## 项目介绍

**背景**

当前，互联网+经济发展势头强劲，运用“互联网”模式建立起一套多样化、可持续性、完整的销售模式，是促进乡村经济发展的一种新手段。这为乡村[产业升级](https://www.zhihu.com/search?q=%E4%BA%A7%E4%B8%9A%E5%8D%87%E7%BA%A7&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7b" \t "https://www.zhihu.com/question/_blank)、精准提供产品服务开辟了新渠道，意义重大。乡村互联网+经济的兴起，促使传播领域逐渐形成了主体多样、交融复杂的媒体传播生态系统，在通过网络将农产品销往各地的同时，产地的产品文化也随之同行、[民俗文化](https://www.zhihu.com/search?q=%E6%B0%91%E4%BF%97%E6%96%87%E5%8C%96&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7b" \t "_blank)也随之发扬光大，而农产品附带的文化要素也让产品实现了增值，两者相辅相成、相得益彰。

农产品通过卖家和买家的直接对接，省去了中间环节的成本，既保证了产品的新鲜度、又节省了成本，在带给买家福利的同时，又通过农产品的损耗增加农民的收入，为民生作出保障。

**项目需求分析**

|  |  |  |
| --- | --- | --- |
| 模块 | | 介绍 |
| 登录 | | 用户名、密码 |
| 注册 | | 姓名、账号、电话、默认地址、密码 |
| 购买 | 搜索 | 搜索目标品类产品 |
| 查看详情 | 查看目标商品详情页 |
| 购买产品 | 确定心仪商品后下单购买 |
| 个人中心 | 历史记录 | 查看已购买商品的全部历史记录 |
| 个人信息 | 查看用户信息 |
| 修改信息 | 修改用户个人信息 |

表1-1 买家功能实现

|  |  |  |
| --- | --- | --- |
| 模块 | | 介绍 |
| 登录 | | 用户名、密码 |
| 注册 | | 姓名、账号、电话、默认地址、密码 |
| 商品模块 | 发布 | 商家发布商品 |
| 处理 | 处理未完成交易的商品，可以选择冻结、重新上架、下架 |
| 查看 | 查看历史商品的信息 |

表1-2 卖家功能实现

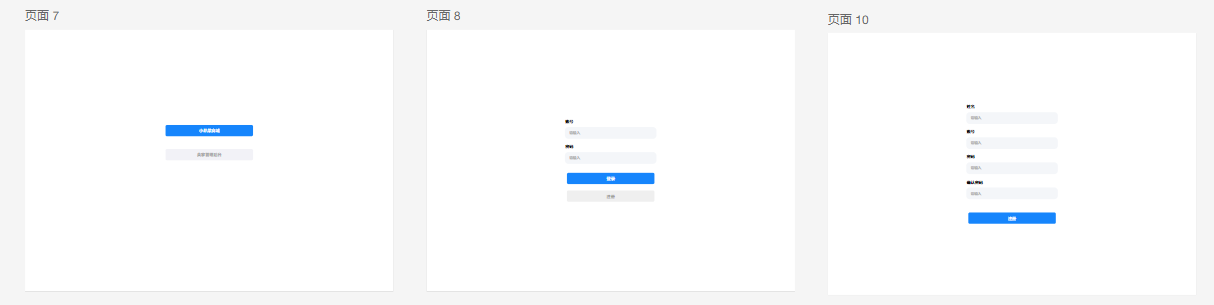
|  |  |  |
| --- | --- | --- |
| 模块 | | 介绍 |
| 管理员 | 查看用户信息 | 查看已注册的用户的信息 |
| 查看历史订单 | 查看所有历史订单 |
| 密码 | 修改密码 |

