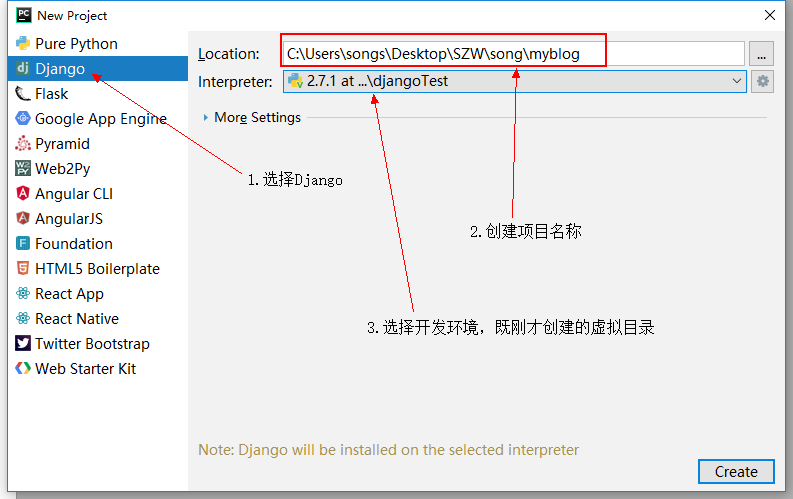
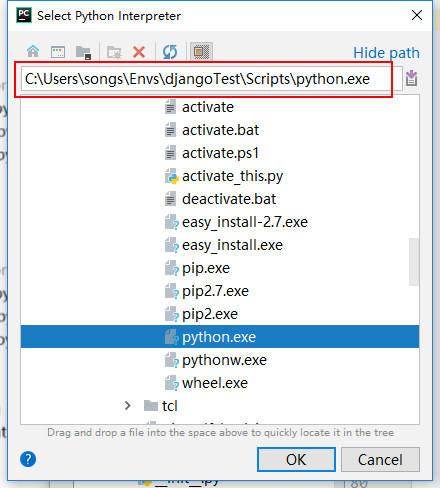
一 安装配置

1. 软件安装
   1. Pycharm
   2. Mysql
   3. Navicat
   4. Python2.7
2. Virtualenv wapper虚拟环境
   1. 使不同应用开发环境独立
   2. 环境升级不影响其他应用，也不影响全局的python环境
   3. 防止系统中出现的包管理混乱和版本的冲突
   4. 安装方法
      1. pip install virtualenvwrapper-win
      2. mkvirtualenv test 🡺创建虚拟环境目录
      3. workon 🡺查看当前存在的虚拟环境
      4. workon test 🡺进入test目录
      5. deactivate 🡺退出虚拟环境
      6. pip lsit 🡺查看开发包
3. 安装好虚拟环境以后就要创建项目了，在pycharm中新建项目，然后在interpreter选项中选择刚才创建的虚拟环境目录下的python.exe





虚拟目录放在C盘下面

user🡺admin（我的电脑改名为songs）🡺Envs🡺虚拟目录(自己创建的)🡺Scripts🡺python.exe

因为上面提示没有django环境，所以我们在cmd中先安装django，然后再创建

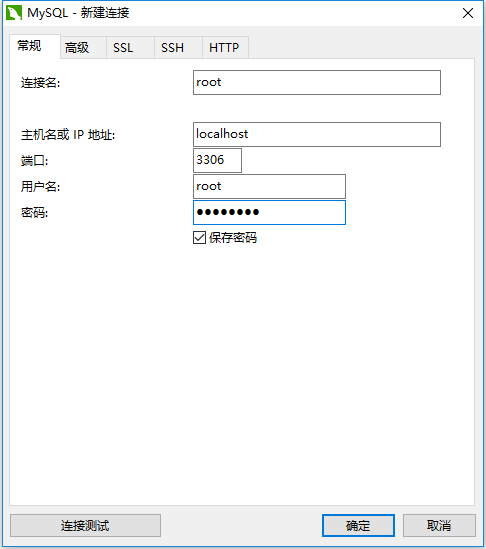
pip install django==1.9.8

这里我们用的版本是1.9.8

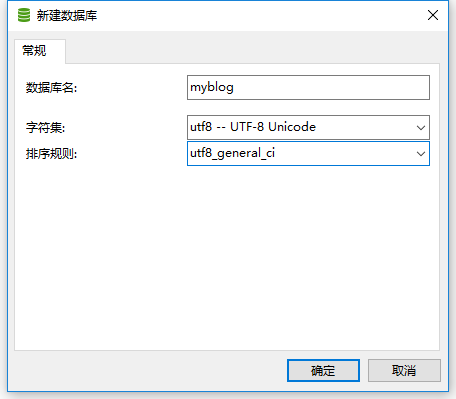
4. 在pycharm中点击run，就可以访问程序了

二 建立数据库

初次使用时，先连接数据库



连接数据库以后新建数据库，在这里要选择utf8的字符集，否则中文可能会有问题



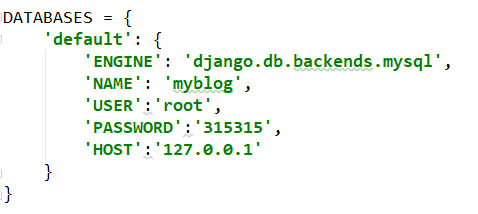
安装mysql驱动

pip install mysql-python

在安装完成以后会报一个错误，需要安装一个包，附教程链接

<http://blog.csdn.net/shaququ/article/details/54292043>

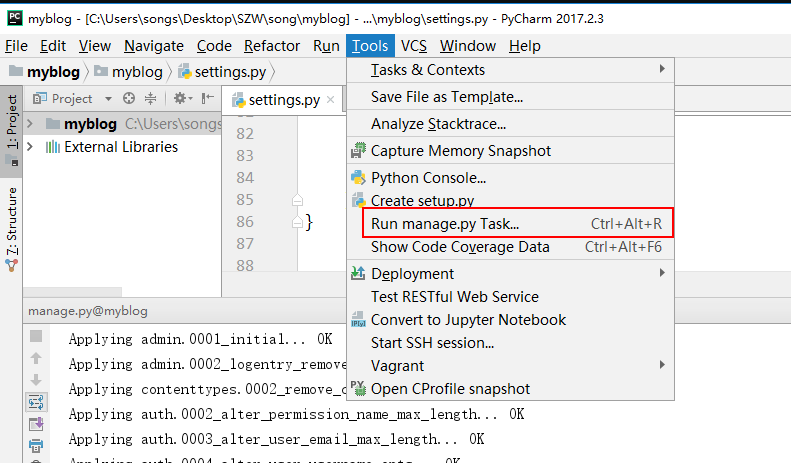
在settings配置中配置数据库

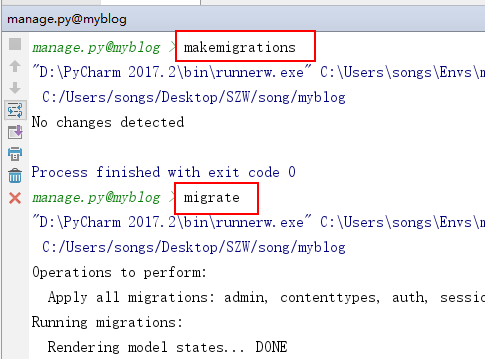


**'ENGINE'，'NAME'，'USER'，'PASSWORD'，'HOST'**都是固定写法，不可以改变

DATABASES = {  
 **'default'**: {  
 **'ENGINE'**: **'django.db.backends.mysql'**,  
 **'NAME'**: **'myblog'**,  
 **'USER'**:**'root'**,  
 **'PASSWORD'**:**'315315'**,  
 **'HOST'**:**'127.0.0.1'** }  
}

在配置完成以后，下一步就是生成数据表，在pycharm中，点击Tools，然后选择Run manage.py Task,执行makemigrations和migrate两条指令，然后刷新数据库，就可以看到django自动生成的数据表如下图所示





