

**毕业论文**

**题 目 企业智能数据管理系统的设计和实现—前端模块**

**英文题目 Design and realization of enterprise intelligent platform management system -- front-end module**

**学生姓名**： **毛慧君 申请学位门类**： **工学**

**学 号： 201420181014**

**专 业： 软件工程**

**学 院： 软件学院**

**指导教师： 邓薇 职称： 讲师**

**二0 一 八 年 三 月 三十 日**

**作者声明**

本人以信誉郑重声明：所呈交的学位毕业设计（论文），是本人在指导教师指导下由本人独立撰写完成的，没有剽窃、抄袭、造假等违反道德、学术规范和其他侵权行为。文中引用他人的文献、数据、图件、资料均已明确标注出，不包含他人成果及为获得东华理工大学或其他教育机构的学位或证书而使用过的材料。对本设计（论文）的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本毕业设计（论文）引起的法律结果完全由本人承担。

本毕业设计（论文）成果归东华理工大学所有。

特此声明。

毕业设计（论文）作者（签字）：

签字日期： 年 月 日

本人声明：该学位论文是本人指导学生完成的研究成果，已经审阅过论文的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

学位论文指导教师签名：

年 月 日

**企业智能数据管理系统的设计和实现—前端模块**

**毛慧君**

Design and realization of enterprise intelligent platform management system – front-end module

Huijun Mao

**2018年4月15日**

# 摘 要

随着科技的发展，社会的进步，网络信息化时代到来，对数据存储、管理、分析的需求大大提高。我们现在处于信息和数据都极快增长的“大数据”时代，数据是一个企业的核心资产，对于企业发展管理产生深远影响。大数据能增强企业防范风险的能力，使得企业管理决策权成功转变，由被动追求价值向主动增加企业价值转变。再者，企业创造价值的方式也会发生改变，管理决策从单一的中高层管理向员工参与决策转变。因而，企业智能平台管理系统的设计实现势在必行。

本文按照系统的生命周期介绍企业智能平台管理，从设计到实现的全过程。该系统前端使用vue框架，后端为java spring技术，再到更底层涉及到信息碎片化以及深度学习和人工智能算法方面为python技术。该系统包括模型管理、碎片化平台、微服务管理、流程管理、权限管理、数据安全管理等部分。按照权限可划分为用户权限、主题权限、密级权限、角色权限、组权限等，可以在其中配置不同用户的权限身份。在本项目中我主要负责前端开发。

前端部分是使用vue-cli工具搭建vue项目框架进行MVVM模块化开发，异步请求接口调用使用axios，它是基于ES6 Promise技术的封装，再者使用vue-router配置项目路由，vuex便于全局变量的管理，其中搭载了webpack打包机制，轻松将代码打包为js、css文件夹和一个index.html入口文件上线。前端的UI框架使用了iView UI，其样式美观，使用便捷，且自定义配置方便。

关键词：数据管理；Vue.js；vue-router； axios

# ABSTRACT

With the development of science and technology, the development of society and the arrival of network information age, the demand for data storage, management and analysis has been greatly improved.We are now in the era of "big data", where information and data are growing extremely fast. Data is the core asset of an enterprise, which has a far-reaching impact on enterprise development management.The big data can enhance the enterprise's ability to prevent risks, and make the enterprise management decision-making power change successfully, from the passive pursuit of value to the initiative to increase the enterprise value change.In addition, the way that enterprises create value will change, and management decisions will change from single middle and senior management to employees.Therefore, the design of enterprise intelligent platform management system is imperative.

This paper introduces the management of enterprise intelligent platform according to the life cycle of the system, from design to implementation.The front end of the system USES the vue framework, the back end is Java spring technology, and the lower level involves information fragmentation and deep learning and artificial intelligence algorithms for python technology.The system includes model management, fragmentation platform, micro-service management, process management, authority management, data security management and so on.According to the permission can be divided into user rights, subject permissions, confidentiality level permissions, role permissions, group permissions, and so on, and can configure different user's authorization status in it.In this project, I am mainly responsible for front-end development.

Front part is to use vue - cli tool set up vue MVVM modular development project framework, an asynchronous request use axios interface call, it is based on ES6 Promise encapsulation technology, moreover to use vue - the router configuration items routing, vuex facilitates the management of the global variables, which carried a webpack packaging mechanism, easy to pack code and an index for js, CSS folder. The HTML entry files online.The front-end UI framework USES the iView UI, which is aesthetically pleasing, easy to use, and easy to customize.

**KEY WORDS:** Residential Property Management; Vue.js; vue-router; axios

# 目 录

[1. 绪论 1](#_Toc483865687)

[2. 选题的背景、目的及意义 2](#_Toc483865688)

[2.1 选题的背景 2](#_Toc483865689)

[2.2 选题的目的 2](#_Toc483865690)

[2.3 选题的意义 2](#_Toc483865691)

[3. 系统分析 4](#_Toc483865692)

[3.1 需求分析 4](#_Toc483865693)

[3.1.1 系统主要目标 4](#_Toc483865694)

[3.1.2 系统功能分析 4](#_Toc483865695)

[3.2 可行性分析 4](#_Toc483865696)

[3.2.1 技术可行性 4](#_Toc483865697)

[3.2.2 经济可行性 5](#_Toc483865698)

[3.2.3 操作可行性 5](#_Toc483865699)

[3.3 关键技术分析 5](#_Toc483865700)

[3.3.1 Vue.js技术 5](#_Toc483865701)

[3.3.2 axios技术 5](#_Toc483865702)

[3.3.3 webpack技术 6](#_Toc483865702)

[3.3.4 iview UI技术 6](#_Toc483865702)

[4. 系统设计 7](#_Toc483865703)

[4.1 系统的基本功能结构 7](#_Toc483865704)

[4.2 业务流程图 7](#_Toc483865705)

[4.3 系统用例图 8](#_Toc483865706)

[4.4 系统数据库设计 9](#_Toc483865707)

[4.4.1 E-R模型 9](#_Toc483865708)

[4.4.2 数据库的逻辑设计 11](#_Toc483865709)

[5. 系统实现过程 13](#_Toc483865710)

[5.1 系统设置 13](#_Toc483865711)

[5.1.1 登录和注册 13](#_Toc483865714)

[5.1.2 登出 14](#_Toc483865714)

[5.1.3 修改密码 15](#_Toc483865714)

[5.2 模型管理 15](#_Toc483865712)

[5.2.1 应用列表 15](#_Toc483865714)

[5.2.2 添加、编辑应用 16](#_Toc483865714)

[5.2.3 删除应用 17](#_Toc483865714)

[5.2.4 场景列表 18](#_Toc483865714)

[5.2.5 添加、编辑场景 19](#_Toc483865714)

[5.2.6 删除场景 21](#_Toc483865714)

[5.2.7 词库列表 22](#_Toc483865714)

[5.2.8 添加、编辑词库 23](#_Toc483865714)

[5.2.9 删除词库 24](#_Toc483865714)

[5.2.10 模型训练 25](#_Toc483865714)

[5.3 微服务管理 26](#_Toc483865713)

[5.3.1 微服务管理列表 26](#_Toc483865714)

[5.3.2 添加、编辑微服务 27](#_Toc483865715)

[5.3.3 删除微服务 28](#_Toc483865716)

[5.4 流程管理 29](#_Toc483865722)

[5.4.1 流程管理列表 29](#_Toc483865723)

[5.4.2 添加、编辑流程 30](#_Toc483865724)

[5.4.3 删除流程 31](#_Toc483865725)

[6. 系统测试和维护 33](#_Toc483865727)

[6.1 系统测试 33](#_Toc483865728)

[6.1.1 系统测试的作用和意义 33](#_Toc483865729)

[6.1.2 系统测试过程和结果 33](#_Toc483865730)

[6.2 系统维护 35](#_Toc483865731)

[结 论 37](#_Toc483865732)

[致 谢 38](#_Toc483865733)

[参考文献 39](#_Toc483865734)

# 绪论

现今，大数据时代来临，数据作为特殊的“资产”，人们更多的是根据长期来的历史数据总结出其规律性，并查找之前的不足和缺陷。并希望在之后的工作中对之前的不足进行改善和弥补。但大数据平台最重要的功能是预测未来，即在对数据的分析中及时针对某方面做出整改。从而降低企业生产运行风险。假使竞争企业可以对行业市场进行预测，但自己却不行，那么企业将失去未来。由于企业系统不能适应环境变化从而将存在风险。而大数据的优点主要是引导价值，促使企业价值向好的方向发展，其价值主要体现在企业管理组织架构中，企业原有组织形式是单一的为企业价值而存在和建立的。大数据是为生产者和决策之以及市场建立连接形成多元化的隐性联系。大数据的建立可以有效从根本上引导管理者和决策者改变传统的决策方式。

由于企业改变了创造价值的方式，在大数据前，主要是依靠业务和因果思维形成管理方式，这是一种先找问题在找解决问题的方法，而通过数据驱动模式，可以通过大数据技术使企业决策之直接看到解决问题的方法，这就直接排除了决策者个人主观判断对问题的影响，决定了企业的决策者思维。大数据技术中蕴藏着丰富的数据信息资源，这能切实为企业带来巨额利益和经济产值。因此，利用好信息资源就要进一步加强大数据技术的完整性、全面性和时效性。

对大数据的处理分析正成为新一代信息技术融合应用的结点。移动互联网、物联网、社交网络、数字家庭、电子商务等是新一代信息技术的应用形态，这些应用不断产生大数据。云计算为这些海量、多样化的大数据提供存储和运算平台。通过对不同来源数据的管理、处理、分析与优化，将结果反馈到上述应用中，将创造出巨大的经济和社会价值。

# 2. 选题的背景、目的及意义

## 2.1 选题的背景

大数据产业在我国已经有了几年的发展，而从整体看，我国信息化程度的确扔不是特别完善，目前还处于探索阶段，若要想进一步发展需要经历几年时间。

随着信息技术飞速发展, 各行各业积累的数据都呈现出爆炸式的快速增长趋势, 我们已不知不觉进入大数据时代. 显然大数据在多个领域都有广阔的应用前景, 已经成为非常重要的战略资源, 存储、管理和分析大数据也已经成为工业界和学术界高度关注的热点。传输、处理、收集、存储、大数据的目的是利用大数据, 而做到有效地利用大数据。

在各种类型的不同数据中，大数据技术即快速获得有价值信息的能力。明白这一点十分重要，使得技术具备走向众多企业的潜力也很大程度上是有这一点决定的。大数据的4个特点有四个层面：第一点，数据体量巨大。从TB级别，一直跃升到PB级别；第二，数据类型众多。例如网络、图片、地理位置、日志、视频信息等等。第三，价值密度低。比如说视频，连续不间断监控过程中，也许有用的数据仅仅有一两秒。第四，处理速度极快。现今，信息时代正渐渐向数据时代转变，数据开始变得愈发重要，它能够反映出公司的经营情况，为公司的经营和决策提供准确的参考依据。而数据平台是连接数据和企业的媒介，可以将这些数据通过一定的规格，清晰地展现给领导。

当前的大数据发展非常复杂。尤其是当用户需要挑选适合自身的大数据技术时，往往是踌躇忐忑的，既要考虑这个技术是不是真正解决了自身的问题，又要考虑该技术是否得到了积极快速的发展，还要忧心是不是还有新的技术加入很多能够提升的新功能。事实上，当不同的技术之间差别不是很大的时候，用户是可以随便选某一种技术以所用。但是纵观当今大数据产业的发展，在未来，很大机会是可能出现更加集中化、一体化的技术。

## 2.2 选题的目的

正是由于现阶段大数据平台对于企业的重要意义，构建一个企业智能数据管理平台具有重大意义。该数据管理平台致力于使得公司形象大幅提升，建立起现代化信息管理体制。能够使企业内部各部门、办事机构的业务流程更加优化高效，再造业务规范，对重点业务实行全面质量监控。

