### 

### 成都职业技术学院

### 毕业设计（项目技术报告）

|  |  |
| --- | --- |
| 题目 | 赛博集市 |
| 学生姓名 | 吴臻海 |
| 专业班级 | 软件203-2班 |
| 学生学号 | 20302030230 |
| 院（部） | 软件分院 |
| 指导教师 | 邓达 |

2023年　4 月 22 日

目录

[1 引言 3](#_Toc20350)

[1.1 编写目的 3](#_Toc19989)

[1.1.1 项目背景 3](#_Toc11341)

[1.1.2 专门定义 4](#_Toc31003)

[1.1.3 参考资料 5](#_Toc15362)

[1.1.4 摘要 5](#_Toc6211)

[2项目概述 7](#_Toc29135)

[2.1 设计目标 7](#_Toc17070)

[2.1.1 运行环境 7](#_Toc32316)

[2.1.2 需求概述 7](#_Toc11523)

[2.1.3 条件和限制 8](#_Toc30859)

[2.2 总体设计 8](#_Toc22990)

[2.2.1 处理流程 8](#_Toc18465)

[2.2.2 总体结构和模块外部设计 10](#_Toc28099)

[2.2.3 功能分配表明各项功能与程序结构的关系 12](#_Toc30274)

[2.3 接口设计 12](#_Toc20230)

[2.3.1 外部接口 12](#_Toc4772)

[2.3.2 内部接口 13](#_Toc12280)

[2.4 数据库结构设计 17](#_Toc27046)

[2.4.1 数据库er图展示 17](#_Toc26894)

[2.4.2 数据库表字段设计 17](#_Toc300)

[3数据结构和处理流向 22](#_Toc25280)

[3.1运行设计 22](#_Toc14703)

[3.1.1 运行模块的组合 22](#_Toc6034)

[3.1.2 运行流程 24](#_Toc6469)

[3.2 出错处理设计 29](#_Toc14277)

[3.2.1 出错输出信息 29](#_Toc18859)

[3.2.2 出错处理对策 30](#_Toc821)

[4 项目总结 31](#_Toc8853)

[4.1 感想 31](#_Toc21349)

[4.2总结 32](#_Toc31573)

摘 要

CRM软件后台管理系统主要担当了数据录入的责任，通过录入符合要求的数据，得以在前台进行显示，所以就要结合前台的业务流程以及前台的展示进行测试。最终目的就是数据可以正确的在前台展示出来。  
 CRM软件后台管理系统最大的作用就是对用户业务的支撑，一个电商如果没有上传商品的地方，没有处理订单的地方，那么用户那边什么都执行不下去;而它第二重要的地方，是对效率的提高，这部分主要是对内容的管理，比如有些功能我们要修改一下首页的Banner图片和链接，通过发布代码的形式也可以做到，但是每次改动都需要。走一遍修改代码到提交测试再到线上发布的过程，尤其是做活动的时候，这些Banner经常会改动，这些看起来不算大的小活其实消耗掉了很多战斗力,如果把这些经常改动的内容都通过后台管理系统来管理的话,那么只需要每次修改的时候打开系统上发布一个新的Banner就可以了，这样不需要开发人员介入，这个事就能很快的完成了。  
CRM软件后台管理系统的模式也比较固定，基本都会有一个功能菜单，然后每个菜单对应着一个功能模块，这个模式很固定。后台管理系统的功能归根结底也是对后端数据进行操作，也就是增删改查,所以在后端管理系统里的功能会相对简单，即使复杂也是在规模上的复杂，而逻辑上一般不会特别麻烦。

关键字：（node.js、mongodb、vue、react）

# 1 引言

## 1.1 编写目的

### 项目背景

目前，随着商品经济之发展，出现了闲置之物品，一些二手交易平台应运而生。但基于当前电商平台的惯性，传统的二手交易仍然需要货币交易，在面临货币严重贬值等特定环境时无法有效进行。本平台除了提供货币交易，还可以通过买卖双方协商确定其他等价交换物（如以物易物）来推进交易，同时留存交易凭证以维护双方权益。

在CRM系统中，承载着客户、企业、员工等各种资源。CRM一.方面对资源分门别类存放，另一方面可以对资源进行调配和重组。CRM就像魔方一样，可以根据需要千变万化地、围绕某个方面去整合资源，并允许同时从其它多个角度探寻事物的相关属性。您不妨想象这样的场景:您可以从任何一个点进入三维空间，您要到达另外一一个点，您可以选择直线或多种曲线、折线飞往目的地,一切都依据您的需要、您认为最合适的方法和途径。

### 专门定义

（1）客户合作管理

客户关系管理系统要突出以客户为中心的理念，首先应当使客户能够以各种方式与企业进行沟通交流，而客户合作管理系统就具备这项功能。

（2）业务操作管理

企业中每个部门都需要与客户进行接触，而市场营销、销售、客户服务部门与客户的接触最为频繁，因此，客户关系管理系统需要对这些部门提供支持，业务操作管理系统便应运而生。业务操作管理系统主要实现了市场营销、销售、客户服务与支持等三种基本功能。

（3）数据分析管理

数据分析管理系统中，将实现数据仓库、数据集市、数据挖掘等工作，在此基础上实现商业智能和决策分析。此系统主要负责收集、存储和分析市场、销售、服务及整个企业的各类信息，对客户进行全方位的了解，为企业市场决策提供依据，从而理顺企业资源与客户需求之间的关系，提高客户满意度，实现挖掘新客户、支持交叉销售、保持和挽留老客户、发现重点客户、支持面向特定客户的个性化服务等目标。

