

Django记录

1、安装Django

Win+ r 输入cmd

输入：

```
pip install django==4.1 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

2、创建Django项目

方法有两种

第一种：使用Pycharm专业版直接创建Django项目即可

第二种：手动创建

1. 在任意位置创建一个文件夹，命名就是你项目的名字
2. 双击进入该文件夹，点击搜索框输入cmd，开启命令行
3. 输入命令，创建虚拟环境

```
python -m venv 项目名称_venv
```

4. 输入

```
项目名称_venv\Scripts\activate
```

激活虚拟环境

deactivate:用来关闭虚拟环境

5. 输入以下命令创建项目

```
django-admin startproject 项目名称 .
```

举例:django-admin startproject ll .

然后安装Django

```
pip install django==4.1 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

7、输入以下创建数据库

```
py manage.py migrate
```

8、启动

```
py manage.py runserver
```

9、浏览器输入127.0.0.1:8000 访问

3、创建App

1、在上面的启动之后，回到manage.py所在目录，在搜索框输入cmd再开一个命令行窗口

激活虚拟环境

```
项目名称_venv\Scripts\activate
```

2、创建App

```
py manage.py startapp learning_logs
```

3、定义数据模型Topic

```
from django.db import models

# Create your models here.

class Topic(models.Model):
    """用户学习的主题"""
    text = models.CharField(max_length=256)
    data_added = models.DateTimeField(auto_now=True)

    def __str__(self):
        """返回模型的字符串显示"""
        return self.text
```

4、更改语言时区、语言

```
LANGUAGE_CODE = 'zh-Hans'
TIME_ZONE = 'Asia/Shanghai'
```

5、激活模型

找到settings.py文件

```
INSTALLED_APPS = [  
    'learning_logs',  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```

由于创建了新的表（模型），需要迁移数据库中的数据。新建一个终端，输入以下命令

```
py manage.py makemigrations learning_logs
```

```
py manage.py migrate
```

```
(ll_env) PS C:\Users\Administrator\Desktop\learning_log> py manage.py makemigrations learning_logs  
Migrations for 'learning_logs':  
  learning_logs\migrations\0001_initial.py  
    - Create model Topic  
(ll_env) PS C:\Users\Administrator\Desktop\learning_log> py manage.py migrate  
Operations to perform:
```

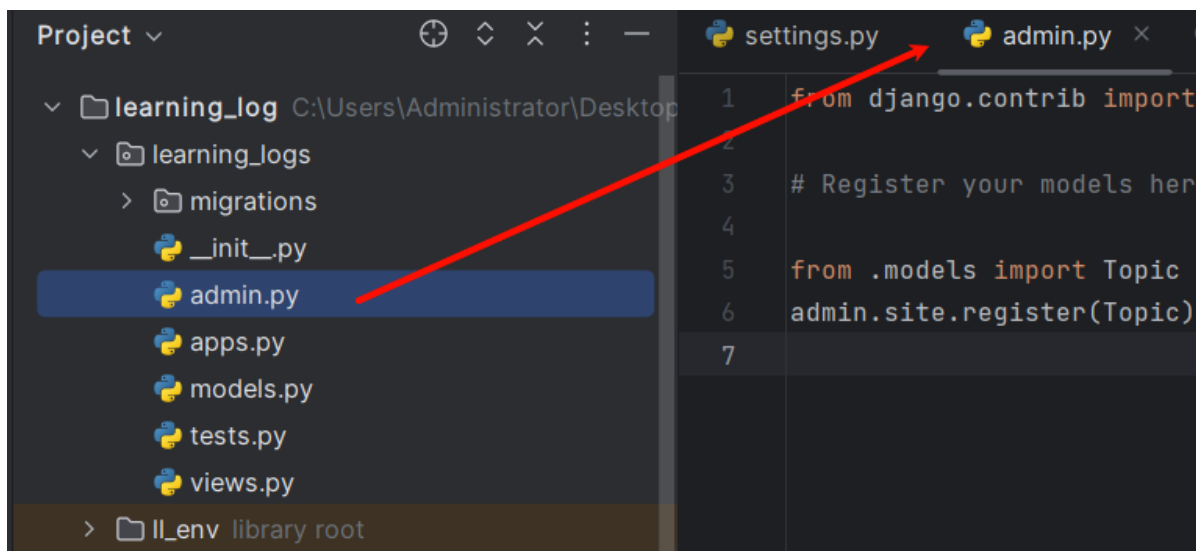
```
(ll_env) PS C:\Users\Administrator\Desktop\learning_log> py manage.py migrate  
Operations to perform:  
  Apply all migrations: admin, auth, contenttypes, learning_logs, sessions  
Running migrations:  
  Applying learning_logs.0001_initial... OK
```

4、创建超级管理员（后台管理员）

```
py manage.py createsuperuser
```

向管理网站注册 Topic，请输入下面的代码：

```
from django.contrib import admin  
  
# Register your models here.  
  
from .models import Topic  
admin.site.register(Topic)
```



定义Entry模型

```
class Entry(models.Model):  
    """某个主题的具体知识"""  
    topic = models.ForeignKey(Topic, on_delete=models.CASCADE)  
    text = models.TextField()  
    date_added = models.DateTimeField(auto_now=True)  
  
    class Meta:  
        verbose_name_plural = 'entries'  
  
    def __str__(self):  
        return f"{self.text[:50]}..."
```

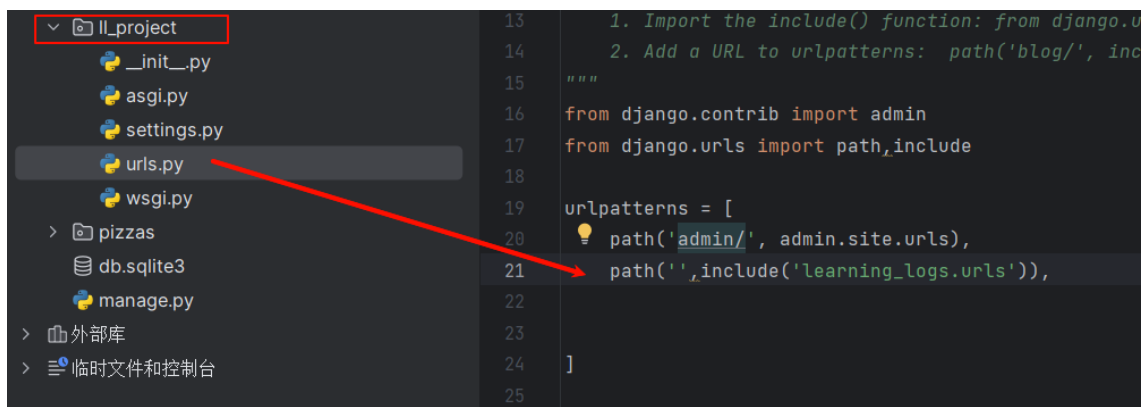
迁移数据库

```
py manage.py makemigrations learning_logs  
  
py manage.py migrate
```

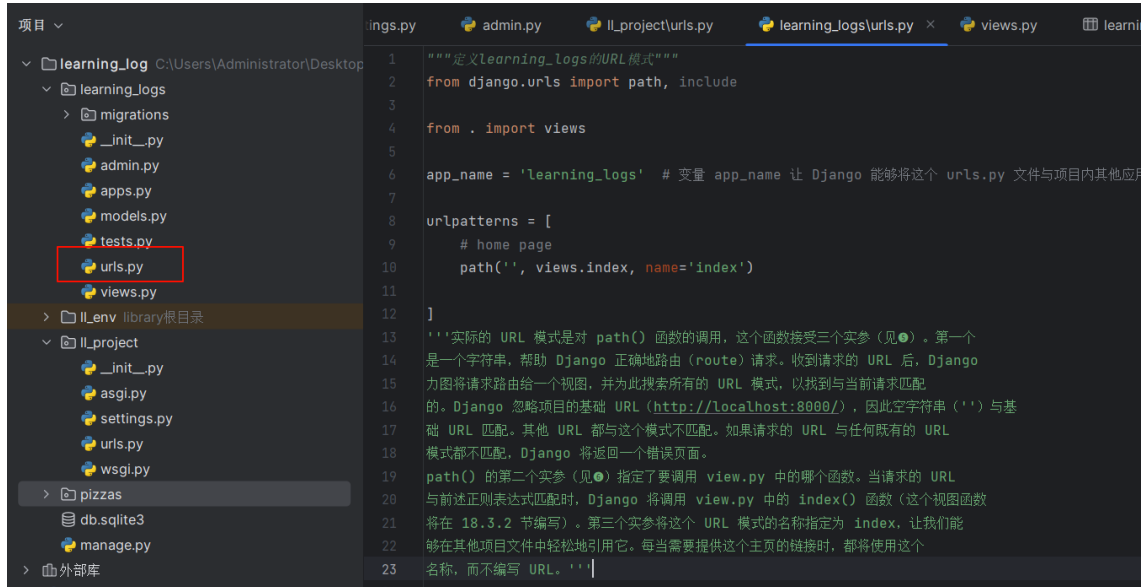
5、创建网页

步骤：定义URL模式、编写视图函数、编写模板 (html)

1. 定义URL模式，在项目的urls文件中导入app的路由



2. 在learning_log app目录中新建urls文件，配置路由

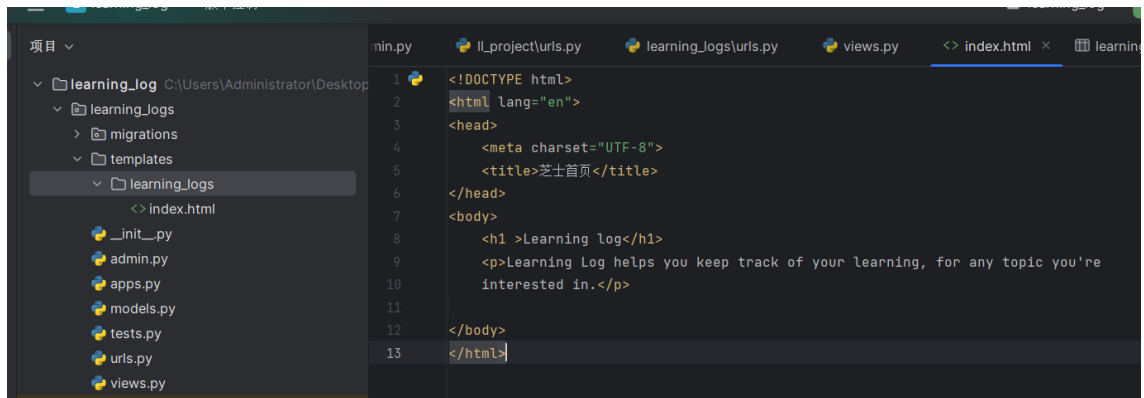


3. 编写index视图函数

