****

本 科 毕 业 论 文（设 计）

**校园二手交易APP的设计与实现**

|  |  |
| --- | --- |
| **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：** | **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5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**

在大学生日益增强的购买力之下，每个人的生活中开始出现各种各样的闲置物品，而如何处理这些闲置物品是大学生们最为烦恼的一个问题。本课题开发了一款基于WebApp的校园二手信息平台。该平台采用前后端分离的开发方式，前端使用的是ES6、Vue.js，采用Vue-cli+WebPack构建前端项目，后端则是采用了node.js的集成框架Express，使用了nodemalier、multer等中间件。数据库采用MySQL。该APP提供了一个校内二手交易信息发布的平台，用户可通过平台出售自己的闲置物品，也可以通过平台了解到自己感兴趣的闲置物品，联系卖家最终在线下完成交易。该平台具有服务性强，效率高等特点。将服务人群范围缩小为校园内，交易速度快，可当场验货。大大避免了因商品问题而产生的售后问题，具有一定的实用价值。

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

**Abstract**

This paper develops a second-hand campus information platform based on webapp. The platform adopts the development mode of front-end and back-end separation. The front-end uses vue.js, one of the three frameworks in full swing, the back-end uses node.js, and the database uses mysql. The platform provides a platform for the release of second-hand trading information in schools. Users can sell their own idle goods through the platform, or they can learn about the idle goods they are interested in through the platform, and contact the seller to complete the transaction online. The platform has the characteristics of strong service and high efficiency. Reduce the scope of service population to campus, with fast transaction speed and on-the-spot inspection. It greatly avoids the after-sale problems caused by Commodity Problems and has certain practical value.

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

**目 录**

**[1 绪论 1](#_Toc28592)**

[1.1 开发背景与意义 1](#_Toc18148)

[1.2 开发方法 1](#_Toc6524)

**[2 系统开发技术与环境 3](#_Toc16201)**

[2.1 系统整体架构 3](#_Toc17044)

[2.2 WebApp 3](#_Toc12878)

[2.3 前端技术介绍 3](#_Toc12098)

[2.3.1 ES6 3](#_Toc26925)

[2.3.2 Vue.js 3](#_Toc10635)

[2.3.3 Less 4](#_Toc23745)

[2.3.4 Vant 4](#_Toc27166)

[2.3.5 Axios 4](#_Toc314)

[2.3.6 Webpack 5](#_Toc8639)

[2.4 服务器端技术介绍 5](#_Toc25685)

[2.4.1 Node.js 5](#_Toc28442)

[2.4.2 Express 6](#_Toc23141)

[2.5 数据库介绍 7](#_Toc11298)

[2.5.1 MySQL 7](#_Toc6334)

**[3系统分析 9](#_Toc5988)**

[3.1 需求分析 9](#_Toc20531)

[3.1.1 角色功能分析 9](#_Toc16631)

[3.1.2 功能流程分析 10](#_Toc12486)

[3.2 数据需求分析 12](#_Toc8180)

[3.2.1 数据流程 12](#_Toc18379)

[3.2.2 数据字典 13](#_Toc7317)

[3.2.3 E-R实体图 15](#_Toc29691)

**[4 系统设计 18](#_Toc31855)**

[4.1 模块设计 18](#_Toc18255)

[4.1.1 登录模块设计 18](#_Toc18791)

[4.1.2 主页模块设计 19](#_Toc11428)

[4.1.3 发布闲置模块设计 20](#_Toc20811)

[4.1.4 搜索模块设计 21](#_Toc5996)

[4.1.5 闲置商品模块设计 21](#_Toc2872)

[4.1.6 个人中心模块设计 22](#_Toc30202)

[4.2 课题难点解决方案设计 23](#_Toc22837)

[4.2.1 验证码功能 23](#_Toc9008)

[4.2.2 session+cookie保持用户操作 24](#_Toc4013)

[4.2.3 利用HTML5离线缓存存储数据 25](#_Toc14997)

[4.2.4 采用Vuex保存公共信息 26](#_Toc14767)

[4.2.5 使用Uploader+Multer实现文件上传 26](#_Toc2588)

**[5系统测试 28](#_Toc2067)**

[5.1 测试用例 28](#_Toc30501)

[5.1.1 注册页测试 28](#_Toc2814)

[5.1.2 密码更新页测试 29](#_Toc6267)

[5.1.3 登录页测试 29](#_Toc4187)

[5.1.4 主页测试 30](#_Toc5474)

[5.1.5 闲置发布测试 31](#_Toc5955)

[5.1.6 搜索模块测试 33](#_Toc6642)

[5.1.7 商品模块测试 34](#_Toc1891)

[5.1.8 个人信息模块测试 36](#_Toc29510)

[5.2测试总结 39](#_Toc5352)

**[6 结论 40](#_Toc14935)**

**[7 致谢 41](#_Toc15480)**

**[参考文献 42](#_Toc27257)**

**[附录A 43](#_Toc24637)**

**1 绪论**

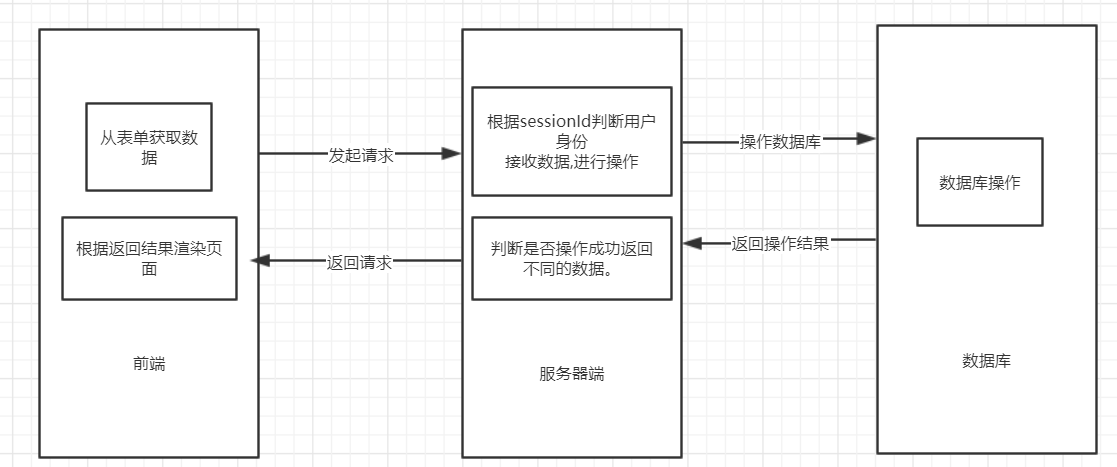
## 1.1 开发背景与意义

在当今时代发展过程中，手机、平板电脑等移动设备的发展速度令人叹为观止，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[1]。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。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[2]。

**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。其次对课题进行总体的架构设计。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[3]。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。



**图1.1 开发流程**

**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技术来快速构建前端项目。数据库则采用了MySQL。

**2.2 WebApp**

手机App分为Ios应用和Android应用，这两个系统互不兼容，采用了完全不同的开发语言进行开发[4]。在传统的开发方式当中，一个APP往往需要开发两套不同的代码。在开发成本、技术和时间花费上比较大。随着APP开发技术的不断发展，WebApp技术横空出世[5]。该技术使得APP可以基于Web端进行开发，我们可以用Web技术来编写APP，在编写完成之后，通过专业的打包工具，把APP打包成Android应用和Ios应用，实现一套代码，多端使用，大大减少了开发的成本[5]。

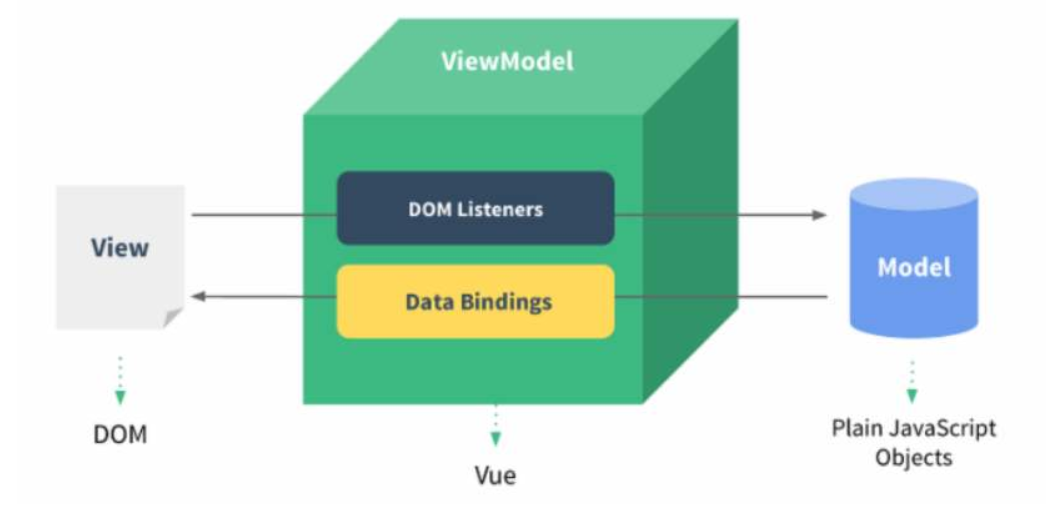
**2.3 前端技术介绍**

**2.3.1 ES6**

ES6是JavaScript新一代的标准，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。ES6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[6]。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。

