南 阳 理 工 学 院

本科生毕业设计(论文)

学院(系)： 软件学院

专 业： 软件工程

学 生： 许冲

指导教师： 惠征

完成日期 2019 年 05 月

南阳理工学院本科生毕业设计（论文）

基于Vue框架的应用商城网站的设计与实现

Design and Implementation of the Application Mall Website Based on Vue Framework

总 计：毕业设计(论文) 28页

表 格： 14个

图 片： 27个

南 阳 理 工 学 院 本 科 毕 业 设 计(论文)

基于Vue框架的应用商城网站的设计与实现

Design and Implementation of the Application Mall Website Based on Vue Framework

学 院(系)： 软件学院

专 业： 软件工程

学 生 姓 名： 许冲

学 号： 1515925231

指导教师(职称)： 惠征 讲师

评 阅 教 师： 段云涛

完 成 日 期： 2019年04月25日

南阳理工学院

Nanyang Institute of Technology

基于Vue框架的应用商城网站的设计与实现

软件工程 许冲

1. 近十年来，智能设备的快速发展刺激了很多行业的兴起和成长，其中应用商店在智能设备的普及过程中也发挥了重要的作用。从官方数据中可以得到，大型应用商店如豌豆荚、应用宝等用户量已经破亿，其中的利润让许多公司争相开发应用商店，这一现象导致了目前的应用市场掺杂着质量、盗版等各种各样的问题。本系统研发的目的是集合当下热门及高质量的应用，致力于向用户打造一个高品质的应用商店。本商城研发技术主要采用了Vue作为项目框架、node为开发环境、Java和Js作为开发语言、MySQL作为数据库，引用的插件有axios网络请求插件、echarts图表插件、iViewUI框架等。本应用商城网站能够为用户提供了一个便捷下载应用的优质平台，从而能够实现用户所需的浏览应用功能、搜索应用功能、购买及加购应用功能和查看应用详情功能，同时本系统也具备用户登录、用户注册和用户退出功能，界面的简洁性和操作的便捷性能够使得用户在下载应用时获得较好的操作体验，从而保证用户下载应用这一过程顺利进行。
2. 应用商城；网站建设；Vue.js

Design and Implementation of the Application Mall Website Based on Vue Framework

Software Engineering Major Xu Chong

**Abstract:** In the past decade, the rapid development of intelligent devices has stimulated the rise and growth of many industries, among which the application store has also played an important role in the popularization of intelligent devices. From the official data, we can see that the number of users of large-scale application stores such as pea pods, application treasures has exceeded 100 million, and the profits of these stores have led many companies to compete for the development of application stores. This phenomenon has led to the current application market mixed with various problems such as quality, piracy and so on. The purpose of this system research and development is to collect current popular and high-quality applications, and to build a high-quality application store for users. The research and development technology of this mall mainly uses Vue as the project framework, node as the development environment, Java and Js as the development language, MySQL as the database. The reference plug-ins include Axios network request plug-in, echarts chart plug-in, iViewUI framework and so on. This application mall website can provide users with a convenient and high-quality platform for downloading applications, so as to realize the browsing application function, search application function, purchase and purchase application function and view application details required by users. At the same time, this system also has the functions of user login, user registration and user exit. The simplicity of the interface and the convenience of operation can make use of it. Users can get better operation experience when downloading applications, so as to ensure that the process of downloading applications goes smoothly.

1. Application Mall; Website construction; Vue.js

目 录

[1 绪论 1](#_Toc10139836)

[1.1 课题背景 1](#_Toc10139837)

[1.2 课题目的和意义 1](#_Toc10139838)

[1.3 主要内容 1](#_Toc10139839)

[1.4 论文组织结构 1](#_Toc10139840)

[2 相关技术 2](#_Toc10139841)

[2.1 数据库 MySQL 2](#_Toc10139842)

[2.2 前端框架 Vue 2](#_Toc10139843)

[2.3 页面框架 iView 2](#_Toc10139844)

[2.4 脚本语言 JavaScript 3](#_Toc10139845)

[2.5 图表插件 ECharts 3](#_Toc10139846)

[2.6 网络请求 axios 3](#_Toc10139847)

[3 需求描述与分析 3](#_Toc10139848)

[3.1 需求描述 4](#_Toc10139849)

[3.1.1 系统需求分析 4](#_Toc10139850)

[3.1.2 系统用例分析 4](#_Toc10139851)

[3.2 非功能性分析 5](#_Toc10139852)

[3.2.1 稳定性 5](#_Toc10139853)

[3.2.2 数据存储量 6](#_Toc10139854)

[3.2.3 兼容性 6](#_Toc10139855)

[3.2.4 安全性 6](#_Toc10139856)

[3.3 可行性分析 6](#_Toc10139857)

[3.3.1 技术可行性分析 6](#_Toc10139858)

[3.3.2 经济可行性分析 7](#_Toc10139859)

[3.3.3 法律可行性分析 7](#_Toc10139860)

[4 系统设计 7](#_Toc10139861)

[4.1 系统架构设计 7](#_Toc10139862)

[4.2 数据库设计 8](#_Toc10139863)

[4.3 界面设计 11](#_Toc10139864)

[4.4 业务流程设计 13](#_Toc10139865)

[5 编码与实现 16](#_Toc10139866)

[5.1 系统架构主要代码 16](#_Toc10139867)

[5.2 核心功能效果 20](#_Toc10139868)

[5.2.1 首页登录效果 20](#_Toc10139869)

[5.2.2 首页注册效果 21](#_Toc10139870)

[5.2.3 本系统的首屏效果 21](#_Toc10139871)

[5.2.4 搜索模块效果 21](#_Toc10139872)

[5.2.5 订单模块效果 22](#_Toc10139873)

[5.2.6 修改密码模块效果 22](#_Toc10139874)

[5.2.7 退出模块效果 22](#_Toc10139875)

[6 测试 23](#_Toc10139876)

[6.1 测试简介 23](#_Toc10139877)

[6.2 测试方法 23](#_Toc10139878)

[6.3 测试内容 23](#_Toc10139879)

[6.4 测试结果 23](#_Toc10139880)

[6.5 测试总结 24](#_Toc10139881)

[结束语 26](#_Toc10139882)

[参考文献 27](#_Toc10139883)

[致谢 28](#_Toc10139884)

# 绪论

本项目采用Vue框架搭建，设计并实现了一个能够提供给用户更方便快捷的应用商城网站。

## 课题背景

互联网技术在逐渐发展中，相应的软件应用技术也在飞速发展，用户在面临应用下载及使用过程中有了更多的选择，而选择的多样性在于应用商店出现，它为应用程序和用户搭建起了便捷的桥梁。应用商店作为一个互联网新兴方向，在近十年来逐渐影响到每个人的生活或工作中，并逐渐形成了一条产业链。但是虽然目前市场上已经有了不少的应用商店，但是部分应用商店充斥着各种广告和捆绑应用，给用户平添了不少的烦恼。在当今处于信息化高速建设的浪潮中，方便快捷的操作方式和体验更加符合潮流并为人们所接受，所以开发一个优秀的应用商城网站对人们的工作和生活也会有极大的帮助。

## 课题目的和意义

应用商店起源于应用而服务于用户，在应用和用户之间搭建起了桥梁，三者目前形成了相互依存的关系。应用商店的存在是必要的，它能够从用户的需求和应用的实践出发，让应用真正的影响人们的工作生活和服务于人们的工作生活。

本课题是最终目的是能够实现搭建一套应用商城网站，以独特的简单流程面向广大用户，提高他们在下载应用过程中的效率，满足正常的应用需求，同时也能够让高质量应用在本网站上进行推广和展示，实现互利共赢的愿景[1]。

## 主要内容

本论文主要内容包括设计目的、设计过程和设计结果，三者共同解释了如何搭建一套基于Vue的应用商城。应用商城主要服务于两大目标群体：用户和应用商，通过应用商上架对应的应用来满足用户的下载需求，通过用户下载应用来满足应用商的盈利推广需求，所以本系统主要实现目标为实现应用的列出和用户的下载操作[2]。

本课题主要实现功能为应用下载，所以本课题会大篇幅的讲解本系统应用下载的流程和下载过程所需要的功能需求，同时本课题还主要讲解了框架搭建有关方面的知识，让读者能够从业务和技术两方面来理解本项目，从而达成本论文预期的目标。

## 论文组织结构

本论文的结构描述如下：

第一部分为绪论，着重介绍本论文的背景条件。

第二部分为相关技术，着重介绍本系统研发的所需技术。

第三部分为需求描述与分析，了解本项目的需求和本项目的总体设计。

第四部分为概要设计，包括本项目的系统架构、数据库设计和系统各个模块的具体功能设计。

第五部分为详细设计，着重介绍本系统的流程与功能。

第六部分为编码和实现，这里会有各核心功能的代码片段展示和各功能的截图展示。

第七部分为测试，对本系统进行测试，确保本系统可以正常操作以及能够实现对应的功能，如果有问题则要找出bug并解决。

# 相关技术

## 数据库 MySQL

MySQL数据库是一个为广大开发人员所喜爱的数据库，对于开发者而言，它可以显著的提高开发者的开发效率，能够让其优势进一步扩大，从而让越来越多的中小企业或个人选择它，这要归功于它的开源特性，直接导致了使用成本的降低和花费时间的降低。

MySQL的使用首先需要创建数据库，其中创建表这部分是比较难的。

增：数据导入表中，可以通过LOAD DATA和INSEART语句来完成此项操作。

查：为检索信息，主要为SELECT语句，例如SELECT \* FORM pet。

如何获取数据库和表的信息也是重点，MySQl也给出了一些语句来解决这一问题，可以通过SHOW DATABASES列出由服务器管理的所有数据库[3]。

## 前端框架 Vue

使用Vue对于开发者而言是比较友好的，因为它能够集百家之长，摒百家之短，它可以同时实现angular和react的优点，也避免了angular和react的缺点，Vue采取了轻量级的特点，通过简单的api集合成齐全的文档，它虽然简单但并不功能单一，且能够完善的构建大型项目得益于它的三个特性：模块化、组件化和路由。通过将页面的一个个文件组件化，能够非常适合于文件的管理和复用，大大减少了开发的时间，显著提高了开发的效率。路由可以实现单页面应用，能够实现各个组件的按需加载等[4]。

