



文本复制检测报告单(全文标明引文)

No: ADBD2020R_20200507151307448552239876

检测时间: 2020-05-07 15:13:07

检测文献: 基于Web的无人值守点餐收银系统的设计与实现

作者: 彭政

检测范围: 中国学术期刊网络出版总库

中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库

中国重要会议论文全文数据库

中国重要报纸全文数据库

中国专利全文数据库

图书资源

优先出版文献库

大学生论文联合比对库

互联网资源(包含贴吧等论坛资源)

英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)

港澳台学术文献库

互联网文档资源

源代码库

CNKI大成编客-原创作品库

个人比对库

时间范围: 1900-01-01至2020-05-07

检测结果

去除本人已发表文献复制比: 13.9%

跨语言检测结果: 0%

引 去除引用文献复制比: 6.3%

总 总文字复制比: 13.9%

单 单篇最大文字复制比: 7.4% (基于Web前端组件化的个人博客系统的设计与实现)

重复字数: [2915] 总字数: [21033] 单篇最大重复字数: [1563]

总段落数: [2] 前部重合字数: [280] 疑似段落最大重合字数: [1739]

疑似段落数: [2] 后部重合字数: [2635] 疑似段落最小重合字数: [1176]

指 标: ☐ 疑似剽窃观点 ☒ 疑似剽窃文字表述 ☐ 疑自我剽窃☐ 一稿多投 ☐ 过度引用

表格: 1

公式: 没有公式

疑似文字的图片: 0

脚注与尾注: 0

18.1% (1739) 基于Web的无人值守点餐收银系统的设计与实现.doc 第1部分 (总9606字)

10.3% (1176) 基于Web的无人值守点餐收银系统的设计与实现.doc 第2部分 (总11427字)

(注释: ■ 无问题部分 ■ 文字复制比部分 ■ 引用部分)

指导教师审查结果

指导教师: 魏崇奎

审阅结果:

审阅意见: 指导老师未填写审阅意见

1. 基于Web的无人值守点餐收银系统的设计与实现.doc_第1部分

总字数: 9606

相似文献列表

去除本人已发表文献复制比: 18.1%(1739) 文字复制比: 18.1%(1739) 疑似剽窃观点 (0)

7.1% (683)

1	<u>基于Web前端组件化的个人博客系统的设计与实现</u> 曾广海(导师: 卢力) - 《华中科技大学硕士论文》 - 2016-12-01	是否引证: 是
2	基于Vue的个人博客设计与实现 程丽彬 - 《大学生论文联合比对库》 - 2019-05-26	3.0% (291) 是否引证: 否
3	基于Vue的个人博客设计 程丽彬 - 《大学生论文联合比对库》 - 2019-06-04	3.0% (291) 是否引证: 否
4	基于Vue的个人博客设计与实现 程丽彬 - 《大学生论文联合比对库》 - 2019-06-12	3.0% (291) 是否引证: 否
5	1408080113-孙晨阳-基于vue 仿美团外卖全栈项目的设计与实现 (1) 孙晨阳 - 《大学生论文联合比对库》 - 2018-06-06	2.5% (242) 是否引证: 否
6	基于Web的移动订餐系统设计与实现 吕继鹏 - 《大学生论文联合比对库》 - 2019-06-05	2.3% (224) 是否引证: 否
7	docker镜像管理系统的设计与实现(开题报告+文献综述) 蔡海强 - 《大学生论文联合比对库》 - 2019-03-27	2.2% (213) 是否引证: 否
8	计算机应用基础题库管理系统 张春婷 - 《大学生论文联合比对库》 - 2017-06-08	2.1% (199) 是否引证: 否
9	七天学会NodeJS_云天妈妈 - 《网络 (http://blog.sina.com) 》 - 2018	2.1% (199) 是否引证: 否
10	七天学会NodeJS - 《互联网文档资源 (http://www.360doc.co) 》 - 2015	2.1% (199) 是否引证: 否
11	七天学会NodeJS - 《互联网文档资源 (http://www.360doc.co) 》 - 2017	2.1% (199) 是否引证: 否
12	基于HTML5/CSS3的动画图片播放器设计与实现 王志刚 - 《大学生论文联合比对库》 - 2015-05-18	2.0% (191) 是否引证: 否
13	响应式开发的计算机图书社区 谢飞龙 - 《大学生论文联合比对库》 - 2018-04-29	1.8% (174) 是否引证: 否
14	基础vue的移动web App的设计与实现 郭晓琴 - 《大学生论文联合比对库》 - 2018-06-11	1.8% (173) 是否引证: 否
15	基于hybrid开发在IOS平台下订餐系统的研究 方帆 - 《大学生论文联合比对库》 - 2017-04-11	1.7% (164) 是否引证: 否
16	基于hybrid开发在IOS平台下订餐系统的研究 方帆 - 《大学生论文联合比对库》 - 2017-04-17	1.7% (164) 是否引证: 否
17	基于hybrid开发在IOS平台下订餐系统的研究 方帆 - 《大学生论文联合比对库》 - 2017-04-24	1.7% (164) 是否引证: 否
18	2013020100013_方帆_计算机科学与技术_基于 hybrid 开发在 IOS 平台下订餐系统的研究_赵彦超 方帆 - 《大学生论文联合比对库》 - 2017-05-02	1.7% (164) 是否引证: 否
19	49_基于MongoDB的云数据库服务系统的设计与实现 基于MongoDB的云数据库服务系统的设计与实现 - 《大学生论文联合比对库》 - 2015-05-25	1.7% (164) 是否引证: 否
20	基于HTML5/CSS3的动画图片播放器设计与实现 王志刚 - 《大学生论文联合比对库》 - 2015-05-14	1.5% (147) 是否引证: 否
21	基于HTML5坦克大战游戏设计与开发 - 《大学生论文联合比对库》 - 2018-05-16	1.5% (147) 是否引证: 否
22	从HTML5移动应用现状谈发展趋势 - 蒋宇捷的专栏 - 博客频道 - CSDN.NET - 《网络 (http://blog.csdn.net) 》 - 2017	1.5% (147) 是否引证: 否
23	基于HTML5的物业管理系统的设计与实现 刘玲 - 《大学生论文联合比对库》 - 2018-05-17	1.5% (147) 是否引证: 否
24	<u>基于大数据时代的数据库与传统数据库的比较研究</u> 蔡伟(导师: 李子强;许志国) - 《湖北工业大学硕士论文》 - 2017-05-20	1.3% (128) 是否引证: 否
		1.1% (103)

25	JC154093 乐昱显 毕业论文 乐昱显 - 《大学生论文联合比对库》 - 2019-04-23	是否引证: 否
26	基于HTML5与CSS3的电子阅读器WebApp的开发与实现 李志远 - 《大学生论文联合比对库》 - 2017-05-25	1.0% (92) 是否引证: 否
27	04131031_欧莹莹_基于Vue.js和MVVM模式下的前端框架研究 欧莹莹 - 《大学生论文联合比对库》 - 2017-04-13	0.8% (78) 是否引证: 否
28	2012110432_龙舟_web爬虫的研究与实现 龙舟 - 《大学生论文联合比对库》 - 2016-06-14	0.7% (63) 是否引证: 否
29	社会经济统计数据可视化系统分析与设计 魏倩(导师: 毛澄映) - 《云南大学硕士论文》 - 2012-10-01	0.4% (38) 是否引证: 否
30	沈阳市发改委重大项目信息监管系统设计与实现 李乐诗(导师: 杨永健;李荣) - 《吉林大学硕士论文》 - 2016-06-01	0.3% (31) 是否引证: 否

原文内容

摘要

随着互联网的蓬勃的发展,智能化点餐系统完全符合当今人们的生活方式,也更能实现餐厅的智能化, 本课题研究无人值守的点餐收银系统的设计和实现, 借鉴无人超市的设计理念, 为商家和消费者带来极致体验, 感受互联网时代的高效率与便捷。

当今web技术的蓬勃发展,用H5和JavaScript实现移动端web方便快捷,兼容性好。本系统将用到Vue框架采用MVVM渲染方式, 该系统主要采用移动端M站布局的 (rem、sass)、Vue、Vuex、Vue+Sass、Koa2 接口、微信Js sdk (扫码、拍照、微信支付)、Vue支付宝支付、Vue微信支付等多设备和平台接入, 平台要具有较强的实用价值和可用性。

本论文主要从研究背景、概要设计、详细设计、开发环境及技术、调试与测试等方面对这款移动web系统的设计与实现进行描述。

关键词: 智能点餐; 移动web; JavaScript; H5; [Vue](#)

[ABSTRACT](#)

[With the flourishing development of the Internet](#), intelligent ordering system completely conforms to the modern people's life style, also can implement the intelligent of the restaurant, this topic research the unattended system design and implementation of the order of the design concept of reference no supermarket, extreme experience for businesses and consumers, high efficiency and convenience of the Internet era.

With the rapid development of web technology, H5 and JavaScript are used to realize the convenience and compatibility of mobile web. This system will use Vue framework USES MVVM rendering mode, the system mainly USES the mobile end M stand layout (rem, sass), Vue, Vuex, Vue + sass, Koa2 interface, WeChat Jssdk (scan code and take photos, WeChat payments), Vue pay treasure payment, Vue WeChat pay multiple devices and platforms, such as access, platform has strong practical value and availability.

This paper mainly describes the design and implementation of this mobile web system from the aspects of research background, brief design, detailed design, development environment and technology, debugging and testing.

Key words: Order smart; Mobile web; JavaScript; Vue

1 绪论

1.1 研究背景

如今的餐饮行业, 早就不是停留在能够满足温饱的需求上了。大街上到处都是餐馆, 可以满足您的要求, 但是就是这样。一些快餐店深受顾客欢迎, 而一些餐馆则被忽略。综观所有这些现象, 它表明现代饮食不再像简单地制作食物那样简单, 例如口味, 文化, 管理, 服务, 经验, 宣传等。如果做得不好, 将会失去机会。随着食品和饮料的发展, 人们的订购模式已经从口头, 纸笔书写转变为计算机的发展。

对比传统点餐, 可以分为三代点餐系统, 在第一代点餐中, 可以理解为消费者前往餐厅消费, 顾客到店入座, 等待服务员点餐, 服务员上前来拿到一本店菜单提供给顾客挑选, 顾客挑选完成后服务员记录下菜品, 同时服务员在面对顾客寻味菜品时还需要耐心准确的解答, 顾客在点餐完成后服务员将记录的菜品提交给后厨, 用餐过程中, 加菜、呼叫等需求, 用餐之后, 服务员结账。该点餐的方式存在许多问题, 结账还要排队菜单不够用, 走进餐厅没有人理, 等半天服务员才来点菜。当餐厅来的多桌客人, 那么服务员同时要照顾多桌客人, 如果服务员人数少, 那么

服务员必须的一桌点完再去点下一桌,将会很容易导致出错,想要同时照顾好每一位客人必须增加服务员,这样餐厅就会大大增加人力成本。为了解决这样繁琐的点餐方式,许多餐厅提供了第二代点餐系统,当消费者去到餐厅中,服务员会提供一个平板电脑,顾客在平板电脑上进行点单,顾客可以直接在平板上进行点餐提交,提交后直接提交到后厨。该方式的点餐系统提供了很大的升级,大大减少了出错的概率,但是该点餐人然存在人力成本的问题,增加了硬件设备的成本。现如今,人们生活质量随着互联网的发展从而逐渐提高,人们急需一款简单、方便、快捷的点餐方式,因为这样的自助点餐、支付,客观上为餐厅解放了劳动力,减少了人工成本。

1.2 研究意义

为了解决以上点餐系统存在的弊端,设计一款基于Web的无人值守点餐收银系统是完成符合当今人们的生活方式,该点餐系统完成无需无人值守,无人收银,消费者只需扫描该桌的二维码后进入代餐系统,查看菜品,同时可以查看菜品销量,点餐完成后系统提交给后厨查看,用餐结束后直接在系统内点击支付,可以支持微信支付方式,同时支持多桌同时扫码点餐,这样解决了传统点餐的人力成本,出错率高的一系列问题。随着电子商务的繁荣和移动互联网的极速发展,生活中随处可见二维码,用户进店扫描桌上的二维码就可以直接选菜下单,自助点餐方便易用,不仅要所谓的好的体验,更要简单、方便、快捷。让顾客参与到点餐、支付的过程中来,这是顾客愿意做的,也是餐厅乐意去推动的事情,因为这样的自助点餐、支付,客观上为餐厅解放了劳动力,减少了人工成本[1]。

基于H5的近年来的高速发展,使用HTML5开发的页面更具有现代网页的特性:界面华丽、人机交互出色、功能强大,现在我们已经很难单纯用传统的表现方式满足用户多种多样的需求,实现时也很难将HTML5与之前的版本割裂开来,所以我们可以认为在移动平台上绝大部分的现代Web App(或者Hybrid App)都将会采用HTML5开发[2]。

1.3 主要研究内容

本课题研究的是一个基于移动Web的H5点餐系统,本系统试图将用户的极致体验为主题,为每一位用户设计一款“即扫即点”的点餐系统,不仅要实现餐厅的基本的点餐需求,还需要为餐厅解决顾客对餐厅留下好印象来增加回头客的需求。因此,本系统主要的功能包括会员个人中心,菜品展示至菜品详情,详情页选择菜品至待购购物车,订单展示与提交支付,四个大的功能模块。

