QQ截图20170609084551

****

**软 件 学 院**

**JAVA程序设计实践课程报告**

**课程项目 外卖点单系统**

**项 目 组 肥宅快乐组**

**姓名学号 麦多健 24320172203184**

**刘子炜 24320172203180**

**姓名学号 洪永团 243201722033137**

**姓名学号 张有坤 24320172203256**

**姓名学号 李吉花 24320172203154**

**实践时间 2019.6.17 — 2019.7.19**

**2019年 7 月 18日**

1. **项目组合影和分工介绍**

（字体为宋体、字号为小四、行间距为1.5倍行距）

分工介绍：

麦多健：前端界面逻辑和后端api设计

刘子炜：前端UI设计

洪永团：service层和controller层编程

张有坤：数据库设计和dao层编程

李吉花：系统测试

**（麦多健完成这部分）**

1. **项目目标与要求**

（字体为宋体、字号为小四、行间距为1.5倍行距）

本次课程设计，需熟练掌握开发语言Java和JDK开发环境idea、webstorm等，根据需求分析合理设计数据库，包括数据、功能和性能需求等，同时结合前后端工作, 按照进度进行设计和开发工作，小组成员之间合理分工、相互积极配合，主动动手进行实践，锻炼个人的编程开发能力和小组合作的能力，协力共同完成网上外卖点单系统，项目的开发，提供给客户方便、快捷的订餐服务。

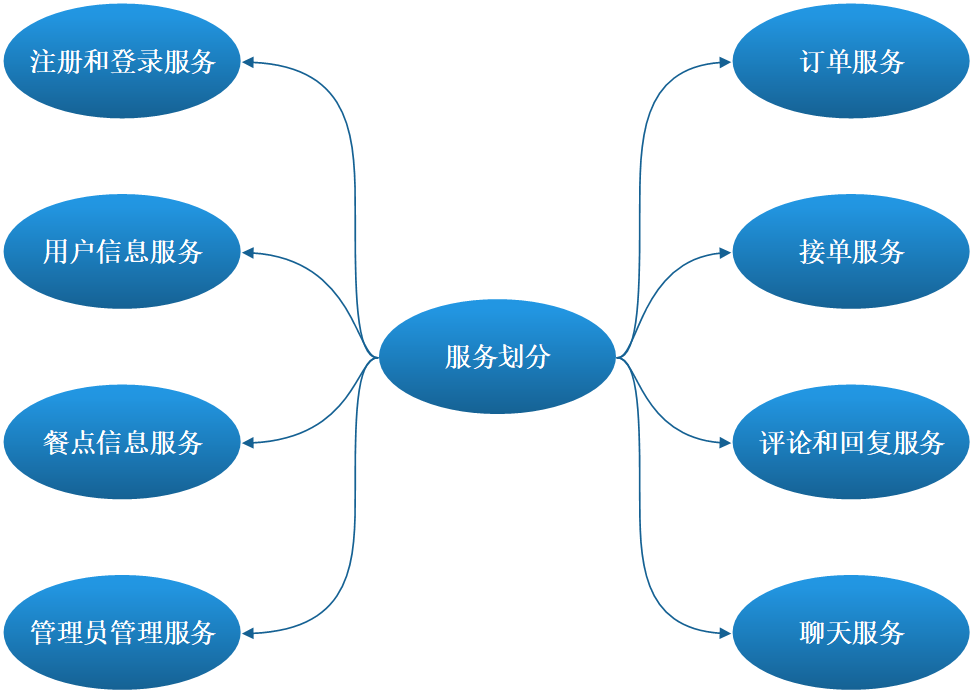
订餐系统需实现基本的功能需求，针对不同的用户，设置不同的权限与功能，完成用户注册登录、用户点餐、个人信息管理、商家菜单管理、购物车管理、订单管理等功能，并且在实现基本功能的基础上添加更全面的功能，完善整个项目系统。