**2.3.2 Vue.js**

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[7]。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，Vue.js的双向绑定如图2.1所示:



**图2.1 双向绑定原理**

**2.3.3 Less**

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，我们可以向写HTML代码的方式写CSS，方便快捷。

**2.3.4 Vant**

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。

**2.3.5 Axios**

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。

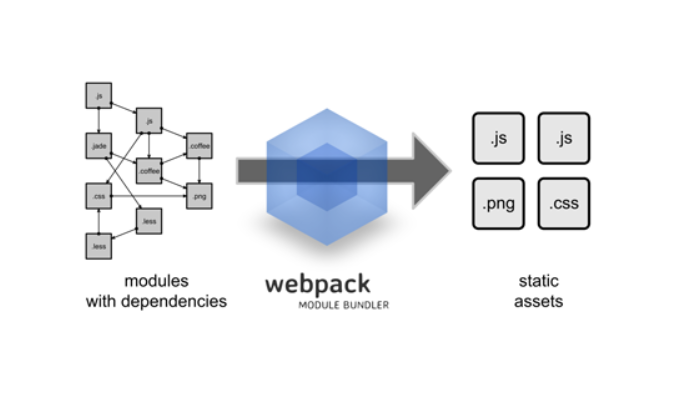
2.Axios能自动转换json数据，Jquery需要我们自己手动指定类型。

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。

**2.3.6 Webpack**

Webpack是一款自动化的模块打包工具。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[8]，Webpack的打包过程如图2.2所示:

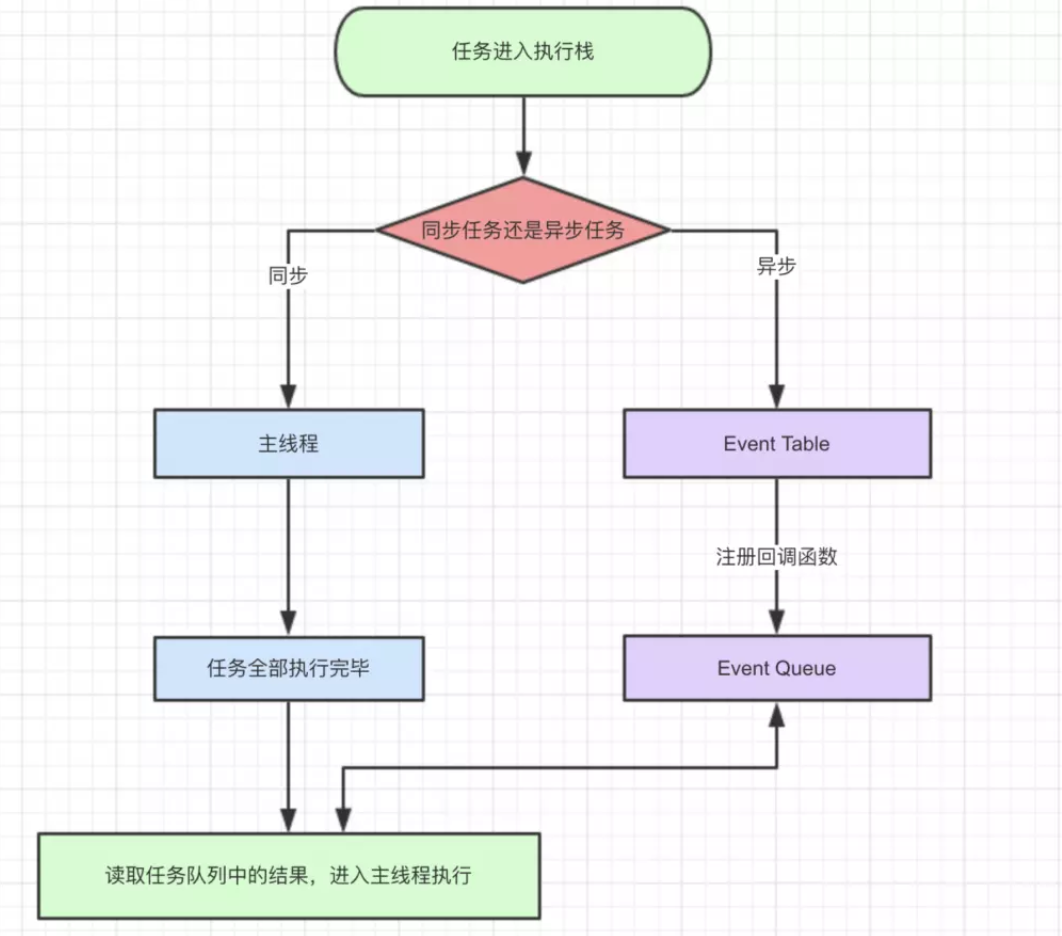


**图2.2 WebPack打包过程**

**2.4 服务器端技术介绍**

**2.4.1 Node.js**

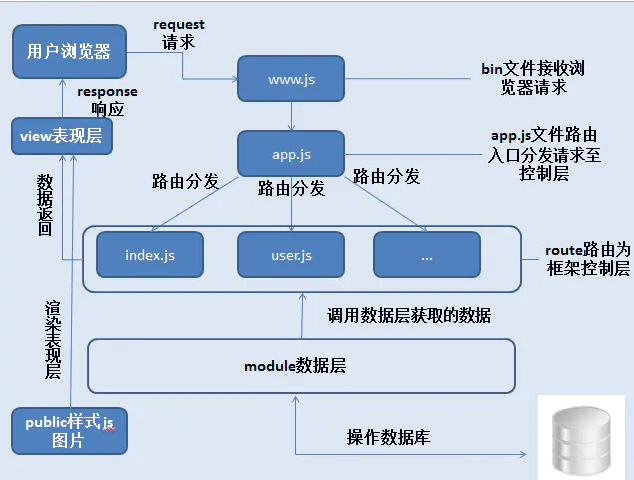
Node.js是一个让JavaScript运行在服务端的开发平台，它使得前端开发者可以采用JavaScript编写服务器端的语言，大大减少了前端开发者开发服务器端程序的学习成本。Java是一门多线程语言，而Node不同，Node属于单线程语言，但是Node采用了一个称为“事件循环(Event Loop）”的架构，使得编写可扩展性高的服务器变得既容易又安全[9]。Node时间循环机制如图2.3所示：



**图2.3 事件循环**

**2.4.2 Express**

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[10]。Express框架的流程如图2.4所示：



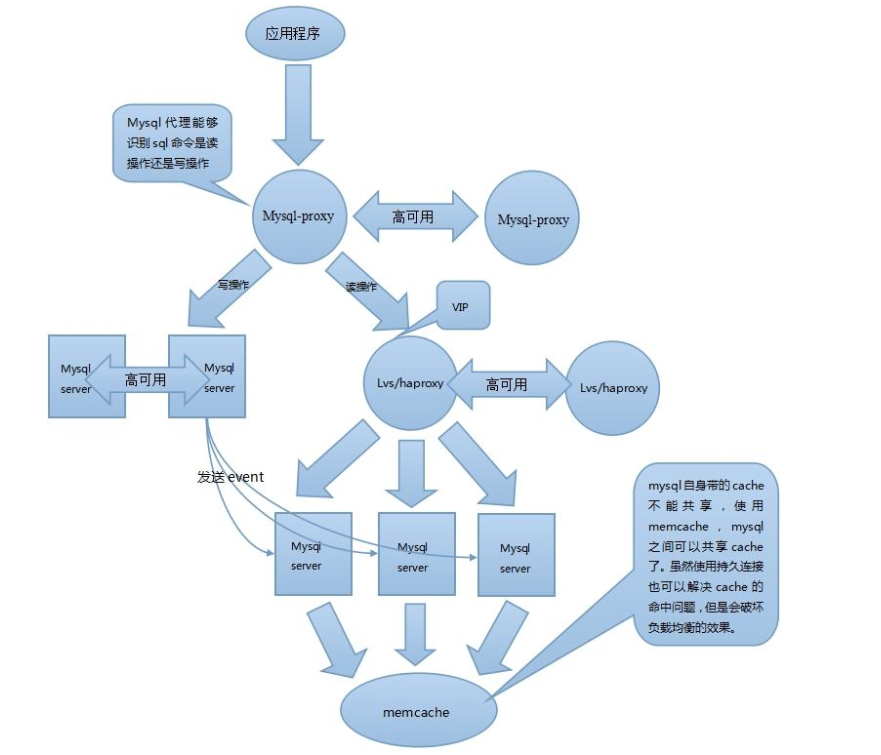
**图2.4 Express执行流程**

**2.5 数据库介绍**

**2.5.1 MySQL**

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[11]。

MySQL采用了标准的SQL格式，减少了学习成本。并且MySQL可以运行在Linux、Windows等多个平台上，支持多种编程语言。MySQL架构如图2.5所示：



