白银电商网站

数据库实训总结报告

团队：2018级软件学院不屈白银队

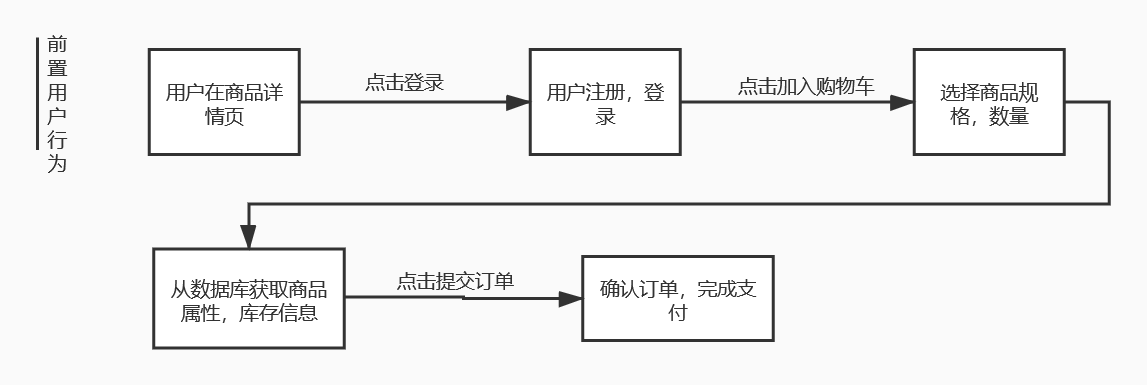
|  |  |  |
| --- | --- | --- |
| 姓名 | 班级 | 学号 |
| 刘启熙（组长） | 软件1班 | 201830730307 |
| 陈嘉俊 | 软件1班 | 201830720063 |
| 龚喜 | 软件2班 | 201830660345 |

github地址：https://github.com/baebae996/Database-project.git

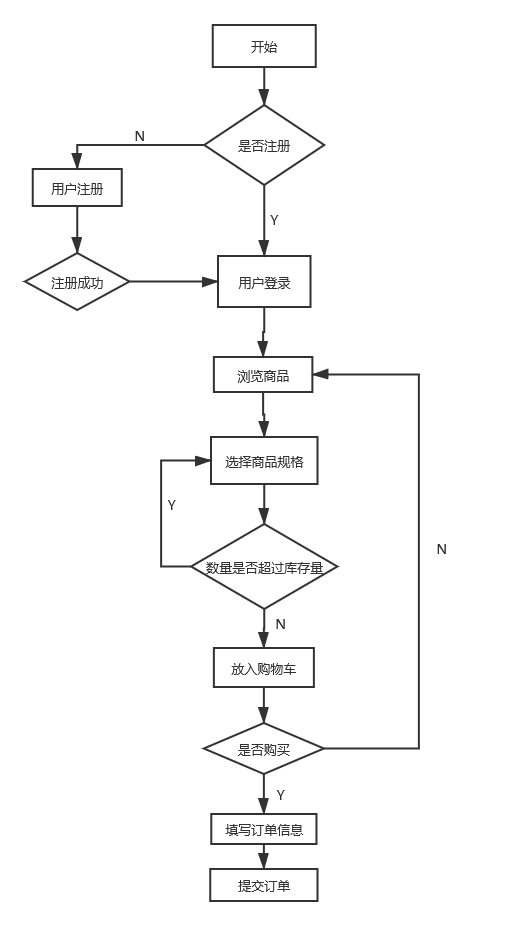
1. **目录**
   1. 项目成果展示
   2. 迭代过程中的困难及思考
   3. 项目设计优势
   4. 相关文档完成情况
   5. 小组经验总结
2. **项目成果展示**