值得注意的是，这是在pycharme中的方法，如果不用该方法，则需要命令行输入，方法自行百度，这里不在描述。

三 新建users app，

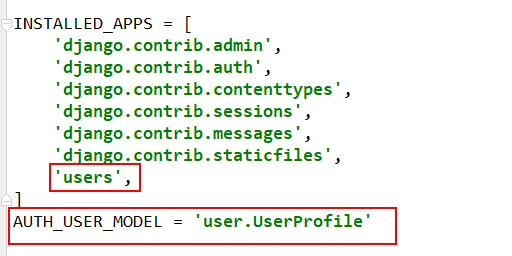
在task下输入startapp users生成users

在所有web网站中，user表应该是第一个被设计的表格

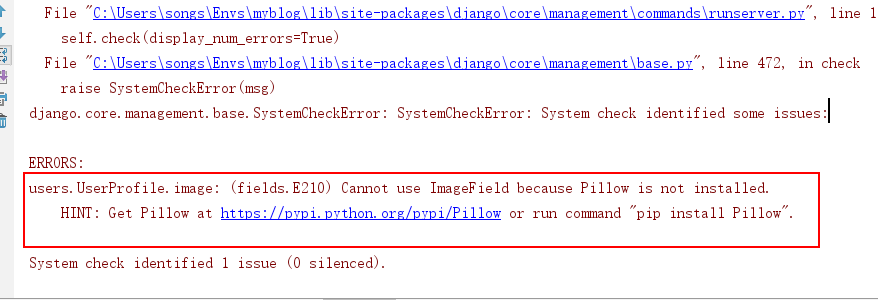
继承django自带的user表并添加自己需要的新的字段

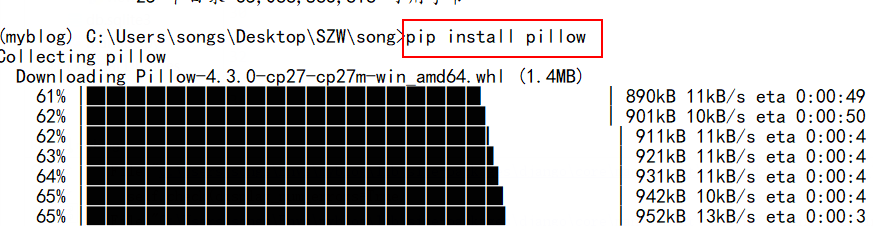


写完users表以后要将app注册进去，在installed\_apps中注册users并重载user的方法

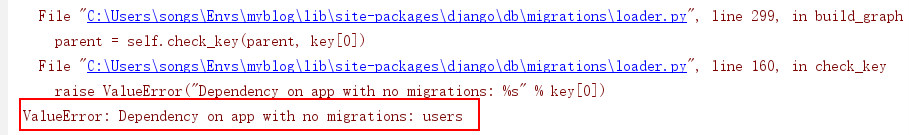


这个时候如果我们启动项目，会发现报user.UserProfile.image的错误，错误代码如下，那是因为我们没有安装Pillow插件，此时返回cmd然后在命令行输入pip install pillow下载





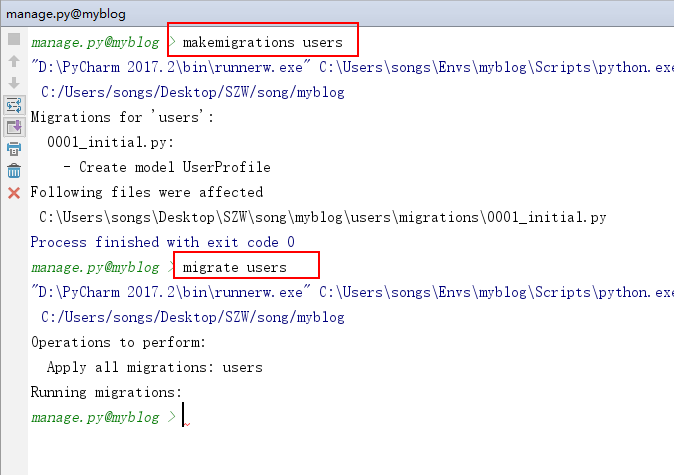
当我们再次重启以后，发现又有新的错误，代码如下：



这个意思是我们这个users的app没有migrations，解决办法，打开task，

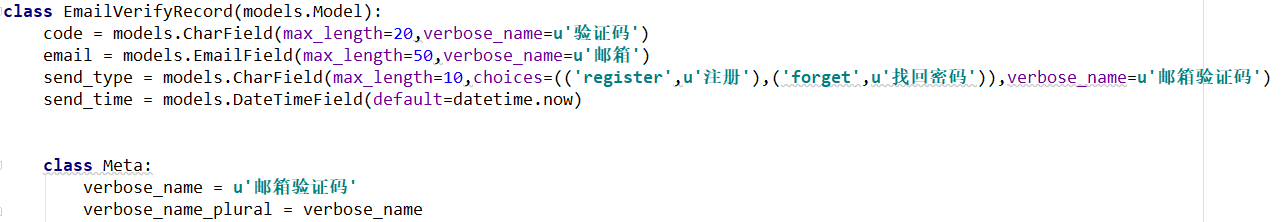
后输入makemigrations users

输入migrate users

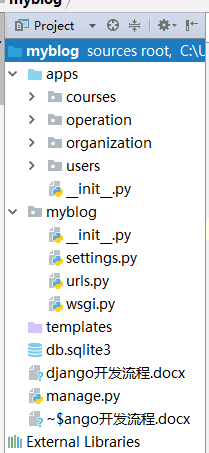


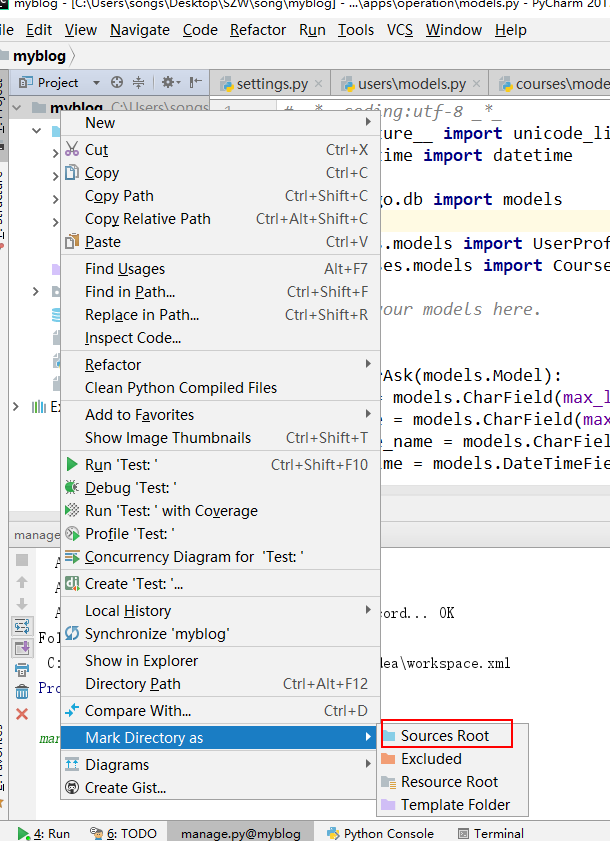
然后输入yes，就生成数据表了，此时在重新运行，就可以正常访问了

邮箱字段添加



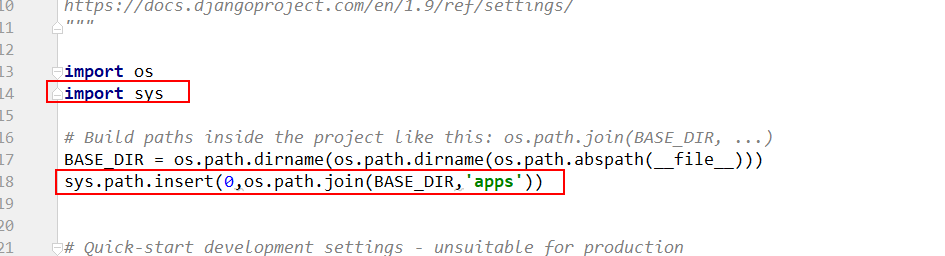
当把应用都统一放到apps目录以后，要将apps目录mark标记为sources root，否则编辑器会报错，但是，把它标记完以后编辑器不报错，python会报错，此时要在settings中将apps手动添加到python的搜索目录之下





