Django File upload:

1. Create a Django Project and App
First, create a Django project and an app within the project:

```
django-admin startproject file_upload_project
cd file_upload_project
python manage.py startapp file_upload_app
```

2. Define Model for File Upload

In `file_upload_app/models.py`, define a model to store uploaded files:

```
from django.db import models

class UploadedFile(models.Model):
    file = models.FileField(upload_to='uploads/')
    uploaded_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.file.name
```

3. Create Form for File Upload
In `file_upload_app/forms.py`, create a form for uploading files:

```
from django import forms
from .models import UploadedFile

class UploadFileForm(forms.ModelForm):
    class Meta:
        model = UploadedFile
        fields = ['file']
```

4. Create Views for File Upload
In `file_upload_app/views.py`, create views for handling file upload:

```
from django.shortcuts import render, redirect
from .forms import UploadFileForm

def upload_file(request):
    if request.method == 'POST':
        form = UploadFileForm(request.POST, request.FILES)
        if form.is_valid():
```

```
            form.save()
            return redirect('upload_success')
    else:
        form = UploadFileForm()
    return render(request, 'file_upload_app/upload.html', {'form': form})

def upload_success(request):
    return render(request, 'file_upload_app/upload_success.html')
```

5. Create HTML Templates

Create HTML templates for the upload form and success page:

`file_upload_app/templates/file_upload_app/upload.html`:

```
<!DOCTYPE html>
<html>
<head>
    <title>File Upload</title>
</head>
<body>
    <h2>Upload File</h2>
    <form method="post" enctype="multipart/form-data">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Upload</button>
    </form>
</body>
</html>
```

`file_upload_app/templates/file_upload_app/upload_success.html`:

```
<!DOCTYPE html>
<html>
<head>
    <title>Upload Success</title>
</head>
<body>
    <h2>File Uploaded Successfully!</h2>
    <a href="{% url 'upload_file' %}">Upload Another File</a>
</body>
</html>
```

6. Configure URLs

Configure URLs in `file_upload_app/urls.py`:

```
from django.urls import path
from . import views

urlpatterns = [
   path('upload/', views.upload_file, name='upload_file'),
   path('success/', views.upload_success, name='upload_success'),
]
```

7. Include App URLs in Project URLs

Include app URLs in the project's `urls.py`:

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
   path('admin/', admin.site.urls),
   path('', include('file_upload_app.urls')),
]
```

Explanation:

- **Model**:

  `UploadedFile` model stores uploaded files with a `FileField` for the file itself and a `DateTimeField` for the upload time.

- **Form**:

  `UploadFileForm` is a model form for the `UploadedFile` model.

- **Views**:

  `upload_file` handles file upload, rendering the upload form and processing POST requests. `upload_success` displays a success message after a successful upload.

- **Templates**:

  `upload.html` contains the form for file upload, while `upload_success.html` displays a success message.

- **URLs**:

    URLs are configured to map to the `upload_file` and `upload_success` views.

- **Settings**:

    Make sure to configure `MEDIA_ROOT` and `MEDIA_URL` settings in `settings.py` to specify where uploaded files are stored and served from.

That's it! With this setup, users can upload files using the provided form, and the uploaded files will be saved in the specified directory.