**湖 北 大 学**

**本 科 毕 业 论 文 （设 计）**

**题 目 校园网上购物商城系统的开发与研究**

**姓 名 吴亚妹 学 号 2014221119300027**

**专业年级 电子信息工程2014级**

**指导教师 潘永才**

**职 称 教授**

**二○ 年 月 日**

目 录

第一章 绪论………………………………………………………………………（1）

1. 系统的分析及相关技术介绍……………………………………………（3）

2.1 需求分析…………………………………………………………………（3）

2.2 功能分析…………………………………………………………………（3）

2.3 HBuilder编译器…………………………………………………………（3）

2.4 MongoDB介绍……………………………………………………………（3）

2.5 Node JS……………………………………………………………………（3）

第三章 数据库设计………………………………………………………………（3）

3.1 数据概念模型设计………………………………………………………（3）

3.2 数据逻辑模型设计………………………………………………………（3）

3.3 数据物理模型设计………………………………………………………（3）

3.4 数据库创建与连接………………………………………………………（3）

3.4.1 外部路由设置……………………………………………………（3）

3.4.2 连接和创建数据库………………………………………………（3）

第四章 系统功能模块设计………………………………………………………（3）

4.1 系统功能模块设计………………………………………………………（3）

4.2 系统功能详细介绍………………………………………………………（3）

4.2.1 用户功能介绍……………………………………………………（3）

4.2.2 管理员附加用户功能介绍………………………………………（3）

第五章 系统功能测试与分析……………………………………………………（3）

5.1 首页功能测试……………………………………………………………（3）

5.2 购物车页面………………………………………………………………（3）

参考文献……………………………………………………………………………（8）

附录…………………………………………………………………………………（8）

致谢…………………………………………………………………………………（8）

校园网上购物商城系统的设计和开发

摘 要

近两年来，Internet的高速发展，互联网已经成为信息通信最重要的途径之一。足不出户便解决生活所有的问题已经不足为奇，这都得益于互联网+迅速崛起，即利用互联网与传统行业的有效结合，提供更加方便快捷的服务。越来越多的企业商家在互联网上搭建起自己的网上商城，向消费者提供一种新颖的购物理念和方式。校园网上购物商城是具有信息交互功能的一种商贸系统，它借助互联网构建一个虚拟购物商城以往的购物系统多是C/S架构，代码集中在服务端和客户端，日常维护麻烦，成本较高，用户需要程序安装才能使用，并且将网上商城系统引入校园与校园实体超市结合的却是少之又少。

针对购物系统存在的日常维护困难，操作复杂等问题，在Web开发平台，用B/S架构，我研发了校园网上购物商城系统，与C/S相比，B/S系统的代码主要集中在服务端，极少部分的代码集中在客服端，大大降低了程序员的日常维护难度，同时B/S不需要程序安装，只需要有web浏览器即可，对用户的使用提供了更便捷的方式。同时将该系统与校园实体超市相结合，根据校园超市消费者（师生）购物时间集中、系数庞大等特点，使购物更高效更便捷。于消费者而言是省时省心，于商家而言可以降低成本，是增大销售量的集中销售。

系统的运行结果显示，我开发的校园网上购物商城系统，能实现管理员对用户和商品管理，如商品上架、订单查看等功能；用户的注册登录、商品加入购物车、添加和查看取货人信息、下订单等一系列网购的基本功能，达到了使用B/S架构搭建一个简易的校园网上商城系统与校园实体超市相结合的目的。

【关键词】互联网+ 购物系统 MongoDB B/S结构

**Design and development of shopping mall system on campus**

#### Abstract

Over the past two years, the Internet has developed rapidly, and the Internet has become one of the most important means of information communication. Never leave home will solve all problems in life are not surprising, this is all thanks to the Internet + rapid rise, the use of the Internet and the traditional industry effective union, provide more convenient service. More and more enterprise merchants have set up their own online shopping mall on the Internet, providing consumers with a novel shopping concept and method. Shopping mall on campus is a commercial system with information interactive function. It constructs a virtual shopping mall via the Internet. Previous shopping system is C/S structure, more focused on the server and the client code, maintenance trouble, cost is higher, users need to program installation to use, and the introduction of the online mall system into the campus, with the supermarket entity is very few.

For shopping system routine maintenance difficulties, complex operation and so on, in the Web development platform, using B/S structure, I developed a campus online shopping mall system, compared with the C/S, and B/S system mainly concentrated in the server code, rarely part of the code on the end of the service, greatly reduces the maintenance difficulty of programmers, B/S don't need to program installation at the same time, you just need to have a Web browser, the use of the user provides a more convenient way. At the same time, the system is combined with the campus physical supermarket, which makes shopping more efficient and convenient according to the characteristics of shopping time concentration and coefficient of the campus supermarket consumer (teachers or students). For consumers, it is time-saving and don't have to worry about it. In the case of merchants, it can reduce costs and increase sales volume.

The operating results of the system show that I developed the shopping mall system on campus, which can realize the functions of administrators on users and commodity management, such as product shelves and order viewing. Registered user login, goods add to cart, and view the pickup information, the basic function of order and a series of online shopping, achieved using B/S structure to build a simple campus online mall system combined with campus supermarket entity.

**【Key words】**Internet + Shopping system Mongo database B/S structure

第一章 绪论

随着互联网普及率的不断提高，互联网+迅速崛起，短时间内得到了极大的推动力，甚至成为人们生活中不可或缺的重要组成部分。互联网+，代表一种新的经济形态，即充分发挥互联网在生产要素配置中的优化和集成作用，将互联网的创新成果深度融合于经济社会各领域之中，提升实体经济的创新力和生产力，形成更广泛的以互联网为基础设施和实现工具的经济发展新形态。互联网+的发展，将网络与传统企业相结合，促进经济快速发展的同时，给人们的生活带来了诸多的惠利，比如网上购物就是最好的例子。

现如今，网上购物已经成为一种时尚潮流，小到几岁的小学生，大到几十岁的大爷大妈，也可以熟练地利用网上购物，给生活带来方便，越来越多的企业商家在互联网上搭建起自己的网上商城系统，向消费者提供一种新颖的购物理念和方式。网上购物系统作为企业对企业，企业对用户，用户对用户的电子商务平台，在商业活动中扮演者着重要的角色。据不完全统计，2017双11全网销售额达2539.7亿。

网上购物商城发展趋近成熟，但是将网购商城引入校园却是比较少的，学校的消费群体主要是老师和学生，由于上课作息时间的一致性，具有购物时间集中、系数庞大等特点。故网上商城系统的引入是十分的有必要的，还能与师生的校园个人信息相结合，实现更加便捷有效的文明购物。

在日本、美国等信息化程度较高的国家和地区，网上商店发展速度迅猛，美国的世界级超一流的零售商，例如凯玛特、沃尔玛、家庭仓储等纷纷跻身于网络经商的行列。有调查显示：美国的许多家庭已经越来越习惯于在家中从网上购物，但是网购商城系统入校园却是极少的。

在网购相对成熟的中国，将商城系统引入校园，比较有名的就是我们的59store。百度百科提供的数据，59store是一个专注于高校市场的生活服务平台。用户可以通过59store的PC端或者移动App，方便地选购自己需要的商品，他们会提供5分钟送货到寝室的快速配送服务，主要提供零食、饮品、打印等服务。这给处于相对封闭校园中的学生带来了巨大的便利。但是59store是一个对于大多数高校而言的，它并不能针对每一个学校的特殊情况量身定制，也不能与学校的实体超市相结合，同时它的商品种类也比较有限。再比如：2011年上海交通大学致力打造的“绿色爱心屋”慈善公益捐助平台，利用实体和网络双重体系接受捐赠、进行援助和重复利用资源，成为社会爱心力量和贫困学生之间的互助互信纽带。大家不仅可以随时随地了解爱心屋库存动态，还能通过网络预约收受货品的地点和时间，志愿者“快递员”就会上门服务。这是针对该校园而开发的校园系统，但不是网上购物系统。