1. **项目内容**

（字体为宋体、字号为小四、行间距为1.5倍行距）

（一）、功能需求

本系统服务功能可划分为以下模块：



各服务的功能点有以下：

注册和登录服务：

1. 商家和用户可以注册和登录（包括短信验证码登录），限制用户在线时长

2. 商家和用户可以通过短信验证修改密码

用户信息服务：

2. 商家和用户可以查看和修改个人信息

3. 商家、用户和管理员可以修改密码

管理员管理服务：

3. 管理员可对用户信息进行查看和管理；管理员对商家进行查看和管理；管理员可查看订单信息；管理员可查看用户操作日志；管理员可以对以上信息进行Excel导出

**餐点信息服务：**

4. 商家录入菜单（餐点名称，餐点类型，库存数量，餐点单价，餐点图片）

5. 菜单的修改、删除

6. 根据类型或关键字筛选相关菜单

12. 餐点欢迎度分析，统计各类餐点的选购频率，按频率排序

订单服务：

7. 用户选择餐点加入购物车

8. 购物车内餐点数量的修改、删除

9. 用户订单提交

11. 历史订单查询，根据用户号或者用户信息查询用户历史订单

接单服务：

16. 商家可以在线收到新订单通知

10. 商家确认点单，库存数量修改

聊天服务：

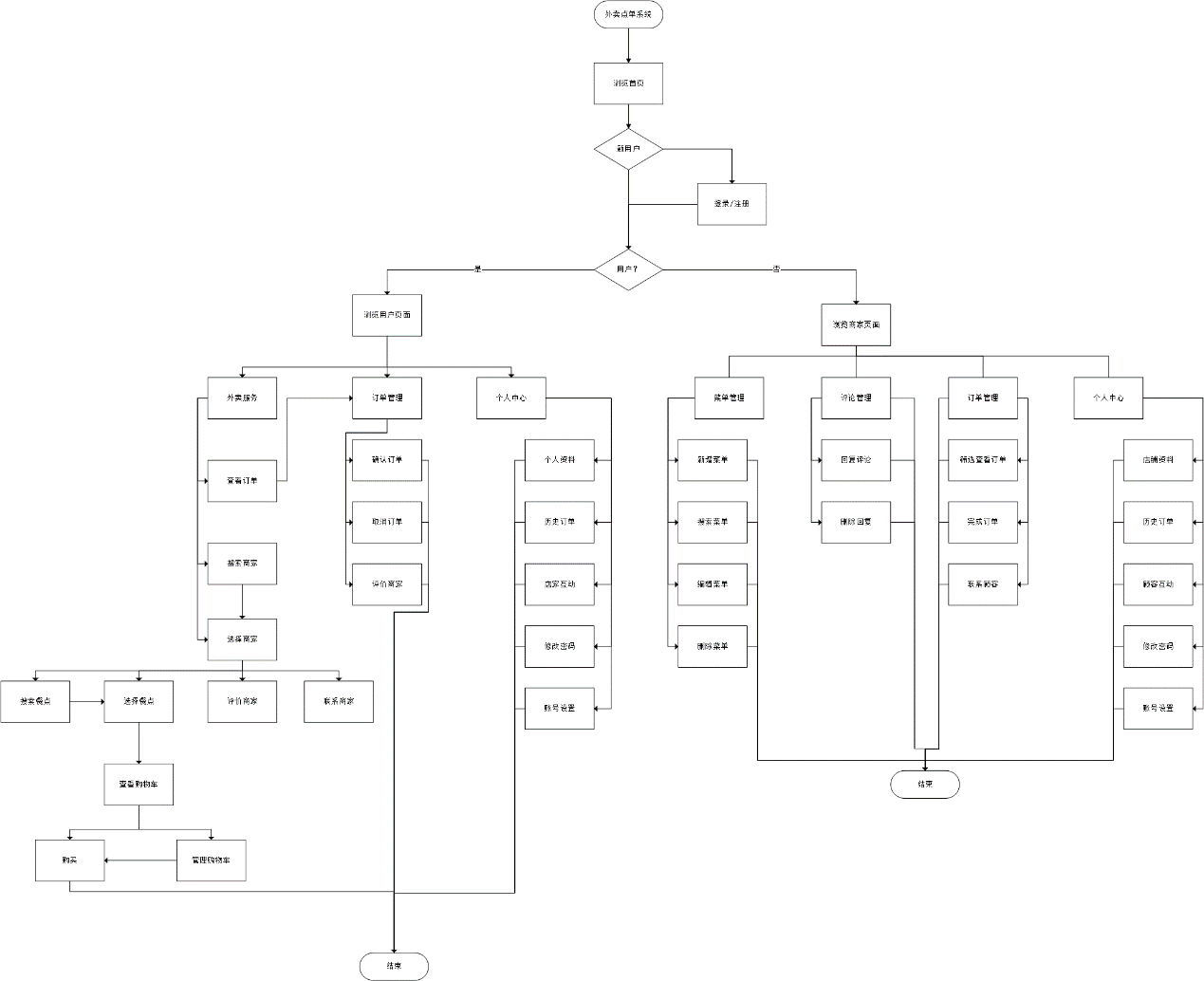
13．商家和用户实时聊天

评论和回复：

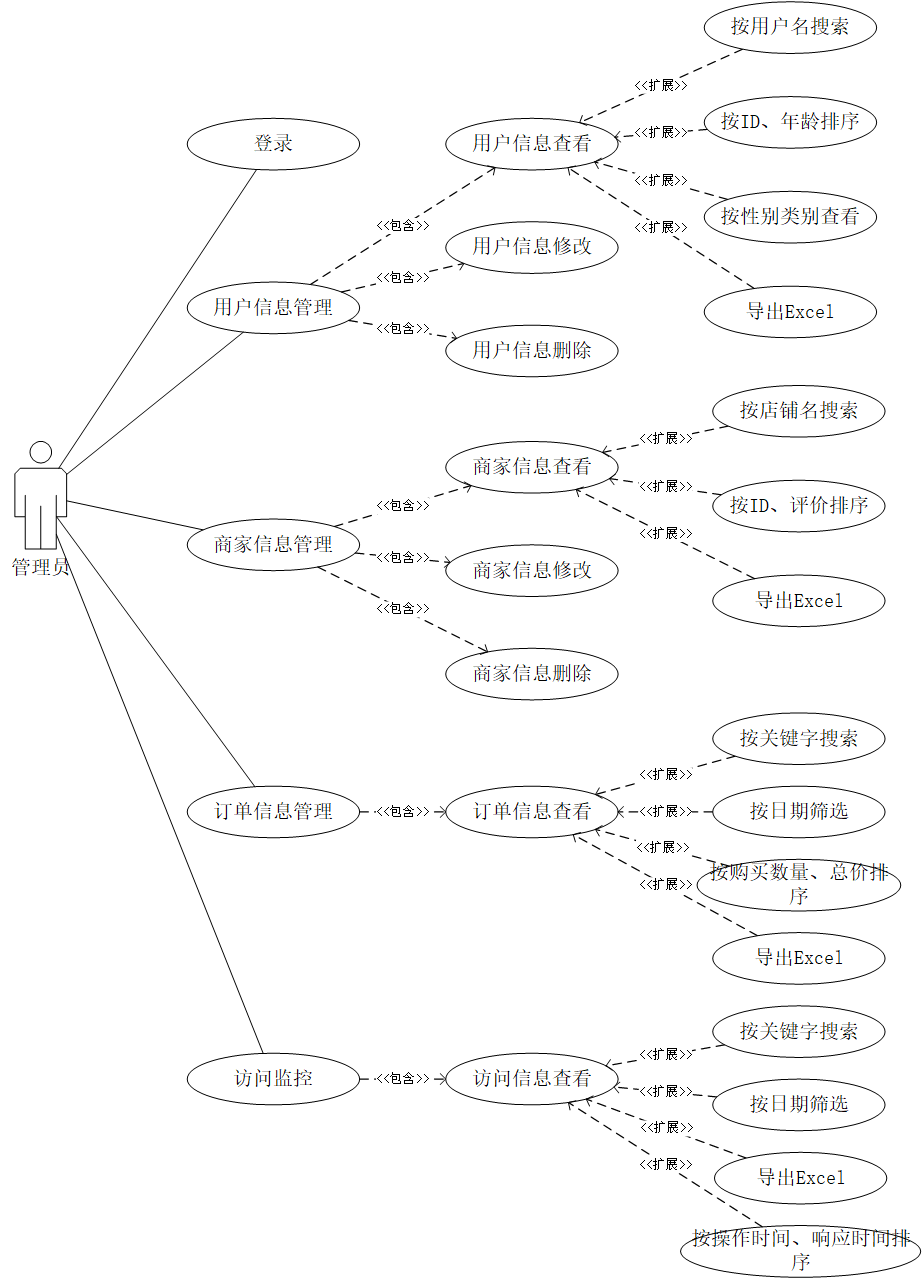
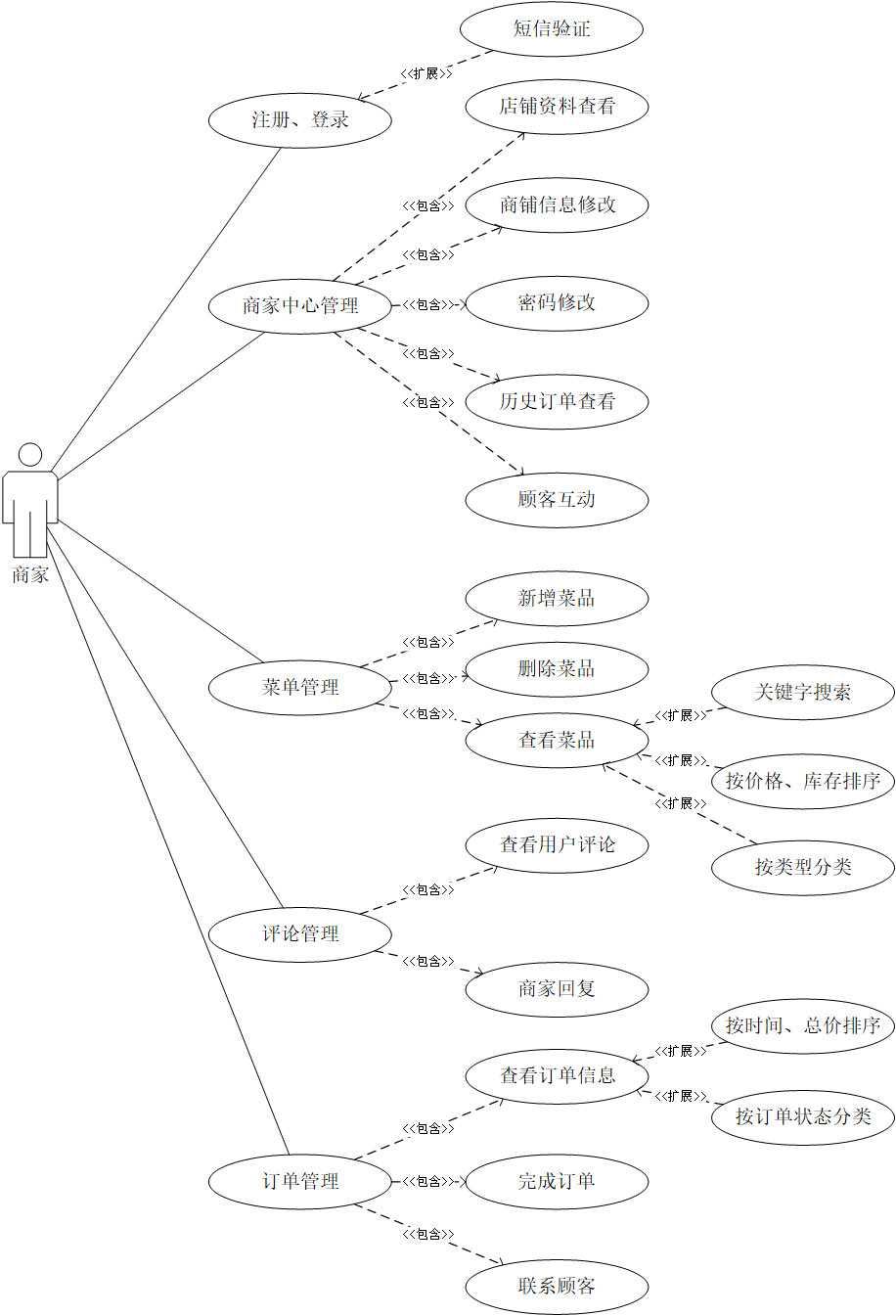
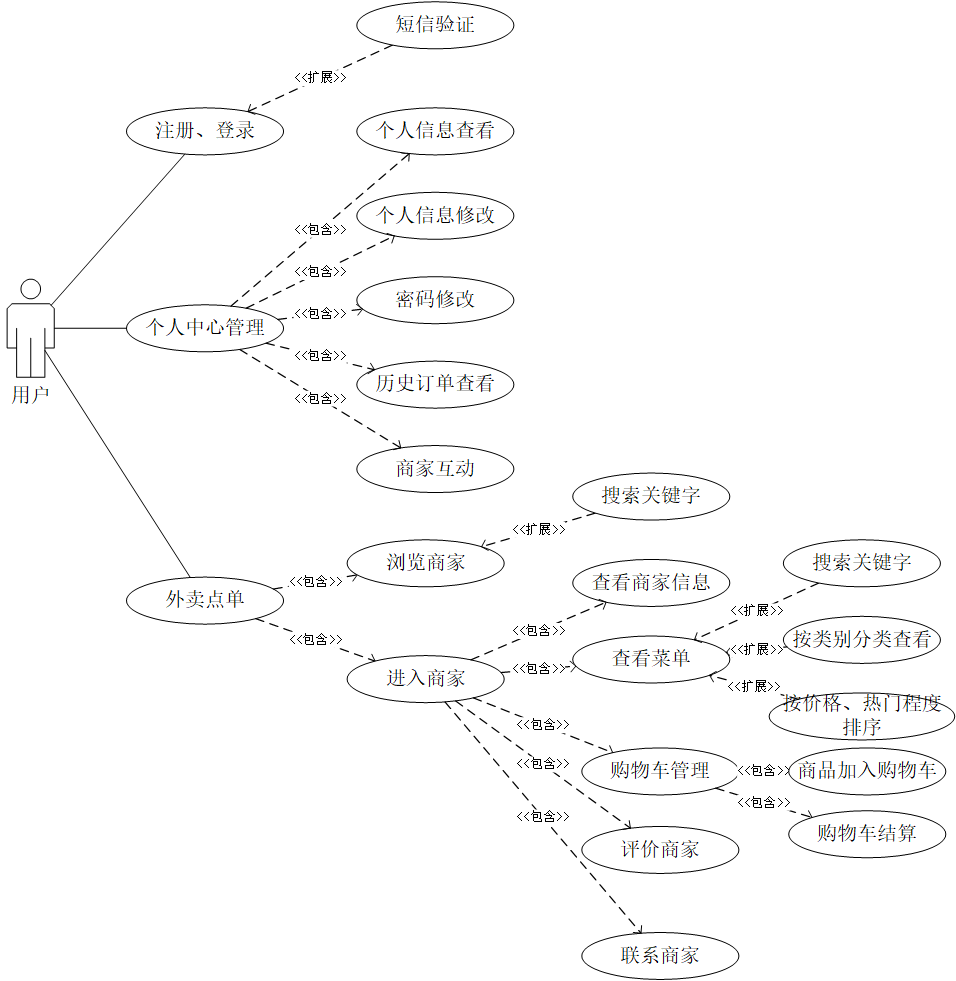
14. 用户可以评价商家