**图2.5 MySQL架构**

**3系统分析**

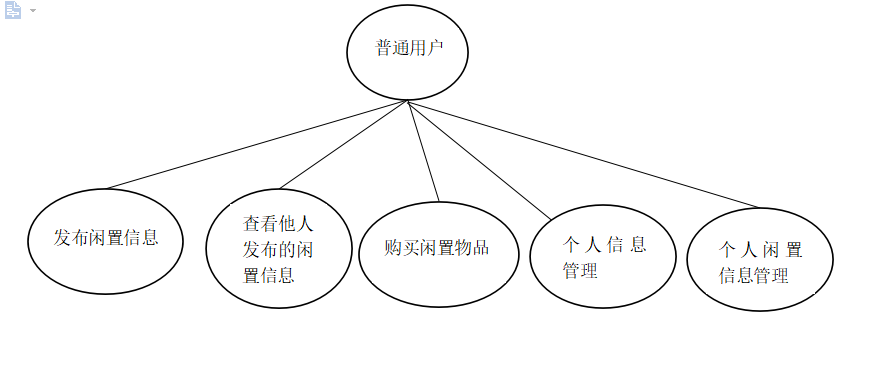
**3.1 需求分析**

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。

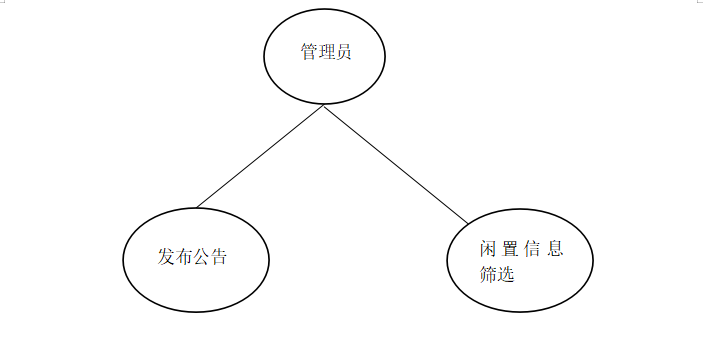
**3.1.1 角色功能分析**

普通用户应该具有的功能包括发布闲置物品、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，个人信息管理等，如图3.1所示：



**图3.1 普通户用例图**

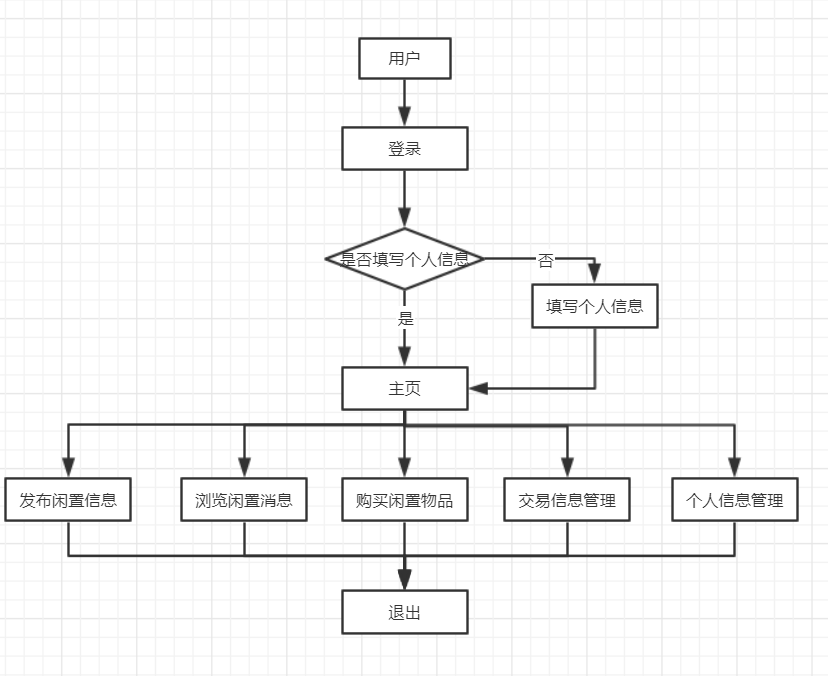
管理员应该具有的功能有发布公告、和筛选用户发布的闲置信息，防止不良信息发布，如图3.2所示：



**图3.2 管理员用例图**

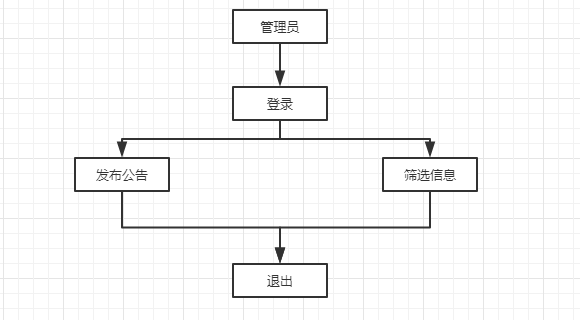
**3.1.2** 功能流程**分析**

在用户登录成功之后，如果是第一次登录，需要先填写个人信息。否则直接进入到主页，进入主页后，用户可以看到系统推荐的一些信息，可以通过主页跳转到闲置信息列表或者选择发布自己的闲置信息。搜索功能可以帮助我们更好的寻找我们所需要的闲置信息。当我们在闲置信息列表发现了心仪的物品时，可以点击到详情页查看更多的信息，觉得合适可以联系卖家进行交易。在个人信息管理中，用户可以看到自己收藏的闲置信息、以及自己发布的信息，还有个人信息的查看和修改。总体流程如图3.3所示：



**图3.3 普通用户流程图**

管理员登录平台之后，可以发布平台的公告。还有对用户发布信息的筛选，发现违禁的信息，可以通过数据库操作将其清除。流程大致如图3.4所示：



**图3.4 管理员流程图**

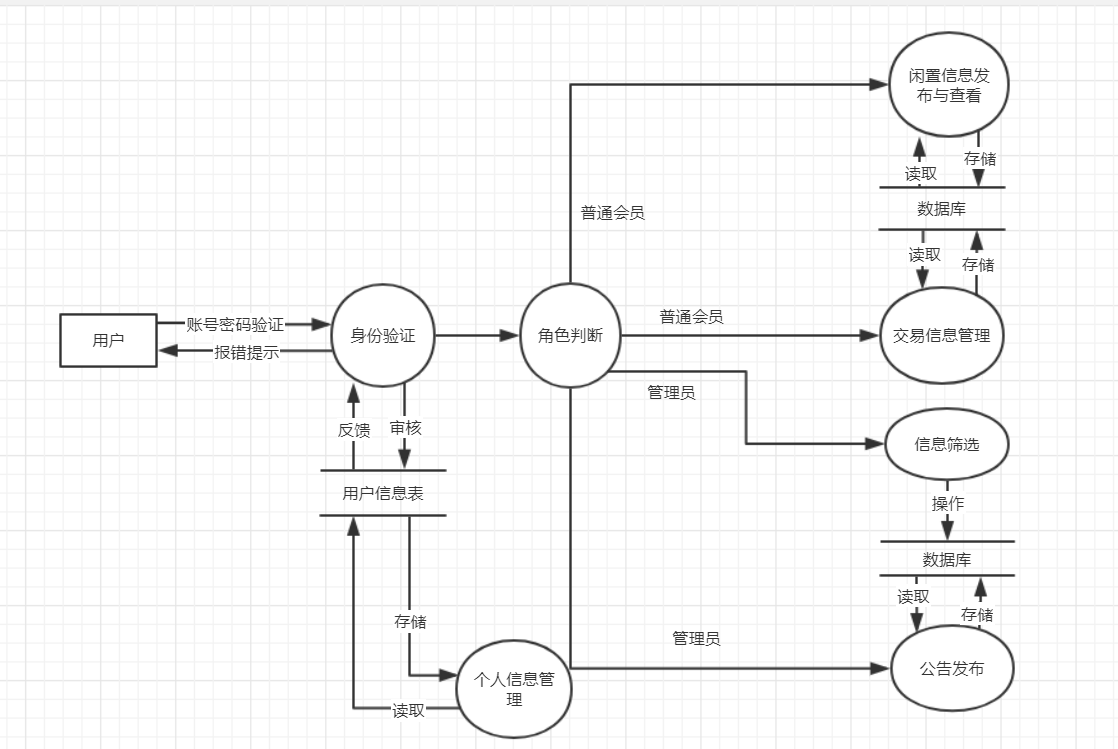
**3.2 数据需求分析**

**3.2.1 数据**流程

当用户注册账号成功的时候，会生成一个用户的id，并且将id和用户注册的账号、密码存进用户表中。当用户进行登录操作时，获得用户输入的账号和密码与用户表的信息进行匹配，如果查询到数据则登录成功。登陆成功后先进行角色判断，普通用户跳转到主页，而管理员则跳转到公告发布页。

当普通用户跳转到主页之前，先判断用户是否填写个人信息，如果没有会跳转到填写页面让用户填写所有个人信息，并将数据存入用户信息表中。用户闲置信息发布成功的时候，会将商品id，闲置信息、发布者id会存入到商品列表中。而用户收藏的闲置信息则会将商品id，收藏着id存储到收藏表当中。当用户修改个人信息的时候，会根据用户id来更改个人信息表的数据。而当用户对自己发布的信息进行管理时，则会根据用户id和商品id对闲置信息表进行修改。