  数据可以给企业带来创收新方式，从而成为其商业模式的一部分。企业可以通过多种途径将数据货币化。将数据出售给客户和第三方，来创建新的收入来源。要从中赚取收入，就是要使企业确保生成的数据能提供增值服务给消费者。

大数据的目的还在于帮助企业留住现有员工，确定最佳招聘渠道，并选择最佳人选。企业有助于分析个人发展，生产力数据，旷工数据等，以获得招聘部门和员工管理的依据。这些数据很大部分可以免费使用。它可以用来改善行业的业务运作。实际上，只要是生成数据的业务流程都可以优化业务从而提高效率。企业必须分析竞争对手的商店/网站，或获知业界消息，以了解他们的运营方式。如今，企业甚至足不出户就可以知道对手在做什么。因为大数据使其财务和其他重要信息随时可用。

开发决策支持系统，为企业决策层提供图形化、报表化的市场分析数据，预测未来的公司业务发展、客户需求发展、市场发展，可以方便地实现与关系部门的数据共享和交换。建设企业相关的INTRANET/INTERNET网络平台。通过INTERNET实现全天候实时服务，充分满足客户的各种需求，全面提升客户服务水平，加强与客户的紧密度，将应用企业塑造成国际性大企业。从而降低企业运作成本，提高公司的运作效率，大幅拓展业务，争取企业利润最大化，进一步提高企业的竞争力。

## 2.3 选题的意义

数据化管理是科学管理的基础。科学管理的目标是目标明确、决策准确、措施有效、执行有力。数据化管理是将业务工作中的基本状况，通过翔实的数据直观地展现，并通过适当地分析，明确经营基本状况，发现业务工作中的不足之处，为管理者提供准确的决策依据，促进管理层进行有针对性地改进和有效地决策，是科学管理的基础。数据化管理是科学领导的参考。领导学认为领导的艺术与方法是达成领导效能与发展的关键因素。数据化管理是优秀的管理方法之一。

完善的数据化管理能够明确指出下属业务工作中存在的各类问题，以实事求是的方法并辅之于其他的管理手段，能够有效地指导若干下属开展工作，能够根据问题的严重性与重要性进行有针对性地改善，促进团队的整体进步，从而实现领导效能，是科学领导的有效参考。数据化管理是企业管理改进的关键。优秀的企业管理应该具备完善的运营数据分析体系。一切企业活动，最终都以数据最为参考， 达成一定的数据指标，循环改进，持续发展。数据化管理存在于企业的每个环节，通过参考经营数据管理的企业体制是确保企业良性发展的关键。数据化管理是一种全新的管理方法，其推广和运用可以促进民族企业的发展，增强国际竞争力。

各类数据在企业生产经营中起着至关重要的作用，数据是企业，生产，经营，战略，等等，几乎所有的经营活动所依赖的，不可或缺的信息。数据就犹如企业经营者的眼睛一样，通过数据可以反映出经营的问题，就犹如舵手依赖导航一样。数据类型可以分为：财务数据，生产数据，销售数据，市场数据，人力资源数据，等等，各种各样的数据，起到的作用也是不一样的。例如可以反映销售状况，通过不同的时间，市场环境，好坏，趋势，等等变量，反映经营状况，生产状况，企业经营者要根据数据做判断，来指导销售，生产，以及库存，制定生产计划等等。例如：去年焦炭企业连续亏损，企业就要根据市场数据做生产调整，来压缩产能，换取市场价格回升。等等。财务数据，生产数据，等等的作用都是必须的，并且都是至关重要的，数据的缺失，或者统计不出来。

# 3. 系统分析

## 3.1 需求分析

所谓"需求分析"，是指对要解决的问题进行详细的分析，弄清楚问题的要求，包括需要输入什么数据，要得到什么结果，最后应输出什么。可以说，“需求分析”就是确定要计算机“做什么”。 需求分析是一项重要的工作，也是最困难的工作。

作为一个企业智能数据管理系统需要切实帮助企业分析管理大数据，通过对数据的管理对过往不足进行反省也对未来形势进行预测，从而使得企业朝着更好地方向蓬勃发展。

### 3.1.1 系统主要目标

实现一个企业智能数据管理系统的目标，一方面在于掌握庞大的数据信息。另一方面则在于大数据与产业的结合，对海量数据进行专业化处理，实现数据的“增值”和应用的“价值”。随着资源整合和产业链拓展，大数据应用正在成为新的经济增长点，逐渐在智能家居，智能制造、智慧出行、智慧医疗、互联网金融等应用中找到用武之地。

### 3.1.2 系统功能分析

通过对数据管理系统的了解分析，企业智能数据管理系统主要包括以下几点：

（1）模型管理：包括应用、场景、词库内容的新建、修改和删除等，并能对需要的有用数据进行有选择的训练，实现算法的深度学习。

（2）微服务管理：用户可以根据需要添加新的微服务，也可以查看、修改微服务详情信息，如觉得不必要也可将该微服务删除。

（3）流程管理：主要包括流程名称、描述、问题、肯定动作和否定动作。可对流程进行修改、添加和删除。它管理一整个动作的流程。

（4）权限管理：主要包括用户权限、主题权限、密集权限、角色权限和组权限等部分。管理员可以配置不同用户的各种相关权限，控制不同身份用户对系统的访问。

## 3.2 可行性分析

企业智能数据管理系统是当今企业发展不可缺少的一部分。可以从技术可行性、经济可行性和操作可行性三方面展开论述。

### 3.2.1 技术可行性

根据该系统目标来衡量所需的技术是否具备，一般可从硬软件的性能要求、环境条件、操作人员水平和数量等方面去考虑和分析。

软件方面，选择了微软的VS Code作为前端开发，界面简约美观，使用友好，搭载许多相关插件，下载方便。

硬件方面，选用硬盘存储空间较大，运行内存也较大的电脑，使得系统运行时更加流畅，不卡顿。在软件和硬件方面都是可行的。

### 3.2.2 经济可行性

经济可行性是指可以使用的资源的可能性。在这里主要从成本、效益两方面进行分析。

从成本看，企业主要成本在于开发人员的薪酬方面，且该系统开发时间大致为三个月，企业完全可以负担起这个成本，即是成本是可行的。

从效益看，该系统大大提高了企业的办事效率，为企业提供了管理、分析数据的平台，用作评估企业运营情况和效率，还能预测之后的发展形势等，在长远来看效益是远大于成本的，即效益是可行的。

### 3.2.3 操作可行性

该系统界面清晰简约，操作简单易用，对用户操作方面十分友好，同时也有相关的严密谨慎的使用文档，和默认案例，使得新用户也能很快上手不需要门槛。

综上所述，通过技术可行性、经济可行性、安全可行性、操作可行性等方面分析，得出本系统开发是可行的这一结论。

## 3.3 关键技术分析

### 3.3.1 Vue.js技术

Vue.js是一套渐进式框架，用以构建用户界面。主要采用自底向上增量开发的设计。Vue的核心库只关注视图层，它不仅容易学习，还便于与第三方库或已有项目。它最大的特点就是双向数据绑定。

主要是先实现一个数据监听器Observer，监听所有数据的对象，如果数据有变化即可拿到新值并通知订阅者，再者实现一个指令解析器Compile，来扫描和解析每个元素的指令，根据指令模板替换数据，以及绑定相应的更新函数。而实现一个Watcher，是连接Observer和Compile的桥梁，能够收到并订阅每个属性变化的通知，执行指令绑定的相应回调函数，从而更新视图。最后mvvm入口函数，整合以上三者。

（1）vue-cli ：vue-cli是Vue.js官方出的脚手架命令行工具，我们可以用它快速搭建好一个Vue.js项目，它主要用来初始化项目，可以使用官方模板，也可以使用自定义模板生成新项目，而且在新版本中，vue-cli新增了build命令，实现了零配置启动Vue.js应用。

（2）vue-router：vue-router是基于Vue.js的路由配置，通过vue-router可以轻松实现项目路由的各种复杂配置。包括可以设置元数据、钩子函数等等。也避免了之前会出现的回调地狱的问题。

（3）vuex：Vuex 是一个专为 Vue.js 应用程序开发的状态管理模式。它采用集中式存储管理应用的所有组件的状态，并以相应的规则保证状态以一种可预测的方式发生变化。Vuex 也集成到 Vue 的官方调试工具 devtools extension，提供了诸如零配置的 time-travel 调试、状态快照导入导出等高级调试功能。

### 3.3.2 axios技术

Axios 是一个基于 promise 的 HTTP 库，可以用在浏览器和 node.js 中。使用ES6新的promise技术实现异步数据请求，调用后端接口，代码简洁优雅易维护，和Vue.js配套使用，效果更佳。有成熟的文档，方便开发。

### 3.3.3 webpack技术

本质上说，webpack 是一个现代 JavaScript 程序的静态模块打包器。当 webpack 处理应用程序时，它会以递归方式构建一个依赖关系图，其中包含应用程序需要的各个模块，最后将所有这些模块打包成一个或多个 bundle。它支持高度配置，主要包括四个部分：入口(entry)、输出(output)、loader和插件(plugins)。

### 3.3.4 iView UI组件

iView是一套基于Vue.js的开源组件库，主要用于PC界面的中后台产品。它高质量，功能丰富，有友好的API可以自由灵活的使用空间。UI细致漂亮，文档详细易读，并且主题也支持高度定制。十分灵活，对于开发者也十分友好。

# 4. 系统设计

## 4.1 系统的基本功能结构

系统主要包括模型管理、微服务管理、流程管理和权限管理等方面。模型管理包括应用、场景、词库和模型训练；微服务管理包括微服务的增删改；流程管理包括流程的增加、编辑、删除；权限管理包括用户权限、主题权限、密集权限、角色权限和组权限等。

图4-1 系统的功能结构图

## 4.2 业务流程图

用户先输入用户名和密码，验证登录成功后即可进入应用列表首页，在首页可以选择某一项直接跳转到该应用下的场景，也可以选择编辑或删除场景；场景列表下也可以编辑和删除操作等，词库也是同理。模型训练页面可以选择需要训练的应用选择训练进行训练，然后刷新页面可看到训练好的列表。而进入流程管理和微服务管理中进行浏览、编辑、添加和删除的操作，最后是权限管理可以对不同用户进行权限的管理。下面用业务流程图来说明：