```
from django.shortcuts import render

# Create your views here.
def index(request):
    '''学习笔记本主页'''
    return render(request, "learning_logs/index.html")
```

4. 新建templates目录，在里面创建learning_logs目录，用于存放learning_logs的前端界面



5. 界面展示



Learning log

Learning Log helps you keep track of your learning, for any topic you're interested in.

6、创建其他网页

在index同一目录下，新建一个base作为模板，里面包含了许多网页共同的排版，节省开发时间，提高效率

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Base</title>
</head>
<body>
    <p>
        <a href="{% url 'learning_logs:index' %}">learning Log</a>&nbsp;
        <a href="{% url 'learning_logs:topics' %}">topics</a>
    </p>
    {% block content %}{% endblock content %}
    {#我们在最后一行插入了一对块标签这个块名为 content，是一个占位#}
    {#符，其中包含的信息由子模板指定。#}
    {#子模板并非必须定义父模板中的每个块，因此在父模板中，可以使用任意多个块#}
    {#来预留空间，而子模板可根据需要定义相应数量的块。#}

</body>
</html>
```

1、新建路由topics

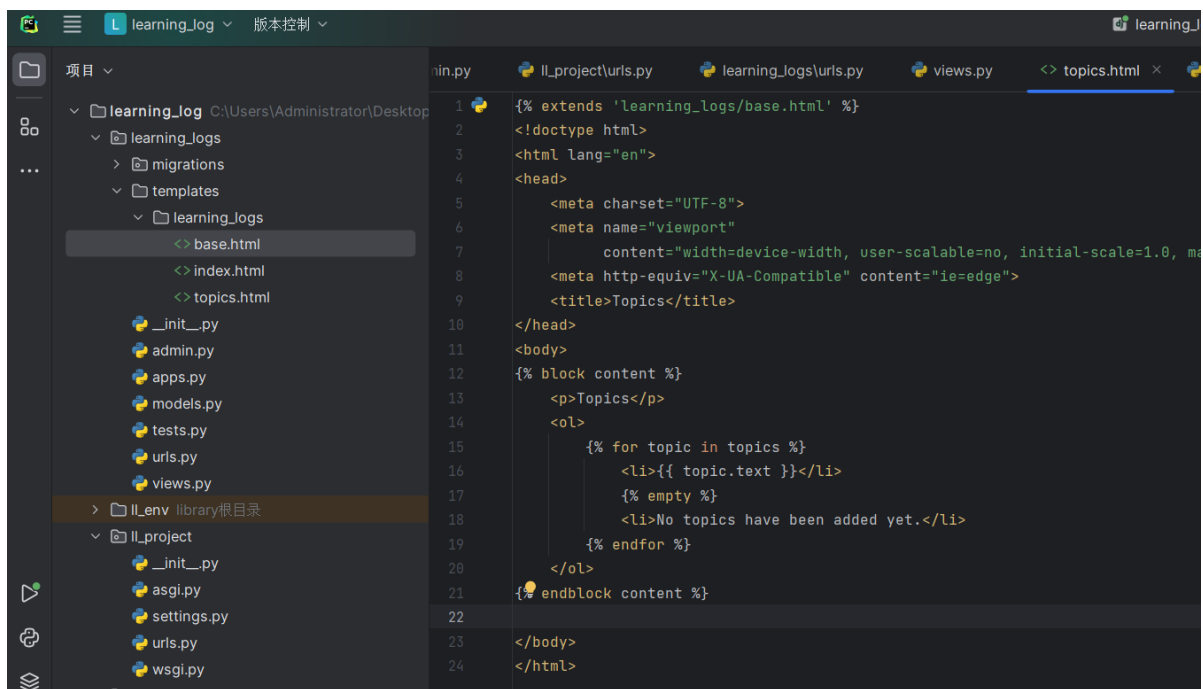


```
1 """定义learning_logs的URL模式"""
2 from django.urls import path, include
3
4 from . import views
5
6 app_name = 'learning_logs' # 变量 app_name 让 Django 能够将这个 urls.py 文件与项目内其他应用程序中的同名文件区分开来
7
8 urlpatterns = [
9     # home page
10     path('', views.index, name='index'),
11     path('topics/', views.topics, name='topics'),
12 ]
13
14 # 实际的 URL 模式是针对 as_view() 函数的调用。这个函数接受三个参数（见 #）。第一个
```

2、编写topics视图函数

```
def topics(request):
    '''学习笔记主题'''
    topics = Topic.objects.order_by("date_added")
    context = {"topics": topics}
    return render(request, "learning_logs/topics.html", context)
```

3、新建topics网页

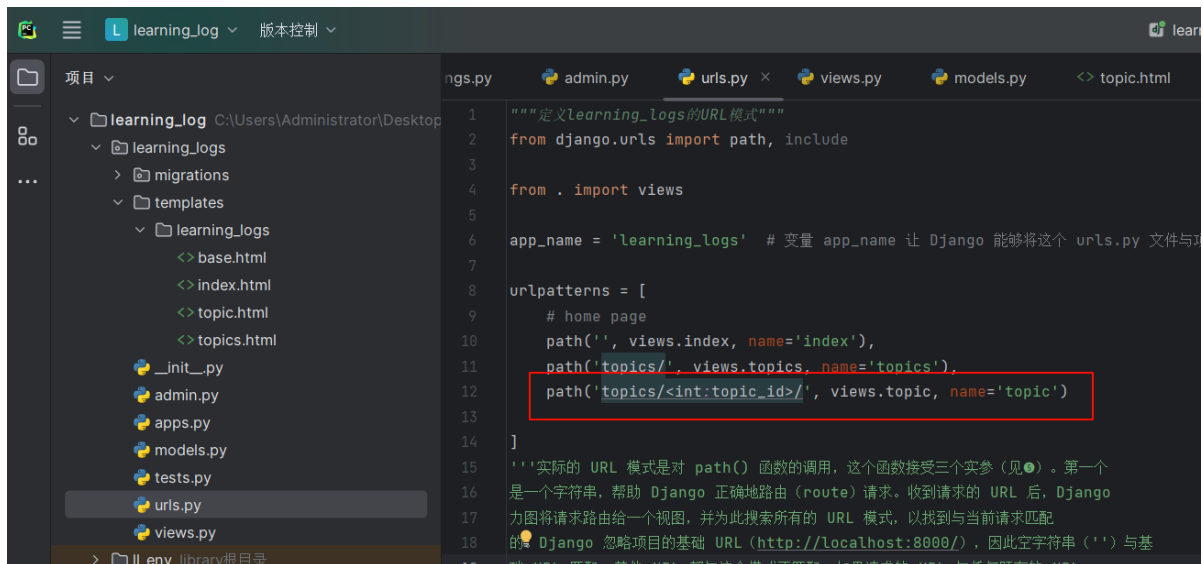


```
{% extends 'learning_logs/base.html' %}
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Topics</title>
</head>
<body>
{% block content %}
    <p>Topics</p>
    <ol>
        {% for topic in topics %}
            <li>{{ topic.text }}</li>
            {% empty %}
            <li>No topics have been added yet.</li>
        {% endfor %}
    </ol>
{% endblock content %}

</body>
</html>
```

显示特定主题の詳細页面

1、定义路由



2、定义视图函数 topic

```
def topic(request, topic_id):  
    """显示单个主题及其所有条目"""  
    topic = Topic.objects.get(id=topic_id)  
    entries = topic.entry_set.order_by("-date_added") #- 表示降序  
    context = {"topic": topic, "entries": entries}  
    return render(request, "learning_logs/topic.html", context)
```

