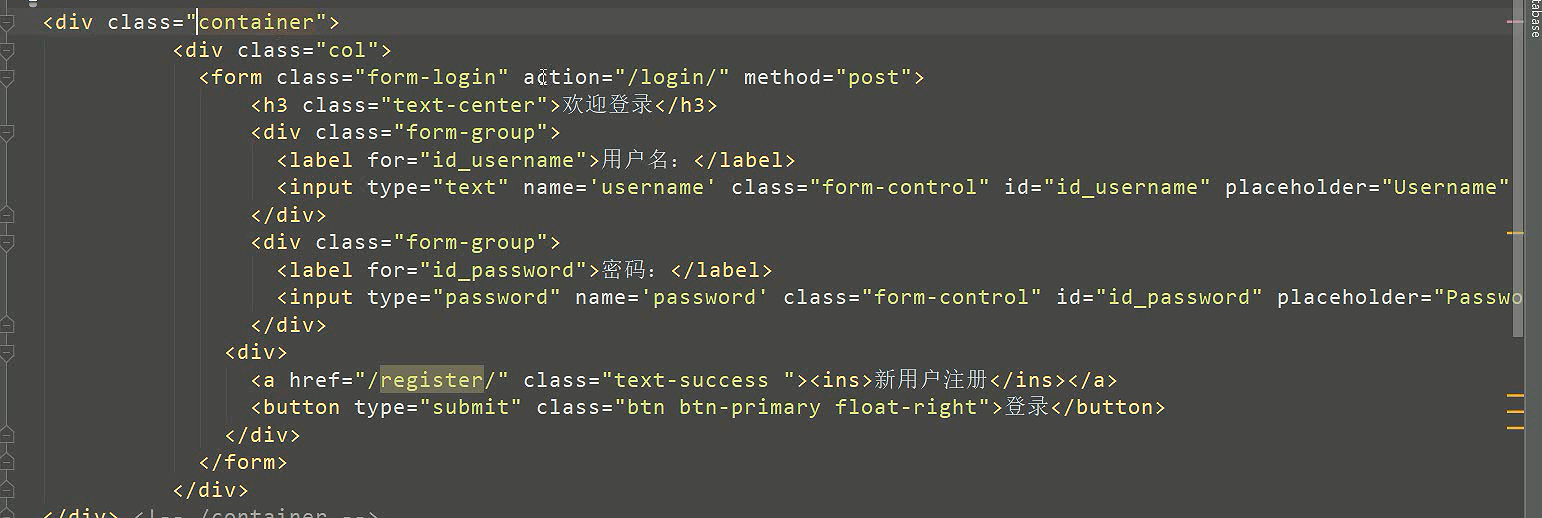
**1. model编写**

**2. 路由编写原则**



**3. login.html**



**4. 如果没有找到sername对应的值则会返回None,不会报错，如果是直接POST["username"]那么找不到就会报错**

clipboard.png

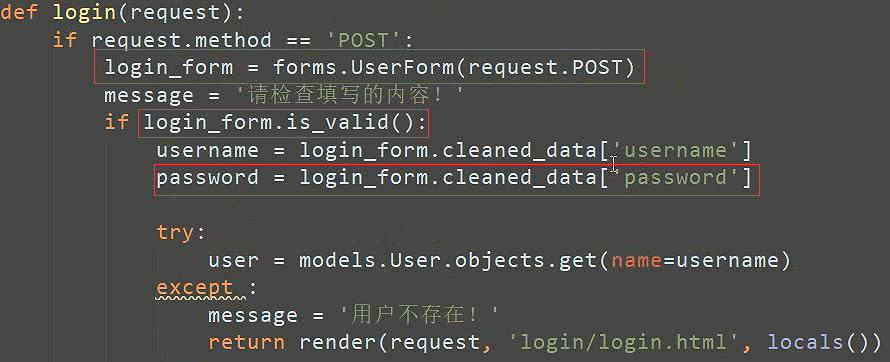
**5. form表单：**

**1. 在app目录下新建forms.py文件**

**通过widget小部件指明password这个输入框为密码输入框**

clipboard.png

**2. 在views里面通过视图函数实现对form表单数据的初始化(用request.POST 填充内容)，合法性验证，数据获取**

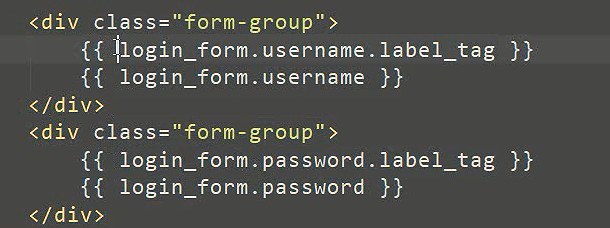


**3. 在login.html里面**

**直接调用{{ login\_form }}标签会少很多样式控制，需要自己手动添加样式 如:class， placeholder，autofocus等，即在forms.py里对username和password的widget小部件里面添加attrs属性通过字典形式赋值**