为了完成上述的工作,首先要完成的是对于移动Web技术以及H5技术的研究。本系统的设计前端和后端是采用JavaScript语言编写,前端使用的Vue.js框架,后端使用的Koa,数据库将采用MongoDB

H5应用随着Web应用程序的出现而出现的,并且是逐步发展起来的。在未来的发展中,移动Web应用的起点要高于PC Web应用。移动Web应用程序将是Web应用程序的扩展。从开发的角度来看,它们应该是相同的。

本文对基于Web的无人值守点餐收银系统的设计与实现划分七个章节描述,主要章节内容如下:

第一章为绪论,主要对系统研究背景与意义以及在本章最后的对研究内容进行阐述。

第二章为关键技术分析,介绍本系统中运用的主要技术,以及阐明选择该技术的目的。

第三章为系统需求分析,从系统设计背景和意义出发对于的需要实现的需求进行分析。

第四章为系统总体设计,分系统总体设计,前端总体的设计,后端总体设计,和数据库的设计。

第五章为系统的功能模块的实现的分析,分功能模块的介绍主要功能实现的方法。

第六章为最后对系统测试的分析,主要画出了测试的用列表,并将功能的测试效果图进行分析。

第七章就是对次的设计总结,对系统的设计做了详细的总结,通过该系统设计自己的收获与成长,并对本次实现过程所用的技术做了简单的回顾。

2 关键技术分析

本章将简要分析在系统开发的过程中所用的关键技术,包括项目的开发语言、构建的框架,项目的MVVM设计模式,以及组件化模块化的编程思想和项目工程化的相关技术,后端数据库的选择,以及各个技术在开发中的优点。

2.1 系统开发环境

本系统的开发为移动Web的开发不涉及硬件设备,主要用的开发环境和使用的开发工具主要有Windows10,处理器是Intel i7-7500U CPU,开发工具是Visual Studio Code,开发语言JavaScript、HTML、CSS,后端环境是Node.js12.16.1版本,数据库是MongoDB数据库,项目的预览运行使用Chrome浏览器进行测试。

2.2 语言JavaScript

JavaScript是解释型语言具有跨平台部署简单的优点,同时也是一门弱类型语言。具有灵活,易上手的优点,但不严谨就成了他的缺点,JS被用来创建整个应用程序,称为SPA(单页应用程序)[3]。当今用于开发JS的大多数Web工具和库,很多用JS写的。JS也被用除前端方面的领域。使用Node.js咱们可以创建服务器端和物联网应用

程序，工业设备，以及更多。但最重要的是，单页应用程序是 JS 最突出的用法之一。在单页面应用中，JS 负责所有的事情，使 UI 流畅，无需任何页面刷新[3]。

2.3 前端框架Vue.js

Vue.js使用前，要明确他是用来构建用户界面的渐进式框架，他有着轻量 and 简洁的特点，Vue.js只要关注的是Web开发应用中的视图层，它能够很容易的跟JavaScript的库进行整合使用，它能够支持大型的Web项目。，使用Vue.js能够在总体上更节约成本,通过使用Vue.js从而提高开发效率和效果。Vue.js有很多重要特点给开发带来了很多便捷。具有以下的重要的特点。

(1) 组件化

在组件化开发是如今构建Web项目的首选开发方式。大型项目中，把多个不同的应用抽离成独立的模块，每个模块封装独有的功能，项目中如果需要使用相应的功能只需要在组件库中选择功能模块进行调用。组件化开发便于团队的合作，减少了没有必要的重复编码，减少了代码的冗余，提高了项目的性能。任何应用都可以拆分成一个个组件进行管理和复用，如果将整个应用比喻成一棵树，那么这个应用会自上而下被拆分成独立的模块，形成一棵“组件树”[4]。

Vue.js的使用完全支持组件化开发，并且组件化也成为了它最大的特点之一，将功能模块封装成能够重复调用的组件，组件也可以理解成一个自定义的元素，Vue.js对该自定义的元素进行编译器的编译，就为这个自定义的元素添加了对应的功能。在Vue.js中使用组件化开发有着提高开发效率、方便重复使用、简化调试步骤、提升整个项目的可维护性、便于多人协同开发等优点。

(2) 响应式

在Vue.js中保持组件的状态跟视图能够同步就是用了其响应式的特点。对于Vue.js而言，它的核心原理数据响应式，它是基于Object.defineProperty来实现数据的响应，Vue.js通过Object.defineProperty的getter/setter对收集的依赖项进行监听,在属性被访问和修改时通知变化,进而更新视图数据，Vue不能检测到对象属性的添加或删除。由于Vue会在初始化实例的时候对属性进行 getter/setter 转化，所以属性必须在data对象上存在才能让Vue转换它，这样才能让它响应的[4]。Vue.js响应式的原理图如图2.1所示。

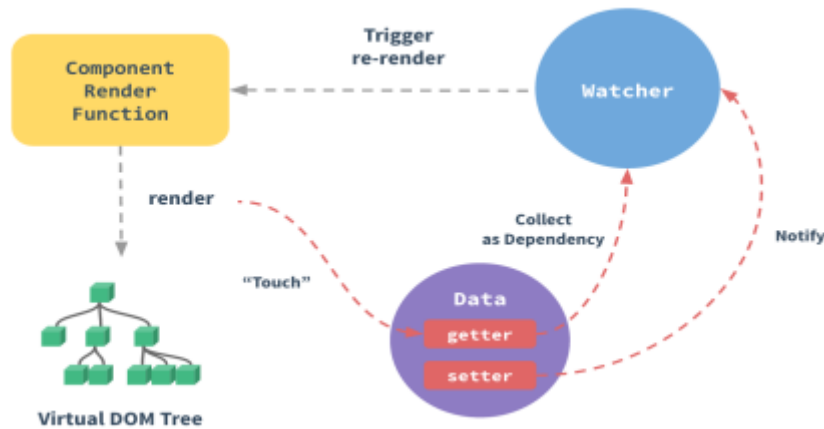


图2.1 Vue.js的响应式原理图

(3) 路由

构建一个单页面应用程序的时候使用Vue.js和Vue Router是很容易创建的，在项目中组成项目程序是通过组合组件来组成的。Vue Router是Vue.js官方的路由管理器。它和Vue.js的核心深度集成，让构建单页面应用变得易如反掌，包含的功能包括参数通过路由传递，路由通过设定访问的路径，将访问的路径跟视图组件连接起来[5]。在Web的单页面应用中，路由之间的跳转与切换实际上就是对应的视图组件之间的跳转，在Vue.js中本身是不具有路由的功能的，所以需要引用Vue Router这个路由库来相互协作，Vue Router支持不同层级的嵌套式路由与相应的嵌套组件的连接关系，为开发者提供了一个细致的控制路径跳转的解决方案，受到广大开发者的青睐。

(4) 状态管理

状态的管理同样也是Vue.js的核心的解决方案的特点，每次视图的改变都可以理解为Vue.js管理的状态改变而造成的，对于大型的Web应用程序，会存在较多的组件，而每个组件都会维持它应有的状态，从而需要对这些状态进行管理。Vuex就是一个专为Vue.js应用程序开发的状态管理模式。它采用的是集中式存储管理应用的所有组件的状态，同时以对应的规则确保状态以一种能够预测的方法发生变化。每一个Vuex应用的核心就是store（仓库），“store”基本上就是一个容器，它包含着你的应用中大部分的状态(state)[5]。

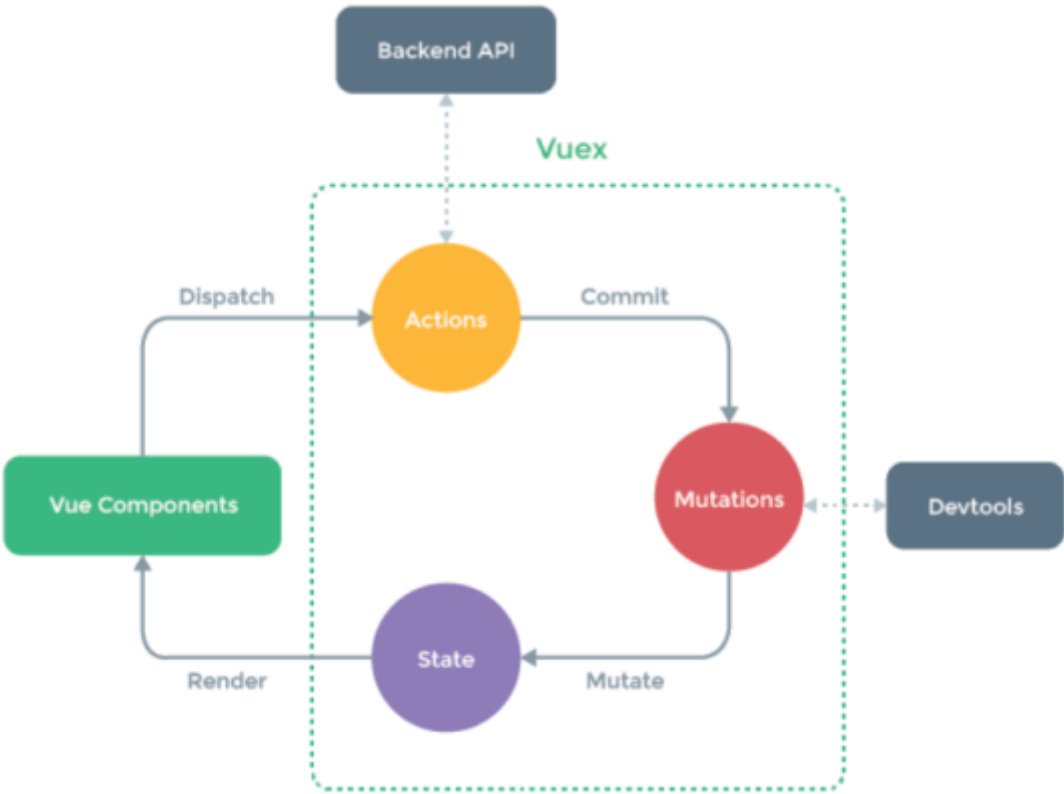


图2.2 Vue.js状态管理原理图

2.4 后端Node.js

JavaScript原本是在浏览器环境下运行的，不能在服务器端运行，这样想要做一个开发者去做前端后端同时开发需要学习前端的JavaScript语言和后端的服务器端语言，这样就极大程度的增加了开发的成本，Node.js可以理解为js的在服务器端的运行环境，随意Node.js的发展，得到了越来越多的开发者的认可，它将前端与后端的开发语言相统一[6]。最开始创造Node.js的目的是为了实现高性能Web服务器，首先看重的是事件机制和异步IO模型的优越性，而不是JavaScript。但是需要选择一种编程语言实现此想法，这种编程语言不能自带IO功能，并且需要能良好支持事件机制。[6]JS没有自带IO功能，天生就用于处理浏览器中的DOM事件，并且拥有一大群程序员，因此就成为了天然的选择。NodeJS在服务端活跃起来，出现了大批基于NodeJS的Web服务。Node.js具有他的突出的地方，比如实现前后端的语言统一，性能也是非常的突出，它是基于chrome的V8引擎运行的，同时也拥有高并发低消耗的特点，还可以支持跨平台的使用[7]。

2.5 设计模式MVVM

对于MVVM设计模式一般需要跟MVC设计模式进行对比。MVC的设计模式式很直观的架构模式，当用户操作的时候View接收到操作者的输入操作，给Controller做业务逻辑的处理，Model用来实现数据持久化，最后将得到的结果反馈给View，这就是完成了一次MVC模式[8]。MVVM是数据双向绑定的思想为核心理念，View和Mode之间ViewModel交互，Model和ViewModel两者的交互为双向的，所以当视图的数据发生变化的时候会修改数据源，数据源数据的变化的时候也会立即影响到View上,所以ViewModel 成为 View 信息的存储结构，ViewModel与View上的信息成了一对一的映射关系[8]。

MVVM在开发中有很多好处,低耦合，视图可以独立于模型进行更改和修改,ViewModel可以绑定到其他视图,视图更改时模型可以保持不变，模型更改时视图可以保持不变,可重用性,您可以在ViewModel中放置一些视图逻辑，并允许许多视图重用此视图逻辑,自主开发,开发人员可以专注于业务逻辑和数据（ViewModel）的开发[9]。设计师可以专注于界面（视图）的设计。可测试性,可以测试ViewModel的接口（视图）[10]。MVVM设计模式的数据模型关系图如图2.3。

