Managing Static and Media Files

Integrate static files for styling and media files for user avators into the TODO App.

Static Files

Implementing static files for css/js/images

Static Files Integration

- Create a `static` directory within the TODO App directory.
- Place CSS and JavaScript files (e.g., `styles.css`, `scripts.js`) inside the `static` directory for styling and interactivity.
- Link the static files in the HTML templates of the TODO App to apply the styles and functionality.

```
# Settings.py
STATIC_URL = '/static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```

```
<!-- Html file -->
<link rel="stylesheet" type="text/css" href="{% static 'css/styles.css' %}" />
```

Media Files in Django

When working with files like images uploaded by users, Django offers convenient APIs for file management. By default, Django stores files locally using the `MEDIA_ROOT` and `MEDIA_URL` settings. However, you can customize storage behavior by implementing custom file storage systems.

Using Files in Models

Consider an example model that uses an `FileField` to store a user's avatar:

```python
from django.db import models

class UserProfile(models.Model):
    # Other fields
    avator = models.FileField(upload_to="avators")
```

- Each `UserProfile` instance will have an `avator` attribute representing the uploaded photo.
- You can access details of the attached photo, such as its name, path, and URL.
- Note that the actual file name on disk may change after saving the model.

Media Files Configuration

Update the `settings.py` module

```python
DEBUG = True

# Optional: Add context processor for media
TEMPLATES[0]['OPTIONS']['context_processors'].append(
    'django.template.context_processors.media'
)

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

By adding `'django.template.context_processors.media'` to the context processors, you ensure that the `MEDIA_URL` variable is automatically available in all your templates. This allows you to easily reference media files in your HTML templates without hardcoding the URL.

Extend the `urlpatterns`

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

# Existing urlpatterns

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

Note: Create a `media` folder in your project root.

Example Usage in an HTML Template

Suppose you want to display a user's avator in an HTML template (`list.html`). You can do so as follows:

```
<h1>Avator</h1>
<!-- Construct the avator image URL using MEDIA_URL -->
<img src="{{ MEDIA_URL }}{{ task.avator }}" alt="Avator" />
```

In your Django view, pass the `avator` object (containing the avator image) to the template context. Adjust the template and view code according to your specific project needs.

Summary

By completing this exercise, you will successfully integrate static files for styling and media files for user avators into the TODO App. You will understand how to configure Django settings for static and media files and implement file upload functionality in Django forms.