**攀枝花学院**

**Panzhihua University**

**攀枝花学院本科毕业设计（论文）**

攀枝花特色农产品展销平台的设计与实现

院 （系）： 数学与计算机学院

专 业： 信息与计算科学

班 级： 2015级信息与计算科学

学生姓名： 王海灵 学 号: 201510802049

二〇一久**年五月二十日**

摘要

在一个民以食为天的社会中，而中国也是一个农业大国，而农产品销售渠道相对来说，在互联网一块并没有很大的影响，以致于很多人都不知道可以再互联网上购买农产品，为了维护农户们的利益，同时也为了让农业在互联网这一块受到重视，提高农产品市场竞争力，在随着科技的不断进步，技术的不断发展趋于成熟的趋势，农产品展销平台的设计将是把生活融入了技术当中。

本次平台的设计是把web技术、后端nodejs技术与数据库相结合，本平台有web浏览器、服务器和数据库结合而成，具有首页、商城、商店管理、购物车这几大模块，为广大用户提供了一个快捷的农产品展销平台，本系统注重为农产品提高竞争力与影响力，构建绿色社会而做出的贡献。

关键词：农产品，web技术，食品安全，node技术

ABSTRACT

In a society where people depend on food for their livelihood, and China is also a big agricultural country, and the distribution channels of agricultural products have relatively little influence on the Internet, so that many people do not know that they can buy agricultural products on the Internet again, in order to safeguard the interests of farmers, but also in order to make agriculture be valued on the Internet and improve the competition in the market of agricultural products. With the continuous progress of science and technology and the maturity of technology.

the design of agricultural products exhibition platform will integrate life into technology. The design of this platform is the combination of web technology, back-end technology and database. This platform is composed of web browser, server and database. It has home page, shopping mall, store management and shopping cart. The module provides a fast agricultural product exhibition platform for the vast number of users. The system pays attention to the contribution of improving the competitiveness and influence of agricultural products and building a green society.

Key words: Agricultural Products, Web Technology, Food Safety, Node Technology

目录

**[摘要](#_Toc7994)** [I](#_Toc7994)

**[ABSTRACT](#_Toc515)** [II](#_Toc515)

**[1 绪论](#_Toc10733)** [1](#_Toc10733)

**[1.1 农产品展销平台研究背景](#_Toc27536)** [1](#_Toc27536)

**[1.2 农产品展销平台的目的和意义](#_Toc15347)** [2](#_Toc15347)

**[1.3 课题解决方案](#_Toc1688)** [2](#_Toc1688)

**[2 关键技术介绍](#_Toc16430)** [1](#_Toc16430)

**[2.1开发平台](#_Toc8205)** [1](#_Toc8205)

[2.11 nodejs技术 1](#_Toc3835)

[2.12 express开发框架 1](#_Toc14662)

[2.12 B/S模式 2](#_Toc11540)

**[2.2开发工具](#_Toc12745)** [3](#_Toc12745)

[2.21 Visual Studio Code 3](#_Toc6844)

[2.22 Navicat 12 for MySQL 4](#_Toc29789)

**[3 系统分析](#_Toc32002)** [1](#_Toc32002)

**[3.1 系统概述](#_Toc27049)** [1](#_Toc27049)

**[3.2 可行性分析](#_Toc31325)** [1](#_Toc31325)

**[3.3 需求分析](#_Toc17656)** [1](#_Toc17656)

**[4 系统概要设计](#_Toc11507)** [1](#_Toc11507)

**[4.1 系统部署架构](#_Toc9167)** [1](#_Toc9167)

**[4.2 系统总体结构](#_Toc24644)** [3](#_Toc24644)

**[4.3 系统业务流程](#_Toc32631)** [5](#_Toc32631)

**[4.4 购物流程图](#_Toc15640)** [6](#_Toc15640)

**[4.4 功能模块](#_Toc10164)** [6](#_Toc10164)

[4.4.1 商品管理 6](#_Toc24025)

[4.4.2 购物车管理 6](#_Toc30768)

[4.4.3 商城管理 6](#_Toc22210)

[4.4.7 前台系统 6](#_Toc18038)

**[4.5 数据库设计](#_Toc22669)** [7](#_Toc22669)

[4.5.1 数据库概要设计 7](#_Toc13410)

[4.5.2 数据库逻辑设计 7](#_Toc2435)

[4.5.3 数据库物理模型 9](#_Toc13171)

**[5 系统详细设计](#_Toc20508)** [1](#_Toc20508)

**[5.1 功能调用流程](#_Toc10090)** [1](#_Toc10090)

**[5.2 前台详细设计](#_Toc25488)** [1](#_Toc25488)

[5.2.1 前台首页 1](#_Toc19222)

[5.2.2 商城商店详情页面 7](#_Toc23054)

[5.2.3 购物车页面 11](#_Toc20538)

**[5.3 后台详细设计](#_Toc26986)** [16](#_Toc26986)

[5.3.1 后台整体设计 16](#_Toc15345)

[5.3.2 商品管理 17](#_Toc30340)

[5.3.3 商品上架管理 21](#_Toc19206)

[5.3.4 商店管理 25](#_Toc12478)

**[6 系统测试](#_Toc17932)** [1](#_Toc17932)

**[6.1 测试目的](#_Toc8753)** [1](#_Toc8753)

**[6.2 前台测试](#_Toc11072)** [1](#_Toc11072)

[6.2.1 登录测试 1](#_Toc8841)

[6.2.2 注册测试 2](#_Toc1149)

[6.2.2 购物车测试 4](#_Toc18855)

**[6.3 后台测试](#_Toc12743)** [5](#_Toc12743)

[6.3.1 我的商店测试 5](#_Toc10367)

[6.3.2 上架下家商品测试 6](#_Toc30009)

**[6.4 测试结论](#_Toc468)** [8](#_Toc468)

**[7 结论](#_Toc13148)** [1](#_Toc13148)

**[参考文献](#_Toc18862)** [1](#_Toc18862)

**[致谢](#_Toc13098)** [1](#_Toc13098)

1 绪论

1.1 农产品展销平台研究背景

在1995年，农产品资源管理开始与互联网正式接轨，截至目前为止，国外对电子化的农产品资源管理的研究已较为成熟。在这时农产品资源管理应运而生，所谓指农产品展销平台，是指农户不仅局限于实体商店售卖农产品，而可以在互联网上这个更大更广的平台推销自己的产品。这种方式将会受到众多网友的喜爱并购买，这是在实体店中无法比拟的优势。在我国，互联网农产品展销平台资源管理研究起步较晚，但在2000年后，更多的农产品展销平台资源管理咨询公司、农产品资源专家、学者开始从事农产品资源管理电子信息化的研究，并在各种期刊的和网站上发表了大量研究成果。国内的一些大规模的企业也在这种氛围下，尝试应用电子化的农产品展销平台资源管理。

攀枝花地处中国西部攀西大裂谷，川滇交界，金沙江与雅砻江交汇处，以南亚热带为基带呈垂直分布的岛状立体气候，是攀枝花得天独厚的气候资源，形成了丰富多样的特色农产品，品质优良，上市季节早的特点，但其地理劣势阻碍了特色农产品走出去，而且目前本市农户分散，缺乏市场意识，品牌意识薄弱，营销战略不健全及营销渠道不畅通。

目前农产品的展销平台任然很少，知道并使用这种平台的也不多，大多都是在自家的店面售卖，相比较于网络渠道，顾客范围就会很明显的少了。现在主要有3中销售方式：1.是传统店面；2.网店的形式；3.传统店面和网店结合。相比较于前2种，第三种更受欢迎，而且销售量也更多，受现今网络迅速发展的影响，网店也是越来越吃香了。

工业品下乡，农产品进城，这是一个老生常谈的话题，电子商务发展到今天已经被越来越多的人接受。近两年，很多农民也体会到了网上购物的乐趣和便利。现在，很多农民都感叹电子商务的发展是如此之快，因为以前几乎没人关顾的乡镇快递服务点，现在每次去取快递都要排队等候。

工业品下乡了，但是农产品进城却成了一个难题，不过欣慰的是有一些农民开始尝试了，有个别几个还走在了前面。每次家乡的苹果一成熟，就会看到不少农民拿着纸箱拉着苹果到镇上邮寄苹果，不过数量还是非常有限，每次只有可怜的五六箱，好在都有这种农产品电商意识了。

不过话又说回来，要想提高自家农产品的知名度和销量，还是得有一个属于自己的网店，这也是长期发展下去的一个必备条件。

1.2 农产品展销平台的目的和意义

目的：攀枝花特色农产品展销平台的设计于实现；

意义：通过调查，了解攀枝花地区各种特色农产品的销售渠道，以及制定出相应的最优销售方案，进一步向攀西人们介绍家乡特色产品以及向外部推广出攀西特产，让攀西特产能够走出地理的限制，向更广的地区发展。

农产品展销平台，就是一个对外销售农产品的平台，可以为农产品的卖家提供一个更好的销售渠道，省去了去租店面的租金，带动农业经济的发展，加速推进社会主义经济价值观的体现，努力实现全民小康的目标。

1.3 课题解决方案

基于node+Ajax的农产品展销平台系统的设计与实现主要采用浏览器、服务器和数据库构成。分别为展示、业务和逻辑。数据库主要对各种数据的操作，包括登录注册，商家上架下架，客户评论；浏览器主要是满足客户浏览页面时的视觉效果，各种页面的交互数据传输等；服务器是对浏览器和数据库的数据交互的实现。这三种相辅相成。本系统开发环境是win 7操作系统，node是开发的后台语言，vscode是开发工具，MySQL为数据库，服务器是Apache，使用h5前端框架jQuery和ajax改善用户体验。

2 关键技术介绍

2.1开发平台

2.11nodejs技术

Node 是让 js语言运行在[服务端](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E7%AB%AF/6492316" \t "https://baike.baidu.com/item/node.js/_blank)的开发平台，它让 JS成为与[PHP](https://baike.baidu.com/item/PHP/9337" \t "https://baike.baidu.com/item/node.js/_blank)、[Python](https://baike.baidu.com/item/Python/407313" \t "https://baike.baidu.com/item/node.js/_blank)、[Perl](https://baike.baidu.com/item/Perl/851577" \t "https://baike.baidu.com/item/node.js/_blank)、[Ruby](https://baike.baidu.com/item/Ruby/11419" \t "https://baike.baidu.com/item/node.js/_blank) 等服务端语言平起平坐的[脚本语言](https://baike.baidu.com/item/%E8%84%9A%E6%9C%AC%E8%AF%AD%E8%A8%80/1379708" \t "https://baike.baidu.com/item/node.js/_blank)。它比较适合在高并发、I/O并发量大、业务逻辑相对较少的场景。