## 页面框架 iView

iView具备高质量、功能丰富的API，同时也是有着自由灵活的使用空间、细致、漂亮的UI、详细的文档和自由度高的自定义样式。通过npm和CDN引入来进行安装，引入项目中可以通过webpack 和main.js 中配置或者直接按需引入（减少文件体积），需要注意的是iView需遵循Vue组件之间规则（子父、父子、兄弟之间传值规范）[5]。

高效率的编程，离不开适合的组件，iView无疑是本系统所适合的那个，结合组件具备的按需加载特性，可以明显的减少文件的体积。iView的标签需要按照书写规范，所有标签首字母大写的形式，但是其中两个组件在使用中需要注意，分别为Switch和Circle，因为这为Vue所限制，可以通过loader配置，来开启标签前缀的写法来解决这问题[6]。

## 脚本语言 JavaScript

Js是浏览器语言，它作为脚本语言而存在，用来给页面添加交互行为。Js具备非常多的优点，它是一种在程序运行时逐行解释的语言，这与其他C、C++语言所不同之处。Js还是如此的简单，他是用弱类型的变量类型，对使用的数据格式并没有严格的要求划分。它是具备动态性的。它还具备跨平台性，Js的使用并不依赖于操作系统，如win和mac，它依赖于浏览器的内核，同一种代码在不同浏览器内核的支持下是不一样的，但是脚本语言Js在经过这么多年的发展，也已经完美的应用在各个浏览器中，已经为各个浏览器所支持 [8]。Js最重要的点在于其对象、逻辑和算数特性，js可以提供多个内建对象也可以自定义对象，对象是具备自身独立的属性的，方法是能够在对象上执行的方法及要点[7]。

## 图表插件 ECharts

ECharts是一个数据可视化的插件，能够在PC端和移动端运行代码，可以兼容当前大多数浏览器，它可以通过与其他矢量图形库相关联，结合成数据可视化图表，它可以实现的特点如下[8]：

首先ECharts具备丰富的可视化类型图表，ECharts具备多种数据格式且直接可以转化使用，前提是通过简单的设置即可完成数据和图像之间的转换，这一简单步骤节省了大多数场景下的数据转化所耗费的时间。

多个数据组件可以共享使用同一接口返回的数据，同一数据可以作用于多个图表组件中使用，这依赖于它的多渲染功能以及跨平台使用。传入数据的改变导致图表视图的改变，因此动态数据的实现也变得异常简单[9]。

## 网络请求 axios

在使用axios之前，理解Promise的概念尤其重要，它是用来传递异步操作数据的一个对象，代表了未来才会知道结果的事件。Promise的出现主要是解决回调问题，比如在获取数据时需要调用很多接口，而其中有些接口依赖于其他接口返回的数据，这时候就需要promise来实现无需嵌套[10]。

# 需求描述与分析

本部分主要阐述基于Vue的应用商城系统的设计与实现的总体需求和相关界定，并规划出本系统的功能逻辑和结构，本部分也是本项目成功开发的前提条件。

## 需求描述

### 系统需求分析

本系统为应用商城系统，目的是提供给用户所需要的应用下载，实现一套正确的下载流程。首先系统需要实现的功能如下：应用展示、搜索、用户信息展示、退出、购物车、订单，其中核心点是应用展示需求，应用展示需求需要展示不同类别的应用，这需要涉及到多页面构建和路由划分，购物车和订单是本项目的独特需求，满足了付费应用的盈利需求。这一切需求都是围绕着用户开展的，所以需要用户参与到各个功能模块中，如何保存用户信息并在每次请求中调用也是需要实现的重点[11]。

### 系统用例分析

本系统只有一个用户类别，应用的增删改查都是直接通过操作数据库数据来实现，每个用户可以进行的功能都是相同的，下图为唯一用户的需求用例图3-1:

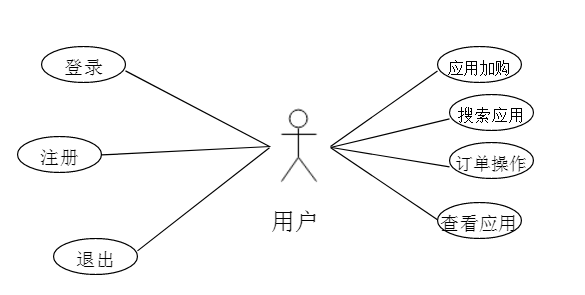


图3-1 用户需求用例图

本系统的主要目标是将应用呈现给用户，所以需要合理的将应用列表展示在页面上，还需要加以区分应用的类别。本系统需要具备搜索功能，因为应用数量很庞大，搜索功能能很快的将用户所需应用呈现给用户。本系统还要对付费应用下载设计购买流程。鉴于本系统具备登录功能，需要增加可以对用户进行改密的模块。以下表为本系统各个功能需求的用例描述[12]：

“搜索应用”用例描述如下表3-1所示：

表3-1搜索应用

| 用例目的 | 搜索应用 |
| --- | --- |
| 参与者 | 普通用户 |
| 前置条件 | 普通用户在搜索框中输入相应的应用名称 |
| 基本事件流 | （1）进入本系统主界面  （2）点击顶部导航栏中的搜索图标  （3）在出现的搜索框中输入相应的应用名称  （4）弹出搜索的应用详情信息 |
| 后置条件 | 搜索成功，应用数据成功展示 |

“加购应用”用例描述如下表3-2所示：

表3-2加购应用

| 用例目的 | 加购应用 |
| --- | --- |
| 参与者 | 普通用户 |
| 前置条件 | 普通用户点击应用详情中的下载按钮 |
| 基本事件流 | （1）进入本系统主界面  （2）点击应用中的下载图标  （3）在如果该应用为付费应用则会加入购物车中  （4）购物车可以增删改查  （5）购物车生成订单 |
| 后置条件 | 加购成功，生成订单成功 |

“订单结算”用例描述如下表3-3所示：

表3-3订单结算

| 用例目的 | 订单结算 |
| --- | --- |
| 参与者 | 普通用户 |
| 前置条件 | 普通用户进入订单列表中 |
| 基本事件流 | （1）进入本系统主界面  （2）点击顶部导航栏中的商店图标  （3）弹出订单列表  （4）订单可以增删改查  （5）结算订单 |
| 后置条件 | 结算订单成功，购买应用成功，下载应用 |

## 非功能性分析

### 稳定性

稳定性是一个网站盈利的前提，网站越稳定，潜在的日增长用户就会越多。如果不稳定，直接会导致偶尔网站打不开的情况，间接导致用户数量减少、搜索引擎排名靠后等等。网站的稳定性由以下几部分决定：

（1）服务器的稳定：服务器的性能是否良好、访问时长是否在用户接受范围内、会不会受到攻击等等，这些都是影响服务器稳定性的关键。

（2）内容的稳定：这是与前端页面排版相关的，但是偶尔加载不出css文件也是存在的，这就需要访问过程中的稳定性支撑。

（3）cdn的稳定：前端页面建设有时会需要cdn，而cdn是以链接形式存在的，这就非常依赖于厂商cdn的稳定，所以直接下载插件到本地是不错的解决办法。

（4）后续更新的稳定性：网站不可能是在创建后不变化的，网站更新依赖于程序员的开发，而开发内容是否符合前者网站的开发特性记忆是否具备之前的风格，这都是在开发过程中需要着重考虑的。

（5）运营的稳定性：这就依赖于网站开发公司的运营情况，公司的运行正常，网站就会不会因服务器到期、黑客入侵等问题导致崩溃。

### 数据存储量

应用商城对于用户来说最主要的就是能否找到所需要的应用信息，这就是与网站的数据存储量息息相关的，数据的存储量是存放在数据库中的，数据库中的应用数据数量越多，用户就越容易搜索到自己想要得到的应用数据。

### 兼容性

浏览器兼容性一直是前端工程师所面临的问题，在IE等老牌浏览器中是否成功以及在谷歌等新版本浏览器中是否合适，浏览器厂商一直都在试图实现新版本css兼容，但是大部分都未能实现真正的标准，目前主流的浏览器内核有[13]：

Trident内核主表代表为IE浏览器，Gecko内核主要代表为Firefox浏览器，Presto内核主要代表为Opera浏览器，Webkit内核主要代表为Chrome浏览器和Safari浏览器。

这些不同的浏览器内核，要想实现统一效果的页面，需要css样式针对每一浏览器进行修改，修改方式为添加私有前缀，如下所示：

Trident内核浏览器前缀为-ms，Gecko内核浏览器前缀为-moz，Presto内核浏览器前缀为-o，Webkit内核浏览器前缀为-webkit。

### 安全性

随着黑客数量的增加和金钱诱惑的驱动，网站面临着被盗取用户信息的安全性问题，所以防止站点被攻击，网站安全性建设是尤为重要的。安全性可能会涉及到用户的个人信息、用户的交易记录、用户绑定的银行卡等数据，一旦泄露或许会造成损失，这是网站和用户都不愿意看到的事情，所以网站的安全性建设也是项目上线后所必备的。

## 可行性分析

可行性分析是对项目的内容和条件各方面进行研究和分析，以及在项目成型后的盈利、影响等方面进行分析，最终得出的项目的建设分析意见和总体流程方向，为项目提供一个明确的实现目标。

### 技术可行性分析

技术分析完成了系统分析，是其中难度较大的部分，技术可行性分析可以包括实现分析和风险分析。技术实现分析是本项目的可维护性、性能程度、拓展性和稳定性的分析。可维护性主要考虑的是项目的代码规范问题，代码规范能够让后续代码更新难度更低，便于程序员进行操作。稳定性主要考虑的是项目的网络问题，因为项目的接口响应时间也是影响用户体验的重要一环。风险分析是因为项目或多或少需要引入三方插件，在享受第三方插件的便利性的同时也需要承担第三方插件的不确定性[14]。

