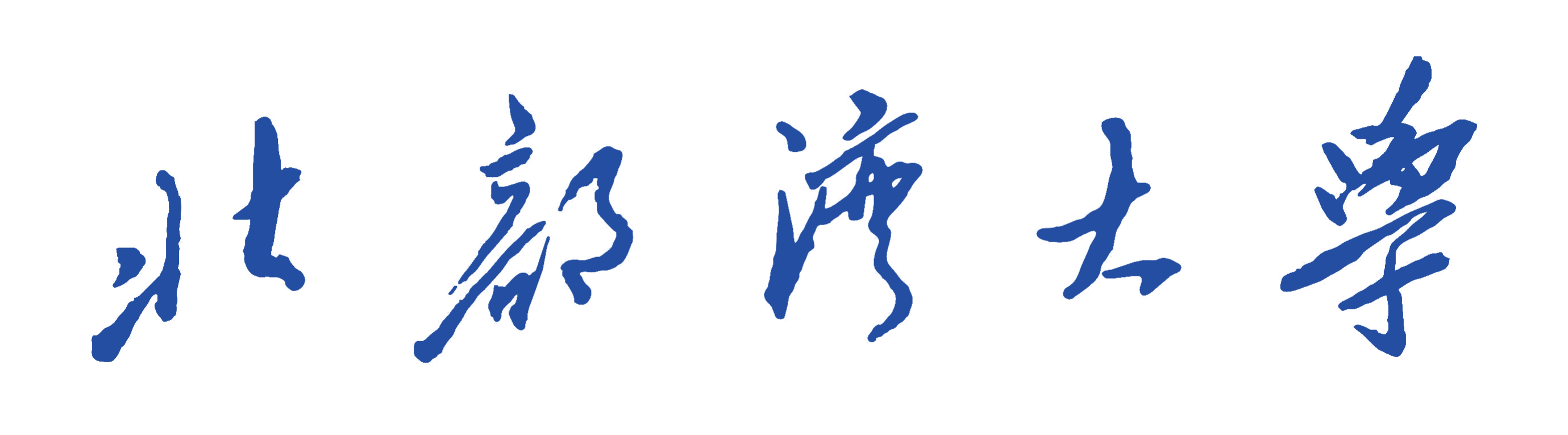
****

本 科 毕 业 论 文（设 计）

**校园二手交易网站设计与实现**

|  |  |
| --- | --- |
| **院 系：** | **电子与信息工程学院** |
| **专 业：** | **物联网工程** |
| **学 生 班 级：** | **2016级3班** |
| **姓 名：** | **梁鸿强** |
| **学 号：** | **1604412333** |
|  |  |
| **指导教师单位：** | **北部湾大学电子与信息工程学院** |
| **指导教师姓名：** | **王中圣** |
| **指导教师职称：** | **工程师** |

**2020年4月**

北部湾大学本科生毕业论文诚信承诺书

本人呈交的毕业论文是在导师指导下独立完成的研究成果。本人在论文写作中参考其他个人或集体已经发表或未发表的研究成果，均在文中以适当方式明确标明，并符合相关法律规范及《北部湾大学本科生毕业论文（设计）规范》。

学生声明（签名）：

年 月 日

**摘 要**

随着互联网技术和移动端设施的快速发展,手机、平板等移动设备在我们的生活中无处不在，扮演着极其重要的角色。webApp以开发的便捷性、以及跨平台开发性成为了现在手机APP开发的主流。人们的生活水平不断提高，闲置的物品也越来越多。在大学校园生活中，应届生在离开学校之际，如何处理这些闲置物品，做到环保不浪费是一件非常重要的事情，开发一个校园二手交易APP，不仅符合现在移动端的主流，还能够对闲置物品进行处理，可谓是一举两得。

校园二手交易APP的开发分为两部分，前端使用的是现在如火如荼的三大框架之一的Vue.js，后端则是采用了node.js，利用了MVVM开发模式，层次分明，成功实现了该系统，数据库采用MYSQL。论文首先阐述了校园二手交易APP的开发背景，并对该项目进行较为详细的需求分析，后面详细的分析了该项目的功能需求、业务流程、数据流程、系统功能和系统架构设计等方面的问题。利用当前先进的开发模式、开发技术、开发思想完成该项目的开发，为校园提供一个较为便捷的闲置物品交易平台。

**关键字**：校园二手交易APP，Vue，node.js，Mysql数据库

**Abstract**

With the rapid development of Internet technology and mobile terminal facilities, mobile phones, tablets and other mobile devices are everywhere in our lives, playing an extremely important role. With the convenience of development and cross platform development, webapp has become the mainstream of mobile app development. People's living standards continue to improve, and there are more and more idle goods. In college campus life, it's very important for new students to deal with these idle items when they leave the school. Developing a second-hand trading app on campus is not only in line with the current mainstream of mobile terminals, but also able to deal with the idle items, which can be called "two birds with one stone".

The development of the second-hand trading app on campus is divided into two parts. Vue.js, one of the three frameworks in full swing at present, is used in the front-end, node.js is used in the back-end, MVVM development mode is used, with clear layers, and the system is successfully implemented. MySQL is used in the database. Firstly, the paper describes the background of the development of the second-hand trading app on campus, and makes a more detailed demand analysis of the project. Then, it analyzes the functional requirements, business process, data process, system function and system architecture design of the project in detail. Using the current advanced development mode, development technology, development ideas to complete the development of the project, to provide a more convenient platform for idle goods trading on campus.

**Keywords:** campus second-hand transaction app, Vue, node.js, MySQL database

**目 录**

**[1绪论](#_Toc1851)** [1](#_Toc1851)

[1.1开发背景与意义 1](#_Toc23446)

[1.2论文结构和内容 1](#_Toc23887)

**[2系统开发技术与环境](#_Toc24284)** [3](#_Toc24284)

[2.1 系统开发语言 3](#_Toc25191)

[2.2 系统前端技术 3](#_Toc22563)

[2.3 系统后端技术 4](#_Toc27403)

[2.4 系统数据库的选择 4](#_Toc13930)

[2.5 系统的运行环境 5](#_Toc26294)

[2.5.1 硬件环境 5](#_Toc649)

[2.5.2 软件环境 5](#_Toc15644)

**[3系统分析](#_Toc26292)** [6](#_Toc26292)

[3.1可行性分析 6](#_Toc5029)

[3.1.1 技术可行性 6](#_Toc23484)

[3.1.2 操作可行性 6](#_Toc28887)

[3.2需求分析 6](#_Toc30423)

[3.3.1 功能需求分析 7](#_Toc32313)

[3.3.1 性能需求分析 7](#_Toc22972)

[3.3系统流程分析 8](#_Toc26088)

[3.4数据流程分析 9](#_Toc18999)

**[4系统设计概要](#_Toc5124)** [10](#_Toc5124)

[4.1 系统开发软件的选择 10](#_Toc24290)

[4.2 系统总体功能设计 10](#_Toc8228)

[4.3 系统数据库设计 10](#_Toc26301)

[4.3.1 数据库结构设计 11](#_Toc26718)