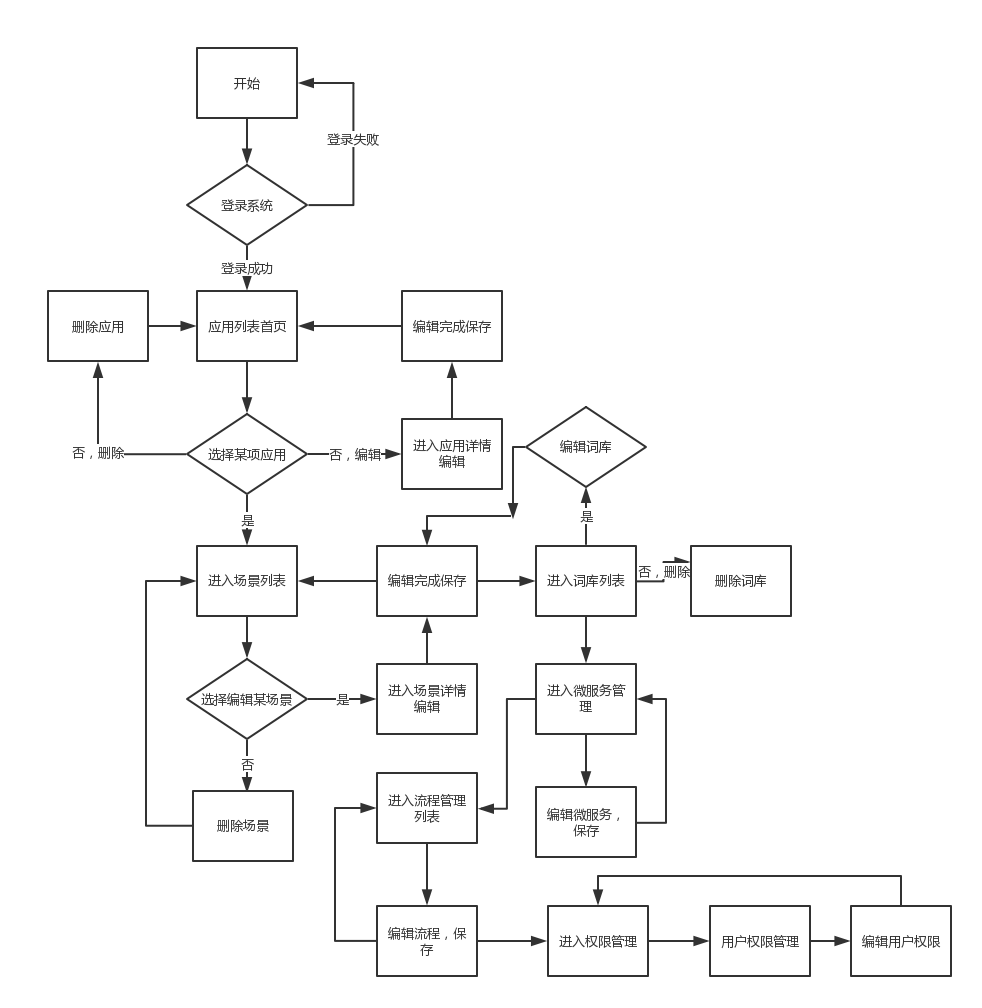


图4-2 业务流程图

## 4.3 系统用例图

登录成功的用户可进行对应用、场景、词库等的编辑操作，具体情况如下图系统用例图所示：

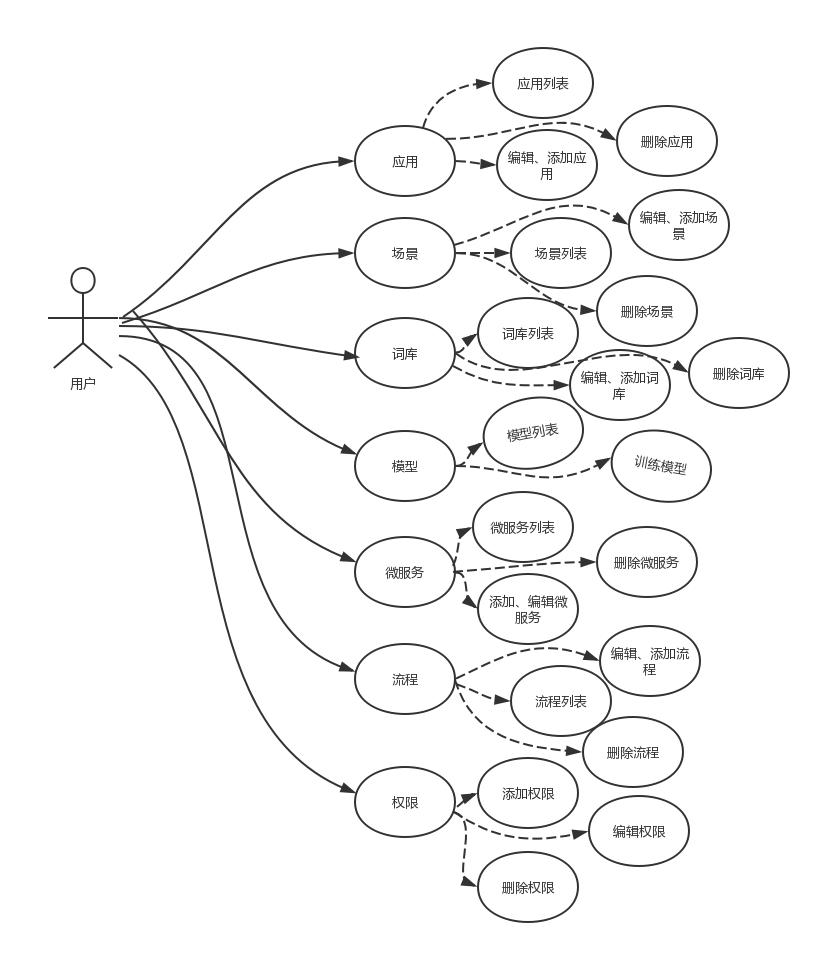


图 4-3 系统用例图

## 4.4 系统数据库设计

数据库技术是管理信息系统、办公自动化系统、决策支持系统的各类信息系统的核心部分，是进行科学研究和决策管理的重要技术手段。

### 4.4.1 E-R模型

E-R图是用来描述某一组织（单位）的概念模型，提供了表示实体、属性和联系的方法。构成E-R图的基本要素是实体、属性和关系。实体是指客观存在并可相互区分的事件；属性指实体所具有的每一个特征。利用E-R图，可以很直观的表现实体与各实体间的联系。

企业智能数据管理系统实现了用户和应用、词库、场景、流程等等实体之间的关系，它们有多对多、多对一等不同关系，具体如图所示：

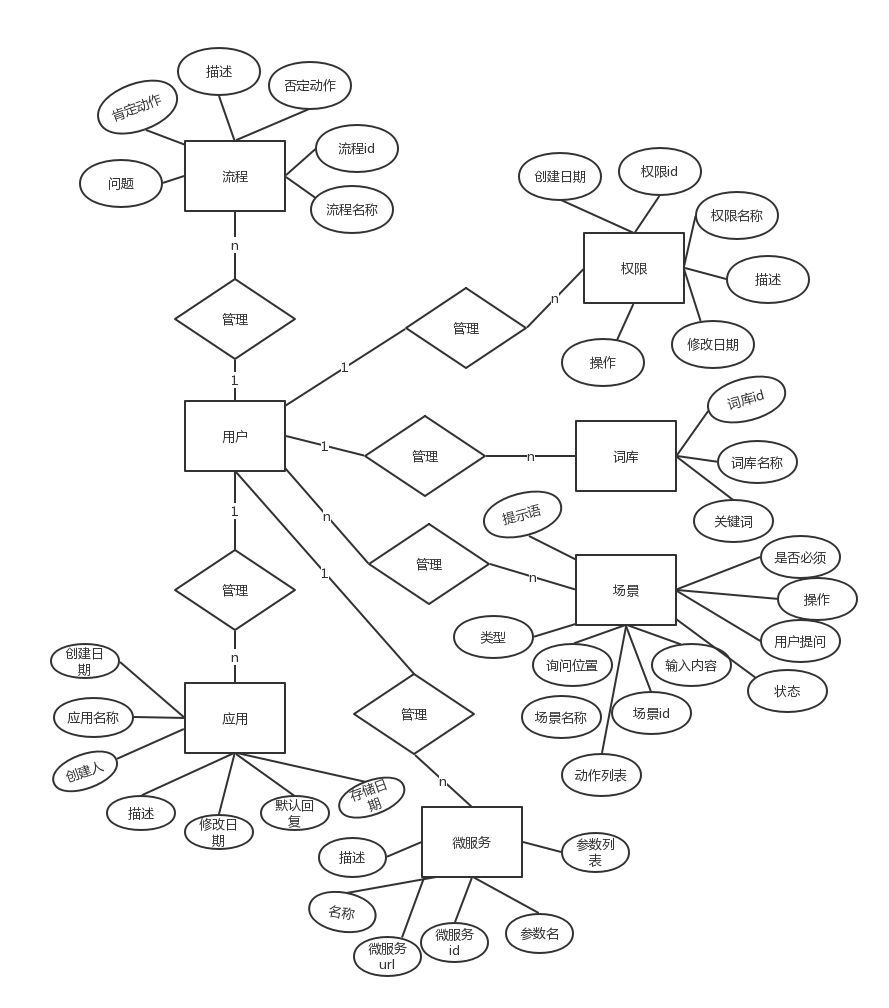


图4-5 系统E-R图

### 4.4.2 数据库的逻辑设计

数据库的逻辑设计就是将上文中所涉及的E-R图转化为SQL数据库程序所支持的数据逻辑模型。在企业智能数据管理系统中，建立的几个主要数据表有，如下所示：

（1）用户表:记录登录用户信息，以用户id为主键，自增长。

表4-1 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| id | int | Not null | 用户id |
| username | varchar | Not null | 用户名 |
| password | varchar | Not null | 密码 |
| token | varchar | Not null | 标志 |

（2）应用详情表：记录用户创建的应用详情，id自动生成且唯一。

表4-2 应用详情表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| id | int | Not null | 应用id |
| CreateBy | varchar | Null | 创建者 |
| defReply | varchar | Null | 默认回复 |
| describe | varchar | Null | 描述 |
| name | varchar | Not null | 应用名称 |
| storage | varchar | Null | 存储方式 |
| updateDate | varchar | Null | 更新时间 |
| createDate | datetime | Null | 创建时间 |

（3）场景信息表：主要是记录用户在应用下创建的场景信息，id为主键唯一且自增长。

表4-3 场景信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| id | int | Not null | 场景id |
| intentName | varchar | Not null | 场景名称 |
| flag | int | Null | 标志 |
| actionName | varchar | Null | 动作名称 |
| check | int | Null | 是否检查 |
| input | varchar | Null | 输入名称 |
| message | varchar | Null | 关键词 |

（4）词库信息表：主要用于记录用户在应用下创建的词库信息，id为主键唯一且自增长。

表4-4 词库信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| Id | int | Not null | 词库id |
| appId | int | Not null | 应用id |
| Name | varchar | Not null | 词库名称 |
| synoymyFlag | int | Null | 是否有同义词 |
| wordList | array | Null | 关键词列表 |

（5）微服务管理表：用来记录用户创建的微服务的详细信息，以id为主键，标志整张表。

表4-5 微服务管理表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| Id | int | Not null | 微服务id |
| Describe | varchar | Null | 微服务描述 |
| Interface | varchar | Null | 微服务接口 |
| url | varchar | Null | 微服务url |
| Updatetime | varchar | null | 修改日期 |
| name | varchar | Not null | 微服务名称 |

（6）流程管理表：记录用户创建的流程描述、问题、名称等内容，以流程id为主键，不得为空。

表4-6 流程管理表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| Id | int | Not null | 流程id |
| Describe | varchar | Null | 流程描述 |
| Ask | varchar | Null | 问题 |
| yAction | varchar | Null | 肯定动作 |
| nAction | varchar | null | 否定动作 |
| updateDate | varchar | Not null | 更新时间 |

（7）用户权限表：设置用户相关权限，每个权限有单读的id，标识其唯一性。

表 4-7 用户权限表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 约束 | 字段说明 |
| Id | int | Not null | 权限id |
| Name | varchar | Not null | 权限名称 |
| Office\_id | varchar | Null | 部门id |
| Create\_by | varchar | Null | 创建人 |
| Phone | varchar | null | 手机号 |
| dept | varchar | null | 部门名称 |
| email | varchar | null | 电子邮件 |

# 5. 系统实现过程

## 5.1 系统设置

系统设置方面主要包括用户的登录、注册以及修改和退出等。

## 5.1.1 登录和注册

用户首次登录之前必须先进行注册。注册成功后，后台会返回一个唯一token，通过这个token，在用户再次登录时后台进行校验成功登录。前端主要是在进行注册请求之后查看状态，如果正常则告知用户注册成功并请他进行登录。而用户在登录成功后将vuex的setUserInfo状态更新为当前用户登录信息。

核心代码：

toRegister () { //注册

this.$axios.post('user/regist', { username: this.username, password: this.password }).then(response => {

console.log(response)

})

},

toLogin () { // 登录

this.$axios.post('login', { username: this.username, password: this.password }).then(response => {

console.log(response)

if (response.status.code === '200') {

this.$store.dispatch('toggleLoginStatus', true)

const resData = response.data

const loginInfo = {

auth: resData.menuList,

token: resData.token.token,

username: resData.username

}

this.$store.dispatch('setUserInfo', loginInfo)

this.$store.commit('SET\_FIRST\_ROUTE', resData.menuList[0])

this.$router.push({ name: this.firstRoute })