当管理员公告发布成功的同时，会将公告id，公告内容存入公告表中，大致数据流图如图3.5所示：



**图3.5 项目数据流程图**

### 3.2.2 数据字典

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)，notice(公告表)。

用户表结构如表3.1所示：

**表3.1 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 |  |

用户信息表结构如表3.2所示：

**表3.2 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 |  |

商品信息表结构如表3.3所示：

**表3.3 good\_info表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| good\_id | varchar(20) | 商品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 |  |
| sell\_time | datetime | 发布时间 | 否 |  |

商品图片路径表结构如表3.4所示：

**表3.4 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) | 图片路径 | 否 |  |
| good\_id | varchar(20) | 商品id | 否 |  |

公告表结构如表3.5所示：

**表3.5 notice表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| notice\_id | init | 公告id | 否 |  |
| notice\_content | varchar(100) | 公告信息 | 否 |  |
| p\_id | varchar(20) | 发布者id | 否 |  |
| title | varchar(20) | 公告标题 | 否 |  |

用户收藏表结构如表3.6所示：

**表3.6 user\_collection表**

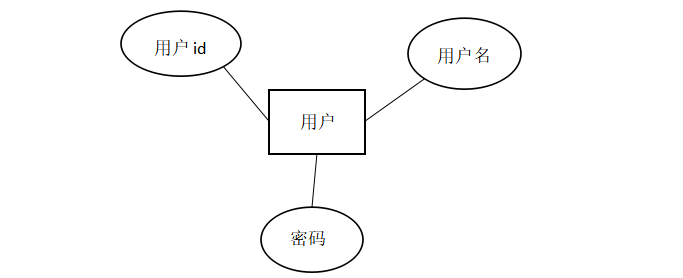
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段含义 | 是否为空 | 注释 |
| collection\_id | int | 收藏id | 否 |  |
| good\_id | varchar(20) | 商品id | 否 |  |
| user\_id | varchar(20) | 用户id | 否 |  |

### **3.2.3** E-R实体图

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。

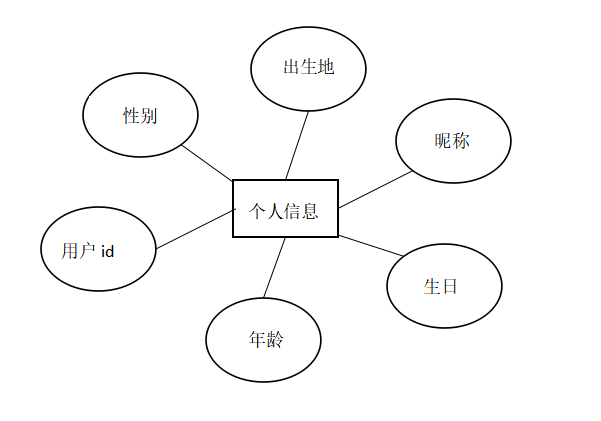
(1)用户实体图设计

用户实体就是针对普通用户和管理员设置id、用户名、密码。如图3.6所示：



**图3.6用户实体图设计**

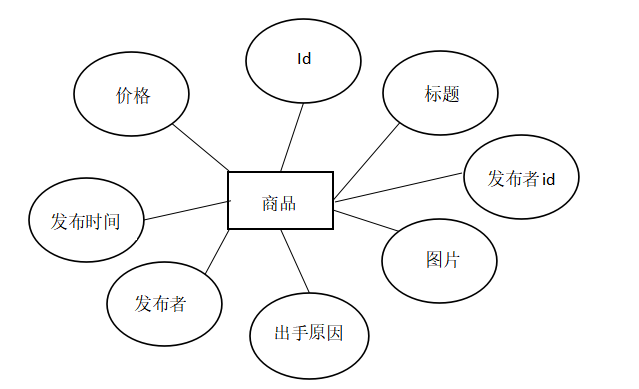
(2)个人信息实体图设计

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。如图3.7所示

**图3.7个人信息实体图设计**

(3)商品实体图设计

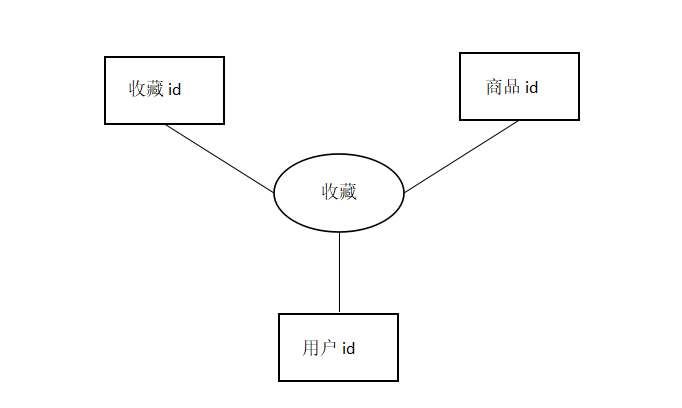
商品实体是普通用户所发布的闲置商品所具有属性，包括id、标题、图片、出售原因、价格等详细信息。如图3.8所示：



**图3.8商品实体图设计**

(4)收藏实体图设计

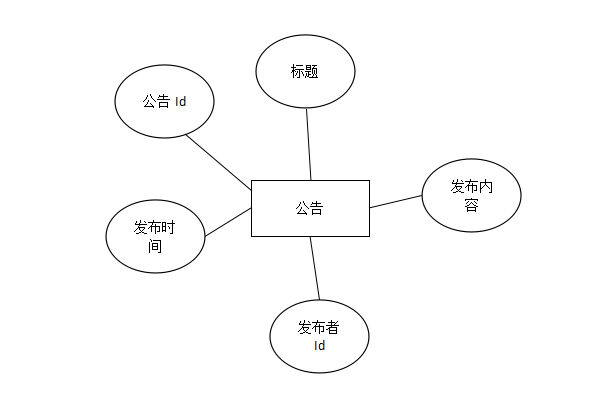
收藏实体是个人信息所具有的属性，具有收藏id，商品id，收藏者id。如图3.9所示：



**图3.9收藏实体图设计**

(4)公告实体图设计

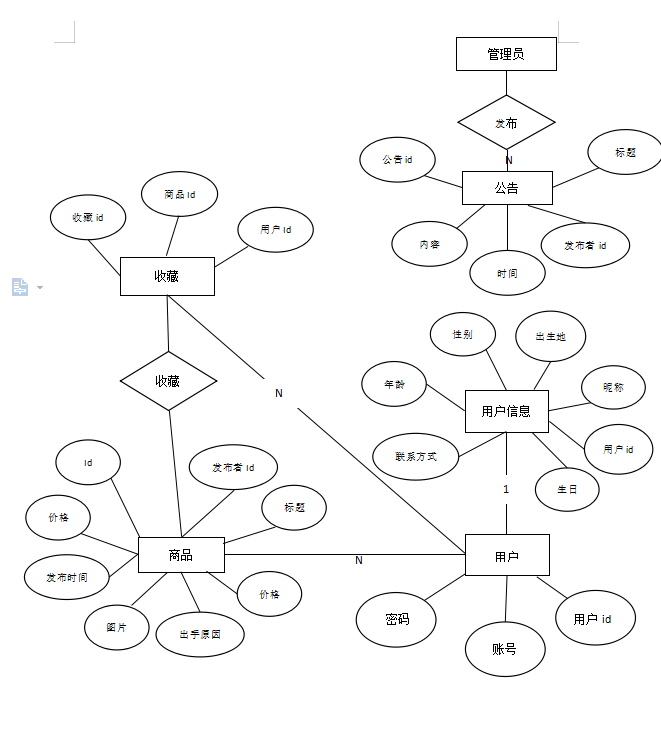
公告实体是公告所具有的属性，具有公告id，公告标题，发布者id，发布日期，发布内容。如图3.10所示：



