

# ⑥

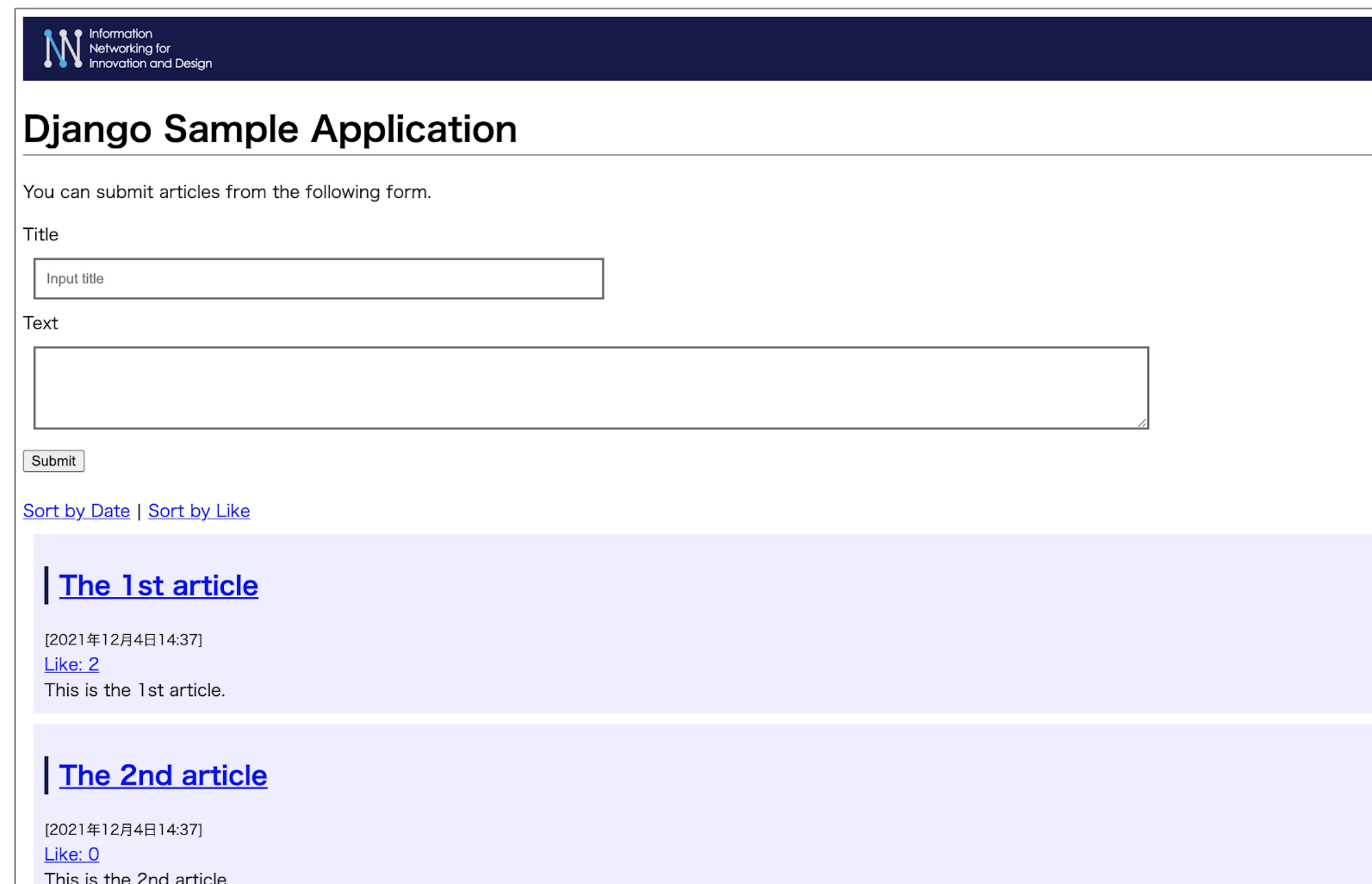
## ソート機能

---

最後に、記事を新着順あるいは「いいね」順にソートする機能も追加しましょう。

# ソート機能を実装する

- 投稿された記事を、新着順あるいは「いいね」順に並べ替える機能を実装します
- そのためには、クエリーパラメータに応じて、モデルの検索結果を並べ替える必要があります