3、编写模板topic.html

```
<!doctype html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport"  
        content="width=device-width, user-scalable=no, initial-scale=1.0,  
maximum-scale=1.0, minimum-scale=1.0">  
    <meta http-equiv="X-UA-Compatible" content="ie=edge">  
    <title>Topic</title>  
</head>  
<body>  
{% extends 'learning_logs/base.html' %}  
{% block content%}  
    <p>Topics: {{ topic.text }}</p>  
    <p>Entries:</p>  
    <ol>  
        {% for entry in entries %}  
            <li>  
                <p>{{entry.date_added|date:'M d, Y H:i'}}</p>  
                <p>{{entry.text|linebreaks}}</p>  
            </li>  
            {% empty %}  
                <li>There are no entries for this topic yet.</li>  
            {% endfor %}  
    </ol>  
</body>
```



```
</ol>
{% endblock content%}
</body>
</html>
```

7、用户账户

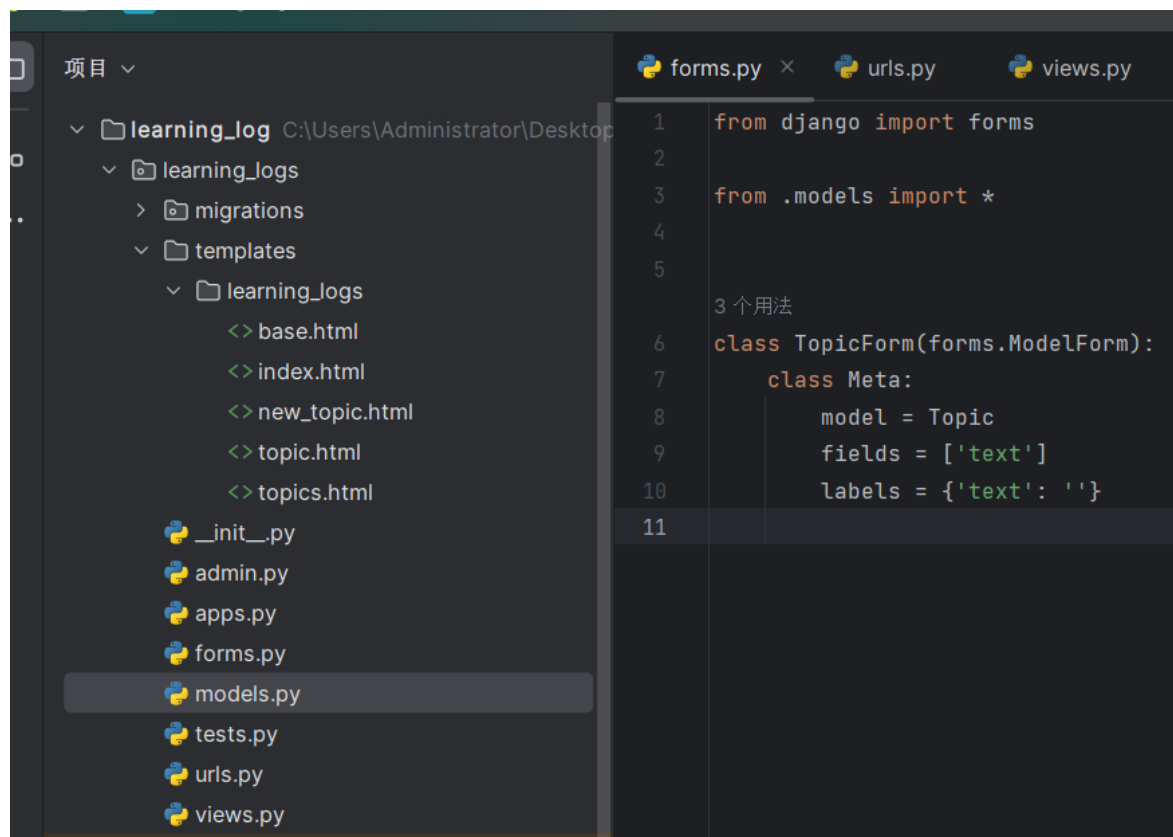
1、添加新主题 (Topic)

1、编辑forms.py文件

```
from django import forms

from .models import *

class TopicForm(forms.ModelForm):
    class Meta:
        model = Topic
        fields = ['text']
        labels = {'text': ''}
```



2、定义URL

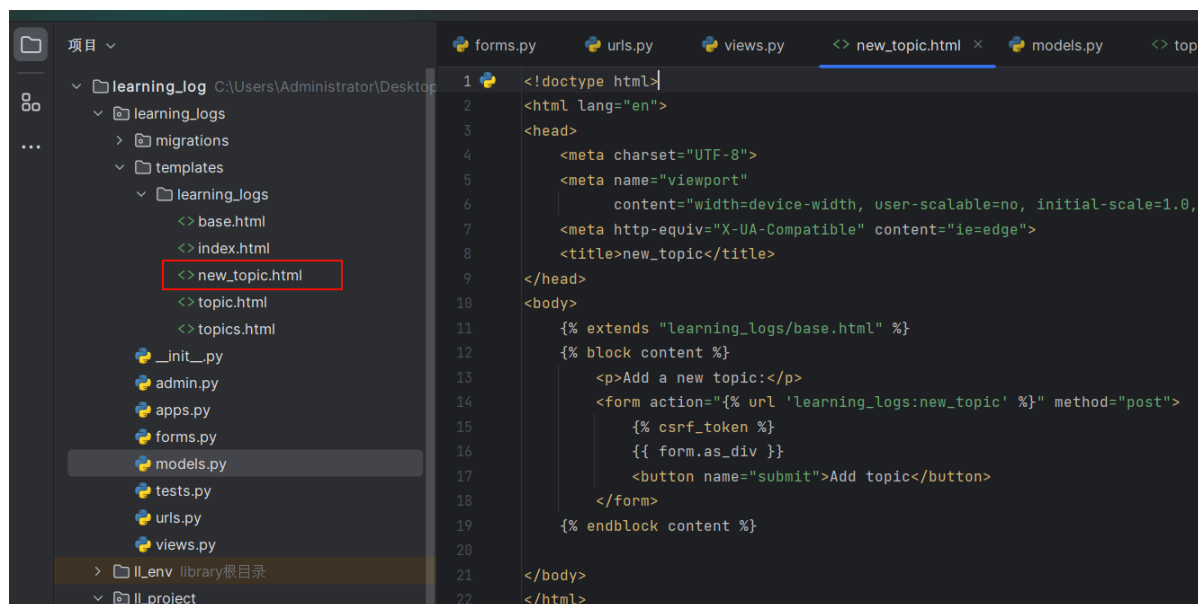
```
urlpatterns = [
    # home page
    path('', views.index, name='index'),
    path('topics/', views.topics, name='topics'),
    path('topics/<int:topic_id>/', views.topic, name='topic'),
    path('new_topic', views.new_topic, name='new_topic')
]
```

3、定义view函数

```
def new_topic(request):
    """添加新主题"""
    if request.method != 'POST':
        """未提交数据: 创建一个新表单"""
        form = TopicForm()
    else:
        form = TopicForm(data=request.POST)
        if form.is_valid():
            form.save()
            return redirect('learning_logs:topics')

    context = {'form': form}
    return render(request, 'learning_logs/new_topic.html', context)
```

4、定义new_topic.html



```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
        maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>new_topic</title>
</head>
<body>
    {% extends "learning_logs/base.html" %}
    {% block content %}
        <p>Add a new topic:</p>
        <form action="{% url 'learning_logs:new_topic' %}" method="post">
            {{ csrf_token }}
            {{ form.as_div }}
            <button name="submit">Add topic</button>
        </form>
    {% endblock content %}
</body>
</html>
```

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
```

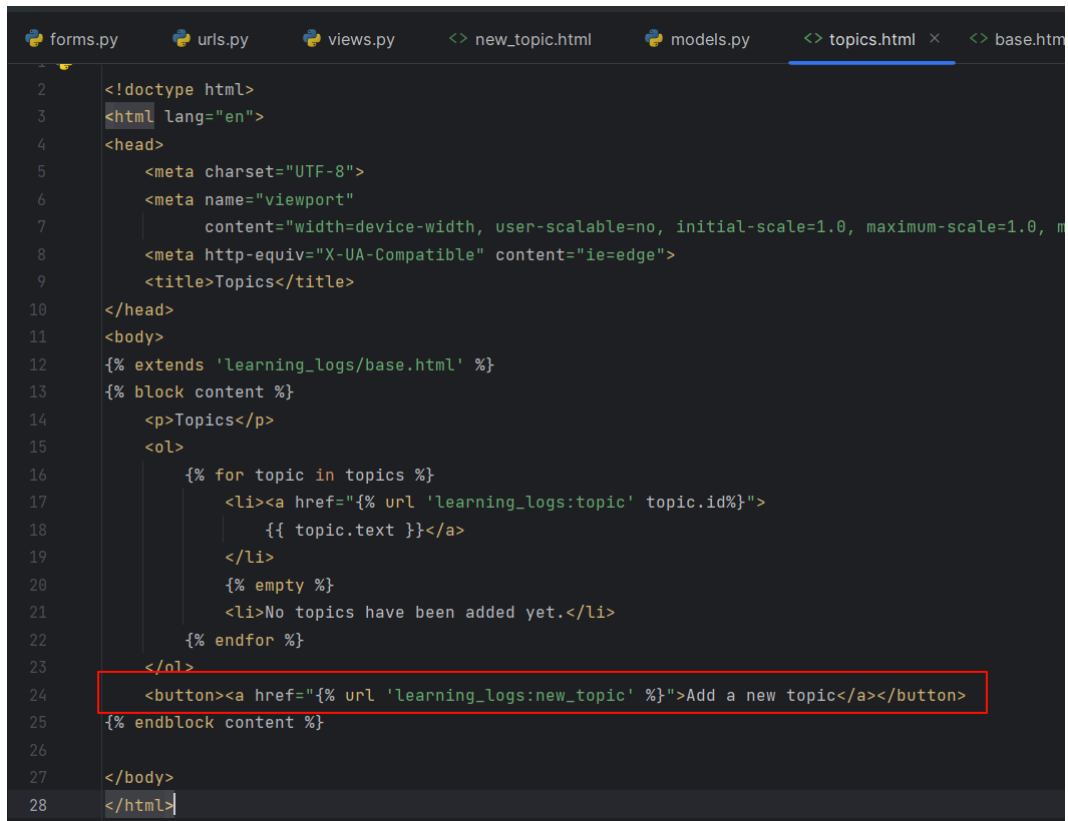
```

