

Project 4: Django博客系统 (Blog)

姓名: 陈铄涵 学号: 2024140014 课程: Python程序设计 日期: 2025/12/28

1 项目概述与工作内容

本项目基于Django 5.2框架开发，**完整实现了作业4-1和4-5的所有要求**，并进行了多项功能扩展。项目涵盖：**BlogPost数据模型**（包含title、text、date_added、owner字段）、**用户认证系统**（注册/登录/登出）、**文章CRUD操作**（创建、查看、编辑、删除）、**权限保护机制**（仅作者可编辑自己的文章）。在此基础上，额外实现了**标签系统、评论功能、点赞机制、图片上传、全文搜索、用户主页等高级特性**。

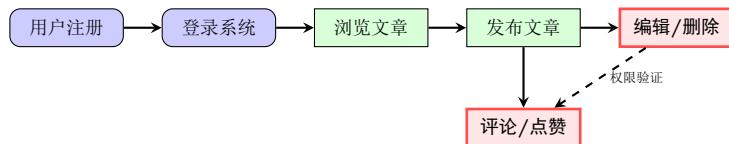


图 1: 项目工作流程图 (红框为扩展功能: 评论点赞系统)

2 项目架构与系统设计

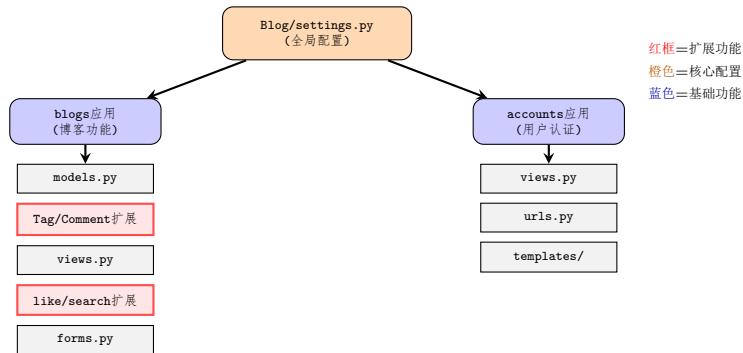


图 2: 项目架构图 (红框标注扩展功能模块)

3 核心代码与技术亮点

1. BlogPost模型

```

1 class BlogPost(models.Model):
2     # ... Required fields ...
3     title = models.CharField(max_length=200)
4     text = models.TextField()
5     date_added = models.DateTimeField(
6         auto_now_add=True)
7     owner = models.ForeignKey(
8         User, on_delete=models.CASCADE)
9
10    # ... Extended fields ...
11    tags = models.ManyToManyField(
12        Tag, blank=True)
13    likes = models.ManyToManyField(
14        User, blank=True,
15        related_name='liked_posts')
16    views = models.PositiveIntegerField(
17        default=0)
18    image = models.ImageField(
19        upload_to='post_images/',
20        blank=True)
21
22    class Meta:
23        ordering = ['-date_added']

```

```

1 @login_required
2 def edit_post(request, post_id):
3     post = get_object_or_404(
4         BlogPost, id=post_id)
5
6     # ... Permission check ...
7     if post.owner != request.user:
8         raise Http404
9
10    if request.method != 'POST':
11        form = BlogPostForm(
12            instance=post)
13    else:
14        form = BlogPostForm(
15            instance=post,
16            data=request.POST,
17            files=request.FILES)
18        if form.is_valid():
19            form.save()
20            messages.success(
21                request, 'Updated!')
22            return redirect(
23                'blogs:post_detail',
24                post_id=post.id)
25
26    return render(request,
27                  'blogs/edit_post.html',
28                  {'post': post, 'form': form})

```

2. 权限保护 (19-5核心)

3. 点赞功能 (扩展)

```

1 @login_required
2 def like_post(request, post_id):
3     post = get_object_or_404(
4         BlogPost, id=post_id)
5
6     # Like/Unlike toggle
7     if post.likes.filter(
8         id=request.user.id).exists():
9         post.likes.remove(request.user)
10        liked = False
11    else:
12        post.likes.add(request.user)

```

```

13        liked = True
14
15    # AJAX response support
16    if request.headers.get(
17        'X-Requested-With') ==
18        'XMLHttpRequest':
19        return JsonResponse({
20            'liked': liked,
21            'total_likes':
22                post.total_likes()
23        })
24
25    return redirect(
26        'blogs:post_detail',
27        post_id=post_id)

```

4. 全文搜索（扩展）

```
1 def search(request):
2     query = request.GET.get('q', '')
3     posts = []
4
```

```
5     if query:
6         # Q object multi-field query
7         posts = BlogPost.objects.filter(
8             Q(title__icontains=query) |
9             Q(text__icontains=query)
10        )
11
12     return render(request,
13         'blogs/search_results.html',
14         {'posts': posts,
15          'query': query})
```