本课题主要讨论校园网上购物系统的开发和研究。旨在使用B/S架构开发一个网上购物商城系统，并将该系统与校园实体超市相结合，如其他购物系统一样，它可以向用户提供信息资源，如商品种类、报价等，师生可以根据商品信息选择商品并下单，超市工作人员会根据订单提前帮客户打包好商品，师生可以凭取货码到超市提取并付款即可。网上订购的交易方式切实降低了商家的时间、空间、金钱等成本，也切实实现了更好为师生服务这一宗旨，还可以将商城系统与师生个人信息系统相结合，实现更加安全便捷的服务。系统后期完善还可以实现网上支付、快递上门等服务。

第二章 系统的分析及相关技术介绍

2.1 需求分析

该系统是一个中小型的网上购物系统，旨在与校园实体超市相结合，为消费者提供便捷的在线选购方式。消费者可以在购物网站上实现参用户登录、注册、浏览商品、加入购物车、查看订单等功能，由于是校园网上购物系统，消费人群主要是师生，大都使用手机，所有在页面搭建时，尽可能少的页面跳转。

2.2 功能分析

系统功能分析可以从两个角度来看：

（1）开发人员，系统主要包括服务器后台接口编写和前端页面的搭建两部分，而每一个接口都是为了前端页面而服务，比如页面的商品展示的物品都是通过接口调用后台数据库中的数据等等

（2）从系统使用者的角度，主要包括用户和管理员两个部分，用户需要有登录注册、浏览商品、加入购物车、下订单等一系列功能；而管理员则需要注册和登录、商品管理（比如添加或者删除商品等）、浏览订单等功能。

下面的第三、四章节将以开发者的角度对该系统进行详细的介绍。

2.3 HBuilder编译器

HBuilder是DCloud推出的一款支持HTML5的Web开发IDE。HBuilder[本身](https://baike.baidu.com/item/%E6%9C%AC%E8%BA%AB/126627" \t "https://baike.baidu.com/item/HBuilder/_blank)主体是由Java编写，它基于[Eclipse](https://baike.baidu.com/item/Eclipse/61703" \t "https://baike.baidu.com/item/HBuilder/_blank)，所以顺其自然地兼容了[Eclipse](https://baike.baidu.com/item/Eclipse/61703" \t "https://baike.baidu.com/item/HBuilder/_blank)的[插件](https://baike.baidu.com/item/%E6%8F%92%E4%BB%B6" \t "https://baike.baidu.com/item/HBuilder/_blank)。快，是HBuilder的最大优势，通过完整的语法提示和代码输入法、代码块等，大幅提升HTML、js、css的开发效率。同时，它还包括最全面的语法库和浏览器兼容性数据。操作简单，下载后启动.exe文件后即可用于开发，可以同时打开多个文件夹，创建各类文如html/css/js等各类文件也非常简单，开发十分便捷。

2.4 Mongo 数据库

MongoDB是一个介于[关系数据库](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/mongodb/_blank)和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。他支持的数据结构非常松散，是类似[json](https://baike.baidu.com/item/json" \t "https://baike.baidu.com/item/mongodb/_blank)的[bson](https://baike.baidu.com/item/bson" \t "https://baike.baidu.com/item/mongodb/_blank)格式，因此可以存储比较复杂的数据类型。它具有高性能、易部署、易使用，存储数据方便的特点。Mongo最大的优势在于他支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立[索引](https://baike.baidu.com/item/%E7%B4%A2%E5%BC%95" \t "https://baike.baidu.com/item/mongodb/_blank)。

2.5 NodeJS

Node.js是一个Javascript运行环境(runtime environment)，发布于2009年5月，由Ryan Dahl开发，实质是对Chrome V8引擎进行了封装。Node.js对一些特殊用例进行优化，提供替代的API，使得V8在非浏览器环境下运行得更好。V8引擎执Javascript的速度非常快,性能非常好.Node.js是一个基于Chrome JavaScript运行时建立的平台， 用于方便地搭建响应速度快、易于扩展的网络应用。Node.js使用[事件驱动](https://baike.baidu.com/item/%E4%BA%8B%E4%BB%B6%E9%A9%B1%E5%8A%A8" \t "https://baike.baidu.com/item/node.js/_blank),非阻塞[I/O](https://baike.baidu.com/item/I/O/84718" \t "https://baike.baidu.com/item/node.js/_blank)模型而得以轻量和高效，非常适合在分布式设备上运行数据密集型的实时应用。

本系统中使用node.js主要用于下载开发所需要的环境依赖和启动服务器等。

第三章 数据库设计

3.1 数据概念模型设计

数据库是整个购物系统个核心所在，系统设计的优劣关系着系统的稳定性和执行效率。数据模型是对现实世界对象的一个模拟，在本系统中一共设计了三个实体的数据模型，下面是它们的实体属性图：

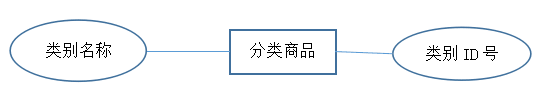


图3.1.1 商品类别信息

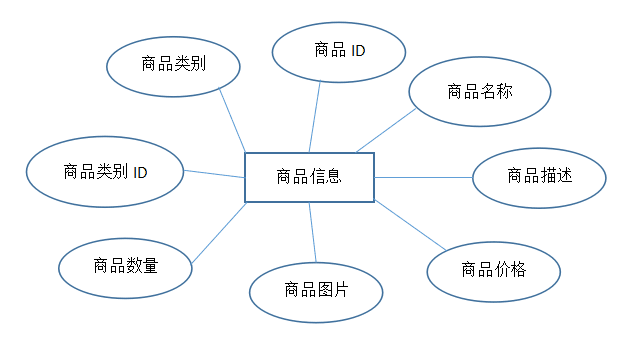


图3.1.2 商品信息

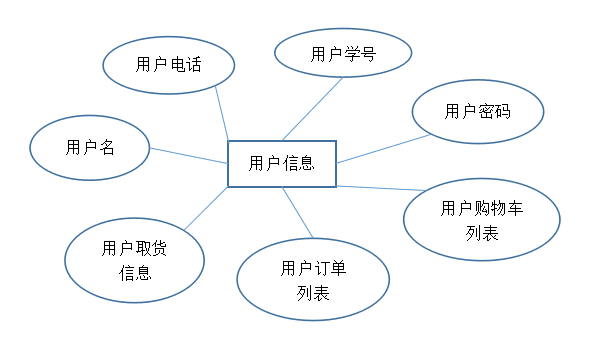


图3.1.3 用户信息

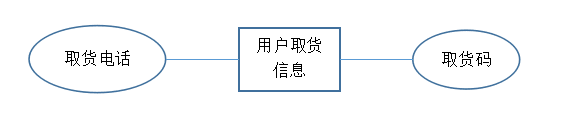


图3.1.3（a） 用户取货信息

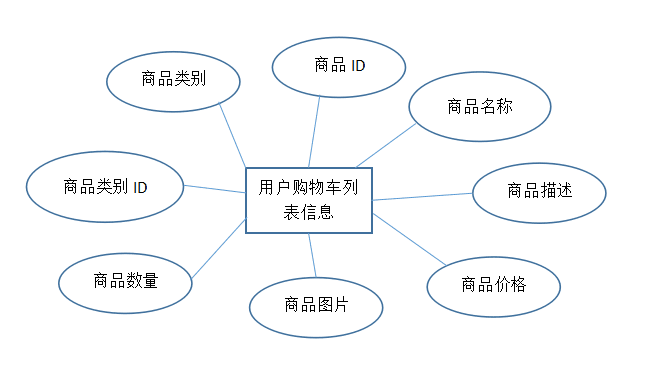


图3.1.3（b） 用户购物车列表信息

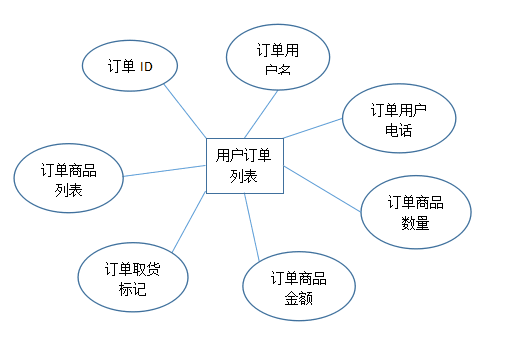


图3.1.3（c） 用户订单列表信息

3.2 数据逻辑模型设计

完成数据概念模型设计，现在开始数据库逻辑模型设计，即把3.1的实体属性图转化为数据表：

1. 商品类别信息：商品类别名称、商品类别ID；
2. 商品信息：商品类别名称商、商品类别ID、商品名称、商品ID、商品描述、商品价格、商品数量、商品图片；
3. 用户信息：用户名、用户学号、用户密码、用户手机号、用户购物车列表、用户订单列表、用户取货信息列表。用户的后三个属性有分别有各自的属性：