**图3.10公告实体图设计**

(5)总体E-R图设计

总体E-R图设计如图3.11所示：

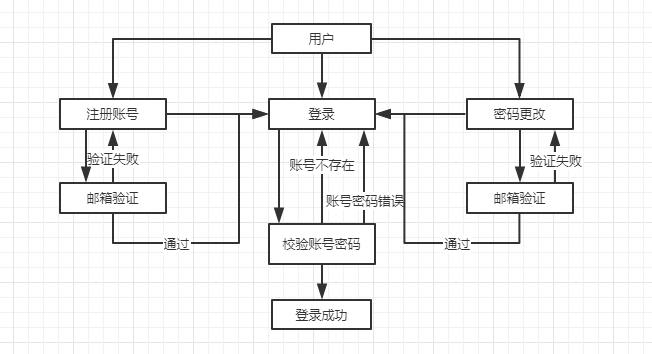


**图3.11总体E-R图设计**

**4 系统设计**

## 4.1 模块设计

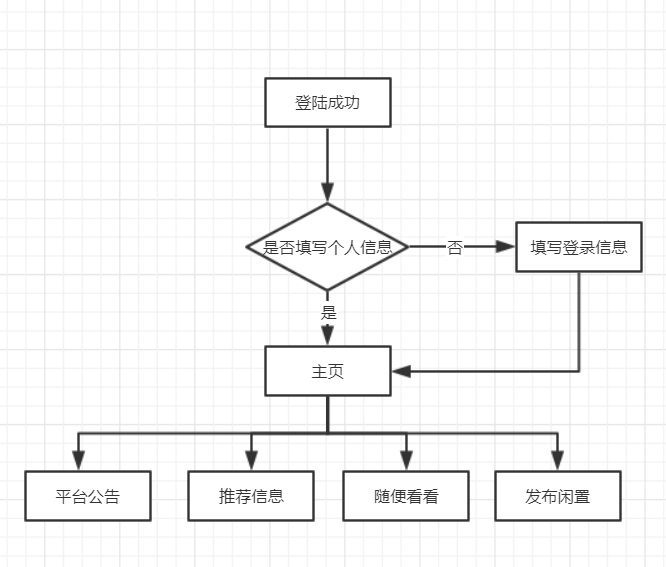
**4.1.1 登录模块设计**

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4.1所示：

**图4.1 登录模块设计图**

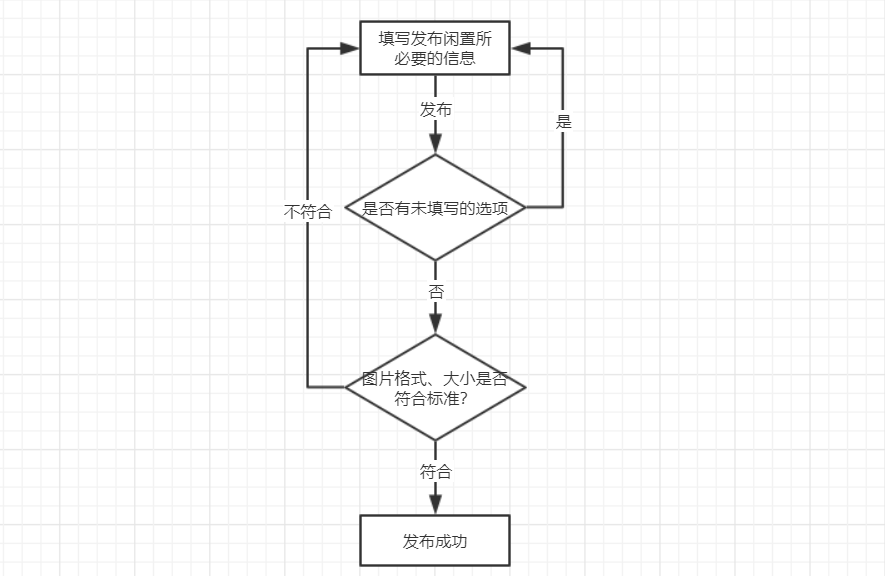
**4.1.2 主页模块设计**

账号登陆成功后要进入主页时，服务器端会通过该用户的id对数据库进行查找，如果查询到了用户信息，则判定该用户不是第一次登录，在获取用户数据的同时直接进入主页；若查询不到用户信息，则判定该用户为第一次登录，会先跳转到信息填写的页面，等待用户填写完信息之后，将信息保存到数据库中，然后跳转到主页。主页有平台管理员发布的公告，还要随机为用户推荐的闲置商品信息。用户可以通过主页跳转到闲置商品列表或者是选择发布自己的闲置商品，如图4.2所示：

**图4.2 主页模块设计图**

**4.1.3 发布闲置模块设计**

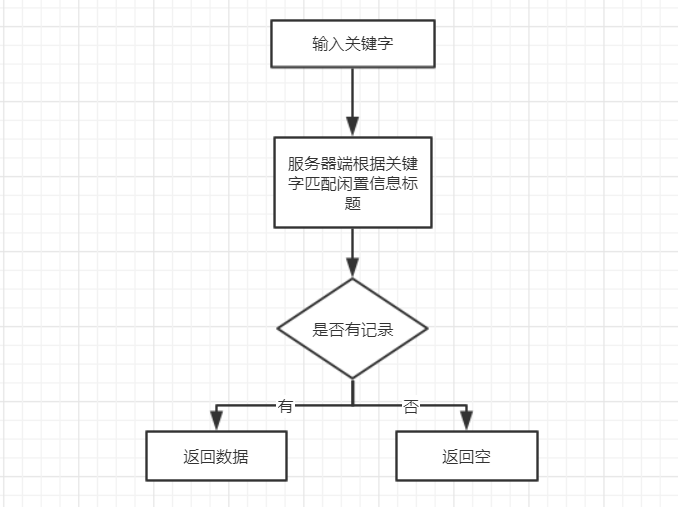
用户发布闲置信息，需要填写闲置信息的标题，闲置信息的原因，包括新旧程度，出售原因，破损情况等等。对于上传的图片做出了一定的限制条件，例如图片数量最多只能有三张，并且每张的大小不能超过500KB，这也是为了缓解服务器的压力。最后再写上自己出售的价格，点击发布按钮，即可发布。设计图如图4.3所示：



**图4.3 发布闲置模块设计图**

**4.1.4 搜索模块设计**

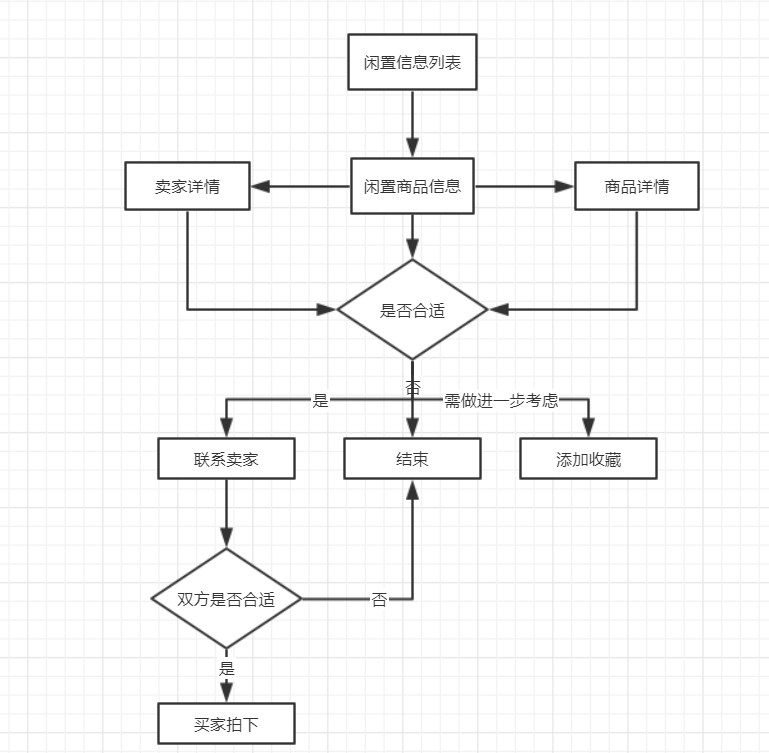
在种类繁多，数量庞大的闲置信息列表中，搜索模块能够帮助我们快速的查找到自己想要的闲置物品信息，用户只需要在搜素框输入关键字，点击确定，系统会自动匹配闲置信息标题，把含有该关键字标题的闲置信息显示出来，大大方便了用户的操作。设计图如图4.4所示



**图4.4 搜索模块设计图**

**4.1.5 闲置商品模块设计**

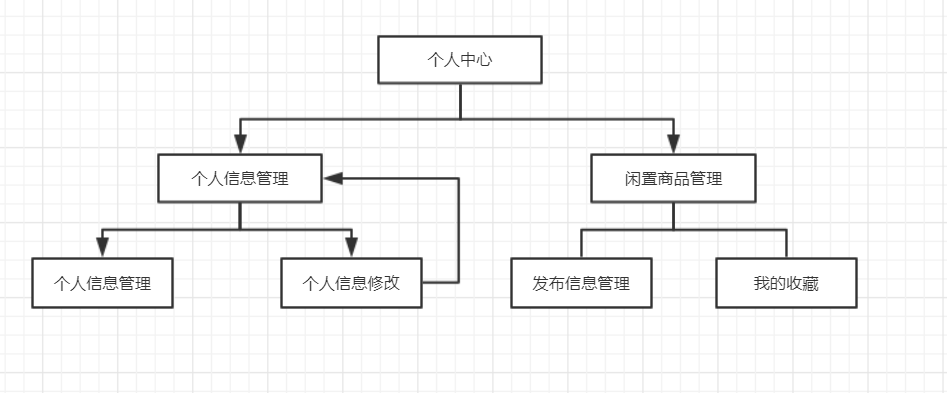
用户可以再信息列表中浏览其他用户所发布的闲置信息，如果看到了心仪的商品，可以选择点击该商品跳转到商品信息页面。商品信息页面中有着该商品更为详细的消息，还有卖家的详细信息提供查看，以获得卖家的联系方式，和商家进行联系，双方觉得合适的话买家就可以拍下该商品，进行交易。如果只是觉得感兴趣，但还需要进一步考虑，还可以考虑把这件商品收藏起来。设计图如图4.5所示：



**图4.5 闲置商品模块设计图**

**4.1.6 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收藏的闲置商品。用户可以在这里看到自己管理发布的所有商品信息以及状态，还可以看到自己收藏的商品。用户的个人信息如果需要作出修改，可以随时更改信息。设计图如图4.6所示：