15. 用户和商家都可以回复用户的评论

总体的业务流程描述为：



按用户、商家和管理员可得到三个用例图：



**（麦多健、洪永团）**

（二）、系统设计

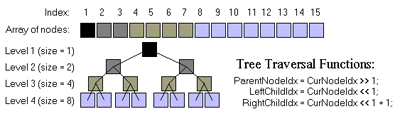


图1 二元树结构

**技术方案：**

**前端：Vue全家桶（ue+vue-router+vuex），ElementUI,axios**

**后端：Spring Boot, Shiro, Mysql**

**还有一些琐碎的技术**

**（麦多健）**

本系统开发过程有数据库设计、前端设计和服务器端设计：

**数据库设计流程：**

1. **数据库设计**

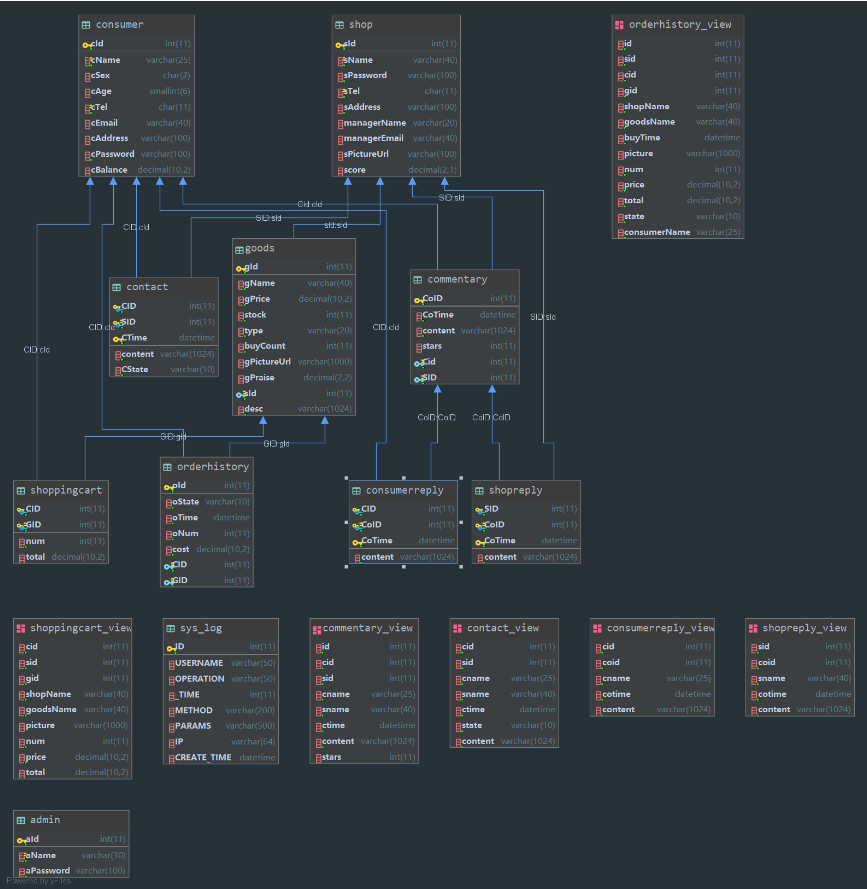


图1 数据库设计图

1. **ER图**

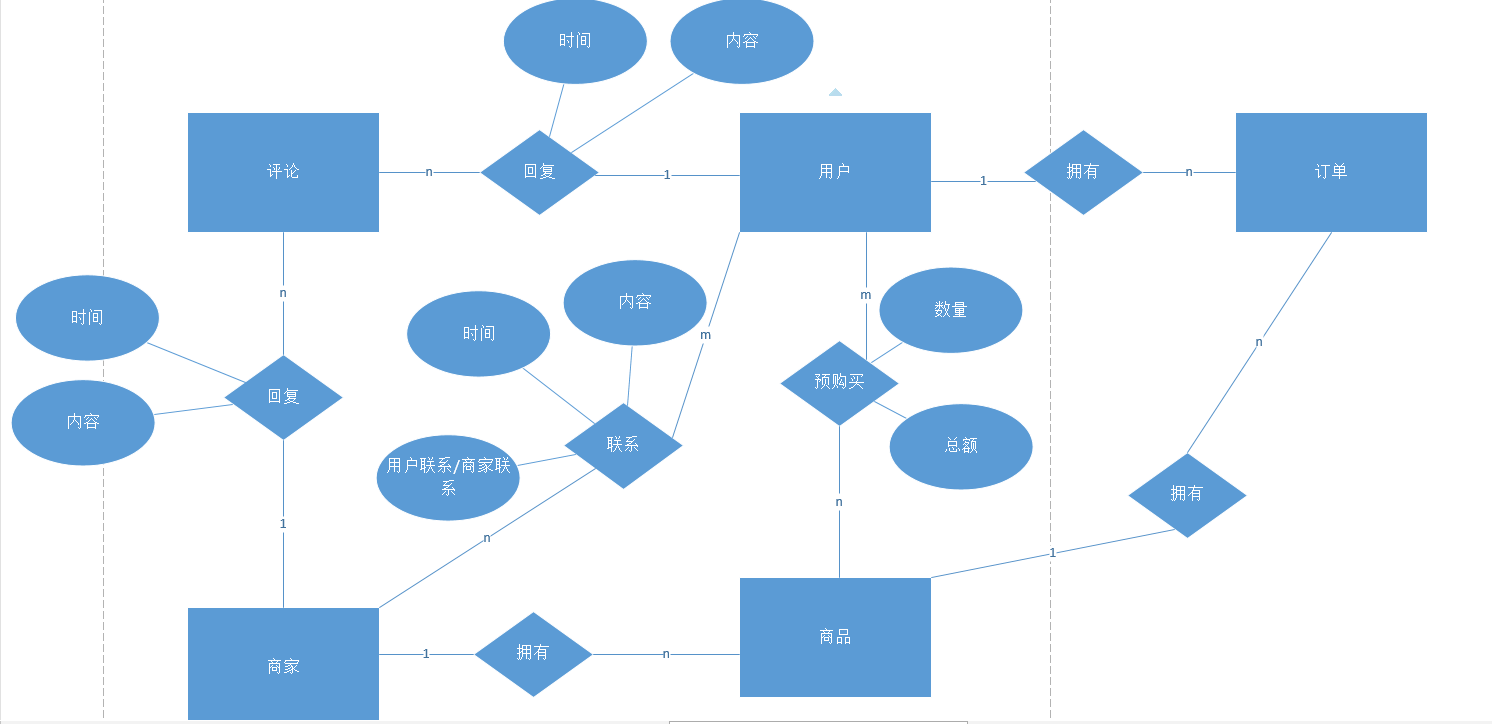


图2 整体ER图

其它具体ER图可以在开发文档文件夹的数据库文件夹查看。

1. **数据流图**

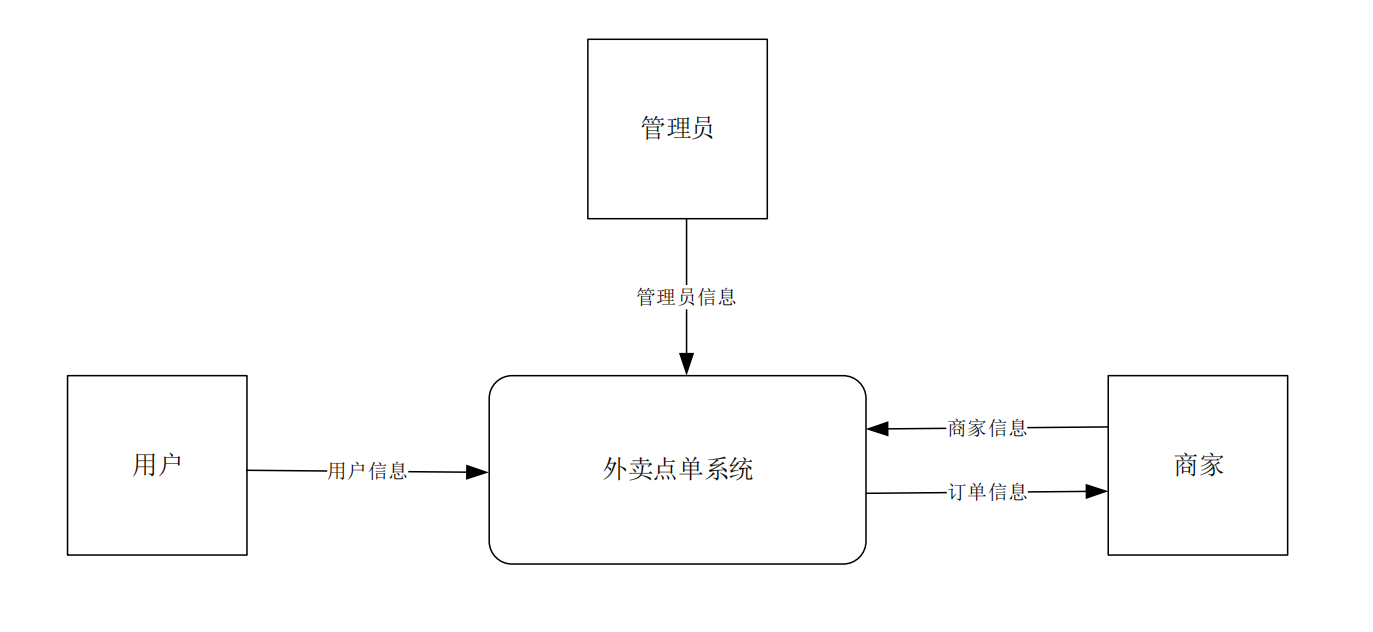


图3 顶层数据流图

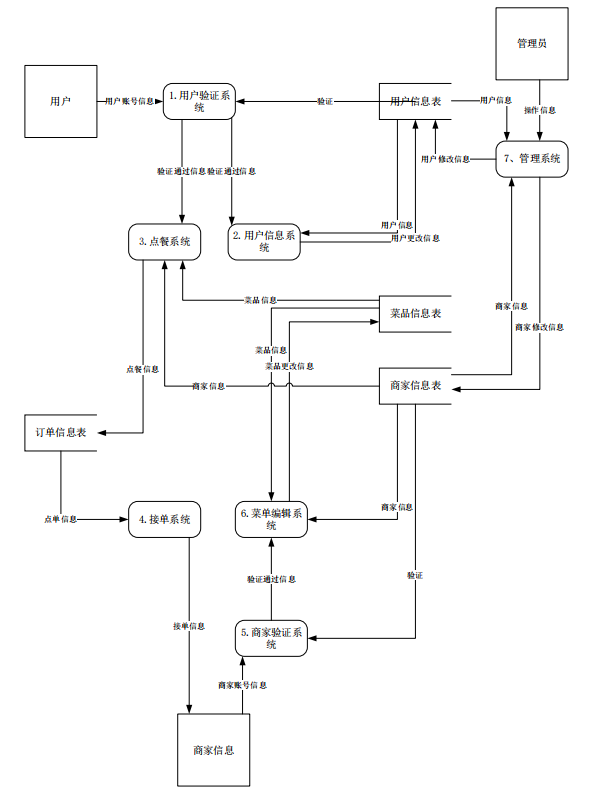


图4 一层数据流图

其它具体数据流图可以在开发文档文件夹的数据库文件夹查看。

1. **关系模式（见下图）**

表1 数据库表Admin的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Aid | 数字 | 整型 | 管理员编号 |
| Aname | 字符串 | 字符型 | 管理员名字 |
| APassword | 字符串 | 字符型 | 管理员密码 |