它的特点：

1. 它是一个Javascript运行环境

2. 依赖于Chrome V8引擎进行代码解释

3. 事件驱动

4. 非阻塞I/O

5. 轻量、可伸缩，适于实时数据交互应用

6. 单进程，单线程

优缺点：

优势：1. 高并发（最大的优点）；2. node的特性可以使你更好、更方便的编写你所想要的URL；3.适合I/O密集型应用；

劣势：1. 对CPU的应用效率过低，node是一个单线程的后台语言，如果有一个较大的算法，运行时间过长，就会导致CPU时间片不能释放，后续I/O操作无法继续执行；2. 对CPU的利用不够充分，浪费CPU的资源；3. 安全性低，如果一个进程崩溃，那么整个线程都会停止运行；

解决方法：1. 对于单线程、单进程的问题，可以开多个线程，绑定多个端口；2. 开多个进程监听一个端口。

2.12 express开发框架

Express 诞生已有时日， 是一个基于 Node.js 平台的极简、灵活的 web 应用开发框架，主要基于 Connect 中间件，并且自身封装了路由、视图处理等功能，使用人数众多。Express 自身集成了视图功能，提供了 consolidate.js 功能，支持几乎所有 JavaScript 模板引擎，并提供了视图设置的便利方法。Express 采用 callback 来处理异步，Express 使用 callback 捕获异常，对于深层次的异常捕获不了，优点：线性逻辑，通过中间件形式把业务逻辑细分、简化，一个请求进来经过一系列中间件处理后再响应给用户，清晰明了。 缺点：基于 callback 组合业务逻辑，业务逻辑复杂时嵌套过多，异常捕获困难。

2.12 B/S模式

B/S结构（Browser/Server，[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8/213911" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)/[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)模式），是[WEB](https://baike.baidu.com/item/WEB" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)兴起后的一种网络结构模式，WEB浏览器是[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF/101081" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)最主要的[应用软件](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E8%BD%AF%E4%BB%B6/216367" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)。这种模式统一了[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF/101081" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)，将系统功能实现的核心部分集中到[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8/100571" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)上，简化了系统的开发、维护和使用。客户机上只要安装一个[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)（Browser英 ['braʊzə]美 ['braʊzɚ]），如[Netscape Navigator](https://baike.baidu.com/item/Netscape Navigator" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)或[Internet Explorer](https://baike.baidu.com/item/Internet Explorer" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)，[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8/100571" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)安装[SQL Server](https://baike.baidu.com/item/SQL Server" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)、[Oracle](https://baike.baidu.com/item/Oracle" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)、MYSQL等数据库。[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8/213911" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)通过Web Server 同数据库进行数据交互。

由于Client/Server结构存在的种种问题，因此人们又在它原有的基础上提出了一种具有三层模式(3-Tier)的应用系统结构浏览器/服务器(Browser/Server)结构。Browser/Server结构是伴随着因特网的兴起，对Client/Server结构的一种改进。从本质上说，Browser/Server结构也是一种Client/Server结构，它可看作是一种由传统的二层模式Client/Server结构发展而来的三层模式Client/Server结构在Web上应用的特例。

Browser/Server结构主要是利用了不断成熟的Web浏览器技术：结合浏览器的多种脚本语言和ActiveX技术，用通用浏览器实现原来需要复杂专用软件才能实现的强大功能，同时节约了开发成本。

B/S最大的优点就是可以在任何地方进行操作而不用安装任何专门的软件，只要有一台能上网的电脑就能使用，[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)零安装、零维护。系统的扩展非常容易。

B/S结构的使用越来越多，特别是由需求推动了AJAX技术的发展，它的程序也能在[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)电脑上进行部分处理，从而大大的减轻了[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)的负担；并增加了交互性，能进行局部实时刷新。

（1）维护和升级方式简单。当前，[软件系统](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)的改进和升级越发频繁，[B/S架构](https://baike.baidu.com/item/B/S%E6%9E%B6%E6%9E%84" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)的产品明显体现着更为方便的特性。对一个稍微大一点单位来说，[系统管理](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)人员如果需要在几百甚至上千部电脑之间来回奔跑，效率和工作量是可想而知的，但B/S架构的软件只需要管理[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)就行了，所有的[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)只是[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)，根本不需要做任何的维护。无论用户的规模有多大，有多少分支机构都不会增加任何维护升级的工作量，所有的操作只需要针对[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)进行；如果是异地，只需要把服务器连接专网即可，实现[远程维护](https://baike.baidu.com/item/%E8%BF%9C%E7%A8%8B%E7%BB%B4%E6%8A%A4" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)、升级和共享。所以客户机越来越“瘦”，而[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)越来越“胖”是将来信息化发展的主流方向。今后，软件升级和维护会越来越容易，而使用起来会越来越简单，这对用户人力、物力、时间、费用的节省是显而易见的，惊人的。因此，维护和升级革命的方式是“瘦”客户机，“胖”[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)。

（2）成本降低，选择更多。大家都知道windows在[桌面](https://baike.baidu.com/item/%E6%A1%8C%E9%9D%A2" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)电脑上几乎一统天下，[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)成为了标准配置，但在[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)上windows并不是处于绝对的统治地位。 当前的趋势是凡使用B/S架构的应用管理软件，只需安装在Linux[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)上即可，而且安全性高。所以[服务器操作系统](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)的选择是很多的，不管选用哪种操作系统都可以让大部分人使用windows作为桌面电脑操作系统不受影响，这就使得最流行免费的[Linux操作系统](https://baike.baidu.com/item/Linux%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)快速发展起来，Linux除了操作系统是免费的以外，连数据库也是免费的，这种选择非常盛行。

比如说很多人每天上“新浪”网，只要安装了[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)就可以了，并不需要了解“新浪”的[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)用的是什么[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)，而事实上大部分网站确实没有使用windows操作系统，但用户的电脑本身安装的大部分是windows操作系统。

（3）应用服务器运行数据负荷较重。由于B/S架构管理软件只安装在[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)端（Server）上，[网络管理](https://baike.baidu.com/item/%E7%BD%91%E7%BB%9C%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)人员只需要管理服务器就行了，用户界面主要事务逻辑在服务器（Server）端完全通过WWW[浏览器](https://baike.baidu.com/item/%E6%B5%8F%E8%A7%88%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)实现，极少部分事务逻辑在前端（Browser）实现，所有的[客户端](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E7%AB%AF" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)只有浏览器，网络管理人员只需要做硬件维护。但是，应用服务器运行数据负荷较重，一旦发生服务器“崩溃”等问题，后果不堪设想。因此，许多单位都备有数据库[存储服务器](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/B/S%E7%BB%93%E6%9E%84/_blank)，以防万一。

2.2开发工具

2.21 Visual Studio Code

[Microsoft](https://baike.baidu.com/item/Microsoft" \t "https://baike.baidu.com/item/visual%20studio%20code/_blank)在2015年4月30日Build 开发者大会上正式宣布了 Visual Studio Code 项目：一个运行于 Mac OS X、[Windows](https://baike.baidu.com/item/Windows" \t "https://baike.baidu.com/item/visual%20studio%20code/_blank)和 Linux 之上的，针对于编写现代 Web 和云应用的跨平台源代码编辑器。该编辑器也集成了所有一款现代编辑器所应该具备的特性，包括[语法高亮](https://baike.baidu.com/item/%E8%AF%AD%E6%B3%95%E9%AB%98%E4%BA%AE" \t "https://baike.baidu.com/item/visual%20studio%20code/_blank)（syntax high lighting），可定制的热键绑定（customizable keyboard bindings），括号匹配（bracket matching）以及代码片段收集（snippets）。Somasegar 也告诉笔者这款编辑器也拥有对 Git 的开箱即用的支持。

微软正在努力为VS Code添加插件功能支持，本次更新还对相关文档进行了更新。使用体验(1) 可以直接点击 GUI 按钮或者右键进行新建 File、Folder 这样的操作。(以前的 Sublime 这方面比较差)(2) 直接分离 tab 和关闭 tab左边的按钮可以直接 split 窗口。比 Sublime 方便多了。右边的x可以直接关闭标签。然后左侧还可以看到一个大的蓝色按钮，可以直接 Open Folder，也比较爽。。这里也就直接暗示了以 Folder 为项目，不然很多新手弄不清楚打开文件、打开 Folder 什么的。(3) 无插件化的代码 diff非常强大的一个功能，目前发现两个方式可以启动 diff 功能。第一种是在左侧，右键文件 select to compare，然后再右键一个文件 Compare with #39;xx #39;，就可以出现 diff。第二种是 git 的模式下，点击右上角的 switch to changes file，就可以启动 diff。然后左上角点击...按钮，会出现然后点击 switch to Inline View，可以并到一个 tab diff。

2.22 Navicat 12 for MySQL

“Navicat”是一套可创建多个连接的数据库管理工具，用以方便管理 MySQL、Oracle、PostgreSQL、SQLite、SQL Server、MariaDB 和/或 MongoDB 等不同类型的数据库，并支持管理某些云数据库，例如阿里云、‎腾讯云。Navicat 的功能足以符合专业开发人员的所有需求，但是对数据库服务器初学者来说又相当容易学习。Navicat 的用户界面 (GUI) 设计良好，让你以安全且简单的方法创建、组织、访问和共享信息。

Navicat 提供了三种平台的版本 - Microsoft Windows、macOS 和 Linux。它可以让用户连接到本地或远程服务器，并提供一些实用的数据库工具以协​​助用户管理数据，包括 Navicat Cloud 协同合作、数据建模、数据传输、数据同步、结构同步、导入、导出、备份、还原、报表创建工具和自动运行。

3 系统分析

3.1系统概述

该系统基于以农产品商家为依托的农产品展销平台而开发的具体个性化系统，作为农产品展销平台，对客户重视程度格外重要。平台想要发展，也就必须以客户需求为主，集中处置上发挥自身优势，因此系统从客户需求出发，将此作为整个信息系统的开始，同时，本平台具有特殊性，本平台是农产品销售平台，我们将发挥这一优势，将在库存上的成本尽力降低至最低。

3.2可行性分析

1.在性能上，本系统力求稳定的环境，所以使用单线程的nodejs会使平台不仅稳定还很高效，为客户带来最佳的浏览体验；并且保证数据的安全性、一致性和准确性，高并发的特性可以使多个用户访问时也能达到信息的准确，稳定和安全；所以在性能上是可行的。