(1)用户购物车列表:商品类别名称商、商品类别ID、商品名称、商品ID、商品描述、商品价格、商品数量、商品图片；

(2）用户取货信息列表：取货用户名、取货用户电话；

(3）用户订单列表：订单号、订单用户名、订单用户电话、订单商品数量、订单总金额、订单取货标志、订单商品列表，订单商品列表的模型与商品模型一致。

3.3 数据物理模型设计

完成数据库逻辑设计模型后，现在开始数据库物理模型设计：

1. 商品类别信息,包括商品类别名称、商品类别ID，其中主键是商品类别ID

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| cat\_id | String | 商品类别ID |
| cat\_name | String | 商品类别名称 |

2.商品信息，包括商品类别名称商、商品类别ID、商品名称、商品ID、商品描述、商品价格、商品数量、商品图片，其中主键是商品ID

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| goods\_id | Double | 商品ID |
| goods\_name | String | 商品名称 |
| cat\_id | String | 商品类别ID |
| cat\_name | String | 商品类别名称 |
| goods\_price | String | 商品价格 |
| goods\_image | String | 商品图片 |
| goods\_desc | String | 商品描述 |
| goods\_num | Double | 商品数量 |

3.用户信息，包括用户名、用户学号、用户密码、用户手机号、用户购物车列表、用户订单列表、用户取货信,其中主键是用户账号

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| userId | Double | 用户账号（学号） |
| password | String | 密码 |
| name | String | 用户名 |
| phone | Double | 电话 |
| cartList | Array | 购物车列表 |
| orderList | Array | 订单列表 |
| userInfo | Array | 取货信息 |

3.（1）用户购物车列表，包括商品类别名称商、商品类别ID、商品名称、商品ID、商品描述、商品价格、商品数量、商品图片，其中主键是商品ID

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| goods\_id | Double | 商品ID |
| goods\_name | String | 商品名称 |
| cat\_id | String | 商品类别ID |
| cat\_name | String | 商品类别名称 |
| goods\_price | String | 商品价格 |
| goods\_image | String | 商品图片 |
| goods\_desc | String | 商品描述 |
| goods\_num | Double | 商品数量 |

3.（2）用户取货信息列表，包括取货用户名、取货用户电话

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| infoName | String | 取货用户名 |
| infoTel | Double | 取货用户电话 |

3.（3）用户订单列表：订单号、订单用户名、订单用户电话、订单商品数量、订单总金额、订单取货标志、订单商品列表，其中主键是订单号（订单商品列表的模型与商品模型一致，后面不再累赘叙述）。

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| orderId | Int32 | 订单号 |
| infoName | String | 订单用户名 |
| infoTel | Double | 订单用户电话 |
| orderTotal | Int32 | 订单商品数量 |
| orderMount | String | 订单总金额 |
| cartList | Array | 订单商品列表 |
| check | String | 订单取货标志 |

3.4 数据库创建与连接

3.4.1 外部路由设置

在server文件夹下创建一个server.js文件,并导入所需要的模块，则外部路由可做如下设置，若访问用户的接口即在端口号后面加/users 即可，在用户内部还可再设置内部路由如用户登录 可设置为/users/login等

var express = require('express')

express().use('/users',UserRouter);

express().use('/cats',CatRouter);

express().use('/goods',Goods);

3.4.2 连接和创建数据库

在server文件夹下创建一个config.js文件，放入如下代码

module.exports = {

'database': 'mongodb://127.0.0.1/hubushop'

};

即连接数据库的hubushop 若hubushop库不存在，则自动创建。

第四章 系统功能模块设计

4.1 系统功能模块设计

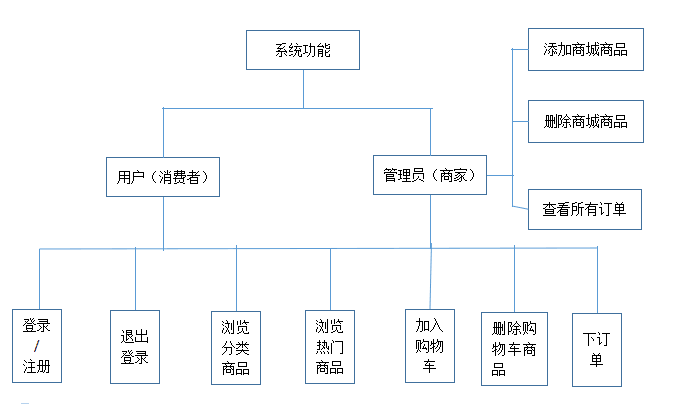


图4.1 系统功能模块

如上图所示，网上购物商城系统的功能主要围绕用户和管理员展开，下面一个章节我将对每一个功能做详细的介绍。

4.2 系统功能详细介绍

4.2.1 用户功能详细介绍

该系统一共开发首页和购物车两个页面，每个页面有不同的功能。

1. 首页
2. 热门商品浏览功能：一打开页面呈现的便是商城的热门商品，用户可以中国热门商品选择超市推荐的商品。
3. 分类商品浏览功能：用户通过点击导航栏的选项，选择性打开自己需要的商品种类，再从中选择需要的商品，减少选购的时间，后期根据需求还可以开发商品搜索功能，可以帮助用户更快更准确地找到需要的商品。
4. 加入购物车功能：每一件商品下方都有一个加入购物车的按钮，点击该按钮，弹出提示框，可以将对应商品添加到自己的购物车中。
5. 注册功能：提供新用户注册功能。点击导航栏上方的注册，会弹出注册窗口，输入注册账号之后，右侧会显示提示信息，告知用户该账号是否已经注册，若已经注册则可以用该账号直接登录，反之可以注册。为了增加安全性，注册时设置密码应为6-20位，若用户设置密码在此范围外则会有对应的提示，并且密码需要二次输入，只有两次密码长度符合且完全相同方可继续操作，验证码的设置也是为了提高安全性。注册成功后弹出提示信息，并跳转到登录窗口。逐项验证的方式可以避免用户注册时重复操作，提供更好的用户体验。
6. 登录功能：提供用户登录自己的账号用于购物。同样点击导航栏上方的登录，会弹出登录窗口，用户输入账号、密码，只有账号密码完全正确方可登录，登录成功后，导航栏的上方会显示登录的账号、购物车、退出登录按钮等。
7. 默认登录功能：在一个设备中一旦登录，设备便具有记忆功能，即使关闭设备，在下一次开启时，只要打开此购物商城系统，会显示上一次登录的账号信息。只有点击退出登录才能清空账号信息。
8. 退出登录功能：提供用户清除当前账号信息或者更换登录用户信息。
9. 显示购物车商品数量功能：导航栏的登录信息旁边显示的是用户的购物车，括号里的数字，即代表商品的件数（相同商品算一件），让用户在没有打开购物车的时候，也能从首页中了解自己购物车中商品的数量，并且每一次把商品添加到购物车，购物车数量也会实时更新。
10. 购物车页面
11. 显示购物车商品列表功能：打开我的购物车，会显示当前用户的购物车列表。用户可以查看当前购物车商品信息，并做相应的操作。
12. 商品信息管理功能：包括数量的改变和商品的清除。可以点击数量右侧小三角逐一增加或减少，也可以输入数量，同时还能点击删除按钮，删掉不需要的商品。
13. 添加取货信息功能：提供用户输入取货信息，包括取货用户名和电话并保存，下次下单可以直接选择，可以同时保存多个取货人信息。
14. 选择取货信息功能：提供用户选择已有的取货人信息，而无需每一次下单都重复输入。
15. 下订单功能：用户将需要的商品都加入购物车后，可以点击下单按钮直接下单，系统会返回订单信息，包括取货码、取货人信息以及商品信息等，用户可凭该信息到校园实体超市付款并提货。后续根据需要还可以开发网上支付送货上门等服务。