表1-3管理员功能实现

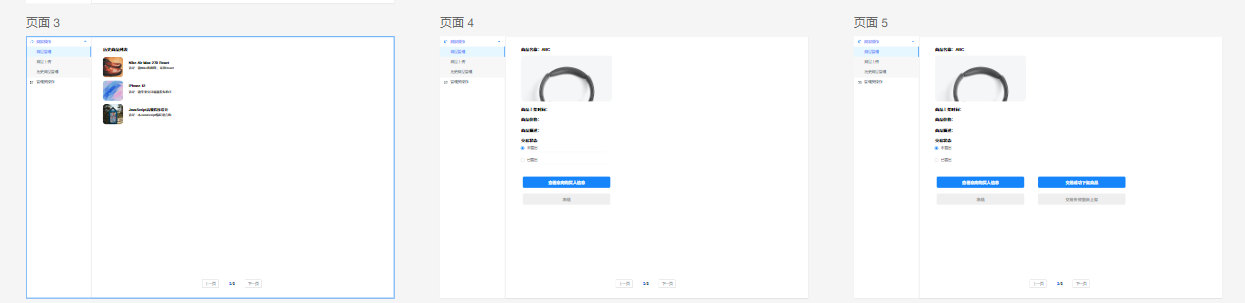
**项目分工**

|  |  |  |
| --- | --- | --- |
| 人员 | 负责工作 | 贡献度 |
| 丛翊旻 | 前端设计、单元测试 | 1/3 |
| 侯安琪 | 原型设计、架构设计、数据库设计 | 1/3 |
| 陆悠非 | 后端编写、数据库设计 | 1/3 |

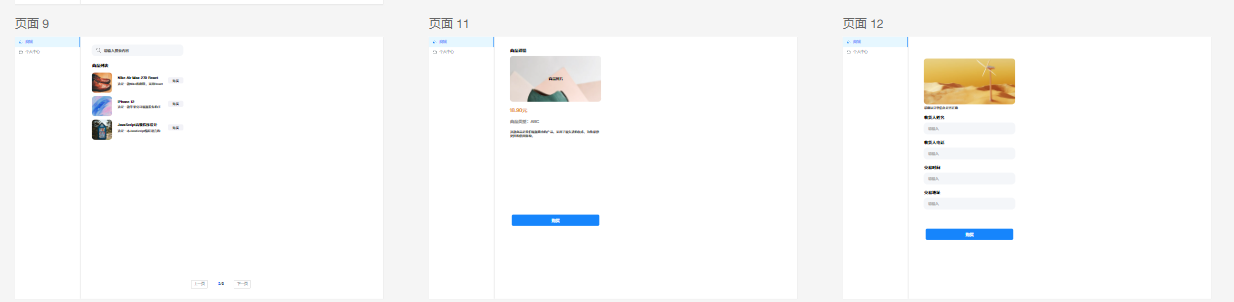
## 界面原型设计



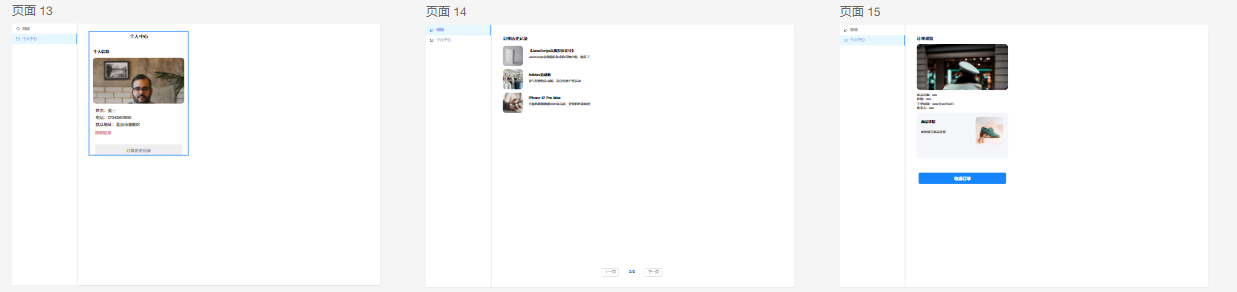
这是登录主界面设计



这是卖家后台界面设计



这是买家求购界面设计



这是个人中心设计

## 系统架构设计

**3.1系统整体架构**

3.1.1运行环境

|  |  |
| --- | --- |
| 环境 | 版本 |
| 操作系统 | windows |
| Vue | 3 |
| Jdk | 8 |
| posterSql | 8.0.29 |