}

})

}

页面截图：

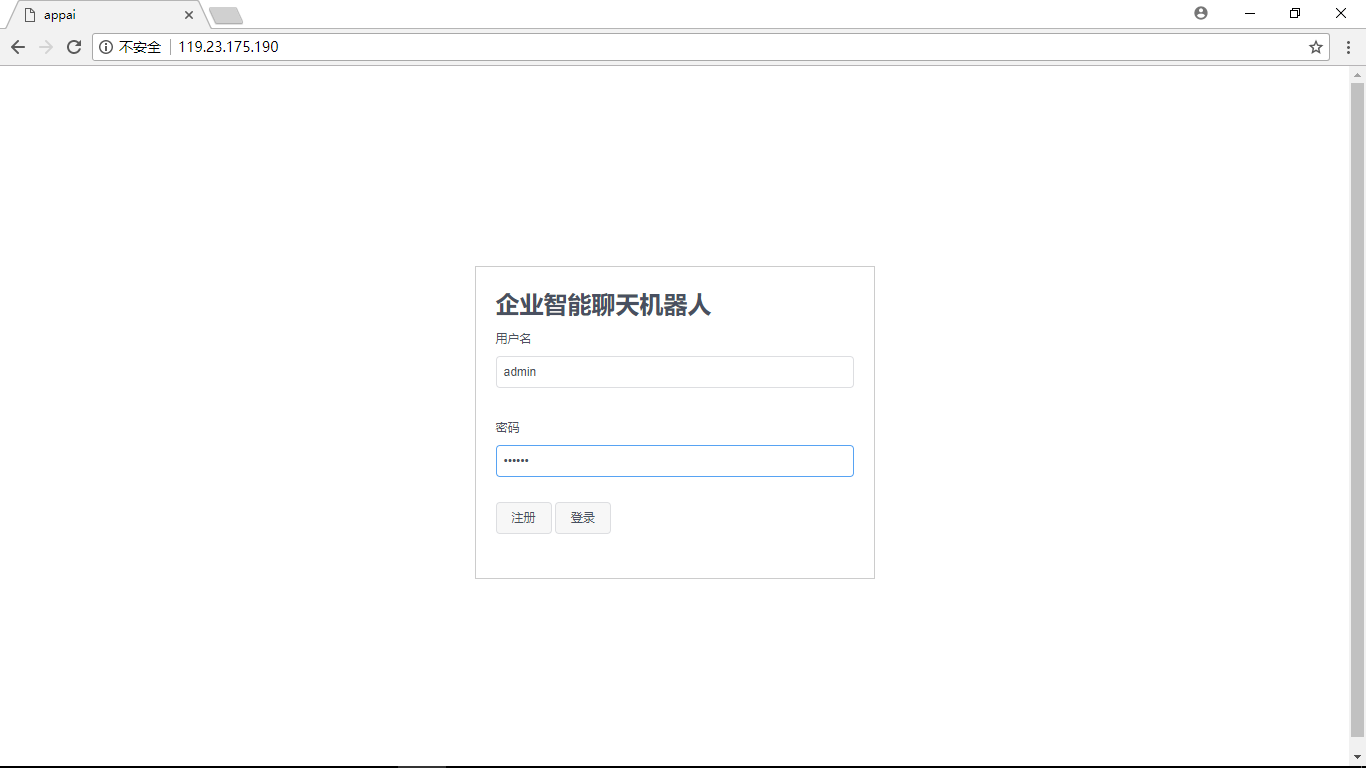


图5-1 登录/注册页面

## 5.1.2 登出

在用户登出后，判断登出成功后就将保存在浏览器本地的信息全部清空。即sessionStorage和localStorage都清除干净，在下次登录时再重新保存更新。并且push到一个新的路由，即返回登录页面。

核心代码：

logout () { // 退出

this.$axios.post('user/logout').then(response => {

if (response.status.code === '200') {

this.$store.dispatch('logoutClear')

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

}

})

}

页面截图：

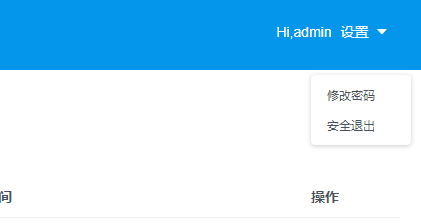


图5-2 登出

## 5.1.3 修改密码

修改密码中需要输入原密码和新设置的密码。前端先验证原密码与新密码不得一致，若验证通过后再调用后台接口传递这两个值。若修改密码成功则跳转登录页面重新输入新密码登录。

核心代码：

modifyPwd () { // 修改密码

let data = {

password: this.password,

newPassword: this.newPassword

}

this.$axios.post('user/chg-passwd', data).then(response => {

console.log(response)

})

}

页面截图：

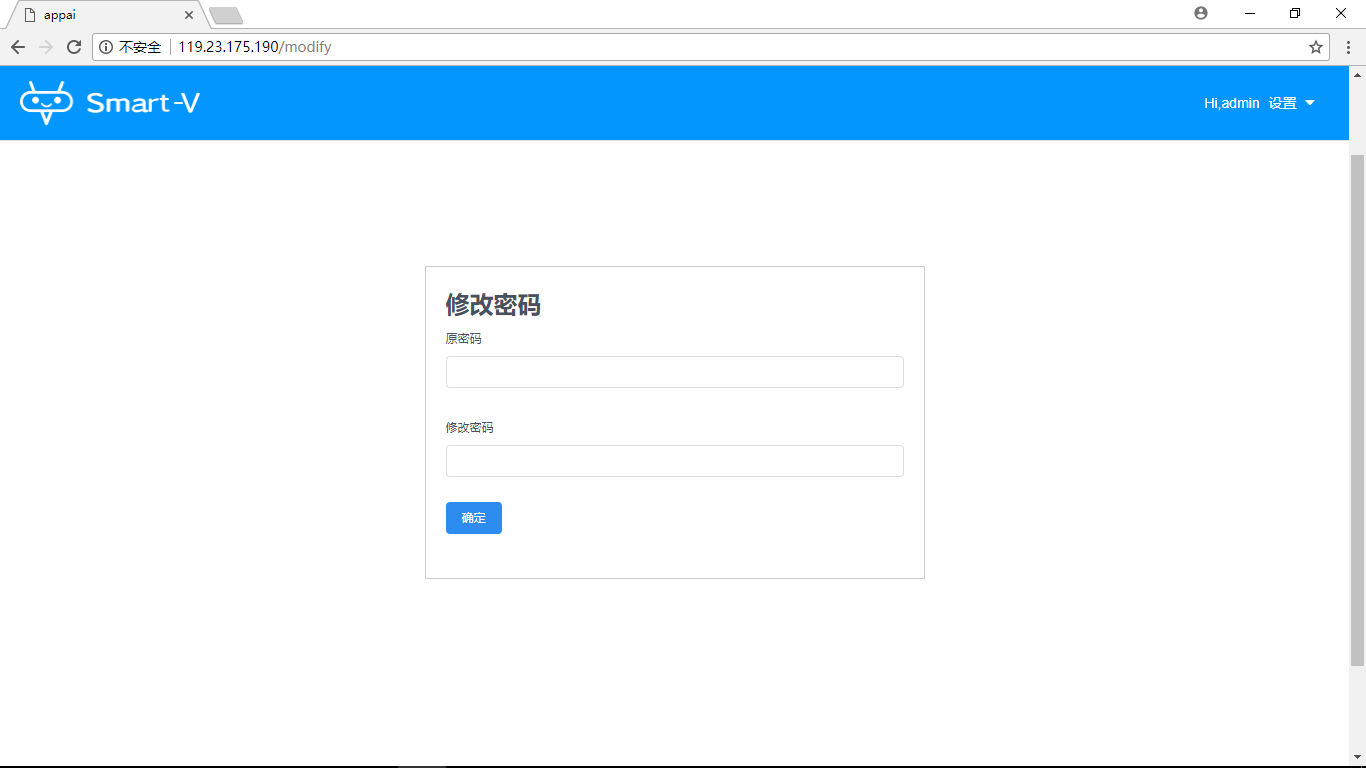


图5-3 修改密码

## 5.2 模型管理

模型管理是该系统中的重要部分，主要是进行模型数据的训练和配置。尤其在场景部分，数据和用户交互比较复杂。

## 5.2.1 应用列表

应用列表主要是对用户新建的应用在表格中进行展示，同时也便于用户对其需要的部分进行相关编辑操作等。在该页面中同时也支持按照关键词和应用修改时间进行模糊搜索和排序。

核心代码：

getAppList () {

let data = {

name: this.name,

date: this.date,

pageSize: this.pageSize,

pageNo: this.pageNo

}

this.$axios.post('app/list', data).then(response => {

if (response.data) {

if (response.data.list.length > 0) {

this.appList = response.data.list

this.total = response.data.total

} else {

this.appList = []

}

}

})

}

页面截图：

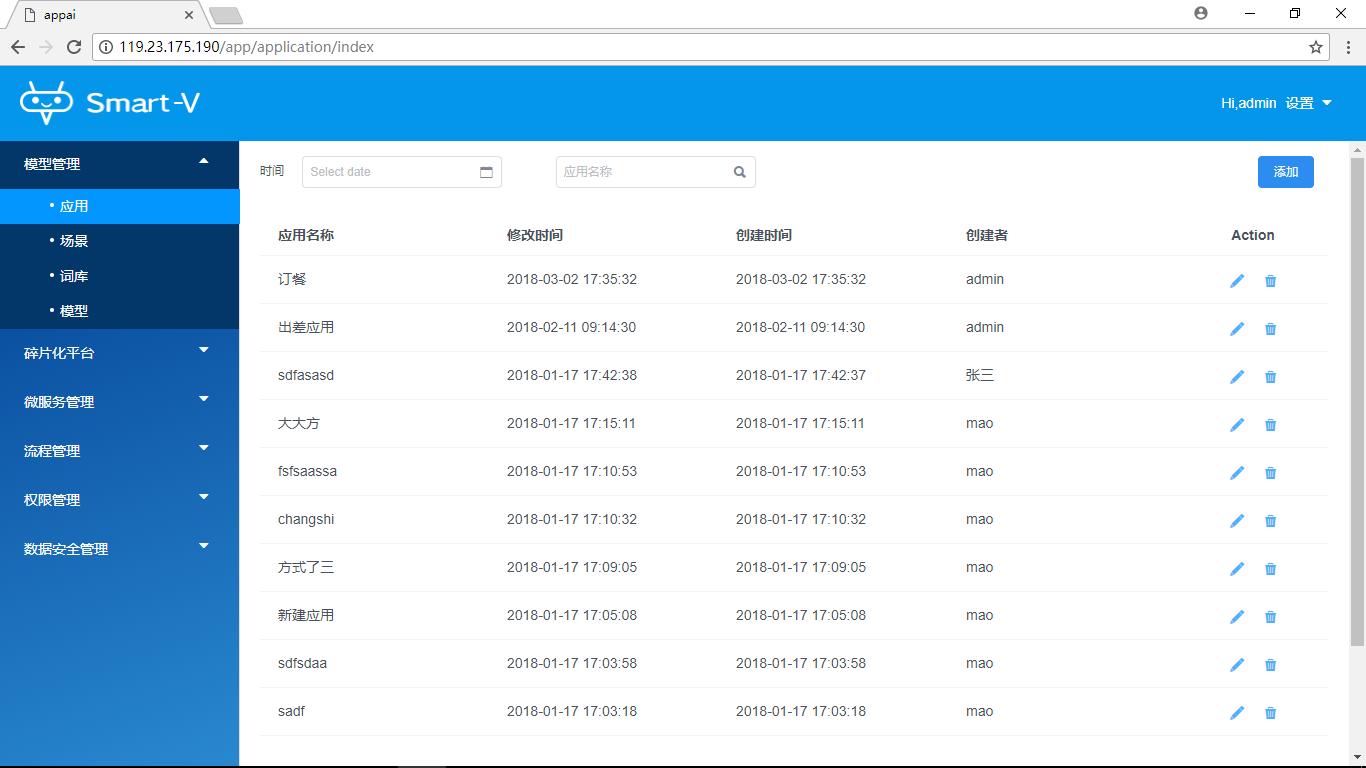


图5-4 应用列表

## 5.2.2 添加、编辑应用

添加编辑主要是实现对应用的相关操作，可以对既有的应用进行修改也可以添加新的应用。其实这二者有很多相似点，主要区别是在编辑的情况下需要先调用获取该应用详情的接口，并在保存的时候返回修改应用的appId，而添加时则直接将appId置为空即可。