### 经济可行性分析

项目最终目的是以盈利为目的，但是项目经济支出也是前期需要注意的重要因素。经济可行性分析是要是很多方面来考虑，对于本项目来说，首先本项目要具备美观的页面和流畅的操作，以吸引更多的用户，从而保证了客源，其次本项目需要具备较少的开发成本，以增加抗风险能力和减少失败成本。

### 法律可行性分析

法律可行性是指法律规定和本项目实现的统一程度，法律规定与项目实现相匹配的程度越高，后续项目上市就不会受到政策影响导致突发情况发生，从而项目的保障性就会越强。

# 系统设计

概要及详细设计是一个通过结合用户操作以及用户需求生成框架模型和视觉模型的过程，能较好的展示出用户交互布局，这能够很好的将用户研究和设计连接起来，是用户需求和目标转换成成果的重要阶段。

## 系统架构设计

系统架构是对需求描述中的所需技术实现构架、规划流程，通过运用整套、完整的工具，完成本系统的任务。本系统的前端部分是用Vue框架搭建，结合Vue框架的模块化、组件化和路由，通过将系统的主体架构进行了优化处理，得出了系统所有功能为几大模块。前端部分采用了axios作为网络请求工具、iView作为UI框架、webStorage作为数据缓存、eCharts作为数据展示工具。本项目系统架构如下图4-1所示：

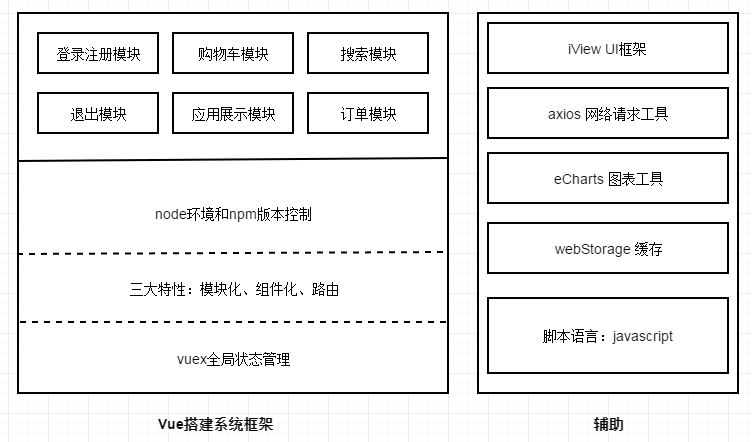


图4-1系统架构图

如图所示：除了本系统模块和所需插件外，本系统还需node环境为本系统的开发环境，git版本控制工具，vuex状态的管理等，当然，万变不离其宗，代码主要还是由脚本语言Js编写。本系统的后端部分由SpringBoot集成SSM开发，开发环境为win10，开发工具为eclipse，服务器为tomcat容器，插件为maven插件。

## 数据库设计

数据库也就是本系统的应用数据和用户数据的存放库，通过对存放数据的增删改查，能够实现本系统中所有对数据的要求。数据库操作中的建表是比较重要的一步，而如何建表则是在建表前所需要考虑的。

本数据库设计主要有以下几类实体。

（1）用户实体，本系统目前的用户群里只有一种，就是普通用户群体，普通用户群体主要包括用户名和用户标识符实体属性，其中标识符为主键，用来鉴别用户登录和用户身份，用户名是用户本系统所展示的，用来在页面上看到用户的登录情况。如下图4-2所示：

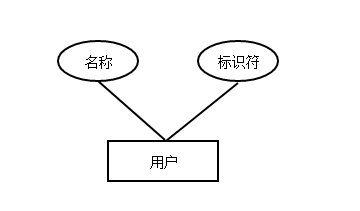


图4-2用户实体属性图

（2）应用实体，包括应用名、类型、描述、链接，其中应用名为主键。应用名的作用是供搜索和展示用途，类型是应用分类的用途，描述是用户详情的用途，链接是供用户下载的用途。如下图4-3所示：

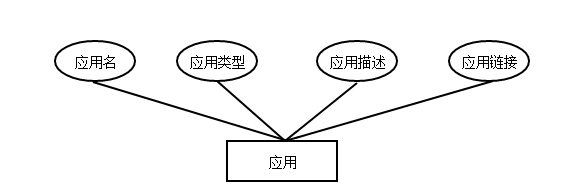


图4-3应用实体属性图

（3）购物车实体，包括应用名、加购时间、应用价格、应用src。购物车可以生成订单，在生成的时候保存加购时间、应用价格和应用名称。如下图4-4所示：

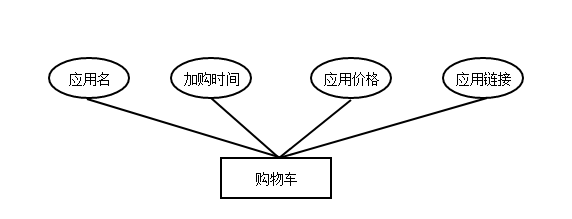


图4-4 购物车实体属性图

（4）订单实体，包括订单名称、创建时间、订单价格、应用数组，其中订单名称为主键。订单可以选择结算和删除，而这些操作依赖于订单名称。创建时间是本订单创建的时间，订单价格用于结算操作，应用该数组用于展示和下载用途。如下图4-5所示：

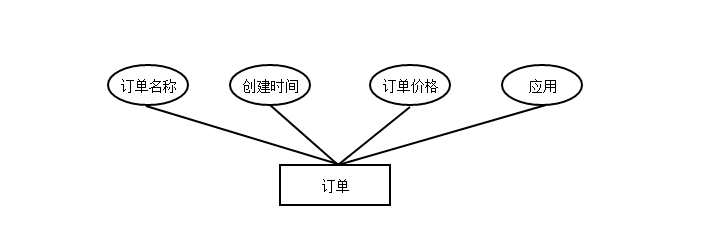


图4-5 应用实体属性图

本系统ER图如下图4-6所示：

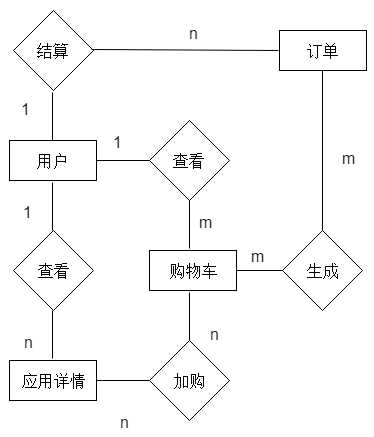


图4-6系统ER图

本数据库设计主要有以下几类表：

（1）用户登录注册信息表，包括用户名username、密码password、验证码text。如下表4-1所示：

表4-1用户登录注册信息表

| 字段名 | 字段描述 | 数据类型 | 长度 | 可空 |
| --- | --- | --- | --- | --- |
| username | 用户名 | varchar(40) | 40 | 否 |
| password | 密码 | varchar(40) | 40 | 否 |
| text | 验证码 | varchar(40) | 40 | 否 |

（2）应用数据信息表，包括应用名appname、应用类型type、应用描述test、应用链接src。如下表4-2所示：

表4-2应用数据信息表

| 字段名 | 字段描述 | 数据类型 | 长度 | 可空 |
| --- | --- | --- | --- | --- |
| appname | 应用名 | varchar(40) | 40 | 否 |
| type | 应用类型 | varchar(40) | 40 | 否 |
| test | 应用描述 | varchar(40) | 40 | 是 |
| src | 应用链接 | varchar(40) | 40 | 否 |

（3）购物车信息表，包括应用名appname、加购时间time、应用价格price、应用链接src。如下表4-3所示：

表4-3购物车信息表

| 字段名 | 字段描述 | 数据类型 | 长度 | 可空 |
| --- | --- | --- | --- | --- |
| appname | 应用名 | varchar(40) | 40 | 否 |
| time | 加购时间 | varchar(40) | 40 | 否 |
| price | 应用价格 | varchar(40) | 40 | 否 |
| src | 应用链接 | varchar(40) | 40 | 否 |

（4）订单信息表，包括订单序号name、订单创建时间time、订单总价格price、应用链接src。如下表4-4所示：

表4-4订单信息表

| 字段名 | 字段描述 | 数据类型 | 长度 | 可空 |
| --- | --- | --- | --- | --- |
| name | 订单序号 | varchar(20) | 20 | 否 |
| time | 订单创建时间 | varchar(40) | 40 | 否 |
| price | 订单总价格 | varchar(40) | 40 | 否 |
| src | 应用链接 | varchar(40) | 40 | 否 |

（5）用户信息表，包括用户名称name、标识符userSession。如下表4-5所示：

表4-5用户信息表

| 字段名 | 字段描述 | 数据类型 | 长度 | 可空 |
| --- | --- | --- | --- | --- |
| name | 用户名称 | varchar(20) | 20 | 否 |
| userSession | 用户标识符 | varchar(40) | 40 | 否 |

## 界面设计

界面设计的解释如下：界面设计是针对系统中每一层次的每一代码块实现的功能进行分析考虑，主要目的是对系统的各个功能进行一次重新划分和构建，这一步骤可以有效的减少系统的错误率和功能的稳定性。界面设计对于用户来说或许是重要性比较高的，因为它是连接用户和程序的唯一媒介，界面设计或清新、或庄重、或酷炫，这带给用户的体验是不同的，从而直接影响到用户的交互体验。

登录界面设计有一个card框，用户可以进行输入账号密码操作。Card框内可以进行登录操作，还会有一个注册的链接，点击会呈现弹出框，用来进行注册操作。登录设计图如图4-7所示：