4 功能演示截图

本项目完整实现了作业要求的所有功能，并进行了多项扩展，以下为关键功能的演示截图。

4.1 核心功能：Admin后台与文章管理

Django Admin后台提供完善的管理界面，支持对Blog posts、Comments、Tags的完整CRUD操作。图3展示了Admin首页和添加文章页面，可以看到BlogPost模型包含了作业要求的基础字段（title、text、owner）以及扩展字段（tags、likes、views、image）。

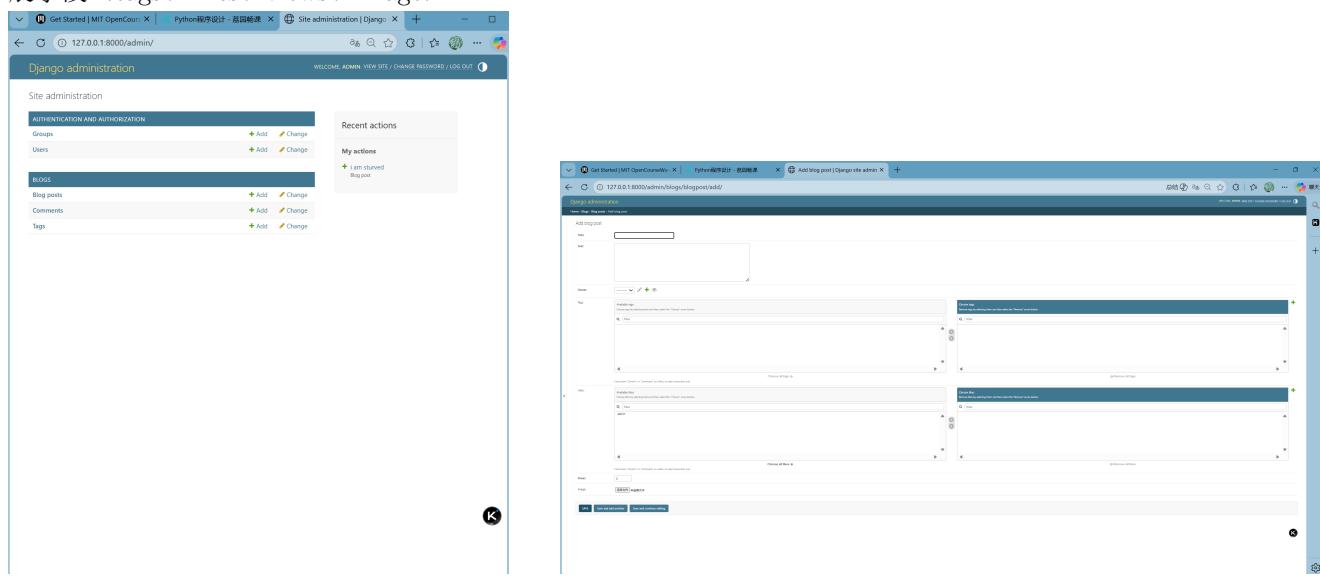


图 3: Django Admin后台管理界面

4.2 核心功能：文章发布与编辑

前台用户界面提供友好的文章发布和编辑功能。图4左图展示发布文章页面，支持输入标题、内容、上传封面图、添加标签；右图为编辑文章页面，系统会验证用户是否为文章作者，只有作者才能编辑和删除自己的文章（4-5核心要求）。

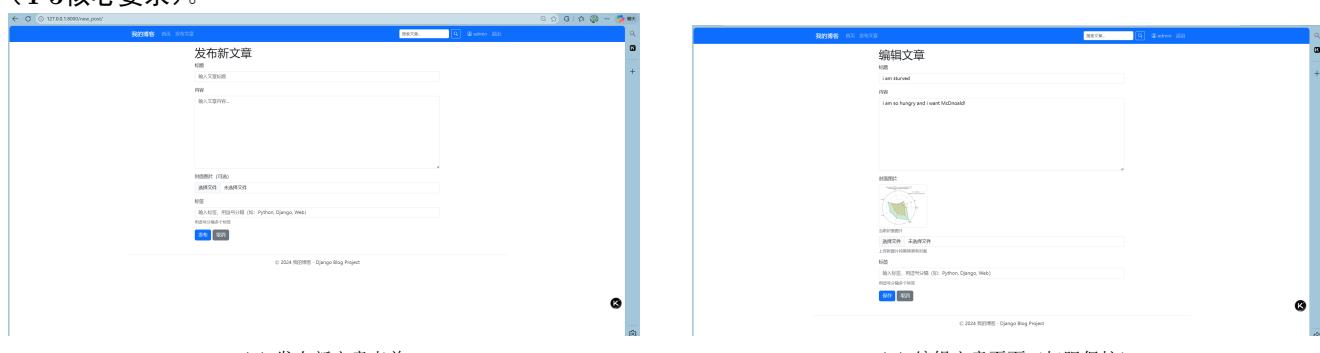


图 4: 文章发布与编辑功能

4.3 扩展功能：社交互动与用户系统

扩展功能包括：(1) 点赞功能（显示点赞数，支持实时点赞/取消）；(2) 浏览量统计；(3) 评论系统；(4) 标签展示；(5) 用户注册/登录；(6) 用户个人主页（统计信息）。图??展示了文章详情页（点赞、评论、浏览量、标签）、用户注册、用户登录和用户个人主页。