核心代码：

getAppDetail () { // 获取应用详情

this.appId = $Storage.sessionStorage.getItem('appId')

this.$axios.post('app/detail', { id: this.appId }).then(response => {

if (response.data) {

let data = response.data

this.createAppForm.id = data.id

this.createAppForm.name = data.name

this.createAppForm.isPrivate = data.isPrivate

this.createAppForm.defReply = data.defReply

this.createAppForm.describe = data.describe

this.createAppForm.storage = data.storage

}

})

}

页面截图：

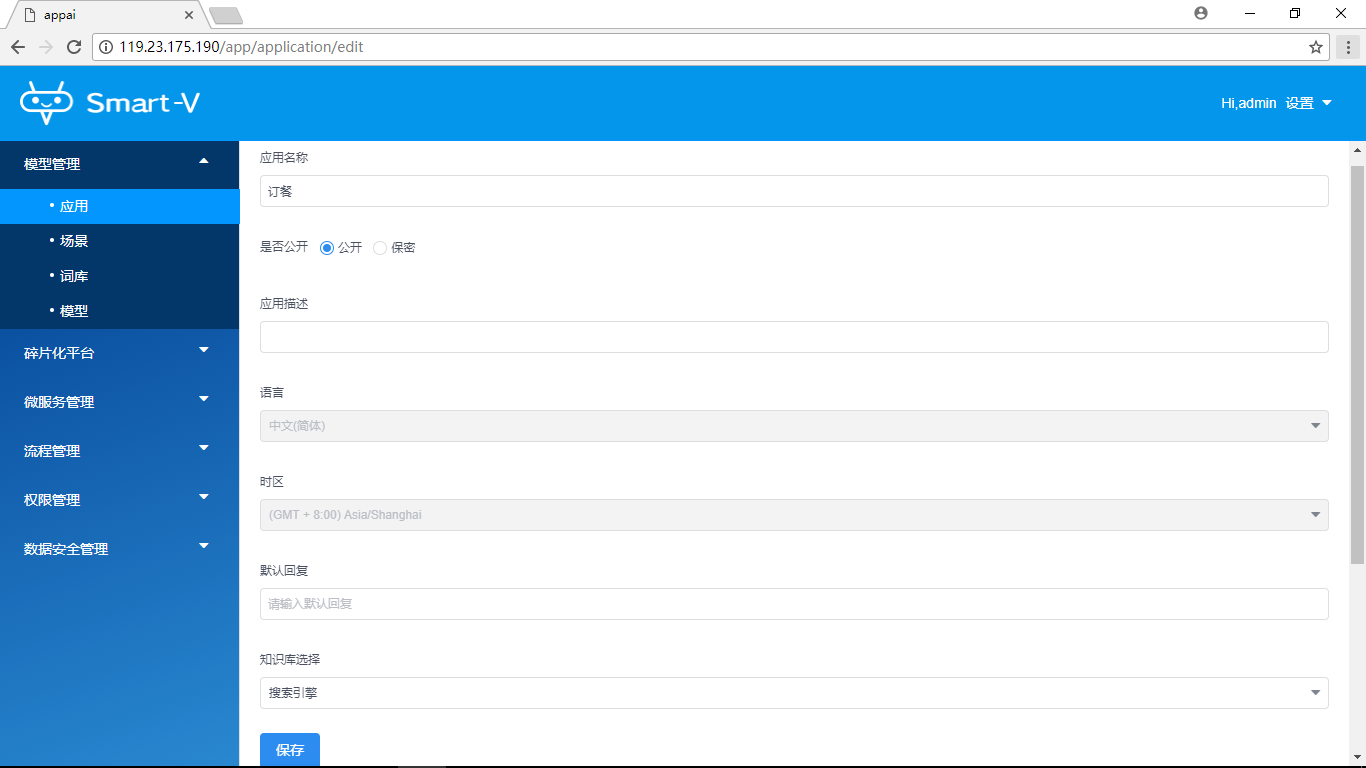


图5-5 应用编辑

## 5.2.3 删除应用

删除某应用，可以在编辑应用详情中选择删除，也可以在应用列表中点击删除图标进行快速删除。并回到应用列表或刷新列表。

核心代码：

deleteApp () { // 删除应用

this.$axios.post('app/del', { id: this.appId }).then(response => {

if (response.data === null) {

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

}

})

}

页面截图：

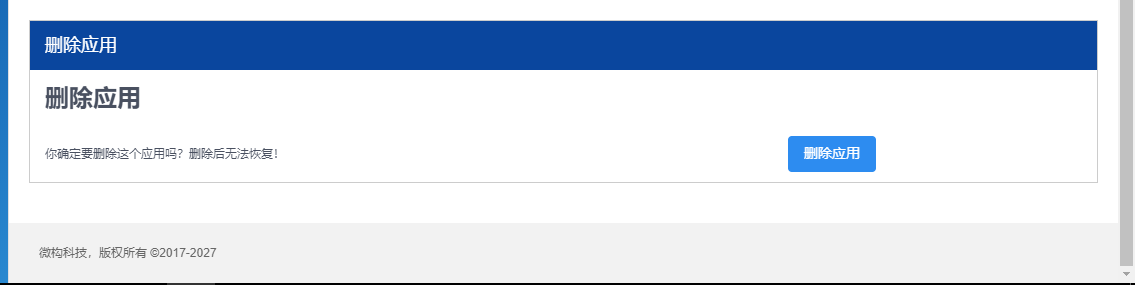


图5-6 删除应用

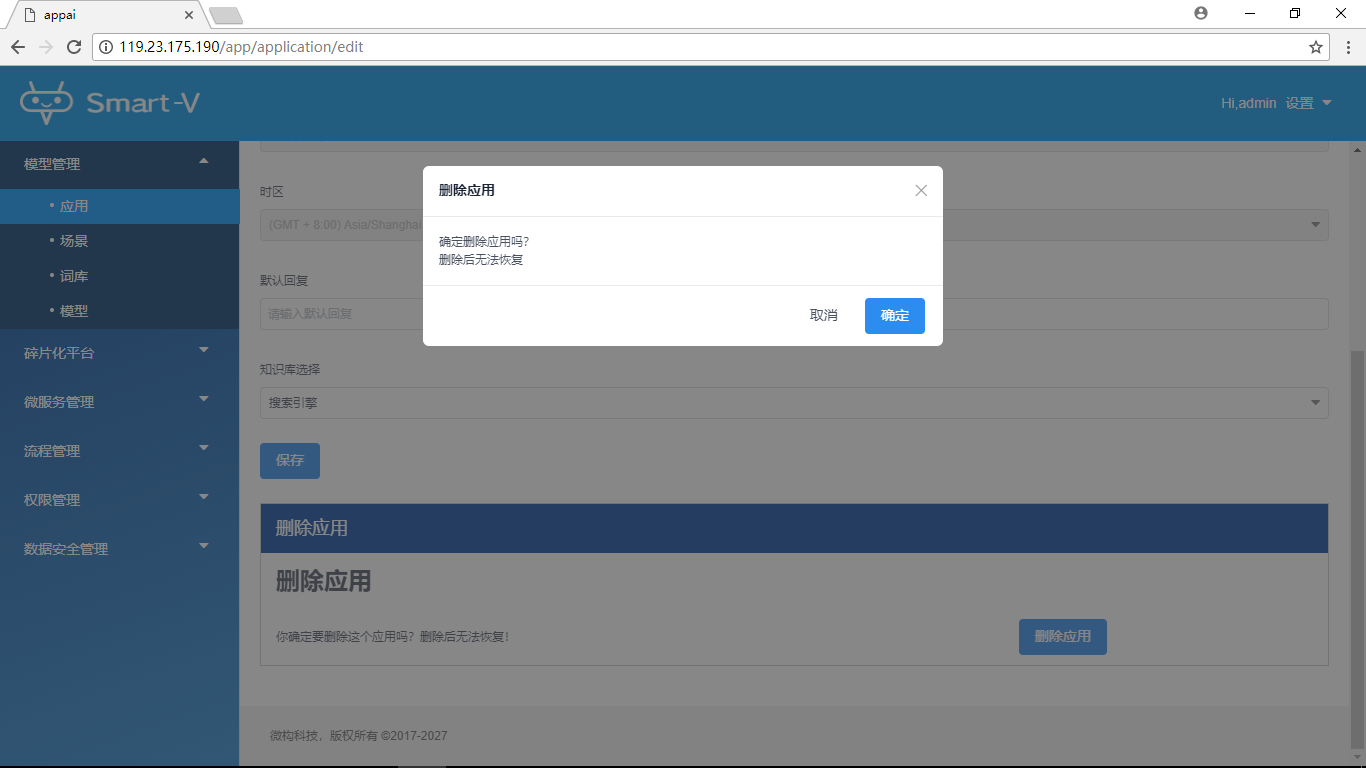


图 5-7 删除应用

## 5.2.4 场景列表

在应用列表中点击某一列即跳转到该应用下的场景列表，并将这个appId存入sessionStorage，即页面刷新时也能记住是那个应用下的场景。同时，该场景列表也按照支持关键词和修改时间进行筛选。

核心代码：

getIntentsList () { // 获取场景列表

let data = {

appId: this.getAppId,

name: this.name,

pageSize: this.pageSize,

pageNo: this.pageNo,

date: this.date

}

if (!this.getAppId) {

this.$Message.warning('请先选择一个应用')

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

} else {

this.$axios.post('intent/list', data).then(response => {

// …

})