（6）显示订单列表功能：为用户提供商品订单列表，包括已取货订单和未取货订单，便于用户了解自己的购物情况。

4.2.3 管理员附加功能介绍

（1）商城系统管理功能：包括商品的添加、删除、修改等功能。为管理员提供商品的功能，超市有新品商品上架，管理员按照商品的类别，向网上商城添加商品；商品下架可删除商城中商品；价格变动则对系统商品信息进行修改

（2）查看所有订单功能：提供管理员查看所有订单列表，即所有用户的下单信息，以便管理员提前将商品打包。也就是说任何用户的下单时，下单信息不仅会存到自己的账户信息中，也会存到管理员的订单信息中。

（3）标记订单已取功能：当用户完成取货，管理标记该订单已取，避免二次备货。

第五章 系统功能测试与分析

5.1 首页功能测试与分析

1.用户注册功能

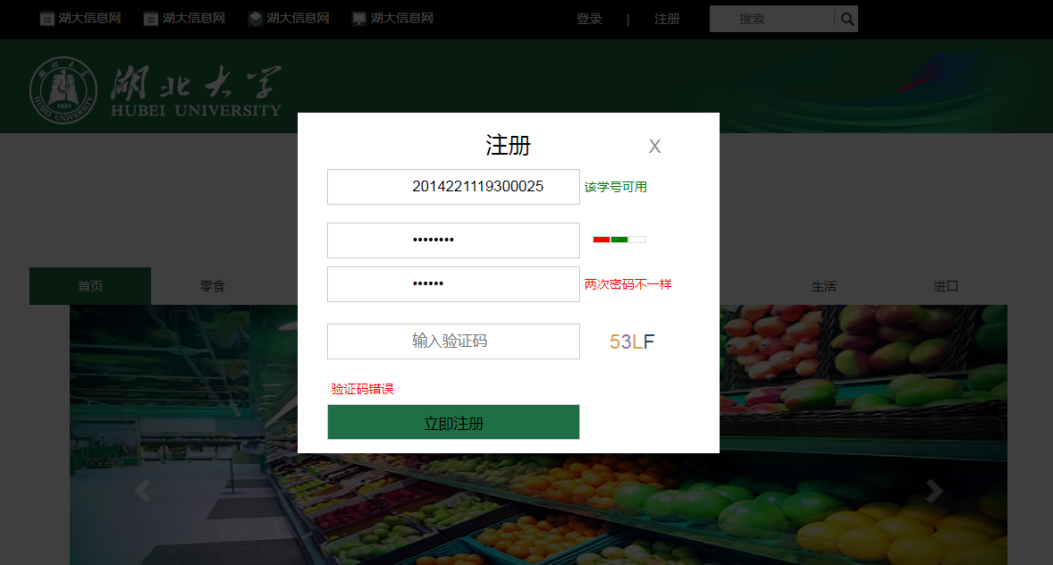


图5.1.1 用户注册功能测试

代码：

//检测用户名是否可用

$(".usernameRegister").blur(function(){

console.log("检测学号是否可用");

var userId = $(".usernameRegister").val();

// console.log(userId)

$.ajax({

type: "post",

url:"http://localhost:3000/users/registerTest",

data:{"userId":userId},

success: function(response) {

response =JSON.parse(response);

$("#testResult").css("display","");

var testResult = `${response.msg}`;

$("#testResult").html(testResult);

if(response.msg == "该学号可用"){

$("#testResult").css("color","green");

}else{

$("#testResult").css("color","red");

}

}

});

})

//验证密码，增加难易度检测。

$('.first').blur(function(){

var password=$(this).val();

if(password.length<6||password.length>20){

alert("密码要在6-20位以内，请重新输入");

}else if(password.length>=6&&password.length<8){

$("#green").css({background:"none"})

$("#blue").css({background:"none"})

}else if(password.length>=8&&password.length<12){

$("#green").css({background:"green"})

}else if(password.length>=12){

$("#green").css({background:"green"})

$("#blue").css({background:"blue"})

}

// var tar= /\$/;

// var a = tar.test(password);

})

var passwordFlag = false;

$('.second').blur(function(){

var password1=$(".first").val();

var password2=$(this).val();

// console.log(password1,password2)

if(password1 !== password2){

$("#testpassword").html("两次密码不一样")

}else{

$("#testpassword").html("")

passwordFlag = true;

}

})

//验证码

var icode=false;

$.idcode.setCode();//加载生成验证码方法

$("#Txtidcode").change(function() {

var r = $.idcode.validateCode();//调用返回值，返回值结果为true或者false

if (r == true) {

icode=true;

} else {

icode=false;

}

})

//验证码提示

$(".icode").blur(function(){

// console.log(icode)

if(icode){

$(".error1").html("")

}else{

$(".error1").html("验证码错误")

}

})

//注册

$("#register").click(function(){

console.log("准备注册");

var userId = $(".usernameRegister").val();

var password = $(".first").val();

// console.log(userId,password)

if(icode && passwordFlag){

console.log("密码一致且验证码正确");

register(userId,password);

}else{

console.log("密码或验证码不正确");

}

})

//注册函数

function register(userId,password){

$.ajax({

type: "post",

url: "http://localhost:3000/users/register",

data:{

"userId":userId,

"password":password

},

success: function(response) {

response =JSON.parse(response);

if(response.msg =="注册成功！"){

alert(response.msg + "去登录吧！");

$(".register").css("display","none");

$(".login").css("display","block");

}

}

});

}

用户注册时，输入学号框失去焦点后，左侧会出现提示，告诉用户输入学号是否可用，不可用则重新输入，输入密码的右侧有三种不同颜色条，表示用户设置密码的强度，以及密码是否可用和验证码，可以说每一步输入完成都会有相应的提示，这样可用提供很好的用户体验，不会出现点击注册是提示输入内容的某一项不符需要重新操作。

2.用户登录功能

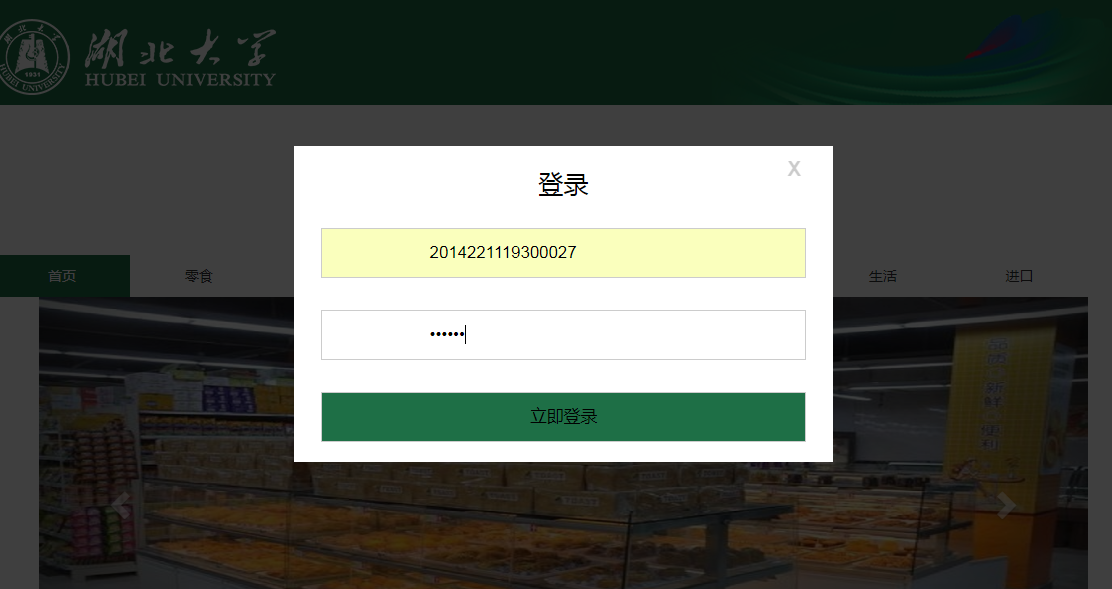


图5.1.2 用户登录功能测试

$("#login").click(function(){

login();

})

//登录函数

function login(){

var username = $(".username").val();

var password = $(".password").val();