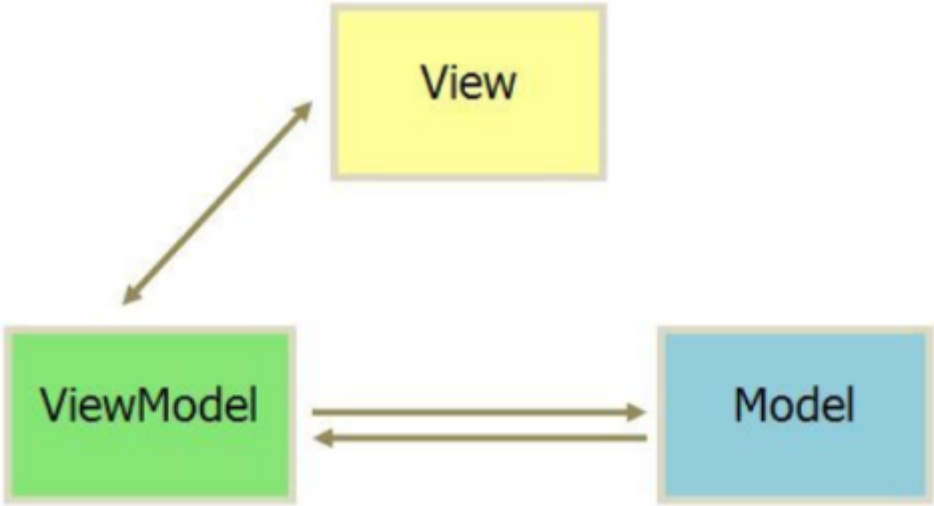


图2.3 MVVM设计模式数据模型关系图

2.4 MongoDB数据库

MongoDB 是一个基于分布式文件存储的数据库，由 C++ 编程语言开发而成，主要是为 Web 应用的可扩展的高性能数据存储提供的一种解决方案[10]。MongoDB 不仅具有良好的文档，强大的社区支持，还对有 SQL 数据库背景的开发人员非常友好[11]。MongoDB是一个文档数据库，这意味着它将数据存储在与JSON文档类似的结构中，这是考虑数据的最自然方法，比传统的行/列模型更具表现力和功能。丰富的JSON文档支持数组和嵌套对象的作为值，允许灵活和动态的模式。查询本身就是JSON，因此很容易组合。不再需要串联字符串来动态生成SQL查询。支持聚合和其他现代用例，例如基于地理的搜索，图形搜索和文本搜索。丰富而富有表现力的查询语言，使您可以按任意字段进行过滤和排序，无论它在文档中有多嵌套。MongoDB是一个文档数据库，具有所需的可伸缩性和灵活性，可用于所需的查询和索引编制。

MongoDB 是目前软件开发行业的 NoSQL 数据库中最发展最快的开源型数据库之一，MongoDB 的集合可以是松散的数据类型，也可以支持复杂的属性，所以其可以被灵活的调整[12]。在同类的 NoSQL 产品中，MongoDB 还允许直接对记录的某个字段进行原子级别的修改，目前在 NoSQL 产品中还支持此功能特性的已很少见[12]。

Node.js与MongoDB搭配使用是目前比较完美的Node.js的Web应用，他的增删该查管理数据库的命令又与JavaScript语法是非常相似的，对于开发者是非常友好的。MongoDB是非关系型数据库，要了解非关系型数据库就必须先了解关系型数据库，关系数据库，是建立在关系模型基础上的数据库[13]。比较有名气的关系型数据库，比如Oracle、DB2、MSSQL、Mysql。非关系型数据库的实质：非关系型数据库产品是传统关系型数据库的功能阉割版，通过减少用不到或很少用的功能，来大幅度提高产品性能，实际开发中，很多业务需求，其实并不需要完整的关系型数据库功能，非关系型数据库的功能就足够使用了。这种情况下，使用性能更高、成本更低的非关系型数据库当然是更明智的选择[14]。使用schema对一个数据库的结构进行描述。在一个关系型数据库里面，schema定义了表、每个表的字段，还有表和字段之间的关系。

3 系统需求分析

对系统的分析与设计是在系统实施之前最重要的部分。本章包含对系统需求的分析以及有关其设计的详细信息。在做描述系统的需求分析时，主要关注功能需求分析和用户需求分析。功能需求的描述主要与系统使用的实践相结合，根据该需求，用户必须根据用户的各种角色提出需求。该系统主要在两个方向上开发：总体系统设计和基本功能模块的开发。在系统的总体结构和模块化组成中描述了总体系统设计，并且根据各种功能模块描述了系统功能的模块化模块。

3.1 系统结构分析

在系统设计之前要明确系统的需求是为了达成什么样的目标，一个明确系统的用户为消费者，在功能设计的时候一定要做到页面简洁，功能丰富，用户通过扫码进入移动Web页面自主点餐下单并付款，享受美味的美食。该系统要设计成一个消费者与店家之间的桥梁，提供消费者完全自主的消费体验，为餐厅最大的节约成本，最大体现移动Web给当今社会带来的便捷。对于系统的总体分析思维导图如图3.1所示。

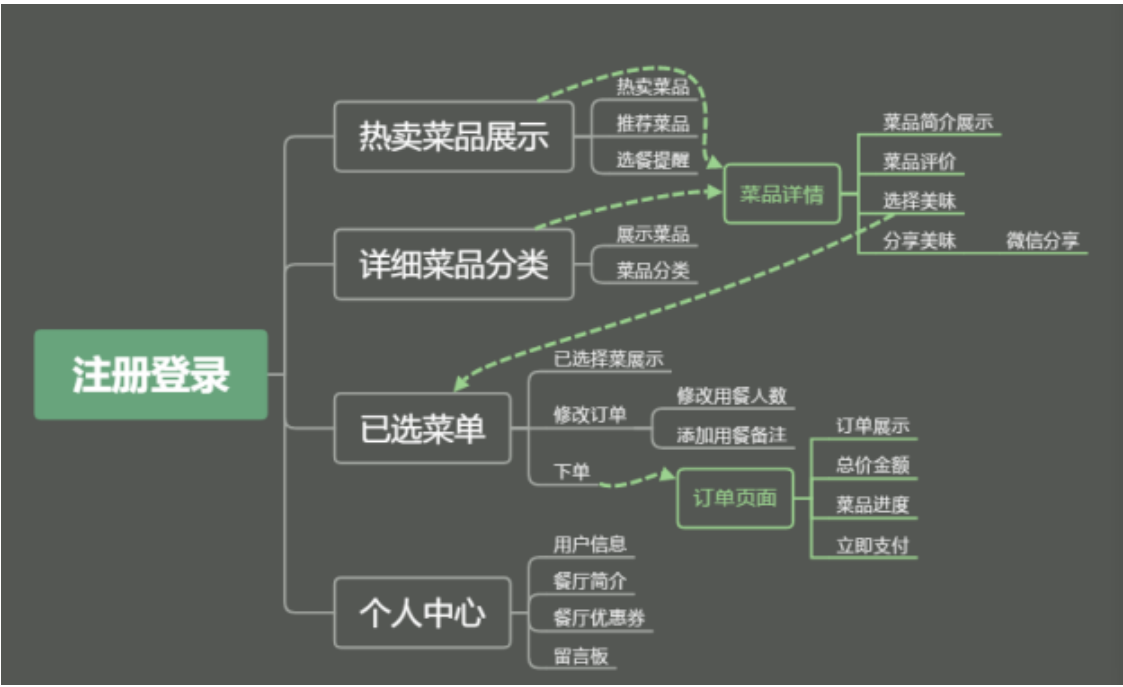


图3.1 功能需求分析思维导图

3.2 功能需求分析

根据在网上进行点调研另外又跟一些餐厅的负责人沟通后,最终确定了该点餐系统的功能模块。主要的功能模块：会员个人中心,菜品展示至菜品详情，详情页选择菜品至待购菜车，订单展示与提交支付，四个大的功能模块。首先用到了二维码扫码进入该点餐系统，进入主系统后对这个四个关键模块的详细功能分别展开详细的分析。根据分析就餐顾客的用例图如图3.2所示。

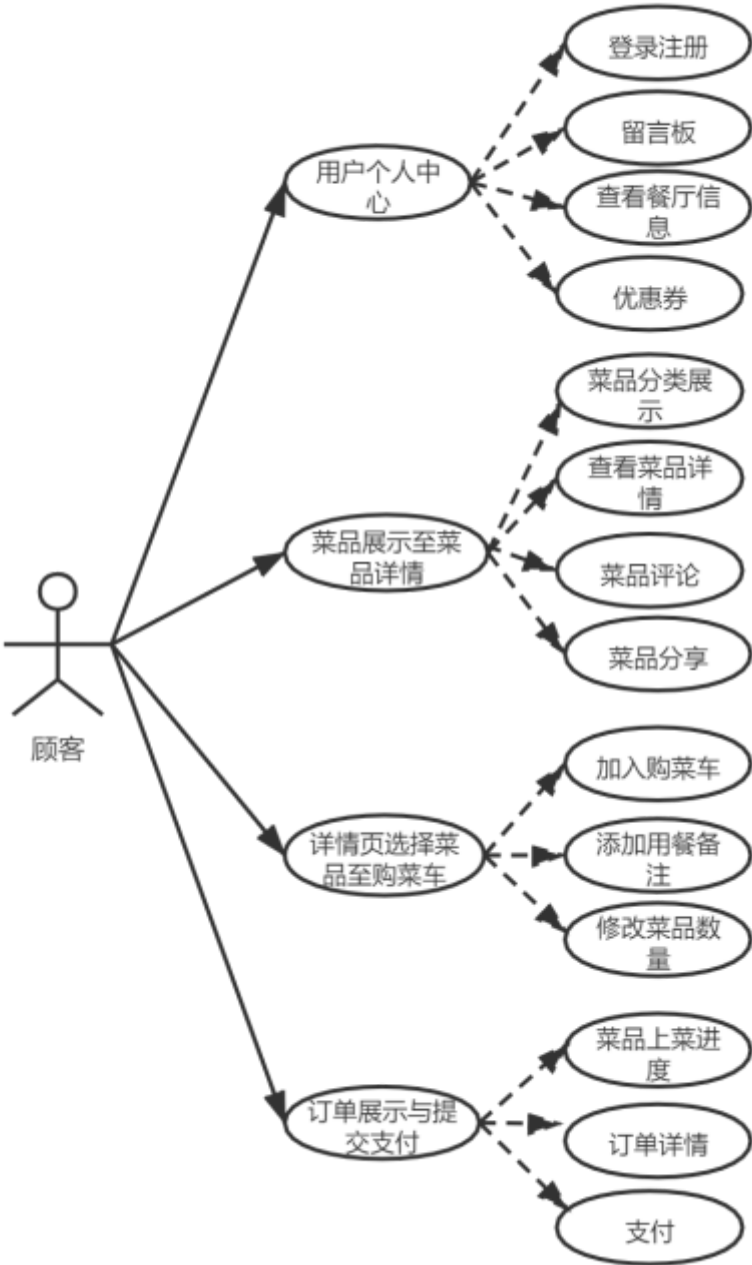


图3.2 就餐顾客的用例图

3.2.1 会员个人中心

会员中心，进入个人中心时先要判断用户是否登录注册，如果用户没有登录或者注册时将会展示登录注册按钮，此时可以经行登录注册。如果已经注册并且登录，将会成为该餐厅的会员，在会员中心页面可查看到自己的餐厅的优惠可用的优惠券，餐厅详细介绍，餐厅留言板等会员的详细功能。在进行登录的时候，在Web浏览器中就会对用户名的合法性进行校验，从而来减少对服务器的请求，以防用户输入不符合要求的用户名和密码还要对后台服务器做请求在，这种请求时无用的并且还浪费了整体项目的性能。在前端程序中进行正则校验通过后，向服务器发起登录或者注册请求，服务器接收到的用户名和密码后进行数据库的查询操作，当输入的用户名和密码通过数据库查询后都是正确的话就会返回登录成功以及已登录的 cookie。当输入的用户名在通过数据库操作后，数据库查询不到改用户名，则提示用户名不存在，需要用户重新检查是否是自己的用户名输入有误还是自己没有注册用户。

3.2.2 菜品展示至菜品详情

菜品展示对于一个点餐系统是至关重要的一步，在一个点餐系统中能够将餐厅的菜清晰主次分明的展示对于商家的销量和用户的体验感是非常重要的。系统中设计两个页面进行展示，热卖菜品展示为餐厅的重点菜品展示，分类菜品展示，是将不类别的菜品分类展示，便于用户的筛选。

热卖菜品展示，用户通过扫描二维码进入到系统中展示到菜单栏的页面，看到展示餐厅的首页的推荐菜品。页面最上方是店家对顾客点餐的提醒备注。首页的火爆商品会通过常见的轮播图展示，阶梯式列表展示，主要就是能够将餐厅里的热卖展示出来。

详细菜品分类展示，菜单栏的菜品将进行分类，把菜品分为饭前水果、热门美食、清单美食、时尚小炒、饮品类等，把相同类型的菜品归类在一起便于用户快速的定位挑选。进去大类分类后，切换成小类的详细分类，便于更有利的挑选。在点击大类，会根据大类的数据库中的ID值对小类进行请求，这就形成了点击大类分类后与小类的联动。通过大类和小类的筛选后展示对应分类后的菜品列表，选择点击菜品图片会展示菜品的详细页面，展示当前菜品的详细图片与描述以及该菜品的评论，显示添加购菜车的功能，能够选择添加购菜车的份数，同时在菜品的详细页面中会提供将菜品分享给微信朋友的功能。

3.2.3 详情页选择菜品至购菜车

当用户刚开始进入带点餐系统并没有点餐时，购菜车模块展示空数据页面，并提醒用户选菜流程。在菜品的详情展示页里点击添加至购菜车，需要先判断该菜品是否已经添加至购菜车，如果已经添加则提醒用户已经添加，如果没有添加，将该菜品成功添加。购菜车模块提供更改订单功能，点击修改按钮跳转到修改备注页面，能够选择用餐人数，添加用户的对菜品的口味的备注，当选择完成后，点击确认修改，跳转回到购菜车模块。