        content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <title>new_topic</title>
    </head>
    <body>
        {% extends "learning_logs/base.html" %}
        {% block content %}
            <p>Add a new topic:</p>
            <form action="{% url 'learning_logs:new_topic' %}" method="post">
                {% csrf_token %}
                {{ form.as_div }}
                <button name="submit">Add topic</button>
            </form>
        {% endblock content %}

    </body>
</html>

```

5、在topics html文件中新建链接



```

forms.py  urls.py  views.py  new_topic.html  models.py  topics.html  base.html
2  <!doctype html>
3  <html lang="en">
4  <head>
5      <meta charset="UTF-8">
6      <meta name="viewport"
7          content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, m
8      <meta http-equiv="X-UA-Compatible" content="ie=edge">
9      <title>Topics</title>
10 </head>
11 <body>
12 {% extends 'learning_logs/base.html' %}
13 {% block content %}
14     <p>Topics</p>
15     <ol>
16         {% for topic in topics %}
17             <li><a href="{% url 'learning_logs:topic' topic.id%}">
18                 {{ topic.text }}</a>
19             </li>
20             {% empty %}
21             <li>No topics have been added yet.</li>
22         {% endfor %}
23     </ol>
24     <button><a href="{% url 'learning_logs:new_topic' %}">Add a new topic</a></button>
25 {% endblock content %}
26
27 </body>
28 </html>

```

2、添加新条目 (Entry)

1、用于添加新条目的表单

```

class EntryForm(forms.ModelForm):
    class Meta:
        model = Entry
        fields = ['text']
        labels = {'text': ''}

```

```
widgets = {'text': forms.Textarea(attrs={'cols': 80})}
```

2、定义URL

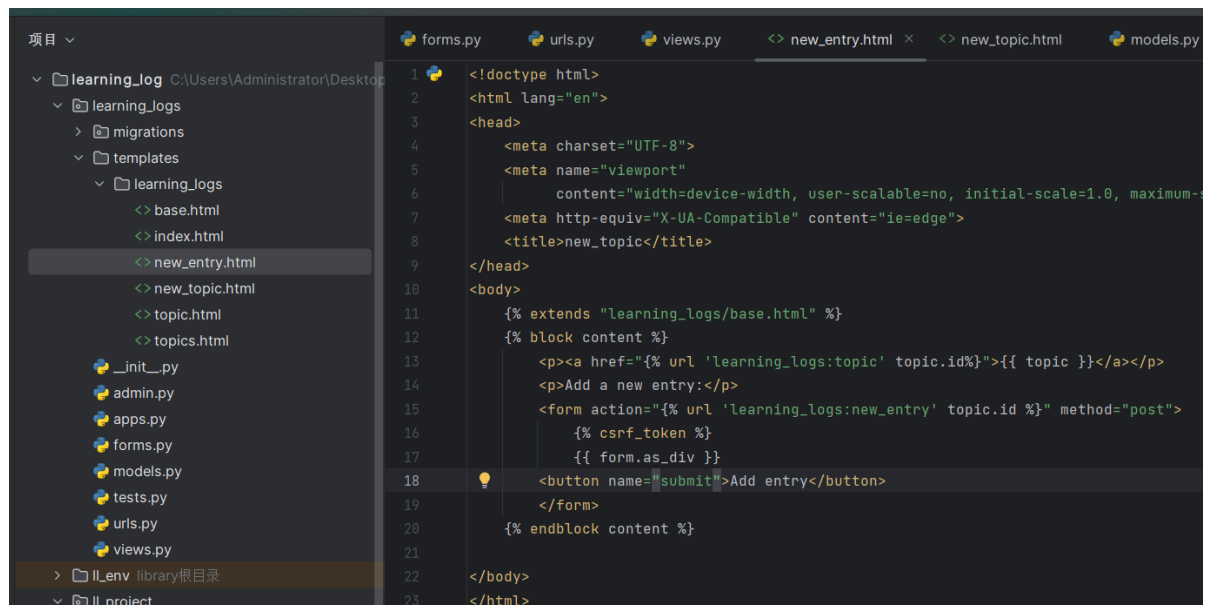
```
urlpatterns = [  
    # home page  
    path('', views.index, name='index'),  
    path('topics/', views.topics, name='topics'),  
    path('topics/<int:topic_id>/', views.topic, name='topic'),  
    path('new_topic', views.new_topic, name='new_topic'),  
    path('new_entry/<int:topic_id>/', views.new_entry, name='new_entry'),  
]
```

```
urlpatterns = [  
    # home page  
    path('', views.index, name='index'),  
    path('topics/', views.topics, name='topics'),  
    path('topics/<int:topic_id>/', views.topic, name='topic'),  
    path('new_topic', views.new_topic, name='new_topic'),  
    path('new_entry/<int:topic_id>/', views.new_entry, name='new_entry'),  
]
```

3、定义view函数

```
def new_entry(request, topic_id):  
    """在特定主题下添加新条目"""  
    topic = Topic.objects.get(id=topic_id)  
    if request.method != 'POST':  
        form = EntryForm()  
    else:  
        form = EntryForm(data=request.POST)  
        if form.is_valid():  
            new_entry = form.save(commit=False)  
            new_entry.topic = topic  
            new_entry.save()  
            return redirect('learning_logs:topic', topic_id=topic_id)  
  
    context = {'topic': topic, 'form': form}  
    return render(request, 'learning_logs/new_entry.html', context)
```

4、定义new_entry.html文件

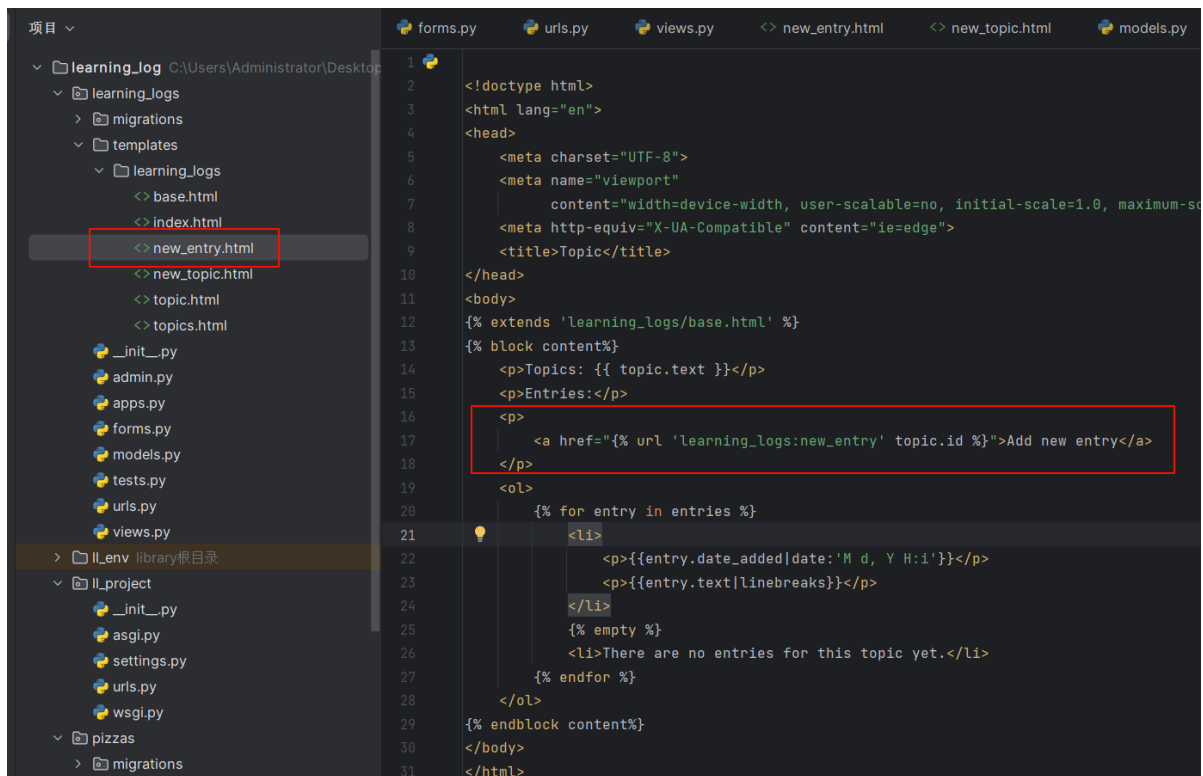