表2 数据库表Consumer的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Cid | 数字 | 整型 | 用户编号 |
| Cname | 字符串 | 字符型 | 用户名 |
| Csex | 字符串 | 字符型 | 用户性别 |
| Cage | 数字 | 整型 | 用户年龄 |
| Ctel | 字符串 | 字符型 | 用户电话 |
| Cemail | 字符串 | 字符型 | 用户邮箱 |
| Caddress | 字符串 | 字符型 | 用户地址 |
| Cpassword | 字符串 | 字符型 | 用户密码 |
| Cbalance | 数字 | 浮点型 | 用户账户余额 |

表3 数据库表Shop的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Sid | 数字 | 整型 | 商家编号 |
| Sname | 字符串 | 字符型 | 商家名 |
| Spassword | 字符串 | 字符型 | 商家密码 |
| Stel | 字符串 | 字符型 | 商家电话 |
| Saddress | 字符串 | 字符型 | 商家地址 |
| managerName | 字符串 | 字符型 | 店主名 |
| managerEmail | 字符串 | 字符型 | 店主邮箱 |
| SPictureUrl | 字符串 | 字符型 | 商家照片 |
| Score | 数字 | 浮点型 | 用户给商家的评分 |

表4 数据库表Goods的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Gid | 数字 | 整型 | 商品编号 |
| Gname | 字符串 | 字符型 | 商品名 |
| Gprice | 数字 | 浮点型 | 商品单价 |
| Stock | 数字 | 整型 | 商品库存数量 |
| Type | 字符串 | 字符型 | 商品类型 |
| buyCount | 数字 | 整型 | 商品被购买的次数 |
| GPictureUrl | 字符串 | 字符型 | 商品图片 |
| Sid | 数字 | 整型 | 商品所属的商家编号 |
| Desc | 字符串 | 字符型 | 商品描述 |

表5 数据库表OrderHistory的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Oid | 数字 | 整型 | 订单编号 |
| OState | 字符串 | 字符型 | 订单状态 |
| OTime | 日期/时间 |  | 订单时间 |
| Onum | 数字 | 整型 | 购买数量 |
| Cost | 数字 | 浮点型 | 总价 |
| Cid | 数字 | 整型 | 订单属于的用户编号 |
| Gid | 数字 | 整型 | 订单购买的商品编号 |

表6 数据库表ShoppingCart的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Cid | 数字 | 整型 | 购物车所属的用户编号 |
| Gid | 数字 | 整型 | 购物车购买的商品编号 |
| Num | 数字 | 整型 | 购买商品数量 |
| Total | 数字 | 浮点型 | 总价 |

表7 数据库表Commentary的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Coid | 数字 | 整型 | 评论编号 |
| Cotime | 日期/时间 |  | 评论发表的时间 |
| Content | 字符串 | 字符型 | 评论的内容 |
| Stars | 数字 | 整型 | 用户对商家的星级评价 |
| Cid | 数字 | 整型 | 评论发出的用户编号 |
| Sid | 数字 | 整型 | 评论针对的商家编号 |

表8 数据库表ShopReply的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Sid | 数字 | 整型 | 发出回复的商家编号 |
| Coid | 数字 | 整型 | 所回复的评论编号 |
| CoTime | 日期/时间 |  | 商家回复的时间 |
| Content | 字符串 | 字符型 | 商家回复的内容 |

表9 数据库表ConsumerReply的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Cid | 数字 | 整型 | 发出回复的用户编号 |
| Coid | 数字 | 整型 | 所回复的评论编号 |
| CoTime | 日期/时间 |  | 用户回复的时间 |
| Content | 字符串 | 字符型 | 用户回复的内容 |

表10 数据库表Contact的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Cid | 数字 | 整型 | 联系的用户编号 |
| Sid | 数字 | 整型 | 联系的商家编号 |
| CTime | 日期/时间 |  | 联系发出的时间 |
| Content | 字符串 | 字符型 | 联系的内容 |
| CState | 字符串 | 字符型 | 联系发送方 |

表11 数据库表sys\_log的数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 数据大小 | 存储信息 |
| Id | 数字 | 整型 | 日志编号 |
| Username | 字符串 | 字符型 | 访问用户名 |
| Operation | 字符串 | 字符型 | 登录方式 |
| \_Time | 数字 | 整型 | 访问时长 |
| Method | 字符串 | 字符型 | 访问路径 |
| Params | 字符串 | 字符型 | 访问参数 |
| IP | 字符串 | 字符型 | 访问IP地址 |
| Create\_time | 日期/时间 |  | 访问时间 |

**数据库和Model的关系处理**

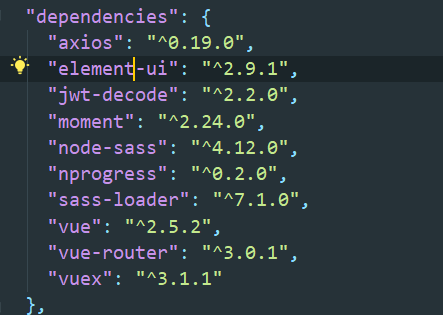
处理方式：将数据库的每一个实体均作为Model中的一个类，一一对应来处理数据库和Model之间的关系。

**（张有坤）**

**前端设计：**

**本系统前端主要负责界面展示（View）**

1. **用到的技术（vue全家桶 axios element-ui …）讲讲分别用来干嘛**

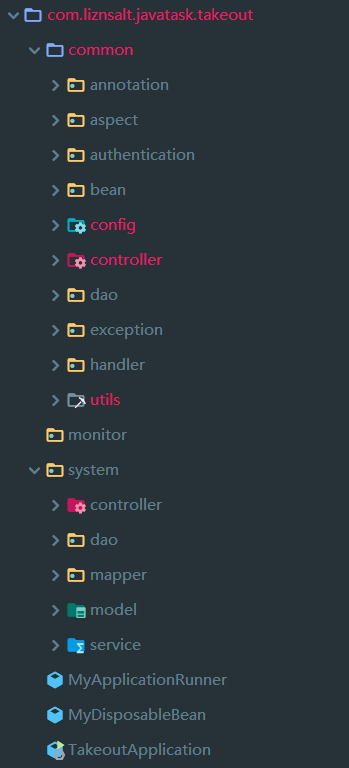


1. **UI设计（可以画个草图）**
2. **vue组件关系（参考附件*前端项目.png*,更新完善它）**
3. **各功能界面实现说明（截图+说明）**
4. **路由关系（画成树形关系）**

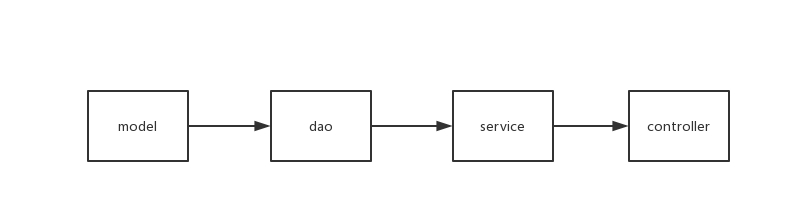
**（刘子炜）**

**服务器端设计：**

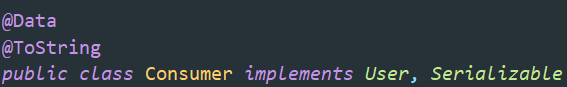
本系统服务器端主要负责Model和Controller，目录结构如下：



每一项需求的设计流程都为：



Model：与数据库中的表一一对应，实现get和set，在本系统中使用Lombok来简化代码，用到的Lombok注解有@Data和@ToString，



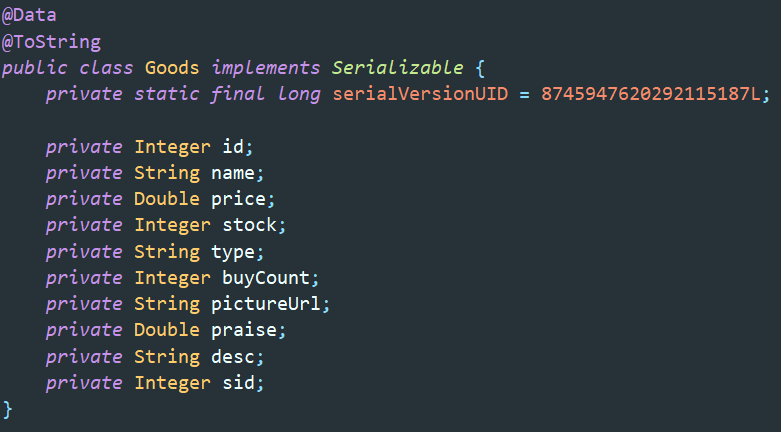
Dao：数据存取对象，主要负责与数据库打交道

Service：业务层，给controller层的类提供接口进行调用

Controller：控制层，负责具体模块的业务流程控制，需要调用service逻辑设计层的接口来控制业务流程, 与前端接口的对接和每个功能接口的实现方法都写在这里，是系统代码的主干部分。