**[5系统详细设计和实现](#_Toc22643)** [14](#_Toc22643)

[5.1 会员模块设计 14](#_Toc13874)

[5.1.1 登录模块设计 14](#_Toc23584)

[5.1.2 首页模块设计 14](#_Toc25492)

[5.1.3 商品模块设计 15](#_Toc14192)

[5.1.4 会员信息模块设计 15](#_Toc15200)

[5.1.5 交易模块模块设计 16](#_Toc13943)

[5.1.6 会员交易信息模块设计 17](#_Toc24618)

5.2 管理员模块设计

[5.2.2 公告发布模块设计 17](#_Toc24618)

**[6系统测试](#_Toc17180)** [18](#_Toc17180)

[6.1 测试方法 18](#_Toc10073)

[6.2 测试实例 18](#_Toc3032)

**[总结与展望](#_Toc16746)** [20](#_Toc16746)

**[参考文献](#_Toc29560)** [21](#_Toc29560)

**[致谢](#_Toc12590)** [23](#_Toc12590)

**1绪论**

## 1.1开发背景与意义

进入二十一世纪以来，以因特网为核心的网络技术和通信技术在全世界得到了飞快的发展和广泛的应用，从以前的2G网络到现在的5G网络，再到无线网络(WIFI)的普及，无一不证明了当前互联网发展之快。不仅仅互联网在发展，手机、平板电脑等移动设备的发展速度也是令人叹为观止，现在的手机，平板等移动设施不仅能够接听电话，电脑能做的事情，在手机上几乎全部能够实现。在强大的移动设施基础和互联网的背景下，手机APP犹如雨后春笋一般不断冒出，深入人们生活中的点点滴滴。

随着生活水平的提高，在校大学生的购买力也越来越强，产品的更新换代速度也越来越快，大学校园存在着更多的闲置物品，例如书籍、衣物、电子产品、健身器材等。虽然有闲鱼、转转等二手交易平台为人们提供交易闲置物品的路径。但是由于其面向全国各地，从距离、运费、以及问题处理等几个方面来说，显得不够便捷、不够快速。校园二手交易APP从空间范围上缩小到了学校，只为校内学生服务，提供一个发布闲置物品的平台。学生可以通过发布信息提供的联系方式，联系到卖家，在线下商量讨论如何完成交易，出了问题也可以在校内第一时间联系到并进行处理，与传统二手物品交易网相比，具有市场细分更明确，服务针对性更强的特点，更容易获得学生群体的认可。

**1.2开发方法**

本课题的开发采用vue.js+node.js+mysql进行实现。首先我们用相关知识、以及对其他相似平台进行了解，并结合相关需求进行需求分析。然后对这个课题进行一个总体的架构设计。界面采用UI框架进行编写，在表单中获取相应的数据，在前端进行验证，通过ajax方法请求后台接口，把数据返回到后台，有后台来操作数据库，操作或返回响应的数据。以及根据用户的状态来调用不同的方法，显示不同的页面和数据。在整个设计过程中，根据需求分析，编写系统的实现过程，和项目所涉及到的技术难点的实现方法。系统开发完成之后，进行多方面的测试和调试，例如压力测试，并发测试等，并做相关的记录。

**2系统开发技术与环境**

**2.1系统整体架构**

该系统采用的是前后端分离的开发架构，采用restful api设计理念。后端采用node.js的集成框架express、采用模块化的思想进行开发，提供API接口。前端JS技术采用了ES6语法进行编写、使用了目前最火的MVVM框架Vue.js，CSS使用了预处理语言Less，UI框架使用vant，ajax技术采用的是Axios，使用了vue-cli+webpack技术来快速构建项目。数据库则采用了Mysql。

**2.2前端技术介绍**

ES6是2015年由Ecma国际组织在2016年推出的一套JavaScript标准，与以往的标准不同，ES6在原来的基础上新增了大量的特性，例如声明变量的新方式let、const、解构赋值、箭头函数、Promise等、首次引入了Class概念、块级作用域等，使得JavaScript越来越往面向对象类型语言发展，解决了很多ES5存在的问题，比如全局泄露、内存泄漏、回调地狱的问题。目前各大主流浏览器都兼容ES6的大部分特性，使得利用ES6开发相当便捷。

Vue.js作为目前最火的前端MVVM框架之一，Vue.js因其上手容易，可拓展性高成为了前端工程师们最为青睐的一个JavaScript框架。Vue.js的核心思想就是双向绑定，View层或Model层的数据发生更新，都会通过Vue.js把数据进行响应式绑定，做到数据实时动态更新。组件化也是Vue.js的最强大的功能之一，可以把组件可以扩展HTML元素，封装可重用的代码，定义自定义组件。Jquery需要我们编写代码去操作DOM元素获取数据。Vue.js则不同，开发者只需关注数据图层，数据的获取、流转等操作由Vue.js内部来完成。

Less是CSS预处理语言中的一种，与普通的CSS语言不同的是，Less将CSS赋予了动态语言的特性，例如变量、继承、运算、函数嵌套，使得CSS正式加入了程序语言的行列，使得CSS代码更容易书写、维护，不过Less语言并不能通过浏览器直接编译，需要通过webpack把.less文件编译成.css文件。

Vant是一款基于Vue.js的移动端UI组件库，提供了大量的商城项目组件，例如购物车，商品列表、图片懒加载、商品导航、优惠券等，同时还支持按需导入组件、组件重制等，是用来完成该项目的不二之选。

Axios是一款基于Promise的HTTP库，通过Promise对Ajax进行了封装，它和Jquery中的Ajax方法相比，优势如下：

1.Jquery处理Ajax函数返回的数据采用的是回调函数，如果有多个回调函数要进行多层嵌套，会导致“回调地狱”，代码将会变得非常杂乱且难以维护，而Axios是通过Promise来发起并处理Ajax请求，其链式写法会极大的改善“回调地狱”所带来的问题。

2.Axios能自动转换json数据，Jquery需要我们自己手动指定类型。

3.Axios浏览器端支持防止CSRF（跨站请求伪造），Jquery不支持。

4.Axios专门用于发起和处理Ajax请求，但是Jquery如果要使用Ajax，就需要导入整个Jquery项目，显得不太合理。

Webpack是一款自动化的模块打包机。Webpack视HTML，JS，CSS，图片等文件都是一种资源，每个资源文件都是一个模块（module）文件，Webpack就是根据每个模块文件之间的依赖关系将所有的模块打包（bundle）起来。Webpack还具有强大的Loader(加载器)和Plugin（插件）功能，我们可以通过安装这些Loder和Plugin，在配置文件中进行配置，启动项目时，Webpack会帮我们打包、压缩各式各样的文件，还可以将代码切割成不同的chunk，实现按需加载，降低了初始化时间，提高兼容性、优化整个项目的运行效率。