```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport"
6     content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-
7     <meta http-equiv="X-UA-Compatible" content="ie=edge">
8   <title>new_topic</title>
9 </head>
10 <body>
11   {% extends "learning_logs/base.html" %}
12   {% block content %}
13     <p><a href="{% url 'learning_logs:topic' topic.id%}">{{ topic }}</a></p>
14     <p>Add a new entry:</p>
15     <form action="{% url 'learning_logs:new_entry' topic.id %}" method="post">
16       {% csrf_token %}
17       {{ form.as_div }}
18     <button name="submit">Add entry</button>
19   </form>
20   {% endblock content %}
21
22 </body>
23 </html>
```

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
    content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>new_topic</title>
</head>
<body>
  {% extends "learning_logs/base.html" %}
  {% block content %}
    <p><a href="{% url 'learning_logs:topic' topic.id%}">{{ topic }}</a></p>
    <p>Add a new entry:</p>
    <form action="{% url 'learning_logs:new_entry' topic.id %}"
method="post">
      {% csrf_token %}
      {{ form.as_div }}
    <button name="submit">Add entry</button>
  </form>
  {% endblock content %}

</body>
</html>
```

5、在topic html文件中新建添加新条目链接



3、编辑条目

1、定义URL

```
urlpatterns = [
    # home page
    path('', views.index, name='index'),
    path('topics/', views.topics, name='topics'),
    path('topics/<int:topic_id>/', views.topic, name='topic'),
    path('new_topic', views.new_topic, name='new_topic'),
    path('new_entry/<int:topic_id>/', views.new_entry, name='new_entry'),
    path('edit_entry/<int:entry_id>', views.edit_entry, name='edit_entry'),
]
```

2、定义view函数

```
def edit_entry(request, entry_id):
    """编辑现有的条目"""
    entry = Entry.objects.get(id=entry_id)
    topic = entry.topic
    if request.method != 'POST':
        # 初次请求：使用当前的条目填充表单
        form = EntryForm(instance=entry)
    else:
        # post提交的数据，对数据进行处理
        form = EntryForm(instance=entry, data=request.POST)
        if form.is_valid():
            form.save()
```

```
        return redirect('learning_logs:topic', topic_id=topic.id)

    context = {'entry': entry, 'topic': topic, 'form': form}
    return render(request, 'learning_logs/edit_entry.html', context)
```

3、定义editentryhtml文件

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>edit_entry</title>
</head>
<body>
    {% extends "learning_logs/base.html" %}
    {% block content %}
        <p><a href="{% url 'learning_logs:topic' topic.id%}">{{ topic }}</a></p>
        <p>Edit entry:</p>
        <form action="{% url 'learning_logs:edit_entry' entry.id %}"
method="post">
            {% csrf_token %}
            {{ form.as_div }}
            <button name="submit">Save changes</button>
        </form>
    {% endblock content %}

</body>
</html>
```

4、在topichtml文件中新建编辑条目链接

```
forms.py  urls.py  views.py  <> edit_entry.html  <> new_entry.html  <> topic.html  ×  models.py

6      <meta name="viewport"
7          content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minim
8      <meta http-equiv="X-UA-Compatible" content="ie=edge">
9      <title>Topic</title>
10     </head>
11     <body>
12     {% extends 'learning_logs/base.html' %}
13     {% block content%}
14     <p>Topics: {{ topic.text }}</p>
15     <p>Entries:</p>
16     <p>
17         <a href="{% url 'learning_logs:new_entry' topic.id %}">Add new entry</a>
18     </p>
19     <ol>
20         {% for entry in entries %}
21         <li>
22             <p>{{entry.date_added|date:'M d, Y H:i'}}</p>
23             <p>{{entry.text|linebreaks}}</p>
24             <p>
25                 <a href="{% url 'learning_logs:edit_entry' entry.id%}">Edit Entry</a>
26             </p>
27         </li>
28         {% empty %}
29         <li>There are no entries for this topic yet.</li>
30         {% endfor %}
31     </ol>
32     {% endblock content%}
33     </body>
34 </html>
```

4、创建用户账户

1、登录页面

创建管理用户账户的app:

```
py manage.py startapp accounts
```

在settings文件中注册app

```
33 INSTALLED_APPS = [
34     'accounts',
35     'learning_logs',
36     'pizzas',
37     'django.contrib.admin',
38     'django.contrib.auth',
39     'django.contrib.contenttypes',
40     'django.contrib.sessions',
41     'django.contrib.messages',
42     'django.contrib.staticfiles',
43
44 ]
```


在主路由中，添加accounts urls

```
urls.py x
1 """ll_project URL Configuration
2
3 The 'urlpatterns' list routes URLs to views. For more information see the
4     https://docs.djangoproject.com/en/4.1/topics/http/urls/
5 Examples:
6 Function views
7     1. Add an import: from my_app import views
8     2. Add a URL to urlpatterns: path('', views.home, name='home')
9 Class-based views
10    1. Add an import: from other_app.views import Home
11    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13    1. Import the include() function: from django.urls import include
14    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path('accounts/', include('accounts.urls')),
22     path('', include('learning_logs.urls')),
23 ]
24
25 ]
26
```

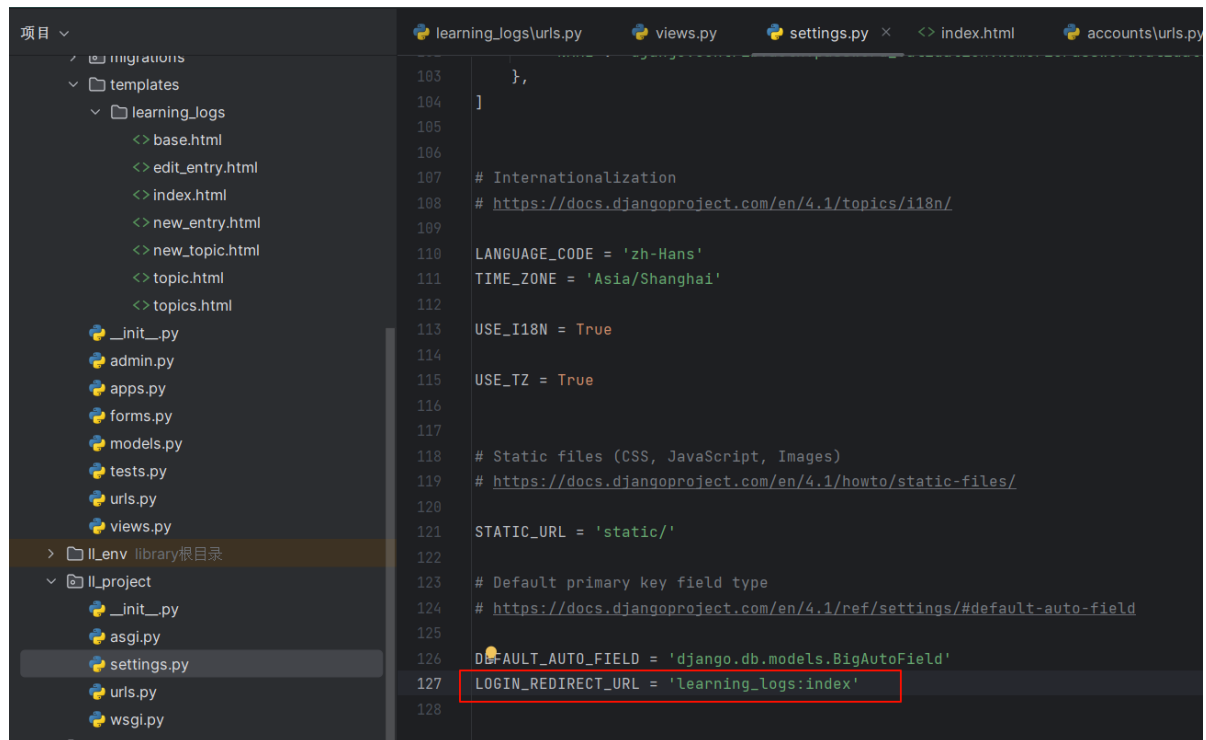
定义accounts urls

项目

- learning_log C:\Users\Administrator\Desktop\learning_log
- accounts
 - migrations
 - templates
 - registration
 - login.html
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - urls.py
 - views.py

```
ll_project\urls.py accounts\urls.py x login.html x
1 """为accounts定义URL模式"""
2
3 from django.urls import path, include
4
5 app_name = 'accounts'
6 urlpatterns = [
7     path('', include('django.contrib.auth.urls')),
8 ]
9
```

在主settings中让django知道用户成功登录后应该重定向到哪里



编写login html页面

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
</head>
<body>
{% extends 'learning_logs/base.html' %}
{% block content %}
    {% if form.errors %}
        <p>Your username and password didn't match. Please try again.</p>
    {% endif %}
    <form action="{% url 'accounts:login' %}" method="post">
        {% csrf_token %}
        {{ form.as_div }}
        <button name="submit">Log in</button>
    </form>
{% endblock content %}
</body>
</html>
```

在base.html页面添加login in选项

127.0.0.1:8000/topics/

谷歌更新 百度一下 hao123 网址导航 360搜索

[learning Log](#) [topics](#) [Log in](#)