一些简单的需求只是对数据库的增删改查，这里举一个例子来说明一下，一些复杂的业务在后面的技术要点会有介绍。这里介绍一下商家录入菜单的设计过程：

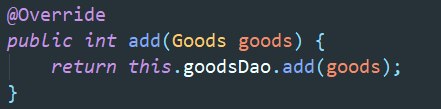
1. 首先需要商品的实体类，创建Goods类



1. 创建商品的dao层，添加add方法



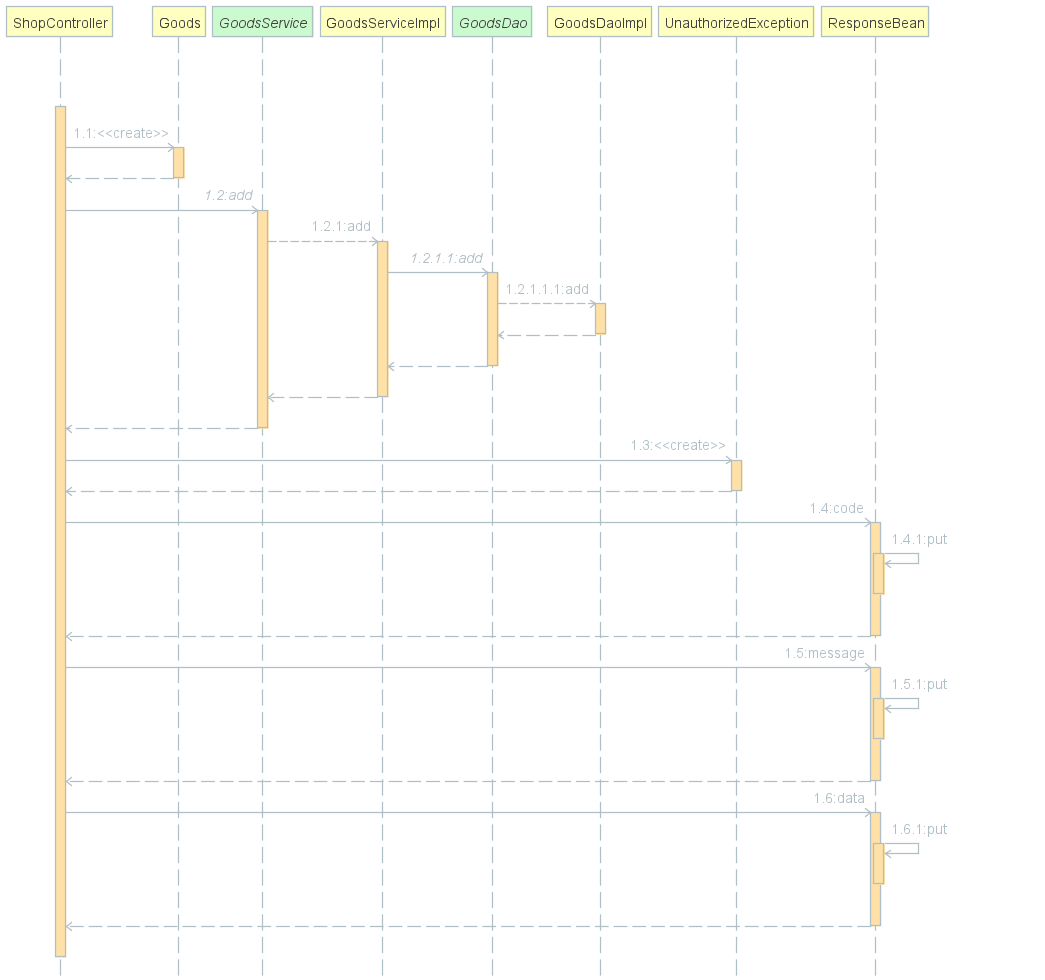
1. 此处业务只是在数据库添加一项，所以service层直接调用dao层的方法即可



1. 接下来书写添加商家的api



这个api的时序图为：



其它api的设计也大致遵循这个流程，如用户查看和修改个人信息、用户购物车添加餐点等业务，其它业务的时序图在时序图文件夹下查看，接下来介绍一些重要的系统设计：

**开发过程中一些重要的技术要点有以下：**

* **服务器环境搭建**

**前端：**

1. **安装配置node.js环境**
2. **使用npm安装vue-cli构建工具，使用“npm install vue-cli -g”命令**
3. **在WenStorm创建新项目，选中Vue.js创建，输入项目路径，选择项目依赖，完成创建**
4. **在项目命令栏中使用“npm install”安装vue模块**
5. **使用“npm run dev”或直接点击WebStorm的项目运行按钮即可运行，运行成功打开能够看到Vue的demo网页**
6. **创建两个项目分别用于外卖系统页面和后台系统页面的开发，分别在配置文件中为它们分配端口号8081和8082**

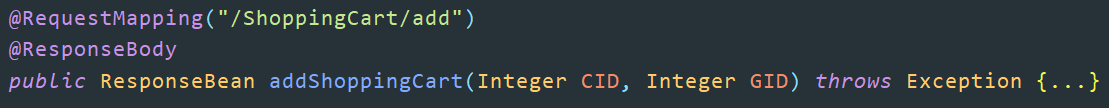
**后端：**

1. **预装java8以上版本、Maven环境**
2. **在IDEA中创建新项目，选中Spring Initializr选项，输入项目命名、勾选所需的依赖、选中路径，完成项目创建**
3. **项目的依赖配置可在pom.xml文件中进行修改，“spring-boot-starter-parent”这一依赖指定了当前项目为Spring Boot项目**
4. **连接mysql数据库：在pom.xml文件中引入mysql驱动依赖，以及spring-boot-starter-jdbc依赖，用于使用JdbcTemplate来操作数据库；创建application.yml文件，配置数据源，输入驱动类型，以及数据库账号密码，即可连接到数据库**
5. **在application.yml文件中为后端配置端口号为8080，访问路径为：/takeout**

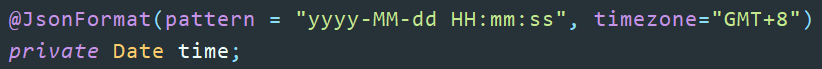
* **JSON技术**

**本项目使用Spring Boot内置的Jackson来完成JSON的序列化操作，主要使用了以下几个注解：**

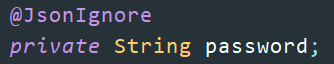
1. @ResponseBody：作用在方法上，能将方法返回的对象序列化为JSON



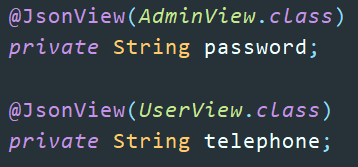
1. @JsonFormat：作用在属性上，可用于日期的格式化



1. @JsonIgnore：作用在属性上，用来忽略所注解的属性



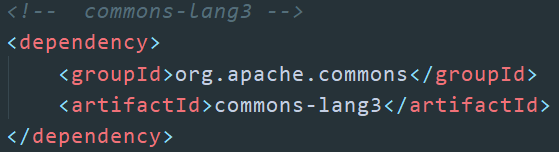
1. @JsonView：作用在类或属性上，用来指定一个序列化组，对于不同组返回显示不同的结果



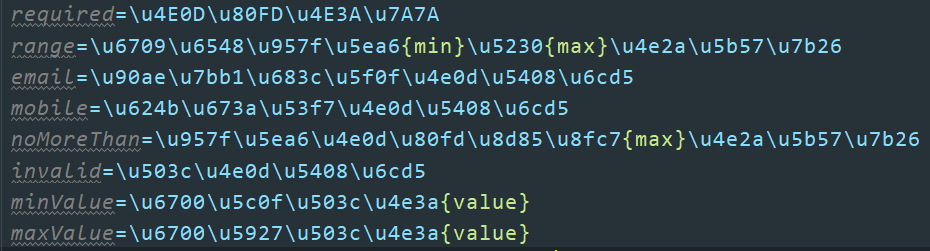
* **Spring Boot配合Hibernate Validator参数校验**

**本项目配合使用了Hibernate Validator进行参数校验，省去了大部分在方法内部的逻辑校验代码。使用步骤：**

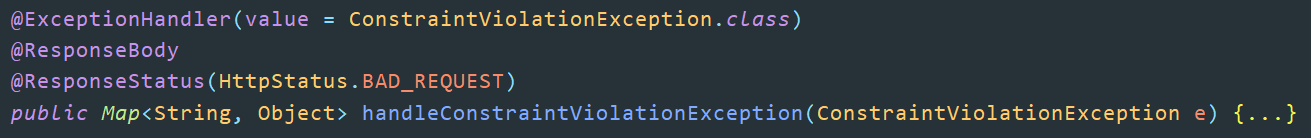
1. 引入依赖：在pom.xml文件中添加依赖



1. 创建占位符提示信息文件：在resources目录下新建ValidationMessages.properties文件，并定义占位符内容，内容为中文转Unicode的值



1. 创建异常处理类，用于处理参数校验不通过时抛出的异常：获取校验不通过的参数名称，然后拼接上提示信息，返回提示信息

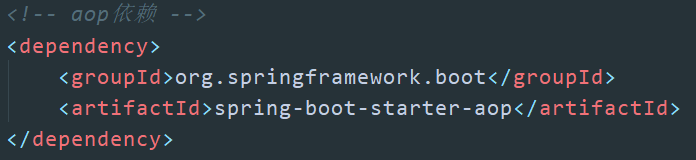


1. 使用注解，在需要参数校验的类上使用@Validated注解，即可在该类中的方法参数上使用校验注解，例如：@NotBlank注解，要求参数不为空，@Min设置参数的最小值，@Email要求匹配邮箱格式，@Length限制字符串长度

