D-Roger

django实战项目 内容管理系统cms (七)视图层 views.py3/4

D.Roger • 2016年6月29日 • 发表评论

> *立夏：立夏之日蝼蝈鸣，又五日蚯蚓出，又五日王瓜生。*

我们继续写views.py

1. views.article

在views.py中加入以下代码:

```
def article(request, article_id):
    '''
    try:   # since visitor input a url with invalid id
        article = Article.objects.get(pk=article_id)  # pk???
    except Article.DoesNotExist:
        raise Http404("Article does not exist")
    ''' # shortcut:
    article = get_object_or_404(Article, id=article_id)
    content = markdown2.markdown(article.content, extras=["code-friendly",
        "fenced-code-blocks", "header-ids", "toc", "metadata"])
    commentform = CommmentForm()
    loginform = LoginForm()
    comments = article.comment_set.all

    return render(request, 'article_page.html', {
        'article': article,
        'loginform':loginform,
        'commentform':commentform,
        'content': content,
        'comments': comments
```

```
21          })
```

当用户直接输入url来访问页面，而页面不存在时我们要进行相关处理，django内置了一个快捷函数get_object_or_404()，参见 django 文档 。

我们在管理页面编辑文章，内容部分并没有支持富文本，django可以集成各种富文本编辑器，如果你想集成ueditor可以参阅 这篇文章，我个人更喜欢用markdown来写文章，所以在django中用markdown2这个模块来实现markdown功能。你可以去github了解这个模块 github markdown2。

需要注意的是代码语法高亮的问题，我们需要去到 这里 下载语 法高亮的css文件，选择其中一个css文件，复制内容存入名为Pygments.css的文件，将此文件放入static/css文件夹中，在article_page.html(我们直接复制index.html去掉<!– Center Column –>部分，放在index.html所在文件夹) 中插入以下代码：

```
1 <link href="{% static "css/Pygments.css" %}" rel="stylesheet">
```

2. article_page.html

我们直接复制index.html为article_page.html去掉<!– Center Column –>部分，放在index.html所在文件夹。然后再去掉<!– Left Column –>部分，更改中间部分（文章界面）所占的比例：

```
1         <!-- Center Column -->
2          <div class="col-sm-9">
```

将6改为9。

接下来写article_page的<!– Center Column –>部分：

```
1         <!-- Center Column -->
2          <div class="col-sm-9">
3              <!-- Article -->
4              <div class="row">
5                  <article class="col-xs-12">
6                      <h2>{{ article.title }}</h2>
7                      <p class="pull-right"><span class="label label-default">{{ article.columr
8                      <p><hr></p>
9                      <p>{{ content | safe }}</p>
10                     <p></p>
11                     <ul class="list-inline">
12                         <li><a href=" "><span class="glyphicon glyphicon-comment"></span> {{
13                         <li><a href="/focus/{{ article.id }}/keep/"><span class="glyphicon gl
14                         <li><a href="/focus/{{ article.id }}/poll/"><span class="glyphicon gl
15                     </ul>
16                 </article>
17             </div>
18             <hr>
19
20             <!-- Comments -->
21             <h3>Comments:</h3>
22             <hr>
23             {% for comment in comments %}
24             <div class="row">
25
26                 <article class="col-xs-12">
27                     <p class="pull-right"> <span class="label label-default">tag</span> <spar
28                     <p>{{ comment.content }}</p>
29                     <ul class="list-inline">
```

```
                        <li><a href=" ">{{ comment.pub_date | date:"j M" }}</a></li>
                        <li><a href=" "><span class="glyphicon glyphicon-comment"></span> {{
                        <li><a href=" "><span class="glyphicon glyphicon-thumbs-up"></span>{{
                    </ul>
                </article>
            </div>
            <hr>
            {% endfor %}
            <div class="row">
                <article class="col-xs-12">
                    <h4>Sharing you comment:</h4>
                    <form action="/focus/{{ article.id }}/comment/" method="post">
                        {% csrf_token %}
                        {{ commentform.as_p }}
                        <input class="btn btn-default" type="submit" value="Submit" />
                    </form>
                </article>
            </div>
            <hr>
        </div><!--/Center Column-->
```

包含三部分：文章，对应的评论，评论框。

> *我们下载的这个模版页面内容虽然很多，但是层次很清晰，因为作者不但有良好的缩进，而且区块的开始和结束都有注释，代码清晰则利于查错，修改。所以，我们自己写的时候也要养成好习惯，层次分明，注释和命名都恰到好处。*

3. 评论 点赞 收藏

在views.py中加入以下代码:

```
@login_required
def comment(request, article_id):
    form  = CommmentForm(request.POST)
    url = urlparse.urljoin('/focus/', article_id)
    if form.is_valid():
        user = request.user
        article = Article.objects.get(id=article_id)
        new_comment = form.cleaned_data['comment']
        c = Comment(content=new_comment, article_id=article_id)  # have tested by shell
        c.user = user
        c.save()
        article.comment_num += 1
    return redirect(url)

@login_required
def get_keep(request, article_id):
    logged_user = request.user
    article = Article.objects.get(id=article_id)
    articles = logged_user.article_set.all()
    if article not in articles:
        article.user.add(logged_user)  # for m2m linking, have tested by shell
        article.keep_num += 1
        article.save()

        return redirect('/focus/')
    else:
        url = urlparse.urljoin('/focus/', article_id)
        return redirect(url)

@login_required
def get_poll_article(request,article_id):
    logged_user = request.user
```

```
33      article = Article.objects.get(id=article_id)
34      polls = logged_user.poll_set.all()
35      articles = []
36      for poll in polls:
37          articles.append(poll.article)
38
39      if article in articles:
40          url = urlparse.urljoin('/focus/', article_id)
41          return redirect(url)
42      else:
43          article.poll_num += 1
44          article.save()
45          poll = Poll(user=logged_user, article=article)
46          poll.save()
47          data = {}
48          return redirect('/focus/')
```

三者都需要登录才能操作。

article_id这个参数是通过article_page.html文件里href传入到urls.py然后在传入到相关处理函数的。

urlparse.urljoin()函数是python用来拼接url的，后期爬虫课程也会用到这个。

redirect()函数是django内置的一个快捷函数。

第19行，因为文章与用户是多对多关系，所以logged_user.article_set.all()会得到这个登录用户对应的所有文章(即它收藏的文章)，参见 django 文档 。

标签： Django

««django实战项目 内容管理系统cms (六)视图层views.py2/4

django实战项目 内容管理系统cms (八)视图层views.py4/4»»

暂无评论

发表评论

已登入为D.Roger。登出?

评论

发表评论