指 标

疑似剽窃文字表述

- 1. Vue.js 应用程序开发的状态管理模式。它采用的是集中式存储管理应用的所有组件的状态，同时以对应的规则确保状态以一种能够预测的方法发生变化。
- 2. 创造Node.js的目的是为了实现高性能Web服务器，首先看重的是事件机制和异步IO模型的优越性，而不是JavaScript。
- 3. JS没有自带IO功能，天生就用于处理浏览器中的DOM事件，并且拥有一大群程序员，因此就成为了天然的选择。NodeJS在服务端活跃起来，出现了大批基于NodeJS的Web服务。
- 4. 关系型数据库的实质：非关系型数据库产品是传统关系型数据库的功能阉割版，通过减少用不到或很少用的功能，来大幅度提高产品性能，实际开发中，很多业务需求，其实并不需要完整的关系型数据库功能，非关系型数据库的功能就足够使用了。
- 5. 系统需求分析 对系统的分析与设计是在系统实施之前最重要的部分。本章包含对系统需求的分析

2. 基于Web的无人值守点餐收银系统的设计与实现.doc_第2部分

总字数：11427

相似文献列表

去除本人已发表文献复制比：10.3%(1176) 文字复制比：10.3%(1176) 疑似剽窃观点 (0)

1	基于Web前端组件化的个人博客系统的设计与实现 曾广海(导师：卢力) - 《华中科技大学硕士论文》 - 2016-12-01	7.7% (880) 是否引证：是
2	基于VUE框架移动终端的个人博客设计与实现 金亚超 - 《大学生论文联合比对库》 - 2019-05-11	3.1% (358) 是否引证：否
3	基于Vue框架移动终端的个人博客设计与实现 金亚超 - 《大学生论文联合比对库》 - 2019-05-15	2.2% (246) 是否引证：否
4	金亚超_15031010101_基于vue框架移动终端的个人博客设计与实现 金亚超 - 《大学生论文联合比对库》 - 2019-05-27	2.2% (246) 是否引证：否
5	金亚超_15031010101_基于vue框架移动终端的个人博客设计与实现 金亚超 - 《大学生论文联合比对库》 - 2019-06-05	2.2% (246) 是否引证：否
6	143526_李贵岭_基于Koa框架的货车调度管理系统的设计与实现 李贵岭 - 《高职高专院校联合比对库》 - 2019-05-21	1.3% (149) 是否引证：否
7	基于Spring Boot实现微信端点餐及管理功能 张良宇 - 《大学生论文联合比对库》 - 2019-06-05	0.7% (82) 是否引证：否
8	基于Spring Boot实现微信端点餐及管理功能 张良宇 - 《大学生论文联合比对库》 - 2019-06-12	0.7% (82) 是否引证：否
9	z3-50704070435533824_研究生毕业论文V3	0.5% (52) 是否引证：否

研究生毕业论文V - 《大学生论文联合比对库》 - 2016-04-19	
10 基于流量分析的BGP报文采集与重构系统实现	0.3% (30)
陈南男(导师：马严) - 《北京邮电大学硕士论文》 - 2019-05-18	是否引证：否

原文内容

在购菜车模块提供清空购物车功能，当用户对所有选择的菜品不太满意的时候，就可以直接一键清空购物车。在购菜车页面同时展示用户当前所在的座位号，该座位号在提交订单时会跟菜品同时传递到订单页面。

在购菜车页面最重要的就是对已选菜品的修改和备注，所以对已选的菜品的清晰的展示是非常重要的，每道已选择的菜品将以列表的形式展示，在每个列表中提供增加该菜的分数，并通过计算展示该菜品的总价。在页面中也实时监控所有选择的菜品的总价，当所有的都确认之后就可以点击下单了，点击下单进入到订单页面，订单页面显示点餐后的详细订单，当前的桌号，选择的菜品，合计消费额。

3.2.4 订单展示与提交支付

在购菜车页面点击下单就会自动跳转到该订单页面，在订单查看页面主要分为菜品上菜进度展示、确认选择的菜品展示、菜品消费总价展示、支付入口，四个需求，在订单页面，用户可以实时查看自己的菜品的进度如何，解决用户在等待上菜的焦虑，确认选择的菜品展示是将自己选择的好的菜名称，数量，就餐人数，就餐备注通过列表展示出来，同时展示用户的消费总价，点击支付按钮便可以直接付款。

4 系统的总体设计

本章主要是对第三章所做的功能需求分析做系统的总体设计，根据上述的需求的分析，本系统将采用Web前端工程化来搭建项目结构，总体采用JavaScript语言设计，前端方面采用的是Vue.js框架，后端采JavaScript的服务端开发环境Node.js的Koa框架，以及MongoDB数据库。

4.1 系统总体架构设计

在体系结构方面，系统采用 B/S架构，B/S架构是一种常见的实现方式，它是指浏览器/服务器的机构，他的核心是Web浏览器的向Web服务器发送数据，通过HTTP请求后服务器向浏览器发送相应的资源，浏览拿到相应的数据资源后通过浏览器引擎解析渲染到页面上。系统总体架构设计图如图 4.1 所示。





图4.1 系统总体架构设计图

4.2 系统前端设计

本系统使用Vue.js来搭建一个大型的单页面Web应用，通过Vue.js提供的路由机制对后端接口请求获取到服务端的数据，把数据渲染到页面上，后端的数据发生改变对应的页面展示出来的内容随之发生改变，很好的体现了MVVM的框架的原理。页面之间的跳转并没有来回的切换URL来实现，实际也是通过组件之间通过路由的跳转进行的。这样能够保证页面的刷新也能够提高数据的返回速度让返回的数据能够很方便的解析，使用Vue.js框架让做到了前后端的真的分离开发。本系统使用组件化的思想，每一个页面都是由不同功能模块的组件构成,每个页面也能算一个单一的组件,组件的样式通过CSS渲染,通过ajax实现前端的接口请求，让顾客的每一次操作都能够及时得到对应的响应。

使用Node.js的Koa2实现系统的后端接口设计，他能够有效的支持JavaScript在服务端运行，同时提供了有着许多强大特性的Koa框架，能够方便快捷快速的创建更好用的API接口。

采用前端工程化的技术搭建项目结构，主要的项目结构设计如下：

- (1) 配置package.json 文件，对项目中添加的JavaScript的所有依赖和插件的信息。
- (2) config文件夹，主要是项目的相关的开发环境的配置npm包管理相关代码，主要是开发环境，线上环境执行的配置。
- (3) 配置static文件夹，是配置项目的静态文件包括一些静态的图片，json数据等不需要从从后端获取的数据。
- (4) 配置src文件夹，这个是项目的源码文件夹，里面包括components编写组件的文件夹，存放资源的assets资源目录，配置路由的router文件夹，App.vue的页面入口文件和main.js程序入口文件。
- (5) 项目依赖包node_modules文件夹。
- (6) .babelrc是babel转es6代码的配置文件。
- (7) webpack.config.js 中配置了代码分割、抽离css代码、分割业务代码和框架代码、生成index.html入口文件，同时文件里的配置是基于webpack3.x的webpack不同版本之间总是存在各种的差异。

4.3 系统数据库设计

本系统主要用到了火爆菜品信息，分类菜品，信息用户信息，订单信息，餐厅信息，留言信息五个信息模块，根据这些信息模块可以设计系统的数据库表，在项目设计实现中，对数据库表的设计是非常重要的，关系到整个项

目的整体新能以及项目的架构方向的问题，对这五个信息模块设计数据库表要考虑信息模块是为解决的具体需求。

根据系统的数据库的设计需要建立数据库表，每个表中的id都是由数据库自动生成的，具体的数据库表说明如表4.1所示。

数据表	中文名	说明
User	用户	存储注册的用户信息,包括用户名,密码,注册时间,登录时间等字段
Goods	菜品详情	存储菜品详情的数据
Categories	菜品大类	存储菜品分类中大类的菜品数据
CategorySubes	菜品子类	存储菜品分类中子类的菜品数据
RestaurantInfos	餐厅信息	存储餐厅信息
Oders	订单	存储用户点餐那后的订单信息

数据表中中文说明

User 用户存储注册的用户信息,包括用户名,密码,注册时间,登录时间等字段

Goods 菜品详情存储菜品详情的数据

Categories 菜品大类存储菜品分类中大类的菜品数据

CategorySubes 菜品子类存储菜品分类中子类的菜品数据

RestaurantInfos 餐厅信息存储餐厅信息

Oders 订单存储用户点餐那后的订单信息

表4.1 数据库表设计说明

项目中的每一个数据库表建立设计的详细分析如下：

(1) User表

用户的User表在后端的schema 文件夹下的 user.js 文件中进行字段配置，在配置中的字段包括了用户的全部信息，详细字段及说明如表4.2所示。

表4.2 User表

字段名	数据类型	说明
id	String	用户id,由MongoDB自动创建
userName	String	用户名
password	String	用户密码
createAt	Date	创建时间
lastLoginAt	Date	最后一次登录时间

字段名数据类型说明

id String 用户id,由MongoDB自动创建

userName String 用户名

password String 用户密码

createAt Date 创建时间

lastLoginAt Date 最后一次登录时间

(2) Categories表

在菜品分类的页面中大类的数据库表通过schema配置大类全部的字段信息。详细的字段字段配置如表4.3所示。

表4.3 Categories表

字段名	数据类型	说明
ID	String	大类分类种类ID
MALL_CATEGORY_NAME	String	大类分类种类名称
IMAGE	String	大类分类种类图片
TYPE	Number	大类分类类型字段
SORT	Number	大类排序字段
COMMENTS	String	评论字段

字段名数据类型说明

ID String 大类分类种类ID

MALL_CATEGORY_NAME String 大类分类种类名称
IMAGE String 大类分类种类图片
TYPE Number 大类分类类型字段
SORT Number 大类排序字段
COMMENTS String 评论字段

(3) CategorySubes表

菜品分类中的菜品子类数据库表，通过schema配置子类的全部的字段信息。详细的字段字段配置如表4.4所示。

表4.4 CategorySubes表

字段名	数据类型	说明
ID	String	子类ID
MALL_CATEGORY_ID	String	对应大类的ID
MALL_SUB_NAME	String	子类分类的种类名称
COMMENTS	String	评论
SORT	Number	子类排序

字段名数据类型说明

ID String 子类ID
MALL_CATEGORY_ID String 对应大类的ID
MALL_SUB_NAME String 子类分类的种类名称
COMMENTS String 评论
SORT Number 子类排序

(4) Goods表

菜品详细信息通过schema配置菜品信息全部的字段信息。详细的字段字段配置如表4.5所示。

表4.5 Goods表

字段名	数据类型	说明
Id	String	菜品id
GOODS_SERIAL_NUMBER	String	菜品序列号
GOOD_TYPE	Number	菜品类型
STATE	Number	状态
NAME	String	菜品名称
ORI_PRICE	Number	菜品原价
PRESENT_PRICE	Number	菜品现价
DETAIL	String	详情图
COMMENTS	String	评论

字段名数据类型说明

Id String 菜品id
GOODS_SERIAL_NUMBER String 菜品序列号
GOOD_TYPE Number 菜品类型
STATE Number 状态
NAME String 菜品名称
ORI_PRICE Number 菜品原价
PRESENT_PRICE Number 菜品现价
DETAIL String 详情图
COMMENTS String 评论

(5) RestaurantInfos表

餐厅信息通过schema配置餐厅的全部的字段信息。详细的字段字段配置如表4.6所示。

表4.6 RestaurantInfos表

字段名	数据类型	说明

ID	String	餐厅ID
NAME	String	餐厅名称

字段名数据类型说明

ID String 餐厅ID

NAME String 餐厅名称

续表4.6

字段名	数据类型	说明
TIME	Date	餐厅营业时间
TYPE	String	餐厅主要菜品类型
SERVICE	String	餐厅特色服务
ADDRESS	String	餐厅地址
PHONE_NUM	String	餐厅的联系电话号码

字段名数据类型说明

TIME Date 餐厅营业时间

TYPE String 餐厅主要菜品类型

SERVICE String 餐厅特色服务

ADDRESS String 餐厅地址

PHONE_NUM String 餐厅的联系电话号码

(6) Oders表

订单信息通过schema配置订单全部的字段信息。详细的字段字段配置如表4.7所示。

表4.7 Oders表

字段名	数据类型	说明
Id	String	订单号
TABLE_NUM	String	座位号
GOODS_ID	String	菜品id
NAME	String	菜品名称
PRICE	Number	总价价格
COUNT	Number	用户选择的菜品选择的数量
P_NUM	Number	用户选择的用餐人数
P_MARK	String	用户添加用餐备注
CREATE_TIME	Date	点餐的创建时间以及下单的时间

字段名数据类型说明

Id String 订单号

TABLE_NUM String 座位号

GOODS_ID String 菜品id

NAME String 菜品名称

PRICE Number 总价价格

COUNT Number 用户选择的菜品选择的数量

P_NUM Number 用户选择的用餐人数

P_MARK String 用户添加用餐备注

CREATE_TIME Date 点餐的创建时间以及下单的时间

根据系统的总体需求以及系统的设计方式,本系数据库设计的E-R图如图4.2所示.