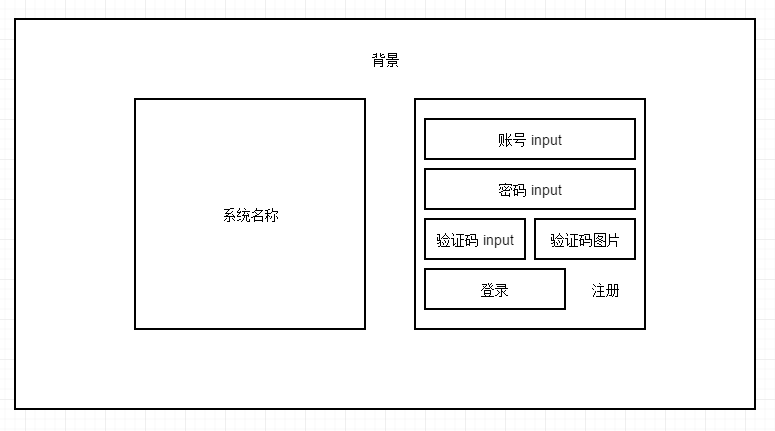


图4-7登录设计概念图

注册操作现在设计的是弹出一个窗体，窗体内包括注册的数据格式，还会有一个确认按钮，用来提交注册的信息。注册设计图如图4-8所示：

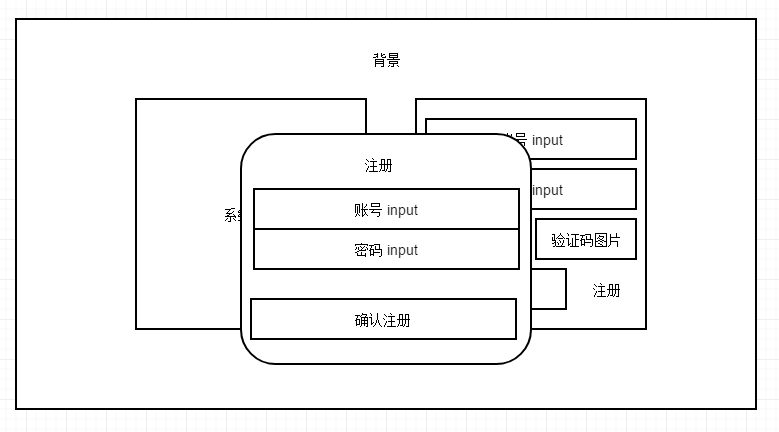


图4-8注册设计概念图

系统界面由三部分组成：分别为顶部工具栏、左侧菜单栏和主要内容区域。顶部工具栏是由订单图标、改密图标、退出图标、搜索图标和用户数据组成。左侧菜单栏将会有本系统的名称和应用的分类列表。主要内容的里面主要呈现应用的数据列表。

综合以上所需需求，设计出各功能模块的设计图如下：

（1）主系统界面设计图如图4-9所示：

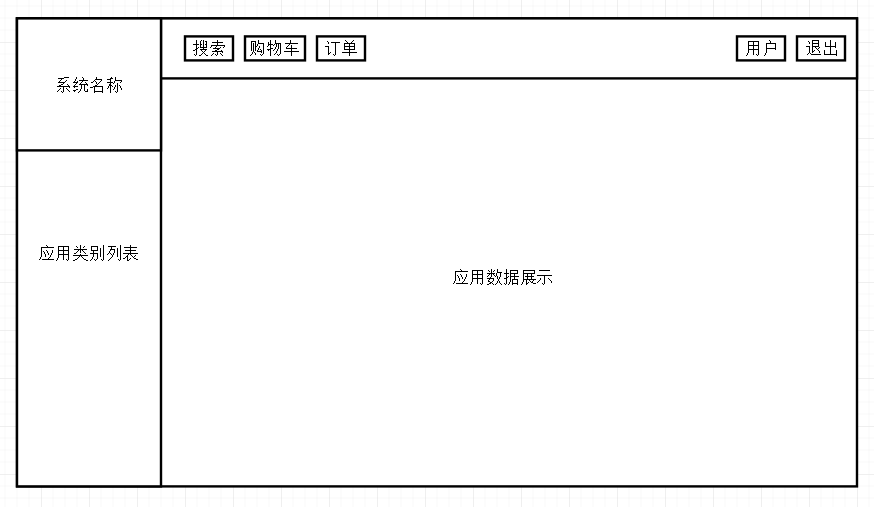


图4-9主系统界面设计概念图

（2）搜索功能设计图如图4-10所示：

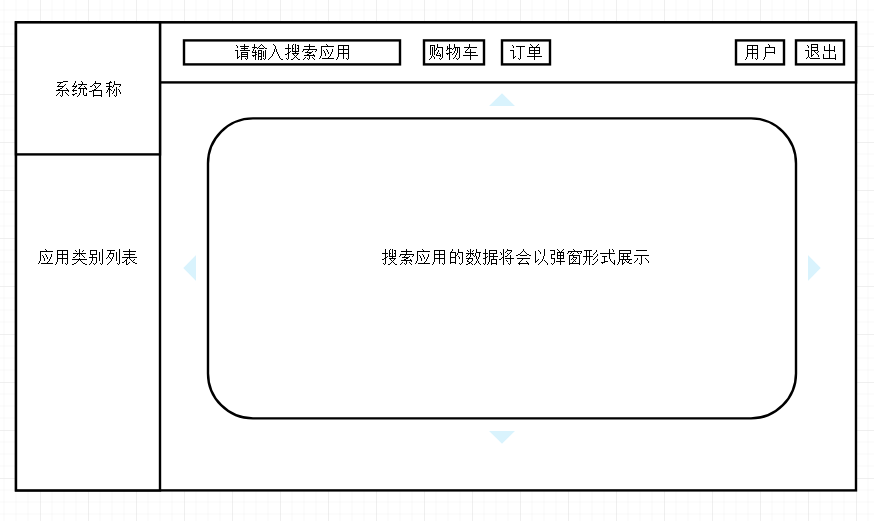


图4-10搜索设计概念图

（3）加购功能设计图如图4-11所示：

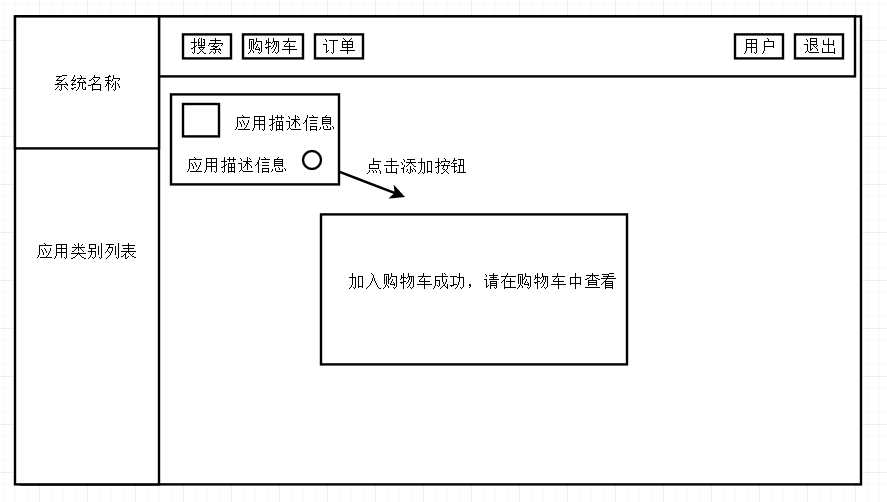


图4-11加购设计概念图

（4）订单功能设计图如图4-12所示：

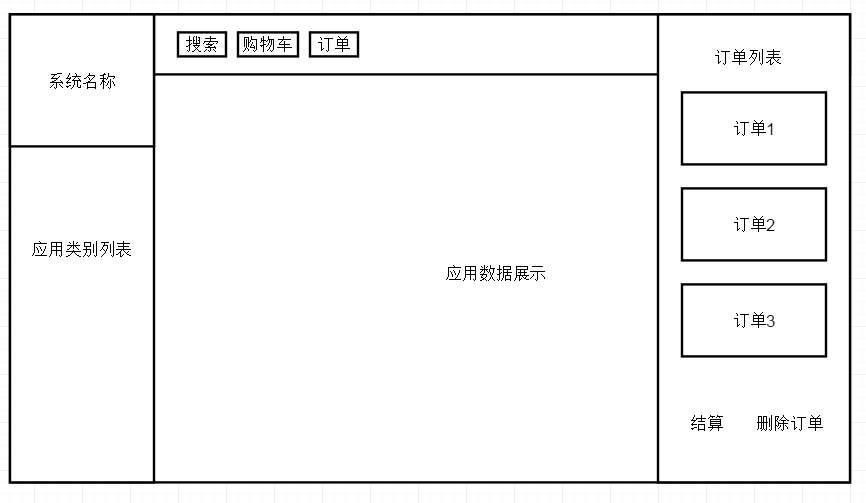


图4-12订单设计概念图

（5）退出功能设计图如图4-13所示：

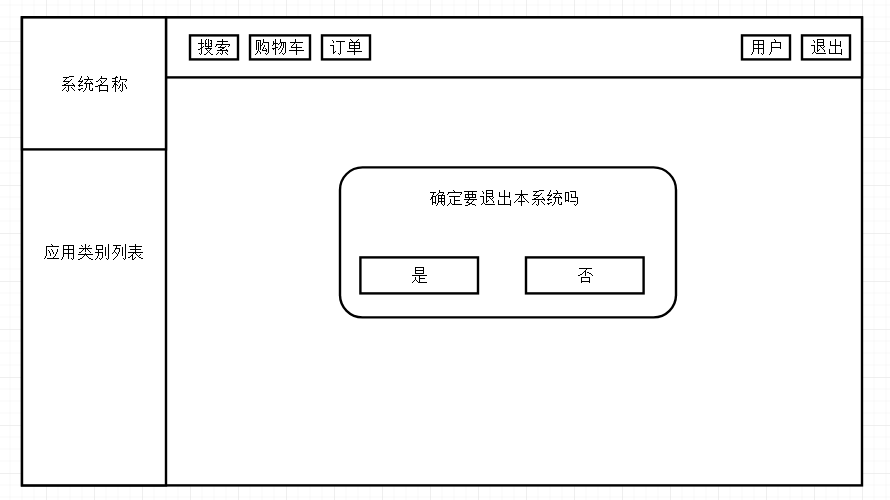


图4-13退出设计概念图

## 业务流程设计

本系统的第一个页面为登录注册页面，登录注册页面能够提供给用户登录和注册功能。登录成功后会进入本系统的主框架及首页，主框架包括左侧的导航栏及顶部的工具栏，导航栏包括三页面（排行、首页和费用），工具栏包括四部分（搜索、订单、改密和退出），每部分与每部分之间都有对应的流程操作。

