Python Web框架Django课程

Django第6天

Session与Cookie

客户端可以设置Cookie发送给服务器
Cookie是key:value格式，每一个客户端可以自行定义key:value，服务器不好管理客户端上传的Cookie，服务器要求客户端保存一些Cookie方便管理。服务器告诉客户端，必须存储一个sesssionid的cookie，并且值必须是服务器给出的。所以浏览器里面有一个sessionid的cookie

通过response.set_cookie来告诉客户端保存服务器给出的cookie

这样每次客户端，把sessionid这个cookie发送给服务器，服务器就知道是哪一个客户端了。

服务器把每一个客户端信息保存在数据库当中，数据库三个字段，

session_key:保存的是sessionid对应的值
session_data:保存的数据信息
expire_date:表示这条记录什么时候过期

django读取session的过程:
读取客户端请求中的Cookie—->取出sessionid—>从数据库中查询出session数据—>解码成一个字典

用户注册与登录

1. 用户注册，新建一条用户记录
2. 用户登录，匹配用户名和密码

限制用户注册: 注册码

明文密码的缺陷
不安全，数据库用户信息被人盗走了，被人可以直接使用用户名密码登录

加密一下
使用简单的MD5在加上自定义的字符串来加密

```
md5((pwd+'askdjjflksjfdksf').encode()).hexdigest()
```

注册:客户端—>明文密码——>服务器加密——>保存加密后的数据
登录:客户端—>明文密码——>计算加密后的密码——>匹配数据库加密数据

Django自带User注册登录

```
from django.contrib.auth.models import User
# 注册
user = User.objects.create_user('john',
'lennon@thebeatles.com', 'johnpassword')
user.save()
```

登录:

```
from django.contrib.auth import authenticate, login,logout

def my_view(request):
    username = request.POST['username']
    password = request.POST['password']
    user = authenticate(request, username=username,
password=password)
    if user is not None:
        login(request, user)
        # Redirect to a success page.
        return HttpResponse('登录成功')
    else:
        # Return an 'invalid login' error message.
        return HttpResponse('用户名或密码错误')
def logout(request):
     logout(request)
     return HttpReponse('退出成功')
```

中间件

客户端请求———>[对request做一些操作]———>处理函数———>[对响应response一些其他操作]———返回给客户端

settings中可以看到django默认开启的中间件，Session就是使用了中间件，还有之前POST请求验证的CSRF验证

常见应用:

1. 统计
2. 黑白名单

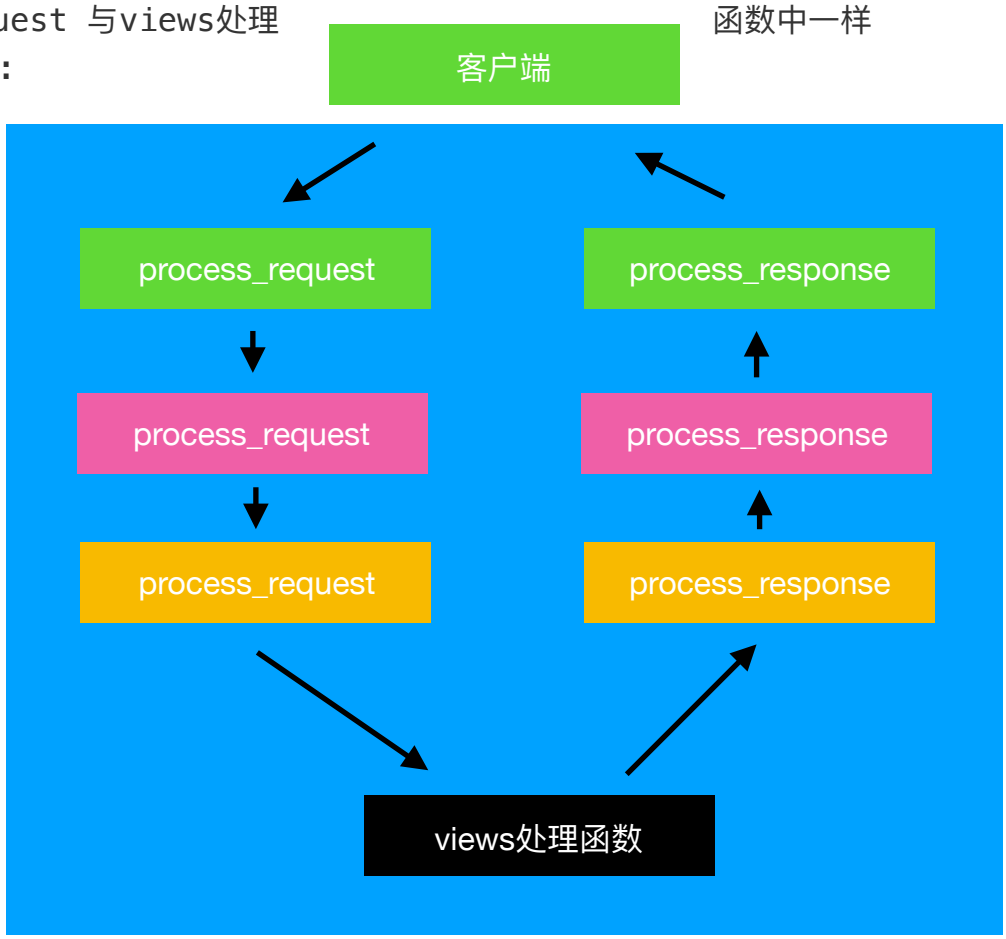
多个中间件执行过程 中间件1—>中间件2—>中间件3....

中间件如何编写

```
from django.utils.deprecation import MiddlewareMixin
class CommonMiddleware(MiddlewareMixin):
    def process_request(self, request,):
        pass
    def process_response(self, request, response):
        return response
```

process_request

参数:
request 与views处理函数中一样
返回:



1. None 继续执行后面的中间件
2. HttpResponse对象,后面的中间件不会被执行,直接把HttpResponse对象返回给客户端

process_response

参数:
request 与views处理函数中一样
response HttpResponse对象，通常是由views中处理函数返回的，或者其他中间件返回的
返回:
必须返回一个HttpResponse对象

中间件处理过程

启用中间件
settings.MIDDLEWARE 添加
关闭中间件
settings.MIDDLEWARE 移除掉

分页简单实现

数据库中包含大量的数据，每一只返回一部分给客户端。

数据库当中 分页使用limit和offset

django中使用queryset[start:end]来实现

```
class PageCuter(object):
    def __init__(self, model, limit=5):
        self.current_page = 1
        self.limit = limit
        self.model = model

    def has_next_page(self):
        pass

    def has_pre_page(self):
        pass

    def get_page(self, page=None):
        pass

    def next_page(self):
        pass

    def pre_page(self):
        pass

    def last_page(self):
```

```
        pass

    def first_page(self):
        pass
```