（4）信息管理

由于客户关系管理的各功能模块和相关系统运行都由先进的技术进行保障，因此对于信息技术的管理也成为CRM系统成功实施的关键

### 参考资料

在本文档中，引用了如下的文档和资料

-- 《软件文档编写规范 - GB2020》

--

--

--

--

# 2项目概述

## 2.1 设计目标

### 2.1.1 运行环境

### 2.1.2 需求概述

**系统的需求只要包括以下的总结性要求。本文档的设计目的就是要通过编制软件来达到这些要求**

1. 管理员需要验证身份进入系统，对客户信息和产品信息进行管理；
2. 数据量过大，可以使用关键字和过滤选择来进行筛选，进行分页展示；
3. 数据简洁化，没有繁琐的数据，对数据进行整理展示，更便于理解；
4. 前后端分离，前端页面可以更改，后端采用xml外置，可以修改一些配置；
5. 数据分析利用图表进行展示，很清晰看到数据的变化；
6. 安全性能高，对每个用户进行权限管理；
7. 每个员工权限有相应的规定，不容许跨权限操作
8. 每条数据都可以获取到相关的数据信息，方便查看

### 2.1.3 条件和限制

1.运行限制

因为是前后端分离项目，所以必须要两个端口同时运行时该项目才能正常运转，但是前台对后台数据无法返回时不会出现解析出错，所以不会出现缺一不可的情况

1. 使用限制

项目功能较多，需要对数据有一定了解，而且对功能有过操作或者了解的人员来操作，第一次没有指导的操作很难熟悉整个项目，无法熟练地掌握好使用整个项目。

## 2.2 总体设计

### 2.2.1 处理流程

1、登录判断



只有登录成功才能访问主页面，直接访问主页面将无法获取主页面信息，退回登录页面进行验证

#### 客户信息

1. 查询客户信息，带分页操作，和条件查询
2. 客户信息详情获取
3. 客户消费记录获取

#### 产品信息

1. 查询产品信息，可以条件查询
2. 编写修改产品信息，有权限要求
3. 查看产品销量分析

#### 售后信息

1. 查询售后记录，带条件和模糊查询
2. 售后处理进度，售后的处理详情
3. 售后人员的申请，获取售后详情信息

#### 员工信息

1. 查询员工基础信息，没有包含不可公开信息
2. 获取员工售后或者员工权限记录
3. 可以编辑和调整员工信息，有权限要求

#### 个人信息

1. 查看个人信息，个人对自己的信息有绝对修改权限
2. 编辑和修改可变更字段

#### 销售信息

1. 查询所有产品销售记录，分页和筛选操作
2. 获取销售的图表分析

### 2.2.2 总体结构和模块外部设计

#### 1、Node.js

使用Spring来管理JAVA类，Spring可以项目更快，更轻松，更安全地进行java编程，而且是一个轻量级的非侵入式框架，提供对事务的管理，更好的对主流框架进行集成。

#### Express.js

Express.js是一个基于Node.js的轻量级Web框架，通过把model，view，controller分离，将web层进行职责解耦，把复杂的web应用分成逻辑清晰的几部分，简化开发，减少出错，方便组内开发人员之间的配合。

清晰的角色分配：前端控制器(dispatcher servelet) , 请求到处理器映射（handler mapping), 处理器适配器（handler adapter), 视图解析器（view resolver）

#### MongoDB与Mongoose

MongoDB是一种非关系型数据库，可以减少项目的配置，减少配置文件，减少代码的编写量，项目更加精简，对整个团队的合作和配合起了至关重要的优势，避免了大量的样板代码、注解和XML配置，而且spring boot遵循“自我默认配置法”，以减少开发工作量。

#### Redis

Redis是一种缓存数据库，但是mybatis对SQL语句的编写更加清晰，能够更加清楚的操作SQL语句，而且可以对数据库返回的结果进行映射和解析，利用SQL拼接，不需要死板的模板样式，只需要修改参数的不同，以至于到达想要的效果，特别适用于数据库表的操作繁琐的项目。

#### Socket.IO

WebSocket的升级封装，可用于实时监听，配合JWT可以更高校、更快捷的管理每个不用账号的权限，而且JWT不会出现数据被泄露和向前台传递敏感信息，更安全的保护信息。

#### Vue

前端管理框架，采用了虚拟DOM技术，不用频繁操作DOM树。特有的响应式编程和组件化，双向的数据绑定，完美的数据和结构分离，极高的提升了运行速度。

vue是单页面应用，使页面局部刷新，不用每次跳转页面都要请求所有数据和DOM，这样大大加快了访问速度和提升用户体验。而且他的第三方UI库很多节省开发时间。

#### React

前端管理框架，同Vue一样，不需要频繁操作DOM，。

#### Ant Design

Ant Design是前端UI库，由阿里出品和维护，搭配React使用更加方便的展示前端优美的页面，而且内置很多js，可以提高和用户的交互需求，是国内React项目主流的UI库。

#### Tauri

与Electron类似，Tauri同样使用前端技术构建桌面应用程序，但不同于Electron，其后台使用Rust语言，相较于Electron使用的Node.js，性能、安全性都有大幅度的提高，同时使用系统自带的WebView库，使得其安装包和安装后的大小大幅度减小（可通过配置文件设置），Tauri未来会将移动端之构建引入其生产环境，大幅减小开发人员构建之压力。