**4. 在login.html通过表单模板调用**



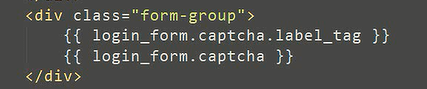
**6. 图形验证码：**

1. pip install captcha，并注册到app里面

2. 在forms.py里面添加captcha字段

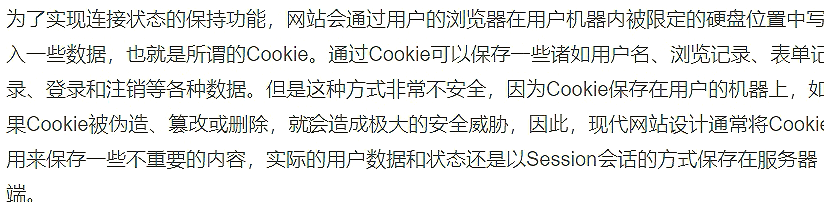
clipboard.png

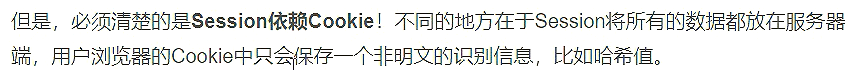
3. 在login.html里面使用



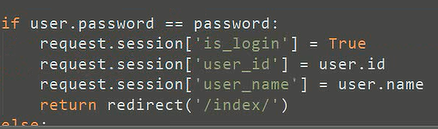
7. session会话

cookie

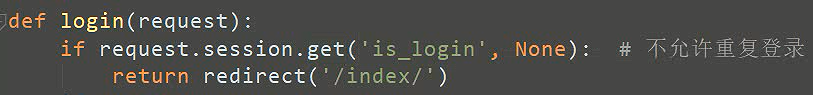




**1. 用户登录成功，则在session添加内容如：把is\_login设为True......**

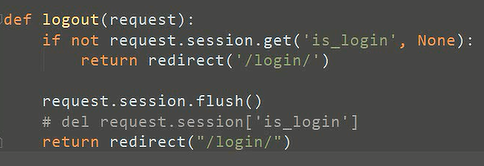