**2.1 预期项目完成结果**

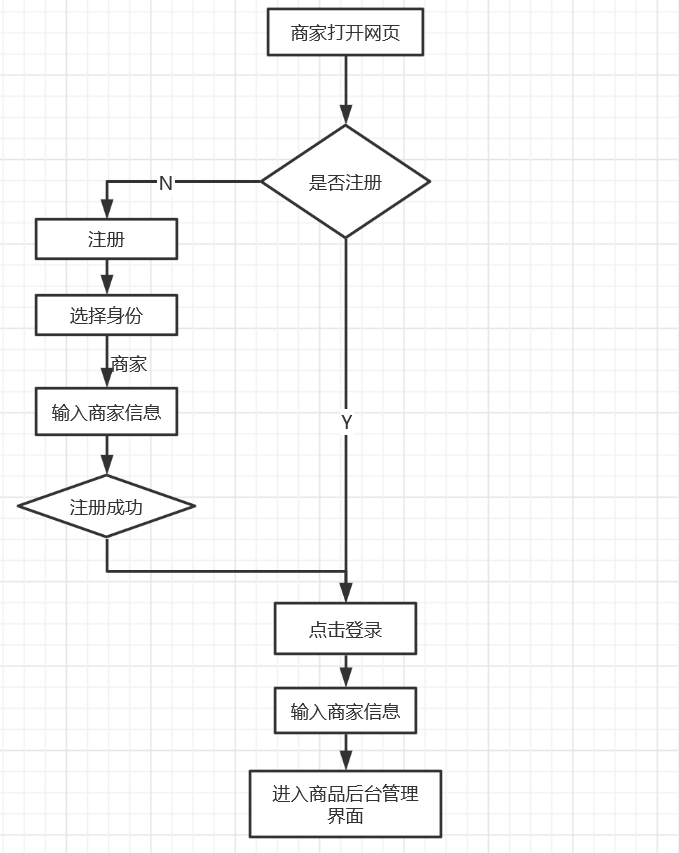
设计前端网页，后端数据库以实现事件流：

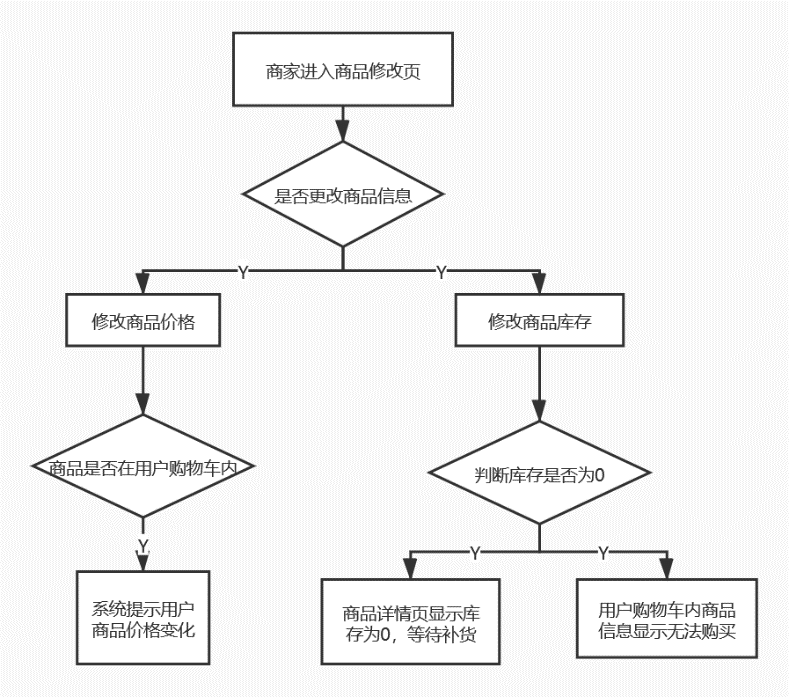


·用户：



·商家：

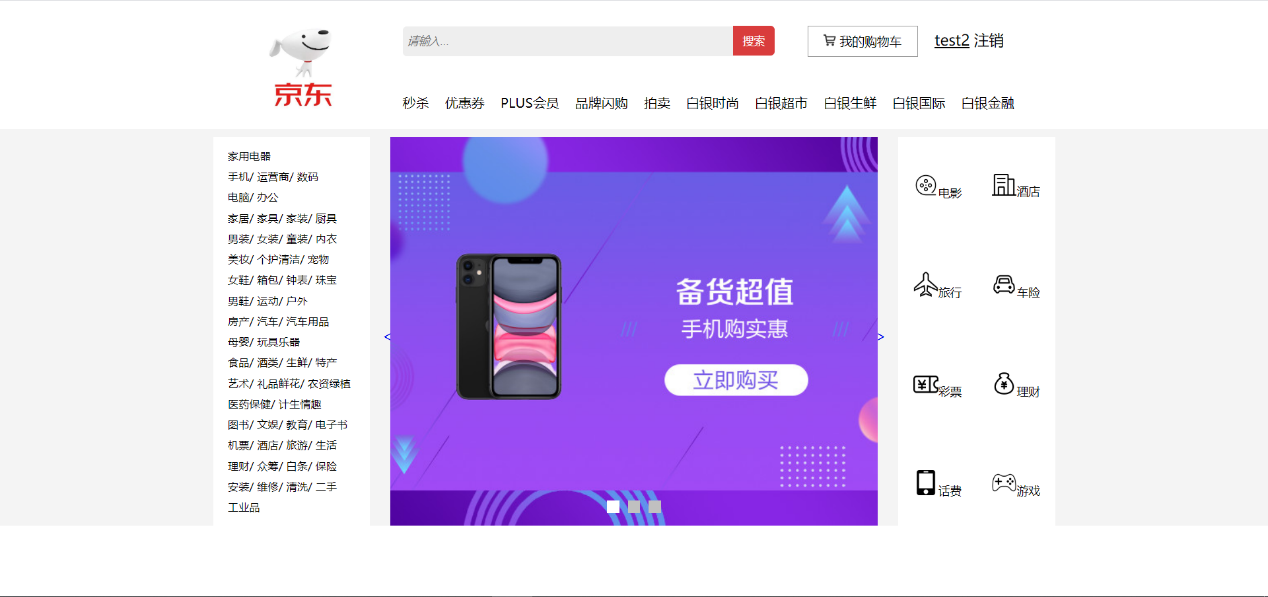




