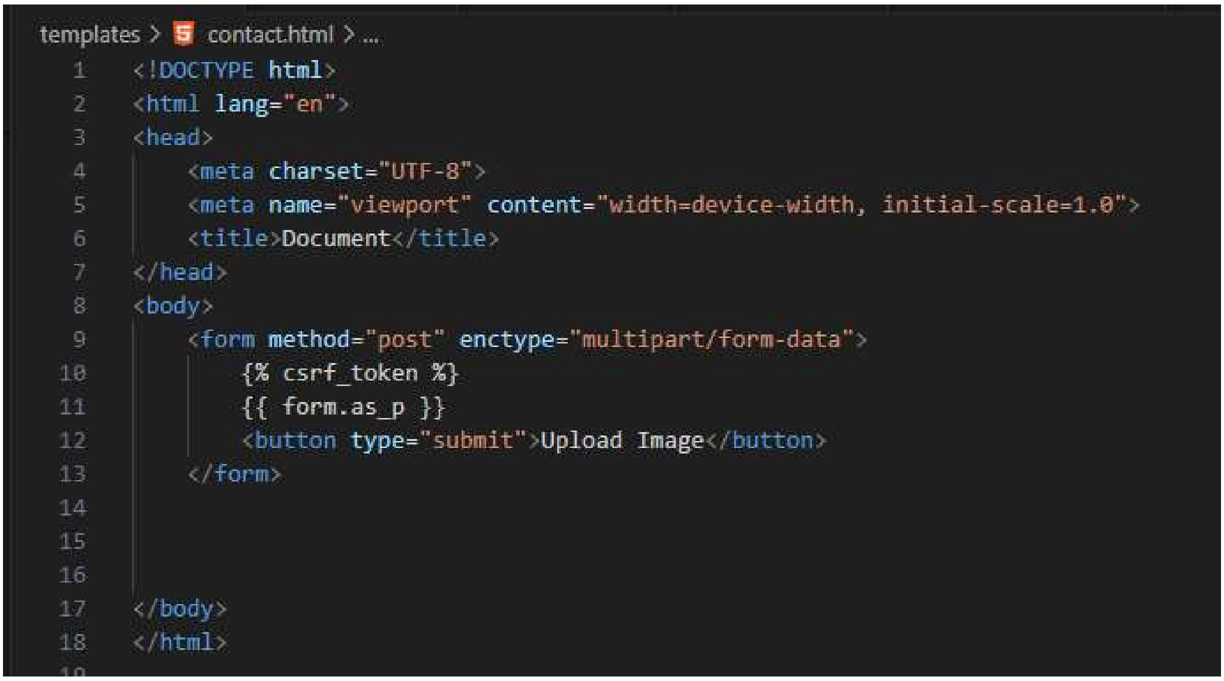
6. HTML Templates

File Upload (upload_file.html)



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form method="post" enctype="multipart/form-data">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Upload File</button>
    </form>


</body>
</html>

```

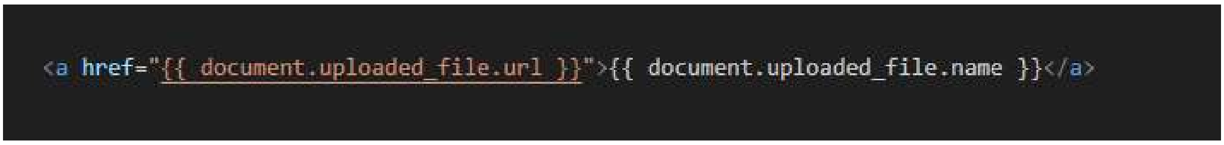
Image Upload (upload_image.html)



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form method="post" enctype="multipart/form-data">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Upload Image</button>
    </form>



</body>
</html>
```

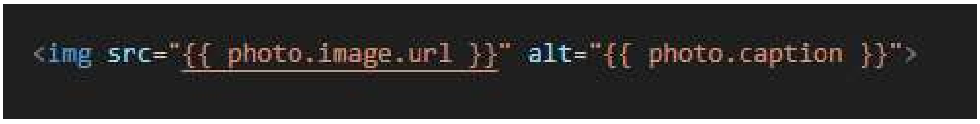
Displaying Uploaded Files & Images

Display files:



```
<a href="{{ document.uploaded_file.url }}">{{ document.uploaded_file.name }}</a>
```

Display images:



```
<img src="{{ photo.image.url }}" alt="{{ photo.caption }}">
```