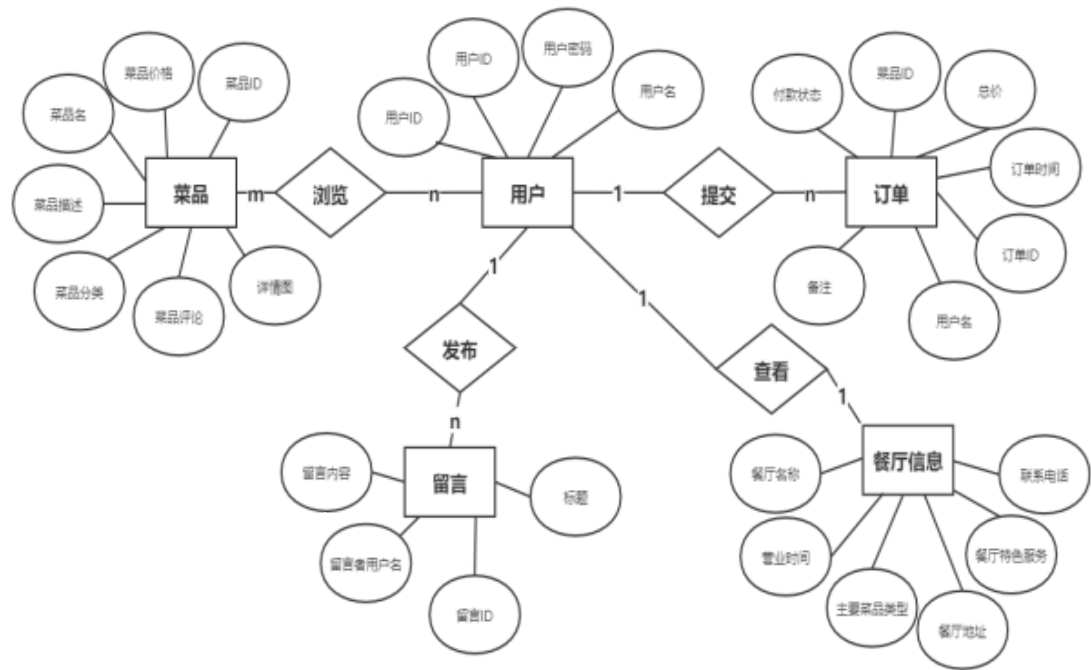


图4.2 数据库设计的E-R图

4.4 系统后端设计

后端采用Koa和MongoDB数据库实现，采用的设计项目结构设计如下：

- (1) 配置后端开发的包管理和插件管理的package.json文件。
- (2) 通过包管理配置文件初始化出mode_modules项目依赖包文件夹。
- (3) 通过schema文件夹定义系统中数据库中数据表的设计。
- (4) 配置index.js文件配置后端的服务的入口文件，并且定于服务的端口号。
- (5) 通过建立appApi文件夹，将所有要用到的服务端接口写入该文件夹的文件中。
- (6) 通过配置database文件夹中的init.js文件，配置要连接的数据库。
- (7) 将要写入数据库的json数据配置到data_json文件夹中。

Koa是现在最流行的基于Node.js平台的web开发框架，它很小，但扩展性很强。Koa给人一种干净利落的感觉，体积小、编程方式干净。通过利用 async 函数，Koa 帮你丢弃回调函数，并有力地增强错误处理[15]。Koa 并没有捆绑任何中间件，而是提供了一套优雅的方法，帮助您快速而愉快地编写服务端应用程序。之所以说Koa作用更纯粹，是因为Koa本身只提供了有限的最基本的功能，一切需要的额外功能都是通过中间件实现，比如路由管理，log日志，错误处理等等。所谓中间件，就像中间人一样，所有与客户端之间的通信都要经过它们，它们会对会话的输入和输出做具体的处理[16]。

使用 Koa 创建一个服务器非常简单，搭建服务设计服务监听端口3000的主要代码如图4.3所示。

```
app.use(async ctx=>{
  ctx.body = '<h1>hello Koa2</h1>'
})
app.listen(3000,()=>{
  console.log('[Server] starting at port 3000')
})
```

图4.3 搭建服务设计端口代码图

启动后实现数据库的连接与数据库的连接如图4.4所示。

```
exports.connect = ()=>{
  //连接数据库
  mongoose.connect(db)
  let maxConnectTimes = 0
  return new Promise((resolve,reject)=>{
    //把所有连接放到这里
    //增加数据库监听事件
    mongoose.connection.on('disconnected',()=>{
      console.log('*****数据库断开*****')
      if(maxConnectTimes<3){
        maxConnectTimes++
        mongoose.connect(db)
      }else{
        reject()
        throw new Error('数据库出现问题，程序无法搞定，请人为修理.....')
      }
    })
    mongoose.connection.on('error',err=>{
      console.log('*****数据库错误*****')
      if(maxConnectTimes<3){
        maxConnectTimes++
        mongoose.connect(db)
      }else{
        reject(err)
        throw new Error('数据库出现问题，程序无法搞定，请人为修理.....')
      }
    })
    //链接打开的时
    mongoose.connection.once('open',()=>{
      console.log('MongoDB connected successfully')
      resolve()
    })
  })
}
```

图4.4 服务连接数据库代码图

5 系统功能实现

本章将根据已经做过的需求分析与系统设计分析对系统进行实现与测试。首先要对本系统所需要的开发环境确定，然后分别针对本系统的菜单、购物车、订单三个功能模块的设计进行具体的实现以及描述，并对模块中的关键的实现技术点和关键函数做分析。

5.1 功能模块概述

主要的功能模块：用户会员个人中心管理,菜品展示至菜品详情，详情页选择菜品至待购购物车，订单展示与提交支付，四个大的功能模块。首先用到了二维码扫码进入该点餐系统，进入主系统。对这个四个关键模块的功能的程序流程设计分析，以及每个模块的前端后端的实现的思路的方法，并展示关键性的代码。

5.2 功能模块的实现

5.2.1 用户会员个人中心管理

基于组件化开发，要将用户的登录的注册独立写成单独的组件，用户的注册与登录所用的视图是一样的模型，视图方面只需要公用组件，最重要的是登录注册的逻辑处理。在该项目中，用户是可以不用选择登录注册的，但是用户只有选择了登录注册，就能成为餐厅的会员，登录注册流程图如图5.1所示。

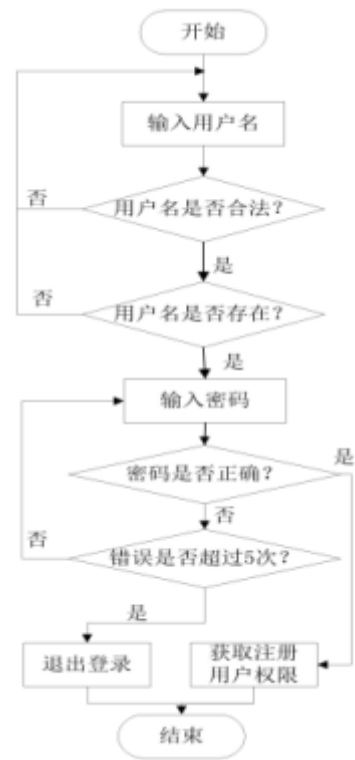


图5.1 登录注册流程图

实现服务搭建与数据库连接后就可以在后端工程项目中建立user.js文件实现登录注册的接口设计，实现登录接口主要是对前端输入的用户和密码与数据库存入的用户名与密码匹配。注册接口只需要将用户输入的格式正确的用户名和密码存入数据库，实现注册的接口设计主要代码如图5.2所示，实现登录的后端接口关键代码如图5.3所示。

```
// 后端注册接口
router.post('/register',async(ctx)=>{
  // 引入数据库的User表的model
  const User = mongoose.model('User')
  let newUser = new User(ctx.request.body)
  await newUser.save().then(()=>{
    ctx.body={
      code:200,
      message:'注册成功'
    }
  }).catch(error=>{
    ctx.body={
      code:500,
      message:error
    }
  })
})
})
```

图5.2注册接口设计


```
// 后端登录接口
router.post('/login', async(ctx) => {
  let loginUser = ctx.request.body
  console.log(loginUser)
  let userName = loginUser.userName
  let password = loginUser.password
  // 引入数据库的User表的model
  const User = mongoose.model('User')
  await User.findOne({userName: userName}).exec().then(async(result) => {
    console.log(result)
    if(result) {
      let newUser = new User()
      await newUser.comparePassword(password, result.password)
        .then(isMatch => {
          ctx.body = {code: 200, message: isMatch}
        })
        .catch(error => {
          console.log(error)
          ctx.body = {code: 500, message: error}
        })
    } else {
      ctx.body = {code: 200, message: '用户名不存在, 请先注册'}
    }
  }).catch(error => {
    console.log(error)
    ctx.body = {code: 500, message: error}
  })
})
})
```

图5.3 登录接口实现

在前端设计用户名和密码的时候，在输入完成后，前端进行正则的检测匹配到用户名的格式为字母和数字组成，长度在2-7之间，密码格式由字母和数字组成，长度在4-10之间。通过axiosLoginUser()方法处理逻辑。在axiosLoginUser()要用到axios方法可以做到用同步的写法进行异步请求，前端通过正则等一系列的逻辑判断后，再请求后端接口，请求接口逻辑处理如图5.4所示。

```
axiosLoginUser() {  
  console.log(666);  
  //先把按钮进行loading状态, 防止重复提交  
  this.openLoading = true;  
  axios({  
    url: url.login,  
    method: "post",  
    data: {  
      userName: this.username,  
      password: this.password,  
    },  
  })  
  .then((response) => {  
    console.log(response);  
    if (response.data.code == 200 && response.data.message) {  
      new Promise((resolve) => {  
        // 本地存储保存用户登录状态  
        let userInfo = {  
          userName: this.userName  
        }  
        localStorage.userInfo = JSON.stringify(userInfo)  
        console.log(resolve);  
        setTimeout(() => {  
          resolve();  
        }, 500);  
      })  
      .then(() => {  
        Toast.success("登录成功");  
        this.$router.push("/shoppingMall");  
      })  
      .catch(() => {  
        Toast.fail("登录失败");  
        this.openLoading = false;  
      });  
    }  
  });  
}
```

图5.4 登录接口实现代码图

5.2.2 菜品展示至菜品详情

菜品展示至菜品详情是系统设计的主要模块之一，用户可以通过扫描桌面的二维码，进入系统码店内点餐功能选择喜爱的菜品，在线下单付款，享受美食。所以为了精简用户的操作流程，把菜单预览点餐页面设计到首页是正确的选择。从点菜到下单支付的程序实现流程图如图5.5所示。

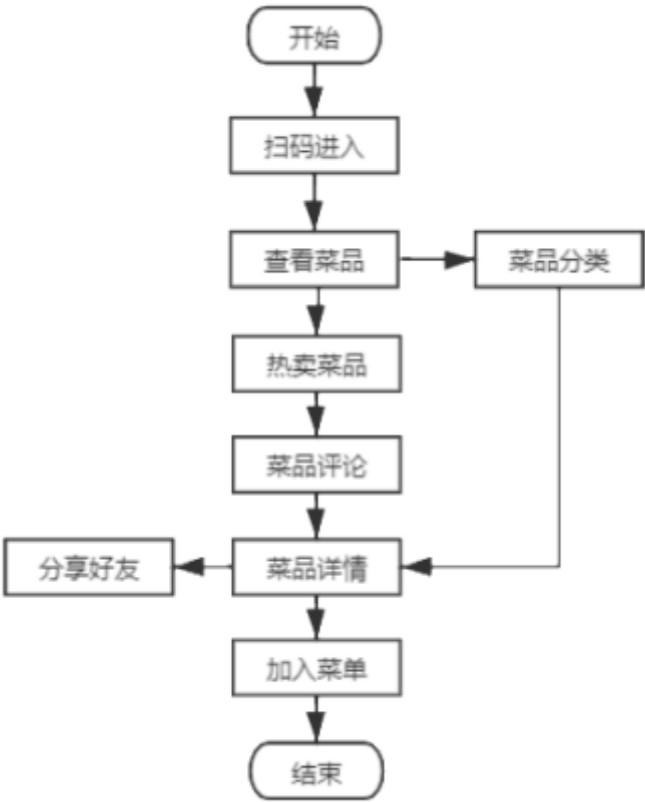


图5.5 菜品展示至菜品详情程序流程图

首先要对首页也就是菜单也进行布局设计在components存放组件的文件夹中写入Home组件，所有的菜单的菜品信息都是有请求后端接口获取到的，所以要使用Vue.js提供的v-for指令循环遍历数据，同时在使用v-for指令的时候更新已渲染的元素列表时,默认用就地复用策略，当数据修改的时候,会根据key值去判断这个值是否已经修改,如果是修改了,就要重新渲染这一项,不然就会复用之前的元素。最后把数据循环遍历后根据html结构和写好的css样式渲染到页面上。给菜单的每个菜品配置路由跳转，并传递参数：router-link:to="/pcontent?id="+itemInfo._id"。使用Vue.js提供的vue-resource插件完成请求页面数据的方法requestData(), 如图5.6所示。

```
// 请求首页页面数据
requestData(){
  var honeRequestApi = this.api+'api/productlist';
  this.$http.jsonp(honeRequestApi).then((response)=>{
    // 第一个response是成功之后的回调
    this.list = response.body.result;
  },(error)=>{
    // 第二个是失败之后的回调
    console.log("首页数据加载失败"+error);
  });
}
```