2.在经济上，本系统分为前端开发和后端开发，前期需要前端的UI设计及页面效果的交互，以及与一些需求分析，为保证客户在浏览页面的直观感受的良好效果，需要的效果比较丰富，由于开题时间较早，开发的时间比较多，所以需要的成本不高；而开发需要用到的设备和资源都是自己有的或者免费开源的，相对开发成本并不高；并且系统本身具有盈利性，所以在经济上具有可行性。

3.在操作上，本系统是针对电脑用户，只要客户有台电脑，就可以对这款系统进行操作，本系统符合www标准，界面简洁明了，并且现在基本上每家每户都有电脑可以上网，且在网页上直接浏览，不需要下载APP,所以本系统在操作上具有可行性。

4.在技术，使用的是现在流行的后端语言nodejs，数据库是用户范围最为广泛的MySQL，系统采用了MVC方式编程、B/S结构，使用的框架为开源的express，而express又有中文的官方文档，提供了非常详细丰富的API，因此在技术上是可行的。

5.在法律上，本系统开发过程中所用到的技术都是免费开源的合法的技术，且前后端代码都是原创的，不存在侵权、或者危害他人利益关系。所以在法律上是可行的。

3.3需求分析

功能需求：

本系统分为前台商品介绍系统和后台的商品管理系统。首先是后台系统：主要有3大模块：创建商店模块；商品管理模块；商品上架模块等，后台系统主要针对的是店主用户，可以上架下架自己店的物品；前台系统主要对购买商品的客户展示各个商家的商品，对自己喜欢的商品加入购物车，方便以后购买等功能。

非功能需求：

响应时间：在正常网速下，前后台系统响应时间不会超过5s；

最大访问量：在我用的服务器上（我的电脑），允许同时最多的访问量为不超过1w，这个是根据网页访问量可以配置不同性能的服务器提升带宽；

数据信息的安全高效：本系统采用了数据库隐私信息加密服务，并且采用了node这一运行速率快的后端语言；可保证数据信息上的安全高效；

可扩展性：本系统的编码格式采用了MVC框架，易读性代码对于本系统的可扩展性带来了极大的便利。

4 系统概要设计

4.1 系统部署架构

前端架构

应用层架构

安全架构 服务层架构 数据采集与监控

存储层架构

后台架构

数据中心机房架构

前端架构

用户请求到达网站应用服务器之前经历的环节，通常不包含网站业务逻辑，不处理动态内容。

1 浏览器优化技术： 页面缓存、合并HTTP减少请求数、使用页面压缩。

2 CDN： 内容分发网络， 部署在网络运营商网络。 静态页面分发到用户最近的CDN服务器。

3 动静分离： 静态资源(js、css)独立部署在专门的服务器集群, 和web动态服务分离， 使用专门

的域名(二级)。

4 图片服务： 独立部署图片服务器集群。

5 反向代理： 应用服务器、静态资源服务器、图片服务器之前， 提供页面缓存服务。

6 DNS: 域名服务， 可以用DNS负载均衡。

应用层架构

处理网站主要业务逻辑的地方。

1 开发框架。 分离美工和开发工程师，易于写作，内置安全策略。

2 页面渲染。 将动态内容和静态业务模板集成， 形成最终效果。

3 负载均衡。

4 session管理。 应用服务器集群，通常设计成无状态的，不保存上下文信息，需要专门的机制

管理session, 集群或者跨集群共享session。

5 动态页面静态化。 访问量特别大，且更新不频繁的动态页面静态化。

6 业务拆分。

7 虚拟化服务器。 一台物理机虚拟化多台虚拟服务器。

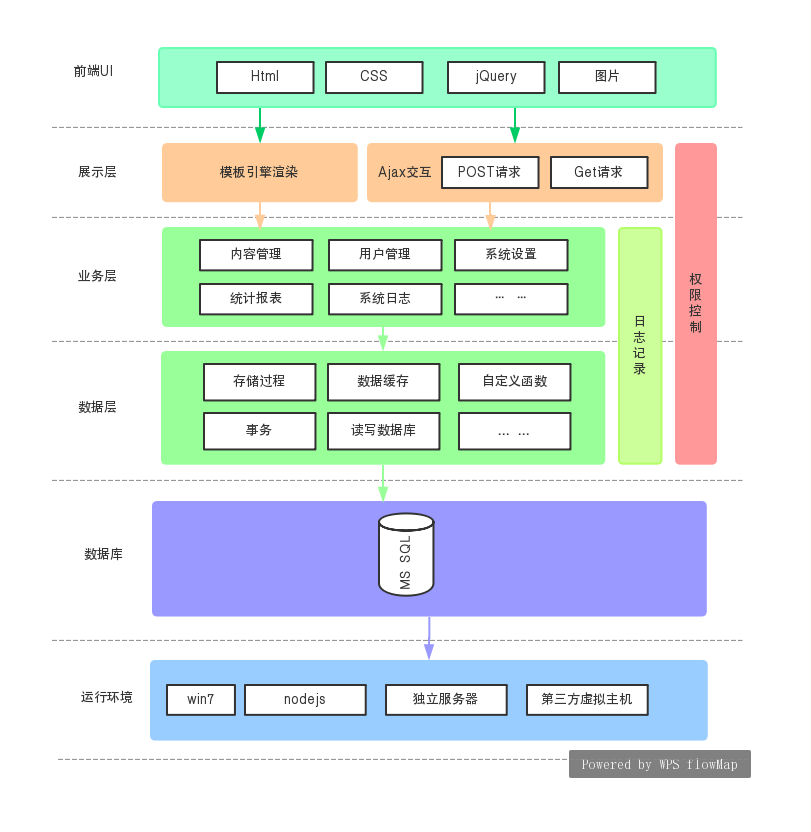
服务层架构

1 分布式消息。 低耦合。 更快的响应， 削峰填谷。

2 分布式服务。 SOA架构。

3 分布式缓存。

4 分布式配置。 配置修改实时推送。



4.2 系统总体结构

B/S体系结构

B/S（Browser/Server）结构，即浏览器服务器结构。它是随着Internet技术的兴起，对C/S结构的一种变化或者改进的结构。在这种结构下，用户工作界面通过浏览器来实现，极少部分[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E4%BD%93%E7%B3%BB%E7%BB%93%E6%9E%84/_blank)逻辑在前端（Browser）实现，主要事务逻辑在服务器端（Server）实现，形成所谓[三层结构](https://baike.baidu.com/item/%E4%B8%89%E5%B1%82%E7%BB%93%E6%9E%84" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E4%BD%93%E7%B3%BB%E7%BB%93%E6%9E%84/_blank)。这样就大大简化了客户端电脑载荷，减轻了系统维护与升级的成本和工作量，降低了用户的总体成本。

根据系统功能需求和主要[模块设计](https://baike.baidu.com/item/%E6%A8%A1%E5%9D%97%E8%AE%BE%E8%AE%A1" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E4%BD%93%E7%B3%BB%E7%BB%93%E6%9E%84/_blank)，系统用户较多，功能复杂，存储信息量大，需要专业技术人员维护和管理系统。在体系结构模式选择过程中，尽量立足于现有网络，在满足安全与稳定要求的同时，使管理维护操作简单，减少开发投入。

单用户体系结构不能满足本系统网络要求；C/S体系结构过于庞大，管理维护复杂；P2P体系结构虽然功能强大，但是本系统并不需要即时通讯和不间断的[数据更新](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E6%9B%B4%E6%96%B0" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E4%BD%93%E7%B3%BB%E7%BB%93%E6%9E%84/_blank)。为使用户能够在简单、易用、单一、统一的可视化界面下，轻松、方便地访问到各种类型的数据，系统采用B/S体系结构。