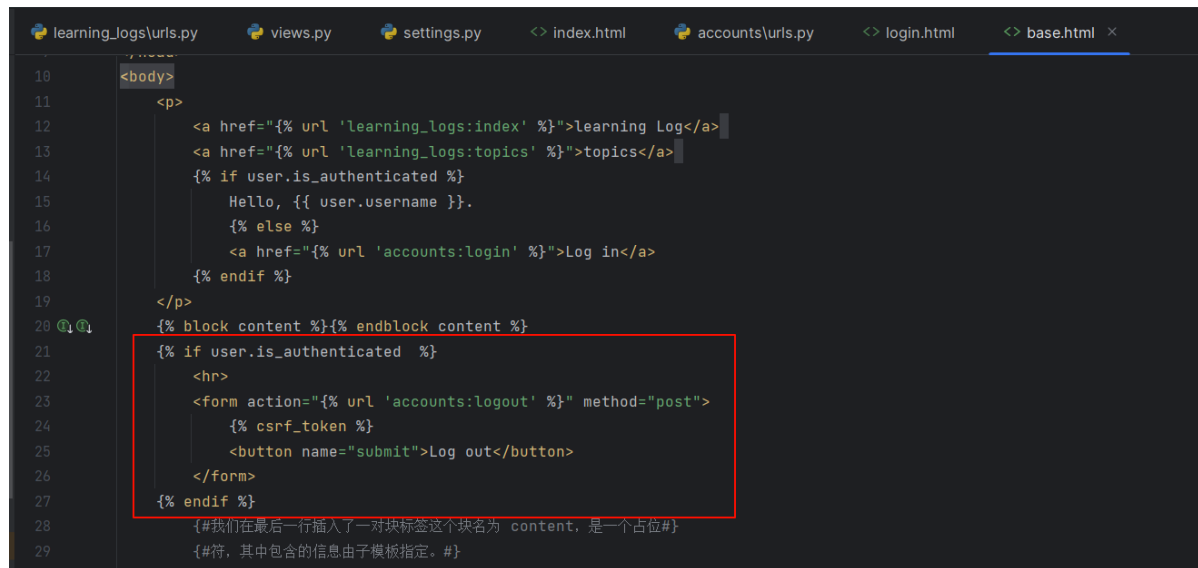
Topics

1. [RockClimbing](#)
2. [Chess](#)

[Add a new topic](#)

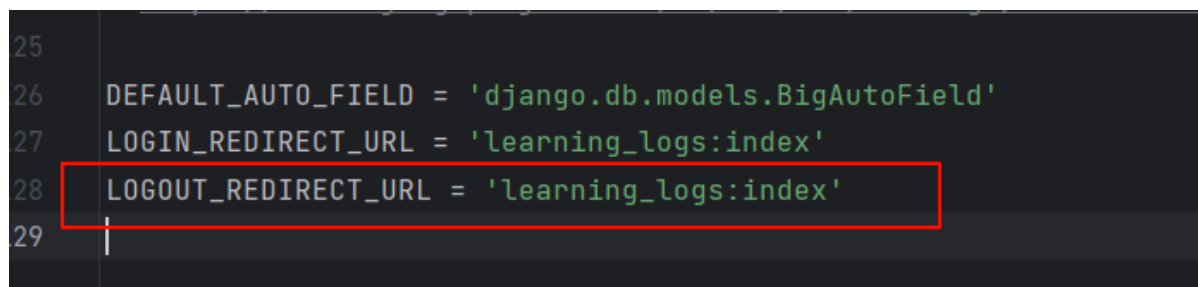
2、注销登录

在basehtml文件中添加注销



```
10 <body>
11 <p>
12 <a href="{% url 'learning_logs:index' %}">learning Log</a>
13 <a href="{% url 'learning_logs:topics' %}">topics</a>
14 {% if user.is_authenticated %}
15     Hello, {{ user.username }}.
16 {% else %}
17     <a href="{% url 'accounts:login' %}">Log in</a>
18 {% endif %}
19 </p>
20 {% block content %}{% endblock content %}
21 {% if user.is_authenticated %}
22 <hr>
23 <form action="{% url 'accounts:logout' %}" method="post">
24     {% csrf_token %}
25     <button name="submit">Log out</button>
26 </form>
27 {% endif %}
28 {#我们在最后一行插入了一对块标签这个块名为 content，是一个占位#}
29 {#符，其中包含的信息由于模板指定。#}
```

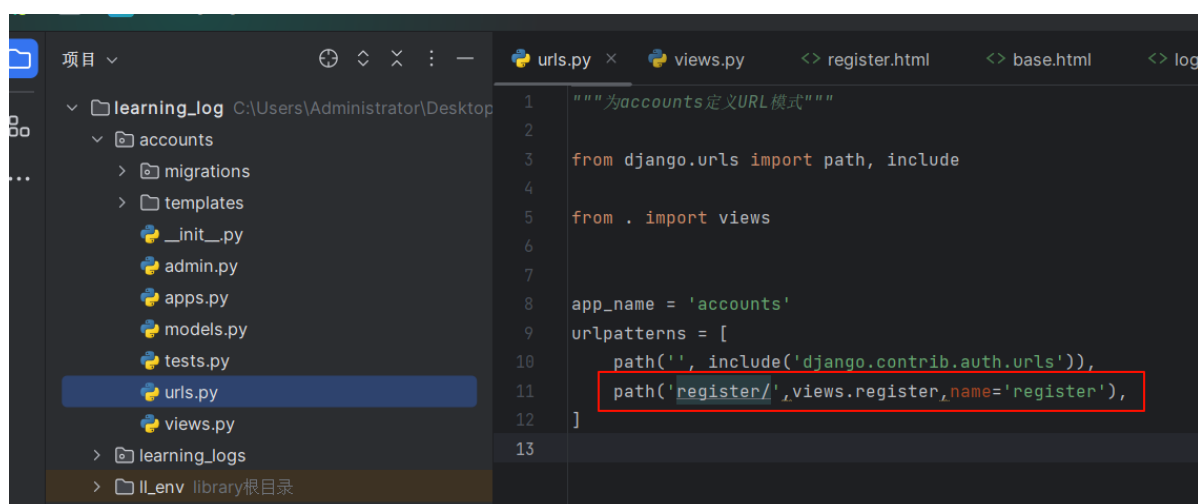
用户点击注销后，在settings告诉django重定向跳转到index页面



```
25
26 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
27 LOGIN_REDIRECT_URL = 'learning_logs:index'
28 LOGOUT_REDIRECT_URL = 'learning_logs:index'
29
```

3、注册页面

定义注册url



```
1 """为accounts定义URL模式"""
2
3 from django.urls import path, include
4
5 from . import views
6
7
8 app_name = 'accounts'
9 urlpatterns = [
10     path('', include('django.contrib.auth.urls')),
11     path('register/', views.register, name='register'),
12 ]
13
```

实现view函数

```
from django.shortcuts import render, redirect
from django.contrib.auth import login
from django.contrib.auth.forms import UserCreationForm

# Create your views here.
def register(request):
    """注册新用户"""
    if request.method != "POST":
        # 显示空的注册表单
        form = UserCreationForm()
    else:
        # 处理填写好的表单
        form = UserCreationForm(data=request.POST)
        if form.is_valid():
            new_user = form.save()
            # 让用户自动登录，再重定向到主页
            login(request, new_user)
            return redirect('learning_logs:index')

    # 显示空表单或指出表单无效
    context = {'form': form}
    return render(request, 'registration/register.html', context)
```

实现register html页面

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
</head>
<body>
{% extends 'learning_logs/base.html' %}
{% block content %}
    <form action="{% url 'accounts:register' %}" method="post">
        {% csrf_token %}
        {{ form.as_div }}
        <button name="submit">Register now</button>
    </form>
{% endblock content %}
</body>
</html>
```

在base.html页面添加register选项

```
urls.py  views.py  <> register.html  <> base.html  <> login.html

1      <!doctype html>
2      <html lang="en">
3      <head>
4          <meta charset="UTF-8">
5          <meta name="viewport"
6              content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1
7          <meta http-equiv="X-UA-Compatible" content="ie=edge">
8          <title>Base</title>
9      </head>
10     <body>
11         <p>
12             <a href="{% url 'learning_logs:index' %}">learning Log</a>
13             <a href="{% url 'learning_logs:topics' %}">topics</a>
14             {% if user.is_authenticated %}
15                 Hello, {{ user.username }}.
16             {% else %}
17                 <a href="{% url 'accounts:register' %}">Register now</a>
18                 <a href="{% url 'accounts:login' %}">Log in</a>
19             {% endif %}
20         </p>
21         {% block content %}{% endblock content %}
22         {% if user.is_authenticated %}
23             <hr>
24             <form action="{% url 'accounts:logout' %}" method="post">
25                 {% csrf_token %}
26                 <button name="submit">Log out</button>
27             </form>
28         {% endif %}
```

5、让用户拥有自己的数据

1、使用@login_required

在访问特定页面时，检查用户是否为登录状态

除了index函数外，其他函数都加上该装饰器

```
urls.py  settings.py  accounts\views.py  learning_logs\views.py  <> register.html  <> b