在settings中添加apps

1. 引入sys模块
2. 添加apps路径



后台管理系统xadmin

特点：权限管理、少前端样式、快速开发

Django在创建后自动生成admin后台面板，需要我们自己创建用户名密码，方法如下：

1.在task下输入命令creatsuperuser

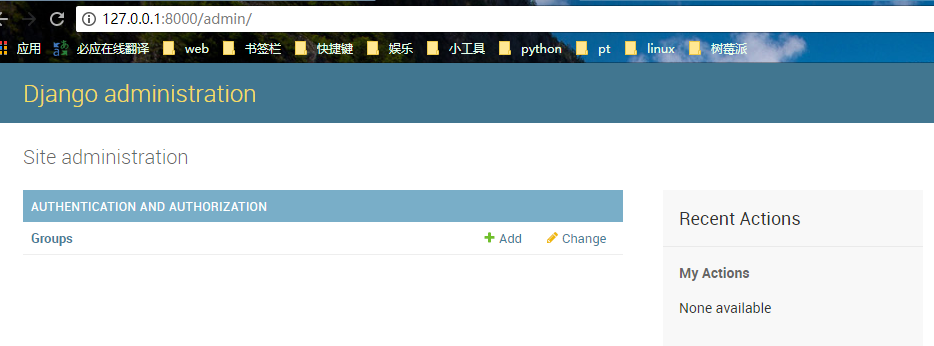
2.输入用户名，邮箱，密码

问题1：

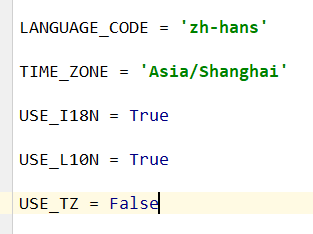
当输入用户名密码后报错，该问题是因为gender字段female有6个字节，我只给了5个，所以需要将maxlength改为6或者更大，然后重新makemigration users和migrate users之后再重新创建超级用户



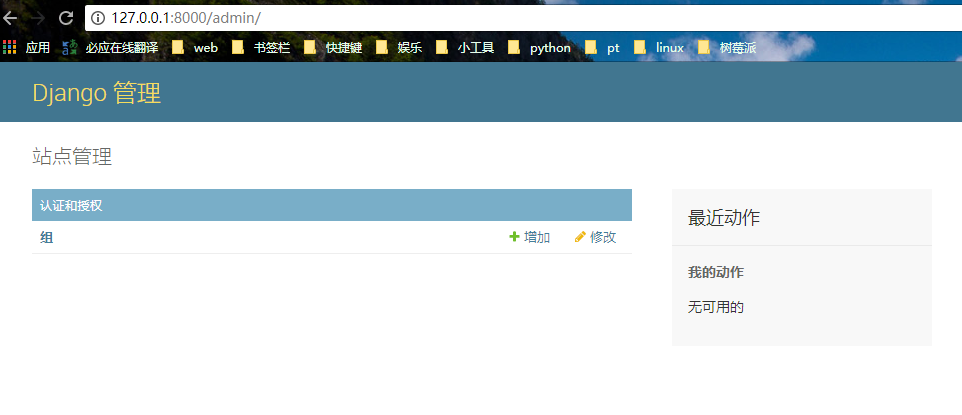
登陆面板后就可以看到后台面板了



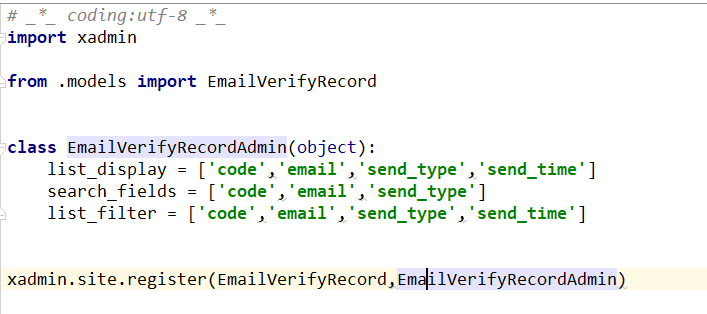
在settings中，设置language-code = “zh-hans”就可以变为中文



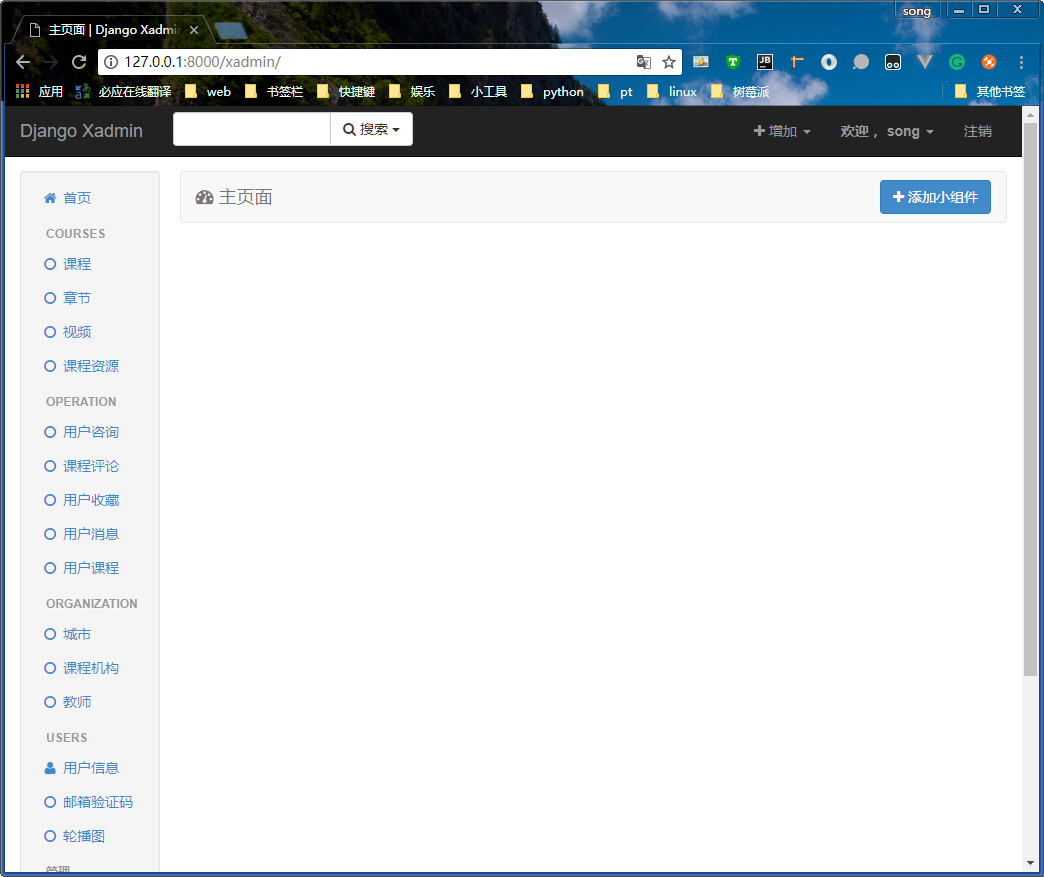
效果如下



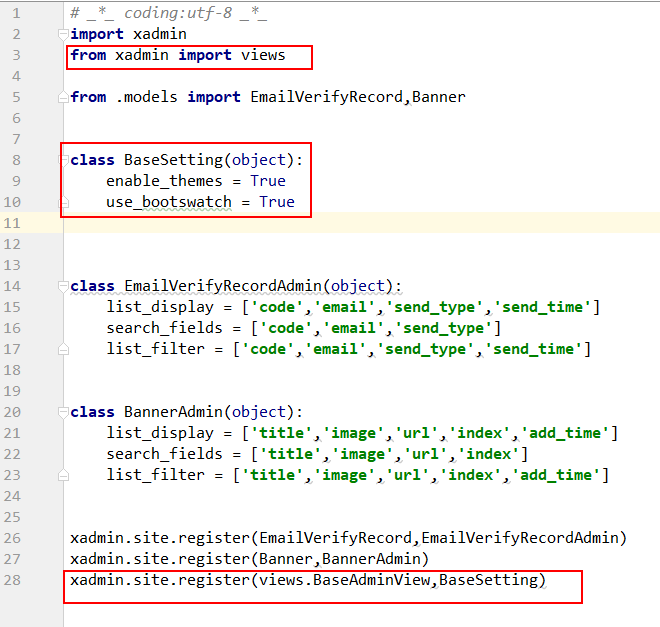
注册app到后台面板



修改主题功能



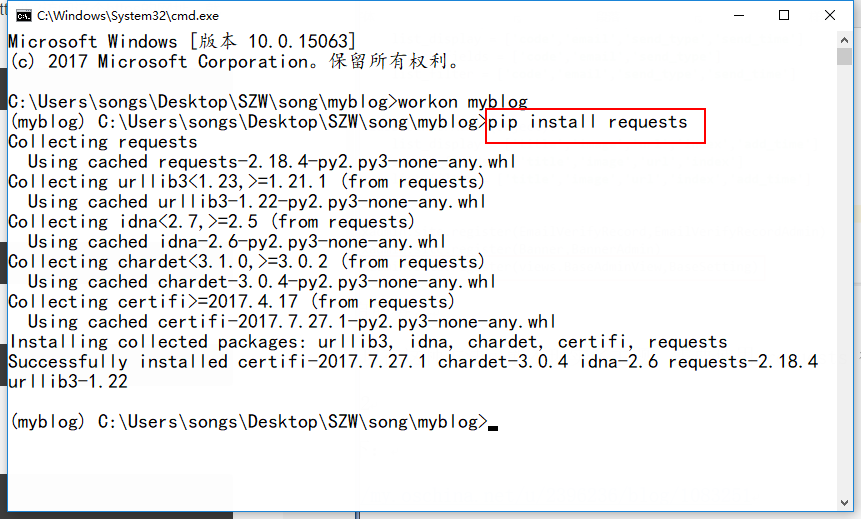
在user的adminx页面中导入xadmin下的views模块，创建BaseSetting并开启主题