#### ECharts

Element UI是前端UI库，搭配vue使用更加方便的展示前端优美的页面，而且内置很多js，可以提高和用户的交互需求，几乎是为vue而生，axios是请求数据的框架，利用axios可以做到局部刷新，减少页面的跳转和刷新，配合vue的数据双向绑定，可以更加快捷方便的对数据进行完美的展示。

### 2.2.3 功能分配表明各项功能与程序结构的关系

#### 1、后台

后台不在对页面进行操作，也不在进行页面的跳转和重定向，后台只需要复制的是数据的传递，前端访问不同的接口，后台操作数据库对数据进行格式化，然后封装发送给前端，后台做到的是权限的校验和对数据的验证，以及反馈前端的请求。最主要的是后台对各个用户的权限进行规定和限制，后台才是最安全的权限检验。

#### 前端

前端需要进行数据的展示和获取，并且处理页面的跳转和访问请求，采用的是局部刷新，所以获取数据时是靠请求来获取，当获取到数据之后，更改界面或者提示用户，不再繁琐地进行刷新和获取，根据不用的请求接口来获取不同数据，利用前端来进行展示来和用户进行交互，提升使用感受。

## 2.3 接口设计

### 2.3.1 外部接口

#### 1.登录主页面

请求路径：/login

请求参数：无

成功返回：登录页面

#### 2.获取各个操作页面

请求路径：/backstage/xxx

请求参数：无

成功返回：对应各个操作的页面

#### 3.404页面

请求不存在的路径

此时跳转404页面

### 2.3.2 内部接口

#### 1.统一请求错误返回格式

错误返回：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| code | Int | 错误编号 |
| message | String | 提示消息 |
| data | Object | 返回数据 |

#### 2.判断是否正确登录

请求接口：/login

请求参数：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| username | String | 账号 |
| password | String | 密码 |
| remember | Boolean | 是否记住密码 |

成功返回：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| JWT | String | JWT凭证 |
| name | String | 账号姓名 |
| Phone | String | 人员电话 |

#### 3.获取个人信息

请求接口：/customer/myself

请求参数：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| JWT | String | JWT凭证 |
| account | String | 账号 |

成功返回：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| name | String | 姓名 |
| account | String | 账号 |
| phone | String | 联系电话 |
| authority | Int | 权限 |

#### 4.产品页面请求

请求接口：/commodity/all

请求参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名 | 类型 | 描述 | 备注 |
| page | Integer | 当前页码 | 必选 |
| size | Integer | 页面存放大小 | 可选 |
| name | String | 名称筛选 | 可选 |
| stores | Integer | 分店筛选 | 可选 |
| stock | Integer | 库存区间 | 可选 |
| price | Float | 价格区间 | 可选 |

成功返回：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| total | Integer | 总条数 |
| list | Commodity | List链表 |
| data | String | 存放数据 |
| page | Integer | 当前页码 |
| size | Integer | 页面大小 |

#### 5.其他请求接口格式规范

请求接口：/对应菜单/对应操作

请求参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名 | 类型 | 描述 | 备注 |
| JWT | String | JWT凭证 | 必选 |
| ... | ... | ... | 其他参数 |

成功返回：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| code | String | 操作编码 |
| message | String | 提示消息 |
| ... | ... | 其他返回 |

请求时会默认带上JWT凭证，这个是证明你身份的最好凭证，后台会根据这个凭证来证明你的身份和权限，然后根据权限分配你的可访问的接口，不同的权限有不同的接口访问，而且每次操作时候都有权限判断，当权限不足时无法成功做到。

后台返回的数据格式是规定的Result格式化，有code编码，data数据，message消息通知，前端接收的一般都是data数据里面的存放的不同的数据，后台将数据编写为JSON格式返回，前端可直接接收

## 2.4 数据库结构设计

### 2.4.1 数据库表字段设计

#### 1.商品表goods

`id` int(11) NOT NULL AUTO\_INCREMENT,

`category` varchar(50) NOT NULL 分类介绍字段,

`state` int(11) NOT NULL 分类状态,

PRIMARY KEY (`id`)

#### 2.帖子表 post

`id` int(10) NOT NULL AUTO\_INCREMENT,

`code` varchar(20) NOT NULL 产品编号,

`name` varchar(50) NOT NULL 产品名称,

`picture` varchar(100) NOT NULL 产品图片路径,

`price` float(10,2) NOT NULL 产品定价,

`state` int(10) NOT NULL 产品状态,

`category` int(10) DEFAULT NULL 产品分类id,

PRIMARY KEY (`id`)

#### 3.评论表 comment

`id` int(10) NOT NULL AUTO\_INCREMENT,

`commodity\_id` int(10) NOT NULL 产品id,

`stores\_id` int(10) NOT NULL 所属分店id,

`stock` int(10) NOT NULL 库存,

`damage` int(10) NOT NULL 损坏,

`sold` int(10) NOT NULL 卖出,

PRIMARY KEY (`id`)

#### 4.交易表 exchange

`id` int(10) NOT NULL AUTO\_INCREMENT,

`name` varchar(10) NOT NULL 客户名称,

`address` varchar(50) NOT NULL 客户地址,

`phone` varchar(11) NOT NULL 客户电话,

`add\_time` date NOT NULL 客户添加时间,

`level` int(5) NOT NULL 客户等级,

`consume` float(10,2) NOT NULL 客户消费,

`stores\_id` int(10) NOT NULL 客户注册所在分店,

PRIMARY KEY (`id`)