**2.2服务器端技术介绍**

Node.js是一个让 JavaScript 运行在服务端的开发平台，它让 JavaScript 成为与PHP、Python、Perl、Java等服务端语言平起平坐的脚本语言。Java是一门多线程语言，而node不同，node属于单线程语言，但是Node采用了一个称为“事件循环(event loop）”的架构，使得编写可扩展性高的服务器变得既容易又安全。提高服务器性能的技巧有多种多样。Node选择了一种既能提高性能，又能减低开发复杂度的架构。这是一个非常重要的特性。并发编程通常很复杂且布满地雷。Node绕过了这些，但仍提供很好的性能。

Express 是一个简洁而灵活的 node.js Web应用框架, 提供了一系列强大特性帮助你创建各种 Web 应用，和丰富的 HTTP 工具。可以快速地搭建一个完整功能的网站。Express对web开发相关的模块进行了适度的封装，屏蔽了大量复杂繁琐的技术细节，让开发者只需要专注于业务逻辑的开发，极大的降低了入门和学习的成本。

**2.3数据库介绍**

MySQL属于传统的关系型数据库产品，其开放式的架构使得用户的选择性很强，而且随着技术的逐渐成熟，MySQL支持的功能也越来越多，性能也在不断地提高，对平台的支持也在增多，此外，社区的开发与维护人数也很多。当下，MySQL因为其功能稳定、性能卓越，且在遵守GPL协议的前提下，可以免费使用与修改，因此深受用户喜爱。

关系型数据库的特点是将数据保存在不同的表中，再将这些表放入不同的数据库中，而不是将所有的数据统一放在一个大仓库里，这样的设计加快了MySQL的读取速度，而且它的灵活性和可管理性也得到了很大的提高。访问及管理MySQL数据库的最常用标准化语言为SQL——结构化查询语言。SQL使得对数据库进行存储、更新和存取信息的操作变得更加容易。

**3系统分析**

**3.1可行性分析**

**3.1.1技术可行性**

现在webApp开发技术非常成熟也非常流行，Vue.js在移动端开发更是无所不在。Node.js也发展的相当的成熟，npm生态圈中也有很多非常好用的拓展插件，再加上模块打包机Webpack的鼎力相助，我们能够非常轻松的构建出一个webApp项目，所以使用Vue+Webpack+Node来开发本项目在技术上来说是可行的。

**3.1.2 经济可行性**

该项目用的所有技术，包括数据库，都是面向开源，并且是免费的。该项目的研发时间虽然比较长，但是该项目的作用多，功能强大，并且可拓展性强。所带来的经济效益，绝对要比付出多，所以在经济上是可行的。

**3.1.3 操作可行性**

对于一个项目来讲，开发之后其操作难度决定了它能否受到群众们的青睐，这是开发者必须考虑的问题。现在的人们对于购物APP，例如淘宝、天猫、闲鱼等软件的操作可以说是得心应手。该项目是按照他们的模式进行开发，所以都是广大群众非常熟练的模式，一切的操作都很简单，易于上手。

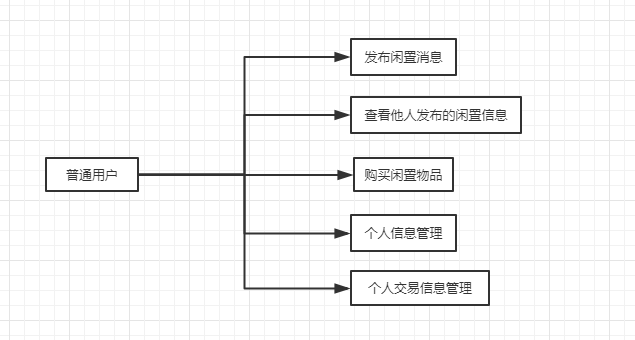
**3.2需求分析**

**3.2.1角色功能分析**

首先对项目进行需求分析，根据用户的需要和要求，进行需求分析并整理出相应的文档。

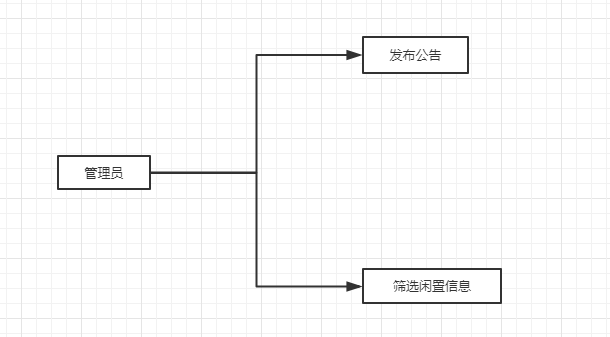
该项目的主要工作所包括：提供一个在校学生们发布闲置物品消息的平台，提供自己的联系方式，有需求的人可以根据联系方式和发布者进行协商，在线下进行交易。