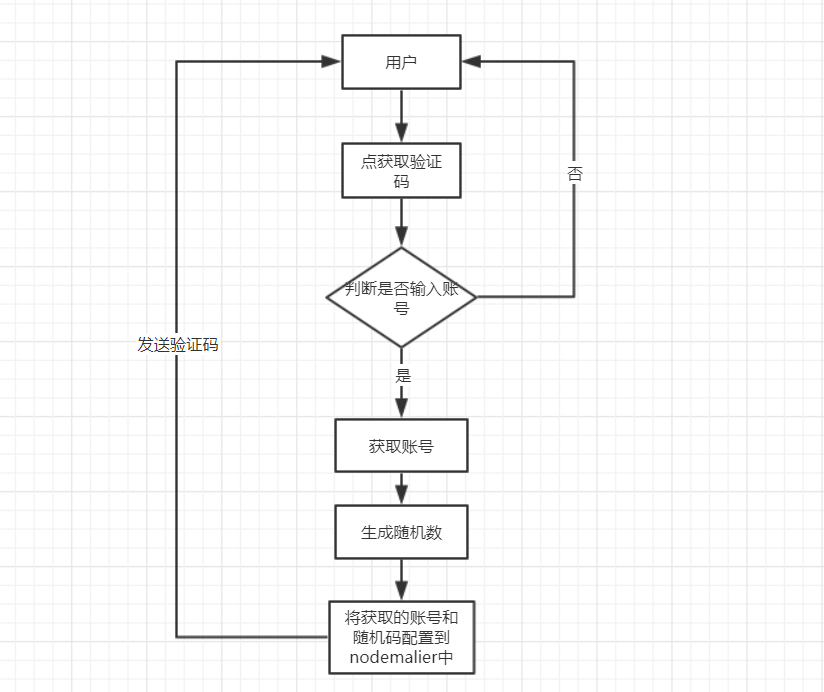
**图4.6 个人中心模块设计图**

## 4.2 课题难点解决方案设计

### 4.2.1 验证码功能

当今的注册方式基本都是采用手机号码或邮箱+验证码的方式进行校验。验证码的好处有两个方面。第一个是保证账号的有效性，验证码会被发送到你的手机号或者邮箱中，如果你的手机号或者邮箱不存在，则无法接收到验证码。第二个是保证用户操作的真实性，能够确保是用户在进行当前的操作。

本课题采用的是邮箱+验证码的方式进行账号注册和修改。采用了node.js的一个拓展插件来nodemalier实现邮箱验证码的发送。先从前端获取到用户的邮箱，然后在服务器端生成一个四位数的随机码，然后通过nodemalier发送到指定的邮箱中，用户就可以根据邮箱中的验证码来填写，完成注册或更改密码。过程如图4.7所示：

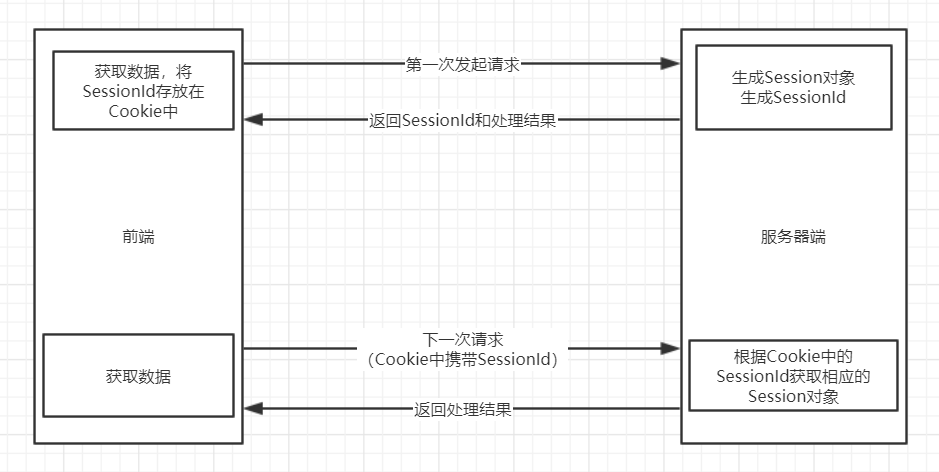


**图4.7 验证码功能设计**

### 4.2.2 session+cookie保持用户操作

http协议是一种无状态协议，所以服务器端不能判断接收到的请求是哪个用户发起的。因此，需要让给服务器端一个标识，使得服务器端可以辨别请求的来源。Session和Cookie很好的解决了这个问题。当用户第一次发起请求时，在服务器端，生成一个Session对象和与其相对应的SessionId。Session对象用来存放用户的一些重要信息，把SessionId返给前端保存在Cookie中。当用户下一次发起请求时，SessionId会跟随Cookie发送给服务器端，这时候服务端就可以根据Cookie中的SessionId来判断是哪个用户发起的请求了。

在本课题中，当用户发起登录请求时，若账号密码匹配成功，则会获得该用户的id和生成一个判断用户是否登陆的标识符，生成一个Session对象来存放这两个数据。然后将Session对象所对应的SessionId作为返回数据发送给前端，前端接收到后将其存放在Cookie中，当前端向服务器发起其他请求时，Cookie也会作为参数之一传到服务器端。服务器端可以从Cookie中获取SessionId，根据SessionId得到对应的Session对象，从而判断是哪个用户进行了操作，然后进行相应的数据处理。过程如图4.8所示：



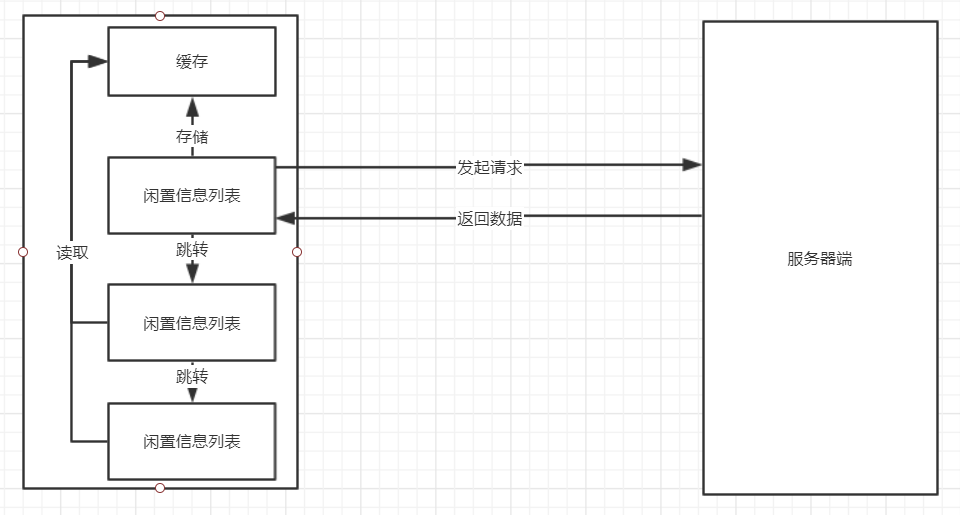
**图4.8 Session+Cookie数据流转**

### 4.2.3 利用HTML5离线缓存存储数据

对于一个web项目来说，频繁的请求会对服务器产生巨大的压力。因此，如果是一些数据不需要改变或者改变次数很少的情况下，前端可以在第一次请求到数据的时候，将其存放在缓存中，下一次直接调用缓存中的数据，不用发起请求，减少了服务器的负担。

前端目前常用的缓存方式有SessonStorage、LoaclStorage、Cookie等。SessonStorage主要是用于暂时缓存数据，当用户退出平台之后缓存就会消失。LoaclStorage则是将数据直接缓存到用户端，可永久访问。Cookie适用于缓存一些定时清除的数据。根据不同的场景采取不同的缓存方式，能够对性能优化起到很好的效果。

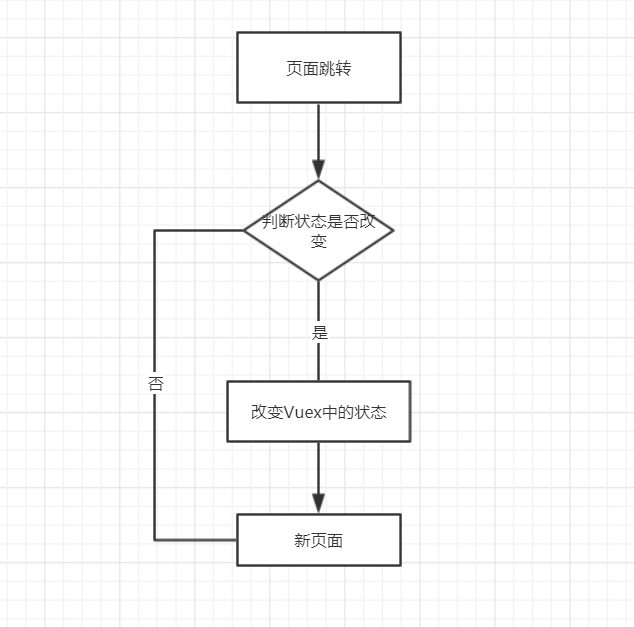
本课题在商品模块使用HTML5的SessionStorage缓存了闲置信息的数据。当用户点击列表中的内容时，会发起请求获得该闲置信息的数据用SessionStorage缓存起来，跳转到信息页，信息页在加载的过程中直接读取缓存的数据并且渲染出来。而当信息页跳转到详情页时，也采取从缓存中读取数据的方式渲染页面。如果用户从列表点击了另一条闲置信息，就会发起新的请求将获取到的新数据把原来的缓存数据覆盖掉。因为商品模块之间很多数据可以共用，所以我们可以缓存起来，而不是进入到一个新页面就向服务器请求一次。减少了不必要的请求，减缓了服务器的压力。过程如图4.9所示：