#### 5.收藏表 favorite

`id` int(10) NOT NULL AUTO\_INCREMENT,

`customer\_name` varchar(20) NOT NULL 客户名称,

`phone` varchar(12) NOT NULL 联系电话,

`stores\_id` int(5) NOT NULL 反馈分店,

`feedback\_time` timestamp NOT NULL 反馈时间,

`remake` varchar(500) NOT NULL 备注,

`state` int(2) NOT NULL 状态,

`type\_id` int(2) NOT NULL 类型id,

PRIMARY KEY (`id`)

#### 浏览历史表 history

`id` int(10) NOT NULL AUTO\_INCREMENT,

`type` varchar(20) NOT NULL 类型名称,

PRIMARY KEY (`id`)

#### 支付表 pay

`id` int(5) NOT NULL AUTO\_INCREMENT,

`power` varchar(20) NOT NULL 权限详情,

PRIMARY KEY (`id`)

#### 用户表 user

`id` int(10) NOT NULL,

`staff\_id` int(10) NOT NULL 员工id,

`promotion\_id` int(10) NOT NULL 更改员工id,

`old\_power` int(10) NOT NULL 旧权限,

`new\_power` int(10) NOT NULL 新权限,

`promotion\_time` timestamp NOT NULL 更改时间,

`remake` varchar(100) NOT NULL 备注,

PRIMARY KEY (`id`)

#### 消息表 message

`id` int(10) NOT NULL AUTO\_INCREMENT,

`customer\_id` int(10) NOT NULL 客户id,

`stores\_id` int(10) NOT NULL 分店id,

`buy\_time` varchar(20) NOT NULL 购买时间,

`total\_price` float(20,2) NOT NULL 消费总价,

PRIMARY KEY (`id`)

#### 黑名单表 blacklist

`id` int(10) NOT NULL AUTO\_INCREMENT,

`purchase\_id` int(10) NOT NULL 购买id,

`commodity\_id` int(20) NOT NULL 产品id,

`number` int(10) NOT NULL 数量,

`discount` float(10,2) NOT NULL 折扣,

PRIMARY KEY (`id`)

#### 举报表 report

`id` int(10) NOT NULL AUTO\_INCREMENT,

`feedback\_id` int(10) NOT NULL 反馈id,

`staff\_id` int(10) NOT NULL 员工id,

`code` varchar(20) DEFAULT NULL 产品编号,

`repair\_time` date DEFAULT NULL 处理时间,

`charge` float(10,2) DEFAULT NULL 处理收费,

`state` int(10) NOT NULL 状态,

PRIMARY KEY (`id`)

#### 管理员令牌表 auth\_token

`id` int(10) NOT NULL AUTO\_INCREMENT,

`code` varchar(10) NOT NULL 员工编号，唯一,

`name` varchar(10) NOT NULL 员工名称,

`phone` varchar(11) NOT NULL 员工电话,

`authority` int(2) NOT NULL 员工权限,

`account` varchar(20) NOT NULL 员工账号,

`password` varchar(100) NOT NULL 员工加密密码,

`request\_time` timestamp NOT NULL 注册时间,

`store\_id` int(10) NOT NULL 所属分店,

`state` int(2) NOT NULL 员工状态,

PRIMARY KEY (`id`)

# 3数据结构和处理流向

## 3.1运行设计

### 3.1.1 运行模块的组合

#### 1.后端框架运行



使用spring管理所有的事务，方便调用和管理，其中，DAO层是interface接口，定义方法，对应映射的XML文件里面负责编写调用和拼接SQL语句，然后通过执行SQL语句之后获取到对应的返还结果，得到结果之后通过映射把结果格式化成对象或者链表返回给调用的函数方法，此时获取到的结果就是操作数据库的响应结果，可以是查询，也可以是添加和修改



根据不用的请求接口路径，来触发不同的controller代码，想进行参数校验，当参数全部正确，或者没有传入参数但是参数有默认值时，进行调用service的方法，在service的方法里面有获取DAO层的数据的请求，获取数据之后service会把数据整理然后返回，最后controller会把返回的数据解析成规范的格式输入给前台的请求

#### 2.前端框架运行



前端使用vue框架，当页面加载完毕之后，调用created方法，对一些属性进行初始化和加载操作，通常是请求后台的数据渲染在页面上面，此时获取的数据全是默认值获取，也不带条件查询，对后台的请求是运用的axios框架，更加方便地处理AJAX请求，当每次用户有点击、选择等操作时，有可能会需要更新数据，此时调用axios获取后台返回数据，然后通过vue的双向绑定数据，对DOM树进行修改，不再跳转页面，实现局部刷新.