1  from django.contrib.auth.decorators import login_required
2  from django.shortcuts import render, redirect
3  from .forms import *
4
5  from .models import *
6
7
8  # Create your views here.
   1个用法
9  def index(request):
10     """学习笔记主页"""
11 <>     return render(request, template_name="learning_logs/index.html")
12
13
   1个用法
14 @login_required
15 def topics(request):
16     """学习笔记主题"""
17     topics = Topic.objects.order_by("date_added")
18     context = {"topics": topics}
19 <>     return render(request, template_name="learning_logs/topics.html", context)
20
21
22 @login_required
23 def topic(request, topic_id):
24     """显示单个主题及其所有条目"""
25     topic = Topic.objects.get(id=topic_id)
26     entries = topic.entry_set.order_by("-date_added") # - 表示降序
27     context = {"topic": topic, "entries": entries}
28 <>     return render(request, template_name="learning_logs/topic.html", context)
```

并且在settings文件中 配置当用户被此装饰器“拦截”后，系统将重定向到哪个页面

```
urls.py  settings.py  accounts\views.py  learning_logs\views.py  <> register.html  <> base.html

99     },
100 ]
101
102 # Internationalization
103 # https://docs.djangoproject.com/en/4.1/topics/i18n/
104
105 LANGUAGE_CODE = 'zh-Hans'
106 TIME_ZONE = 'Asia/Shanghai'
107
108 USE_I18N = True
109
110 USE_TZ = True
111
112 # Static files (CSS, JavaScript, Images)
113 # https://docs.djangoproject.com/en/4.1/howto/static-files/
114
115 STATIC_URL = 'static/'
116
117 # Default primary key field type
118 # https://docs.djangoproject.com/en/4.1/ref/settings/#default-auto-field
119
120 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
121 LOGIN_REDIRECT_URL = 'learning_logs:index'
122 LOGOUT_REDIRECT_URL = 'learning_logs:index'
123 LOGIN_URL = 'accounts:login'
124
```

2、将数据关联到用户

在models文件中添加两行代码

```
urls.py  settings.py  models.py  ×  accounts\views.py  learning_logs\views.py
1  from django.contrib.auth.models import User
2  from django.db import models
3
4
5  # Create your models here.
6
7  7 个用法
7  class Topic(models.Model):
8      """ 用户学习的主题 """
9      text = models.CharField(max_length=256)
10     date_added = models.DateTimeField(auto_now=True)
11     owner = models.ForeignKey(User, on_delete=models.CASCADE)
12
13     def __str__(self):
14         """ 返回模型的字符串显示 """
15         return self.text
16
```

执行数据库迁移: `py manage.py makemigrations`

`py manage.py migrate`

3、让用户只能看自己的主题

```
urls.py  settings.py  models.py  accounts\views.py  learning_logs\views.py  ×  <> register.html
4
5  from .models import *
6
7
8  # Create your views here.
9  1 个用法
9  def index(request):
10     """ 学习日志主页 """
11     <> return render(request, template_name="learning_logs/index.html")
12
13
14  1 个用法
14  @login_required
15  def topics(request):
16     """ 学习日志主题 """
17     topics = Topic.objects.filter(owner=request.user).order_by("date_added")
18     context = {"topics": topics}
19     <> return render(request, template_name="learning_logs/topics.html", context)
20
```


4、保护用户的主题

```
@login_required
def topic(request, topic_id):
    """显示单个主题及其所有条目"""
    topic = Topic.objects.get(id=topic_id)
    if topic.owner != request.user:
        raise Http404
    entries = topic.entry_set.order_by("-date_added") # - 表示降序
    context = {"topic": topic, "entries": entries}
    return render(request, "learning_logs/topic.html", context)
```

如果当前主题的所有者非登录用户，抛出404

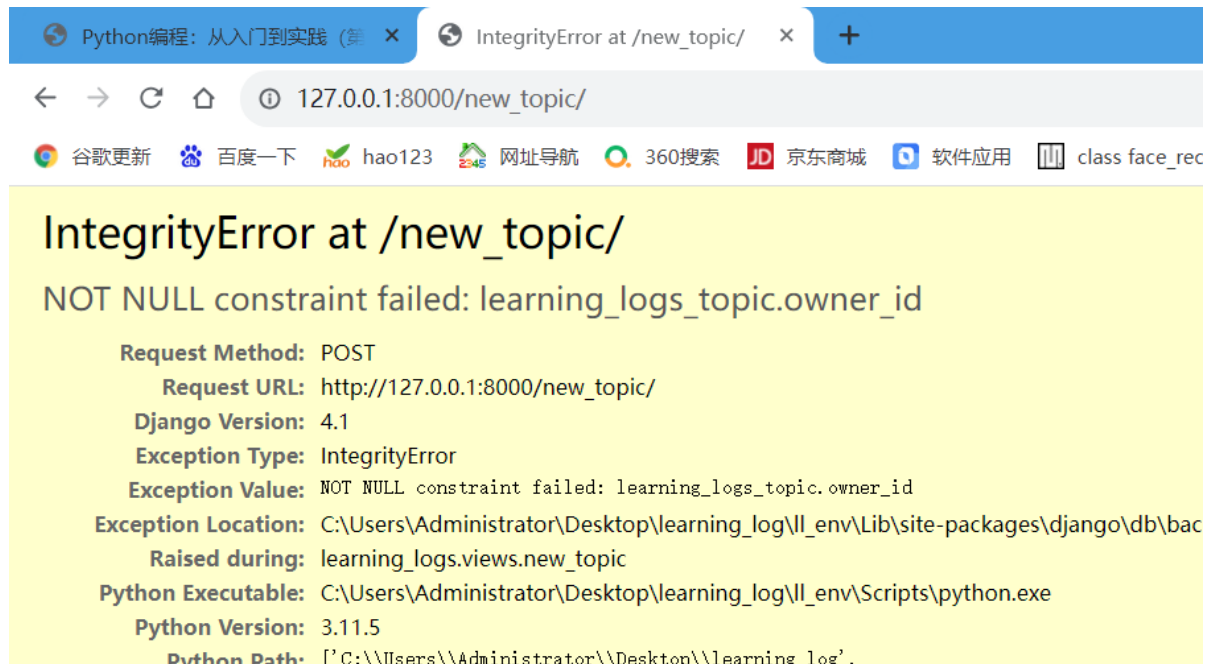
5、保护页面edit——entry

```
@login_required
def edit_entry(request, entry_id):
    """编辑现有的条目"""
    entry = Entry.objects.get(id=entry_id)
    topic = entry.topic
    if topic.owner != request.user:
        raise Http404
    if request.method != 'POST':
        # 初次请求：使用当前的条目填充表单
        form = EntryForm(instance=entry)
    else:
        # post提交的数据，对数据进行处理
        form = EntryForm(instance=entry, data=request.POST)
        if form.is_valid():
            form.save()
            return redirect(to='learning_logs:topic', topic_id=topic.id)