攀枝花农产品展销平台

前台系统

商品列表

商品属性

商品分类

商品管理

订单管理

修改密码

订单统计

超时未确认订单

待发货订单

所有订单列表

系统管理

首 页

分 类

购物车

商品浏览

查询

商品分类

按类别查询商品

结算购物车

查删购物车清单

图4.2.1 前台系统功能模块图

网上购物系统

后台管理系统

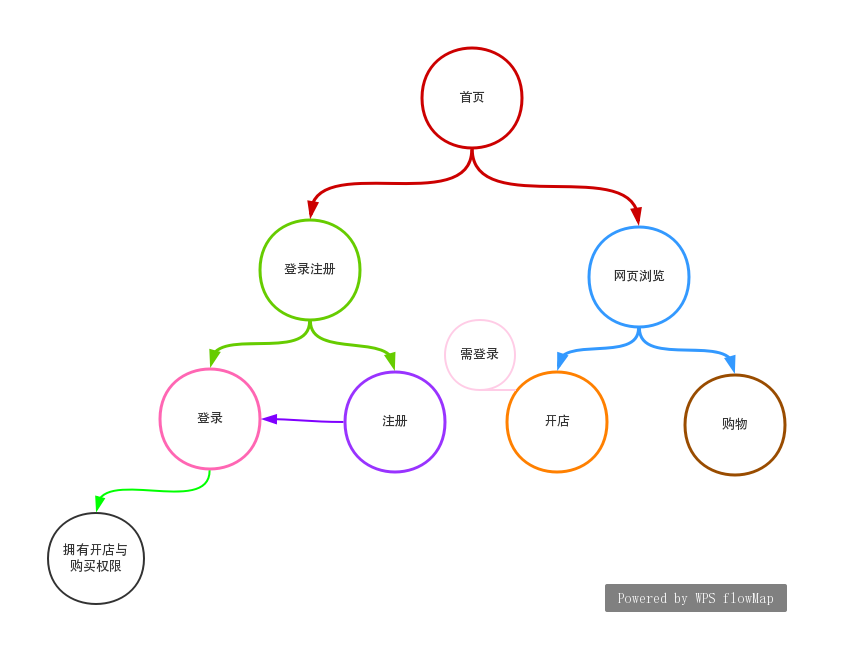
限购管理

商品限购

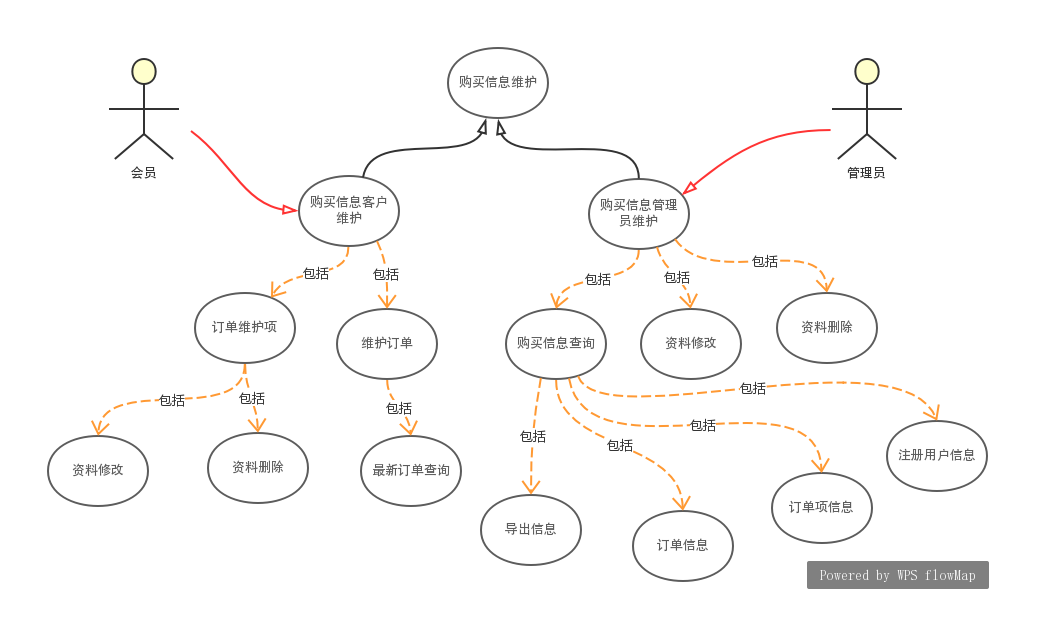
图4.2.2 后台系统功能模块图

4.3 系统业务流程

业务是指企业管理中必要且逻辑上相关的、为了完成某种管理功能的一系列相关的活动。在系统调研时，通过了解组织结构和业务功能，我们对系统的主要业务有了一个大概的认识。但由此我们得到的对业务的认识是静态的，是由组织部门映射到业务的。而实际的业务是流动的，我们称之为业务流程。一项完整的业务流程要涉及到多个部门和多项数据。例如，生产业务要涉及从采购到财务，到生产车间，到库存等多个部门;会产生从原料采购单，应收付账款，入库单等多项数据表单。因此，在考察一项业务时我们应将该业务一系列的活动即整个过程为考察对象，而不仅仅是某项单一的活动，这样才能实现对业务的全面认识。将一项业务处理过程中的每一个步骤用图形来表示，并把所有处理过程按一定的顺序都串起来就形成了业务流程图。



4.4 购物流程图



4.4 功能模块

功能模块是指数据说明、可执行语句等程序元素的集合，它是指单独命名的可通过名字来访问的过程、函数、[子程序](https://baike.baidu.com/item/%E5%AD%90%E7%A8%8B%E5%BA%8F/3941697" \t "https://baike.baidu.com/item/%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%9D%97/_blank)或宏调用。功能模块化是将程序划分成若干个功能模块，每个功能模块完成了一个子功能，再把这些功能模块总起来组成一个整体。以满足所要求的整个系统的功能。

4.4.1 商品管理

后台系统都是由管理员对商品进行增删改查的操作；对商品进行编辑，分类等操作，以便前台展示；

4.4.2 购物车管理

主要是对用户喜欢的商品做一个收藏，以便用户购买，同时还包含对商品数量和价格的统计等。

4.4.3 商城管理

每个用户可以创建一个自己的商店，在购物页面可显示每位用户的商店，并在各自的商店中展示店内的商品；

4.4.7 前台系统

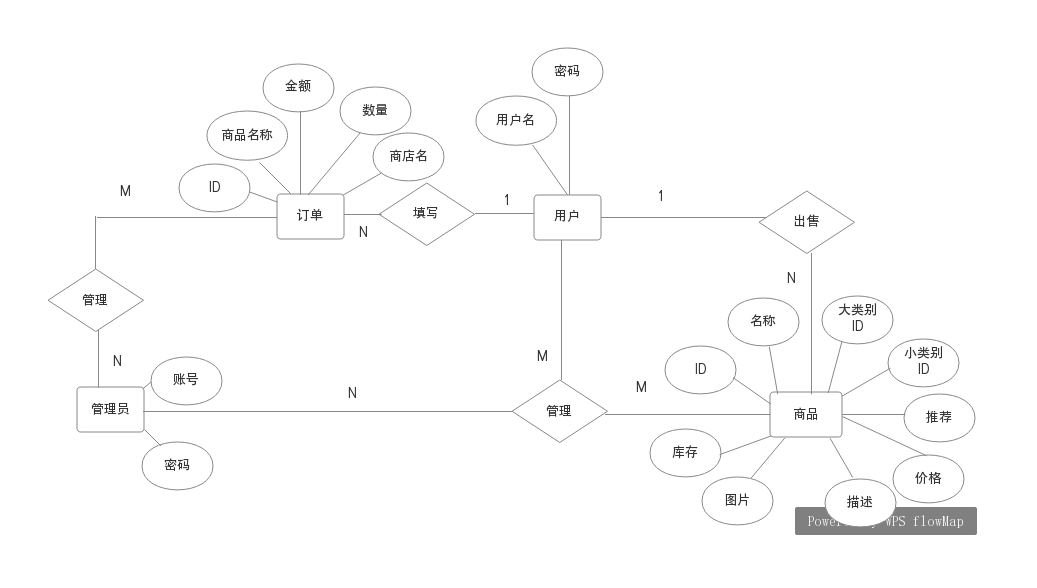
本系统属于pc端页面，用户可在电脑上输入对应的网页进入本页面，在本页面，用户可以观看商城页面，进入商店管理页面需要登录权限后方可访问。

4.5 数据库设计

本次农产品平台选中的数据库是使用开源免费的、最为广泛的MySQL，版本为5.7.25，采用编码格式为utf-8，collation格式为utf8\_general\_ci；

4.5.1 数据库概要设计

本系统开发核心为数据库的设计，它决定这本系统的运行方式，关系着系统最终的成败，下面是这次的数据库开发主要实体的ER模型图：



4.5.2 数据库逻辑设计

逻辑结构是独立于任何一种数据模型的，在实际应用中，一般所用的数据库环境已经?关系数据库，因此首先需要将实体-关系[图](https://baike.baidu.com/item/%E5%9B%BE" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%9A%84%E9%80%BB%E8%BE%91%E7%BB%93%E6%9E%84%E8%AE%BE%E8%AE%A1/_blank)转换为关系模型，然后根据具体数据库管理系统的特点和限制转换为指定数据库管理系统支持下数据模型，最后进行优化。下面是各表的详细设计：

1. 用户信息详情表：userinfo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 | 名称 |
| 用户名 | username | varchar | 255 | √ |  |  |
| 密码 | password | varchar | 255 |  |  |  |

1. 创建商店信息存储表：creatshop

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| 用户名 | username | varchar | 255 |  | √ |
| 商店名 | shopname | varcahr | 255 |  |  |

1. 商店商品详情表：trade

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| ID | Id | int | 10 | √ |  |
| username | username | varchar | 255 |  | √ |
| 商品名称 | tradename | varchar | 255 |  |  |
| 价格 | price | int | 10 |  |  |
| 商品信息 | tradeinfo | varchar | 255 |  |  |
| 商品图片 | tradeimg | varchar | 255 |  |  |
| 所属商店 | Shopname | varchar | 255 |  |  |

1. 购物车表：gouwuche

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| id | id | int | 10 | √ |  |
| username | username | varchar | 255 |  | √ |
| 商品名 | tradename | varchar | 255 |  |  |
| 价格 | price | int | 10 |  |  |
| 数量 | num | int | 10 |  |  |

1. 商户账号表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| 商人账号id， | id | int8 | 8 | √ |  |
| 商人id | mid | int8 | 8 |  | √ |
| 商人账号 | account | VARCHAR(128) | 128 |  |  |
| 密码 | password | VARCHAR(256) | 256 |  |  |
| 注册日期 | register\_date | TIMESTAMP |  |  |  |
| 是否有效 | is\_active | BOOL |  |  |  |
| 商户账号头像 | img\_url | VARCHAR(256) | 256 |  |  |

1. 首页商品种类表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| id,自增 | id | int8 | 8 | √ |  |
| 商品种类id | category\_id | int4 | 4 |  | √ |
| 动作类型编码 | action | int4 | 4 |  |  |
| 动作参数 | action\_parameter | VARCHAR(128) | 128 |  |  |
| 指定图像 | img\_url | VARCHAR(256) | 256 |  |  |
| 商品名称 | goods\_name | VARCHAR(256) | 256 |  |  |
| 商品价格 | goods\_price | DECIMAL |  |  |  |
| 商品文字描述 | goods\_brief | VARCHAR(256) | 256 |  |  |
| 商品显示排序 | goods\_order | int4 | 4 |  |  |
| 是否删除 | is\_delete | BOOL |  |  |  |

1. 商品图片表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| ID，自增 | id | int8 | 8 | √ |  |
| 所属商品ID | gid | int8 | 8 |  | √ |
| 图片类型 | type | int |  |  |  |
| 显示类型 | display\_type | int |  |  |  |
| 图片排序 | img\_order | int4 | 4 |  |  |
| 图片地址 | url | varchar(255) | 255 |  |  |
| 外部ID | ext\_id | varchar(128) | 128 |  |  |

1. 商品扩展描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 数据类型 | 长度 | 主键 | 外键 |
| ID | id | int8 | 8 | √ |  |
| 所属商品ID | gid | int8 | 8 |  | √ |
| 类型 | type | int |  |  |  |
| 内容 | content | text |  |  |  |
| 外部ID | ext\_id | varchar(128) | 128 |  |  |

4.5.3 数据库物理模型

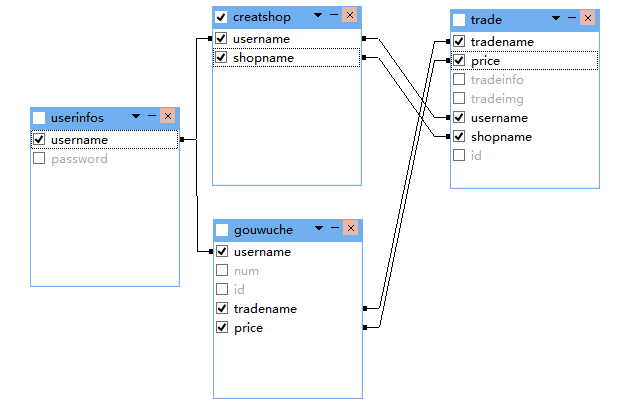
物理数据模型（Physical Data Model，[PDM](https://baike.baidu.com/item/PDM/312565" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)），提供了系统初始设计所需要的基础元素，以及相关元素之间的关系。即用于存储结构和访问机制的更高层描述，描述数据是如何在计算机中存储的，如何表达记录结构、记录顺序和访问路径等[信息](https://baike.baidu.com/item/%E4%BF%A1%E6%81%AF/111163" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)。