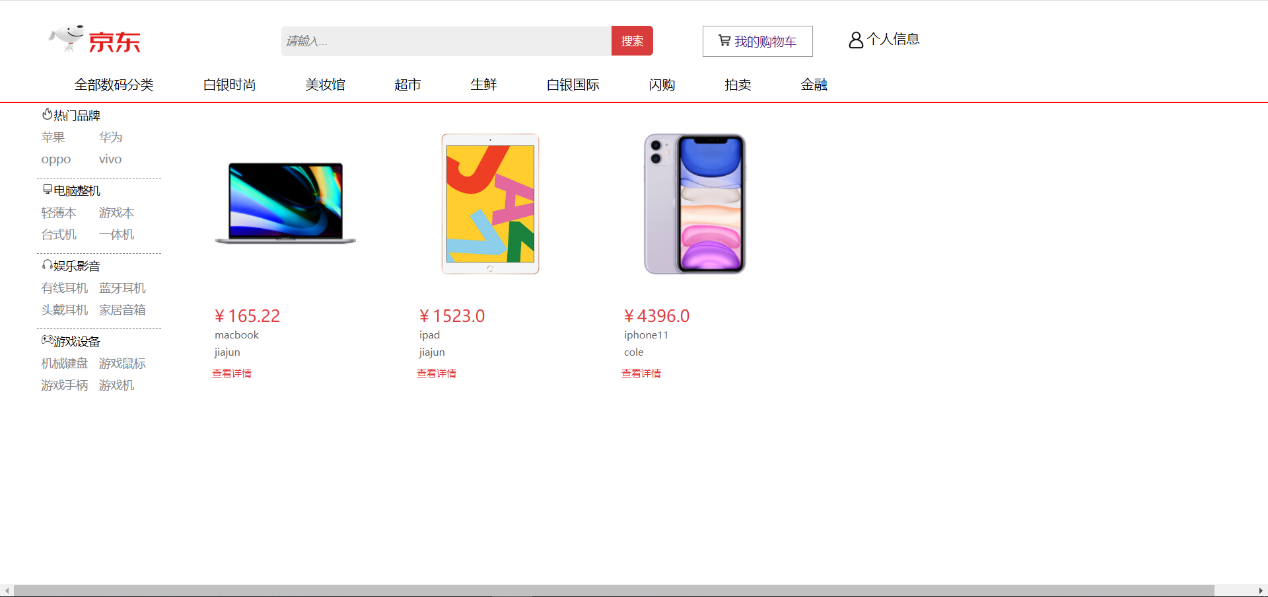
**2.2 实际完成情况和功能列表**

·前端网页：

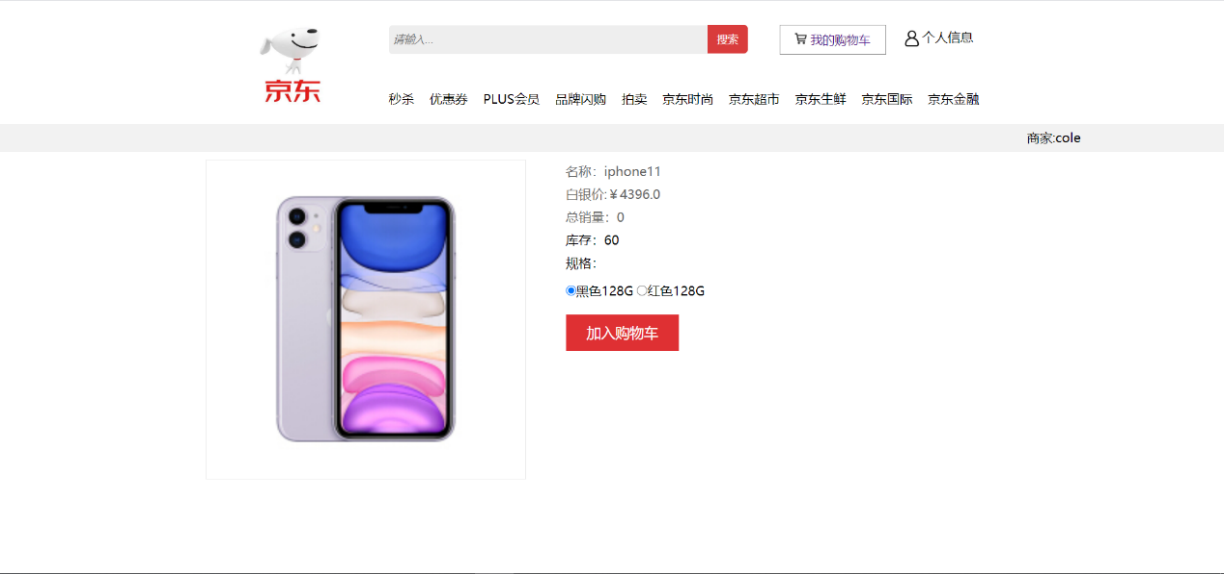
网站首页



商品展示页面



商品详情页面



购物车页面：  


订单页面

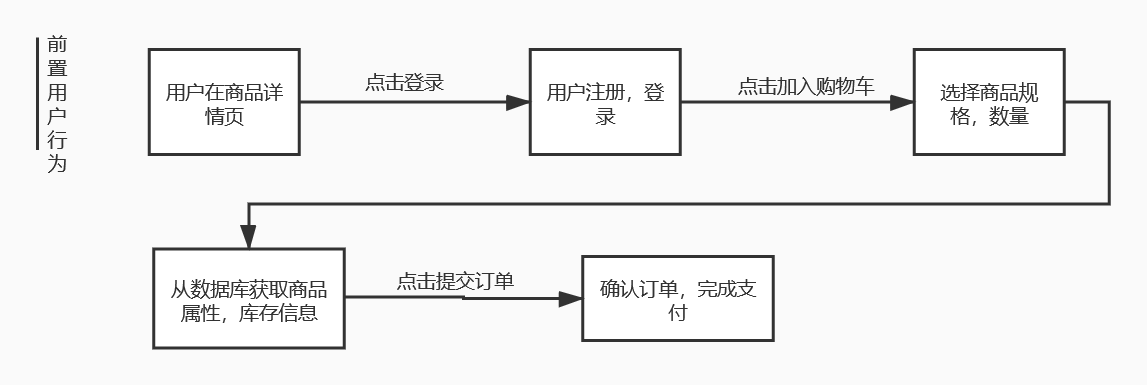