3.1.2后端系统

Springboot 2.7.3

Mysql8.0.29

七牛云存储 分布式文件存储中心

Mybatis 数据库中间件

阿里云druid数据源工具

3.1.3前端系统

Vue3

Element-ui的vue ＵＩ框架

富文本编辑框

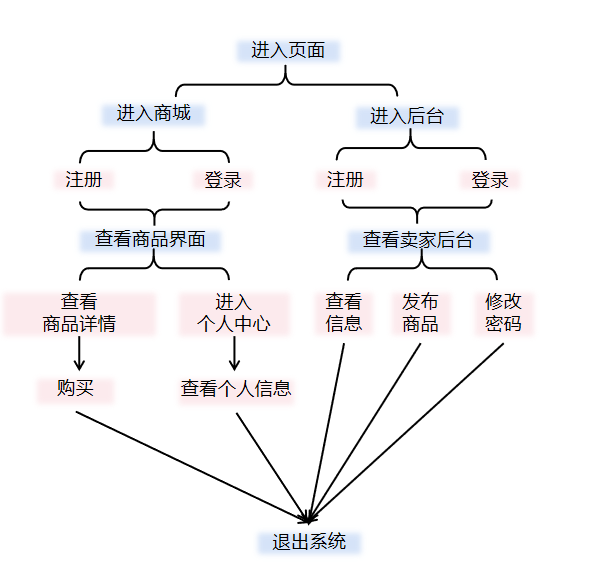


图3.1 项目总体框架实现

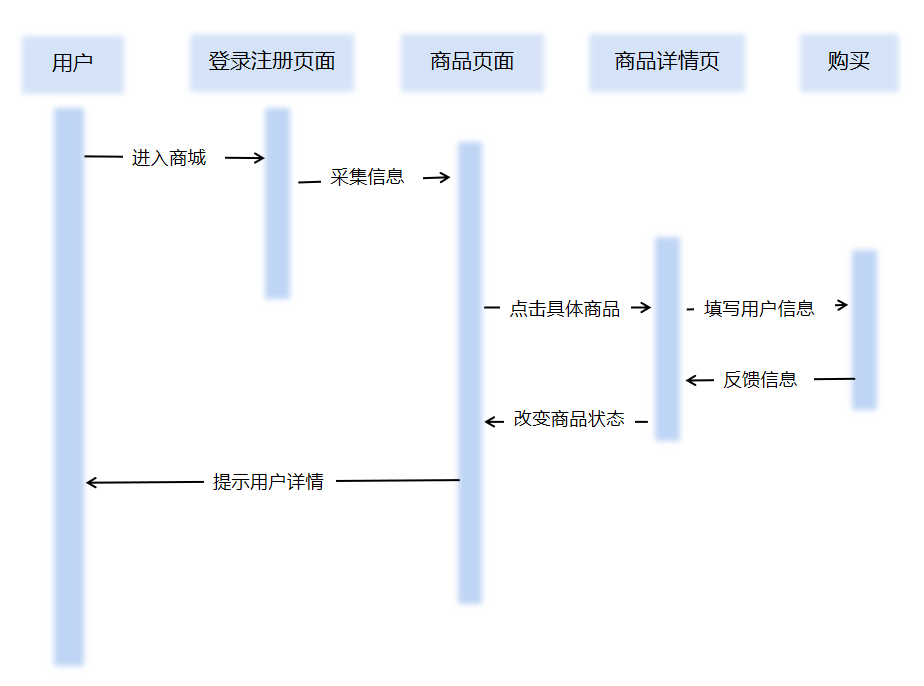


图3.2 项目整体架构

## API设计

|  |  |
| --- | --- |
| **接口描述** | 用户登录 |
| **接口地址** | http://localhost:8082/user/login |
| **请求方法** | post |
| **请求参数** | account  password |
| **相应内容** | 账号  密码 |
| **错误代码** | 0 |
| **实例** | http://localhost:8082/user/login?account=19957462231&password=qianchao90 |

表4.1 用户登录接口

|  |  |
| --- | --- |
| 接口描述 | 查看历史商品 |
| 接口地址 | http://localhost:8081/goods/viewGoods |
| 请求方法 | Post |
| 请求参数 | token |
| 相应内容 | 头部 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/goods/viewGoods？token=1 |

表4.2 历史商品接口

|  |  |
| --- | --- |
| 接口描述 | 修改密码 |
| 接口地址 | http://localhost:8081/seller/Passwordupdate |
| 请求方法 | Post |
| 请求参数 | Password |
| 相应内容 | 密码 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/Passwordupdate？Possword=1 |

表4.3修改密码接口

|  |  |
| --- | --- |
| 接口描述 | 查看购买人 |
| 接口地址 | http://localhost:8081/seller/checkUser |
| 请求方法 | post |
| 请求参数 | id |
| 相应内容 | 序号 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/checkUser？Id=1 |