物理数据模型（Physical Data Model）[PDM](https://baike.baidu.com/item/PDM/312565" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)，提供了系统初始设计所需要的基础元素，以及相关元素之间的关系。即用于存储结构和访问机制的更高层描述，描述数据是如何在计算机中存储的，如何表达记录结构、记录顺序和访问路径等信息。使用物理数据模型，可以在系统层实现数据库。数据库的物理设计阶段必须在此基础上进行详细的后台设计，包括数据库的[存储过程](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E8%BF%87%E7%A8%8B/1240317" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)、操作、触发、视图和索引表等；至2013年，还没有太多的物理数据模型。

物理模型优点如下：

除了提供[数据库结构](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%BB%93%E6%9E%84/5507713" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)的可视化提取，定义物理数据模型的一个重要的好处是您可以自动从模型中导出数据库的架构。

这是可能因为物理数据模型捕获的[元数据](https://baike.baidu.com/item/%E5%85%83%E6%95%B0%E6%8D%AE/1946090" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)非常丰富，并且它密切映射数据库架构，如表、 列、 主键和[外键](https://baike.baidu.com/item/%E5%A4%96%E9%94%AE/1232333" \t "https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B/_blank)等方面。

我的物理数据模型如下：

5 系统详细设计

## 5.1 功能调用流程

1. 前端页面通过ajax想后端发送数据；

2. 后端通过body或者URL接受前端发送过来的数据，并处理这些数据发送给数据库；

3. 数据库响应这些数据，并返回响应的数据给后台

4. 后台把数据发送给前端；

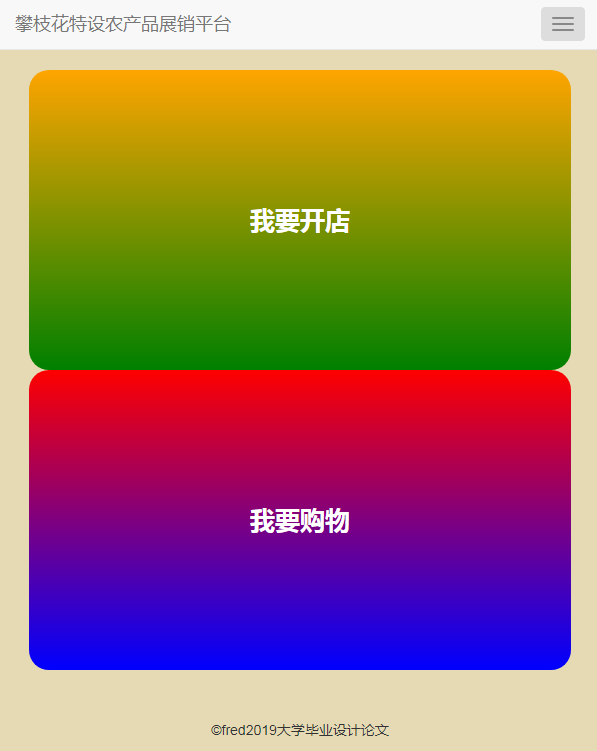
5. 前端把数据渲染到页面。

## 5.2 前台详细设计

5.2.1 前台首页



使用bootstrap搭建的一个响应式网页：他会根据屏幕宽度，改变页面的结构，可以适配pc端和手机端，而不需要在写一套需要适配手机端的css样式表：





关键代码：

<link href="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/js/bootstrap.min.js"></script>

点击“我要购物”可进入商城页面，浏览每个人创建的商店：



这里是有一个商店示例；

在进入“我要开店”时需要登录后才会授权：一个简单的权限设计，主要是把数据先存在localstorage中，在通过js查看是否有该数据，如果有就会授权若没有就会提示登录，按确认按钮会跳转到登录页面：



关键代码：

if (localStorage.getItem('username') == null) {

$(".myshop").on('click', function () {

if (confirm('亲，需要登录哦！请问是否需要登录')) {

window.location.href = '/login'

}

})

$('.username').on("click", function () {

window.location.href = '/login'

})document.getElementsByClassName("username")[0].innerHTML = document.getElementsByClassName("username")[0].innerHTML

$('.quit1').hide();

} else {

$(".myshop").on('click', function () {

window.location.href = '/myshop.html';

});

$('.quit2').hide();

document.getElementsByClassName("username")[0].innerHTML = document.getElementsByClassName("username")[0].innerHTML + ',' + localStorage.getItem('username');

}

在登录后会在右上角显示用户名：在登录页面登录后跳转至首页的会在左上角显示用户名，否则会显示登录按钮，提醒用户登录



用户名是在登录获取用户名存储在浏览器的localstorage中

关键代码：

if (window.location.pathname == '/') {

localStorage.clear();

} else {

$.get('/username', function (data, stat) {

console.log(data);

localStorage.setItem("username", data);

});

}

$(".myshop").on('click', function () {

window.location.href = '/myshop.html';

});

$('.quit2').hide();

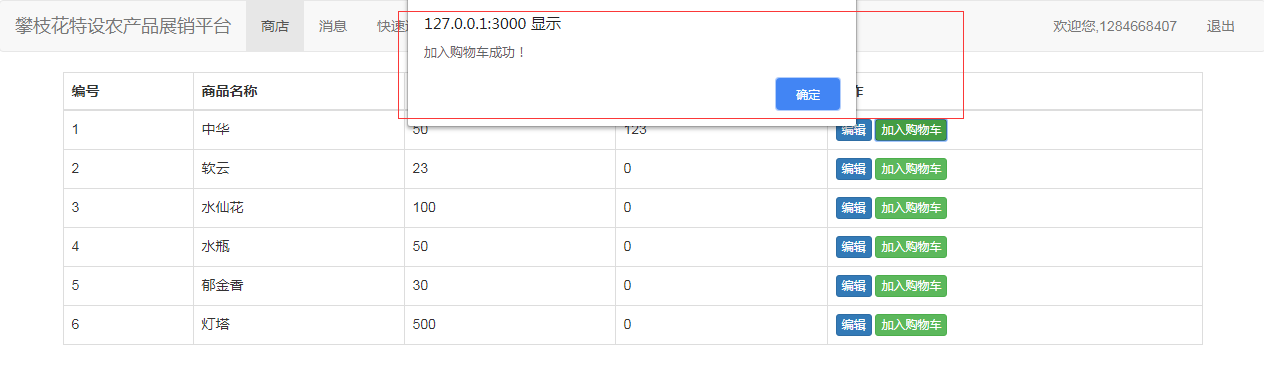
document.getElementsByClassName("username")[0].innerHTML = document.getElementsByClassName("username")[0].innerHTML + ',' + localStorage.getItem('username');

5.2.2 商城商店详情页面

商店详情页的商品使用table格式显示渲染出来的；每行商品后都会有个编辑按钮和加入购物车按钮；编辑按钮可以改变商品数量；加入购物车后，会在自己的购物车中看到相关商品；







相关代码：

$.ajax({

url: "/showtrade",

method: "post",

data: {

username: GetRequest().username

}

}).then(function (res) {

console.log(res);

res.forEach(function (re, idx) {

var html = `<tr data-ids='${re.id}'>

<td>${idx + 1}</td>

<td>${re.tradename}</td>

<td>${re.price}</td>

<td><span>0</span><input type="number"></td>

<td>

<button class="btn btn-primary btn-xs btn-edit" id="btn-edit">编辑</button>

<button class="btn btn-xs btn-success btn-gouwu"" id="btn-gouwu">加入购物车</button>

<button class="btn btn-success btn-xs btn-ok" id="btn-ok">确定</button>

<button class="btn btn-warning btn-xs btn-cancel" id="btn-cancel">取消</button>

</td>

</tr>`;

$("tbody").append(html);

})

})

$('tbody').on('click', function (e) {

e = e || window.e;

var target = e.target || e.srcElement;

var tr = target.parentNode.parentNode;

switch (target.id) {

case "btn-edit": window.edit(tr);

break;

case "btn-cancel": window.cancel(tr);

break;

case "btn-ok": window.ok(tr);

break;

case "btn-gouwu": window.gouwu(tr);

break;

default:

break;

}

})

function edit(tr) {

tr.classList.add('edit');

$(tr).find("span").each(function (ind, ele) {

if ($(this).text() == 0) {

$(this).next().val("")

} else {

$(this).next().val($(this).text());

}

})

};

function ok(tr) {

tr.classList.remove('edit');

var ids = tr.dataset.ids;

$(tr).find("input").each(function (idx, val) {

if ($(this).val() == "") {

$(tr).find("input").prev().text("0");

} else {

$(tr).find("input").prev().text($(this).val());

}

})

}

function cancel(tr) {

tr.classList.remove('edit')

}

function gouwu(tr) {

console.log($(tr).children()[1].innerHTML,

$(tr).children()[2].innerHTML,

$($(tr).children()[3]).children()[0].innerHTML);

$.ajax({

url: "/gouwuche",

method: "post",

data: {

tradename: $(tr).children()[1].innerHTML,

price: $(tr).children()[2].innerHTML,

num: $($(tr).children()[3]).children()[0].innerHTML,

username: localStorage.getItem("username")

}

}).then(function () {

alert("加入购物车成功！")

})

}

5.2.3 购物车页面

购物车模块是给用户喜欢的商品但有暂时不想买的可以加入购物车，方便以后想买的时候再来买。

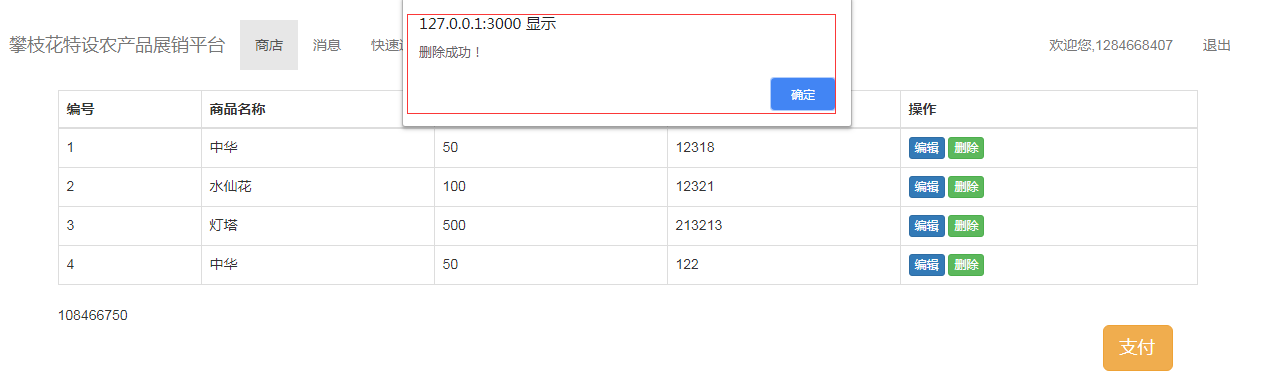
将商品加入购物车是在商店详情页面加入是点击加入购物车向后端发送ajax请求，再有后端处理数据存入数据库中，最后在购物车页面通过ajax获取请求数据，从而加入购物车中。