图5.6请求菜品页面数据代码

菜品的分类展示，通过大类分类的小类分类的两层筛选后展示出对应的菜品列表。通过上文的数据库设计，明确了导航的数据设计，要在后端设计获取大类列表和获取小类分类列表两个接口。在service文件夹中的goods.js文件中设计相关的接口。这两个接口的设计需要定义GET请求，通过async的语法，和引用的mongoose查询对应的表，.find().exec()实现对数据数据查询，接口设计代码如图5.7所示。

```

/** 读取大类数据的接口 */
router.get('/getCategoryList',async(ctx)=>{
  try{
    const Category = mongoose.model('Category')
    let result = await Category.find().exec()
    ctx.body={code:200,message:result}
  }catch(error){
    ctx.body={code:500,message:error}
  }
})
/** 读取小类的数据 */
router.post('/getCategorySubList',async(ctx)=>{
  try{
    let categoryId = ctx.request.body.categoryId
    //let categoryId=1
    const CategorySub = mongoose.model('CategorySub')
    let result = await CategorySub.find({MALL_CATEGORY_ID:categoryId}).exec()
    ctx.body={code:200,message:result}
  }catch(error){
    ctx.body={code:500,message:error}
  }
})

```

如图5.7 分类展示菜品接口设计关键代码

菜品的展示分为餐厅热卖商品和商品分类展示，需要通过后端操作数据库，实现后端的接口设计，主要是获取菜品详情数据，通过菜品的id在数据库表中查找，后端接口设计代码如图5.8所示。

```

/** 获取商品详情信息的接口
router.post('/getDetailGoodsInfo',async(ctx)=>{
  try{
    let goodsId = ctx.request.body.goodsId
    const Goods = mongoose.model('Goods')
    console.log(goodsId)
    let result= await Goods.findOne({ID:goodsId}).exec()
    ctx.body={code:200,message:result}
  }catch(error){
    ctx.body={code:500,message:error}
  }
})

```

图5.8 获取菜品详情接口设计关键代码

使用微信平台提供接口，实现对菜品分享给好友的功能，微信给开发者提供wx.onMenuShareAppMessage()接口实现分享给微信朋友的功能。该接口传递传递一个title确定分享的标题，dese确定分享的描述，link确定分享的链接，imgUrl分享图标，success用户确认分享的回调函数，cancel用户取消分享后的回调函数。在文件中建立wxconfig.js文件进行配置，配置关键代码如图5.9所示。

```
// 分享给微信好友
wx.onMenuShareAppMessage({
  title: obj.title, // 分享标题
  desc: obj.desc, // 分享描述
  link: obj.link, // 分享链接
  imgUrl: obj.imgUrl ? obj.imgUrl : "",
  success: () => {
    typeof callback === "function" && callback();
  },
  cancel: () => {
    // 用户取消分享后执行的回调函数
  }
});
```

图5.9 菜品分享微信接口配置关键代码

5.2.3 详情页选择菜品至购物车

点击进入购物车页面，页面主要式展示用餐人数和备注并提供修改功能，展示选择的菜品的数以及菜品的合计的总钱数，展示选择的菜品列表并提供修改菜品数量的功能，在项目中增加Cart组件。在备注栏的使用Vue.js提供的v-if指令操作和三元运算符来控制显示备注的内容。他实际上就是根据判断是否填写备注的条件的true和false触发是都渲染DOM元素。但是v-if绝对是更消耗性能的，因为v-if在显示隐藏过程中有DOM的添加和删除，所以在频繁的切换不能够使用这个指令，频繁的切换可以用到Vue.js提供的v-show指令来切换切换元素的 display 属性。

原本以有添加至购物车的菜品都是通过localStorage进行web存储的数据，原本的菜品添加也是通过web存储的。菜单的功能模块的程序流程图如图5.10所示。

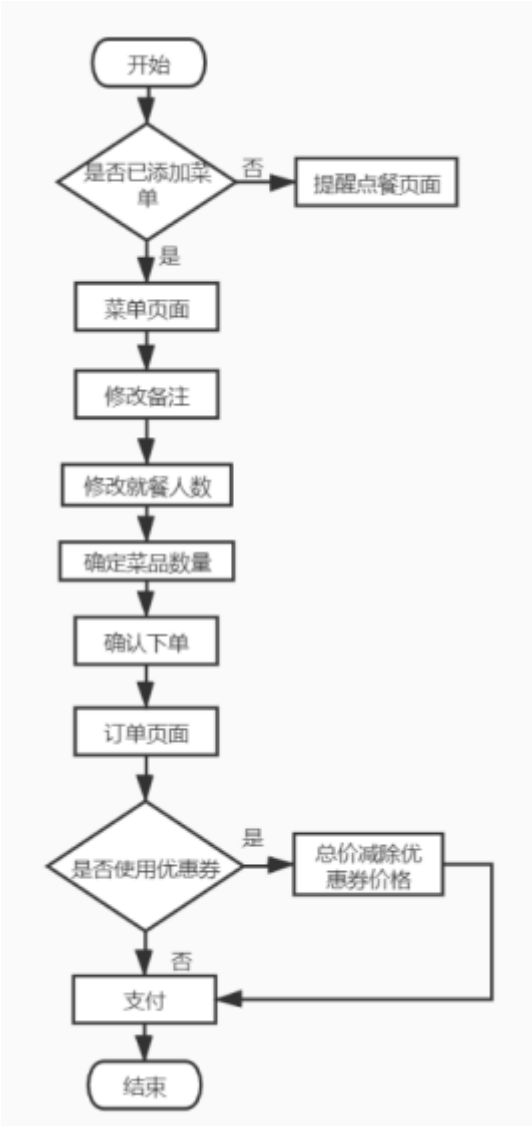


图5.10 菜单程序流程图

用户通过扫描桌子的二维码获取到座号获取到桌子id，由于考虑到进入首页菜单页面可能是在其他页面通过底部导航和其他操作跳转至此，随意在首页就需要封装一个获取购物车菜品数量的方法，该方法可以放在Vue.js生命周期函数中执行。获取购物车的数量getCartCount()代码如图5.11所示。

```
getCartCount(){/*获取购物车的数量*/
    //桌子id 桌子号: 是扫描二维码从url 获取的
    var uid=storage.get('roomid') || 1;
    var api=this.api+'api/cartCount?uid='+uid;
    this.$http.get(api).then((response)=>{
        console.log(response.body.result);
        this.cartNum=response.body.result;
    },(err)=>{
        console.log(err);
    })
}
```

图5.11 获取购物车数量方法代码

获取的购物车数量通过在接收服务器广播过来的addCart ()方法中重新调用，该实现方式使用socket通信实现的socekt通信重新调用getCartCount()方法。

目前将所要执行的事件方法封装好之后，击导航改变页面位置的asideDomInit()方法，请求首页页面数据的requestData()方法和获取购物车商品数量的getCartCount()方法，是在首页中组件挂载完成过后就立即执行的方法，要实现Vue.js提供的生命周期函数mounted()实现，在mounted()钩子函数中通过this关键字调用组件中封装的方法。

页面的底部将展示要选择菜品的数量，以及添加至购菜车的按钮，需要对更改菜品数量做事件处理，所以在做底部的布局的时候要将元素绑定点击事件，同时使用Vue.js通过的v-model指令对input的数据跟后台提供的数据进行数据双向绑定，v-model会忽略所有表单元素的 value、checked、selected attribute 的初始值而总是将Vue 实例的数据作为数据来源。

对绑定的事件做逻辑处理，选择的数量是通过数据双向绑定的原理，直接给绑定的事件中更改数据模型的数据。点击购菜车跳转路由，并且调用后端接口，加入购菜车成功后给服务器发消息事件逻辑处理代码如图5.12所示。

```
// 购物车数量增加
addNum(){
  ++this.numValue;
},
// 购物车数量减少
subNum(){
  if(this.numValue > 1){
    --this.numValue;
  }
},
// 加入购物车
addCart(){
  // 获取数据，加入购物车
  var cartPath = this.api + 'api/addcart';
  var objParams = {
    uid:Storage.get('roomid'),           // 桌号（一个桌子有多位用户，他们的桌号一样） 必传（001 002）
    title:this.list.title,               // 菜品名称必传
    product_id:this.list._id,            // 菜品id 必传
    img_url:this.api+this.list.img_url,  // 菜品图片必传
    price:this.list.price,               // list菜品价格必传
    num:this.numValue,                   // 菜品数量必传
    open_id:'666666'                     // 选填
  }
  this.$http.post(cartPath,objParams).then((response)=>{
    console.log(response);
    if(response.body.success){
      // 加入购物车成功之后给服务器发送消息
      this.$socket.emit('addcart','addcart');
      // 编程式导航返回首页
      this.$router.push({path:'home'});
    }
  }),(error)=>{
  })
}
```

图5.12 事件逻辑处理代码如图

点击修改图标会根据配置的好的路由跳转到修改备注页面，通过新建的EditUserInfo组件，在页面中可以选择就餐的人数，填写就餐的备注，给选择的人数添加DOM事件操作，具体代码实现如图5.13。

```

// 点击更改按钮事件
addChangeEvent() {
    // 保存this
    var that = this ;
    // 选择人数的dom操作
    var aLiDoms = document.querySelectorAll('.user_list li');
    // console.log(aLiDoms)
    for(var i=0 ; i<aLiDoms.length ; i++){
        aLiDoms[i].onclick = function(){
            // 去掉所有li的class, 让被点击的li添加active样式
            for(var j=0 ; j<aLiDoms.length;j++){
                aLiDoms[j].className = '';
                this.className = "active"; //this为li
                that.p_num = this.querySelector('span').innerHTML.trim(); //trim()为原生js里的去除空格的方法
                console.log(that.p_num)
            }
        }
    }
    // 选择口味的dom操作
    var aLi = document.querySelectorAll(".mark_list li");
    // 给节点添加点击事件
    for(var k = 0; k<aLi.length;k++){
        aLi[k].onclick = function(){
            // 先清除所有li的class, 给点击的li添加active样式
            for(var l=0;l<aLi.length;l++){
                aLi[l].className = '';
                this.className = 'active';
                // console.log(that.p_mark);
            }
            that.p_mark = `${that.p_mark} ${this.querySelector('span').innerText}` ;
        }
    }
}

```

图5.13 修改就餐人数与备注方法代码

购物车页面中的展示点餐的菜总数量和合计总价与选择出的菜品展示列表。菜品展示的列表跟可以操作选择数量，更改选择的数量跟页面中动态显示的菜品数域菜品总钱数是动态的数据绑定，能够将

三个展示的数据动态跟新。要实现这个功能需要在Cart组件中循环遍历添加到购物车的菜品，并为每个循环出来的菜品list绑定增加数量和减少数量的事件处理操作。

处理计算购物车的总钱数和购物车商品总数量的方法需要进入该页面的时候调用，还有需要在更改菜品的数量的时候触发，首要是封装该getTotalResult()方法。计算总价格是用简单的四则运算实现，购物车商品总价=该菜品的总价*菜品的总数。当然这只是一道菜品的总价，选择多种菜品获取到的是cartList数据，所以把cartList数据利用循环遍历，把每种菜品总价累加的到总价。购物车的总数量只要在循环遍历的时候累加数量即可得到。封装好getTotalResult()方法如图5.14所示。

```

// 获取购物车商品总数量及总价
getTotalResult(){
    var allPrice = 0 ;
    var allNum = 0 ;
    for(var i=0 ; i<this.cartList.length;i++){
        allPrice += parseFloat(this.cartList[i].price*this.cartList[i].num) ;
        allNum += this.cartList[i].num;
    }
    this.allPrice = allPrice ;
    this.totalNum = allNum ;
}

```

图5.14 获取购物车商品总数及总价代码

5.2.4 订单提交支付功能模块

在购物车模块将选择菜品数量、就餐人数和就餐备注更改确定完成后，点击下单按钮请求后端接口，将数据传到后端，前端给后端传递参数的是通过将参数当作属性存入pramas对象，在根据post请求把数据传递到后端，并通过路由的配置跳转到订单页面。addOrder()方法封装代码如图5.15所示。

```
// 下单
addOrder(){
  var requirePath = this.api+'api/addOrder';
  console.log(requirePath)
  // var requirePath = `${this.api}api/getOrder`;
  var pramas = {
    uid : Storage.get('roomid') ,           // 桌号id    必传
    p_num : '6人', //this.orderInfo.p_num,    // 用餐人数 必传
    p_mark : '特辣', // this.orderInfo.p_num,    // 备注信息 必传
    order:JSON.stringify(this.cartList), // 菜品信息（数组，见菜品参数说明） 序列化的数组
    total_price : this.allPrice,           // 总价格
    total_num : this.totalNum              // 总数量
  }
  this.$http.post(requirePath,pramas).then((response)=>{
    console.log(response)
    // 如果请求数据成功则跳转订单页面
    if(response.body.success){
      this.$router.push({path:'order'});
    }else{
      alert('提交数据有误')
    }
  }),(error)=>{
    console.log(error );
  })
}
```