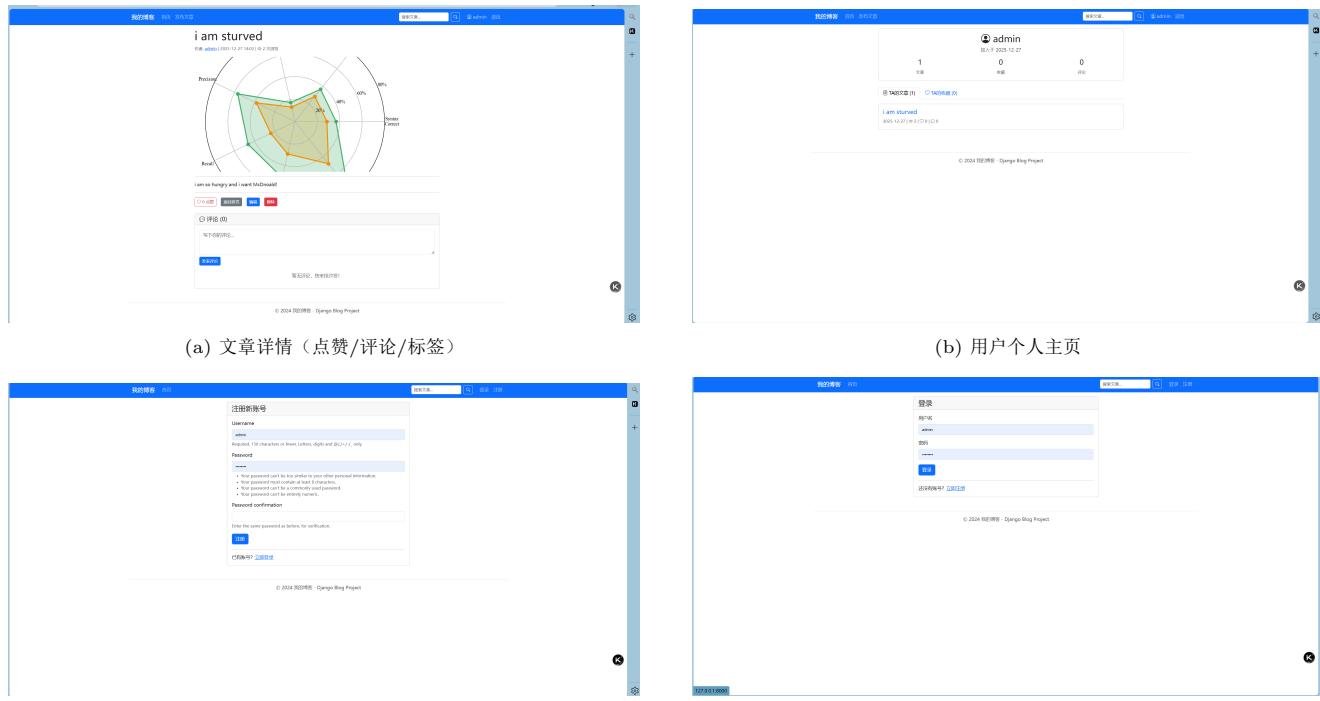


图 5: 扩展功能展示: 社交互动与用户系统

5 技术总结与作业完成情况

- 作业完成度:** 100% 实现 4-1 和 4-5 所有要求。BlogPost 模型包含 title、text、date_added、owner 字段；实现了完整的用户认证系统；文章按时间倒序显示；支持新建和编辑文章；编辑前验证文章所有权 (`if post.owner != request.user: raise Http404`)。
- 扩展功能 (红色标注):** 标签系统 (ManyToMany 关系)、评论功能 (Comment 模型)、点赞机制 (AJAX 实时更新)、图片上传 (ImageField + 媒体文件管理)、全文搜索 (Q 对象多字段查询)、用户主页 (统计信息)、分页显示。
- 技术栈与架构:** Django 5.2 + SQLite + Bootstrap 5 + Pillow；采用 MVT 模式 (Model-View-Template)；模块化设计 (blogs 应用 + accounts 应用)；使用 Django 内置认证系统和 Admin 后台。
- 代码质量:** 使用 `@login_required` 装饰器保护视图；表单验证 (`form.is_valid()`)；消息提示 (Django messages 框架)；遵循 DRY 原则 (Don't Repeat Yourself)；充足的注释说明。

技术特点: Django MVT 架构 —— 用户认证与权限 —— ManyToMany 关系 —— AJAX 异步交互 —— 媒体文件管理

6 安装与运行说明

项目结构: 详见 README.md 文档, 完整说明了目录结构、数据模型、功能特性和使用指南。

7 教师评语

评分: _____ 日期: _____