### 3.1.2 客户端运行流程

#### 1.首页



在登陆页面键入账号和密码，管理员账户固定为admin，密码为123456。登录信息正确，则写入MongoDB和Redis中。

#### 2.发现（即帖子）



在登陆页面键入账号和密码，管理员账户固定为admin，密码为123456。登录信息正确，则写入MongoDB和Redis中。

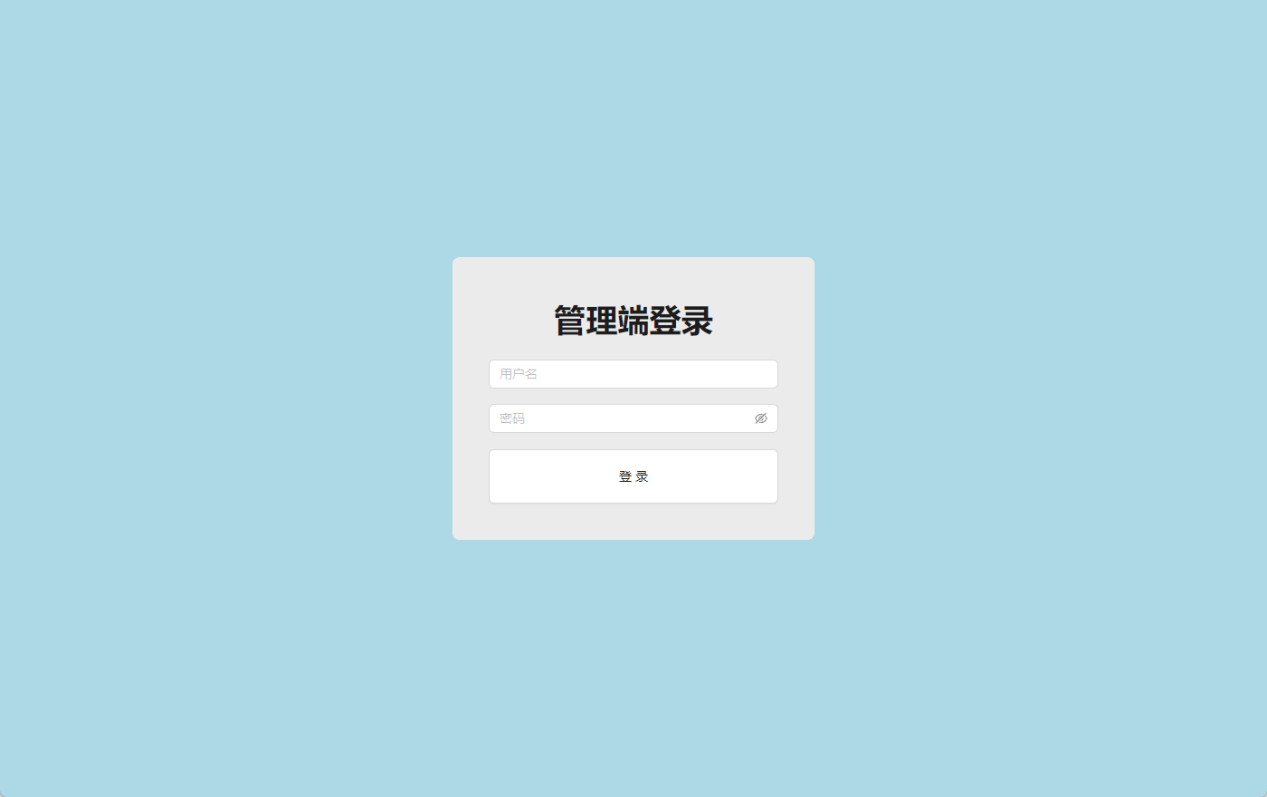
#### 1.首页



在登陆页面键入账号和密码，管理员账户固定为admin，密码为123456。登录信息正确，则写入MongoDB和Redis中。

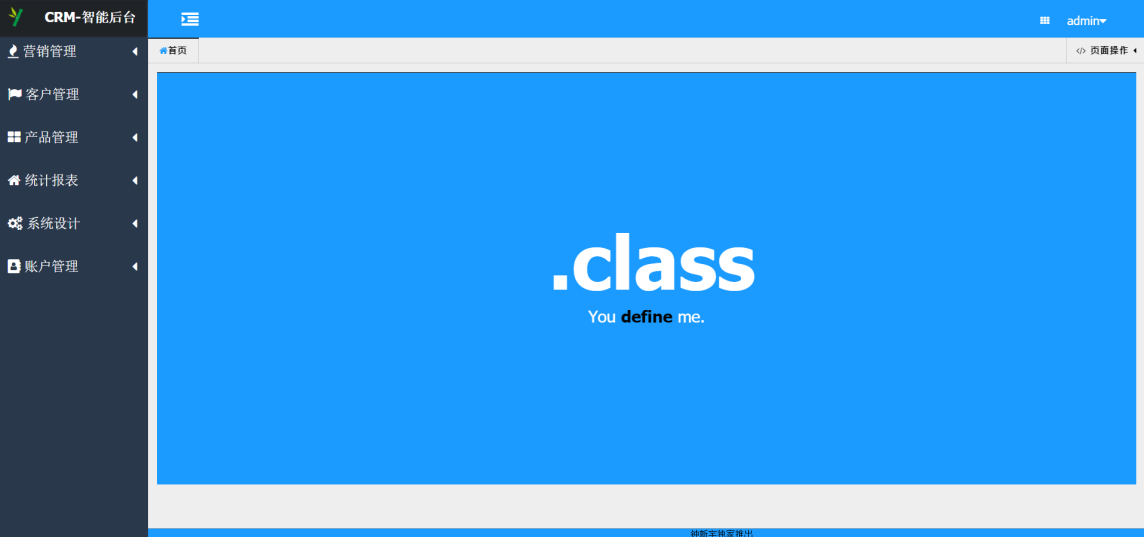
### 3.1.2 管理端运行流程

#### 1.登录，验证身份



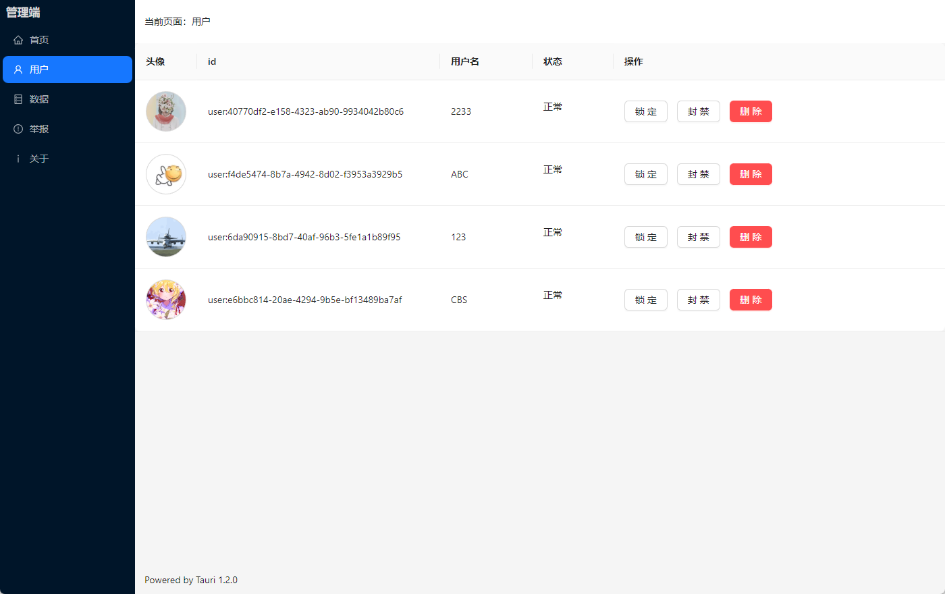