**2. 在视图里限制用户登录**



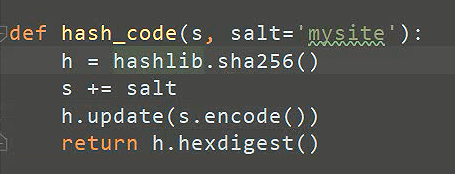
**3. 用户登出后，清楚session里面的内容**

flush一次清空所有内容，也可以指定键值一个一个清除



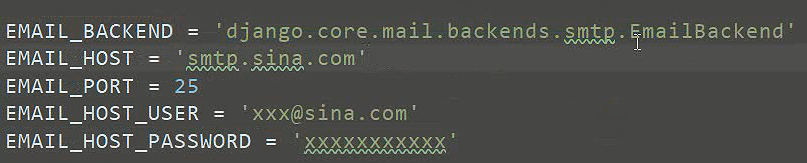
**8. 注册视图**

密码加密：



**9. Django发送邮件**

1. 在settings里面配置：

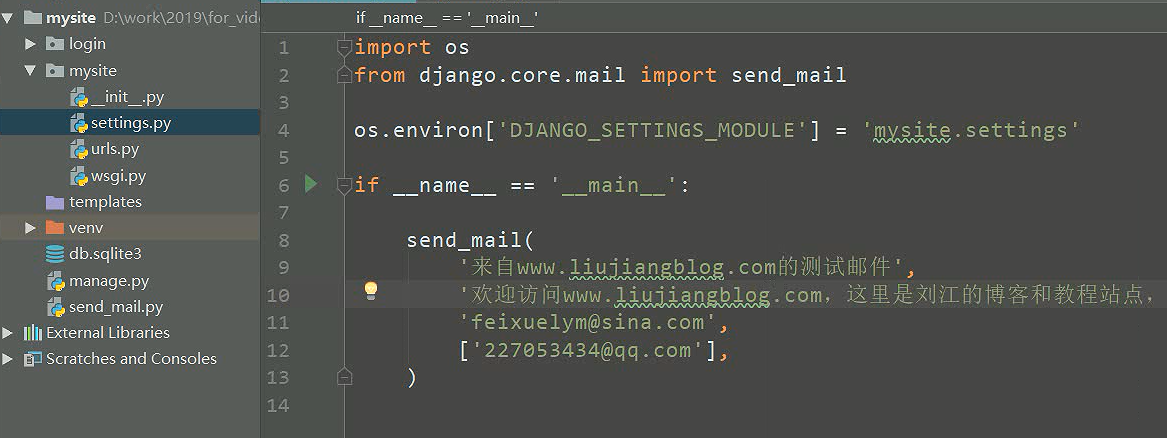


2. 邮件主题

3. 邮件内容

4. 发件人

5. 收件人

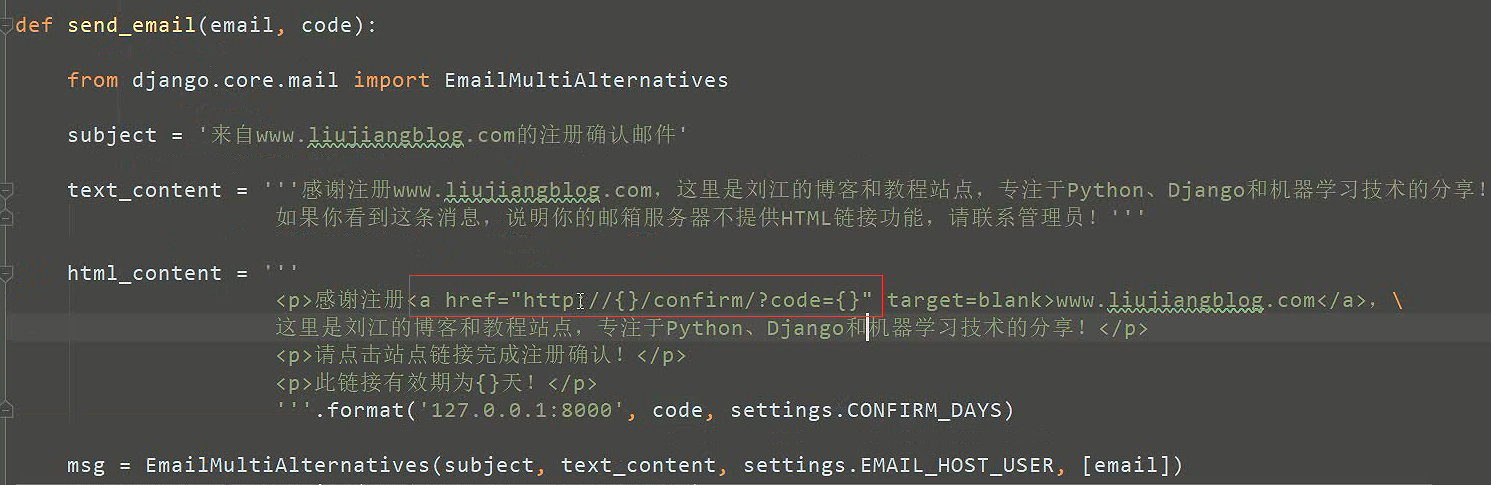


或者

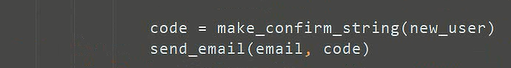


**10. 注册邮件确认**

1. 构造确认链接



用户在时注册执行如下方法并发送邮件



**2. 编写url接受确认请求**

clipboard.png

**3. 编写视图函数处理确认请求**



**4. 编写confirm.html处理确认请求：**

2秒后自动跳转到/login/