（1）用户登录流程：当用户准备进入应用商城系统时需要登录，如果账号或密码错误，会提示信息错误并重置账号密码输入框的内容，如果输入数据正确，会提示信息正确并进入系统主页面。如图4-14所示：

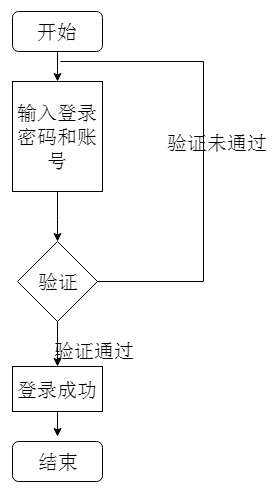


图4-14 用户登录流程图

（2）用户修改密码流程：用户需要输入内容为旧密码和新密码，在点击提交按钮时，后台会验证数据是否正确，验证正确后会提示需要输入新密码，验证格式正确后会提示修改密码成功。如下图4-15所示：

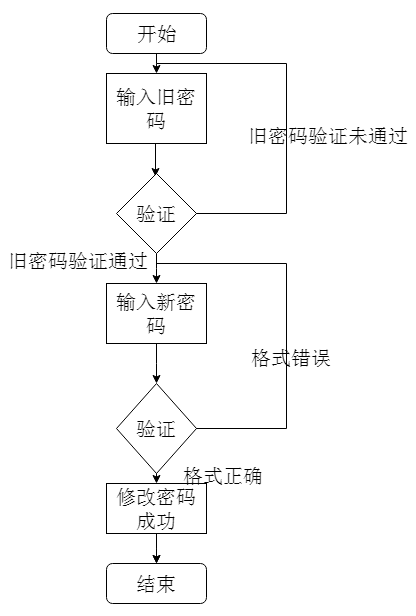


图4-15用户修改密码流程图

（3）用户搜索应用流程：用户可以在搜索框中搜索应用名称，搜索到对应的应用信息后，会显示出该应用信息，不然将会提示未找到。如下图4-16所示：

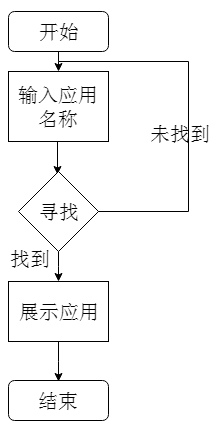


图4-16用户搜索应用流程图

（4）用户下载并加购流程：用户下载应用时会判断该应用是否是付费应用。如果不是付费应用则直接可以下载，如果是付费应用，则需要进行加购操作，添加进购物车后可以自主选取并生成订单。如下图4-17所示：

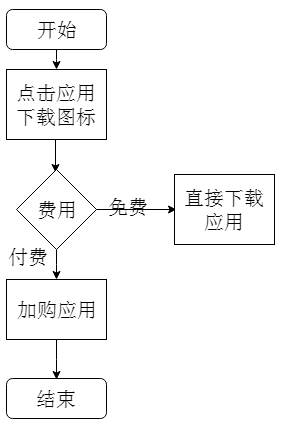


图4-17用户下载并加购流程图

（5）用户处理订单流程：购物车生成的订单会在订单模块进行展示，主要展示每一个订单中的应用数据，可以进行结算或删除操作。其中，删除会清楚掉对应的按订单和购物车信息，结算会对本订单加购的应用进行结算操作。流程如下图4-18所示：

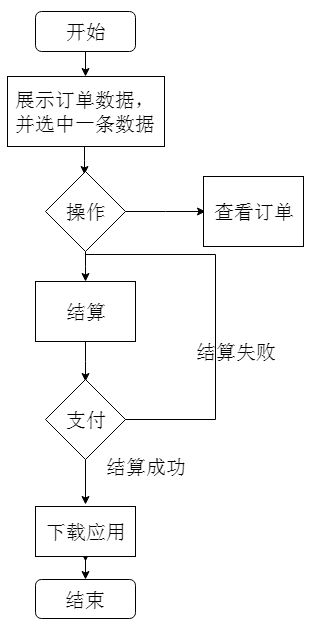


图4-18 用户处理订单流程图

以上各图为本系统各功能模块的流程图，其中重要的模块流程在于应用模块、购物车模块和订单模块，三者相辅相成，通过应用模块提供数据给购物车模块和购物车模块提供数据给订单模块，共同组成了本系统的核心流程。

# 编码与实现

运行的软件是需要编写的，在系统编写的过程中是需要编码来实现的。任何一个网站页面，换成源码得到的就是一个个编码组成的文件，经过浏览器编译就得到我们看到的模样[7]。

## 系统架构主要代码

本系统核心功能主要代码分别如下：

（1）Vuex状态管理功能：这里是对本系统的加载状态、用户登录状态等进行管理，通过引入vuex和vue来达到预期的目的。其中，storageToStore函数是来判断store中是否有对应的token信息，如果没有，则调用localStorage中缓存的数据。SignOut是退出函数，用来清空localStorage中缓存的数据和vuex状态存储的数据。routerToStore是在浏览器刷新时能够记录当前状态，不会跳转到默认页面。如下所示：

import Vue from 'vue' import Vuex from 'vuex' Vue.use(Vuex)

const state = {token:'',show:false,active:'', open:'',}

const mutations = {

storageToStore(state, data) {

let token = JSON.parse(localStorage.getItem('token'))

if (token != undefined) { state.token = token}

},

signOut(state){ localStorage.clear();state.token = ''},

routerToStore(state,data) {

state.active = data.meta.active

state.open = data.meta.open}}

export default new Vuex.Store({state,mutations,})

（2）main.js全局插件管理：这里引入了全局可用的插件，包括iView按需引入、路由router、index.js组件化管理、base.js全局函数、http.js全局网络请求函数等等，主要语法就是import…form…（从路径引入）、vue.use(使用插件)和vue.prototype（使用插件的某个外部可以引用的代码块）。如下所示：

import Vue from 'vue'

import App from './App'

import store from './vuex/store'

import router from './router'

import './assets/css/common.css'

import 'iview/dist/styles/iview.css'

import './assets/css/theme.less'

import components from './components/index.js'

Object.keys(components).forEach((key) => {var name = key.replace(/(\w)/, (v) => v.toUpperCase())Vue.component(`vue${name}`, components[key])})

import vuex from 'vuex' Vue.use(vuex)

import VueRouter from 'vue-router' Vue.use(VueRouter)

import base from './assets/js/base.js'; Vue.use(base)

import axios from 'axios' Vue.prototype.$axios = axios

import http from './assets/js/http.js' Vue.prototype.$http = http

Vue.config.productionTip = false

new Vue({el: '#app',router,store,components: { App },template: '<App/>'})

（3）项目信息管理：这里是本系统的总体信息，包括项目的名称、项目的版本号、项目的描述、项目的作者、项目的命令行操作、项目所需插件等等。如下所示：

{

"name": "xuchong","version": "1.0.0","description": "A Vue.js project",

"author": "xuchong <xuchong@nbapi.com>","private": true,

"scripts": {

"dev": "webpack-dev-server --inline --progress --config build/webpack.dev.conf.js",

"start": "npm run dev",

"unit": "jest --config test/unit/jest.conf.js --coverage",

"e2e": "node test/e2e/runner.js",

"test": "npm run unit && npm run e2e",

"lint": "eslint --ext .js,.vue src test/unit test/e2e/specs",

"build": "node build/build.js"

},

"dependencies": {"axios": "^0.18.0","echarts": "^4.1.0","iview": "^3.2.2","less": "^3.9.0","less-loader": "^4.1.0","qs": "^6.7.0","vue": "^2.5.2","vue-router": "^3.0.1","vuex": "^3.0.1"},

}

可以看到，本系统目前所引入插件为axios、eCharts、iView、less、less-loader、qs、vue、vue-router、vuex，并包括了每个插件对应的版本号。

（4）项目网络请求：这里是通过插件axios来管理网络请求，本系统主要由post和get两种请求方式，如下所示：

import axios from 'axios' import Vue from 'vue' import store from '../../vuex/store' import router from '../../router/index' import qs from 'qs'

let http = {} var completeUrl = 'http://localhost:8080 '

http.post = (url, body) => {

return axios({

method: 'post',url: completeUrl+url,data: body,

headers: {'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8'},

transformRequest: [function (data) {

let ret = ''

for (let it in data) {

if(data[it].constructor === Array){

ret += encodeURIComponent(it) + '=' + JSON.stringify(data[it]) + '&'

}else{

ret += encodeURIComponent(it) + '=' + data[it] + '&'

}

}

return ret

}]

}).then((response) => response.data).catch(error => error)

}

http.get = (url, params) => {

return axios.get(completeUrl+url, { params }).then((response) => response.data)

.catch(error => error)

}export default http

（5）项目路由管理：这里通过引入各个页面的本地地址，在判断目标页面属性的情况下，来对本系统的页面url进行管理。

import Vue from 'vue' import Router from 'vue-router' import store from '../vuex/store'

import { LoadingBar } from 'iview' Vue.use(Router)

const routes = [{path: '/',name: 'login',

component: resolve => require(['../views/login.vue'], resolve)}。。。。。。{}]

const router = new Router({mode: 'hash',base: '/',routes})

LoadingBar.config({color: '#5cb85c',failedColor: '#50e37f',height: 3})

router.beforeEach((to, from, next) => {

LoadingBar.start()

if(to.matched.length == 0){next('/error')}

else{

if (!store.state.token) {store.commit('storageToStore')}

if(to.name != 'login' && to.name != '404' && to.name != null)

{store.commit('routerToStore',to)}next()