图5.16 获取购物车商品总数及总价代码

在购物车页面点击确认下单按钮后，通过addOrder()方法配置的路由跳转到订单展示页面。订单展示页面主要的功能是让顾客最后一次确认菜品数量以及总钱数，并提供支付操作入口。在项目工程中新建Order组件，请求Order详细数据请求代码如图5.17所示。

```
getOrder(){
  var uid=storage.get('roomid');
  var api=this.api+'api/getOrder?uid='+uid;
  this.$http.get(api).then((response)=>{
    console.log(response);
    this.list=response.body.result[0];
  }),(error)=>{
    console.log(error);
  })
},
```

图5.17 请求Order详细数据请求代码

点击微信支付请求支付宝和微信提供的支付接口，封装支付方式doPay()和doWeixinPay()，微信支付也是采用JSAPI接口的，在微信支付官方文档中，明确了用户通过消息或扫描二维码在微信内打开网页时，可以调用微信支付完成下单购买的流程。

首先商户最后请求拉起微信支付收银台的页面地址我们称之为“支付目录”，商户实际的支付目录必须和在微信支付商户平台设置的一致，否则会报错“当前页面的URL未注册。”登录微信支付商户平台（pay.weixin.qq.com）-->产品中心-->开发配置。提供在微信浏览器里面打开H5网页中执行JS调起支付。接口输入输出数据格式为JSON。主要配置的参数如下：timestamp：支付签名时间戳，注意微信js sdk中的所有使用timestamp字段均为小写。但最新版的支付后台生成签名使用的timeStamp字段名需大写其中的S字符；nonceStr：支付签名随机串，不长于32位；package：统一支付接口返回的prepay_id参数值；paySign：支付签名；success：支付成功回调；fail：支付失败回调；cancel：取消支付回调。详细接口配置代码如图5.18所示。

```
pay() {
  let params = {
    payAmount: this.totalMoney,
  }
  let promises = Http.post('/transaction/do', params)
  let that = this
  promises.then((response) => {
    if (response.status === 0) {
      let recordOrderGid = response.content.recordOrderGid
      try {
        this.wx.chooseWXPay({
          timestamp: response.content.timeStamp, // 支付签名时间戳，注意微信js
            名使用的timeStamp字段名需大写其中的s字符
          nonceStr: response.content.nonceStr, // 支付签名随机串，不长于 32 位
          package: 'prepay_id=' + response.content.prepayId, // 统一支付接口
          signType: response.content.signType, // 签名方式，默认为'SHA1'，
          paySign: response.content.sign, // 支付签名
          success: function () {
            that.doWxPayCallback(recordOrderGid)
          },
          fail: function () {
            that.doWxPayCallback(recordOrderGid)
          },
          cancel: function () {
            console.info('取消支付，如需支付请继续。')
          },
        })
      } catch (e) {
        console.info('订单支付异常，请稍后重试。')
      }
    } else {
      console.info(response.message)
    }
  })
}
```

图5.18 支付接口配置关键代码

6系统调试与测试

对系统的调试和测试是系统开发必不可少的一个步骤，本系统的调试主是自己多次操作后体验系统的流畅程度以及系统的设计的不合理之处，另外检查出系统是否存在bug。对于测试，本系统是个自动化工具搭建的项目，对项目的工程化经行了管理，所以测试使用到的自动化测试，对JavaScript代码的语法规范经行检测，保证代码的高质量以及程序逻辑的正常运行。本章展现分功能模块的测试用例与测试结果分析。

6.1 测试用例设计

主要通过检测运行测试系统的功能是否能够正常运行，并且能够完整实现业务逻辑，按照上文的[需求分析验证系统的功能将需求全部覆盖。本次的功能测试就是按照不同需求模块进行功能测试](#)，按照需求分析，需要进行功能测试的模块主要有用户会员个人中心管理，菜品展示至菜品详情，详情页选择菜品至待购车车，订单提交支付功能模块四个大的功能模块测试。

一个完整的[系统开发都需要先由开发人员对系统的各个需求功能模块进行分析，然后提交给专业的测试人员进行测试所有功能测试都是需要一份详细的测试文档展示测试的详细内容 and 结果。这样可以为系统的全面性提供依据，以方便系统今后的维护。](#)

[将软件需求设计文档中设计的功能视为预期结果，并根据需求进行测试。根据功能模块对用例进行测试，测试结果将成为本次测试的为实际结果，然后整理出测试文档。下表简要描述了功能测试的内容和结果。](#)

[系统的各大功能模块的测试的内容与测试结果说明及系统功能性测试用例表如表6.1所示。](#)

表6.1 系统功能性测试用例表

模块名称	测试内容	预期结果	实际结果	结论
用户会员	注册	首先进行输入的非空和字段长度的校验，校验成功后则发起注册请求，如若注	与预期	测试

个人中心		册成功则返回注册成功，失败则返回相应的提示信息	一致	通过
	登录	首先进行输入的非空和字段长度的校验，校验成功后则发起成功请求，如若成功则返回登录成功，失败则返回相应的提示信息	与预期一致	测试通过
	优惠券	用户登录成功领取一张可用优惠券	与预期一致	测试通过
	餐厅信息	用户登录成功展示餐厅详细信息，便于用户更加详细的了解餐厅信息	与预期一致	测试通过

模块名称测试内容预期结果实际结果结论

用户会员个人中心注册首先进行输入的非空和字段长度的校验，校验成功后则发起注册请求，如若注册成功则返回注册成功，失败则返回相应的提示信息与预期一致测试通过

登录首先进行输入的非空和字段长度的校验，校验成功后则发起成功请求，如若成功则返回登录成功，失败则返回相应的提示信息与预期一致测试通过

优惠券用户登录成功领取一张可用优惠券与预期一致测试通过

餐厅信息用户登录成功展示餐厅详细信息，便于用户更加详细的了解餐厅信息与预期一致测试通过

续表6.1

模块名称	测试内容	预期结果	实际结果	结论
	餐厅留言板	用户登录成功可以在餐厅留言板，留下自己的想说的话	与预期一致	测试通过
	订单查看	已登录用户可以在个人中心查看个人的点菜订单	与预期一致	测试通过
菜品展示至菜品详情	首页热卖菜品展示	首页展示用户点餐说明，通过轮播图和阶梯式列表将热卖菜品展示	与预期一致	测试通过
	菜品分类展示	点击大类分类展示对应的小类分类，再点击小类分类展示对应筛选出的菜品列表	与预期一致	测试通过
	点击显示菜品详情	点击菜品，展示对应菜品的详情图，价格，和菜品评论	与预期一致	测试通过
	分享给微信好友	点击分享给好友，能够成功分享	与预期一致	测试通过
详情页选择菜品至待购菜车	点击添加购菜车	需要先判断该菜品是否已经添加至购菜车，如果已经添加则提醒用户已经添加，如果没有添加，将该菜品成功添加	与预期一致	测试通过
	修改用户用餐备注	在购菜车页面点击修改备注，能够修改用户用餐人数与用餐的备注	与预期一致	测试通过
	修改菜品数量	更改菜品的数量，当数量变化时，对应的总价能实时动态变化。	与预期一致	测试通过
	清空购菜车	点击一键清空购菜车，成功清除购菜车里所有菜品	与预期一致	测试通过
订单提交支付功	确认下单	点击确认下单，提交订单到后台，跳转至订单页面	与预期一致	测试通过
	菜品进度	订单页面实时显示，已点菜品的上菜进度	与预期一致	测试通过
	订单信息展示	展示座位号，用餐备注，菜品总价	与预期一致	测试通过
	支付	点击支付，调起微信支付，并能成功支付	与预期一致	测试通过

模块名称测试内容预期结果实际结果结论

餐厅留言板用户登录成功可以在餐厅留言板，留下自己的想说的话与预期一致测试通过

订单查看已登录用户可以在个人中心查看个人的点菜订单与预期一致测试通过

菜品展示至菜品详情首页热卖菜品展示首页展示用户点餐说明，通过轮播图和阶梯式列表将热卖菜品展示与预期一致测试通过

菜品分类展示点击大类分类展示对应的小类分类，再点击小类分类展示对应筛选出的菜品列表与预期一致测试通过

点击显示菜品详情点击菜品，展示对应菜品的详情图，价格，和菜品评论与预期一致测试通过

分享给微信好友点击分享给好友，能够成功分享与预期一致测试通过

详情页选择菜品至待购菜车点击添加购菜车需要先判断该菜品是否已经添加至购菜车，如果已经添加则提醒用户已经添加，如果没有添加，将该菜品成功添加与预期一致测试通过

修改用户用餐备注在购菜车页面点击修改备注，能够修改用户用餐人数与用餐的备注与预期一致测试通过

修改菜品数量更够菜品的数量，当数量变化时，对应的总价能实时动态变化。与预期一致测试通过

清空购菜车点击一键清空购菜车，成功清除购菜车里所有菜品与预期一致测试通过

订单提交支付功确认下单点击确认下单，提交订单到后台，跳转至订单页面与预期一致测试通过

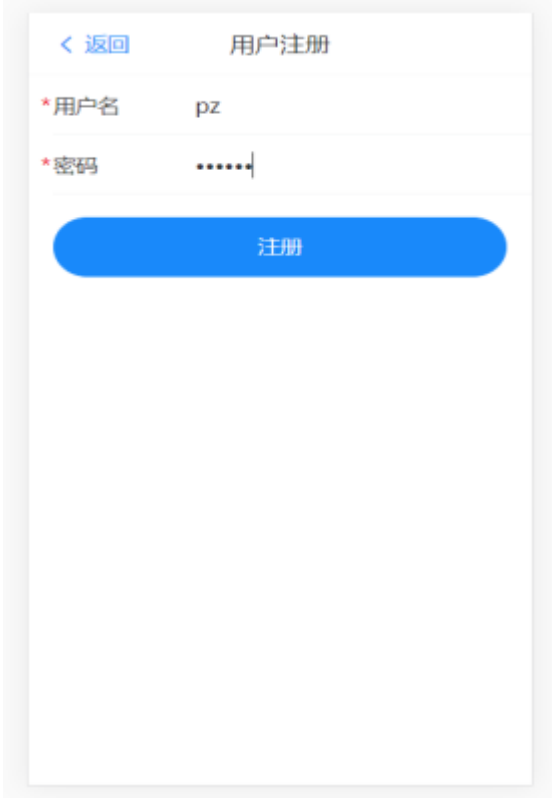
菜品进度订单页面实时显示，已点菜品的上菜进度与预期一致测试通过

订单信息展示展示座位号，用餐备注，菜品总价与预期一致测试通过

支付点击支付，调起微信支付，并能成功支付与预期一致测试通过

6.2 测试结果分析

用户注册页面与用户登录展示如图6.1所示。



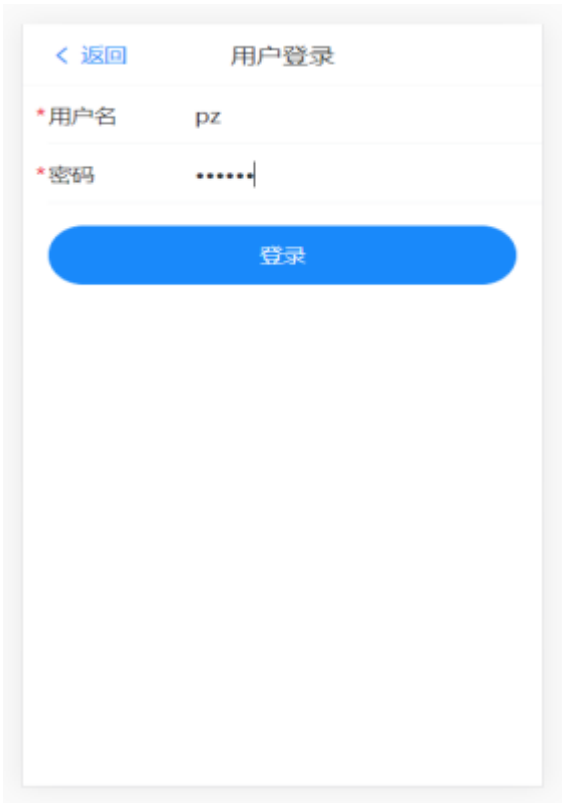


图6.1 用户登录注册测试图
个人中心页面与优惠券如图6.2所示。





图6.2 个人中心页面与优惠券测试图
餐厅留言板与餐厅信息如图6.3所示。





图6.3 餐厅留言板与餐厅信息测试图
热卖菜品首页展示与菜品分类展示如图6.4所示。

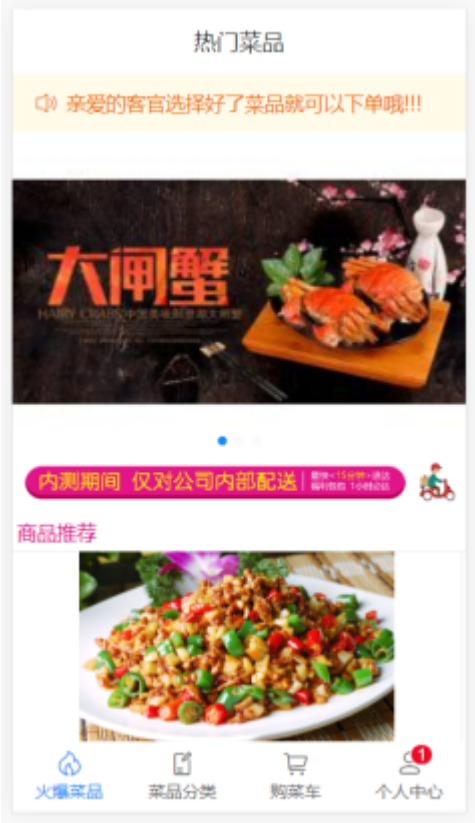




图6.4 餐厅留言板与餐厅信息测试图
菜品详情页面功能测试如图6.5所示。



价格:49.90

菜品详情

菜品评价

鲜活鲍鱼

包装数量：6只
净含量：300g

烹饪建议：清蒸、炖汤
储存方式：0-4℃冷藏保鲜

当日下单，次日上午9点到达濮阳开始配送，保证存活

百姓量贩

大连鲜活鲍鱼

鲜活鲍鱼



加入购物车

分享此菜品



鲜活鲍鱼300g约6个/份

价格:49.90

菜品详情

菜品评价

推荐指数: 