在登陆页面键入账号和密码，管理员账户固定为admin，密码为123456。登录信息正确，则写入MongoDB和Redis中。

#### 2.主页面，快速查看状况



在主页面中，左侧栏有所有的操作选择，选择每一个操作可在右边的空间进行不同的操作，此时右边头部会出现导航栏，以提示当前操作页面和当前已打开的页面浏览，可以通过点击导航栏或者左侧侧边栏进行更换不同的页面。

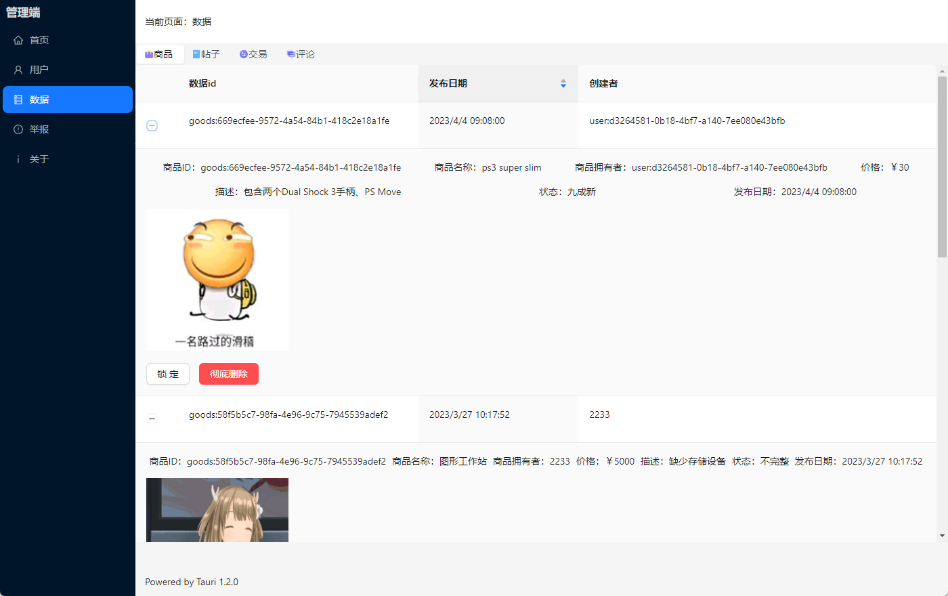
#### 3.用户管理



产品管理界面，在这里你可以看到所有的产品信息，而且将产品的信息逐一清晰的展示出来，你可以看到每个产品在各个分店的库存，卖出和损坏，其中查询产品时你还可以根据名称或者价格，以及类型进行查找，对于产品的操作，你可以对产品的信息进行编辑，但是涉及权限问题，还扣以下架产品以及重新上架产品，或者说是添加新的产品到选项里面。