在点击编辑的时候，会在数量一列出现输入框，可以改变需要的商品的数量：



点击删除可以删除自己不想要的商品：



在表格下方左边是商品总价，右边是支付按钮；

相关代码：

$.ajax({

url: "/gouwuche\_select",

method: "get",

data: {

username: localStorage.getItem("username")

}

}).then(function (res) {

var totalMoney = 0;

res.forEach(function (re, idx) {

var html = `<tr data-ids='${re.id}'>

<td>${idx + 1}</td>

<td>${re.tradename}</td>

<td>${re.price}</td>

<td><span>${re.num}</span><input type="number"></td>

<td>

<button class="btn btn-primary btn-xs btn-edit" id="btn-edit">编辑</button>

<button class="btn btn-xs btn-success btn-gouwu"" id="btn-del">删除</button>

<button class="btn btn-success btn-xs btn-ok" id="btn-ok">确定</button>

<button class="btn btn-warning btn-xs btn-cancel" id="btn-cancel">取消</button>

</td>

</tr>`;

$("tbody").append(html);

totalMoney += (re.price \* re.num);

})

$(".total").html(totalMoney);

})

$('tbody').on('click', function (e) {

e = e || window.e;

var target = e.target || e.srcElement;

var tr = target.parentNode.parentNode;

switch (target.id) {

case "btn-edit": window.edit(tr);

break;

case "btn-cancel": window.cancel(tr);

break;

case "btn-ok": window.ok(tr);

break;

case "btn-del": window.del(tr);

break;

default:

break;

}

return false;

})

function edit(tr) {

tr.classList.add('edit');

$(tr).find("span").each(function (ind, ele) {

if ($(this).text() == 0) {

$(this).next().val("")

} else {

$(this).next().val($(this).text());

}

})

};

function ok(tr) {

tr.classList.remove('edit');

var ids = tr.dataset.ids;

var num;

$(tr).find("input").each(function (idx, val) {

num = $(this).val();

if ($(this).val() == "") {

$(tr).find("input").prev().text("0");

} else {

$(tr).find("input").prev().text($(this).val());

}

})

$.ajax({

url: "/gouwuche\_ok",

method: "get",

data: {

num: num,

id: ids

}

})

}

function cancel(tr) {

tr.classList.remove('edit')

}

function del(tr) {

var ids = tr.dataset.ids;

$(tr).remove();

$.ajax({

url: "/gouwuche\_del",

method: "post",

data: {

id: ids

}

}).then(function (res) {

if (res.res\_code == 1) {

alert("删除成功！")

}

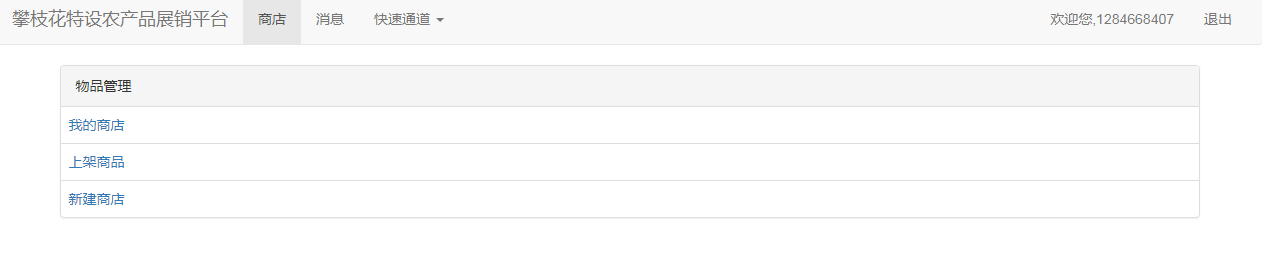
})

}

## 5.3 后台详细设计

5.3.1 后台整体设计

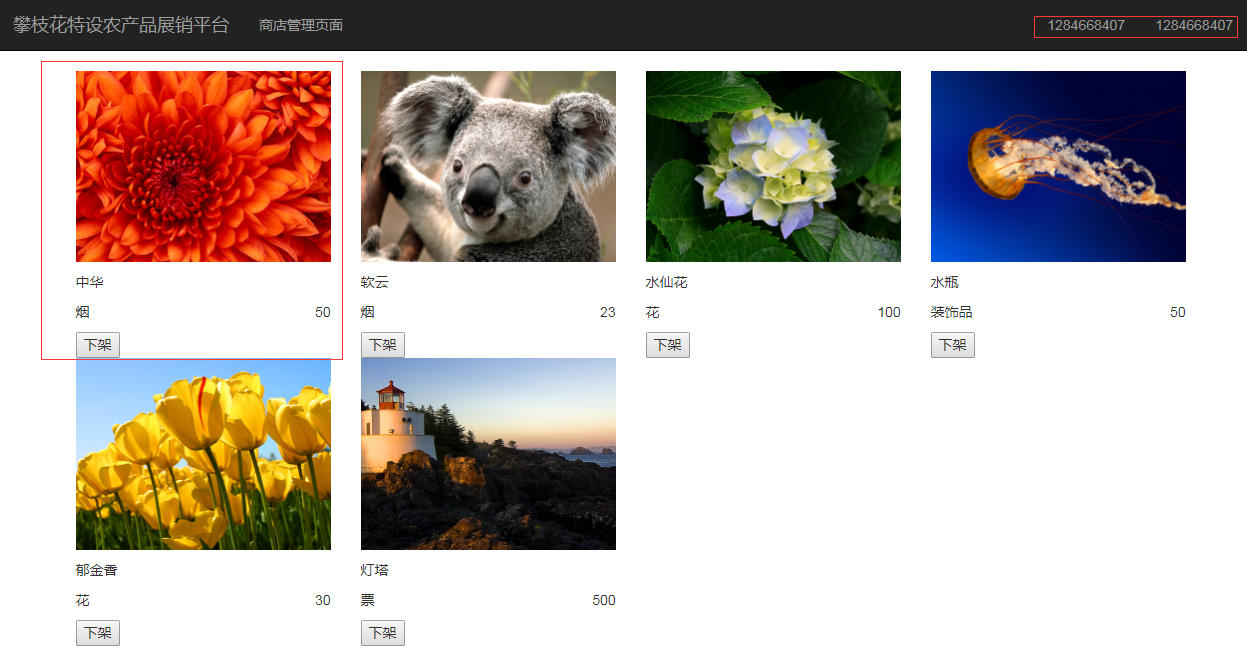
后台模块主要分为3个板块：商品管理（下架），商品上架、创建商店；这三个为最基础也是最重要的三个模块；使用bootstrap ui搭建的模块；



为方便管理员操作；每个模块分为一行，简洁明了

5.3.2 商品管理

在我的商店里是用户商店所有商品的展示，可对商品进行管理：



右上角是用户名和用户的商店的名字；每个商品下都有一个下架按钮，可对商品进行删除操作；

相关代码：

var mysql = require('mysql');

var connection = mysql.createConnection({

host: 'localhost',

user: 'root',

password: '123456',

// port: '3306',

database: 'mydatabase'

});

var connectiond = connection;

// 查询所有数据

var selectAll = function (sql, callback) {

connectiond.query(sql, function (err, result) {

if (err) {

console.log("错误信息：", err.sqlMessage);

let errNews = err.sqlMessage;

callback(errNews, 'errnews');

return;

}

var string = JSON.stringify(result);

var data = JSON.parse(string);

callback('data', data);

})

}

// 插入一条数据

var insertData = function (table, datas, callback) {

var fields = '';

var values = '';

for (var k in datas) {

fields += k + ',';

values = values + "'" + datas[k] + "',";

}

fields = fields.slice(0, -1);

values = values.slice(0, -1);

console.log(fields, values);

var sql = "INSERT INTO " + table + ' (' + fields + ') VALUES (' + values + ')';

connectiond.query(sql, callback);

}

// 更新数据

var updateData = function (table, sets, where, callback) {

var \_SETS = '';

var \_where = '';

var keys = '';

var values = '';

for (var k in sets) {

\_SETS += k + "='" + sets[k] + "',";

}

\_SETS = \_SETS.slice(0, -1);

for (var k2 in where) {

\_where += k2 + "=" + where[k2];

}

var sql = "UPDATE " + table + " SET " + \_SETS + " WHERE " + \_where;

connectiond.query(sql, callback);

}

// 删除数据

var deleteData = function (table, where, callback) {

var \_WHERE = '';

for (var k2 in where) {

// 多个筛选条件 \_WHERE += k2 + "=" + where[k2] + "AND"

\_WHERE += k2 + "=" + where[k2]

}

var sql = "DELETE FROM " + table + " WHERE " + \_WHERE;

connectiond.query(sql, callback);

}

module.exports.selectAll = selectAll;

module.exports.insertData = insertData;

module.exports.updateData = updateData;

module.exports.deleteData = deleteData;

// 下架商品

app.post('/xiajia', function (req, res) {

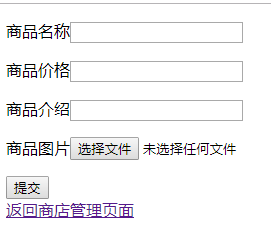
var deleteID = { id: req.body.id };

mysql\_way.deleteData("trade", deleteID, function (e, r) { })

})

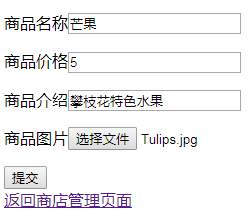
5.3.3 商品上架管理

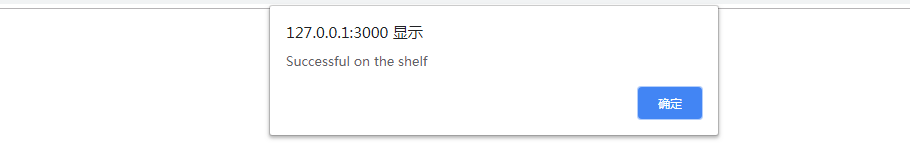
在上架页面可对自己的商店进行上架商品操作：



本商品有上传文件功能；提交成功后会有提示成功的信息，并会在商店管理页面显示出来

上传成功后返回的数据：





关键代码：

app.post("/shangjia", function (req, res, next) {

// 设置保存路径

var form = new formidable.IncomingForm();