商家后台管理页面

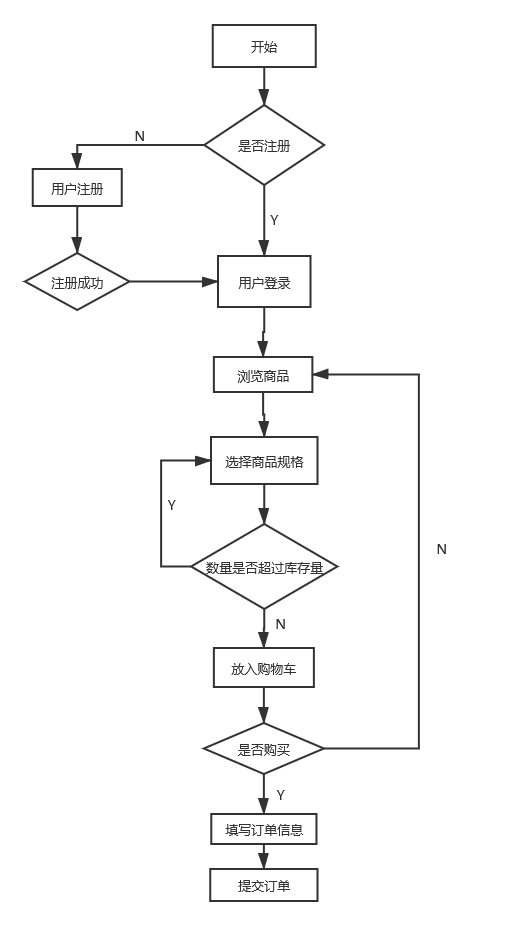


·完成事件流：

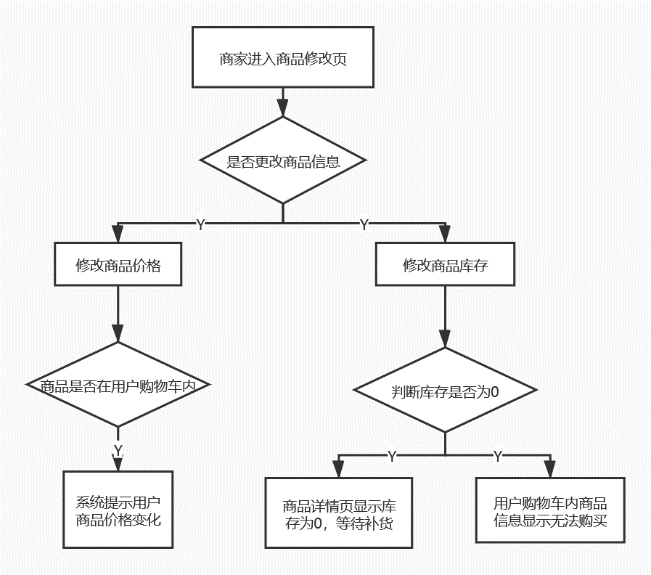
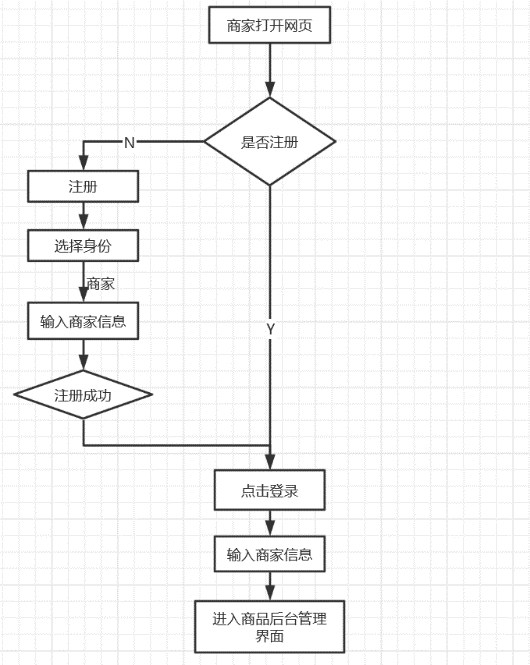
前端网页，后端数据库以实现事件流：