**图4.9 HTML5缓存流程设计**

### 4.2.4 采用Vuex保存公共信息

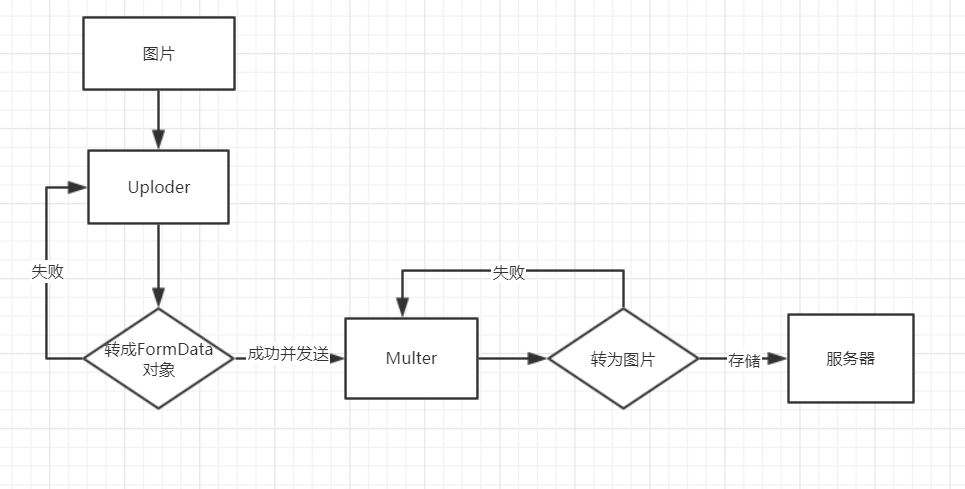
Vuex是Vue.js为我们提供的一个状态管理工具。本质上就是一个共用仓库。所有的组件都可以获取到Vuex中的值，我们可以通过Watch或者是Computed去监听这个状态值的变化，当其中一个组件改变了这个状态值的时候，其他的组件会立马同步更新数据。该工具广泛用于购物车，顶部标题栏改变等使用场景，是单页面应用的核心工具之一。本课题利用Vuex来实现用户标题栏、控制底部导航栏的的变化，一开始先给标题栏和底部导航栏定义一个初始状态，表示进入的第一个页面是否需要显示出来。当跳转页面的时候，根据页面是否需要显示来改变状态，就能够做到在页面切换的时候控制标题栏和导航栏的显示效果。过程如图4.10所示：



**图4.10 Vuex实现状态改变**

### 4.2.5 使用Uploader+Multer实现文件上传

Web应用开发中，文件上传和下载功能是非常常用的功能。但是实际上，从前端上文件到服务器端中间的过程非常的繁琐，需要进行编写代码各种各样的文件操作，实现起来难度较大且需要花费不少时间。本课题使用了Uploader+multer来进行闲置信息图片的上传。Uploader是前端框架Vant提供一个上传文件的组件。可以自动把文件转化成FormData类型的对象，通过Ajax直接把这个文件对象传输到服务器端。Multer是Node.js提供的一个拓展插件，这个插件专门处理FormData对象，可以FormData对象转成文件，并且存储到本地中。过程如图4.11所示：



**图4.11 实现图片上传**

**5系统测试**

## 5.1 测试用例

### 5.1.1 注册页测试

注册页测试结果如表5.1所示：

**表5.1 注册测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试数据 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  密码789456123  验证码 4525 | 显示注册成功  跳转回登录页 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  密码123456789  验证码 7812 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  密码888888888  验证码2036 | 正常 |



**图5.1用户注册界面**



**图5.2邮箱验证码**

### 5.1.2 密码更新页测试

密码更新页测试结果如表5.2所示：

**表5.2 更新密码测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试数据 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  更改密码qwer1234  验证码 9632 | 显示更新密码成功  跳转回登录页 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  密码987654321  验证码 1003 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  密码00000000  验证码1569 | 正常 |

### 5.1.3 登录页测试

登录页测试结果如表5.3所示：

**表5.3 登录测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试数据 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  密码qwer1234 | 登陆成功，跳转主页 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  密码987654321 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  密码00000000 | 正常 |



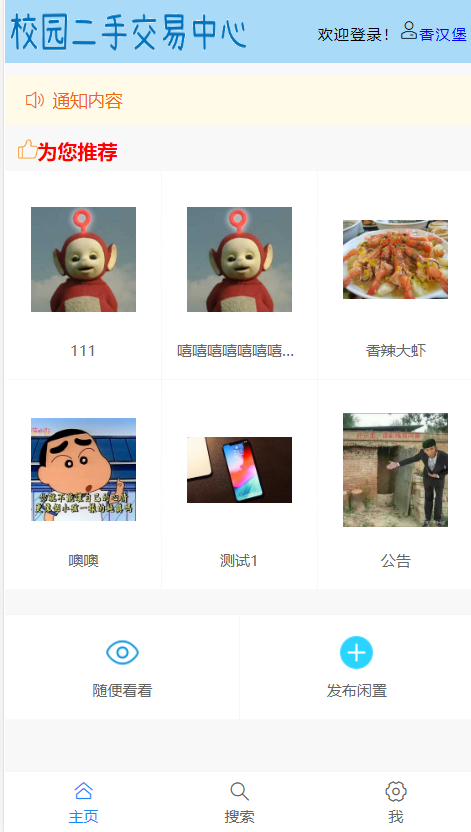
**图5.3用户登录界面**

### 5.1.4 主页测试

主页测试结果如表5.4所示：

**表5.4 主页测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试账号 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户1 | 1.第一次登录会有拦截，跳转到填写信息页面。填写成功后跳转到主页  2.主页坐左上角显示用户名，用户推荐显示出随机信息  3.公告栏显示管理员发布的消息 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户2 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户3 | 正常 |



**图5.4用户主页**

### 5.1.5 闲置发布测试

闲置发布测试结果如表5.5所示：

**表5.5 闲置发布测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试账号 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户1 | 显示发布闲置成功，发布的物品可以再闲置列表上看到，并且图片能显示出来。 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户2 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户3 | 正常 |



**图5.5闲置信息发布**

### 5.1.6 搜索模块测试

搜索测试结果如表5.6所示：

**表5.6 搜索测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试数据 | 预期结果 | 测试结果 |
| T1 | 搜索关键字：手机 | 所有标题包含关键字的闲置信息全部显示出来。 | 正常 |
| T2 | 搜索关键字：衣服 | 正常 |
| T3 | 搜索关键字：篮球 | 正常 |



**图5.6闲置信息搜索**

### 5.1.7 商品模块测试

商品模块测试结果如表5.7所示：

**表5.7 商品模块测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试账号 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户1 | 1.从闲置列表点击跳到信息页数据显示正确。  2.点击查看详情跳转到详情页，数据和图片均显示正确。  3.点击卖家详情，成功跳转到卖家详情页并且数据显示正确  4.点击我的收藏，显示收藏成功。在我的收藏可以看到收藏的物品。 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户2 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户3 | 正常 |



**图5.8闲置信息**



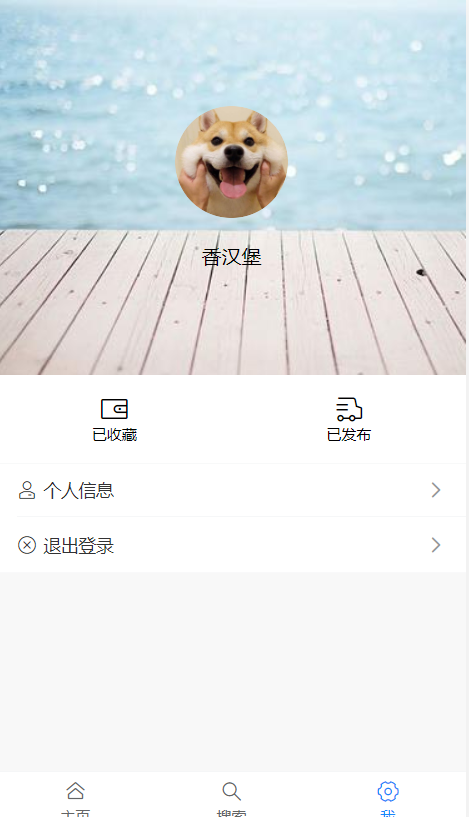
**图5.9闲置信息详情**

### 5.1.8 个人信息模块测试

个人信息测试结果如表5.8所示：

**表5.8 个人信息测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试账号 | 预期结果 | 测试结果 |
| T1 | [账号1292082437@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户1 | 1.个人主页显示自己的头像以及用户名。  2.在已发布可以看到自己发布的信息，可以删除自己发布的信息。  3.可以操作自己收藏的闲置信息  4.点击个人信息，数据显示正确，修改信息后提示修改成功。回到主页已经替换成修改后的数据。 | 正常 |
| T2 | [账号1635335844@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户2 | 正常 |
| T3 | [账号401612508@qq.com](mailto:账号1292082437@qq.com)  用户名：测试用户3 | 正常 |