表4.4查看购买人接口

|  |  |
| --- | --- |
| 接口描述 | 注册 |
| 接口地址 | http://localhost:8081/seller/register |
| 请求方法 | get |
| 请求参数 | Account  password |
| 相应内容 | 账号  密码 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/register？account=123465&possword=44545 |

表4.5注册接口

|  |  |
| --- | --- |
| 接口描述 | 查看意向购买人 |
| 接口地址 | http://localhost:8081/seller/intendingUSer |
| 请求方法 | post |
| 请求参数 | goods\_id |
| 相应内容 | 商品序号 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/intendingUSer?goods\_id=1 |

表4.6查看购买用户接口

|  |  |
| --- | --- |
| 接口描述 | 注册验证 |
| 接口地址 | http://localhost:8081/seller/registerComfirm |
| 请求方法 | get |
| 请求参数 | account |
| 相应内容 | 账号 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/registerComfirm？Account=1 |

表4.7注册验证接口

|  |  |
| --- | --- |
| 接口描述 | 查看订单数量 |
| 接口地址 | http://localhost:8081/seller/getTradecount |
| 请求方法 | get |
| 请求参数 | 无 |
| 相应内容 | 无 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/getTradecount |
| 接口描述 | 查看已注册用户 |
| 接口地址 | http://localhost:8081/seller/checkUser2 |
| 请求方法 | get |
| 请求参数 | 无 |
| 相应内容 | 无 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/checkUser2 |

表4.8订单数量接口

|  |  |
| --- | --- |
| 接口描述 | 查看买家订单 |
| 接口地址 | http://localhost:8081/seller/seeUsertradition |
| 请求方法 | Get |
| 请求参数 | user\_id |
| 相应内容 | 用户序号 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/seeUsertradition？User\_id=1 |

表4.9卖家订单接口

|  |  |
| --- | --- |
| 接口描述 | 根据订单编号查询订单信息 |
| 接口地址 | http://localhost:8081/seller/seeTraditionbyid |
| 请求方法 | get |
| 请求参数 | id |
| 相应内容 | 序号 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/seeTraditionbyid？id=1 |

表4.10查询订单信息接口

|  |  |
| --- | --- |
| 接口描述 | 查看历史商品 |
| 接口地址 | http://localhost:8081/seller/seehistoricalgoods |
| 请求方法 | get |
| 请求参数 | 无 |
| 相应内容 | 无 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/seehistoricalgoods |

表4.11历史商品接口

|  |  |
| --- | --- |
| 接口描述 | 查看未被交易商品 |
| 接口地址 | http://localhost:8081/seller/goodsnotbuy |
| 请求方法 | Post |
| 请求参数 | 无 |
| 相应内容 | 无 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/goodsnotbuy |

表4.12未背交易商品接口

|  |  |
| --- | --- |
| 接口描述 | 登录测试 |
| 接口地址 | http://localhost:8081/seller/Sellerlogin |
| 请求方法 | Post |
| 请求参数 | Account  possword |
| 相应内容 | 账号  密码 |
| 错误实例 | 0 |
| 实例 | http://localhost:8081/seller/Sellerlogin？Account=123&possword=123 |

表4.12登录测试接口

## 数据库设计

|  |  |  |  |
| --- | --- | --- | --- |
| **主键/外键** | **字段名** | **字段类型** | **说明** |
| PK | Id | int | 商品编号 |
|  | name | varchar（50） | 商品名称 |
|  | imgUrl | longtext | 商品图片地址 |
|  | description | longtext | 商品描述 |
|  | price | float（10） | 商品价格 |
|  | type | varchar（20） | 商品类型 |
|  | status | int | 0 代表商品处在正常情况中 1 代表商品正在交易 2 代表商品被商家冻结 3 代表商品已经被下架了 |
|  | supply\_id | int | 提供者id |
|  | supply\_time | datetime | 上架时间 |
|  | stock | int | 库存 |
| FK | img\_id | varchar（255） | 其他图片的id 用空格分隔 |
|  | Seasonal | varchar（255） | 时令 |

表5.1 goods表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | img | longtext |  |

表5.2 img表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | account | varchar(20) | 账号 |
|  | password | varchar(20) | 密码 |
|  | name | varchar(10) | 姓名 |

表5.2 img表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | goods\_id | int | 交易商品Id |
|  | user\_id | int | 购买者id |
|  | buy\_time | datetime | 购买时间 |
|  | trade\_time | varchar（20） | 交易时间 |
|  | result | int | 交易结果 0 代表交易失败 1 代表交易成功 2 代表正在交易 3 有意向 |
|  | trade\_place | varchar（50） | 交易地点 |