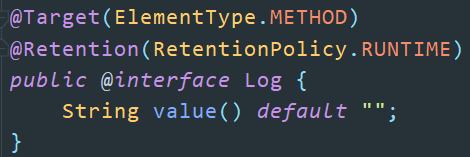


* **AOP编程，记录用户操作**<https://mrbird.cc/Spring-Boot-AOP%20log.html>

1. 引入依赖：在pom.xml文件中添加依赖

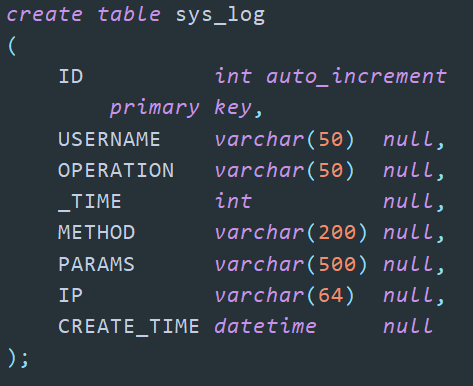


1. 自定义注解@Log，作用在方法上，用于标注需要监控的方法：

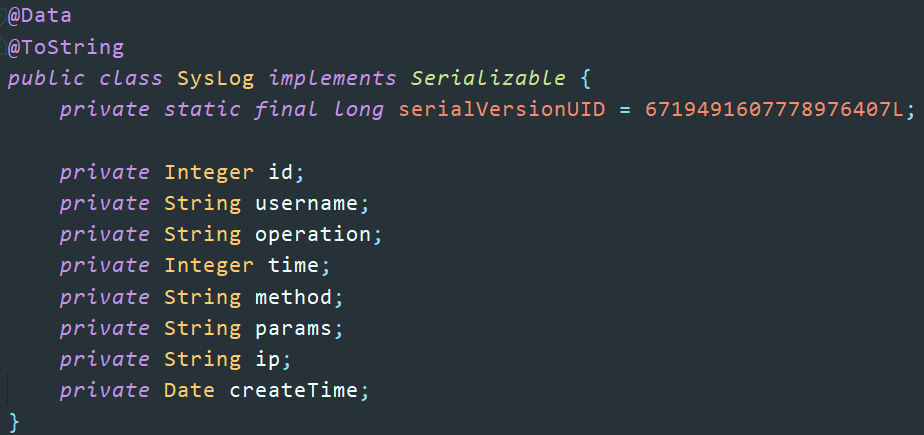


1. 创建数据库表和实体类：

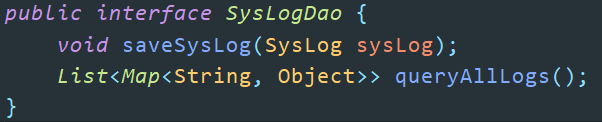
数据库建表



定义实体类



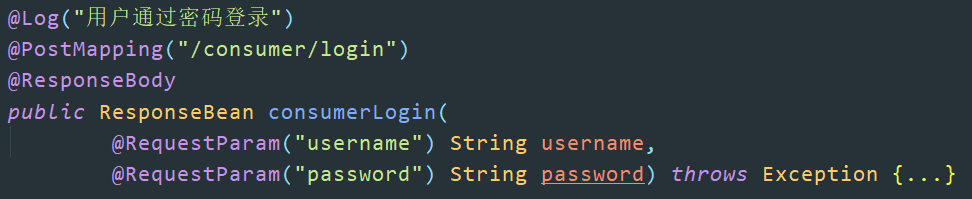
1. 实现保存日志的方法：先定义一个日志接口，在实现类中使用JdbcTemplate来操作数据库



1. 定义LogAspect类，用@Aspect注解标注作为切面



1. @Log标注需要生成日志的API作为切点，每次使用该API都能够产生日志，并储存入数据库

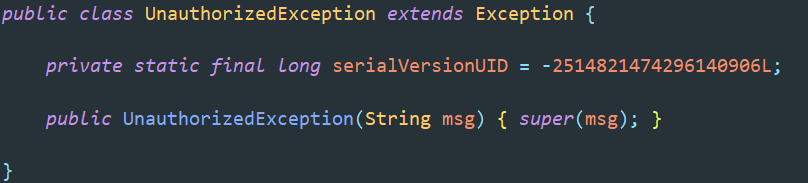


**（洪永团）**

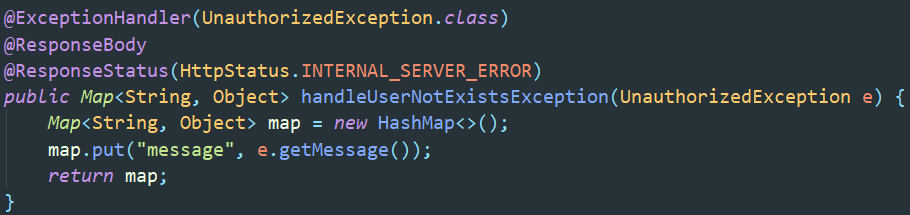
* **服务器端异常处理**

**因为本系统使用前后端分离，所以异常处理需要跨域解决，数据之间的传输使用JSON，利用Spring Boot的ResponseBody注解则可以前端返回一个Object。处理方式如下：**

1. **首先先自定义一个异常类，用来规范整个系统的异常：**



1. **然后定义GlobalExceptionHandler类，全局捕捉UnauthorizedException类型的错误，将错误信息返回给前端：**



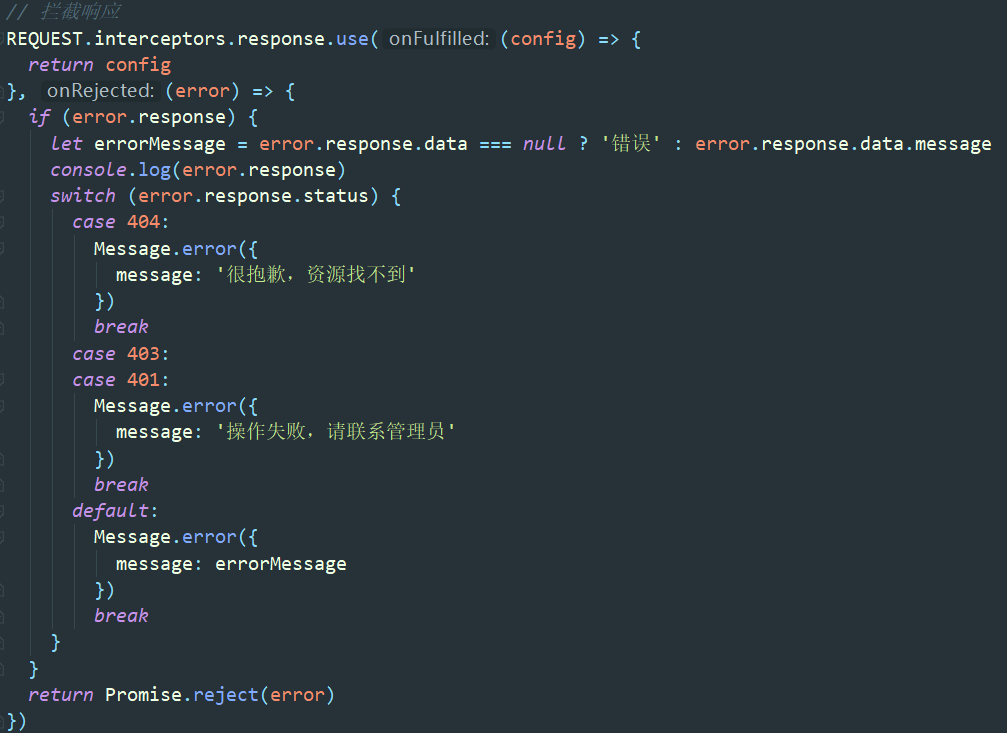
1. **在需要报错的地方都使用此类型的异常错误即可。**

* **axios请求封装,请求异常统一处理**

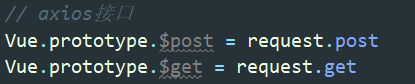
**后端的异常处理已经解决，但是前端需要对axios进行封装，不然在每次请求时都要去判断各种错误，如连接超时、服务器错误等错误。本系统有两种请求，一种不需要在登录情况下发起（syspost、sysget），如注册、登录等，一种需要在登录情况下进行（post、get），如购买，这里主要对需要登录的请求进行封装：**



**然后使用axios的拦截器功能，先对一些常见错误（404等）进行统一处理，其它则是得到后端返回的异常信息，显示给用户：**



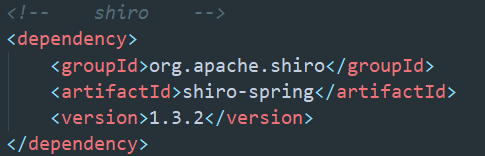
**对post和get请求进行封装后再main.js处把两个方法挂载在Vue的原型上，之后对后端api进行请求使用这两个方法即可。**



* **用户认证和密码加密**

**本项目使用Spring Shiro进行用户认证和MD5算法进行密码加密。**

**首先引入依赖：**



**在Spring Boot中集成Shiro进行用户认证的主要过程为：**

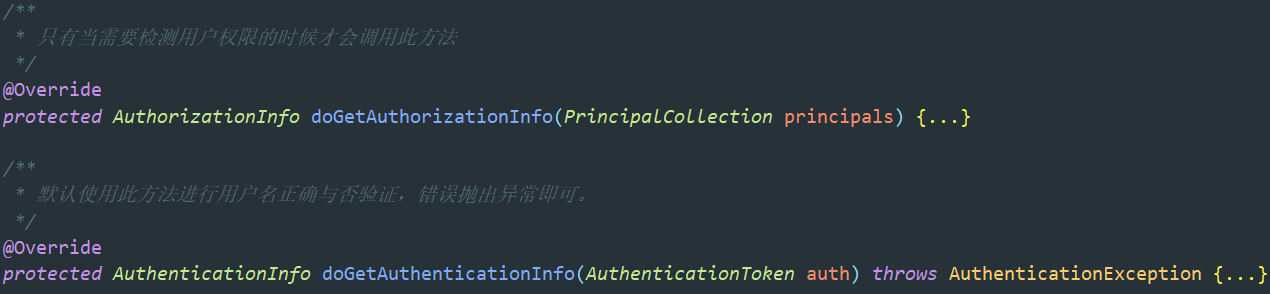
1. **定义一个ShiroConfig，然后配置SecurityManager Bean，SecurityManager为Shiro的安全管理器，管理着所有Subject；**