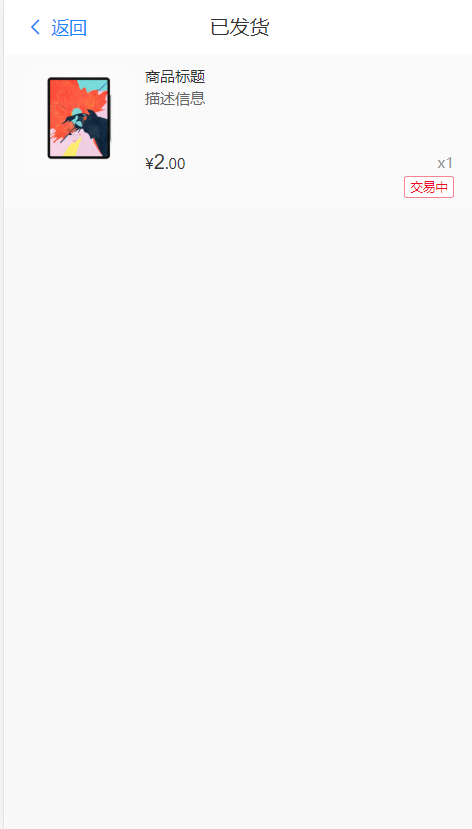
**图5.9 个人主页**



**图5.10 个人信息修改**



**图5.11 收藏列表**

  
 **图5.12 发布信息管理**

**5.1.9 发布公告模块测试**

发布公告模块测试结果如表5.9所示：

**表5.9 个人信息测试表**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 测试账号 | 预期结果 | 测试结果 |
| T1 | [账号1](mailto:账号1292082437@qq.com)0001  用户名：测试管理员1 | 发布公告显示发布成功。主页显示最新的公告通知。 | 正常 |

## 5.2测试总结

本次测试总结如下：

1. 登录模块所有功能正常，注册账号和更新账号邮箱验证码成功发送。前端与后端的数据交互无任何问题，登陆成功之后，普通用户会进入主页，管理员则跳转到发布公告界面。
2. 普通用户进入主页后，用户如果是第一次登录则会跳转到个人信息填写页面，否则就直接跳转到主页。
3. 在发布闲置的模块中，图片能够成功上传，支持图片预览。发布成功后可以在闲置信息列表看到刚刚发布的闲置信息。
4. 从闲置信息列表跳转到闲置信息页面再到详情页，数据都能够精确显示。
5. 个人模块信息显示正常，更改个人信息之后，会立马得到更新。在个人收藏可以管理自己收藏的闲置信息，在我的发布可以管理自己发布的闲置信息。
6. 管理员添加公告成功后，主页公告栏会出现最新的一条公告。

**6 结论**

本课题基于WebApp为基础，采用Node.js、Vue.js、Webpack、MySQL等技术来实现该课题的设计和开发。采用前后端分离的开发模式，后端只负责提供接口，前端只负责调用接口，极大降低了耦合度。Vue.js使得前端只需要关心数据方面的操作，而不需要考虑如何操作DOM，而Webpack能够将整个前端项目压缩打包，使其变得体积更小、性能更优。Node.js的非阻塞特性能很好的解决并发请求的问题。由于时间和技术方面的限制，该项目的功能并不是特别的完善。该系统还可以做进一步的开发，比如，增加聊天系统、线上交易系统等，将这个平台拓展成一个线上购物的平台，具备更加强大的功能。

在经历两个半月的不懈努力下成功完成。我们独立实现了将一个产品从0到1的开发。从确定课题，到技术选型，再到程序的编写，最终成功的完成了本课题的设计，这其中的每一个环节，都对自己有着极其重要的意义。通过这个毕业设计，我将我学的知识运用到了实际中，而不是单纯的掌握了理论，大大提高了我的动手能力。在这个过程中，也遇到过了许许多多的问题，通过请教老师、上网查阅资料等方式，克服了一个个的困难。其实人生也就是如此，唯有通过不懈努力爬上顶峰，你才能够看到更广阔的、更美丽的世界！

**7 致谢**

从一开始课题的选择，学习论文的编写、资料的查找、论文结构的设计、到初稿成型，然后再加以修改，最终得以定稿。这是一个繁琐而又艰苦的过程，在即将为这篇论文划上句号的时候，我的心中有无限的感慨。体会到了学术论坛上那么多篇论文从开题到正式发布、其中的过程是有多么的不容易。我要向我的指导老师王中圣表达我最衷心的感谢。他在百忙之中抽空帮我点评论文，他耐心的指导我、教我怎么去完成一篇优秀的论文,没有王中圣老师的指导，就没有我这篇论文的完成！我要再次向王老师说一句：您辛苦了！

金无足赤人无完人，我的论文还有很多的不足之处，希望各位老师的批评和指正！

参考文献

1. 杨周，周磊.大学生二手物品交易调查分析[J].产业与科技论坛，2013(12).
2. 孙孺石.智能手机操作系统[M].人民邮电出版社, 2016.
3. 储久良.Web前端开发技术课程教学改革与实践[J].计算机教育，2015
4. 俞华锋.基于HTML5的网页设计与实现[J].科技信息，2015（29）
5. 刘春华.基于HTML5的移动互联网应用发展趋势[J].移动通信，2017（09）：64-68
6. 阮一峰.ES6标准入门[M].中国工信,2017,27(2):20-24.
7. 刘博文.深入浅出Vue.js[M].人民邮电出版社.2019
8. 吴浩鳞.深入浅出Webpack[M].电子工业出版社.2017
9. 尤嘉.Node.js进阶之路[M].清华大学出版社.2018
10. Ethan Brown.Node与Express开发[M].人民邮电出版社.2018
11. 西泽梦路.MySQL基础教程[M].人民邮电出版社.2020

**附录A**

邮箱验证码实现代码

const config = {

// 163邮箱 为smtp.163.com

host: 'smtp.qq.com',//这是qq邮箱

//端口

port: 465,

auth: {

// 发件人邮箱账号

user: '1292082437@qq.com',

//发件人邮箱的授权码 这里可以通过qq邮箱获取 并且不唯一

pass: 'qnrwyloraunubada'

}

}

const transporter = nodemailer.createTransport(config)

//发送验证码的方法

async function send(code,email){

const mail = {

// 发件人 邮箱 '昵称<发件人邮箱>'

from: '校园二手交易注册中心<1292082437@qq.com>',

// 主题

subject: '验证码',

// 收件人 的邮箱 可以是其他邮箱 不一定是qq邮箱

to: email,

// 内容

text: `您的验证码是${code}` ,

//这里可以添加html标签

html: ''

}

result = await transporter.sendMail(mail)

return result;

}

Mysql连接配置

const mysql = require('mysql')

const client = (sql, arg, callback) => {

//1.创建连接

let config = mysql.createConnection({

host: 'localhost',

user: 'root',

/\* password: 'root', \*/

password: 'qwer1234',

database: 'school\_secondhand'

})

//2.开始连接

config.connect()

//3.对数据库进行增删查改

config.query(sql, arg, (err, data) => {

callback && callback(err, data)

})

//4.关闭数据库

config.end()

}

Multer实现文件上传

var storage = multer.diskStorage({

destination:function(req,file,cb){

cb(null,'./static/user')

},

filename:function(req,file,cb){

let ext = file.originalname.split('.')[1];

Let tmpname = req.session.userId+(parseInt(Math.random()\*10000)).toString();

cb(null,`${tmpname}.${ext}`)

}

});

var upload = multer({

storage:storage

})

CROS跨域

app.all('\*', function(req, res, next) {

res.header("Access-Control-Allow-Origin", req.headers.origin);

res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");

res.header("Access-Control-Allow-Methods","PUT,POST,GET,DELETE,OPTIONS"

res.header("Access-Control-Allow-Credentials",true);

res.header("X-Powered-By",' 3.2.1')

res.header("Content-Type", "application/json;charset=utf-8");

next();

});

VUEX保存用户数据

getUserInfo(context){

return new Promise((resolve,reject)=>{

axios.post('http://localhost:3000/index/getUserInfo')

.then((response) =>{

if(response.data.code === -1){

resolve('isNotLogin')

}

else if(response.data.success === 1){

context.commit('saveUserInfo',response.data.data)

resolve('success')

}

else if(response.data.success === 0){

resolve('noMessage')

}

else{

reject()

}

})

.catch(() => {

reject()

})

})

},

updateUserInfo(context,param){

return new Promise((resolve,reject)=>{

axios.post('http://localhost:3000/userInfo/updateUserInfo',param)

.then((res)=>{

if(res.data.success === 1){

resolve('success');

}

else{

reject();

}

})

.catch(()=>{

reject();

})

})

}

}