表5.3 trade\_action表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | one | int | 一级标签的id |
|  | name | varchar（255） |  |

表5.4：twotype表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | name | varchar(10) | 用户姓名 |
|  | tele | varchar(11) | 电话号码 |
|  | password | varchar(20) | 密码 |
|  | default\_trade\_place | varchar(255) | 默认交易地点 |
|  | account | varchar(20) | 账号 |

表5.5 use表

|  |  |  |  |
| --- | --- | --- | --- |
| 主键/外键 | 字段名 | 字段类型 | 说明 |
| PK | Id | int |  |
|  | name | varchar(32) | 类别名称 |
|  | createTime | timestamp |  |

表5.6 vegetables\_type表

## web端实现

## 前端都放在View文件下

**图形用户界面, 应用程序

描述已自动生成**

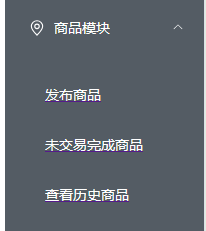
一．第一个文件夹放了一些图片资源

**图形用户界面, 应用程序

描述已自动生成**

二．

**1.第一个文件Aside实现了后台下拉菜单的功能**

****

**<router-link to="/background/publishGoods">  
<el-menu-item index="1-1">发布商品</el-menu-item>  
</router-link>  
<router-link to="/background/goods">  
<el-menu-item index="1-2">未交易完成商品</el-menu-item>  
</router-link>  
<router-link to="/background/goodsRecords">  
<el-menu-item index="1-3">查看历史商品</el-menu-item>  
</router-link>**

**2.第二个文件Header实现了上端页面的布局**

****

**<template>  
 <el-table :data="tableData" style="width: 100%">  
 <el-table-column prop="date" label="Date" />  
 <el-table-column prop="name" label="Name" />  
 <el-table-column prop="address" label="Address" />  
 </el-table>  
</template>**

**3.第三个文件login实现了买家注册页面**

**<template>  
 <div id="divm">  
 <div class="divo">  
 <h1 id="title">欢迎来到我们真菜</h1>  
 <!-- <img src="../../assets/img6.jpg" id="pic"> -->  
 <el-form ref="editFormData" :label-position="labelPosition" label-width="80px" :model="formData" id="for" :rules="rules" status-icon>  
 <el-form-item label="账号" class="formData" prop="account">  
 <el-input v-model="formData.account"></el-input>  
 </el-form-item>  
 <el-form-item label="密码" class="formData" prop="password">  
 <el-input type="password" v-model="formData.password" show-password></el-input>  
 </el-form-item>  
 <el-button type="primary" id="submit" @click="login">登录</el-button>  
 </el-form>  
 </div>  
 </div>  
</template>**

**三．买家后台**

**图形用户界面, 文本, 网站

描述已自动生成**

主要实现买家购买的功能，需要买家输入信息获取订单

**文本

描述已自动生成**

返回买家订单，买家可以根据单号查询

****

四．功能性页面

**图形用户界面, 文本, 应用程序

描述已自动生成**

Changpassword.vue实现了修改密码页面

Goods.vue实现了商品显示页面

Goodrecords实现了展示购买记录的功能

PublishGoods实现了发布商品的功能

Regist实现了登录界面功能

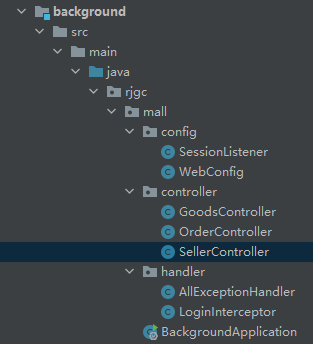
showBuyer类似一个历史记录可以展示卖出去的商品给了谁

五．Router实现数据库的保护

实现登录拦截，保护数据库的安全

**const router = createRouter({  
 history: createWebHistory(process.env.BASE\_URL),  
 routes  
})  
//全局路由设置，实现登录拦截  
router.beforeEach((to,from,next)=>{  
 if(to.meta.requiredToken){  
 if(localStorage.getItem('token') != null){  
 next()  
 }else {  
 next({path:'/background/login'})  
 }  
 }else {  
 next()  
 }  
})**