·用户：



·商家：



**功能列表：**1、 用户身份：  
 注册登录

根据品类浏览商品

查看商品详情（价格、库存、销量、规格）

选择规格加入购物车

提交订单

查看订单

2、商家身份：  
注册登录

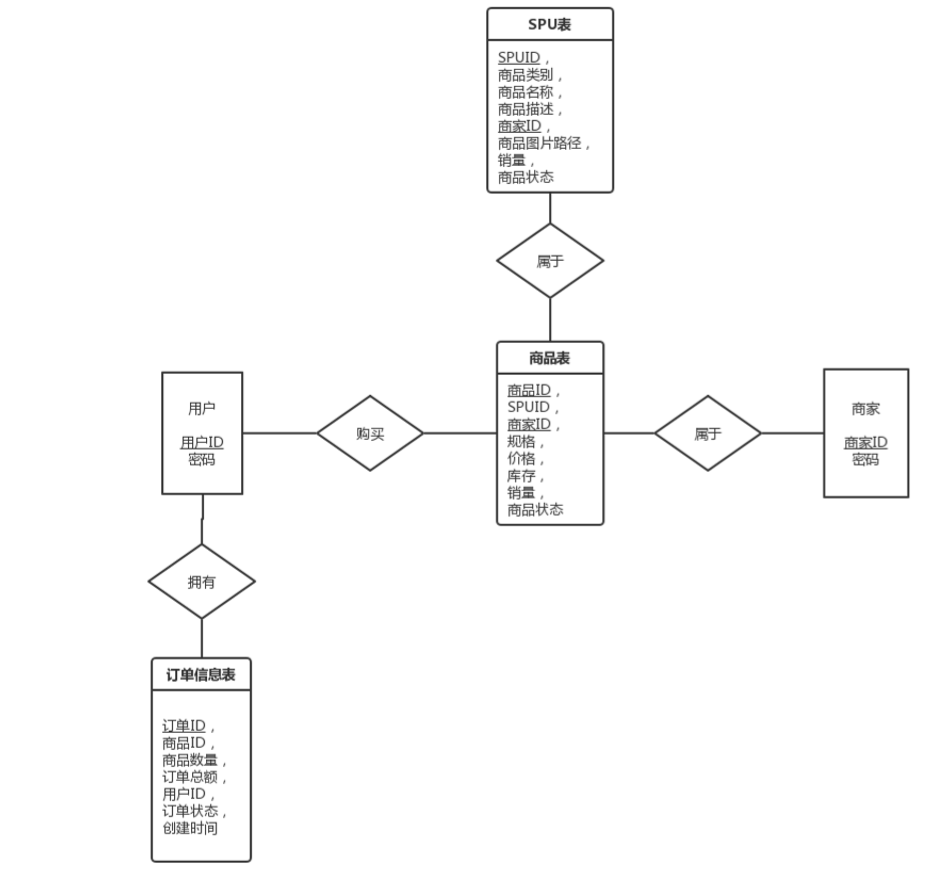
上架新商品

下集商品

修改商品规格信息（价格、库存）

查看不同级别的商品销量

**2.3 实际后台是如何支持这些流程？**

****

·SPU表(SPUID，商品类别，商品名称，商品描述，商家ID，商品图片路径，销量)

这个表主要用于在SPU（标准化产品单元）的层面上管理商品数据

* 主码SPUID和商家ID用于唯一地标识一件商品
* 商品类别用于标识商品所属的品类，便于分类展示商品
* 销量信息，一方面可以让商家了解自己的销售情况，另一方面，也可以为用户提供一个参考

·商品表(商品ID，SPUID，商家ID，规格，价格，库存，销量)

这个表主要用于在SKU（最小库存单元）的层面上管理商品数据

* 商品ID和商家ID用于唯一地标识一个最小库存单元
* 规格主要指的是商品的销售属性，例如“颜色”“内存”等

（在我们设计的系统中，存入数据表中的规格是组合后的销售属性，例如“黑色128G”、“白色256G”等，这样设计的好处在于，用户打开商品详情页面后，只需对规格作一次选择，即可将商品加入购物车，通过节省用户选择的时间，一定程度上提高了用户的购物体验）

·订单信息表(订单ID，商品ID，商品数量，订单总额，用户ID，订单状态，创建时间)

这个表主要用于收集关于商品和用户的订单数据，进行库存和销量的更新

* 当用户将商品加入购物车时，就生成一条订单记录，并标记订单状态为“未完成”
* 当用户在购物车提交订单时，对应的订单记录将会被更新，并标记订单状态为“已完成”