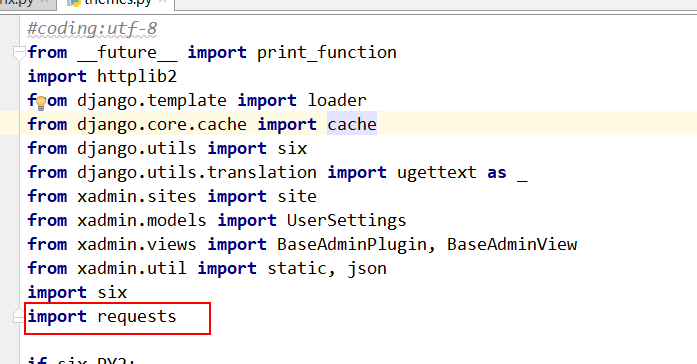
然后重启服务器

注意，，现在这种方法可能不现实多主题，需要使用requests替代httplib2

1. 安装requests



2.在/xadmin/plugins/themes.py 引入requests

3.修改block\_top\_navmenu方法



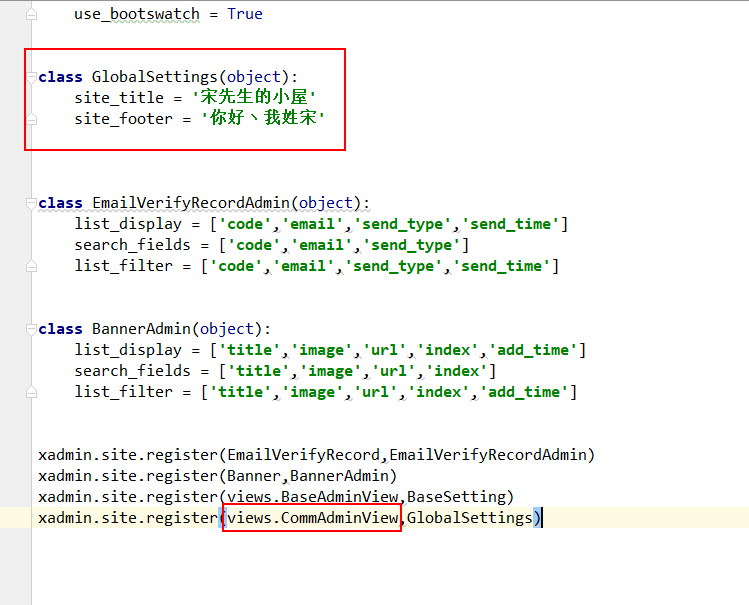
链接如下：

<https://my.oschina.net/u/2396236/blog/1083251>

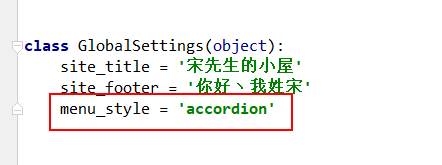
至此，后台主题功能就可以正常使用了

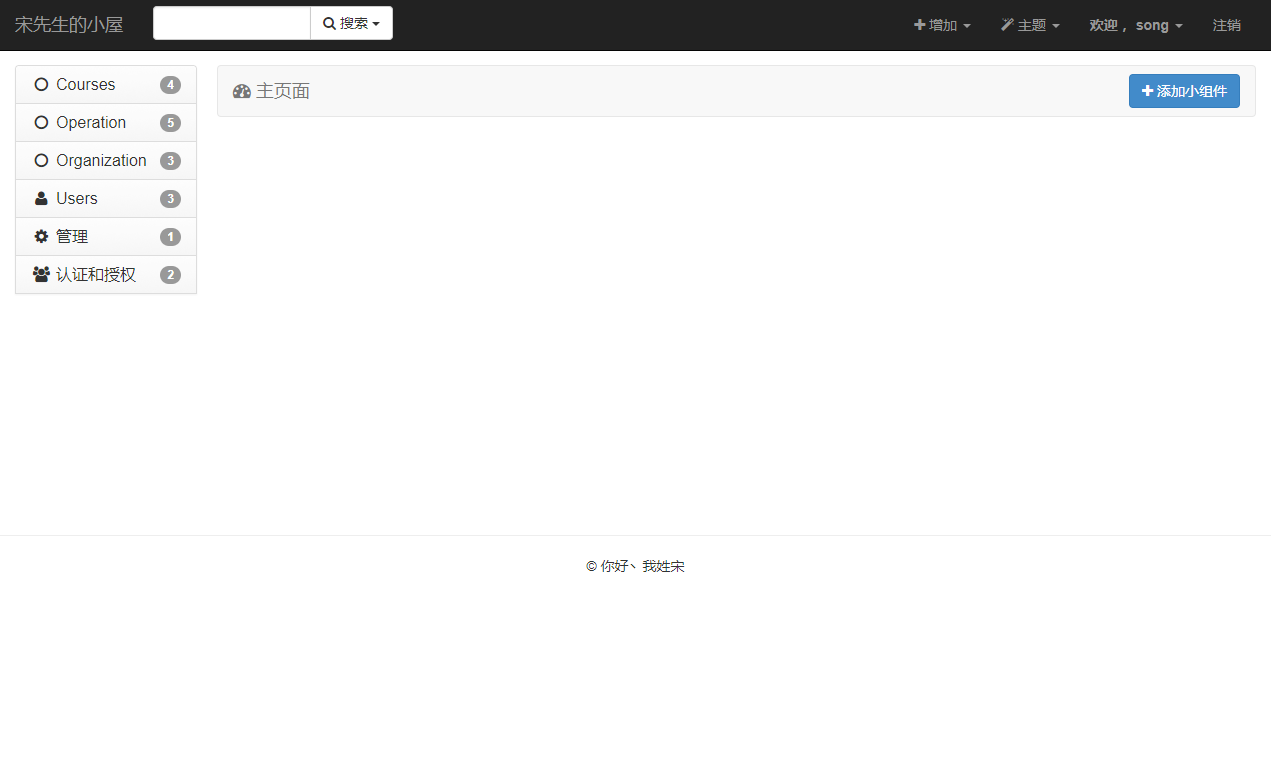
设置后台公共部分

修改左上角和底部内容



修改左侧导航栏样式

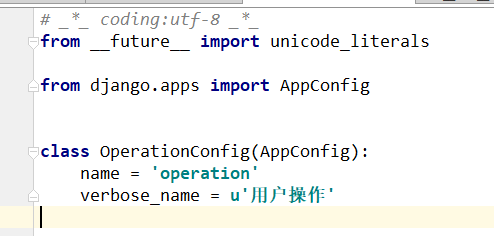




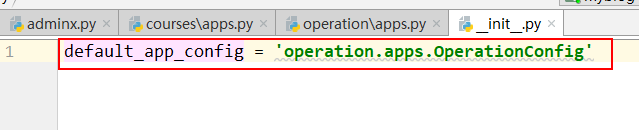
app设置中文

1.在app目录下找到apps.py，输入verbose\_name = u’要写的名字

’记得加utf8编码格式



2.在当前\_\_init\_\_.py文件夹下配置default\_app\_config

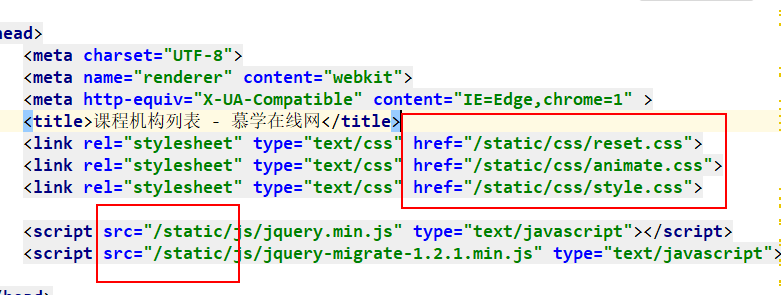


添加前端展示页面

1. 把html文件放到templates文件夹中，把css，js，img等文件放到static文件夹中，然后在settings中配置static路径
2. 在urls中配置要展示的页面路径，