## 后端实现



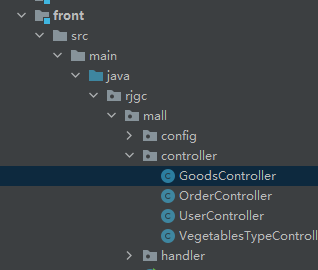
Background文件夹是卖家后端。src/main/java/rigc/mall/controller文件夹中的sellerControler实现了商家登录修改密码选择购买人等功能。

**@RequestMapping("/Sellerlogin")  
public Result Sellerlogin(  
 String account,  
 String password){  
 Seller seller = sellerService.Sellerlogin(account);  
 if(seller==null){  
 return new Result(false,MessageConstant.SELLER\_ACCOUNT\_NON\_EXISTENT);  
 }  
 else if(!seller.getPassword().equals(password)){  
 return new Result(false,MessageConstant.SELLER\_PASSWORD\_WRONG);  
 }  
 else{  
 HttpSession session = request.getSession();  
 session.setAttribute("seller",seller);  
 MySessionContext myContext = MySessionContext.getInstance();  
 myContext.AddSession(session);  
 return new Result(true,MessageConstant.SELLER\_LOGIN\_SUCCESS,session.getId());  
 }  
}**

GoodsController实现了商家管理控制后台。

**@RequestMapping("/putawayGoods")  
public Result putawayGoods(Integer goodsId) {  
 goodsService.putawayGoods(goodsId);  
 return new Result(true, MessageConstant.GOODS\_PUTAWAY\_SUCCESS);  
}  
@RequestMapping("/viewGoods")  
public Result viewGoodsList (@RequestHeader("token") String token,  
 @RequestParam(defaultValue = "1") Integer pageNum,  
 @RequestParam(defaultValue = "10") Integer pageSize){  
 MySessionContext myContext = MySessionContext.getInstance();  
 HttpSession session = myContext.getSession(token);  
 Seller seller=(Seller) session.getAttribute("seller");  
 Integer sellerId = seller.getId();  
 PageInfo<GoodsAndUserAndAction> goodsPageInfo= goodsService.viewGoodsList(pageNum,pageSize,sellerId);  
 return new Result(true,MessageConstant.GOODS\_VIEW\_SUCCESS,goodsPageInfo);  
}**

OrderController实现了查看订单信息，修改订单状态的功能。



Front文件夹是买家后端/src/main/java/rigc/mall/controller中GoodsController实现购物车的功能。

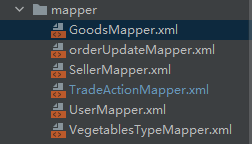
UserController实现了购买的功能。

**@RequestMapping("/buy")  
public Result buy(@RequestBody BuyParam buyParam) {  
 Map<String,Integer> map = userService.buy2(buyParam.getUser(), buyParam.getTradeTime(), buyParam.getTradePlace(),buyParam.getGoodsId());  
 return new Result(true, MessageConstant.GOODS\_BUY\_SUCCESS,map.get("orderId"));  
}**

VegetablesTypeController实现获取蔬菜类别的功能。

**@RequestMapping("/vegetablesType")  
public class VegetablesTypeController {  
 @Autowired  
 private VegetablesTypeService vegetablesTypeService;  
 @RequestMapping("/getAllType")  
 public Result getAllType(){  
 return new Result(true, MessageConstant.GET\_TYPE\_SUCCESS,vegetablesTypeService.getAllType());  
 }  
}**

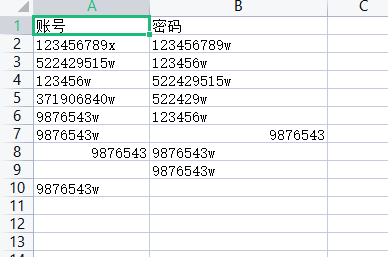
Ｉｎｔｅｒｆａｃｅ／中ｍａｐｐｅｒ中实现了数据库中的操作语言

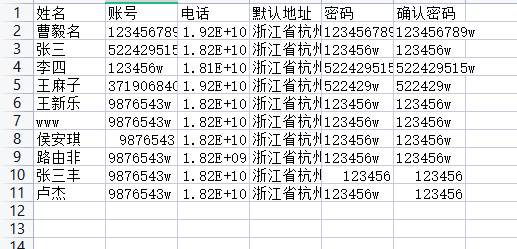


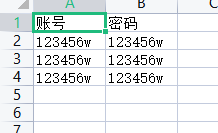
## 系统测试

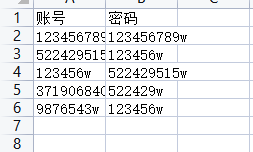
本次测试是为了保证“小趴菜商城”的质量和可靠性，通过运行程序验证其是否符合软件需求，针对测试过程中出现的错误进行修复改良。本次测试的主要范围包括基线需求、升级需求包A、B、C。本次测试主要目标是确保软件系统的功能能够满足需求，测试其是否可以正常运行以及每个功能是否存在缺陷。