**2.4 本小组的实现方法在实际应用中可能存在问题**

1. 关于商品图片大小的限制

·情况:以京东商城为参考，为了保持界面风格的统一性，商品图片绝大多数都是正方形的，而在我们目前完成的“上架新商品”页面中，我们只规定商家上传的图片大小不应超过2MB，并没有对图片的具体形状作进一步的要求和限制，因此在我们的白银电商网站中，若商家上传了非正方形的商品图片，可能会导致界面的展示不够美观。

1. 关于商品规格在数据库中的定义和存储形式

·情况：目前在后端数据库中，我们团队是这样定义商品规格的：每一种具体的规格（即每一个最小库存单元）对应商品表的一个元组。（图）

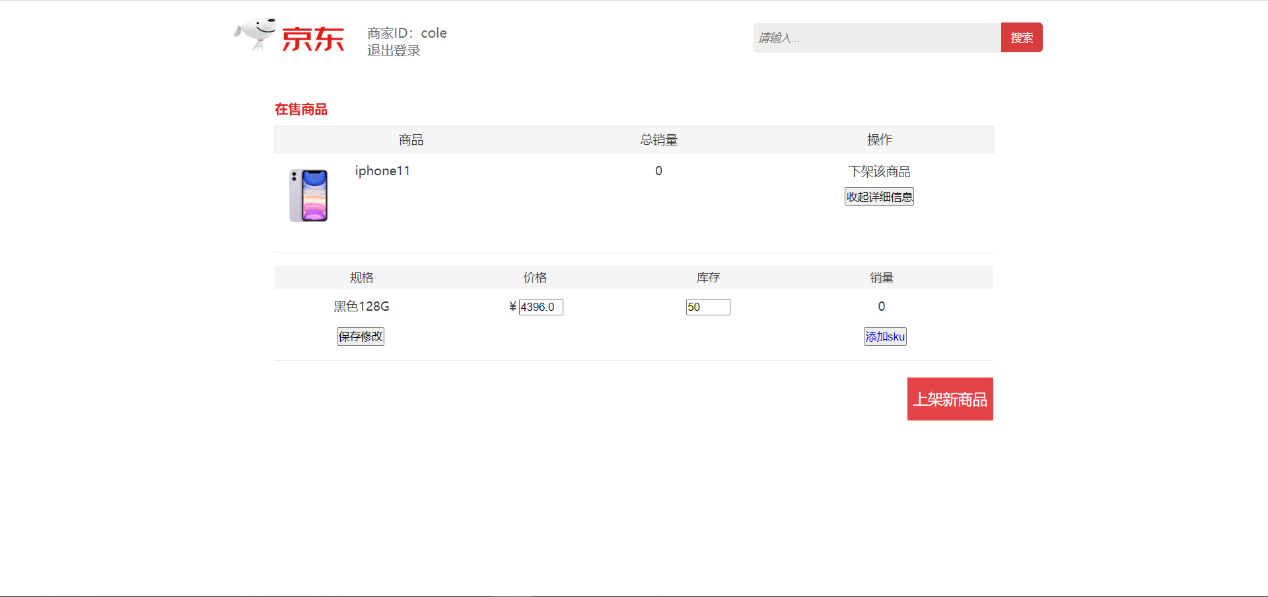
·实现方法的优势在于：用户打开商品详情页后，只需对规格作一次选择，即可将商品加入购物车；而在实际应用中，每种商品可能会有很多可选的规格，因此商品表中元组的数量会随着商品数量的增长而不断增长，导致商品表的空间开销越来越大；另一方面，商品详情页面在加载时会向后端发送GET请求，查询并获取当前商品所具有的全部规格信息，当商品表越来越大时，需要的查询时间也会有所增长，从而间接地增加了页面被完整加载的时间。

1. 关于平台对商家群体的管理

目前在我们的白银电商网站中，赋予商家群体的功能并不算多，包括：注册、登录、上架商品、下架商品、修改商品规格信息等等，这主要是项目开发周期较短、开发时间较紧张所导致的；在实际应用中，电商平台会实施更多商家管理的内容，例如：审核待上架的商品、维护一个针对商家群体的信用评价体系等等。

1. 关于商品的下架

目前在商家后台管理页面中，我们的平台只提供了以SPU为单位下架商品的功能。



也就是说，商家无法以SKU为单位下架商品，只能通过将SKU的库存修改为0，从而间接实现停止售卖某种规格的商品的目的，这种处理方案不够直观、不够人性化，这主要是我们前期对前端页面设计的讨论和考虑不够周全、不够细致所导致的。