}

}

页面截图：

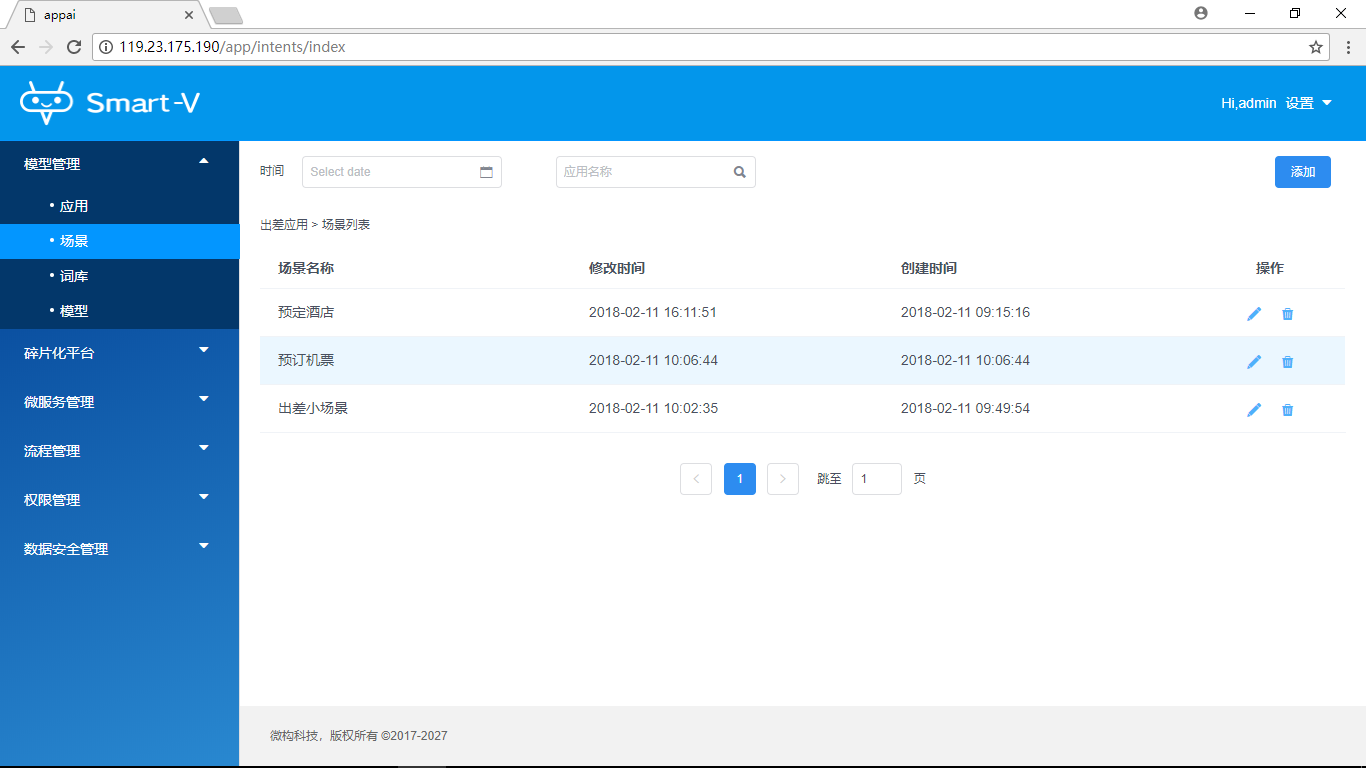


图 5-8 场景列表

## 5.2.5 添加、编辑场景

添加和编辑场景部分相对比较复杂，主要是由于业务逻辑需要，一再删改，实现了鼠标选中文字事件（onSelect）,能返回鼠标选中的文字，并触发下拉框的显示。传递数据的时候，对象属性会自动变成数组格式而使得后台报错，从而使用forEach函数，遍历每个对象，使用模板字符串将格式强行转换。

核心代码：

// 获取某场景详情

getIntentsDetail (selectIntent) {

this.intentId = selectIntent

this.$axios.post('intent/detail', { id: this.getIntentId }).then(response => {

if (response.data) {

// 为每个对象赋值

this.selectInputList = data.inputObj || []

for (let index = 0; index < this.entitiesList.length; index++) {

for (let i = 0; i < this.askList.length; i++) {

for (let j = 0; j < this.askList[i].entitys.length; j++) {

if (this.askList[i].entitys[j].entity === this.entitiesList[index].pinyin) {

this.askList[i].entitys[j].type = this.entitiesList[index].name

}

}

}

}

if (this.askList.length < 1) {

this.askList.push({ text: this.askList.text, entitys: [] })

}

this.slotList = data.slotList

if (this.slotList.length < 1) {

this.slotList.push({ defValue: this.slotList.defValue, typeName: this.slotList.typeName, dictName: this.slotList.dictName, flag: false, message: '' })

} else {

for (let item of this.slotList) {

if (item.flag === 'false') {

item.flag = false

} else if (item.flag === 'true') {

item.flag = true

}

}

}

this.actionName = data.actionName

// 获取动作弹框 详情

this.acForm = data.actService

}

})

},

页面截图：

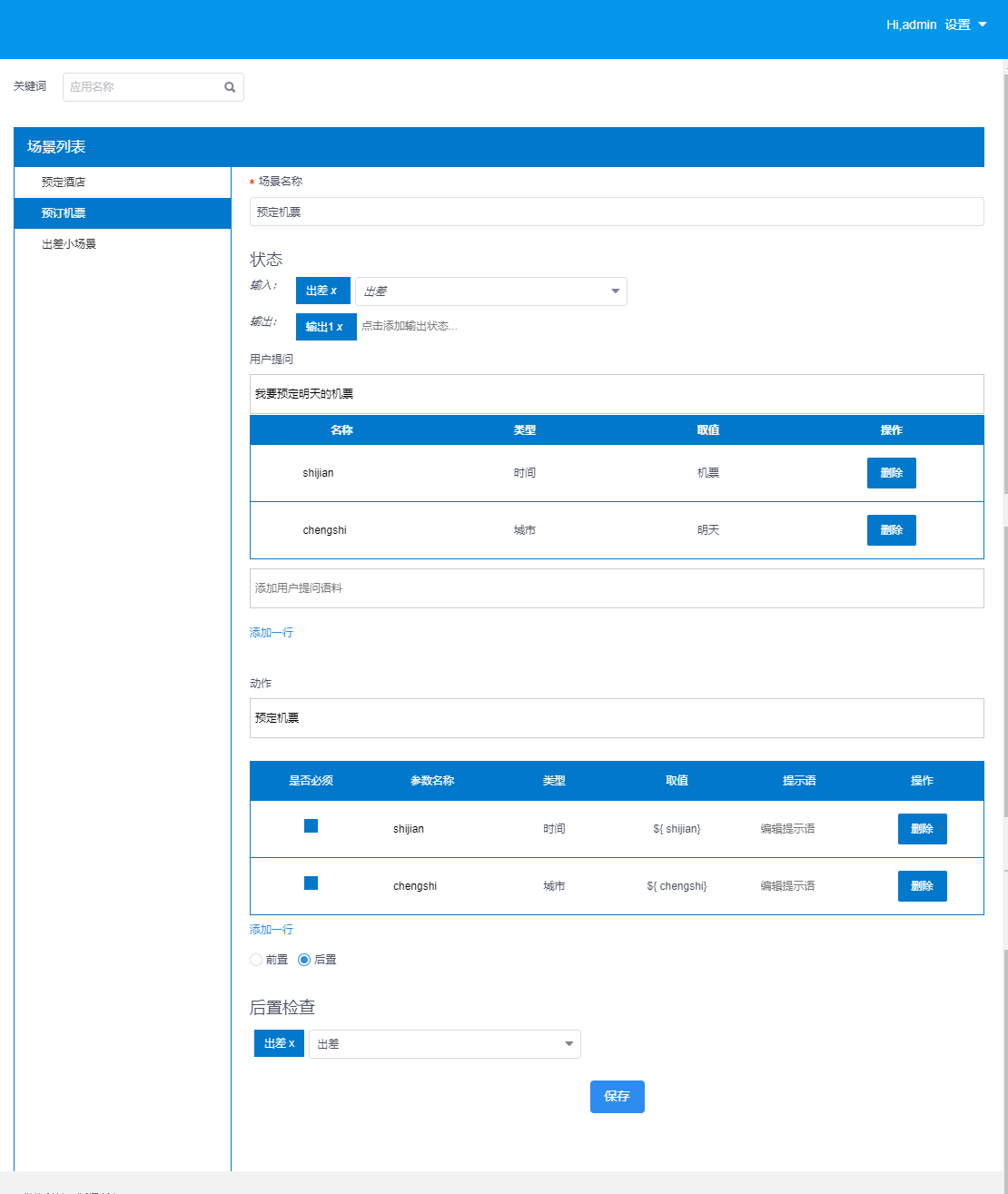


图 5-9 添加、编辑场景

## 5.2.6 删除场景

点击删除按钮，显示模态框，并将要删除的场景id传递给全局，当用户点击确定时，再调用删除接口，并将场景id传递给该接口。如果返回值为空，则删除成功，全局提示场景删除成功，返回场景列表，并刷新该列表。

核心代码：

del (params) {

this.showModal = true

this.delId = params.id

},

delIntents () {

this.$axios.post('intent/del', { appId: this.appId, id: this.delId }).then(response => {

if (response.data === null) {

this.$Message.success('提交成功！')

this.getIntentsList()

}

})

},

页面截图：

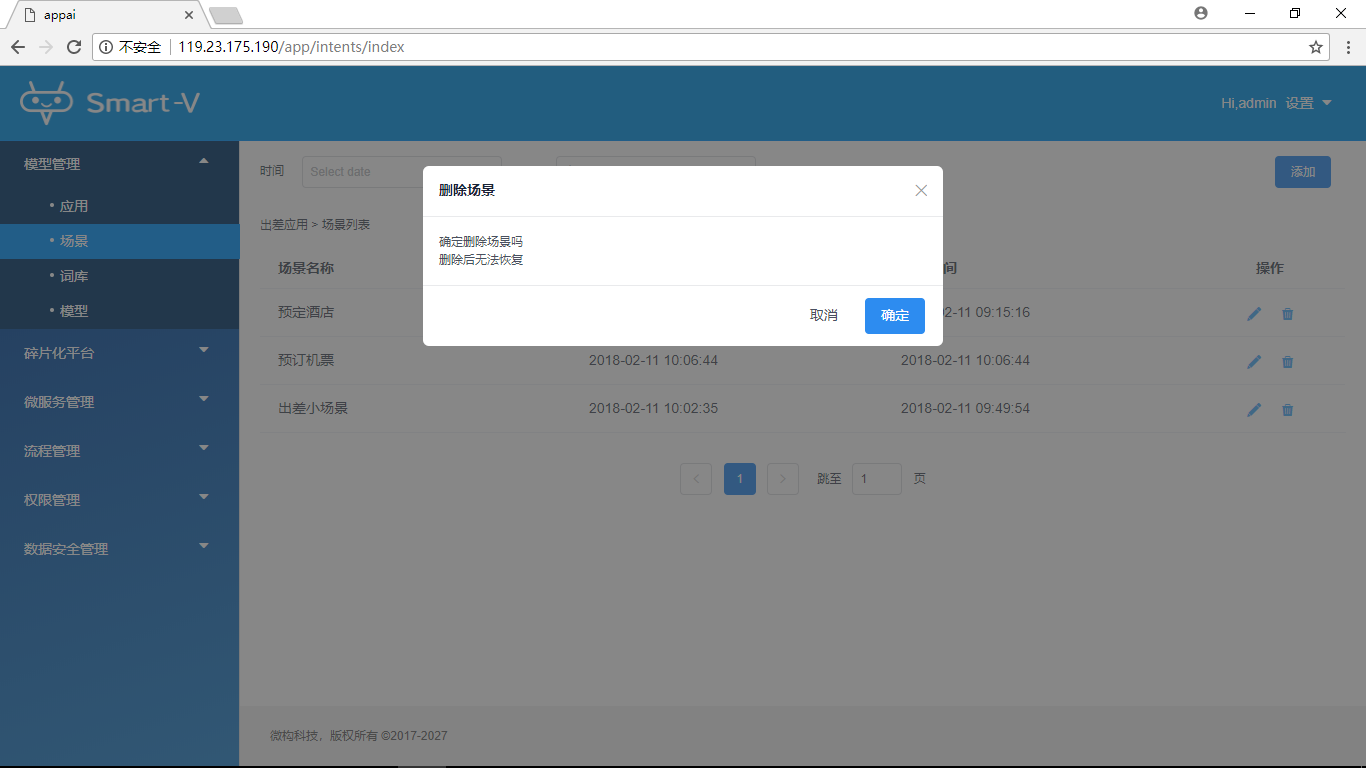


图 5-10 删除场景

## 5.2.7 词库列表

词库列表也是与应用相关联，即一个应用下可有多个场景和多个词库，场景和词库相关联，某个场景下的词库他们都有同一个应用。词库即slot,模型训练很大一部分也就是填充插槽即slot词库，最后将信息填充完整。该词库列表也支持关键词和修改时间进行模糊查询。

核心代码：

// 获取词库列表

getEntitiesList () {

let data = {

// … 变量赋值

}

if (!this.getAppId) {

this.$Message.warning('请先选择一个应用')

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

} else {

this.$axios.post('dict/list', data).then(response => {

if (response.data.dictList.length > 0) {

this.entitiesList = response.data.dictList

} else {

this.entitiesList = []