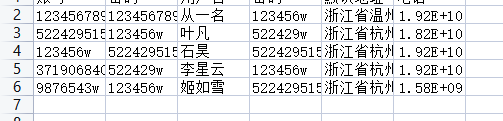
我们组在开发之前讨论了软件的需求，然后我针对所讨论的需求设计出测试用例，当开发人员开发出该功能后进行手动执行测试用例，然后根据测试过程中出现的问题进行跟踪并反馈给开发人员。本学期测试以共进行了285个测试用例，测试功能共32个。在手动执行测试用例后，我采用selenium编写了自动化测试脚本进行自动执行测试用例，目前共完成自动化测试11个功能。通过本次软件测试，“小趴菜商城”基本符合要求，在以后的软件开发和测试工作中我们将继续努力，争取提高软件系统的质量和可靠性，确保软件系统符合实际需求。







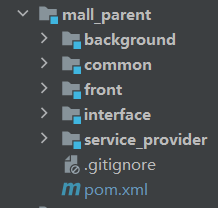




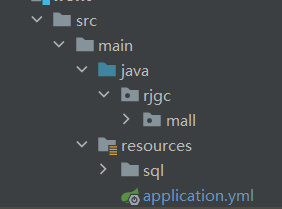
## 系统部署

**（一）项目结构以及pom.xml文件**

我们项目主要使用springboot+vue的框架来搭建我们的项目，是前后端分离项目。首先是后端项目结构，springboot本身是带有maven的：



这是其中一个目录的结构：



mall\_parent为父项目，它的打包方式为pom形式，



其下的各个目录则是jar包形式的。



**（二）后端项目构建**

一切操作都是在父工程中进行。以idea为例

进入父项目中的lifecycle中，点击clean,清理之前编译好的残存文件



然后点击编译，进行后端项目的编译



最后点击打包，对文件进行打包

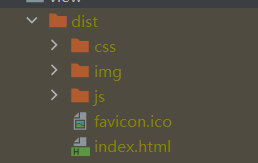


**（三）前端项目构建**

命令行进入前端工程目录： cd view



输入命令 npm run build 对前端项目进行构建打包，出现新的dist目录



前端项目就打包 完毕了

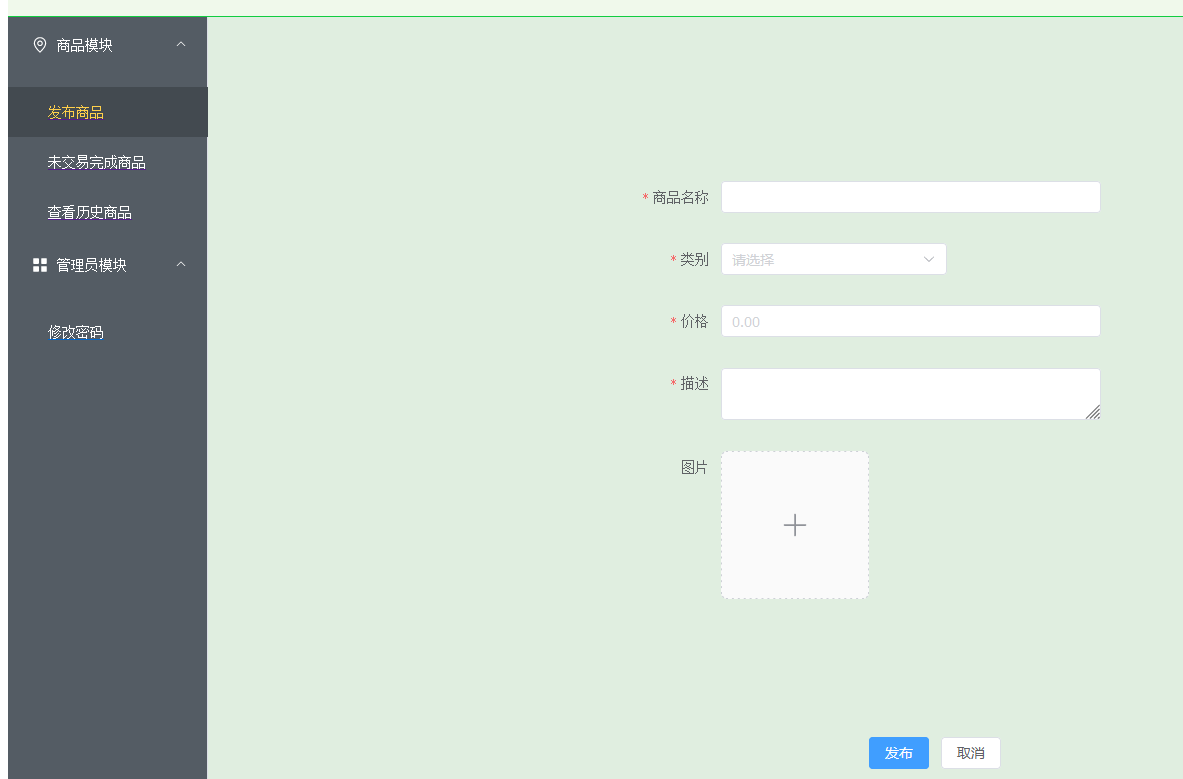
## 功能展示



这是主界面



卖家登录



卖家发布商品信息



卖家修改密码



买家购买界面



订单信息



卖家选择买家



## 清单

项目提交清单