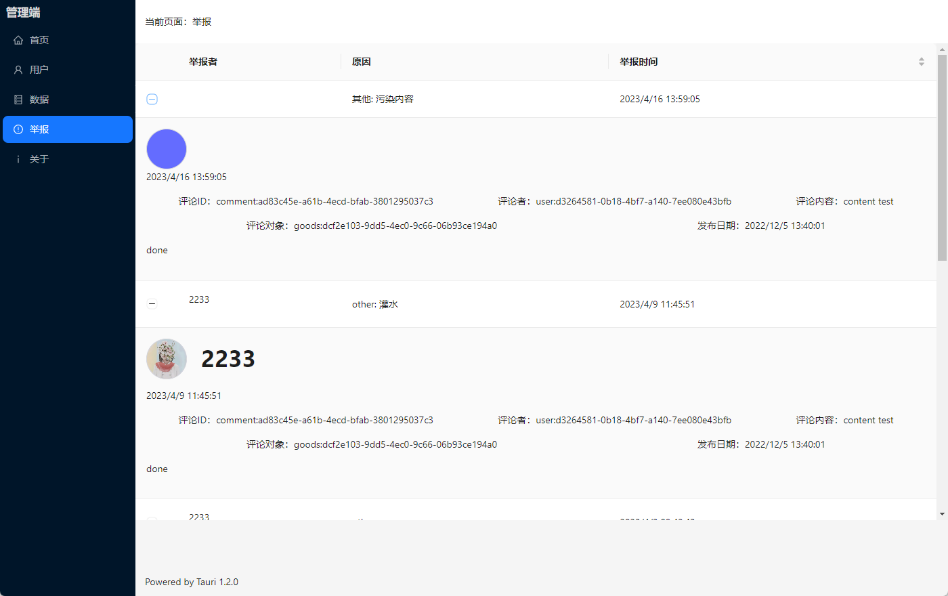
对产品信息的修改包括更改图片，所以需要向后台更换图片的时候需要选择一张图片，修改信息和图片是分开的，这样的原因避免出现错误。

#### 4.数据管理



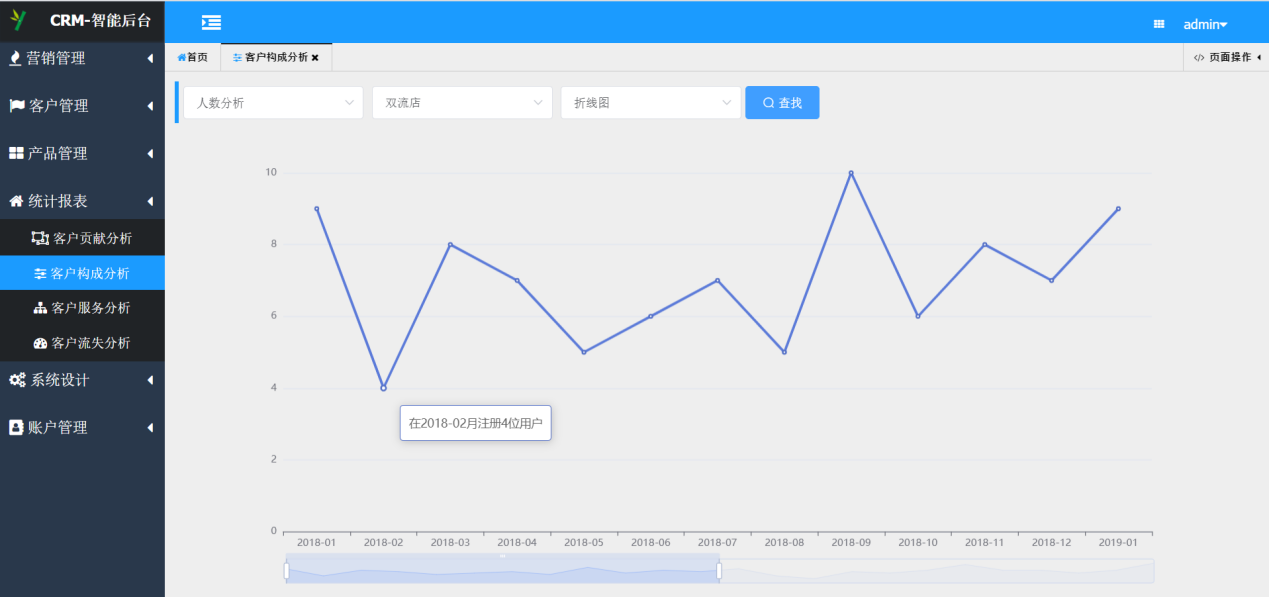
数据管理页面，本页面展示了对商品、帖子、评论和交易数据之管理，通过切换表格上的选项来预览不同类型的数据，可以通过上方的筛选条件进行条件筛选查找客户，该页面展示了客户的基本信息，你也可以通过指定客户获取该客户的消费记录。还可以看到所有客户反馈的消息，通过这些反馈可以对产品的售后进行连接，可以直接看到反馈处理进度之类的。