1. **迭代过程中的困难及思考**

**3.1** **前后端数据交互受阻**

·情况：

由于负责后端开发的同学租用的云服务器未能正式投入使用，我们的项目只能运行在本地服务器上，这就导致了负责开发的其他成员无法在自己的设备上运行完整的代码

·解决方案：

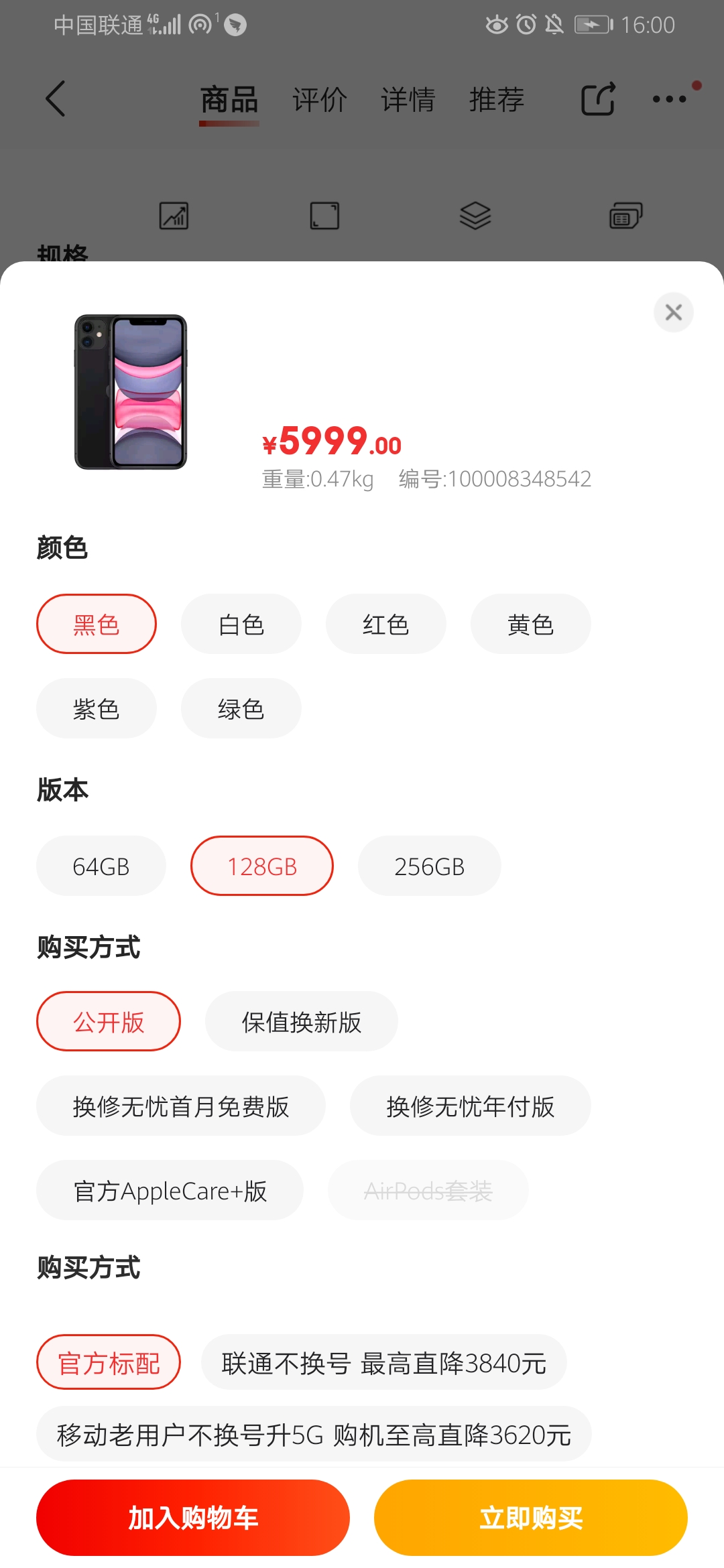
通过多次的开启屏幕分享(QQ电话)来排查错误，一定程度上降低了调试和开发的效率。

**3.2商品规格的展示问题**

·问题具体描述：

商品表里关于商品规格的2种定义方式及其对应数据存储方式

1. 京东：{“颜色”:[xxx,xxx], “存储大小”:[xxx,xxx] }
2. 淘宝 组合后的：黑色128G



·优劣势对比

（1）京东将商品的每个属性都单独排成一行，用户在将商品加入购物车之前，需要对每一个属性都作出选择，以iphone11为例，用户需要选择颜色、存储大小、套装、保障等等。这种排列方式与“商品类别表（SPUID，商品名称，属性）{“存储”：[128G,256G]，“颜色”：[黑色，白色，黄色]}”的设计理念一致，一个商品详情页对应这个表中的一条记录

·优点：用户可以清晰地看到属性的所有选项，并且数据表的冗余程度不会特别高

·缺点：不是任何类型的数据库都支持{key:[value,value], key:[value,value]}这种存储格式，并且，当某种属性的组合无库存或不存在时，用户就需要重新选择属性，一定程度上降低了用户体验，例如：用户可能在选择了“紫色”之后，才发现这家商店没有提供“紫色+256G”这样的组合