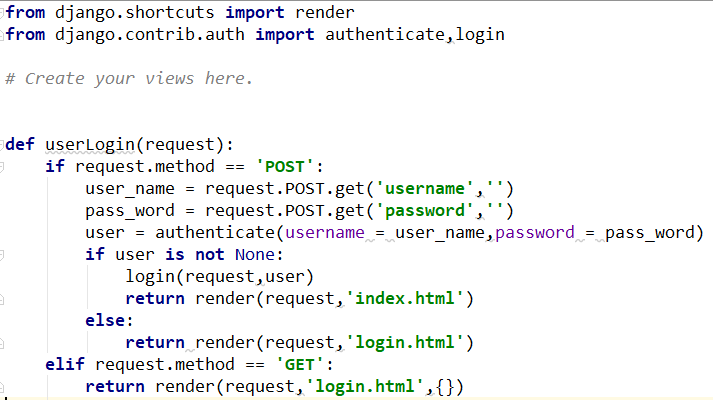


1. 在静态页面中修改路径，将css，js以及路径引用的src都换成static模板可用的



验证登陆

1.导入模块 from Django.contrib.auth import authenticate, login



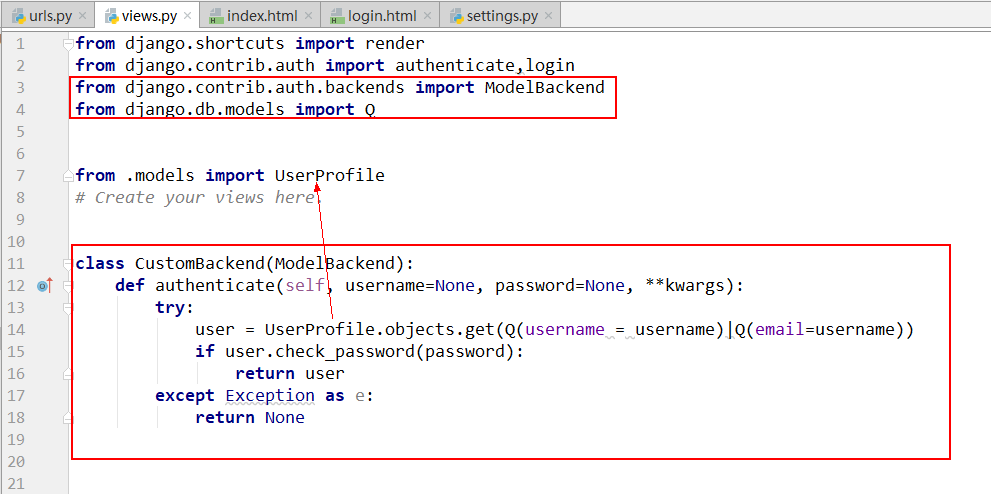
2.user = authenticate(username,password)

通过authenticate验证用户名密码是否正确，如果正确就把结果赋值给user，如果不正确就返回None，所以当我们判断如果user不等于None时候，进行登陆操作

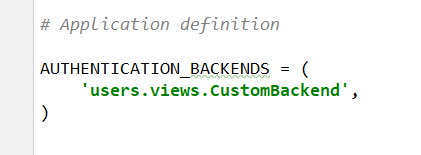
1. authenticate方法只想数据库发起验证请求看账号密码是否正确，实际登陆就需要调用login方法，login方法接收两个参数，一个request一个user