$.ajax({

type: "post",

url: "http://localhost:3000/users/login",

data:{

"userId":username,

"password":password

},

success: function(response) {

response =JSON.parse(response);

//显示登录结果信息

if(response.message == '登录成功!'){

$("#msg").css("color","green");

//延时3s关闭登录窗口

setInterval(function(){

$(".login").css("display","none");

$(".zhezhao").css("display","none");

},3000);

//显示登录的信息

$(".nologin").css("display","none");

$(".info").css("display","block");

var useInfo = `

<a href="javascript:;">${response.result.userId}</a>`;

$(".info li").eq(0).html(useInfo);

};

var msg=`${response.message}`;

$("#msg").html(msg);

localStorage.setItem("userId",response.result.userId);

localStorage.setItem("token",response.token);

}

});

}

用户输入账号和密码即可实现登录，同时登陆窗口关闭，用户账号显示在头部信息中，如图5.1.3所示；当输入账号或者密码出现错误会显示错误信息。

1. 商品分类浏览功能



图5.1.3 商品分类导航测试

用户登录成功后，导航栏会出现分类商品名称，点击可帮助用户快速找到所需要的商品。同时登录的账号和购物车中商品种类数量信息用红色数字显示在账号旁边，点击我的购物车即可跳转到购物车和订单页面。

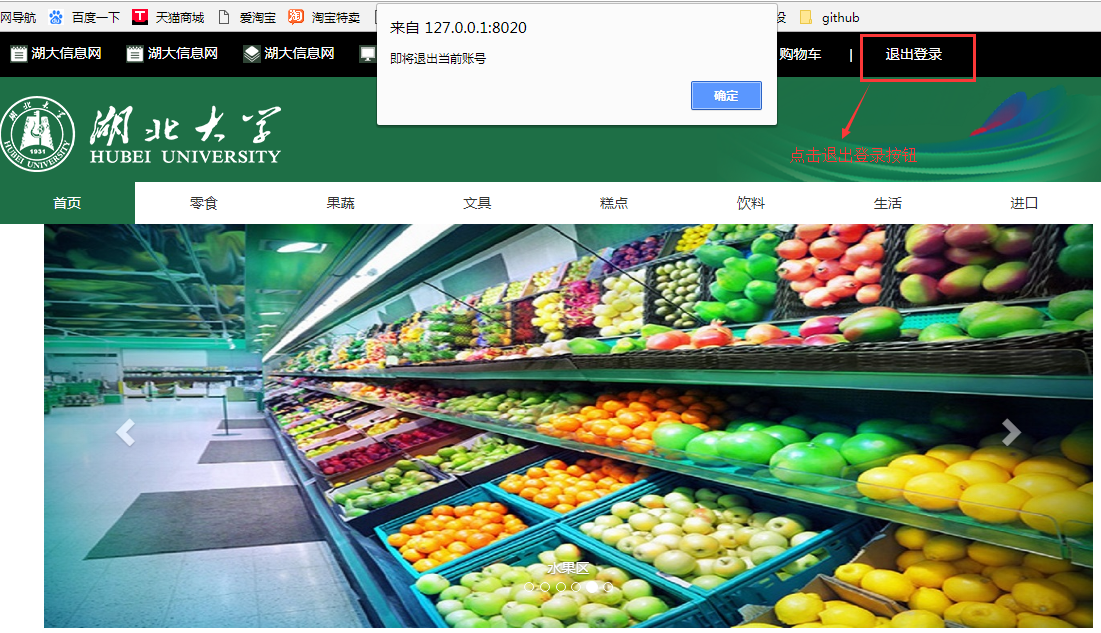
1. 用户退出登录功能

图5.1.4 用户退出登录功能测试

//退出登录则还原并清空本地存储

$(".exit").click(function(){

alert("即将退出当前账号")

$(".nologin").css("display","block");

$(".info").css("display","none");

//清空本地存储

window.localStorage.clear();

})

当用户想退出当前账号或者切换其他账号时，可点击退出登录，此时弹出一个提示框，点击确定，即可退出当前账号，用户可重新登录后进行选购商品。

5. 热门商品浏览及加入购物车功能

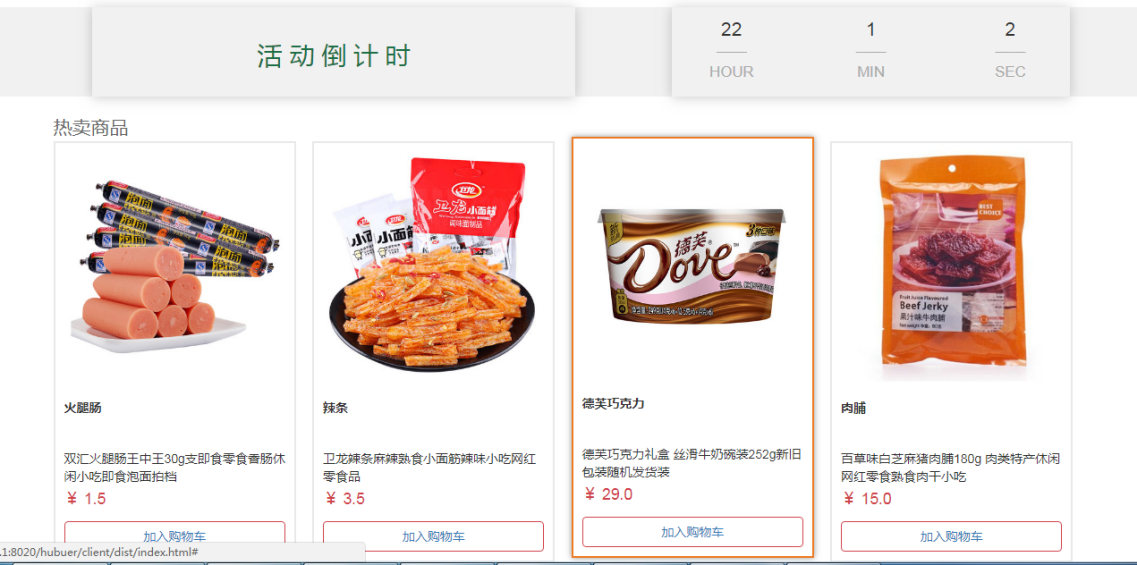


图5.1.5 热门商品浏览及加入购物车功能测试

//商品列表函数

$.ajax({

type: "get",

url: "http://localhost:3000/goods",

success: function(response) {

response =JSON.parse(response).data;

// console.log(response);

var html="";

for(let i = 0; i < response.length; i++) {

html+=`

<li>

<div class="pic">

<a href="#"><img src="../img/product/${response[i].good\_image}" alt=""></a>

</div>

<div class="main">

<div class="name">${response[i].goods\_name}</div>

<div class="decse">${response[i].goods\_desc}</div>

<div class="price">￥ ${response[i].goods\_price}</div>

<div class="btn-area">

<a href="javascript:;" class="btn btn--m" goods\_id=${response[i].goods\_id}><span>加入购物车</span></a>

</div>

</div>

</li>

`

}

$(".goods\_list").html(html);

$(".btn").click(function(e){

e.preventDefault();

var goodsId=$(this).attr("goods\_id");

//调用加入购物车函数

addCart(window.localStorage.userId,goodsId);

showLength();

})

}

});

//加入购物车

function addCart(userId,goods\_id){

var msg = "";

$.ajax({

type:"post",

url:"http://localhost:3000/users/addCart",

data:{

"userId":userId,

"goods\_id":goods\_id

},

success:function(response){

response = JSON.parse(response)

// console.log(response)

msg = response.result;

alert(msg)

}

})

}

一旦打开页面，便可看到系统中所用商品，点击任意商品的加入购物车按钮，会弹出一个提示框，告诉用户是添加成功信息，信息中可看出该商品在购物车是否存在，若存在即数量+1，即可实现将商品加入购物车，同时购物者中的商品种类数量也会随之变化。

5.2 购物车页面功能测试与分析

1. 购物车商品展示功能



图5.2.1 购物车商品展示功能测试

//购物车商品展示

showCart()

function showCart(){

// if(!window.localStorage.userId) return ;

// console.log(window.localStorage.userId);

$.ajax({

type:"get",

url:"http://localhost:3000/users/cartList?userId="+window.localStorage.userId,

success: function(response) {

var str =JSON.parse(response)

console.log(str)

response =JSON.parse(response).result.cartList;

var allMount = 0;

var html="";

