Web 项目1-day07-课堂笔记

商品模块-商品搜索功能

1.1 【理解掌握】商品搜索功能实现

1）**django-haystack配置**：具体参考讲义

2）**索引类定义和索引数据生成**：具体参考讲义

3）**商品搜索API接口实现**：具体参考讲义

1.2 【理解掌握】商品详情页面静态化操作

目的：每个 SKU 商品生成一个对应的静态详情页面

确定了要对商品详情页进行静态化操作之后，首先我们来思考一个问题：

商品详情页上的 **商品分类数据、面包屑导航数据、 SKU商品数据** 是基本不会发生变化，但并不是说一定不会变化，为了保持静态页面上的数据和数据库的数据保持一致，我们需要在上面这些数据发生变化时，让对应商品的静态详情页重新生成，该如何操作？？？

对于这个问题，有些同学马上会想到使用定时任务去做，但是商品详情页和首页数据相比，其更新频率很低，没有固定的更新频率，使用定时任务往往不能达到理想的效果。对于这个情况，我们可以将商品详情页静态化操作的过程封装成一个 Celery 的任务函数，当某个商品详情数据被更改时，直接通知 Celery 的 Worker 调用商品详情页静态化操作的任务函数，重新生成指定商品的详情静态页面即可。

购物车模块-购物车记录存储方案

2.1 【理解掌握】购物车记录存储方案

目的：用户无论是否登录，都可以进行购物车记录的添加。

1）**登录用户购物车记录存储**：

存储位置：存储到 redis 中

购物车记录数量小，操作频繁，我们可以将其放在 redis 数据库中进行存储，具体存储结构如下：

1）使用 hash 数据存储指定用户购物车记录中对应商品的 id 以及该商品在购物车中的数量，hash 的 key 以用户 id 进行标识

```
"cart_用户id": {
  "商品id": "数量",
  "商品id": "数量",
  "商品id": "数量",
  ...
}
```

2）使用 set 数据存储指定用户购物车记录被勾选记录对应的商品 id，set 的 key 以用户 id 进行标识

```
"cart_selected_用户id": (
  "商品id",
  "商品id",
  "商品id",
  ...
)
```

2）**未登录用户购物车记录存储**

存储位置：存储到 cookie 中

由于用户未登录，服务端无法拿到用户的 id，所以服务端在存储购物车记录时很难唯一标识该记录，基于这个情况，我们可以将未登录用户的购物车数据缓存到用户浏览器的 cookie 中，每个用户自己浏览器的 cookie 中存储属于自己的购物车记录数据。

1）在 cookie 中存储未登录用户的购物车记录数据时，存储一个 json 字符串数据即可，格式如下：

```
{
  "商品id": {
    "count": "数量",
    "selected": "是否勾选"
  },
  "商品id": {
    "count": "数量",
    "selected": "是否勾选"
  },
```

```
  "商品id": {
    "count": "数量",
    "selected": "是否勾选"
  },
  ...
}
```

2.2 【掌握使用】pickle 和 base64 模块基本使用

pickle 模块：

```
import pickle

# pickle.dumps方法：将`python原始数据`转换为`bytes数据`
pickle.dumps("python原始数据")

# pickle.loads方法：将转换后的`bytes数据`转换为`原始数据`
pickle.loads("转换后的bytes数据")
```

base64 模块：

```
import base64

# base64.b64encode方法：
# 将`Base64 编码前的bytes二进制数据`进行 Base64 编码操作，返回`编码后的bytes数据`
base64.b64encode("Base64编码前的bytes二进制数据")

# base64.b64decode方法：
# 将`Base64 编码后的bytes二进制数据|str数据`进行 Base64 解码操作，返回`解码后的bytes数据`
base64.b64decode("Base64编码后的bytes二进制数据|str数据")
```