}

})

}

},

页面截图：

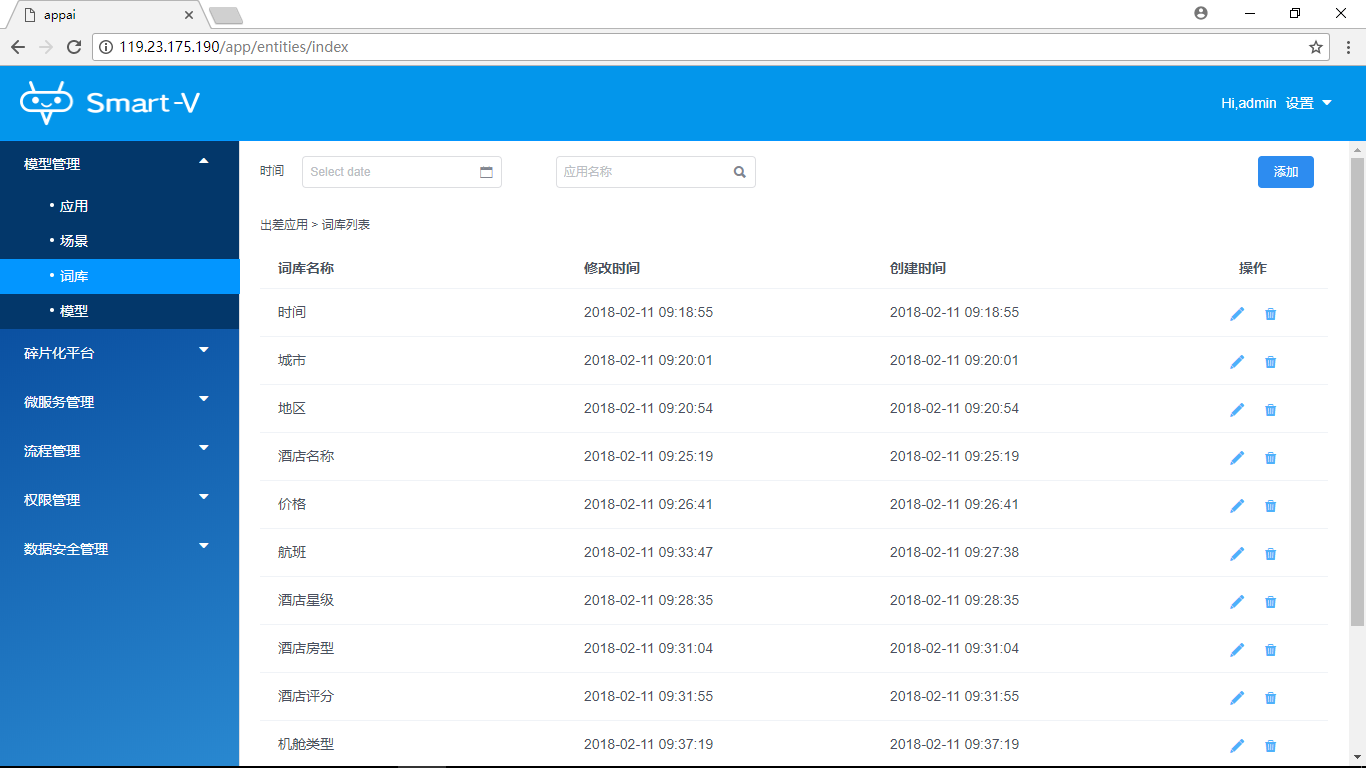


图 5-11 词库列表

## 5.2.8 添加、编辑词库

添加、编辑词库即重新定义词库的详细内容，可以为词库定义新的关键词，实现Enter键即添加新的一行，使用自定义指令实现，并绑定鼠标按下事件（onKeyUp），而在页面销毁时，解除该事件绑定。

核心代码：

getEntitiesList () { // 获取词库列表

this.appId = this.$store.state.appId

if (!this.appId) {

this.appId = this.getAppId

}

this.$axios.post('dict/list', { name: this.name, appId: this.appId }).then(response => {

if (response.data.dictList.length > 0) {

this.entitiesList = response.data.dictList

this.hasEntities = true

}

})

},

页面截图：

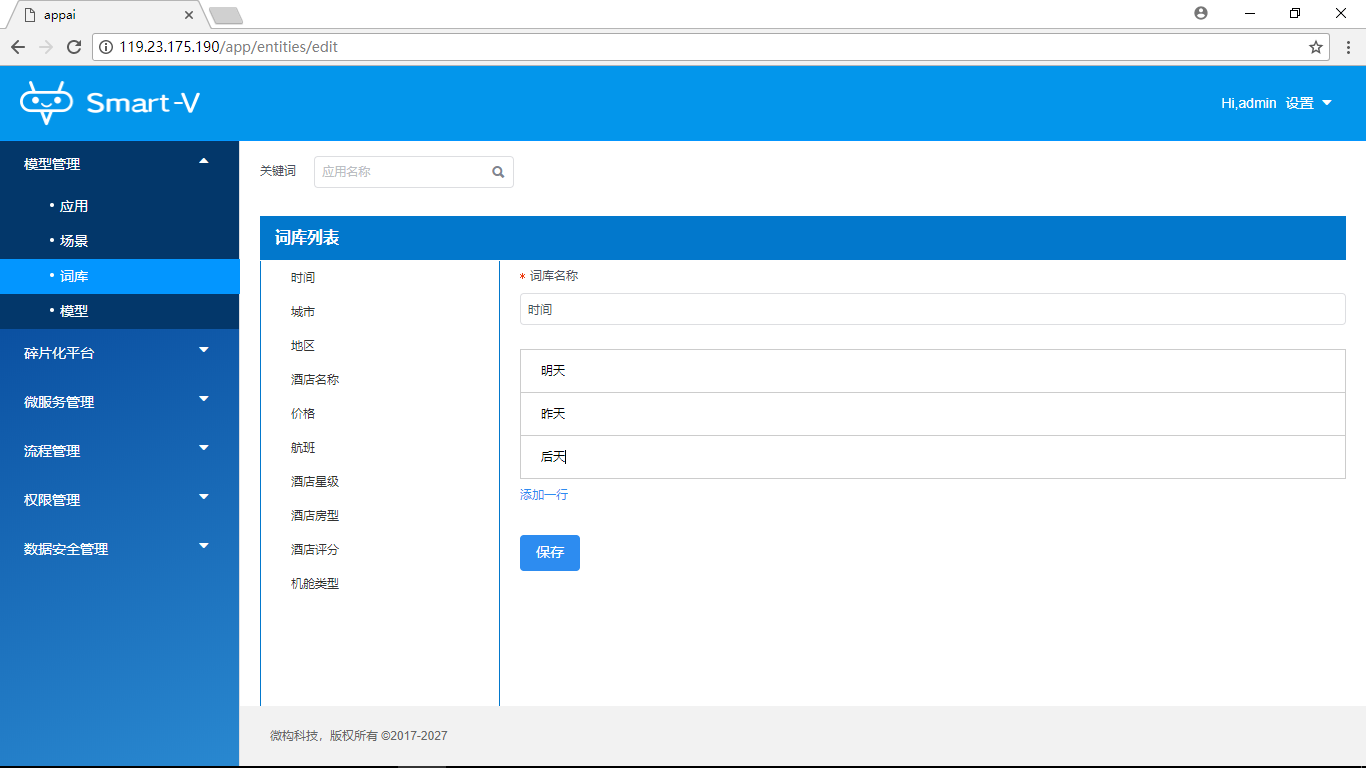


图 5-12 添加、修改词库

## 5.2.9 删除词库

在词库列表中点击删除按钮，即可将要删除的词库id保存到全局中，点击确定时，就调用删除接口，并将词库id传递给删除接口。

核心代码：

// 删除词库

delEntities () {

this.$axios.post('dict/del', { id: this.delId }).then(response => {

if (response.data === null) {

this.$Message.success('提交成功！')

this.getEntitiesList()

}

})

},

页面截图：

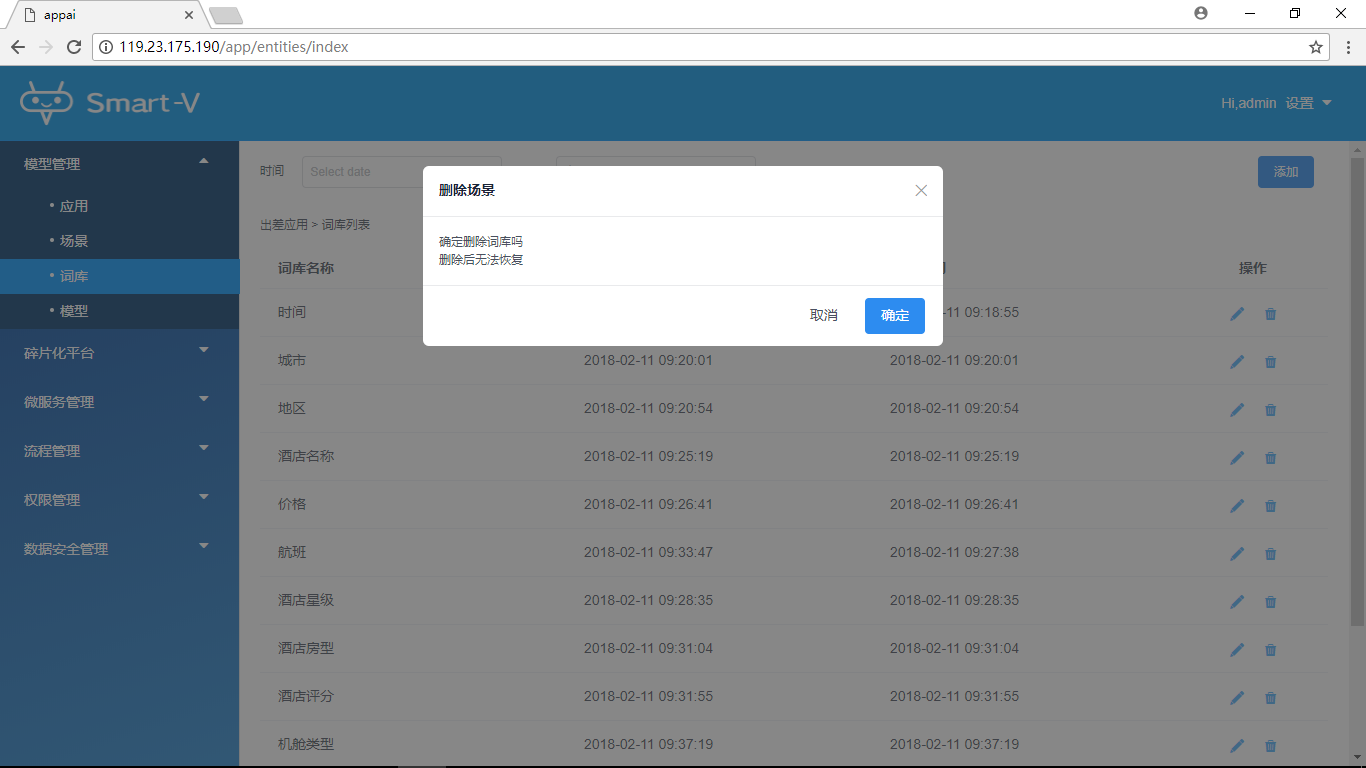


图 5-13 删除词库

## 5.2.10 模型训练

下拉框中是所有的应用名称选项，可以选择某一项应用进行数据训练，这边主要是对接了python的接口，进行操作。

核心代码：

getAppList () {

let data = {

name: this.name,

date: this.date,

pageSize: this.pageSize,

pageNo: this.pageNo

}

this.$axios.post('app/list', data).then(response => {

if (response.data) {

this.appList = response.data.list

}

})