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

allMount+=response[i].goods\_price\*response[i].goods\_num;

html+=`

<ul>

<li></li>

<li>

<img src="../img/product/${response[i].good\_image}" title="${response[i].goods\_desc}" />

</li>

<li>

<span >${response[i].goods\_name} </span>

</li>

<li>￥<em>${response[i].goods\_price}</em></li>

<li><input type="number" min="1" name="" class="num" goods\_id="${response[i].goods\_id}" value="${response[i].goods\_num}"/></li>

<li>￥<em>${response[i].goods\_price\*response[i].goods\_num}</em></li>

<li goods\_id="${response[i].goods\_id}" class="delete">

<input type="button" name="" value="删除" />

</li>

</ul>

`;

}

//显示购物车商品件数

var allNumber = response.length;

$(".itemNum").html(allNumber);

$(".goods\_List").html(html)

//显示购物车商品总金额

$(".itemMount").html(allMount);

// console.log(allMount);

$(".goods\_List").html(html)

//删除商品

$(".delete").click(function(){

alert("确定删除该商品?")

var goodsId=$(this).attr("goods\_id");

// console.log(goodsId);

remove(window.localStorage.userId,goodsId);

showCart();

})

//改变数量

$(".num").blur(function(){

console.log("准备改变数量")

var goodsId=$(this).attr("goods\_id");

var goodsNum=$(this).val();

console.log(goodsId,goodsNum);

fixNum(window.localStorage.userId,goodsId,goodsNum);

showCart();

})

}

})

}

在购物车页面中，主要有两个界面，购物车列表和订单列表，页面显示的内容即标题中红色字体部分：我的购物车（或者我的订单），用户可通过点击这两部分的信息随意切换，进行具体操作。

1. 用户下单功能

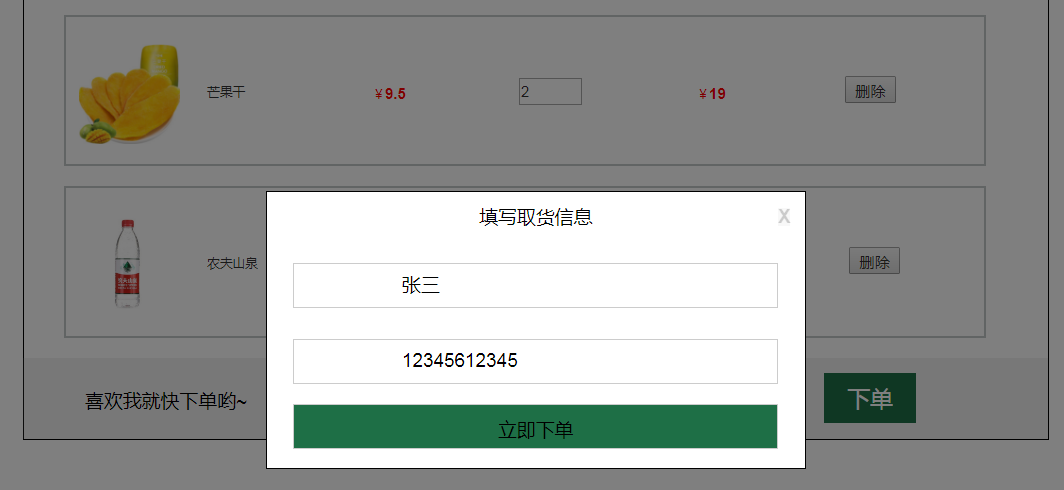


图5.2.2 用户下单并获取取货信息功能测试

用户下单功能代码：

$("#goorder").click(function(){

console.log("下单")

var infoName = document.getElementsByClassName("infoName")[0].value;

var infoTel = document.getElementsByClassName("infoTel")[0].value;

var orderTotal = $(".itemNum").html();

var orderMount = $(".itemMount").html();

console.log(orderTotal,orderMount)

goOrder(window.localStorage.userId,infoName,infoTel,orderTotal,orderMount)

alert("下单成功！请到我的订单中查看取货信息，1个小时候后即可到超市取货！")

$(".goOrder").css("display","none")

// console.log(infoName,infoTel)

return false;

})

$(".closeGoOrder").click(function(){

console.log("关闭填写取货信息窗口")

$(".goOrder").css("display","none");

return false;

})

function goOrder(userId,infoName,infoTel,orderTotal,orderMount){

var goodsArray = [];

$.ajax({

type:"get",

url:"http://localhost:3000/users/cartList?userId="+userId,

success:function(res){

res = JSON.parse(res).result.cartList

console.log(res)

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

let item = {

"goods\_id":res[i].goods\_id

}

goodsArray.push(item);

}

console.log(goodsArray)

$.ajax({

type:"post",

url:"http://localhost:3000/users/order",

data:{

"userId":userId,

"infoName":infoName,

"infoTel":infoTel,

"orderMount":orderMount,

"orderTotal":orderTotal,

"goods\_array":JSON.stringify(goodsArray)

},

success:function(response){

response = JSON.parse(response).data.orderList;

console.log(response[response.length-1])

}

})

}

})

}

用户点击购物新有下家的下单按钮，选择或输入取货人信息点击确定，点击立即下单，下单成功后会获得取货信息，也可到我的订单出查看取货信息，其中包括取货码，取货人姓名电话以及订单的详细清单，根据取货信息到超市付款并取货即可。

1. 查看订单功能



图5.2.3查看订单功能功能测试

//展示订单代码

showOrderList(window.localStorage.userId)

function showOrderList(userId){

if(userId){

console.log("订单展示")

$.ajax({

type:"get",

url:"http://localhost:3000/users/getorder?userId="+userId,

success:function(response){

response = JSON.parse(response).data.orderList

console.log(response)

var html="";

for (var i = response.length-1; i >= 0; i--) {

var inHtml = "";

for (var j = 0; j < response[i].cartList.length; j++) {

inHtml+=`

<ul>

<li>商品名称:${response[i].cartList[j].goods\_name}</li>

<li>商品单价：￥${response[i].cartList[j].goods\_price}</li>

<li>商品数量：${response[i].cartList[j].goods\_num}</li>

</ul>

<br/>

`;

}

console.log(inHtml)

html+=`

<li>

<span>取货码：

<b>${response[i].orderId}</b></span>

<span>取货人姓名：

<b>${response[i].infoName}</b></span>

<br />

<span>电话：<b>${response[i].infoTel}</b></span>

<span>商品种类数：<b>${response[i].orderTotal}</b></span>

<p>总金额：<b>${response[i].orderMount}</b></p>

<div class="goods\_list">商品清单：

${inHtml}

</div>

</li>

`;

}

$(".order\_list").html(html)

}

})

}

}

用户通过购物车页面的标题切换到我的订单，这里主要给管理员使用，用了一个订单器定时更新，一旦有用户下单即可实时监控，并且新订单会在在列表最前面，便于管理员及时备货打包。

结束语

前后花费一个多月的时间，终于把这个毕业设计完成了。该系统能实现管理员对用户和商品管理，如商品上架、订单查看等功能；用户的注册登录、商品加入购物车、添加和查看取货人信息、下订单并获取取货信息等一系列网购的基本功能。该系统的优点是，客户不需要安装程序，只需要web浏览器即可，并且代码主要集中在服务端，对于程序员日常维护带来了很大的方便。同时由于时间和知识的有限，系统中仍存在很多的不足，比如页面较少、功能相对简单，页面设计也做得不尽满意等。但还是有挺有成就感的，特别是通过这个毕业设计，让我对数据库的内容有了更深一层的认识，设计中途翻看数据库的理论知识让我受益匪浅，也觉得数据库的操作特别有意思，今后会进行更加深入的学习。希望今后自己能把系统不断地完善，使功能更加齐全，可以做到兼容各个尺寸的手机端，实现完全的响应式，以及让页面更加的美观。