1. **在ShiroConfig中配置ShiroFilterFactoryBean，其为Shiro过滤器工厂类，依赖于SecurityManager；**

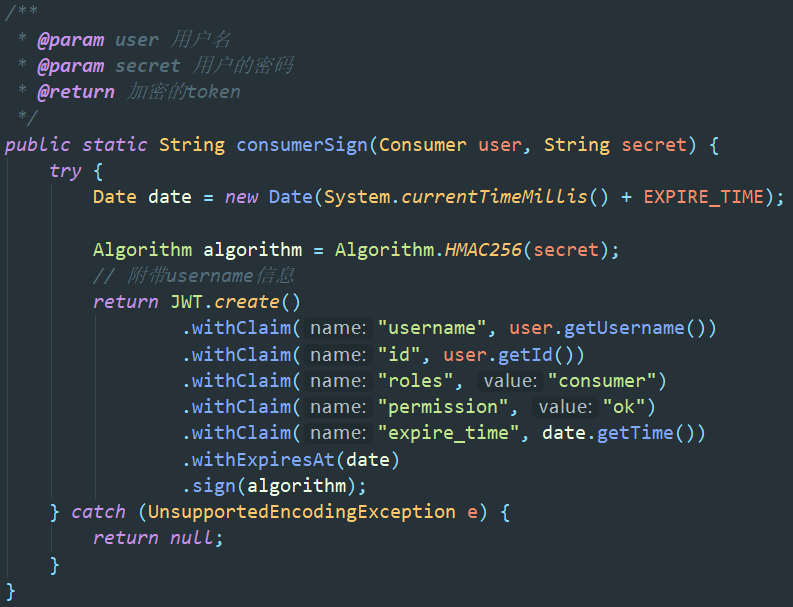


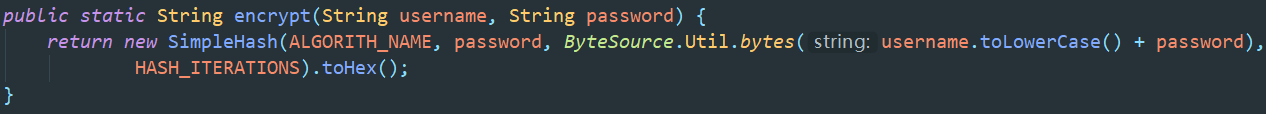
1. **自定义Realm实现，Realm包含doGetAuthorizationInfo()和doGetAuthenticationInfo()方法。因为本系统有普通用户、商家和管理员三种用户，所以定义了三个Realm（AdminShiroRealm、ShopShiroRealm、ConsumerShiroRealm）。**