//设置编辑

form.encoding = 'utf-8';

//设置文件存储路径

form.uploadDir = "./upload/";

//保留后缀

form.keepExtensions = true;

//设置单文件大小限制

// form.maxFieldsSize = 2 \* 1024 \* 1024;

form.parse(req, (err, fields, files) => {

if (err) {

throw err;

}

var t = (new Date()).getTime();

//生成随机数

var ran = parseInt(Math.random() \* 8999 + 10000);

//拿到扩展名

var extname = path.extname(files.tradeimg.name);

//path.normalize('./path//upload/data/../file/./123.jpg'); 规范格式文件名

var oldpath = path.normalize(files.tradeimg.path);

//新的路径

let newfilename = t + ran + extname;

var newpath = "./upload/" + fields.username + '/' + newfilename;

var saveData = {

'tradename': fields.tradename,

'price': fields.tradeprice,

'tradeinfo': fields.tradeinfo,

'tradeimg': newpath,

'username': fields.username,

'shopname': fields.shopname

};

// form.uploadDir = \_\_dirname + "/upload/" + saveData.username;

fs.exists("./upload/" + saveData.username, function (exists) {

if (!exists && saveData.username != undefined) {

fs.mkdir('./upload/' + saveData.username, { recursive: true }, function (err) {

if (err) {

return console.error(err, "目录新建失败");

}

console.log("目录创建成功。");

});

} else {

return;

}

});

mysql\_way.insertData('trade', saveData, function (e, r) {

if (e) {

res.status(200).json({ "status": false, "msg": e, "data": [] });

}

res.end("<script>alert('Successful on the shelf')</script>")

})

fs.rename(oldpath, newpath, function (err) {

if (err) {

console.error("改名失败" + err);

}

});

res.setHeader("refresh", "0;/shangjia.html");

});

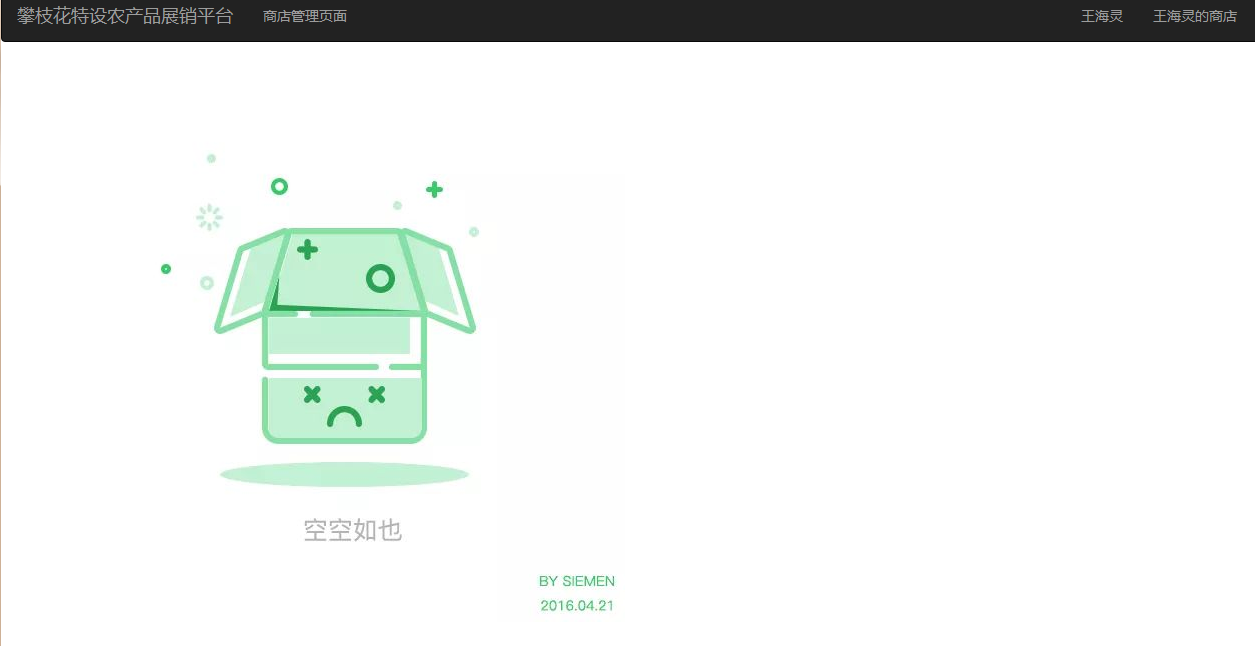
})

5.3.4 商店管理

在商店创建模块中，用户可以创建一个自己喜欢的名字，但是没有修改功能，所以在创建的时候要想好商店名称哦，下面是商店创建页面：



在创建完成后，商店默认没有商品时页面如下：



app.post("/creatshop", function (req, res) {

var username = req.body.username;

var shopname = req.body.shopname;

console.log(username, shopname, req.body)

var istrue = false;

connection.query("SELECT \* FROM creatshop", function (err, result) {

if (err) {

return;

}

for (var i = 0; i < result.length; i++) {

if (result[i].username == username) {

data.shopname = shopname;

console.log(data, '157')

app.get('/shop', function (req, res) {

res.json(data);

})

istrue = true;

break;

}

}

if (istrue) {

fs.readFile("./public/creatshop.html", function (err, data) {

if (err) {

throw err;

}

res.setHeader("refresh", "3;/myshop.html");

res.end("<h1>Creat shop error, maybe you have your own shop!</h1>");

})

} else {

var addSql = 'INSERT INTO creatshop(shopname,username) VALUES(?,?)';

var addSqlParams = [shopname, username];

//增

connection.query(addSql, addSqlParams, function (err, result) {

if (err) {

console.log('[INSERT ERROR] - ', err.message);

return;

}

console.log('--------------------------INSERT----------------------------');

//console.log('INSERT ID:',result.insertId);

console.log('INSERT ID:', result);

console.log('-----------------------------------------------------------------\n\n');

});

res.setHeader("refresh", "0;/myshop.html");

res.end();

}

})

});

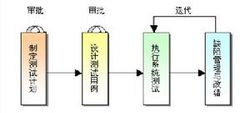
6 系统测试

6.1 测试目的

系统测试，英文是System Testing。是对整个系统的测试，将硬件、软件、操作人员看作一个整体，检验它是否有不符合系统说明书的地方。这种测试可以发现系统分析和设计中的错误。如安全测试是测试安全措施是否完善，能不能保证系统不受非法侵入。再例如，压力测试是测试系统在正常数据量以及超负荷量(如多个用户同时存取) 等情况下是否还能正常地工作。

系统测试是将经过[集成测试](https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E6%B5%8B%E8%AF%95" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E6%B5%8B%E8%AF%95/_blank)的软件，作为计算机系统的一个部分，与系统中其他部分结合起来，在实际运行环境下对计算机系统进行的一系列严格有效地测试，以发现软件潜在的问题，保证系统的正常运行。

流程如图1所示。系统测试的目的是验证最终软件系统是否满足用户规定的需求。

[](https://baike.baidu.com/pic/ç³»ç»æµè¯/3073399/0/966aca074e98b48c7b894780?fr=lemma%26ct=single)图1 系统测试流程图

主要内容包括：

·[功能测试](https://baike.baidu.com/item/%E5%8A%9F%E8%83%BD%E6%B5%8B%E8%AF%95" \t "https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E6%B5%8B%E8%AF%95/_blank)。即测试软件系统的功能是否正确，其依据是需求文档，如《产品需求规格说明书》。由于正确性是软件最重要的质量因素，所以功能测试必不可少。