参考文献

1. 美,Nicholas C.Zakas,著.JavaScript高级程序设计.李松峰,曹力,译.第三版.人民邮电出版社,2012.
2. 美,Elisabeth Robson,Eric Freeman,著.Head First HTML与CSS.徐阳,译.第二版.中国电力出版社,2013.
3. 郭远威.大数据MongoDB 实战指南.人民邮电出版社,2015.
4. 和凌志.全栈开发之道(MongoDB+Express+AngularJS+Node.js).电子工业出版社, 2017.
5. 李军爱,任正坤.超市网上购物系统的设计与实现[J].机械工程与自动化,2006,(12):23.
6. 罗军舟,吴文甲,杨明.移动互联网终端、网络与服务.计算机学报,2012,(34):11.
7. 林闯,万剑雄,向旭东等.计算机系统与计算机网络中的动态优化模型、求解与应用.计算机学报,2012,(35):7
8. 王伶俐,张传国.基于NodeJS+Express框架的轻应用定制平台的设计与实现.计算机科学,2018,44(z2).
9. 柯肇丰,曾霞霞.基于HTML5+nodeJS+MongoDB构建在线图像编辑器系统.福建电脑,2015,(6).
10. 王珊,萨师煊.数据库系统概论.第五版.高等教育出版社,2006.
11. 隋玉磊，张云鹏.基于ASP.NET的网上购物商城系统的设计与实现.西北工业大学.科学技术与工程，2007
12. 李杏杏.[B2C网站商品评论挖掘技术的研究](http://kns.cnki.net/kcms/detail/detail.aspx?filename=1014374375.nh&dbcode=CMFD&dbname=CMFD2015&v=" \t "http://kns.cnki.net/KCMS/detail/kcmstarget)[D].北京交通大学,2014
13. 郭姣.[网上商城后台管理平台的设计与实现](http://kns.cnki.net/kcms/detail/detail.aspx?filename=1015598439.nh&dbcode=CMFD&dbname=CMFD2015&v=" \t "http://kns.cnki.net/KCMS/detail/kcmstarget)[D].吉林大学,2015
14. 阮小平.[B2C电子商务模式下的网络审计技术应用](http://kns.cnki.net/kcms/detail/detail.aspx?filename=1017014484.nh&dbcode=CMFD&dbname=CMFD2017&v=" \t "http://kns.cnki.net/KCMS/detail/kcmstarget)[D].南昌大学, 2016
15. 李莉.[SQL Server 2000数据库服务器的安全策略探析](http://kns.cnki.net/kcms/detail/detail.aspx?filename=DNZS200635007&dbcode=CJFQ&dbname=cjfd2006&v=" \t "http://kns.cnki.net/kcms/detail/frame/kcmstarget)[J].电脑知识与技术,2006(35)
16. Jim Arlow,IIa Neustadt.UML 2 and the Unified Process:PracticalObject-Oriented Analysis and Design Edition,2006
17. Ningning Wang,Xi Cheng,Qinglong Gou.[Four Express Service Cooperation Modes for B2C E-Commerce: Models and Analysis](http://kns.cnki.net/kcms/detail/detail.aspx?filename=SJIG15072400000114&dbcode=SJIG" \t "http://kns.cnki.net/kcms/detail/frame/kcmstarget)[J].International Journal of Knowledge-Based Organiza,2015 (4)
18. Harish Patil,Brig.[Inventory Management Challenges for B2C E-commerce Retailers](http://kns.cnki.net/kcms/detail/detail.aspx?filename=SJES14110700518145&dbcode=SJES" \t "http://kns.cnki.net/kcms/detail/frame/kcmstarget)[J]. Rajiv Divekar.Procedia Economics and Finance ,2014
19. Kartika Firdausy,Samadri,Anton Yudhana.Web based Library Information System Using PHP and MYSQL.TELKOMNIKA,2008
20. Hsin-Yu Shih. [The dynamics of local and interactive effects on innovation adoption: The case of electronic commerce](http://kns.cnki.net/kcms/detail/detail.aspx?filename=SJES13011501986671&dbcode=SJES" \t "http://kns.cnki.net/kcms/detail/frame/kcmstarget)[J] . Journal of Engineering and Technology Management,2012 (3)

附录：

服务器部分代码：

1.goods模型

var express = require("express");

var router = express.Router();

//导入cats模型 相当于构造函数 下面可以利用它来实例化

var Goods = require('../models/goods'); //分类数据模型

//获取所有商品

router.get('/',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

Goods.find({},function(err,result){

if(err){

res.write(JSON.stringify({success:false,msg:"获取所有商品失败！"}))

res.end();

}else{

var length = result.length ;

res.write(JSON.stringify({

success:true,

msg:"获取所有商品成功！",

data:result,

allNumber:length

})

)

res.end();

}

})

})

//存入商品

router.post('/',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var param=req.body;

// console.log(param);

Goods.findOne({"goods\_id":param.goods\_id},function(err,doc){

if(err){

res.write(JSON.stringify({"status":"1",msg:err.message}))

res.end();

}else{

if(doc){

res.write(JSON.stringify({"status":'1',msg:'该商品数据库中已存在，无需再添加'}))

res.end();

}else{

var good= new Goods({

"cat\_id": param.cat\_id,

"cat\_name": param.cat\_name,

"goods\_id": param.goods\_id,

"goods\_name": param.goods\_name,

"goods\_desc": param.goods\_desc,

"goods\_price": param.goods\_price,

"good\_image": param.good\_image,

"goods\_num": param.goods\_num

})

good.save(function(err) {

if(err) {

res.write(JSON.stringify({

success: false,

message: "商品入库失败" + err

})

)

res.end();

}else{

res.write(JSON.stringify({

success: true,

"status": '0',

msg: '商品入库成功！',

doc: doc,

good: good

})

)

res.end();

}

})

}

}

})

})

//获取分类商品

router.get('/cat\_goods',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

Goods.find({"cat\_id":req.query.cat\_id},function(err,result){

if(err){

res.write(JSON.stringify({

success:false,

msg:"获取分类商品失败！"

})

)

res.end();

}else{

let number = result.length ;

res.write(JSON.stringify({

success:true,

msg:"获取一类商品成功！",

data:result,

number:number

})

)

res.end();

}

})

})

//通过路由端口导出

module.exports = router;

1. users代码

var express = require("express");

var router = express.Router();

var app = express();

var User = require('../models/users');//用户数据模型

var Goods = require('../models/goods');//用户数据模型

var jwt = require('jsonwebtoken'); // 用来创建和确认用户信息

var config = require('../../config'); //找到config.js文件

app.set('superSecret',config.secret); //超级密码

//检测账号是否可用

router.post('/registerTest',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

let param =req.body;

User.findOne({userId:param.userId},function(err,doc){

if(err){

res.write(JSON.stringify({

"status":"1",

msg:err.message

})

)

res.end();

}else{

if(doc){

res.write(JSON.stringify({

"status":'1',

msg:'该学号已存在'

})

)

res.end();

}else{

res.write(JSON.stringify({

"status":'0',

msg:'该学号可用'

})

)

res.end();

}

}

})

});

//注册

router.post('/register',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

let param =req.body;

User.findOne({userId:param.userId},function(err,doc){

if(err){

res.write(JSON.stringify({

"status":"1",

msg:err.message

})

)

res.end();

}else{

if(doc){

res.write(JSON.stringify({

"status":'1',

msg:'该学号已存在'

})

)

res.end();

}else{

var admin= new User({

"name": param.name,

"password": param.password,

"userId":param.userId,

"phone":param.phone

});

console.log(admin.password.length);

if(admin.password.length < 6 ||admin.password.length >20){

res.write(JSON.stringify({

"status":'1',

msg:'密码长度应为6-20位，请重新设置'

})

)

res.end();

}else{

admin.save(function(err){

if(err){

res.write(JSON.stringify({

success: false,

message: "注册失败"+err

})

)

res.end();

}

res.write(JSON.stringify({

success: true,

"status":'0',

msg:'注册成功！',

doc:doc,

user:admin

})

)

res.end();

})

}

}

}

})

})

//登录

router.post('/login',function(req,res){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

let param = req.body;

//同过用户传递的信息 在mongoDB数据库中查找

User.findOne({userId:param.userId},function(err,result){

if(err) throw err;

if(!result){

res.write(JSON.stringify({

success:false,

message:"登录失败！用户信息不存在"

})

)

res.end();

}else if(result.password != param.password){

res.write(JSON.stringify({

success:false,

message:"登录失败！密码不正确"

})

)

res.end();