普通用户所具有的功能有发布闲置物品、查看其他用户发布的闲置信息，购买闲置物品、查看卖家信息等，如图：



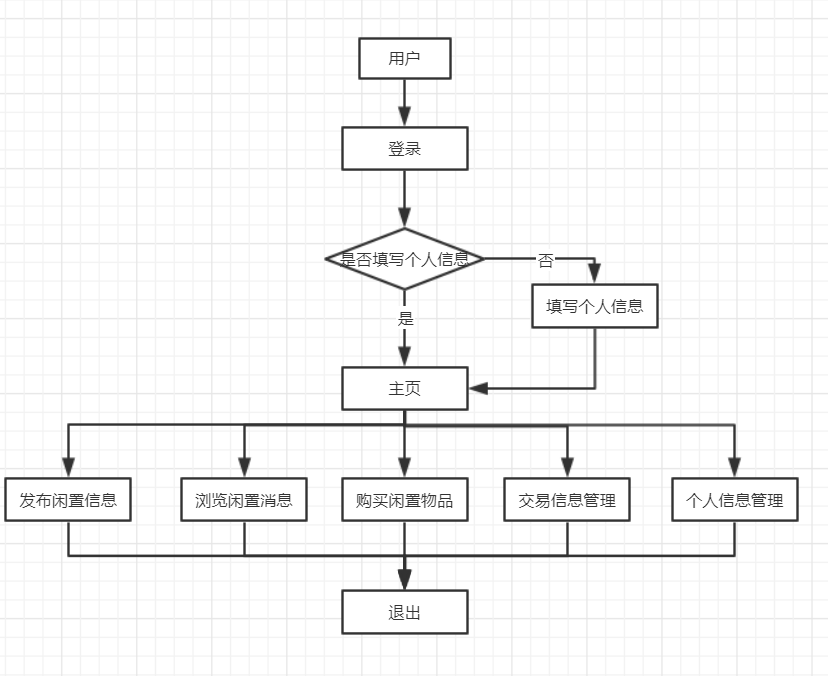
**图3.2 普通户用例图**

管理员所具有的功能有发布公告、和筛选用户发布的闲置信息，防止不良信息发布，如图：

**图3.3 管理员用例图**

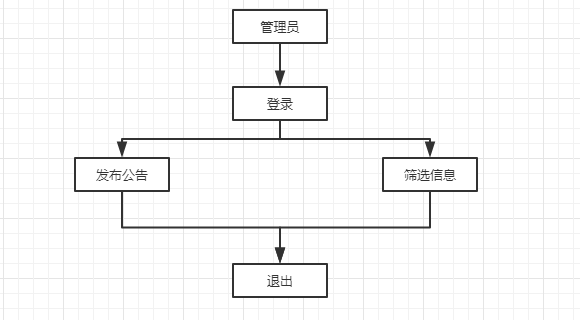
**3.2.2**功能流程**分析**

普通用户的流程大致如下：



**图3.4 普通用户流程图**

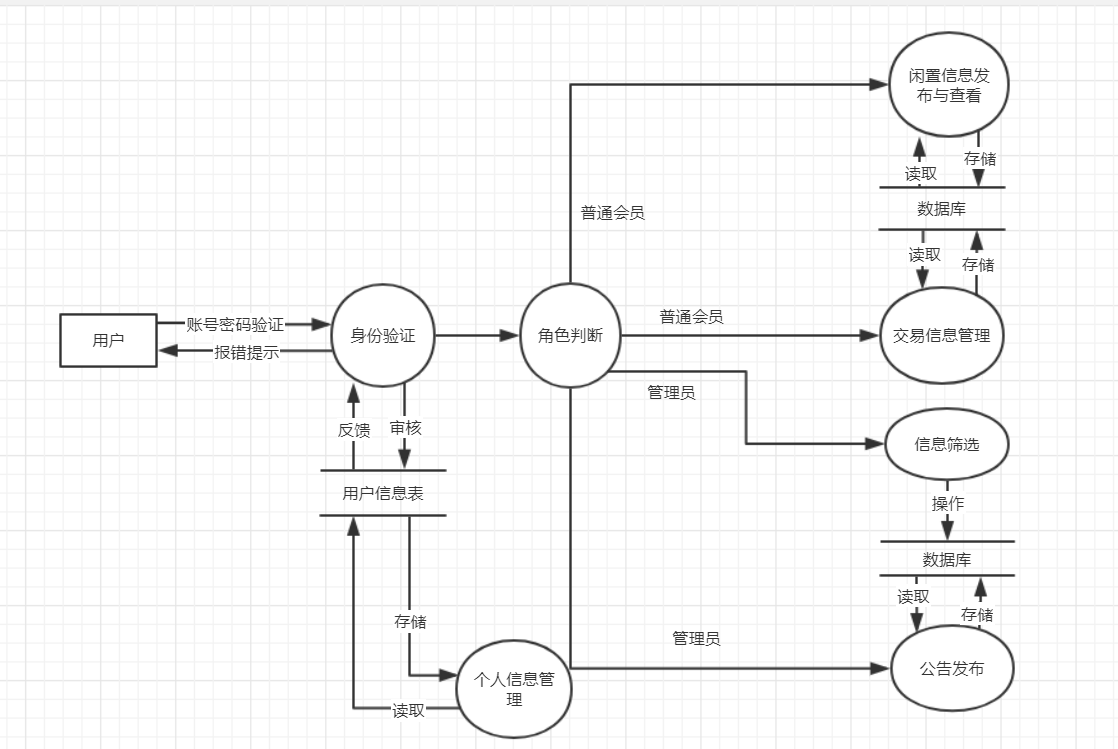
管理员流程大致如下：



**图3.5 管理员流程图**

**3.2.3数据**流程**分析**

本项目的数据处理过程如图所示：

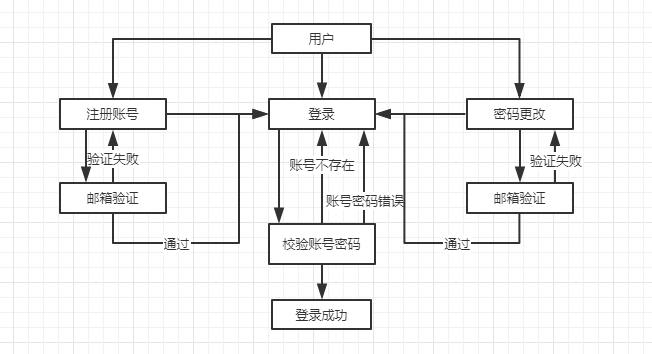


**图3.6 项目数据流程图**

**4 系统设计分析**

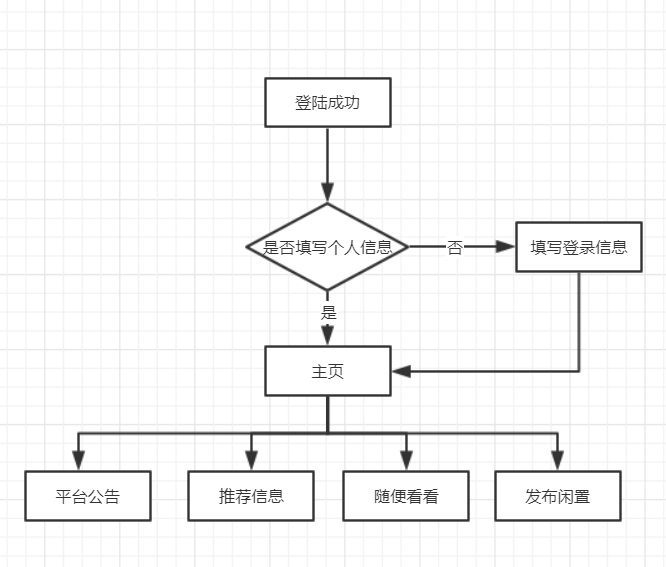
## 4.1 模块设计分析

**4.1.1 登录模块设计**

账号统一使用邮箱号进行登录，如果没有账号需要用邮箱号注册，系统会向你所填写的邮箱名发送验证码，当输入邮箱验证码通过时，才能够成功激活你的账号，更改密码也需要通过邮箱验证，如图：

**图4.1 登录模块设计图**

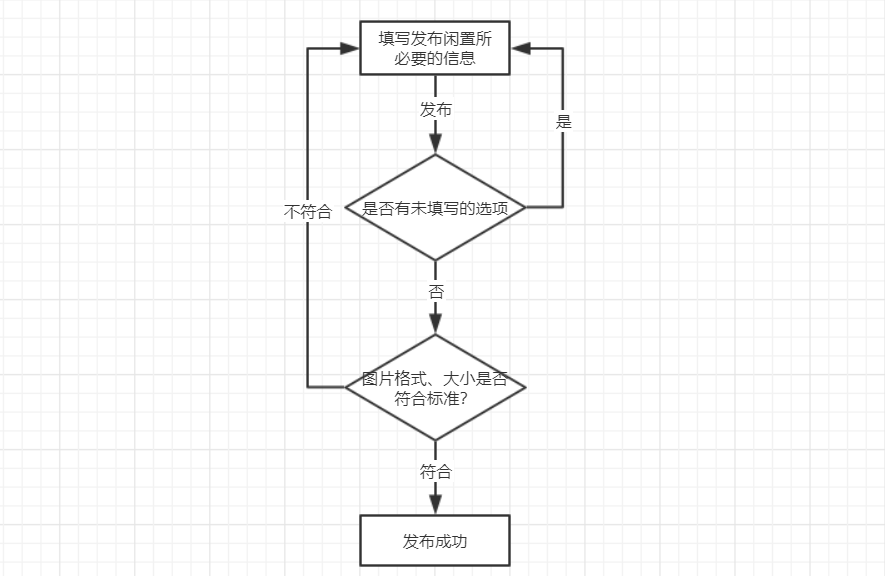
**4.1.2 主页模块设计**

账号登陆成功后要进入主页时，会先检测用户是否已经填写个人信息，如果没有会让用户先填写个人信息，然后再进入主页。主页有管理员发布的公告，还有随机为用户推荐的闲置消息，用户可以通过主页跳转到发布闲置的页面或者其他用户发布的闲置信息列表。如图：