·健壮性测试。即测试软件系统在异常情况下能否正常运行的能力。健壮性有两层含义：一是容错能力，二是恢复能力。

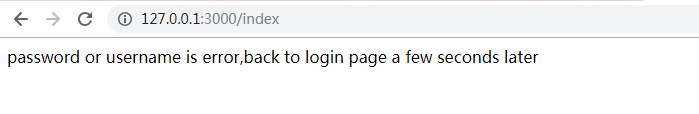
6.2 前台测试

6.2.1 登录测试

在登陆页面使用密码账号不匹配是，不会进入首页，也不会出现用户名：



失败后会出现如下页面，并在1s后返回登录页面：



6.2.2 注册测试

查看在用户注册后是否可以使用账号和密码：

在注册完成后会自动跳转到登录页面：







6.2.2 购物车测试



在购物页面加入商品；

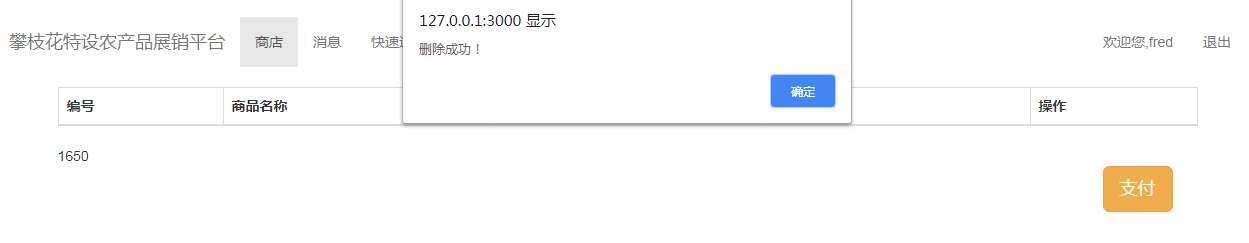


在购物车中会出现这个商品；

并且可以改变这个商品的数量：



还可以删除这个商品：



6.3 后台测试

6.3.1 我的商店测试

在商店没有创建时是否可以查看我的商店的页面测试：

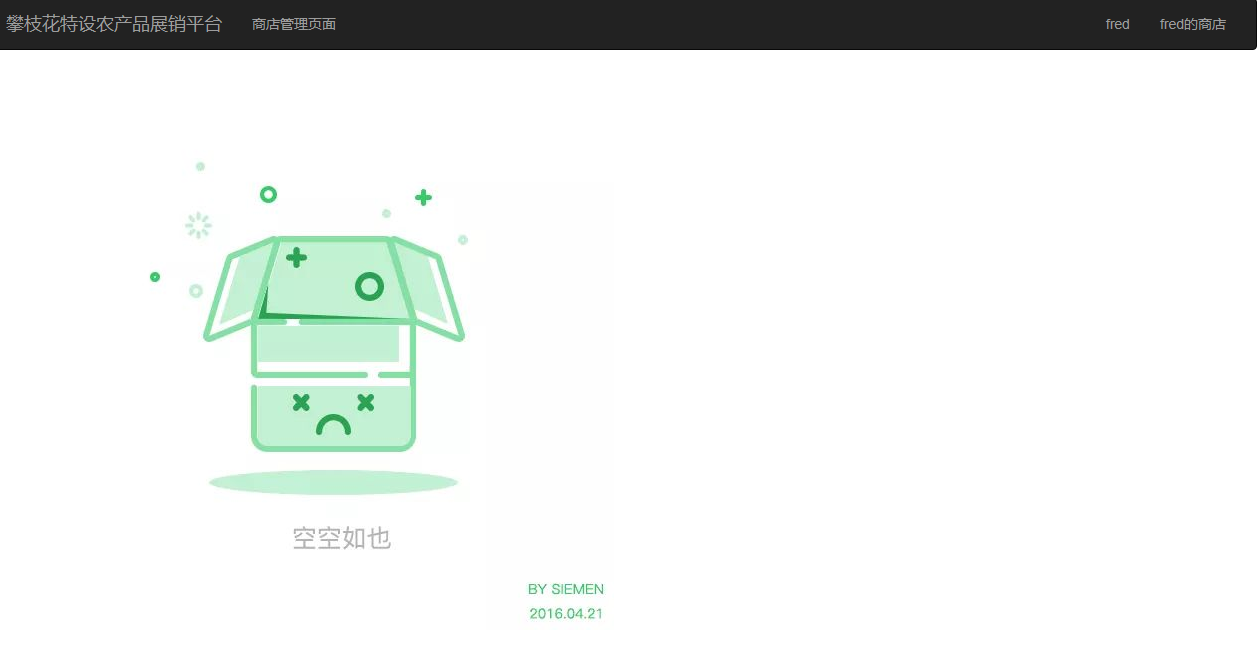


在没有商店时需要先创建商店：



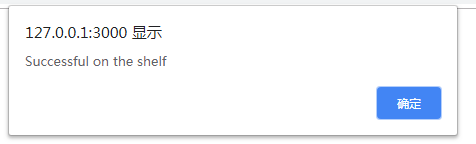
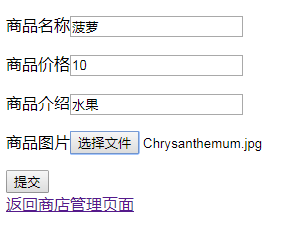
可以自己给商店去想要的名字。

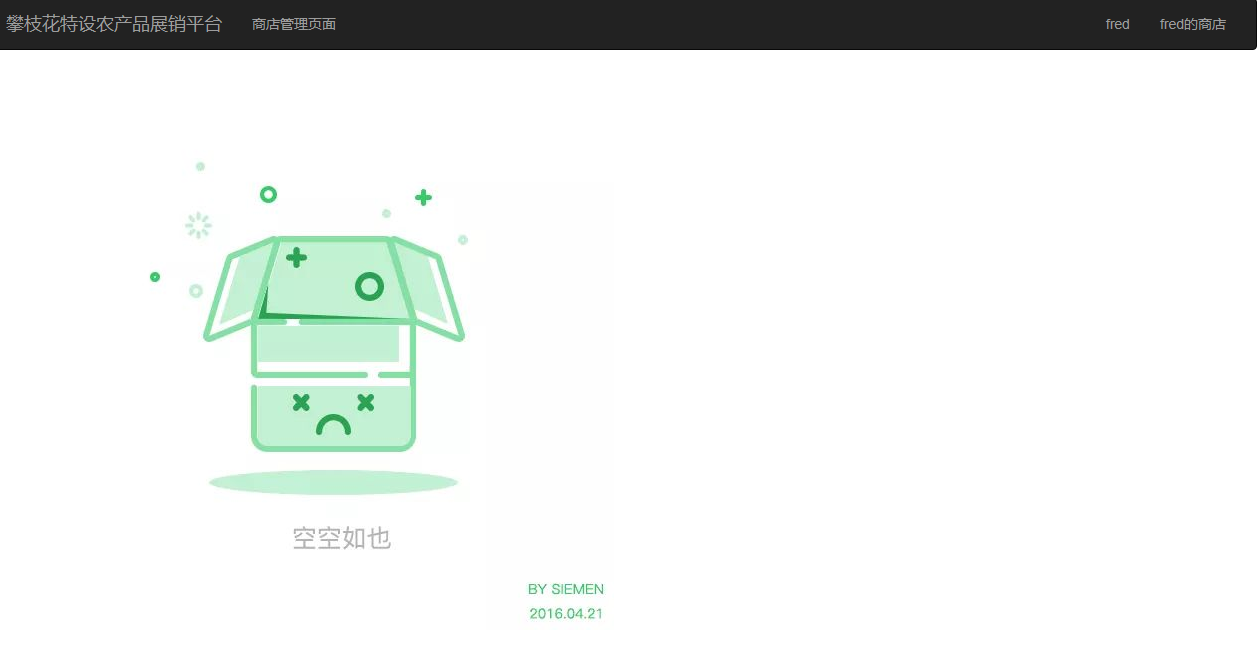
创建商店后可以进入我的商店页面：



6.3.2 上架下家商品测试

在上架页面中，可以上架自己拥有的商品，以供客户购买：



在创建后可以在自己的商店列表中看到新上架的商品：



创建商品后每个商品下方都有一个下架按钮，可以下架商品：

6.4 测试结论

所测试的所用功能均能正常运行，没有发现有重大bug，功能略显粗糙，基本功能较为完善，不过有些地方的提示信息比较少，缺乏文案，某些细节可能做得不是太好。

7 结论

互联网的推动，农产品电商逐步渗入到农村以及农民的生活，使他们接触到了网络。从起初对网络的陌生到慢慢地熟悉，农产品经营者们也逐渐意识到了农产品网上交易平台建设的重要性，以及其可以带来的盈利效果。

农产品网上交易平台集提供信息、质检、交易、结算和运输等功能于一体。具有许多优势：

（1）销售模式增加：从农产品传统销售模式向多渠道发展，并提供零售和批发的销售模式，使农产品销售到全国各地，扩大了销售范围。

（2）线上线下结合，增加销售额：农产品网上交易平台的建设，意味着农产品可以借助互联网在网上进行交易，除此以外，农产品经营者亦可以在线下就行销售，线上线下结合，从而增加了农产品的销售额。

（3）迅速售卖农产品，减少农产品滞留浪费问题：不同的季节有不同的农产品生产，因此需要在一定季节里把农产品销售出去，避免滞留浪费问题。

（4）借助大数据平台，抓取消费者的喜好：农产品在网上交易后，大数据平台可以抓取有效信息，对消费者的喜好进行分析，从而有助于经营者对农产品电商平台的价格等因素进行调整，促使消费者购买。

农产品网上交易平台建设可以推动农产品电商平台的发展，促使传统农业向互联网转型，使其更加信息化、标准化和品牌化，并且促进特色农产品得到发展。

参 考 文 献

1. 解颐，方红亮，曲珍，余涵，王学昌。 基于HTML5脚本的旅游软件开发[J]. 电脑编程技巧与维护，2017,（07）：35-37.  
   [2]王燕枫，陈高，王冬。 基于Hadoop和HTML5的环保WebGIS系统架构优化设计与实现[J]. 中国市场，2017,（09）：29-30+42.  
   [3]马玉娴。 HTML5技术在融媒体新闻中的应用[J]. 新闻研究导刊，2017,（06）：281-282.  
   [4]陈伟江。 基于HTML5的3D数据可视化[J]. 电子世界，2017,（06）：165.
2. 解颐，方红亮，曲珍，余涵，王学昌。 基于HTML5脚本的旅游软件开发[J]. 电脑编程技巧与维护，2017,（07）：35-37.  
   [6]王燕枫，陈高，王冬。 基于Hadoop和HTML5的环保WebGIS系统架构优化设计与实现[J]. 中国市场，2017,（09）：29-30+42.  
   [7]马玉娴。 HTML5技术在融媒体新闻中的应用[J]. 新闻研究导刊，2017,（06）：281-282.  
   [8]陈伟江。 基于HTML5的3D数据可视化[J]. 电子世界，2017,（06）：165.

[9]LuisAtencio.[FunctionalProgramminginJavaScript](https://book.douban.com/subject/26829118/" \t "https://www.douban.com/doulist/45562340/_blank).ManningPublications,2016-6-18

[10]死月。Nodejs：来一打c++扩展.电子工业出版社，2018，（06）：152-153

[11]赫尔曼。Effective JavaScript.机械工业出版社，2014，（01）：10-11

[12]John Resign。JavaScript忍者秘籍.人民邮电出版社，2015，（10）：224

[13]Kyle Simpson。你不知道的JavaScript.人民邮电出版社，2015，（04）：112-113

[14]Nicholas。高性能JavaScript.电子工业出版社，2010，（11）：236-237

[15]Nicholas。JavaScript高级程序设计.人民邮电出版社，2012，（03）：101-102

[16]Douglas。JavaScript语言精粹.电子工业出版社，2009，（04）：94-95

[17]Addy。Learning JavaScript Design Patterns.O'Reilly Media，2012，（08）：133

[18]特谢拉。Nodejs高级编程.清华大学出版社，2013，（05）：132-133

[19]朴灵。深入浅出的nodejs.人民邮电出版社，2013，（03）：66-67

[20]大卫 冈萨雷斯。Nodejs微服务.电子工业出版社，2017，（03）：65-67

[21]马里奥 卡西罗。Nodejs设计模式.电子工业出版社，2018，（02）：33-34

致谢

四年的美好大学时光转瞬即逝，依稀记得刚来学校时学长和老师的热情洋溢、同学们的青涩脸庞；依稀昨日记得操场上的奔跑，那是我逝去的青春；依稀记得老师们在讲台上的淳淳教育、激情洋溢地带个大家知识，和我们度过了一个美好且多彩的大学生活。在这里我非常感谢学校里的每一位老师，你们才是学校的顶梁柱，而我们只是你们这里的以为匆匆过客罢了，大学四年说长不长，说短也不短，也许在你们心里只是这一届走了，下一届学生又来了，而对于我们来说这是一段过了就在也不会有的时光了。在这里我非常感谢老师们的默默付出，也非常感谢我的指导老师吕俊瑞，感谢您的细心、感谢您的耐心、感谢您的关心。