}else{

//生成token 引用jwt模块的sign方法 前面一个对象随意填 后面一个字符串

var token = jwt.sign({name:'foo'},app.get('superSecret'))

//回馈信息

res.write(JSON.stringify({

success:true,

message:"登录成功!",

token:token,

result:result

})

)

res.end();

}

})

})

//查看购物车

router.get('/cartList',function(req,res,next){

let userId = req.query.userId;

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

User.findOne({userId:userId},function(err,doc){

res.write(JSON.stringify({

status:0,

msg:'查看购物车成功！',

result:doc//.cartList

})

)

res.end();

})

})

//商品加入购物车 及数量修改

router.post('/addCart',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var userId = req.body.userId;

var goodsId = req.body.goods\_id;

var goodsNum = req.body.goods\_num;

User.findOne({userId:userId},function(err,userDoc){

let goodsItem = '';

// 当我们添加商品的时候，判断购物车里面有没有这个商品

userDoc.cartList.forEach(function(item){

// console.log(item)

if(item && (item.goods\_id == goodsId)){

goodsItem.good\_image=item.good\_image

goodsItem = item;

if(goodsNum){

if(goodsNum == 0){

item.goods\_num=1;

}else if(goodsNum == (-1)){

item.goods\_num--;

}else{

item.goods\_num = goodsNum;

}

}else{

item.goods\_num++;

}

}

})

if(goodsItem){

userDoc.save(function (err2, doc2) {

if(err2) throw err2;

res.write(JSON.stringify({

status: '0',

msg: '',

result: '商品数量修改成功',

data:doc2

})

)

res.end()

})

}else{

Goods.findOne({ 'goods\_id': goodsId }, function (err, goodsDoc) {

userDoc.cartList.push(goodsDoc);

userDoc.save(function (err3, doc3) {

if(err3) throw err3;

res.write(JSON.stringify({

status: '0',

msg: '',

result: '加入购物车成功！',

data:doc3

})

)

res.end()

})

})

}

})

})

//删除购物车中的商品

router.post('/removeGoods',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var userId = req.body.userId;

var goodsId = req.body.goods\_id;

console.log(req.body)

User.findOne({userId:userId},function(err,userDoc){

//查找购物车里中的商品

userDoc.cartList.forEach(function(item){

if(item && (item.goods\_id == goodsId)){

// console.log(item)

userDoc.cartList.remove(item);

// console.log(userDoc);

userDoc.save(function(err3, doc3) {

if(err3) throw err3;

res.write(JSON.stringify({

status: '0',

msg: '',

result: '删除购物车商品成功！',

data: doc3

}))

res.end()

})

}else{

console.log("查无此物")

}

})

})

})

//添加取货信息 姓名、电话、取货号

router.post('/addInfo',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

let info = req.body;

var newInfo = {

"infoName":info.infoName,

"infoTel":info.infoTel

}

User.findOne({userId:info.userId},function(err,userDoc){

userDoc.userInfo.push(newInfo);

userDoc.save(function (err3, doc3) {

if(err3) throw err3;

res.write(JSON.stringify({

status: '0',

msg: '添加用户信息成功！',

data:doc3.userInfo

})

)

res.end();

})

})

})

//查看用户信息

router.get('/getInfo',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var userId = req.query.userId

User.findOne({userId:userId},function(err,userDoc){

if(err) throw err;

res.write(JSON.stringify({

status: '0',

msg: '查看用户信息成功！',

data:userDoc.userInfo

})

)

res.end();

})

})

//下单

var orderId = 10000;//返回取货码

router.post('/order',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

orderId++;

var orderNew =req.body;

var goodsArray =JSON.parse(req.body.goods\_array);

console.log(orderNew);

console.log(goodsArray);

var infoItem ={

"orderId":orderId,

"infoName":orderNew.infoName,

"infoTel":orderNew.infoTel,

"orderTotal":5,

"orderMount":10,

"cartList":[]

};

//存入用户订单

User.findOne({userId:orderNew.userId},function(err,userDoc){

// console.log(userDoc)

goodsArray.forEach(function(item){

Goods.findOne({ 'goods\_id': item.goods\_id}, function (err, goodsDoc) {

infoItem.cartList.push(goodsDoc);

})

})

setTimeout(function(){

//存入管理员订单

User.findOne({userId:20180000},function(err,userAdmin){

infoItem.check="check";

userAdmin.orderList.push(infoItem);

userAdmin.save(function (errAdmin, Admin) {

})

})

},500)

//延时器

setTimeout(function(){

infoItem.check="check";

userDoc.orderList.push(infoItem);

userDoc.save(function (err3, doc3) {

if(err3) throw err3;

res.write(JSON.stringify({

status: '0',

msg: '下单成功',

data:doc3

})

)

res.end()

})

},500)

})

})

//查看订单

router.get('/getorder',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

let userId = req.query.userId;

User.findOne({userId:userId},function(err,userDoc){

if(err) throw err;

res.write(JSON.stringify({

status: '0',

msg: '查看订单成功',

data:userDoc

})

)

res.end()

})

})

//删除订单

router.post('/removeOrder',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var userId = req.body.userId;

var orderId = req.body.orderId;

User.findOne({userId:userId},function(err,userDoc){

console.log(userDoc);

//查找购物车里中的商品

userDoc.orderList.forEach(function(item){

if(item && (item.orderId == orderId)){

console.log(item)

userDoc.orderList.remove(item);

userDoc.save(function(err, doc) {

if(err) throw err;

res.write(JSON.stringify({

status: '0',

msg:'删除订单成功！',

data: doc

})

)

res.end()

})

}

})

})

})

//标记已取订单

router.post('/outOrder',function(req,res,next){

res.writeHead(200, {

"Access-Control-Allow-Origin": "\*"

});

var userId = req.body.userId;

var orderId = req.body.orderId;

User.findOne({userId:userId},function(err,userDoc){

// console.log(userDoc);

//查找购物车里中的商品

userDoc.orderList.forEach(function(item){

if(item && (item.orderId == orderId)){

item.check = "checked";

// userDoc.orderList.remove(item);

userDoc.save(function(err, doc) {

if(err) throw err;

res.write(JSON.stringify({

status: '0',

msg:'取货成功！',

data: doc

})

)

res.end()

})

}else{

console.log("查无此订单")

}

})

})

})

//路由端口监听

module.exports = router;

致 谢

毕业设计的完成，也意味着我的四年大学生活也即将结束，此时此刻，有完成本设计的欣喜，更有对老师同学们的不舍。

首先，要感谢在这次毕业设计中帮助我的老师和同学朋友。在这一次的系统开发和研究过程遇到了很多问题，通过网上搜索和向老师以及一些正在做相关工作的朋友请教，特别感谢他们每一次的热心帮助。其中主要有我的论文指导老师潘永才教授，从论文题目的一字字推敲，到整体框架的把握都给了我很大的帮助。每一次见面他都非常的温柔耐心，给出的意见一针见血，让我的论文在书写过程变得更加的清晰简洁。

其次，要感谢四年以来，所有的勤勤恳恳教给我知识的老师们。是您们的帮助让我从高中的学习模式顺利过渡到大学学习模式，教会我不只是知识，最重要的是学习的方法提高我的专业素养，让我在毕业之前就清楚自己想要什么，要做什么。特别要感谢我的班主任，在给我们无限的自由的同时也时刻给我们敲醒警钟，公平的对待每一个人，时刻关注我们在学校的情况也关心我们家庭情况并及时给予帮助。告诫我们现实的残酷也鼓励我们勇往直前！

当然，还要感谢我的同学朋友和室友们。相识是缘分，感谢你们在学习和工作中给予我的支持和帮助，特别是我的室友们，感谢你们四年来对我的包容和体贴。特别是在我每一次打球崴脚后，你们的表面指责之下，是对我迫切的关心和无微不至的照顾。始终记得每一次体侧前的相互陪练，始终记得每一次出门逛街都会吃撑相互搀扶懒懒散散回宿舍，始终记得每一次大考前一起看书到深夜，不时抽出小零食几个人吃的津津有味......

最后，也最应该感谢的是我的家人。是你们的绝对信任，给了我莫大的安慰，对我做的任何决定总是无限的支持，在我遇到问题时，给我开导和安慰，谢谢你们，我的家人！