}

})

router.afterEach((to, from) => {LoadingBar.finish()})

export default router

（6）项目配置：这里是对项目运行进行配置，包括本地运行配置和打包运行配置。如下所示：

'use strict'const path = require('path')

module.exports = {

dev: {assetsSubDirectory: 'static',assetsPublicPath: '/',host: 'localhost',port: 8080,autoOpenBrowser: false,errorOverlay: true,notifyOnErrors: true,poll: false,useEslint: false,showEslintErrorsInOverlay: false,devtool: 'cheap-module-eval-source-map',cacheBusting: true,cssSourceMap: false},

build: {

index: path.resolve(\_\_dirname, '../dist/index.html'),

assetsRoot: path.resolve(\_\_dirname, '../dist'),

assetsSubDirectory: 'static',

assetsPublicPath: '/',

productionSourceMap: false,

devtool: '#source-map',

productionGzip: false,

productionGzipExtensions: ['js', 'css'],

// bundleAnalyzerReport: process.env.npm\_config\_report

bundleAnalyzerReport: true

}

}

其中，dev为本地环境运行配置，build为打包环境运行配置。项目本地运行代码：npm run dev，项目打包线上运行代码：npm run build。

（7）webpack.base.conf：此文件为本项目的配置文件，能够处理本项目个文件的相关配置。

'use strict'const path = require('path')const utils = require('./utils')const configrequire('../config')const vueLoaderConfig = require('./vue-loader.conf')

function resolve (dir) {return path.join(\_\_dirname, '..', dir)}

const createLintingRule = () => ({

test: /\.(js|vue)$/,

loader: 'eslint-loader',

enforce: 'pre',

include: [resolve('src'), resolve('test')],

options: {

formatter: require('eslint-friendly-formatter'),

emitWarning: !config.dev.showEslintErrorsInOverlay

}

})node: { setImmediate: false,dgram: 'empty',fs: 'empty',net: 'empty',tls: 'empty',child\_process: 'empty'}

}

（8）退出登录代码：这里调用了iView的modal弹出框组件，通过判断用户是否确认退出系统来执行相应的功能函数。

signOut(){this.$Modal.confirm({

title: '确定要退出本应用商城系统吗？',okText: '确定退出',

onOk: () => {

this.$store.state.show = true

this.$http.post(this.userLogout, {

userSessionId:this.$store.state.token.userSessionId

}).then(response => {

if(response.result.success == 'true')

{

this.$store.commit('signOut')

this.$router.push({ name: 'signin' })

this.$Message.success('退出登录成功！')

this.$store.state.show = false

}else{

this.$Message.error(response.result.message)

this.$store.state.show = false}

}).catch(error => {

this.$Message.error('出错了，刷新试试呢')

this.$store.state.show = false

})

},});},

下图为本项目前端部分的文件目录图，其中build为构建脚本目录，config为项目配置，src为源码目录，static为静态资源目录，test为测试目录。如下图5-1所示：

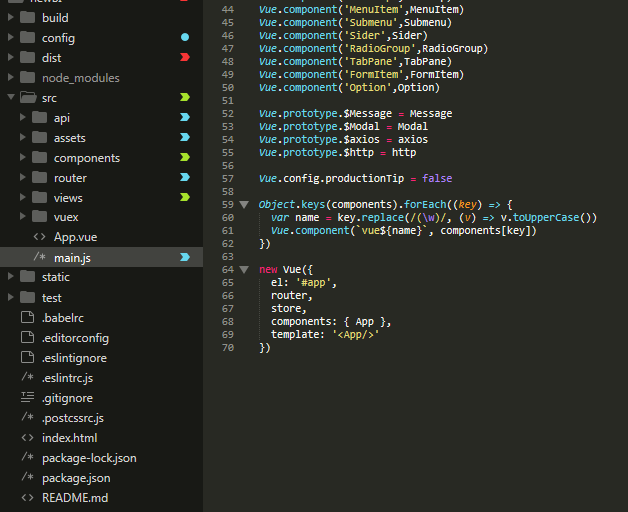


图5-1项目文件目录图

## 核心功能效果

### 首页登录效果

由于是目前是本地node环境npm运行，执行命令npm run dev将项目启动，浏览器跳转<http://localhost:8080>（项目配置的host及port）， 即可来到本系统的登录页面。如图5-2所示：