通过email和用户名都可以登陆设置

1，在views页面引入ModelBackend，定义authenticate



1. 在settings中添加设置



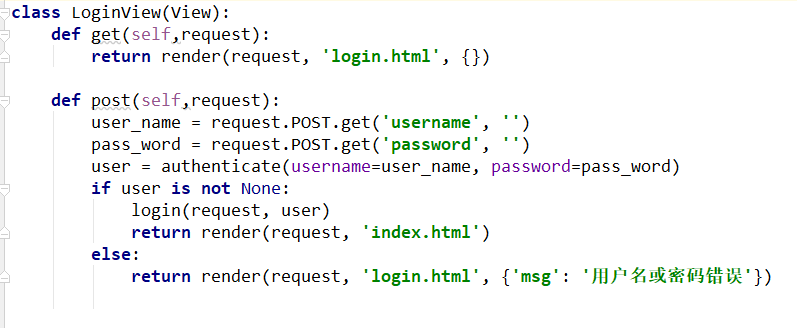
就可以使用邮箱和用户名登陆了

函数方法基于类的书写

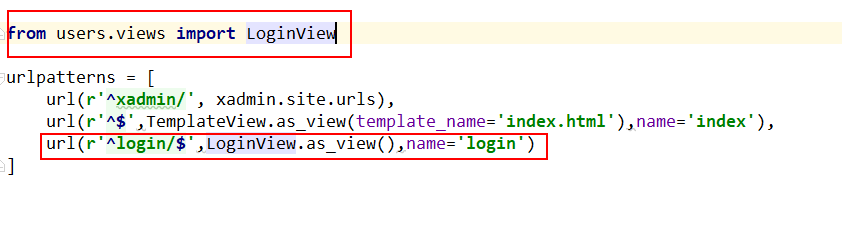
1. 引入View模块



1. 定义类

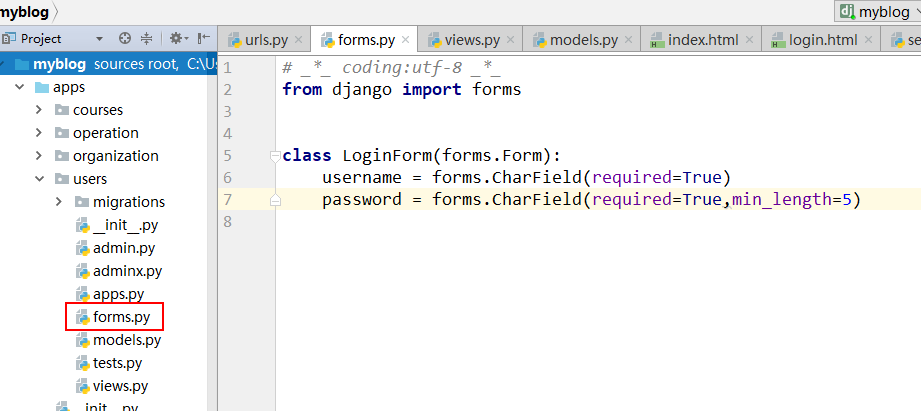


1. 在url中引入这个类，并注册到url中

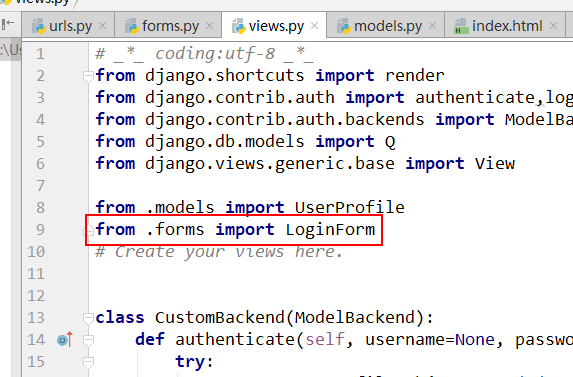


通过forms验证信息

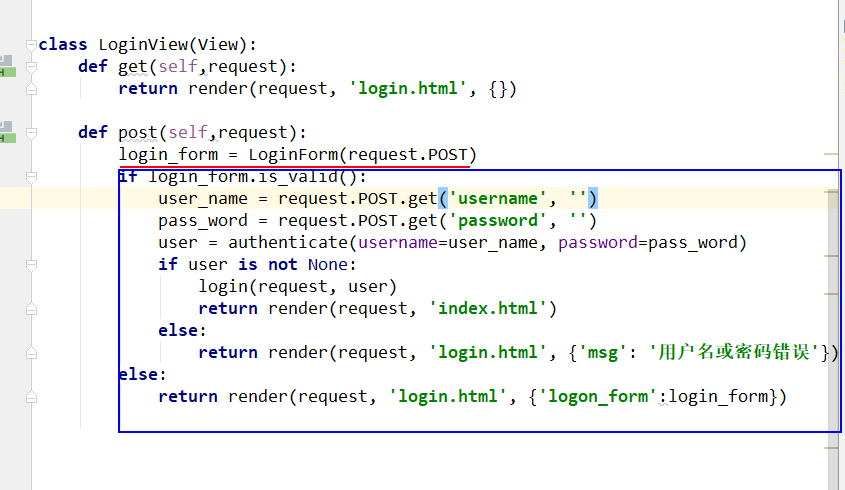
1.新建forms，创建要验证的类



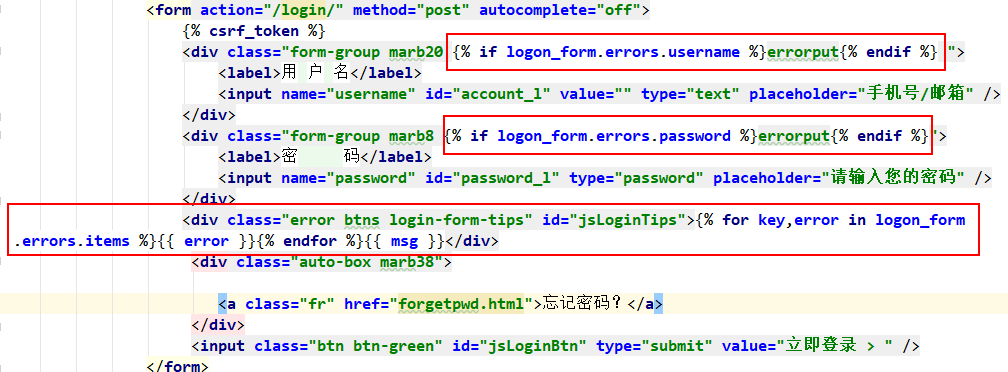
2.在views中导入forms中创建的类



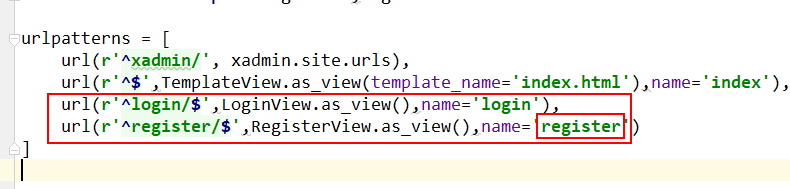
3.验证信息

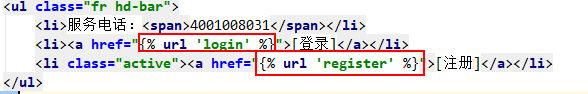


4.前端显示



url路径配置方法



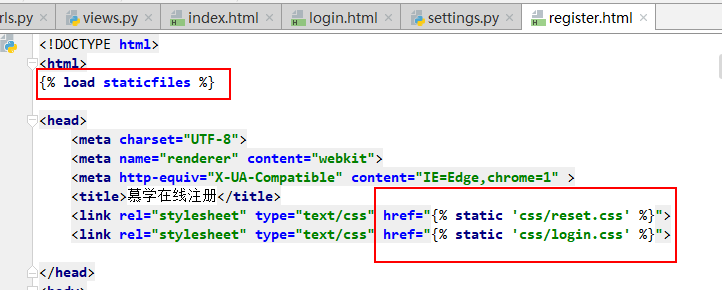


静态文件地址配置方法

1.在html页面load staticfiles

{% **load staticfiles** %}

2.将要引用的静态文件路径改为{% **static 'css/login.css'** %}

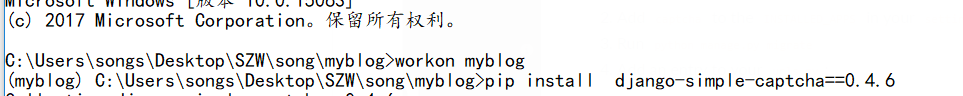


验证码：

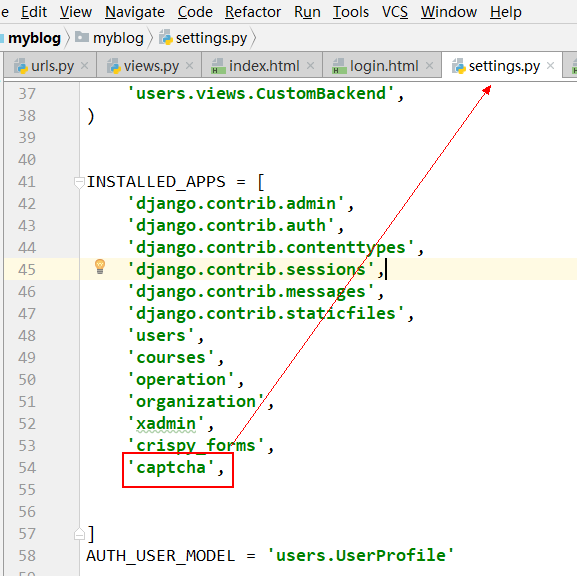
链接如下：