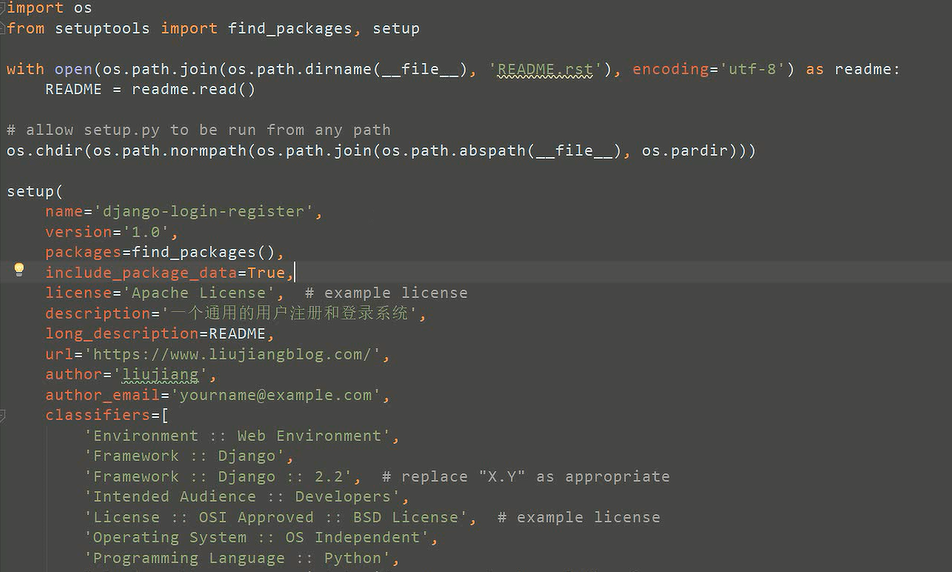


**5. 用户登陆时判断用户是否经过邮箱确认**



**11. 打包app**

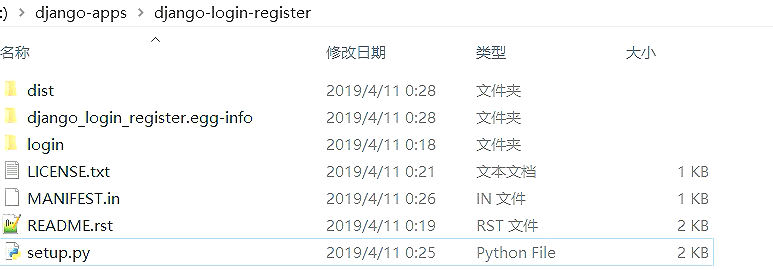
新建django-login-register文件，里面编写setup.py



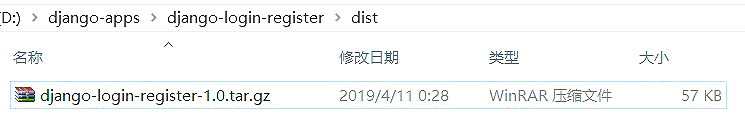
在pycharm的terminal终端下切换到django-login-register目录下执行：

python setup.py sdist

在该目录下会生成dist和django-login-register-info



在dist里面



**11. 重用app**

**1. 在pycharm的terminal终端下**

clipboard.png

**2. 配置settings(注册app，配置邮箱设置等)**

**3. 编写urls**

**4. 创建表**

**5. 运行服务**