## 实验总结

### 项目背景

当前，互联网+经济发展势头强劲，运用“互联网”模式建立起一套多样化、可持续性、完整的销售模式，是促进乡村经济发展的一种新手段。这为乡村产业升级、精准提供产品服务开辟了新渠道，意义重大。乡村互联网+经济的兴起，促使传播领域逐渐形成了主体多样、交融复杂的媒体传播生态系统，在通过网络将农产品销往各地的同时，产地的产品文化也随之同行、民俗文化也随之发扬光大，而农产品附带的文化要素也让产品实现了增值，两者相辅相成、相得益彰。

### 项目目标

我们的项目目标是开发一个功能完善、界面友好、性能稳定的电商app，能够满足用户的基本需求，如浏览商品、下单、支付等。我们还希望通过我们的app，能够增加用户的购物体验和信任感，从而提高用户的忠诚度和转化率。

### 项目过程

我们的团队共有三名成员，分别负责前端、后端和文档。我们采用了敏捷开发的方法，将项目分为几个迭代周期，每个周期完成一部分功能，并进行测试和评估。我们使用了以下的技术和工具：

- 数据库：我们使用了Mysql作为数据库，存储了商品、订单、用户等信息。

- 后端：我们使用了Java编写后端代码，利用了Springboot框架，实现了商品管理、订单管理、用户管理等模块，并提供了RESTful风格的接口。

- 前端：我们使用了网页作为前端，使用了HTML、CSS和JavaScript等技术，实现了商品展示、购物车、订单确认等页面，并调用了后端接口。

### 项目结果

经过几个月的努力，我们最终实现了预期的效果，完成了一个电商app。我们对我们的app进行了测试和评估，发现它能够正常运行，没有出现明显的错误或缺陷。我们还邀请了一些用户进行试用和反馈，得到了一些积极的评价和建议。

### 项目反思

通过这次实践活动，我们收获了很多。我们不仅学习了计算机软件专业相关的知识和技能，还锻炼了团队合作和沟通能力。同时，我们也发现了一些问题和不足，如：

- 团队沟通问题：我们在项目过程中存在一些沟通问题，比如对需求和任务分配不清晰，导致出现一些重复或遗漏的工作。我们应该在项目开始前就明确需求和分工，并定期进行沟通和汇报。

- 项目目标方向问题：我们在中途改变了项目目标方向，从最初抽卡模拟器变成了电商app。我们应该在项目开始前就确定好项目目标方向，并尽量避免中途改变。

- 项目文档问题：我们在项目过程中没有重视文档的编写，导致文档不完整或不规范。这样不利于项目的管理和维护，也不利于项目的交流和分享。我们应该在项目过程中及时编写文档，并遵循一定的标准和格式。

总之，这次实践活动是一次宝贵的学习和成长的机会，我们感谢老师和同学们的指导和帮助，我们也希望能够在今后的学习和工作中，继续努力，不断进步。

## 参考文献