**图4.2 主页模块设计图**

**4.1.3 发布闲置模块设计**

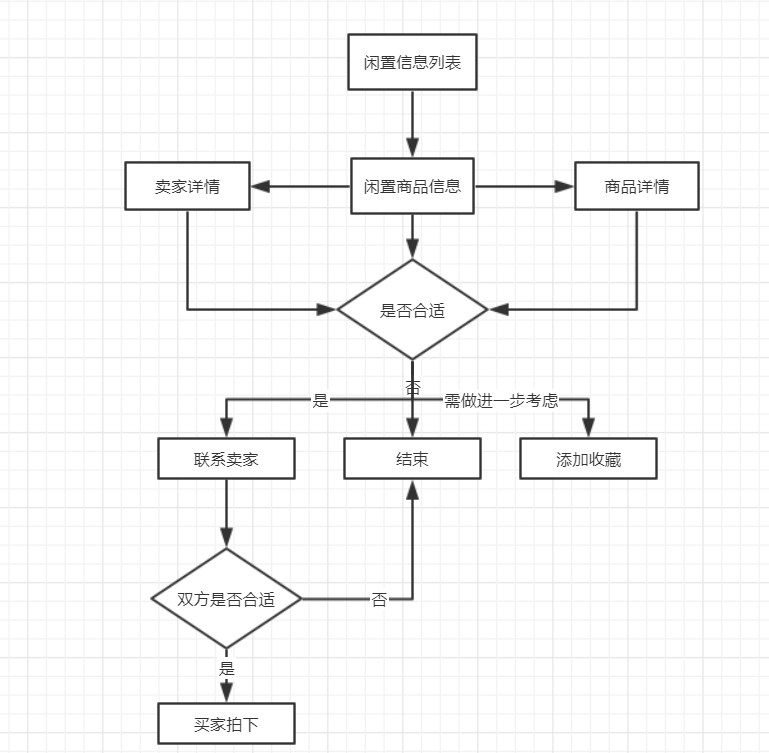
用户发布闲置信息，需要填写闲置信息的标题，闲置信息的原因，包括新旧程度，出售原因，破损情况等等。上传的图片最多只能有三张，并且每张的大小不能超过500KB，这也是为了缓解服务器的压力。最后再写上自己出售的价格，点击发布按钮，即可发布。设计图如下：



**图4.3 发布闲置模块设计图**

**4.1.4 闲置交易模块设计**

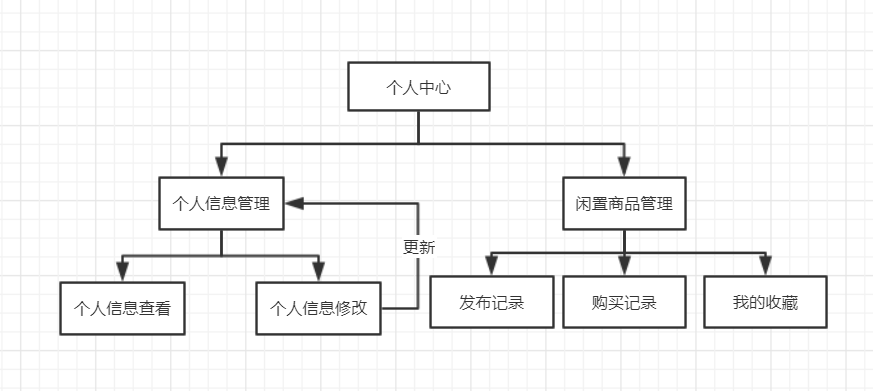
用户可以再信息列表中浏览其他用户所发布的闲置信息，如果看到了心仪的商品，你可以选择点击该商品跳转到商品信息页面。商品信息页面中有着该商品更为详细的消息，还有卖家的详细信息提供查看，以获得卖家的联系方式，和商家进行联系，双方觉得合适的话买家就可以拍下该商品，进行交易。如果只是觉得感兴趣，但还需要进一步考虑，还可以考虑把这件商品收藏起来。设计图如下：



**图4.4 闲置交易模块设计图**

**4.1.5 个人中心模块设计**

个人中心主要分两个部分，一个是个人资料，一个是闲置商品发布和交易记录以及收藏的闲置商品。用户可以在这里看到自己发布的所有商品信息以及状态，还可以看到自己的购买记录以及自己收藏的商品。用户的个人信息如果遇到了改变，可以随时更改信息。设计图如下：



**图4.5 个人中心模块设计图**

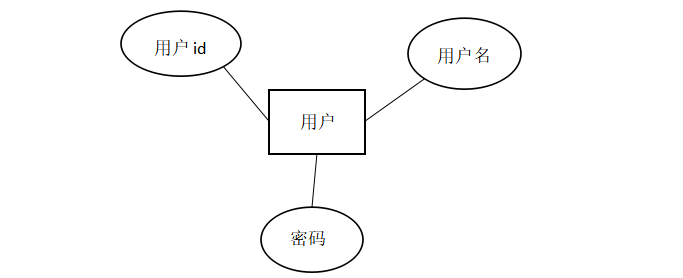
### 

### 4.2 关系设计

E-R图也称实体-联系图(Entity Relationship Diagram)，提供了表示实体类型、属性和联系的方法，用来描述[现实世界](https://baike.baidu.com/item/%E7%8E%B0%E5%AE%9E%E4%B8%96%E7%95%8C/688877" \t "https://baike.baidu.com/item/E-R%E5%9B%BE/_blank)的[概念模型](https://baike.baidu.com/item/%E6%A6%82%E5%BF%B5%E6%A8%A1%E5%9E%8B/3187025" \t "https://baike.baidu.com/item/E-R%E5%9B%BE/_blank)。我们可以通过E-R图更加清晰的了解到每一个实体的属性，以及实体与实体之间的关系。