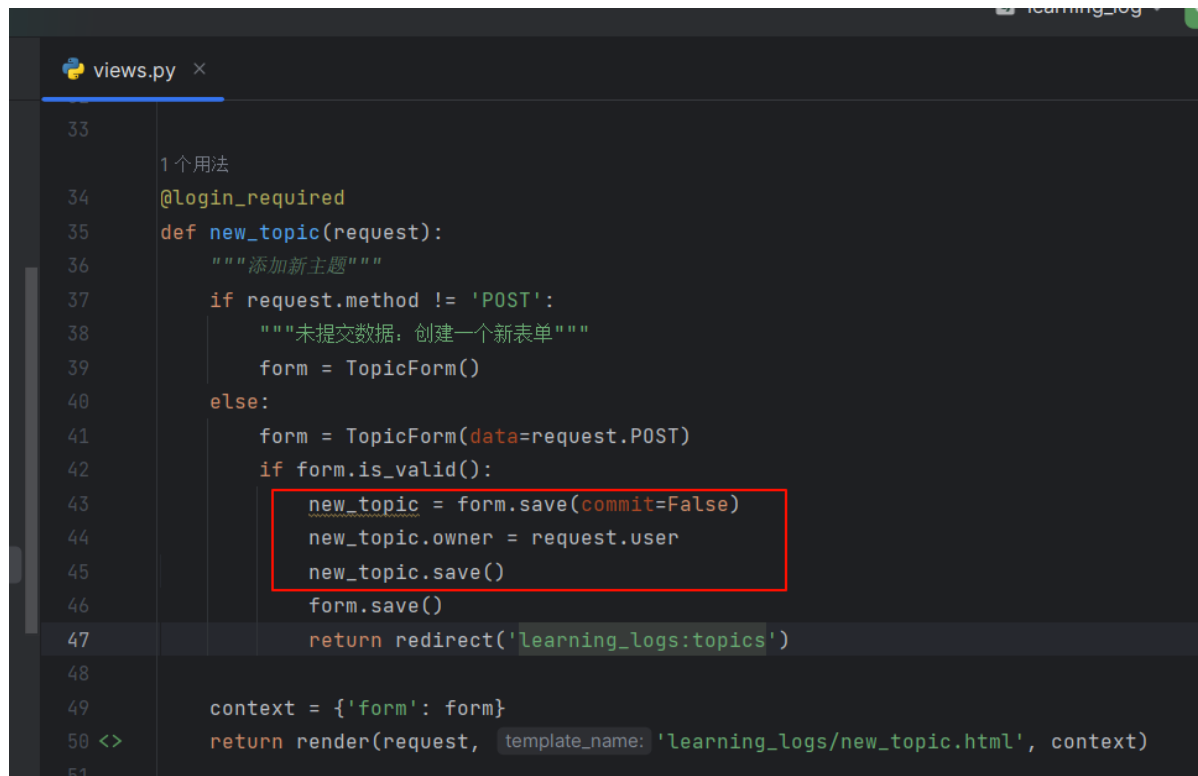
    context = {'entry': entry, 'topic': topic, 'form': form}
    return render(request, "learning_logs/edit_entry.html", context)
```

如果当前条目的所有者非登录用户，抛出404

解决添加新主题时的报错



添加如下代码

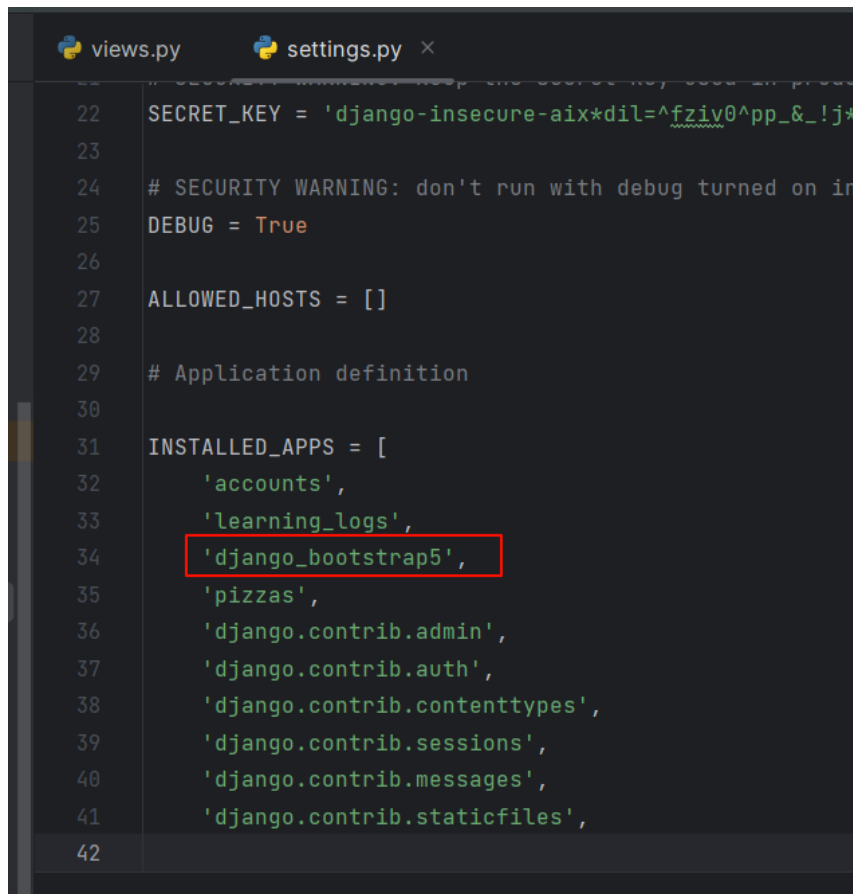


8、添加样式美化网站

1、安装bootstrap5

1、 pip install django-bootstrap5

2、在settings里添加代码



```
views.py settings.py x
22 SECRET_KEY = 'django-insecure-aix*dil=^fziv0^pp_&!j*'
23
24 # SECURITY WARNING: don't run with debug turned on in
25 DEBUG = True
26
27 ALLOWED_HOSTS = []
28
29 # Application definition
30
31 INSTALLED_APPS = [
32     'accounts',
33     'learning_logs',
34     'django_bootstrap5',
35     'pizzas',
36     'django.contrib.admin',
37     'django.contrib.auth',
38     'django.contrib.contenttypes',
39     'django.contrib.sessions',
40     'django.contrib.messages',
41     'django.contrib.staticfiles',
42
```

2、修改basehtml文件

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Base</title>
    {% load django_bootstrap5 %}
    {% bootstrap_css %}
    {% bootstrap_javascript %}
</head>
<body>
<nav class="navbar navbar-expand-md navbar-light bg-light mb-4 border">
    <div class="container-fluid">
        <a class="navbar-brand" href="{% url 'learning_logs:index' %}">
            学习日志</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
            data-bs-target="#navbarCollapse" aria-controls="navbarCollapse"
            aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarCollapse">
            <ul class="navbar-nav me-auto mb-2 mb-md-0">
                <li class="nav-item">
                    <a class="nav-link" href="{% url 'learning_logs:topics' %}">
                        主题</a>
```

```

    </li>
    </ul> <!-- 定义导航栏左侧链接的代码到此结束 -->
    <ul class="navbar-nav ms-auto mb-2 mb-md-0">
    {% if user.is_authenticated %}
        <li>
            <span class="navbar-text me-2">你好啊,{{ user.username }}!
</span>

        </li>
        {% else %}
            <li class="nav-item"><a class="nav-link" href="{% url
'accounts:register' %}">注册账号</a></li>
            <li class="nav-item"><a class="nav-link" href="{% url
'accounts:login' %}">登录</a></li>
        {% endif %}
    </ul><!--与账号相关的链接到此结束-->
    {% if user.is_authenticated %}
        <form action="{% url 'accounts:logout' %}" method="post">
            {% csrf_token %}
            <button name="submit" class="btn btn-outline-secondary btn-
sm">退出当前账号</button>
        </form>
    {% endif %}
</div> <!-- 定义导航栏可折叠部分的代码到此结束 -->
</div> <!-- 定义导航栏容器的代码到此结束 -->
</nav> <!-- 定义导航栏的代码到此结束 -->

<main class="container">
    <div class="pb-2 mb-2 border-bottom">
        {% block page_header %}{% endblock page_header %}
    </div>
    <div>
        {% block content %}{% endblock content %}
    </div>
</main>

    {#我们在最后一行插入了一对块标签这个块名为 content，是一个占位#}
    {#符，其中包含的信息由子模板指定。#}
    {#子模板并非必须定义父模板中的每个块，因此在父模板中，可以使用任意多个块#}
    {#来预留空间，而子模板可根据需要定义相应数量的块。#}

</body>
</html>

```

3、设置index文件

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>芝士首页</title>
</head>
<body>
    {% extends 'learning_logs/base.html' %}
    {% block page_header %}
        <div class="p-3 mb-4 bg-light border rounded-3">

```

```

        <div class="container-fluid py-4">
            <h1 class="display-3">记录你的学习成果</h1>
            <p class="lead">Make your own Learning Log, and keep a list of
the topics you're learning about.
                whenever you learn something new about a topic, make an entry
summarizing what you've learned.
            </p>
            <p class="lead">制作你自己的学习日志，并把你正在学习的主题列在一张清单上。
每当你对一个话题有了新的了解，就做一个总结。
            </p>
            <a class="btn btn-primary btn-lg mt-1" href="{% url
'accounts:register' %}">点我注册 &raquo;</a>
        </div>
    </div>
{% endblock page_header %}

</body>
</html>

```

4、设置login文件

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>登录</title>
</head>
<body>
{% extends 'learning_logs/base.html' %}
{% load django_bootstrap5 %}

{% block page_header %}
    <h2>登录您的账号.</h2>
{% endblock page_header %}
{% block content %}
    {% if form.errors %}
        <p>您的用户名和密码不匹配。请再试一次。</p>
    {% endif %}
    <form action="{% url 'accounts:login' %}" method="post">
        {% csrf_token %}
        {% bootstrap_form form %}
        {% bootstrap_button button_type='submit' content='Log in'%}
    </form>
{% endblock content %}
</body>
</html>

```

5、修改topics文件

```

<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">

```

```

<meta name="viewport"
      content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>主题</title>
</head>
<body>
{% extends 'learning_logs/base.html' %}
{% block page_header %}
    <h1>你的主题</h1>
{% endblock page_header %}
{% block content %}
    <ul class="list-group border-bottom pb-2 mb-4">
        {% for topic in topics %}
            <li class="list-group-item border-0"><a href="{% url
'learning_logs:topic' topic.id%}">
                {{ topic.text }}</a>
            </li>
            {% empty %}
            <li class="list-group-item border-0">当前您还没有添加任何主题。</li>
        {% endfor %}
    </ul>
    <a href="{% url 'learning_logs:new_topic' %}">点击添加新主题</a>
{% endblock content %}

</body>
</html>

```

6、修改topic文件

```

<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>主题详情</title>
</head>
<body>
{% extends 'learning_logs/base.html' %}
{% block page_header %}
    <h1>{{ topic.text }}</h1>
{% endblock page_header %}
{% block content%}
    <p>
        <a href="{% url 'learning_logs:new_entry' topic.id %}">添加新条目</a>
    </p>
    {% for entry in entries %}
        <div class="card mb-3">
            <h4 class="card-header">
                {{ entry.date_added|date:'M d, Y H:i' }}
            <small>
                <a href="{% url 'learning_logs:edit_entry' entry.id%}">

```

```

                                编辑条目
                                </a>
                                </small>
                                </h4>
                                <div class="card-body">
                                    {{ entry.text|linebreaks }}
                                </div>
                                </div>
                                {% empty %}
                                <li>当前主题还没有条目.</li>
                                {% endfor %}
                                {% endblock content%}
                            </body>
                        </html>

```

9、效果展示