http://django-simple-captcha.readthedocs.io/en/latest/usage.html#installation

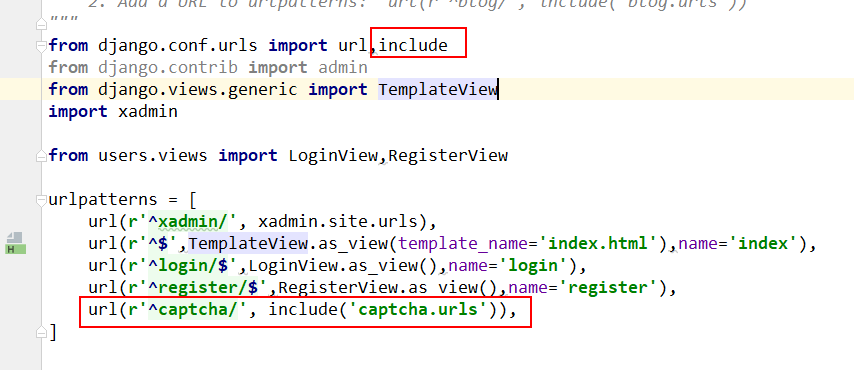
1．在虚拟环境中安装captcha



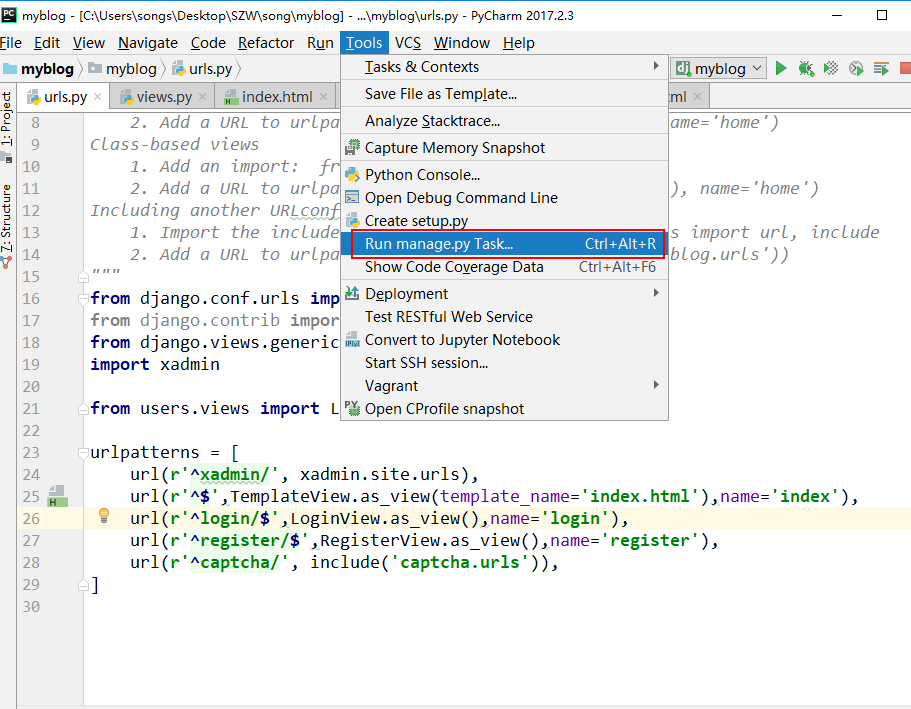
2.在installed\_apps中添加captcha

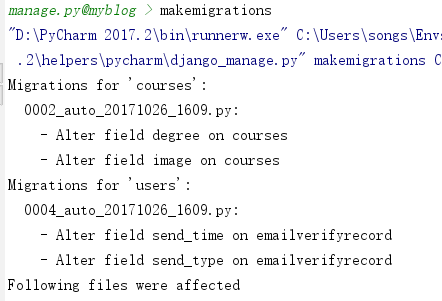


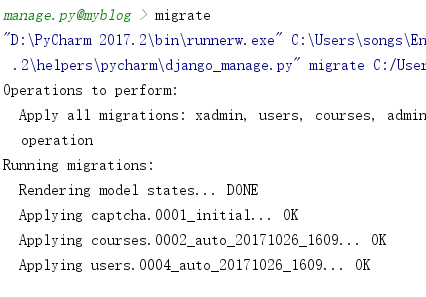
3. 在urls中添加图片路径



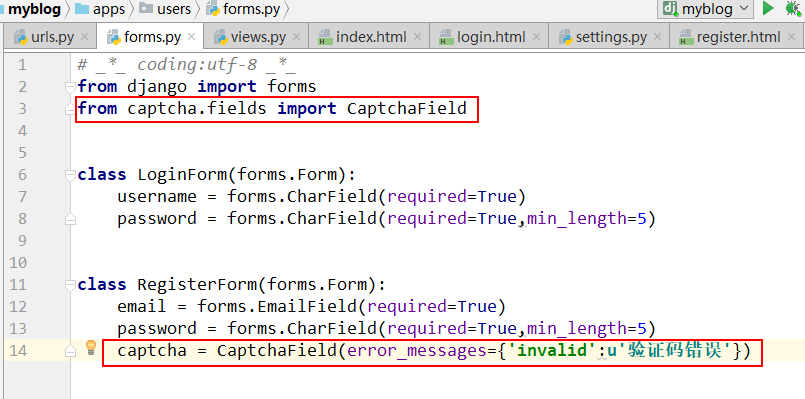
4．运行migration，migrate



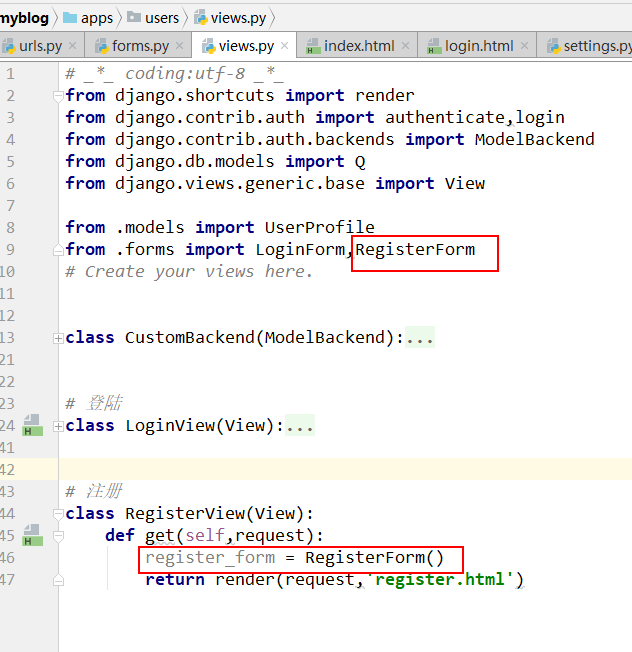




1. 在forms中引入captchafield



1. 在views中引入RegisterForm



1. 然后在需要验证码的地方添加{{ **register\_form**.**captcha** }}

**register\_form为自己写的接受变量名**

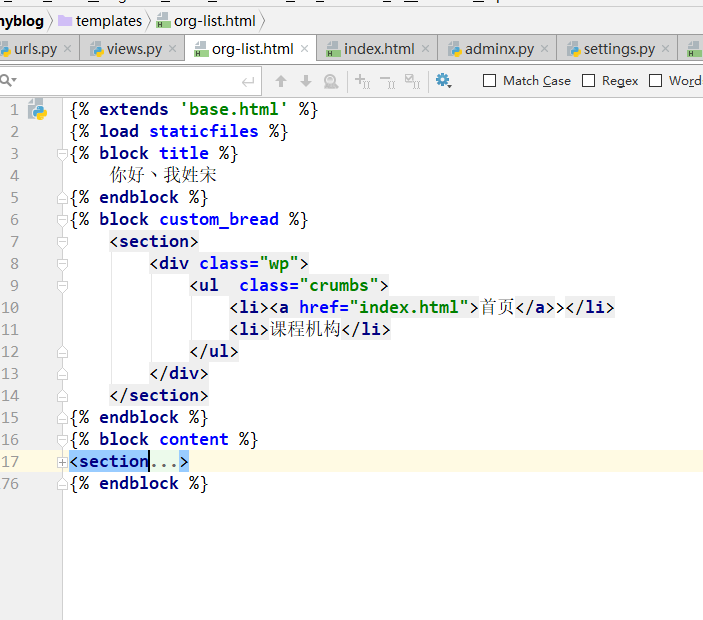


Template静态模板

1. 在template中创建一个base.html文件

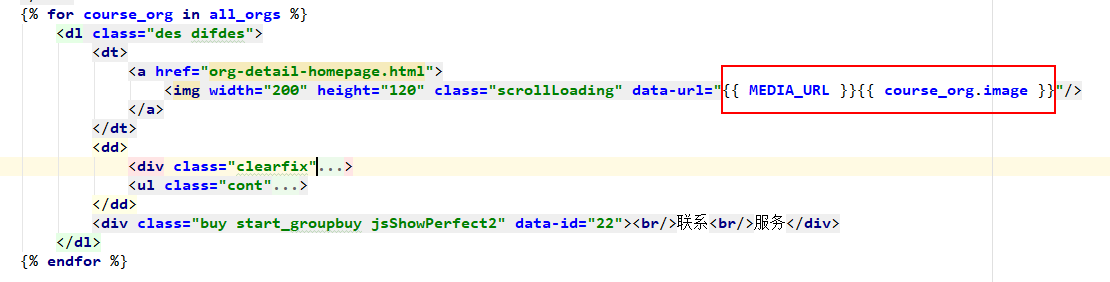


1. 在页面中调用需要修改的block内容

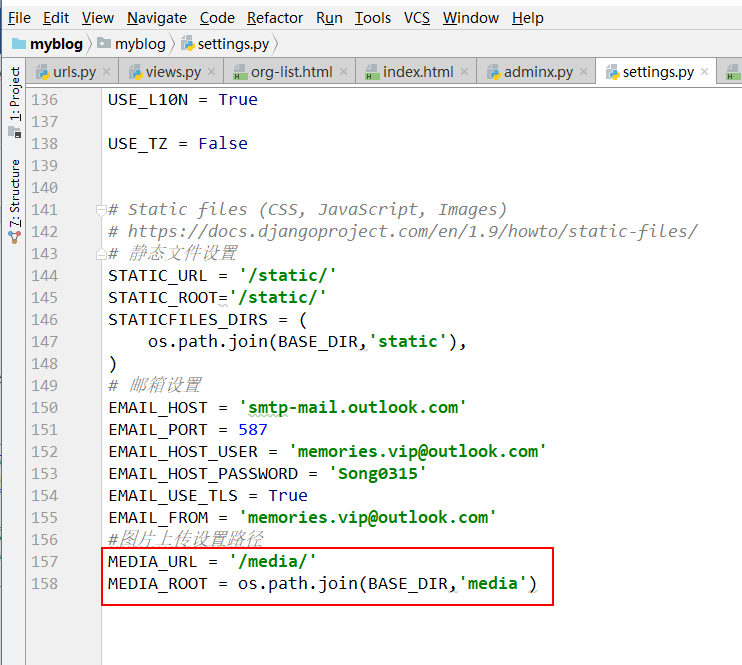


解决后台图片路径问题

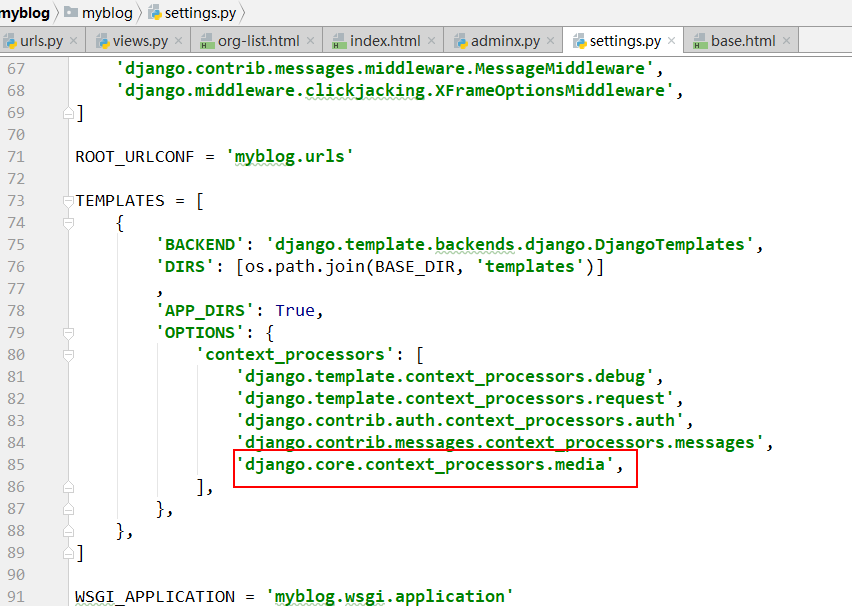
1.在html页面中填写url路径