(1)用户实体E-R图设计

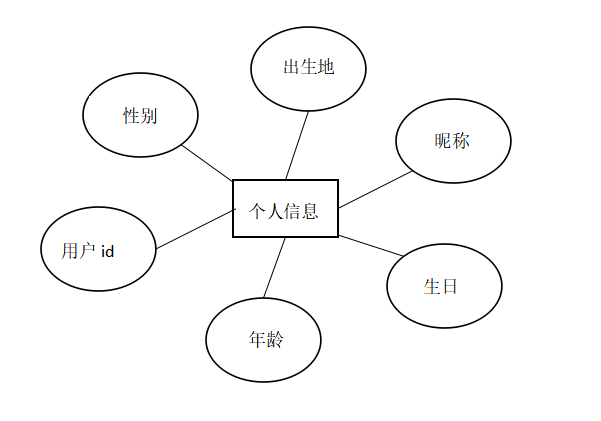
用户实体就是针对普通用户和管理员设置id、用户名、密码。E-R图如下所示：



**图4.6用户实体E-R图设计**

(2)个人信息实体E-R图设计

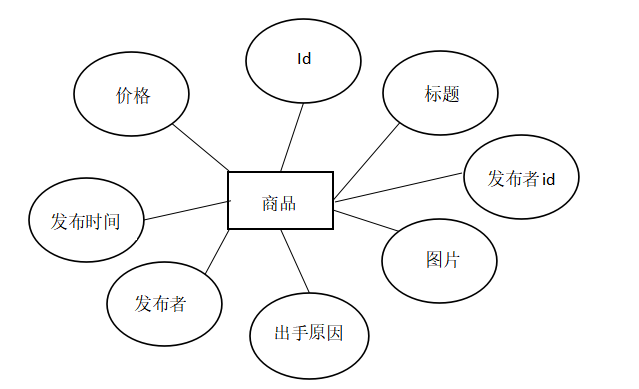
个人信息实体是普通用户个人信息所具有属性，包括id、昵称、出生地、生日、年龄、联系方式等详细信息。E-R图如下所示：



**图4.7个人信息实体E-R图设计**

(3)商品实体E-R图设计

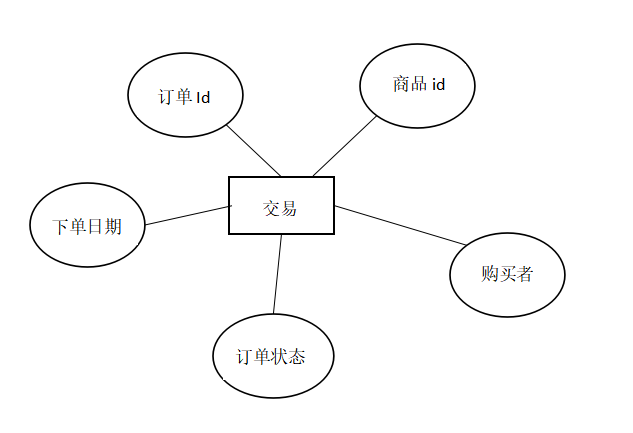
商品实体是普通用户所发布的闲置商品所具有属性，包括id、标题、图片、出售原因、价格等详细信息。E-R图如下所示：



**图4.8商品实体E-R图设计**

(4)交易实体E-R图设计

交易实体是交易订单所具有的属性，具有订单id，商品id，下单日期，购买者，订单状态等。E-R图如下：



**图4.9交易实体E-R图设计**

### 4.3 数据库设计

本项目一共具有6个数据表，包括user(用户表)，user\_info(用户信息表)，goods\_info(商品信息表)，goods\_img(商品图片路径表)，user\_collection(用户收藏表)，good\_business(交易记录表)，notice(公告表)。

**表4.1 user表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| user\_id | varchar(20) | 用户id | 否 | 主键 |
| user\_name | varchar(20) | 用户账号 | 否 |  |
| User\_pwd | varchar(20) | 密码 | 否 |  |

**表4.2 user\_info表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| u\_id | varchar(20) | 用户id | 否 | 主键 |
| u\_name | varchar(20) | 用户昵称 | 否 |  |
| u\_area | varchar(20) | 用户地区 | 否 |  |
| u\_headimg | varchar(50) | 用户头像 | 否 |  |
| u\_sex | int | 性别 | 否 |  |
| u\_age | int | 年龄 | 否 |  |
| u\_birthday | varchar(20) | 生日 | 否 |  |
| u\_phonenum | varchar(20) | 电话号码 | 否 |  |
| u\_mail | varchar(20) | 邮箱 | 否 |  |
| u\_qq | varchar(20) | QQ号码 | 否 |  |
| u\_wechat | varchar(20) | 微信号码 | 否 |  |

**表4.3 good\_info表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| good\_id | varchar(20) | 用户id | 否 | 主键 |
| good\_title | varchar(20) | 用户账号 | 否 |  |
| good\_info | varchar(20) | 密码 | 否 |  |
| good\_price | decimal | 价格 | 否 |  |
| seller\_id | varchar(20) | 发布者id | 否 |  |
| seller | Varchar(20) | 发布者昵称 | 否 |  |

**表4.4 good\_img表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| good\_img\_url | varchar(20) | 用户id | 否 |  |
| good\_id | varchar(20) | 用户账号 | 否 |  |

**5系统模块实现**

**5.1 用户账户模块实现**

**5.1.1邮箱验证码实现**

在注册和更改密码的时候，采用邮箱验证码进行验证。邮箱验证码采用的是nodemalier插件，该插件可以帮助使我们实现发送验证码的效果。首先需要用npm install mailer下载该插件。下载成功后，先进行配置，代码如下：



**图5.1.1 nodemailer配置代码**

在配置完以后，封装一个发送验证码的函数，代码如下：



**图5.1.2 发送验证码函数代码**

在注册账号中我们输入账号，点击获取验证码，结果如下：



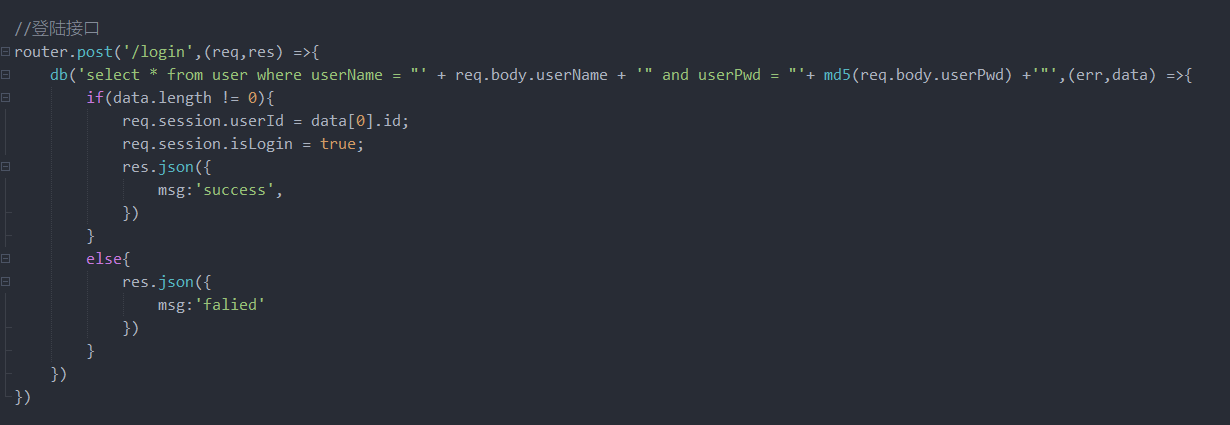
**图5.1.3 获取验证码**

**5.1.2 用户登录、注册、更新密码实现**

当用户输入账户和密码时，前端会进行一些基础的校验，比如验证设置的密码长度是否大于或等于8位数，填写账号是否为邮箱码格式，非空校验。在确认输入的一切正常之后，由axios发起ajax请求，向后台请求数据，后端接收到传来的数据之后，会进入到数据库中进行校验，返回相应的结果给前端判断。