（2）淘宝 组合后的：黑色128G

用户在将商品加入购物车前，对于每一个商品属性组合进行选择。以iPhone 11为例，用户直接选择所需机型。

·优点：用户可以看到商品的所有类型，并且直接选择喜爱类型，可以避免因为没有相应属性组合而重复选择的情况。

·缺点：若商品属性过多，数据库存储属性组合的开销变大，数据返回较慢。

·解决方案

因为呈现属性组合更易实现且更方便用户选择，而呈现属性组合的方式来引导用户选择商品

**3.3** **SPU和SKU设计**

·情况：中小型购物商城，商品的种类繁多，每一种商品具有不同的规格，不同的规格有相应的价格和库存

·核心问题：**不同商品具有不同的属性，如何设计通用的存储方案来达到高效管理商品的目的**

**·解决：根据商品的属性类别分成两个层次，商品的关键属性对应SPU（**不会影响到库存和价格的属性**，**与商品是一对一的关系**）**，商品的销售属性对应SKU（会影响到库存和价格的属性，与商品是多对一的关系），以此来实现不同级别的统计分析。

**3.4 下架商品操作**

·情况:商家下架某商品时，在不修改数据表结构的前提下，只能在SPU表中删除一条记录，无法保存这个下架商品的相关信息

·解决方案：在SPU表增加了一个新的字段“状态”，取值为在售/下架，用于短暂地保存下架商品的信息

**3.5 下架商品管理的思考**

实际上，在一段时间过后，继续保存已下架商品的相关信息意义不大，因此，平台的后端管理/开发人员，可以定期（例如半年/一年）清除数据表中状态为“下架”的商品信息，从而在一定程度上释放数据表的空间

1. **项目设计优势**

1、在数据库的设计方面，我们的项目优势在于：

数据表字段的冗余程度较低、数据定义的方式清晰明确

2、在网站前端的设计方面，我们项目的优势有如下三点：

* 页面简洁：在刚刚的展示中可以看到，我们的网站内容划分明确，几乎每个页面都没有多余的元素，所有的内容都是对用户或者商家有用的信息
* 风格统一：网站背景为纯白色，文字颜色主要为黑色或者浅灰色，导航栏主要分布在顶部和左侧
* 操作简便、重点突出：在白银电商网站中，对用户或商家具有重要导向意义的按钮（例如：“加入购物车”、“结算”、“提交订单”、“上架新商品”等等）都使用了鲜明的红色来进行标注；同时将网站促销活动的信息放在首页的轮播图中，确保用户在进入网站时就能看到重要的信息

1. **相关文档完成情况**

5.1 软件需求归约

5.2 用例需求归约文档

5.3 用例实现归约文档

* 1. 软件架构文档
  2. 迭代评估文档

1. **团队经验总结**

**6.1团队沟通及迭代计划总结**

·情况：项目开发后期，发现只能下架某一类别的商品，而不能仅下架该类别商品的某一单品，如：只能一次性下架iPhone 11，而不能仅下架iPhone 11 白色 128G的型号。

·原因：团队前期对于下架功能讨论定义不到位，后期发现该情况时，由于数据库已确定，修改工作量较大，放弃修改，保留一次性下架商品功能。

·团队经验总结：

开发过程中遇到的问题主要是由项目前期设计不够细致、成员间沟通不到位所导致的。因此，在日后的团队协作中，我们应该通过会议尽可能细致地确定具体的实施方案，再通过代码来实现具体的功能。

**6.2 产品展示，与客户的沟通**

·情况：在每周的迭代成果展示过程中，我们团队主要以文字(word文档)的形式进行展示，这种方式并不能让客户直观地感受我们的项目进度。

·团队经验总结：每次的项目成果展示突出重点，在时间允许的情况下制作PPT，从而让客户更直观的地感受我们的项目进度

**6.3 专业技能的收获**

**刘启熙：**

在这次数据库开发实训中，我主要负责的是前端页面的设计和实现。在开发网页的过程中，我学习并掌握了一些常见html标签的用法、常见CSS属性的用法，并且学会了利用浏览器的开发者工具，对网页的静态布局进行在线调试；同时，这个电商网站所涉及的一系列交互操作也让我对javascript这门语言变得更加熟悉：一方面，我接触到了前后端通信的几种常见方案（表单验证、GET请求、POST请求等），并且学会了利用ajax技术向后端发送GET或POST请求的具体写法；另一方面，在开发这十几个网页的过程中，出现过大大小小的bug，无形之中也提升了我一步步找出问题、排查错误、解决问题的能力，丰富了自身编写代码的经验，也进一步加深了我对前端开发的学习兴趣和动力。

**陈嘉俊：** 在这次实训中，我主要负责的是后端的设计与实现，主要的工作是：进行数据建表、数据处理以及数据渲染，这个过程进一步加深了我对python的Django框架的认识和理解。在前后端通信的过程中，我也学会了如何选择合适的数据格式（例如JSON对象或数组）来封装后端的数据，学会了如何正确地接收和响应前端发送的请求，并对数据作进一步的处理。

**龚喜：**

在这次实训中，我与陈嘉俊同学共同负责后端的设计与实现。在实训开始之前，我并没有接触过python这门脚本语言，但java和C++的基础让我能够快速上手这门语言，并在两周的时间内学会利用Django框架实现用户/商家注册登录时的表单验证。同时，在后续的开发中，我也深刻认识到，对数据表关系的定义必须十分慎重，尤其是外键的引用方面，这也进一步加深了我对mysql的认知。