图5-2首页登录效果图

### 首页注册效果

在登录页面基础上点击注册，即可弹出注册弹窗，注册完成后可以隐藏注册弹窗。如下图5-3所示：



图5-3首页注册效果图

### 本系统的首屏效果

登录成功后会进入本系统的首屏（PC端的首页）。如下图5-4所示：

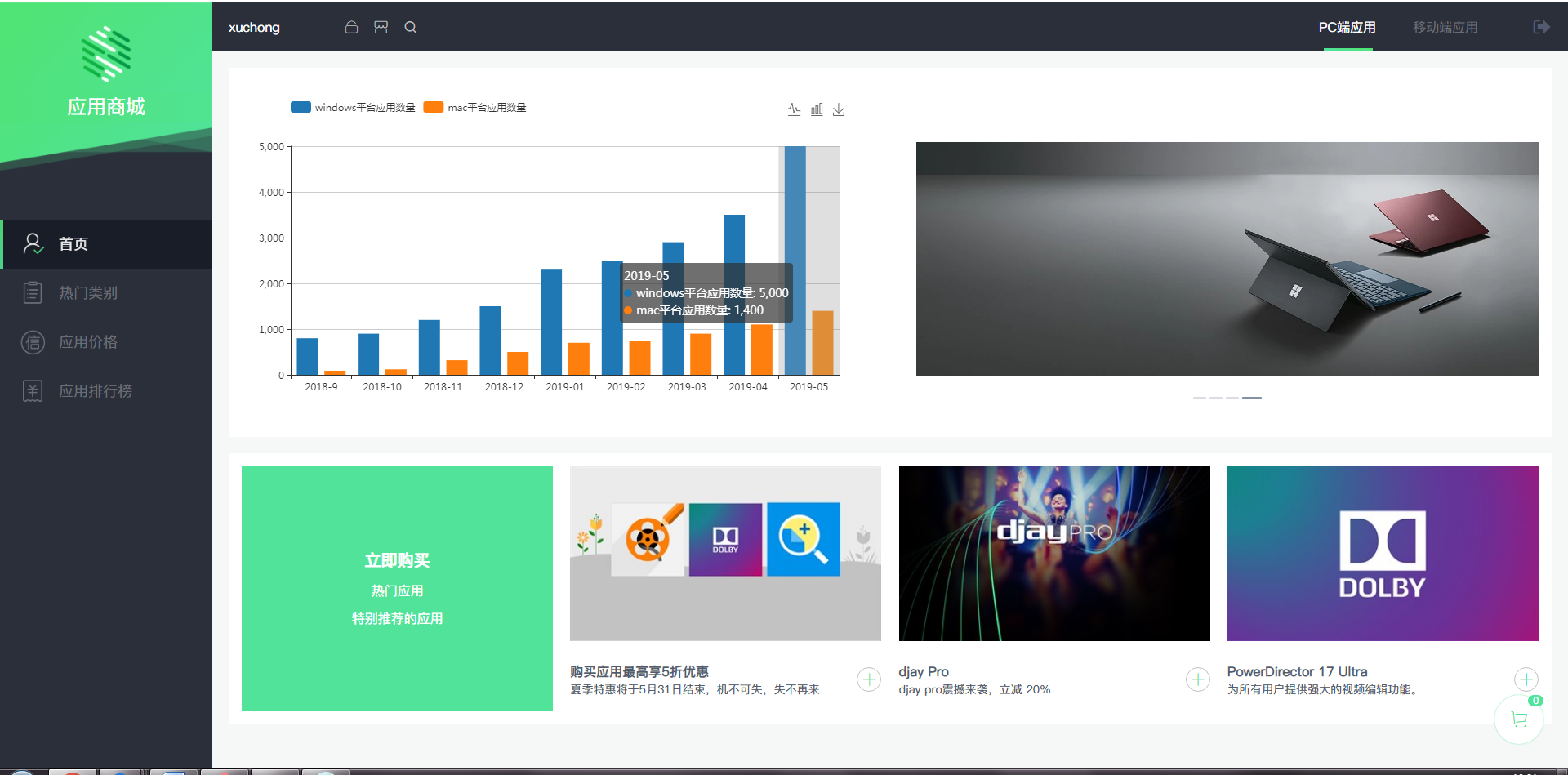


图5-4首屏效果图

### 搜索模块效果

点击放大镜按钮即可出现搜索框，搜索应用名称即可弹出对应的应用信息。如下图5-5所示：

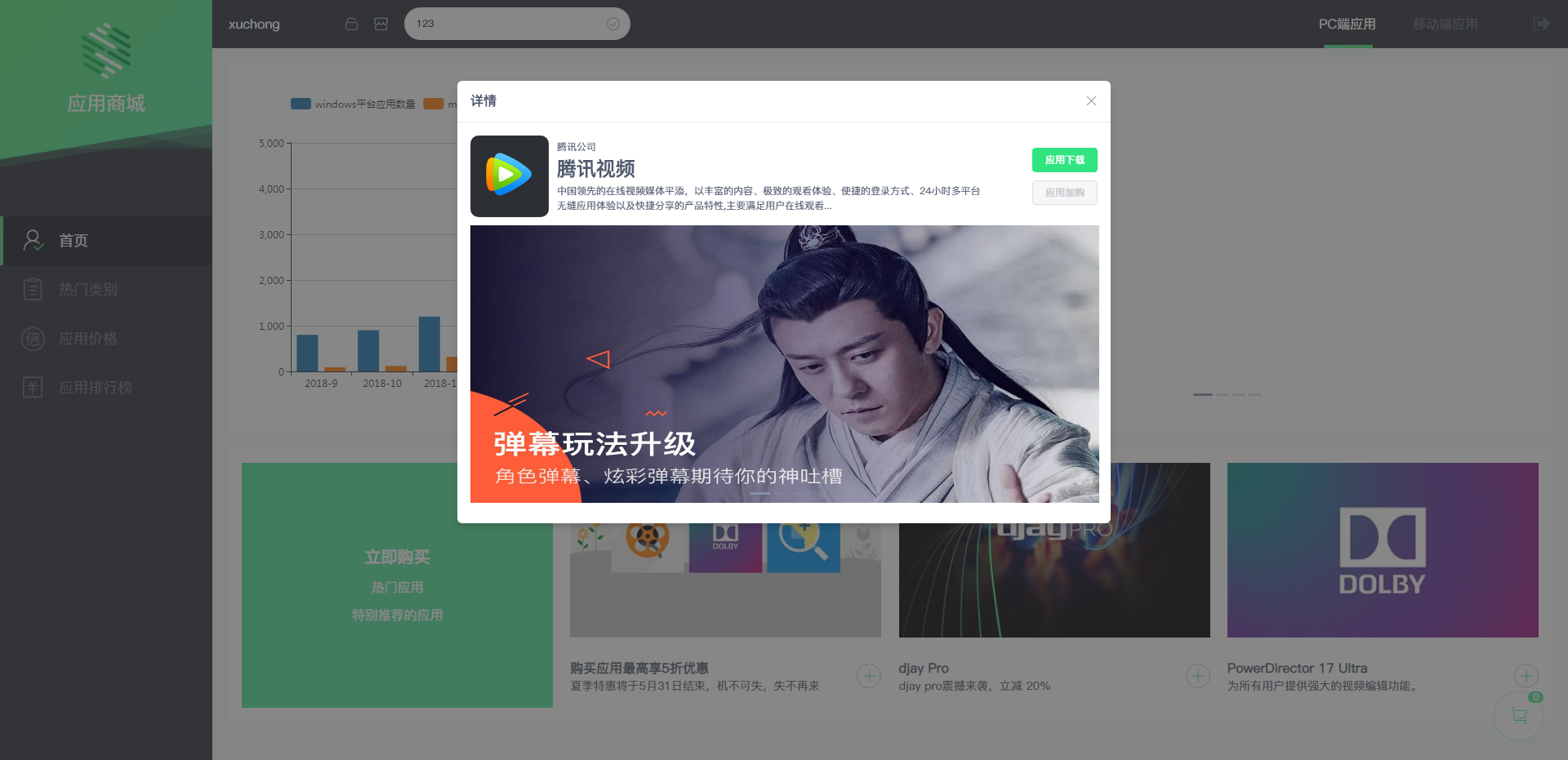


图5-5搜索模块效果图

### 订单模块效果

点击上方购物车图标即可弹出订单模块，订单数据为加购数据生成。如下图5-6所示：



图5-6订单模块效果图

### 修改密码模块效果

这里可以对用户密码进行修改，需要输入旧密码及两次验证密码。如下图5-7所示：

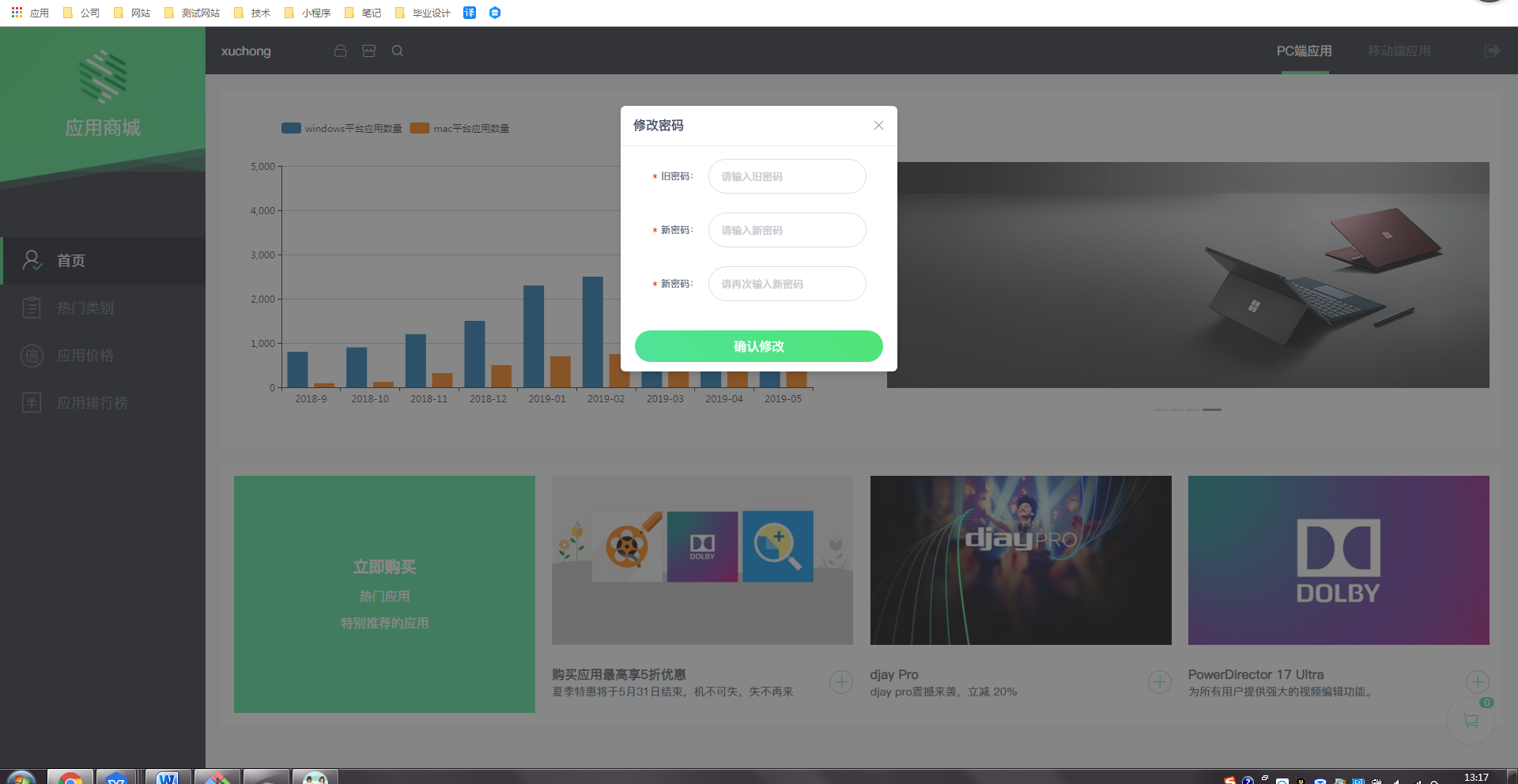


图5-7修改密码模块效果图

### 退出模块效果

这里是本系统的退出模块，调用后台退出接口。退出成功后，将会清除缓存，返回登录页面。效果图如下图5-8所示：

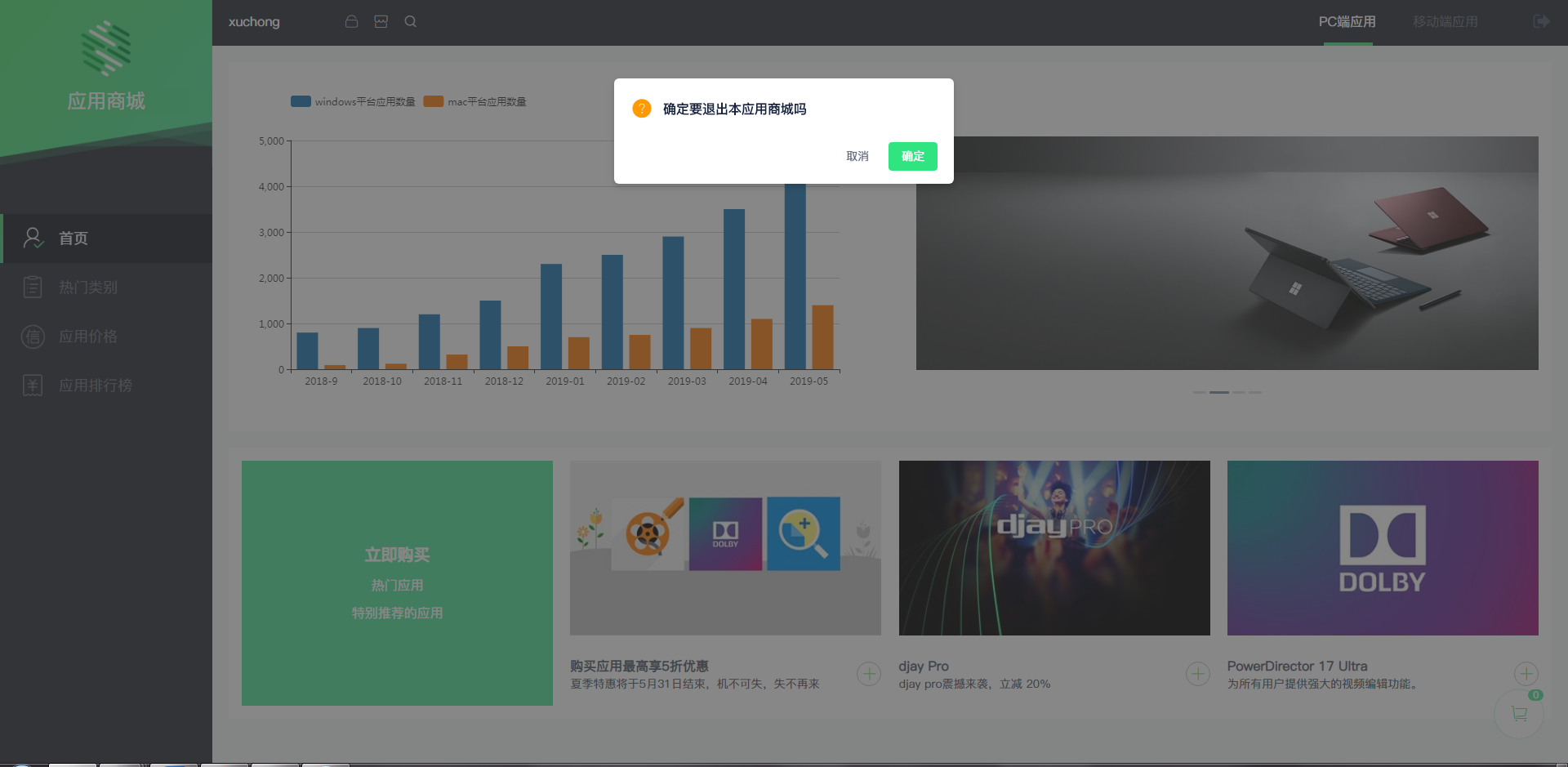


图5-8退出模块效果图

# 测试

## 测试简介

什么是测试？测试时用来检测程序的正确性的必不可少的一步，通过检测测试结果与预期结果之间是否存在差异性来决定下一步对程序需不需要修改以及修改何处。通常测试时在各种极端条件下与平时操作条件下对程序进行各种操作，已检测程序的稳定性，从而得到程序的优化方案和分析报告。