**图5.1.4 登陆请求代码**



**图5.1.5 登陆接口代码**

**5.2 主页模块实现**

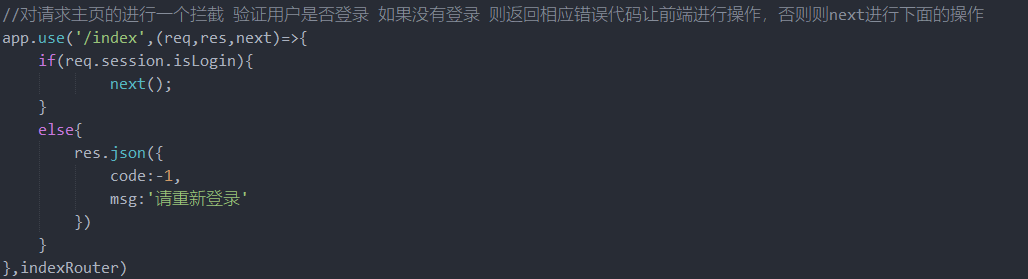
**5.2.1 用户登录拦截实现**

当用户登录成功跳转进入主页时，要先判断用户是否已经登录，如果没有应该跳转到登录的页面，如果有则成功进入主页。



**图5.2.1 主页拦截**

该需求的实现主要是采用了express框架的路由拦截，所有访问主页的请求都会先被拦截下来，进行验证。通过了才能够访问这个路由下的api接口。



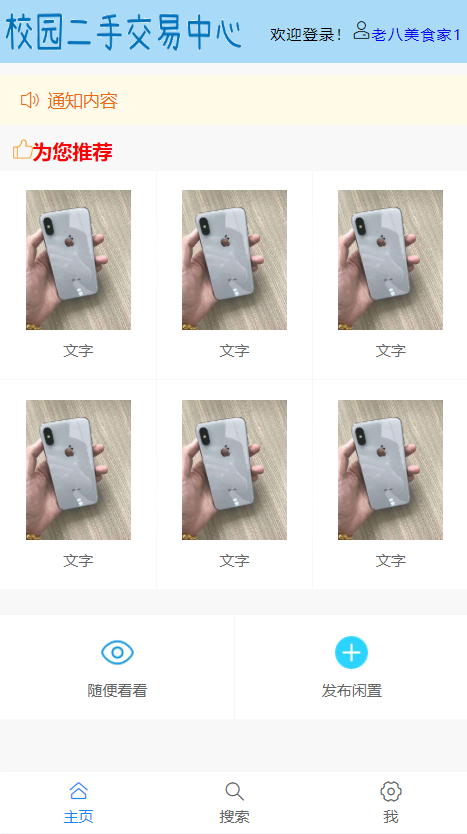
**图5.2.2 路由拦截代码**

**5.2.2 主页信息加载实现**

进入主页前，会通过判断用户个人信息是否存在来判断用户是否是第一次登录，如果是，则先跳转到添加个人信息的面，否则读取用户和商品的数据，渲染到页面。



**图5.2.3 主页加载代码**

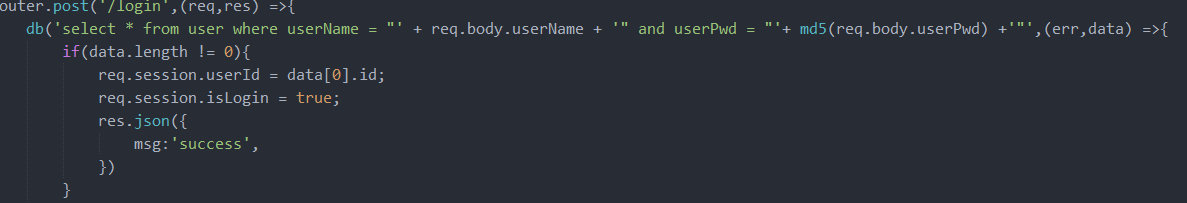


**图5.2.4 主页界面**

**5.3 个人信息模块实现**

**5.3.1 利用session+cookie实现信息存储**

众所周知，http是一种无状态的协议，服务器并不知道是谁发起的请求，因此我们需要让服务器端知道，是哪个用户发起的请求。通过session+cookie可以很好的解决这个问题。在用户登录成功的时候，服务器端生成一个session对象，里面存放着用户的数据，比如登录状态、id。



**图5.3.1 生成session对象**

Session对象生成之后，返回数据时，顺带会将sessionid传给前端，存放在cookie中。而因为在前端每一次向后台请求时，都会携带cookie。服务器端可以通过cookie获取sessionid，从而获取到相应的session对象，来判断是谁发送了请求，并进行操作返回相应的数据。



**图5.3.2 获取session数据**

**5.3.2 利用vuex保存公共信息**

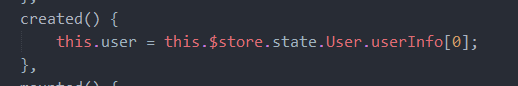
一般来说，都是在页面加载时，向后台请求数据，得到后台返回的数据并将其渲染到界面上。也就是每个页面加载时都需要发起请求，但是如果多个页面实际上只共用一套数据，那么这种做法实际上是在增加没必要的请求，同时增大了服务器的压力。因此，面对多页面使用同一套数据的情况，我们可以把共享的数据，放到vuex中。Vuex是一个vue提供的状态管理工具，可以把它理解成一个公共仓库，所有的修改，都在vuex中进行。而修改的数据，会立即反映到其他组件中，这样子大大减少没必要的请求，减缓了服务器的压力。