#### 5.举报管理

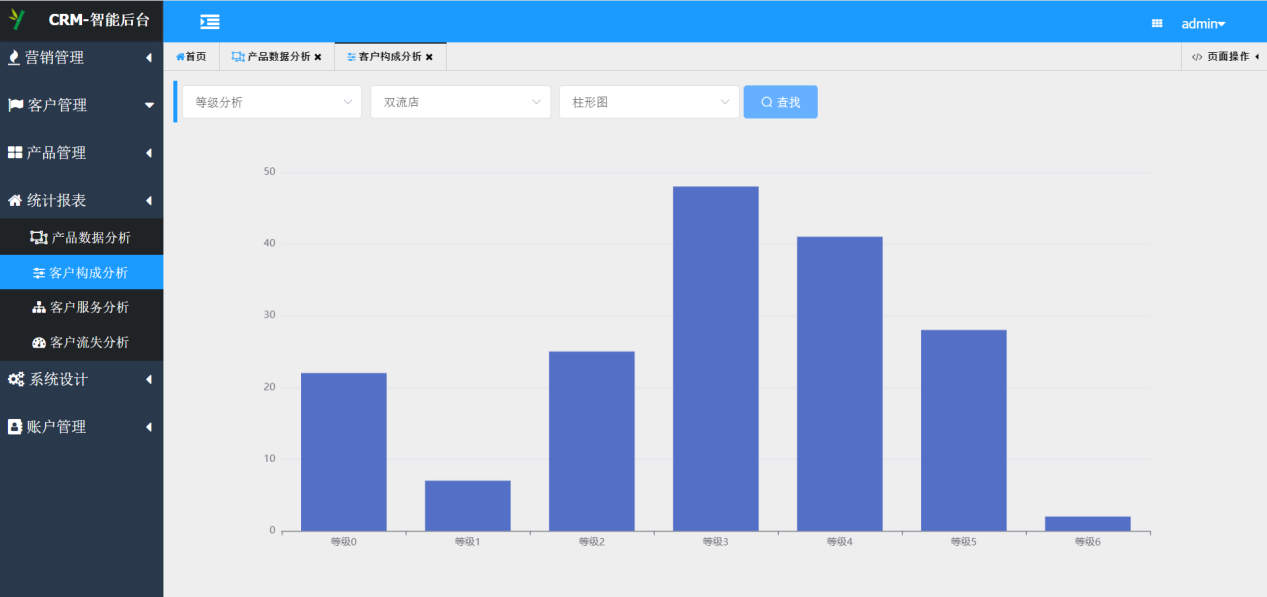


员工信息界面，以及个人信息界面。展示员工信息，展示员工所在分店、姓名、联系电话，可以根据姓名、编号、注册时间进行帅选查找，选择一个员工，可以查看该员工的详细信息，查询时可以对员工信息进行编辑和修改，不过必须是分店级管理，员工根据个人信息照看，可以修改自己力所能及的信息，列如上传图片，更换联系电话，查看自己的历史记录等。

#### 6.数据分析



数据分析页面，在几个页面中，对各个数据进行分析、整理以及展示，数据展示分为了折现以及柱状图，可以更加直观地看清楚数据的流动和行走趋势，在数据展示中可以清晰获取到数据的多少和分布，根据不同的选择，后台会发送不同的数据展示格式，以方便于数据的理解，列如可以获取所有分店的产品分布，也可以获取指定一个分店的所有产品的详情，还可以获取客户在不同分店的等级或者人数整理



数据展示有几种方式，折线能更加清晰看到数据的递增或者递增走向，而柱形图可以看清数据的大小比较更加直白，饼图可以看到几组数据的占比比例。

#### 售后处理信息



售后处理，方便与处理客户购买产品之后产生的问题，然后派遣人员前往处理，一直监督处理进度。还扣以在售后界面中，查询该客户一单订单的详情和记录，可以根据客户的信息安排处理人员和处理方式，更加明白客户的需要，更清楚地提高客户的感受

其中每个售后都是由一名人员来进行的，所以可以根据人员来查找售后进度或者售后处理详情，可以获取该售后的所有详情更方便

## 3.2 出错处理设计

### 3.2.1 出错输出信息

#### 1.控制台输出

在IDEA的运行控制台中，输出代码错误信息，一般在控制台输出的信息，除了人为，其他都未预料到或者说未捕获到的异常错误，在控制台会输出错误索引和错误信息简介，可以根据提示消息找寻错误的代码所在位置，以做修正

#### 2.接口返回信息输出

在前端请求数据时，如果未带必选参数，或者进行无权操作，获取访问不属于自己可以接触的信息时，后台会向前端发送错误提示，并且带上信息和错误编码，这种错误都是可预料的、可捕获的，属于项目中可以预料和判断的错误。

### 3.2.2 出错处理对策

#### 1.全局异常捕获

JAVA采用全局异常捕获，当数据出现异常之时，不会影响到数据的传递以及程序的运行，此时根据不同的异常处理，会在控制台输出不同的异常信息，而且会返回给前台异常错误的信息，诸如：code：200，message:”传递数据为空，请查找数据”，此时数据已经返回，不会继续执行后面的语句

#### 2.弹窗提醒

前端是采用前后端分离处理方法，所以每次获取数据的时候，不是进行刷新页面，而是用AJAX来进行异步获取数据，所以当获取数据失败的时候，我可以提前预测这个情况的发生，在JS文件中，在每次接收到返回的response时，后台返回的都是规定的result格式，所以判断返回的code就知道是否是错误或者失败的请求，当code编码不是400时，代表请求出现错误，然后根据错误的信息和编码，通过Element UI 来调用弹窗提醒操作人员，操作人员就可以知道自己的错误操作所在

4 项目总结

## 4.1 感想

今年形势异常困难，，但我仍然克服了它们，以较高的水准完成了这个设计。

技术上，。

## 4.2总结

时间飞逝，转眼间就到了毕业。大学生活历历在目，在这四年的学习生活中，收获了很多，而这些成绩的取得是和一直关心帮助我的人分不开的。在大学最后的时间里，很认真的做完了毕业设计，因为它是我大学的最后一次作业了。完成了它我大学生活接近尾声了。回想我的四年大学生活真的有很多的收获和感触。在这里的四年时间我从一个懵懂少年成长为一个懂事懂礼懂得做人的人，在这里我不但学到了文化知识，实践动手能力，更形成了我的许多价值观，学会了许多做人做事的道理。最后谢谢我的母校对我的教育，我将以更加饱满的热情投入今后的工作中，去完成自己的理想!