## 测试方法

本项目测试方法主要采用了结构测试和功能测试。

（1）结构测试：通过把一个整体功能模块划分成一段一段具备小功能的代码块，执行相应操作后，根据执行结果与检测执行结果是否符合来测试。

（2）功能测试：主要为在用户角度测试本项目的各功能模块，通过模拟不同的用户操作习惯，以便于获取不同条件下的不同测试结果。

由于项目具有独特性，本项目的测试流程为先从各功能开始，列出本系统各个功能的列表，得到了每个功能应该进行测试的详细计划。在测试各功能过程中，将各种突发情况需要记录下来（比如请求时间太长、操作无反应等等），并需要找出问题发生的原因并记录，如果正常返回结果，则需要与预期结果进行对比，找出其中差异。其次需要测试页面效果，在各浏览器内核下，测试本系统页面展示是否正常，有没有布局错乱，内容溢出等情况发生。测试完毕后，需要进行缺陷审核和验证操作，确认缺陷实际存在后，需要进行消除缺陷操作，针对问题代码进行修改，代码修改完毕则再次进入测试步骤，直到完善为止。

## 测试内容

通过分析本项目实际情况，得到本项目所需要进行测试的功能如下：

登录注册功能测试，应用列表展示功能测试，应用加购功能测试，订单功能测试，应用搜索功能测试，退出功能测试。

由于各浏览器存在差异性，在功能测试完成后，还需要对页面布局进行浏览器测试，本项目前端页面布局效果测试需要在以下浏览器中进行，并检查测试结果是否正常。

火狐浏览器，谷歌浏览器，IE高版本浏览器。

## 测试结果

（1）登录注册功能：在登录注册页面，通过切换登录和注册，然后输入不同情况下的数据进行测试分析，分析结果如下表6-1所示：

表6–1登录注册模块分析表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 功能 | 输入 | 预期 | 结果 |
| 1 | 登录 | 账号密码为空 | 显示用户需要登录 | 合格 |
| 2 | 登录 | 账号错误，密码不为空 | 显示账号未注册 | 合格 |
| 3 | 登录 | 账号正确，密码错误 | 显示密码错误 | 合格 |
| 4 | 登录 | 账号密码正确 | 显示登录成功，跳转 | 合格 |
| 5 | 注册 | 账号密码为空 | 显示不能为空 | 合格 |
| 6 | 注册 | 账号输入已存在的账号 | 显示该用户已注册 | 合格 |
| 7 | 注册 | 账号密码输入正确 | 显示注册成功 | 合格 |

（2）应用列表展示功能：这次测试需要在不同浏览器下检测布局效果，主要从布局整齐性和内容数据显示方面测试，分析结果如下表6-2所示：

表6–2应用列表模块分析表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 浏览器 | 布局效果 | 预期 | 结果 |
| 1 | 谷歌 | 数据正常显示 | 数据正常显示 | 合格 |
| 2 | 火狐 | 数据正常显示 | 数据正常显示 | 合格 |
| 3 | IE | 低版本不支持 | 数据正常显示 | 不合格 |

（3）应用加购功能测试：随机选取部分付费应用，添加到购物车中，加购模块主要测试内容为应用的增删查和订单的生成，分析结果如下表6-3所示：

表6–3应用加购模块分析表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 操作 | 预期 | 结果 |
| 1 | 应用加购 | 应用能够显示在购物车列表中 | 合格 |
| 2 | 购物车应用的删除 | 应用数组删除对应数据 | 合格 |
| 3 | 生成订单 | 购物车清空，生成订单 | 合格 |
| 4 | 清空购物车 | 执行一键删除操作 | 合格 |

（4）订单功能测试：加购操作生成订单后，会在订单列表中展示对应的数据，订单主要测试内容有订单的删除和结算，分析结果如下表6-4所示：

表6–4订单模块分析表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 操作 | 预期 | 结果 |
| 1 | 订单删除 | 删除对应数据 | 合格 |
| 2 | 订单结算 | 能够执行结算操作 | 合格 |
| 3 | 订单清空 | 执行一键删除操作 | 合格 |

（5）应用搜索功能测试：在本系统的顶部菜单栏中会有搜索框存在，输入搜索信息会弹出对应的应用数据展示，分析结果如下表6-5所示：

表6–5搜索模块分析表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 操作 | 预期 | 结果 |
| 1 | 不输入搜索数据 | 不会执行搜索操作 | 合格 |
| 2 | 输入搜索数据，且应用不存在 | 提示应用未找到 | 合格 |
| 3 | 输入搜索数据，且应用存在 | 展示搜索到的应用信息 | 合格 |

（6）退出功能测试：主要测试用户的退出功能是否正常，如下表6-6所示：

表6–6退出模块分析表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 操作 | 预期 | 结果 |
| 1 | 点击退出图标 | 弹出退出提示框 | 合格 |
| 2 | 点击确认退出按钮 | 退出本系统，回到登录页 | 合格 |
| 3 | 点击取消按钮 | 不执行操作，退出框消失 | 合格 |

## 测试总结

程序的开发为本系统的基础，程序的测试为本系统的提升，基础打好了，测试的深度决定了本系统的高度。测试能够检查各个阶段是否接近预期目标，通过对各个阶段结构进行测评，能够尽早的发现并解决问题，从而保证了程序的完整性以及提高了开发的效率，避免后续出Bug影响项目上线。

结束语

本次论文的研究方向是开发一套应用商城网站，通过了严谨的功能模块设计和清晰的代码逻辑视图，本论文向读者勾勒出一套完整的项目开发流程，从而达到本论文预期的效果。目前，线上的应用商城多而繁杂，且部分应用商城下载应用过程中会强制下载捆绑软件，这对于现在快节奏的工作或生活来说是不合适的。为有效解决该问题，在本次应用商城网站的设计中，采用了功能简单但完善的特点，从而简化了用户的操作，提高了网站的效率。

本系统的不足之处有以下几点：

（1）应用数据来源问题：应用的历史版本更替、应用数据的数量、数据库的更新频率这三个方面并没有实现，所以本系统数据来源还有待进一步优化。

（2）应用展示问题：本系统目前只具备展示应用的图片、详情和下载，用户评论、操作视频教程、版本更新情况等等并没有实现。

（3）本系统目前只适用于PC端，移动端是一个更广阔的场地，所以需要开发出一套自适应布局来适应PC端和移动端。

总的来说，该网站在功能完善上还有很多的想法，有些观点和阐述会有部分疏漏，这需要后续进一步的优化。

参考文献

1. BenForta福塔.MySQl必知必会 [M].人民邮电出版社，2009.
2. AlanBeaulieu.SQL学习指南 [M].人民邮电出版社，2010.
3. 扎卡斯.JavaScript高级程序设计 [M].人民邮电出版社，2006.
4. 扎卡斯.高性能JavaScript [M].电子工业出版社，2010.
5. 王晶.利用jQuery操作HTML元素[J]. 农业网络信息, 2008:98-99.
6. Jeremy Keith.JavaScript DOM编程艺术 [M].人民邮电出版社，2007.
7. [弗拉纳根](https://baike.baidu.com/item/%E5%BC%97%E6%8B%89%E7%BA%B3%E6%A0%B9" \t "_blank).JavaScript权威指南 [M].机械工业出版社，2007.
8. Douglas Crockford.JavaScript语言精髓 [M].电子工业出版社，2009.
9. 关伟哲. SQLServer后台数据库安全性管理与应用研究[J]. 电脑知识与技术, 2013(21):4771-4772.
10. Kathy Sierra Bert Bates.Head First Java [M].中国电力出版社，2007.
11. Robert Sedgewick.算法（第四版）[M].人民邮电出版社，2012.
12. 福勒.重构 改善既有代码的设计 [M].人民邮电出版社，2010.
13. Brian Goetz.Java并发编程实战 [M].机械工业出版社，2012.
14. 董英茹.简谈AngularJS在下一代Web开发中的应用[J]. 软件工程师, 2015(5):30-31.
15. 曾汝佳.基于跨平台开发的联网收费运营管理APP[D]. 广东工业大学, 2015(2).

致谢

转眼间，美好的大学生活在不久的未来即将划上句号，在母校生活的时光里以及在课堂上学习的过程中，不仅带给我了知识学问上的充实，也带给了我毕业的不舍和丰收的喜悦。历时两个月左右的毕业设计也即将划上句号，在此我特别感谢给与我无数帮助和建议的惠征老师，没有他的无私指导和谆谆善诱，我无疑将会遇到种种困难，毕业设计过程中我也将会遭遇种种挫折。经历这次毕业设计后，我也具备了掌控项目的能力和全局观念的意识，并且了解和巩固了时下热门的语言和技术，熟练了对项目中所需插件。所以，在此我感谢学校组织的这次毕业设计，让我的知识储备和能力能够更上一层楼。与此同时，我还要真诚的感谢我的任课老师们，没有他们的无私教导，我是无法踏入知识的殿堂的。我还要感谢给与帮助的同学们和技术博客作者们，他们在我遇到技术问题时给与了我正确的解决办法和思路。

语至此，感慨万千。再见，我的大南工！