**图5.3.3 vuex结构代码**

**5.3.3 用户信息模块实现**

用户信息主要的获取主要就是通过vuex来获取，用户主页的信息。



**图5.3.4获取用户信息**



**图5.3.5用户信息界面**

当我们需要修改用户的信息时，我们可以通过操作vuex中的action模块，向后台发起更改数据的请求。

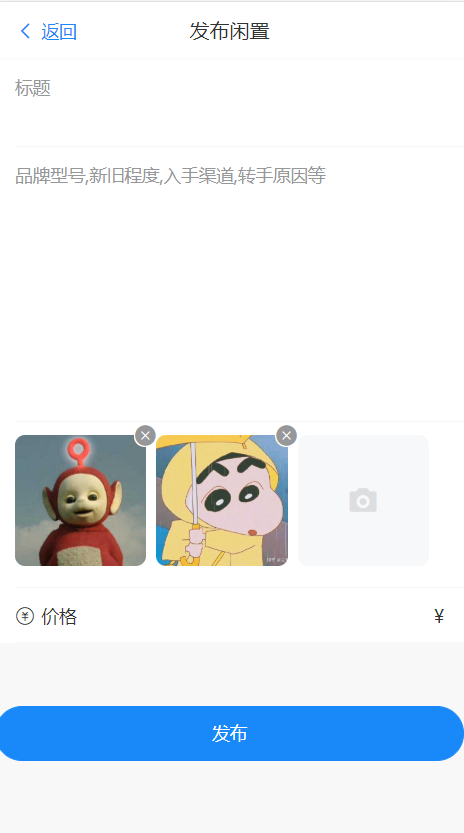


**图5.3.6更改用户信息**

**5.4 商品模块实现**

**5.4.1 uploder+multer模块实现商品发布**

闲置商品的发布，除了基本的一些文字信息之外，商品的图片是必须存在的。多图片的上传采用了uploader+multer进行实现。

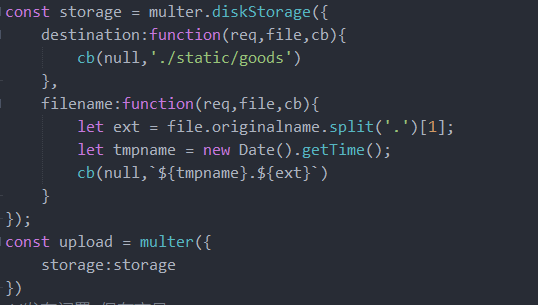


**图5.4.1发布闲置界面**

Uploader是前端框架vant提供的一个上传图片的插件，支持各种图片上传，上传图片预览等功能。并且获得每一个上传图片的对象，存放在一个数组中。我们把其转为formdata对象，发起请求，将这个对象传给服务器进行处理。

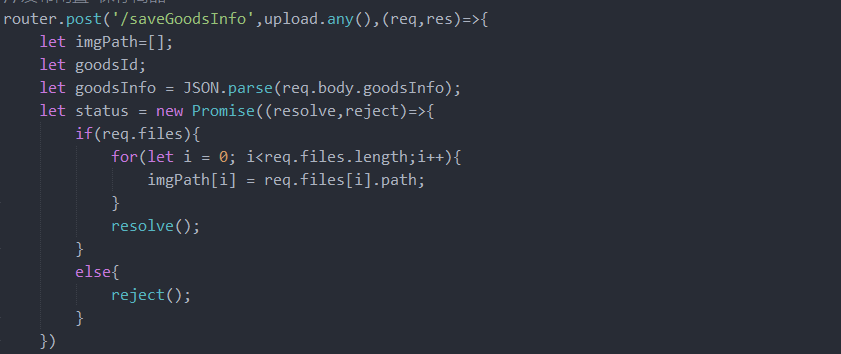


**图5.4.2发布闲置请求代码**

Multer是node提供的一个处理文件的模块，使用其处理文件之前，我们需要进行配置。

**图5.4.3 multer基本配置**

配置完之后，我们就可以将前端传过来的FormData对象重新转化为图片，并添加到本地。同时截获图片的路径，并将图片路径和其他数据存入数据库中。



**图5.4.4 multer图片处理**

**5.4.3 闲置信息列表实现**

用户的所发布的闲置信息会放在一个列表当中，我们可以通过浏览列表信息，先进一步了解该闲置信息，如果觉得合适，点击该项，跳转商品信息页面，进一步了解该闲置物品的信息。



**图5.4.5闲置信息列表**

商品列表使用了懒加载的效果，一次直询固定数量的数据，当你刷到底部时，会触发事件，再次请求剩下的数据，如果则接着渲染商品信息，否则则提示加载完成。



**图5.4.6列表懒加载代码**

**5.4.4 闲置商品信息实现**

商品信息显示了商品的基本信息，比如实拍图片、卖家信息等，当我们需要了解这个商品更为详细的信息，可以点击该页面查看商品详情的按钮，跳转到详情页，你还可以点击左下角的星号，把该商品添加到收藏列表中。



**图5.4.7闲置信息页面**



**图5.4.8卖家信息页面**



**图5.4.9闲置信息详情页面**



**图5.4.10闲置信息详情代码**

**5.5 交易模块实现**

交易主要展现了用户购买商品的记录和发布闲置物品的记录。还可以看订单的交易状态。

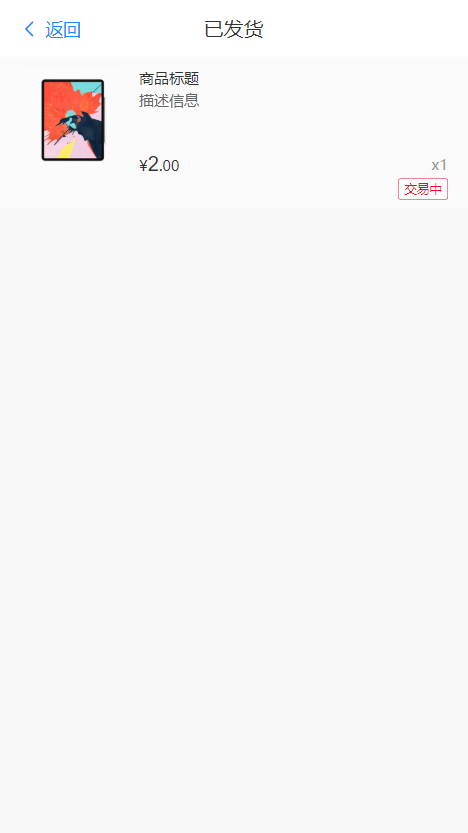
已出单记录的是用户发布的闲置信息交易记录，已下单是记录用户拍下的商品中处于交易状态的记录，而全部订单则显示了用户所有拍下的记录。



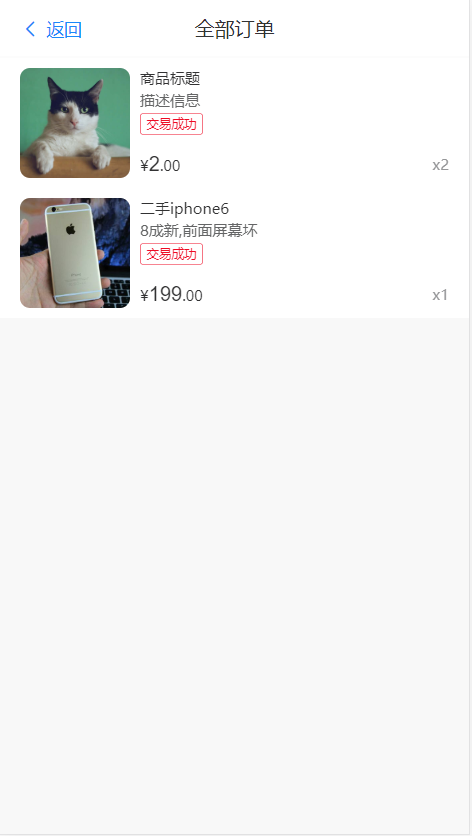
**图5.5.1交易信息管理**



**图5.5.1下单记录管理**



**图5.5.2发货记录管理**



**图5.5.3全部订单**



**图5.5.4已下单模块代码**

**6 系统测试**