**完成上面的配置便可以在Spring Boot中使用它的认证、授权、会话和加密功能，但在本项目中只使用了其认证和加密功能。**

**登录认证搭配了JWT使用加密的Json来生成Token并发送给前端，Token里面有用户的各种信息（当然，不包括密码）和过期时间，如下图：**

**密码加密可以直接使用Shiro的MD5算法，**

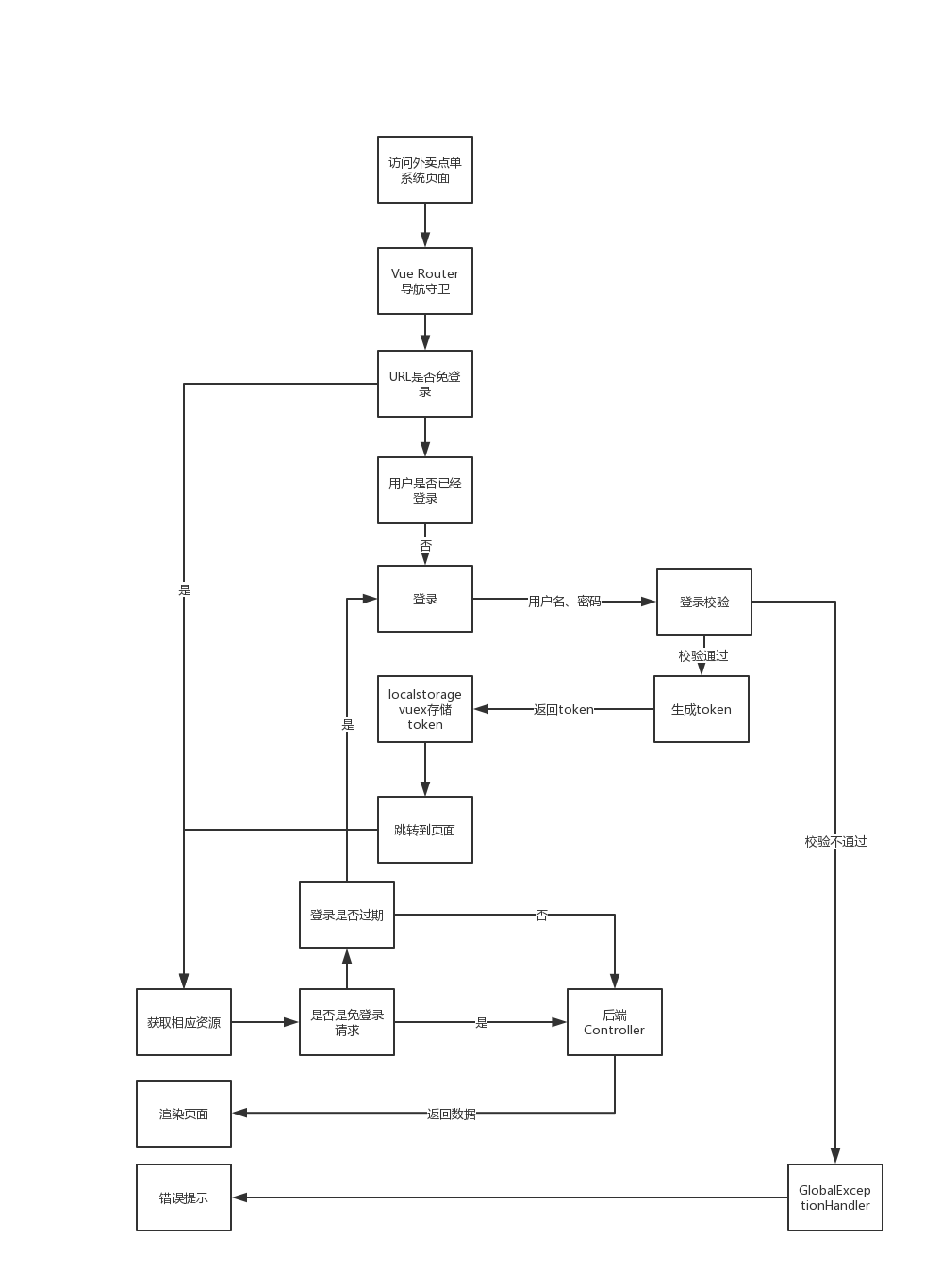


**之后便可以在用户注册和登录的时候对密码进行加密处理。注册成功后数据库长这样：**

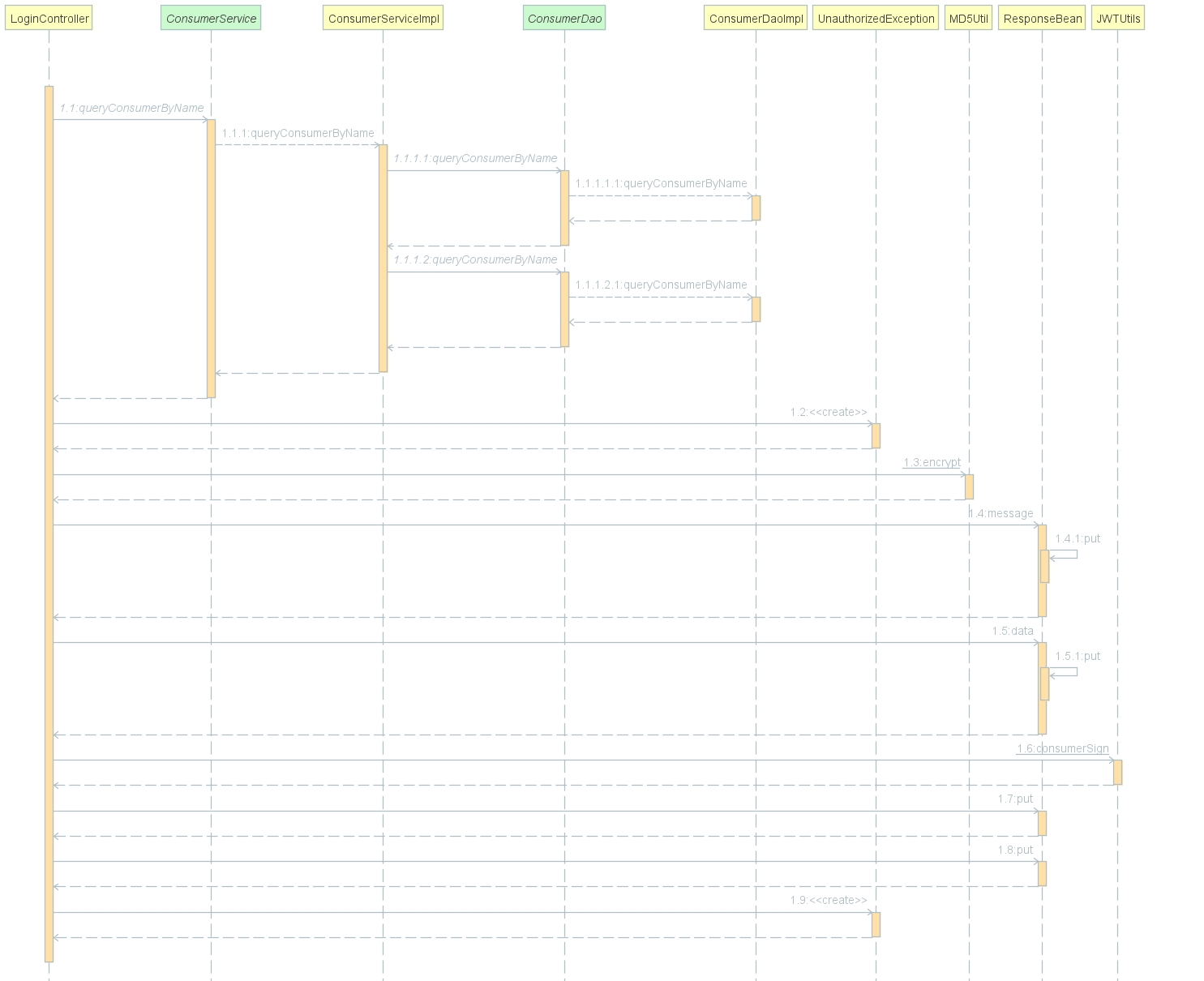


* **请求流程**

**结合上面的异常处理、前端axios请求处理和登录认证，可以得到本系统完整的请求流程，要点是前端localstorage存储后端返回的token，并在前端请求数据时检验是否过期。流程图如下：**

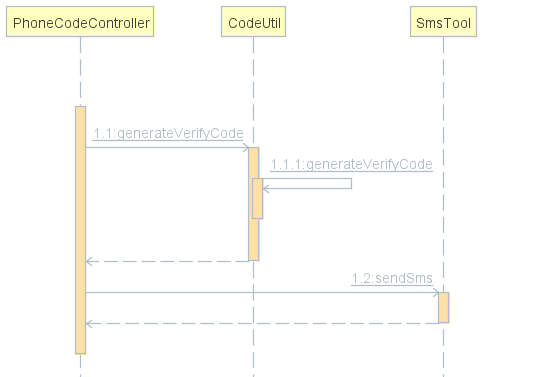
****

**这里展示普通用户登录的时序图，商家和管理员的登录时序图可以在时序图文件夹下查看。**

****

* **短信验证**

**短信验证是保证安全和密码找回常用的手段，本系统使用阿里云的通信短信服务实现短信验证码的发送功能，参考官方的教程，然后自定义随机生成验证码的函数，即可实现发送短信的功能，其时序图如下：**

****

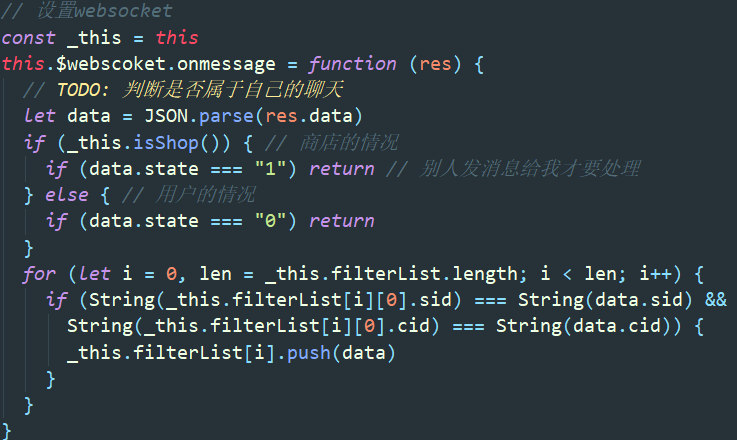
* **在线聊天**

**为了方便用户和商家之间的协商处理，本系统提供两者之间的实时聊天。在线聊天使用WebSocket实现，本系统实现过程为：**

1. **首先后端需要websocket依赖和websocket配置类**
2. **创建websocket处理类controller，规定每接收到一条信息则对所有在线用户进行广播**



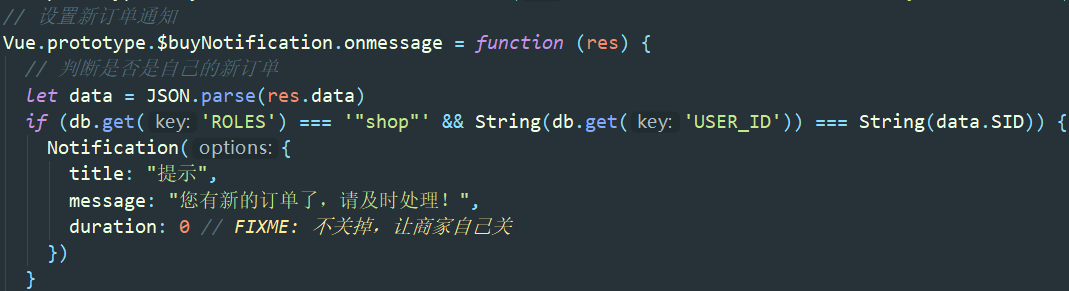
1. **前端这边在main.js处先挂载连接到后端的WebSocket类，然后在聊天的组件处设置websocket收到消息后的执行函数，这里主要是判断聊天是否属于自己，如果是别人发送给自己的则更新页面数据：**



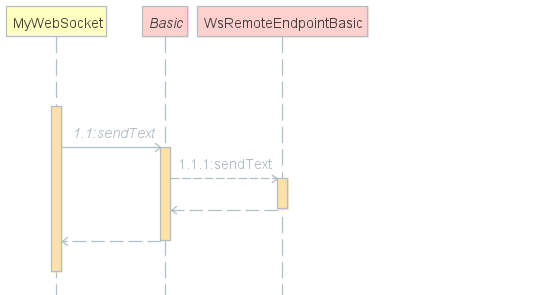
1. **用户发送消息则直接调用WebSocket的send函数，将对话的信息（包括发送者、接收者、发送时间和发送内容）变成JSON字符串后发送给后端，让后端进行广播通知。**

* **新订单通知实现**

**为了及时通知商家有用户购买商品，本系统实现了新订单的在线通知，原理和在线聊天一样，都是利用websocket的群发消息。后端配置基本一样，前端更加简单，直接在main.js处设置通知方式即可：**



**Websocket的时序图如下：**

****

**（麦多健）**

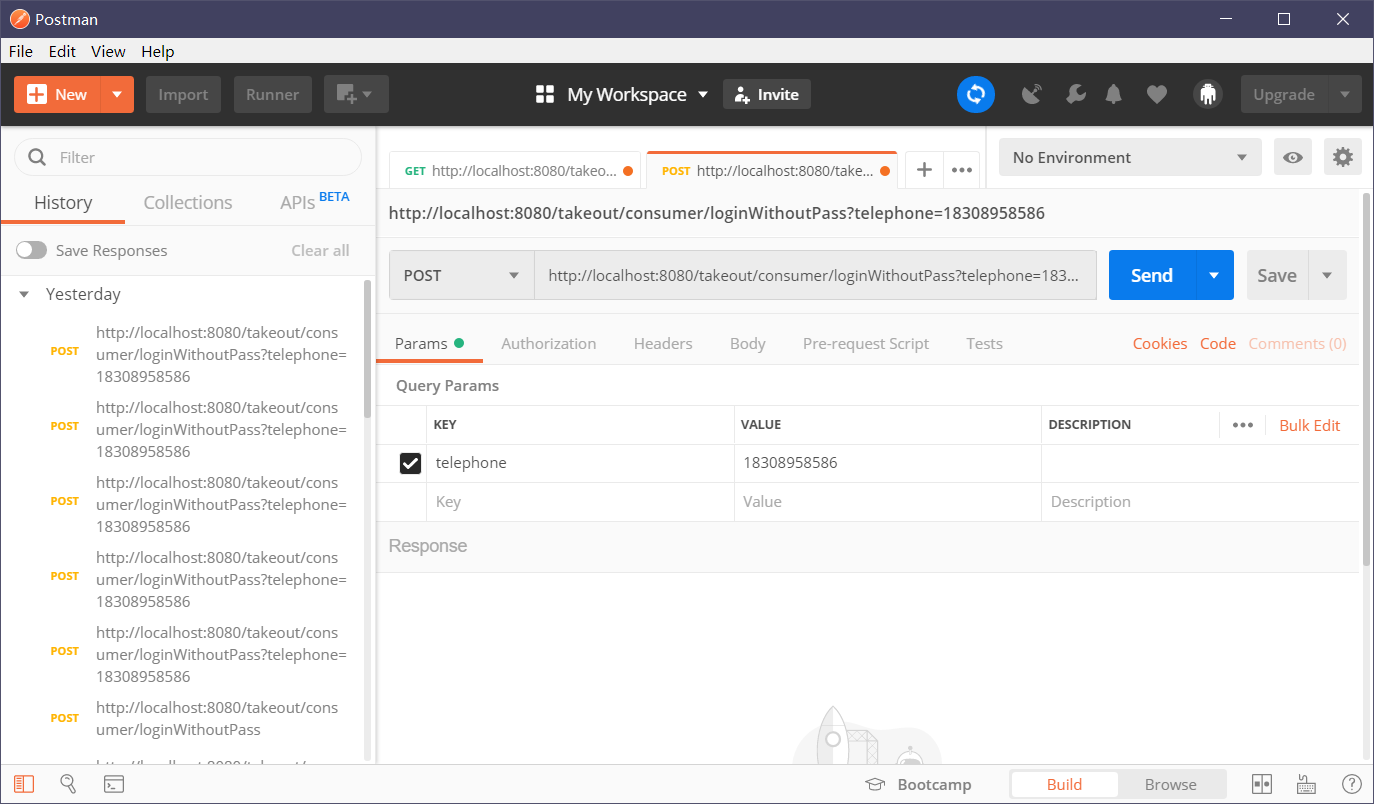
（三）、系统测试

**1. 后端测试**

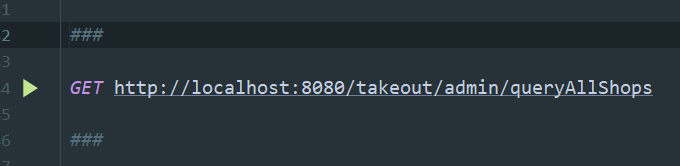
**测试环境：win10，tomcat和nodejs的虚拟服务器**

**1 测试工具Junit4，参考文章<https://mrbird.cc/Spring-Boot%20TESTing.html>，拿几个controller的测试来说明一下，可以自己写几个极端情况来测试**

**2 Postman测试后端API（可以自己多测试几个api，截图说明一下）**

****

**3 IDEA内置的Request Edit**

****

**2. 前端测试**

**测试环境：Chrome，Microsoft Edge，360浏览器**

**1 界面测试**

**2 功能测试（把各个需求功能的相应界面截图，包括实时聊天、新订单通知等）**

**（截图）**

**（李吉花）**

1. **课程总结**

（字体为宋体、字号为小四、行间距为1.5倍行距）

**麦多健：**

**李吉花：**

**通过本次程序设计实践课程，我对程序设计有了全新的认识，在这个过程中接触了很多专业知识，学习了很多东西，包括Java语言的学习，以及熟悉JDK开发环境、SpringBoot框架和VUE的学习等，该课程实践性很强，因此对我个人的动手能力有很大的提升，同时也锻炼了我们团队合作的能力。课程起初的学习是有一些困难的，但是在查阅资料、主动学习，还有小组成员的帮助下，我对项目开发也有了更深入的理解，并且逐渐克服困难，主动接受并共同完成了项目的开发，并在后期体验到了项目开发的快乐。**

**洪永团：**

**张有坤：**

**刘子炜：**

1. **组内贡献度排名**

请按对项目贡献程度进行组内排名，排名最靠前的同学贡献度最大，按贡献度由大到小顺序列出组内成员名单。

**麦多健**

**刘子炜**

**洪永团**

**张有坤**

**李吉花**