2.在settings中添加图片地址



1. 在settings中的TEMPLATES中添加media

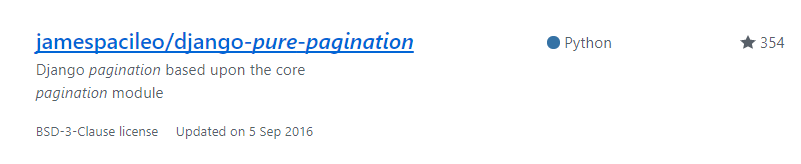


1. 在urls中导入server和MEDIA\_ROOT，并添加路径



分页插件

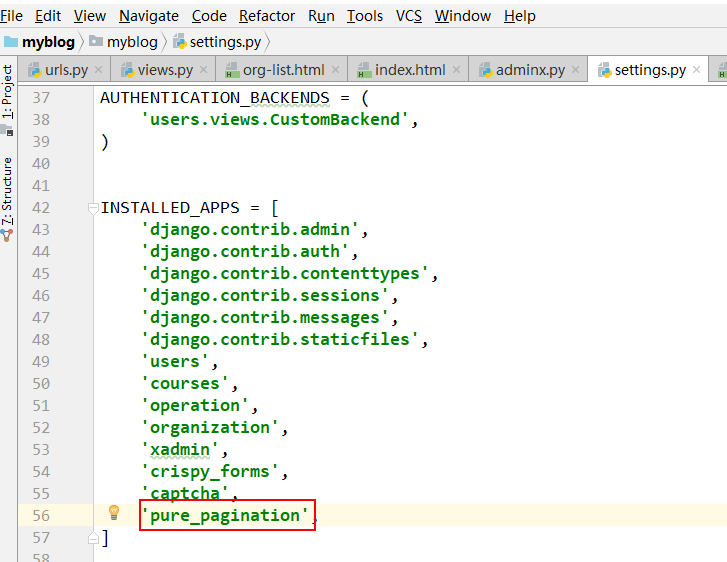
1. 在github中搜索pure pagination,这是django的一个专门用于分页的开发



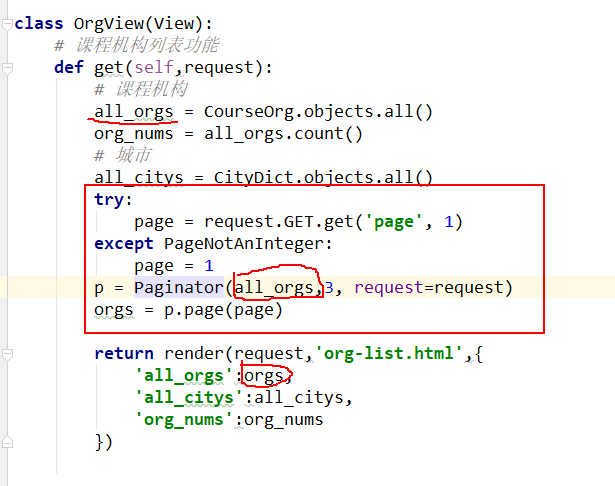
2.进入虚拟环境，安装开发包

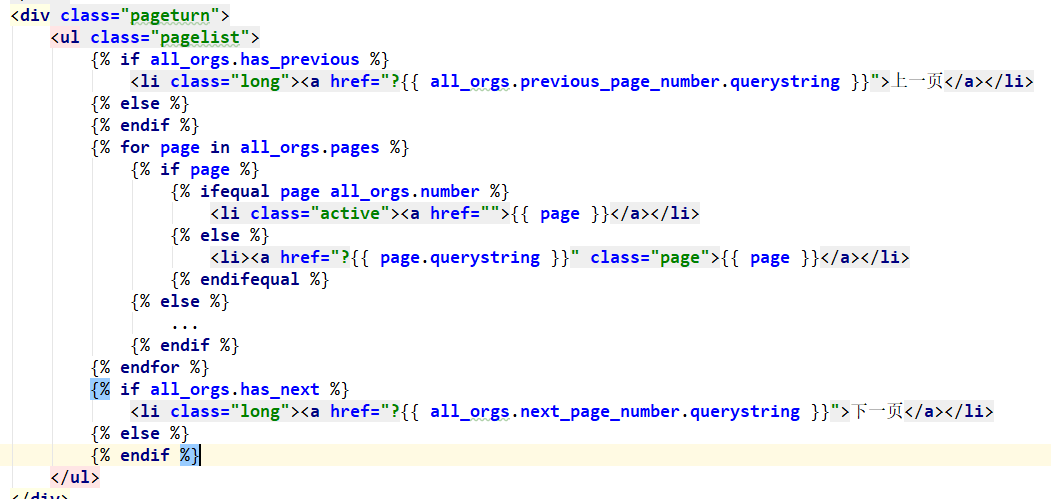
pip install django-pure-pagination

3.在settings中注册apps



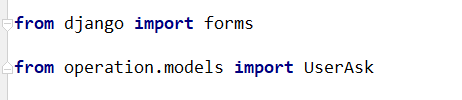
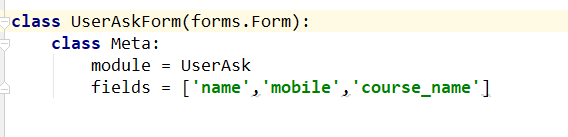
4.把模板套进去





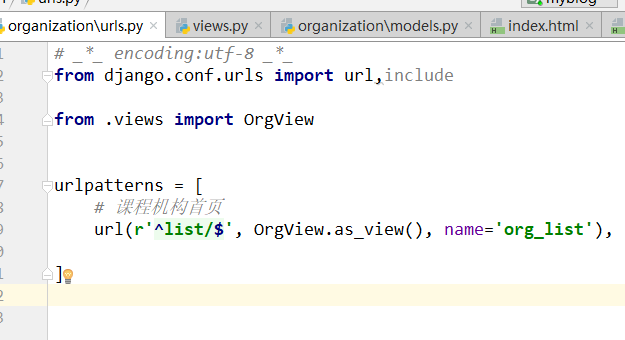
Github链接：<https://github.com/jamespacileo/django-pure-pagination>

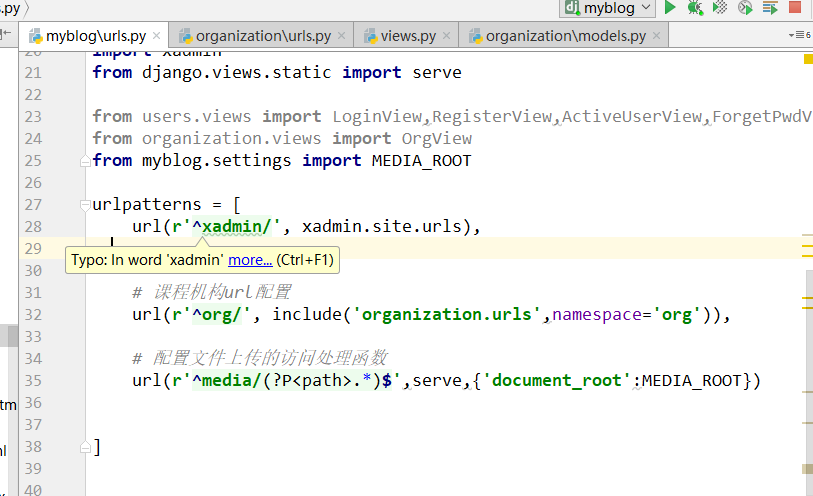
使用ModelForm代替Form验证form表单

1. 新建一个forms.py
2. 导入forms，还有要验证的数据
3. 继承并添加自己要的内容

Django的url分类

1.在每个app下分别创建一个urls，然后将url路径拷贝过去



2.在总的urls页面中导入个个app中的urls

3.使用时候就先引用总url里面的名字，然后冒号，在引用app里面具体的url地址