The screenshot displays the Django Sample Application interface. At the top, there is a dark blue header with the INIAD logo and the text "Information Networking for Innovation and Design". Below the header, the title "Django Sample Application" is centered. A message states, "You can submit articles from the following form." The form consists of a "Title" field with a placeholder "Input title" and a "Text" field. A "Submit" button is located below the form. Below the form, there are two links: "Sort by Date" and "Sort by Like". The list of articles is displayed below the links. The first article is titled "The 1st article" and was posted on "2021年12月4日14:37" with "Like: 2". The second article is titled "The 2nd article" and was also posted on "2021年12月4日14:37" with "Like: 0".

# (再掲)リクエストからのデータ取り出し: `HttpRequest.GET`

- `HttpRequest.GET`
  - クエリーパラメータを辞書型で取得できる
- クエリーパラメータとは、サーバーに情報を送信するためにURLの末尾に付け加える文字列のこと。URLの中の「**?**」以降の文字列。
  - 「**? パラメータ名=パラメータの値**」構造になっている。例) `http://127.0.0.1:8000?param=1`

例) リクエストURLが「`http://127.0.0.1:8000?param=1`」で、このリクエストを処理する関数が `index` の場合、

```
1 from django.shortcuts import render
2 from django.http import HttpResponse
3
4 def index(request):
5     if request.method == 'GET':
6         param = request.GET['param']
7         return HttpResponse('Query param: {}'.format(param))
8
9     elif request.method == 'POST':
10        return HttpResponse('POST request received')
```

`request.GET` は、KeyとValueで構成された辞書型オブジェクト。  
この例では、Keyの名前「`param`」で、対応するValue「`1`」を取得。

← → ↺ ⓘ 127.0.0.1:8000/?param=1

Query param: 1

# 手順

1. テンプレートの編集  
→ `blog/templates/blog/index.html`
2. Controllerにおけるリクエストの処理の追加  
→ `blog/views.py`

# 1. テンプレートの編集：

- `blog/templates/blog/index.html` を編集し、ソートのためのリンクを追加します

```
...  
    <div>  
        <button type="submit">Submit</button>  
    </div>  
</form>  
<br>  
  
    <div>  
        <a href="{% url 'index' %}?sort=date">Sort by Date</a> |  
        <a href="{% url 'index' %}?sort=like">Sort by Like</a>  
    </div>  
  
    {% if articles %}  
    {% for article in articles %}  
...  

```

## 2. Controllerへの処理の追加 : `blog/views.py`

- `blog/views.py` の `index` 関数を編集し、クエリーパラメータに応じて、Articleモデルからの検索結果を並べ替えます

```
def index(request):
    if request.method == 'POST':
        article = Article(title=request.POST['title'], body=request.POST['text'])
        article.save()
        return redirect(detail, article.id)

    if ('sort' in request.GET):
        if request.GET['sort'] == 'like':
            articles = Article.objects.order_by('-like')
        else:
            articles = Article.objects.order_by('-posted_at')
    else:
        articles = Article.objects.order_by('-posted_at')

    context = {
        "articles": articles
    }
    return render(request, 'blog/index.html', context)
```

?sort=like : 「いいね」順  
?sort=date : 新着順  
クエリーパラメータなし : 新着順

# 動作確認

- Djangoサーバを起動し、ソート機能が動作することを確認しましょう

The screenshot displays the Django Sample Application interface. At the top, there is a dark blue header with the INIAD logo and the text "Information Networking for Innovation and Design". Below the header, the title "Django Sample Application" is prominently displayed. A message states, "You can submit articles from the following form." The form includes a "Title" field with a placeholder "Input title" and a larger "Text" area. A "Submit" button is located below the form. Underneath the form, there are two links: "Sort by Date" and "Sort by Like". The page lists two articles, each in a light blue box. The first article is titled "The 1st article", dated "[2021年12月4日14:37]", with "Like: 2" and the text "This is the 1st article." The second article is titled "The 2nd article", dated "[2021年12月4日14:37]", with "Like: 0" and the text "This is the 2nd article."

<http://127.0.0.1:8000/>