色: 

香: 

味: 

加入购物车

分享此菜品



图6.5 菜品详情页面功能测试图
购菜车页面与修改备注测试如图6.6所示。





图6.6 购菜车页面与修改备注测试图
空购菜车页面与订单页面测试如图6.7所示。





图6.7 空购物车页面与订单页面测试图
点击支付测试如图6.8所示。



图6.8 点击支付测试图

以上通过对测试用例的分析以及各个模块的功能性测试分析,能够得出本此系统设计的功能性是完整的,这对系统正确性至关重要,也保证了系统的稳定性。通过本测试环节,明确了系统功能模块是否将系统功能的需求完全实现,由以上的测试用例和功能实现图可以分析出系统的功能性设计比较完善。通过不断的测试,该系统已经能够保持稳定的运行,展示完整,需求实现完整,达到了系统设计的要求。

7 总结

移动互联网的快速发展,餐饮服务的需求也在不断增长,一大批移动手机用户可以在微平台上创建餐厅微博点餐平台,具有广阔的发展前景。与此同时,许多餐馆正在积极研究商业模式+互联网模式,将线下和线上服务结合

起来，支持现有的消费群体，利用潜在的消费群体，提高餐馆的盈利能力。在此基础上，开始了一个基于Web的智能订餐系统的开发与实现。在设计过程中，完全按照标准化的系统设计开发流程进行开发。

Web用户是使用该系统进行自助服务的用户。手机Web系统是消费者和商家之间的桥梁，商家可以达到自己省时省力的目的，并大大节省了业务开支。在开始设计工作之前，笔者在网上进行了一次调查，了解消费者需求的结构，通过访谈的方式讨论餐厅的功能需求。系统的最终目标。最后,提出了在设计和实现的详细点餐系统包括拓扑分析系统,系统软件的总体框架实现,具体功能模块等功能模块的实现,本文开发的Web系统主要有用户自定义点餐的功能、用户提供信息访问等功能，非常实用的多功能模块供消费者使用，拥有去新餐馆的经体验。

在系统的设计过程中对全栈的H5技术有了深刻的理解,对于常用的技术设计实现手段有了更深刻的了解。通过系统设计各项工作，让本系统在设计和实现的两耳光角度都有了丰富的功能，保证了系统完整性。

参考文献

[1] 胡秀华,宋艳妮,王长元.基于移动平台的点餐系统设计与实现[J].电子技术与软件工程,2018(15):39-40.

[2] 蒋宇捷. 从HTML5移动应用现状谈发展趋势[J]. 程序员, 2013(5):88-91.

[3] Kylesimpson. 你不知道的JavaScript.上卷[M]. 人民邮电出版社, 2015.

[4] 吕英华. 渐进式JavaScript框架Vue.js的全家桶应用[J]. 电子技术与软件工程, 2019(22).

[5] 李日斌, 詹舒波. 基于Vue的前端组件化研究与实践[J]. 2016.

[6] 曾广海. 基于Web前端组件化的个人博客系统的设计与实现[D].华中科技大学,2016

[7] Chaniotis IK, Kyriakou KID, Tselikas ND. Is Node.js a viable option for building modern web applications? A performance evaluation study. Computing, 2015,97(10): 1023-1044

[8] 封宇, 陈宁江. 基于 MVVM 架构的移动 Web 前端展示方案. 计算机与现代化,2014(11): 1-4

[9] 王金龙, 宋斌, 丁锐. Node.js: 一种新的 Web 应用构建技术. 现代电子技术,2015, 38(6): 70-73

[10] 占东明, 洪家伟, 陈希杨. Web 新兴前端框架与模式研究. 电子商务, 2016, 10(1): 65-66

[11] Plugge E, Membrey P, Hawkins T. The Definitive Guide to MongoDB. Apress,2010:2-8

[12] 黄贤立. NoSQL 非关系型数据库的发展及应用初探. 福建电脑, 2010, 26(7):30-31

[13] Cattell R. Scalable SQL and NoSQL data stores. Acm Sigmod Record, 2010, 39(4):12-27

[14] Smith K. Simplifying Ajax-Style Web Development. Computer, 2006, 39(5):98-101

[15] 牛仁腾. 基于Vue.js的表单可视化构建系统的设计与实现[D]. 2019.

[16] 钟强. Node.js平台下Web前端架构的研究: [硕士学位论文]. 武汉: 华中科技大学图书馆, 2013

指 标

疑似剽窃文字表述

- 1. Koa 并没有捆绑任何中间件，而是提供了一套优雅的方法，帮助您快速而愉快地编写服务端应用程序。之所以说Koa作用更纯粹，
- 2. 用户通过消息或扫描二维码在微信内打开网页时，可以调用微信支付完成下单购买的流程。
- 3. 在微信浏览器里面打开H5网页中执行JS调起支付。接口输入输出数据格式为JSON。
- 4. nonceStr：支付签名随机串，不长于32位；package：统一支付接口返回的prepay_id参数值；

表格检测详细结果

原文表格1:表4.2 User表

字段名	数据类型	说明
id	String	用户id,由MongoDB自动创建
userName	String	用户名
password	String	用户密码
createAt	Date	创建时间
lastLoginAt	Date	最后一次登录时间

相似表格1： 表3-5 Login方法

相似度： 44.44%

来源： 921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

Login
该函数为用户登录时的调用函数，该函数直接调用数据访问层的函数User GetUserByUserName (UserName)

返回值		
User表示账户信息		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码

相似表格2：表3-6 UserRegister方法
相似度： 44.44%
来源： 921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

UserRegister		
该函数为用户注册时的调用函数，该函数直接调用数据访问层的函数InsertUser(Password，UserName，UserRole)		
返回值		
Bool isSuccess表示操作成功与否		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码
UserRole	int	用户类型

相似表格3：表3-7 DelUser方法
相似度： 44.44%
来源： 921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

DelUser		
该函数为删除账户时的调用函数，该函数直接调用数据访问层的函数DelUserByUserId(UserId)		
返回值		
Bool isSuccess表示该用户删除是否成功		
参数	数据类型	描述
UserId	Int	用户ID
EditUser		
该函数为修改用户信息时的调用函数，该函数直接调用数据访问层的函数EditUserByUserId (UserId，UserName，Password，UserRole)		
返回值		
Bool isSuccess表示修改账户信息是否成功		
参数	数据类型	描述
UserId	Int	用户ID
UserName	String	用户名
Password	String	用户密码
UserRole	int	用户类型

相似表格4：表3-5 Login方法
相似度： 44.44%
来源： 921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

Login		
该函数为用户登录时的调用函数，该函数直接调用数据访问层的函数User GetUserByUserName (UserName)		
返回值		
User表示账户信息		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码

相似表格5：表3-6 UserRegister方法
相似度： 44.44%
来源： 921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

UserRegister		
该函数为用户注册时的调用函数，该函数直接调用数据访问层的函数InsertUser(Password，UserName，UserRole)		
返回值		
Bool isSuccess表示操作成功与否		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码

UserRole	int	用户类型
----------	-----	------

相似表格6：表3-7 DelUser方法

相似度：44.44%

来源：921_J200862063_黄裕谋_20131029-黄裕谋-《学术论文联合比对库》-2013-11-11

DelUser		
该函数为删除账户时的调用函数，该函数直接调用数据访问层的函数DelUserByUserId(UserId)		
返回值		
Bool isSuccess表示该用户删除是否成功		
参数	数据类型	描述
UserId	Int	用户ID
EditUser		
该函数为修改用户信息时的调用函数，该函数直接调用数据访问层的函数EditUserByUserId (UserId, UserName, Password, UserRole)		
返回值		
Bool isSuccess表示修改账户信息是否成功		
参数	数据类型	描述
UserId	Int	用户ID
UserName	String	用户名
Password	String	用户密码
UserRole	int	用户类型

相似表格7：Table 4.1 The entity table of user表4.1 用户实体表

相似度：55.56%

来源：006_045110106_SS_085211_徐帅-045110106-《学术论文联合比对库》-2013-12-04

字段	数据类型	备注
key	Key	主键
username	String	用户名
password	String	用户密码
type	Integer	用户类型
createTime	Date	用户创建时间
lastLoginTime	Date	用户上次登录时间

相似表格8：表3-1 用户基本信息数据表

相似度：50.00%

来源：武秀萍-51101201047-信息学院计算机系-武秀萍-《学术论文联合比对库》-2013-04-26

字段名	数据类型	备注
userID	Long	标记用户唯一性的ID
userName	String	用户名
password	String	用户密码
email	String	邮箱（google账户）
roleID	Int	用户角色

相似表格9：表3-4 Login方法

相似度：44.44%

来源：wys099_市民报料论坛系统的设计与实现 20131023(定稿版)-市民报料论坛系统的设计与实现-《学术论文联合比对库》-2013-10-24

Login		
该函数为用户登录时的调用函数，该函数直接调用数据访问层的函数User GetUserByUserName (UserName)		
返回值		
User表示账户信息		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码

相似表格10：表3-5 UserRegister方法

相似度：44.44%

来源：wys099_市民报料论坛系统的设计与实现 20131023(定稿版)-市民报料论坛系统的设计与实现-《学术论文联合比对库》-2013-10-24

UserRegister

该函数为用户注册时的调用函数，该函数直接调用数据访问层的函数InsertUser(Password, UserName, UserRole)		
返回值		
Bool isSuccess表示操作成功与否		
参数	数据类型	描述
UserName	String	用户名
Password	String	用户密码
UserRole	int	用户类型

相似表格11：表3-6 DelUser方法

相似度：44.44%

来源：wys099_市民报料论坛系统的设计与实现 20131023(定稿版)-市民报料论坛系统的设计与实现-《学术论文联合比对库》-2013-10-24

DelUser		
该函数为删除账户时的调用函数，该函数直接调用数据访问层的函数DelUserByUserId(UserId)		
返回值		
Bool isSuccess表示该用户删除是否成功		
参数	数据类型	描述
UserId	Int	用户ID
EditUser		
该函数为修改用户信息时的调用函数，该函数直接调用数据访问层的函数EditUserByUserId (UserId, UserName, Password, UserRole)		
返回值		
Bool isSuccess表示修改账户信息是否成功		
参数	数据类型	描述
UserId	Int	用户ID
UserName	String	用户名
Password	String	用户密码
UserRole	int	用户类型

相似表格12：表2.3 user数据格式

相似度：50.00%

来源：CDU928183944-《学术论文联合比对库》-2014-05-05

字段	类型	含义
username	String	用户名
password	String	用户密码
created_date	Date	创建日期
updated_date	Date	上次登录日期
phone_number	long	电话号码
mac	String	终端MAC地址

相似表格13：表 5-1用户信息表（cadre_info）

相似度：38.89%

来源：92740038738380610_2-《学术论文联合比对库》-2014-10-12

数据表字段	类型	描述
userId	Integer	ID编号,主键
userName	String	用户名
password	String	用户密码
department	String	所属部门
company	String	所属单位
Duty	String	职务
isAdmin	Boolean	管理员权限

相似表格14：表 4 用户信息表

相似度：55.56%

来源：21_GS1121139_丁凯旋-丁凯旋-《学术论文联合比对库》-2016-11-14

字段名称	字段类型	字段备注
Username	String	用户名
password	String	用户密码
Email	String	用户邮箱

Contact	String	联系方式
Create_time	date	创建时间

相似表格15：表SEQ 表_ * ARABIC4用户信息表

相似度： 55.56%

来源： 21_GS1121139_丁凯旋-丁凯旋-《学术论文联合比对库》-2016-12-23


字段名称	字段类型	字段备注
Username	String	用户名
Password	String	用户密码
Email	String	用户邮箱
Contact	String	联系方式
Create_time	date	创建时间


- 说明：
- 1.指标是由系统根据《学术论文不端行为的界定标准》自动生成的

2.红色文字表示文字复制部分;绿色文字表示引用部分;棕灰色文字表示作者本人已发表文献部分

3.本报告单仅对您所选择比对资源范围内检测结果负责

4.Email：amlc@cnki.net

 <http://e.weibo.com/u/3194559873>

 http://t.qq.com/CNKI_kycx

<http://check.cnki.net/>