}

页面截图：

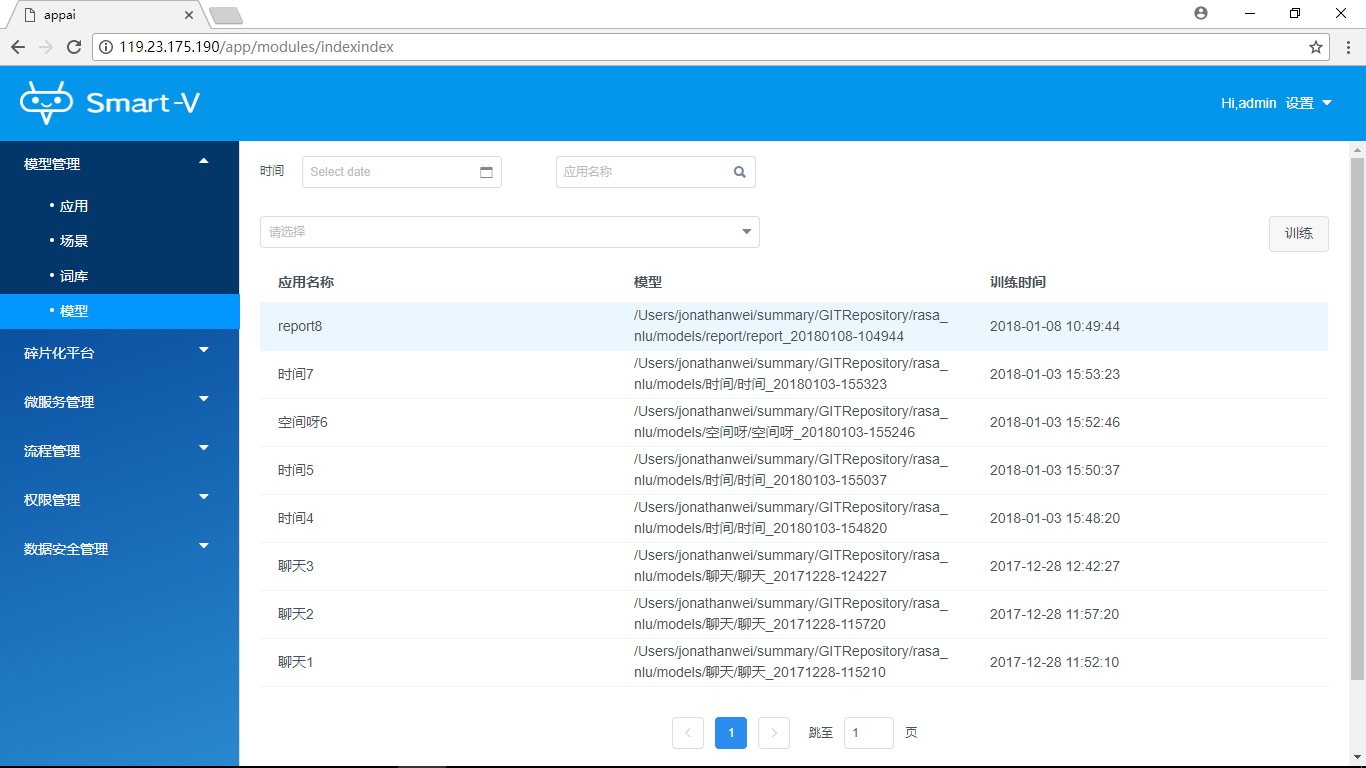


图 5-14 模型训练

## 5.3 微服务管理

## 5.3.1 微服务管理列表

微服务管理列表列出了所有应用下的微服务，以及它对应的描述、接口、url等等。用户可以随时对其进行删除、编辑等操作。选中某项微服务将microId 存入sessionStorage保证页面在刷新时，仍能保存相关数据。同时，该列表也支持关键词和日期模糊查询。

核心代码：

getMicroList () {

let data = {

name: this.name,

date: this.date,

pageSize: this.pageSize,

pageNo: this.pageNo

}

this.$axios.post('mic\_service/list', data).then(response => {

if (response.data) {

this.microList = response.data.mServiceList

}

})

}

页面截图：

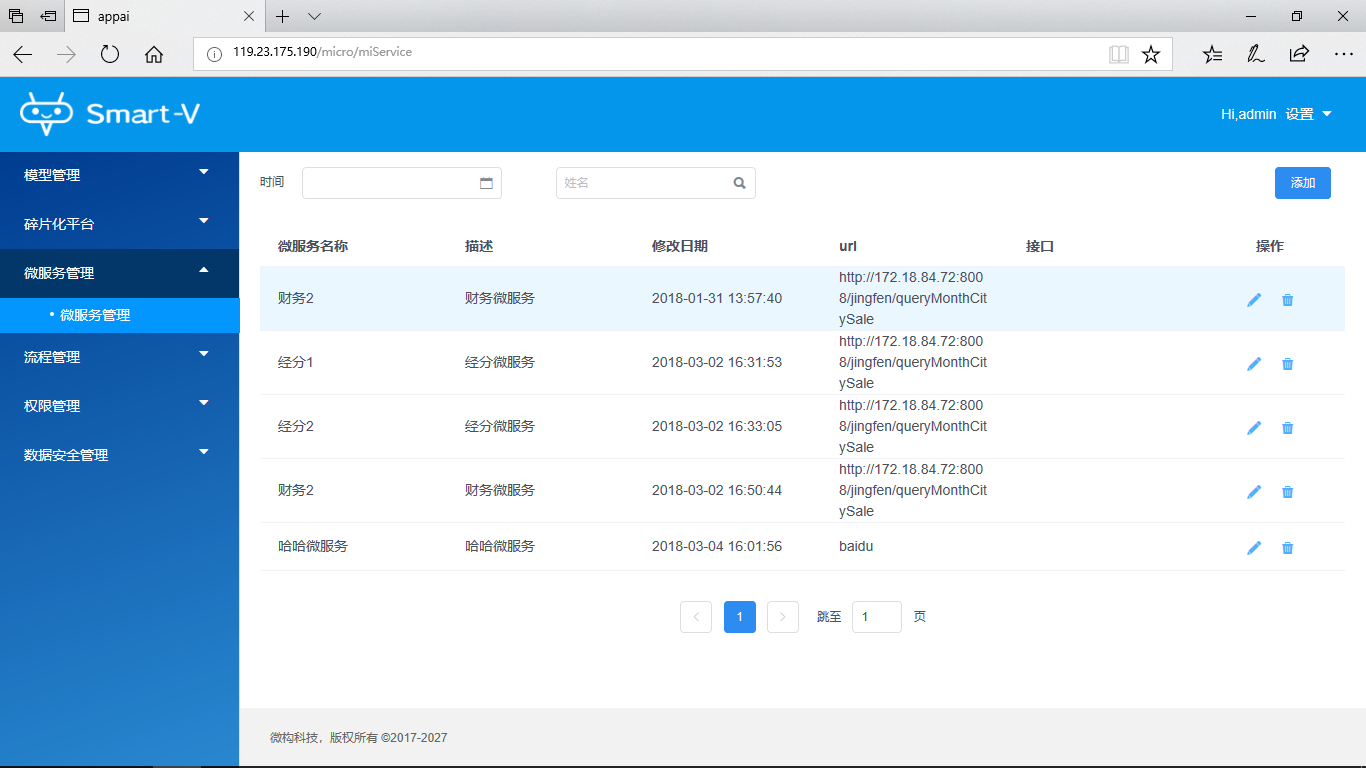


图 5-15 微服务管理

## 5.3.2 添加、编辑微服务

一个微服务中可以存在多个参数和值，而名称描述和url只有一个，因此在添加、编辑页面中可以另外多添加几个参数，点击添加时弹出模态框，对需要内容进行填充，最后保存所有修改内容即可。

核心代码：

getMicroDetail () {

let id = this.$store.getters.microId

if (id) {

this.$axios.post('mic\_service/detail', { id: id }).then(response => {

if (response.data) {

this.microForm = response.data

this.attrList = response.data.attrList

console.log('attrList', this.attrList)

}

})

}

},

addParams () {

this.attr = {}

this.showModal = true

},

页面截图：

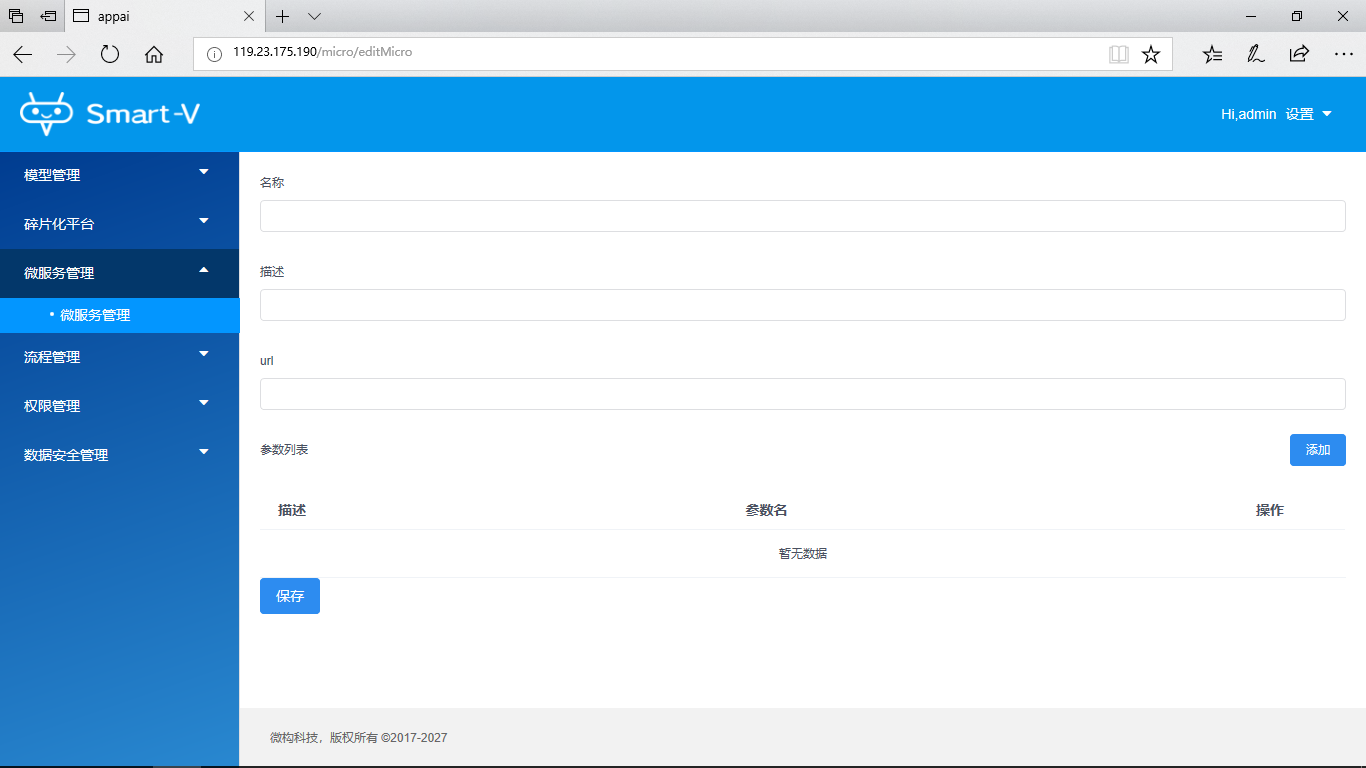


图 5-16 添加、编辑微服务

## 5.3.3 删除微服务

在微服务列表中，选中需要删除的某一项弹出模态框，请用户确认是否要删除，点击确定按钮，删除该项微服务，隐藏模态框并刷新微服务列表。

核心代码：

deleteMicro () {

this.$axios.post('mic\_service/delete', { id: this.delId }).then(response => {

if (response.data === null) {

this.$Message.success('删除成功！')

this.getMicroList()

}

})

}

页面截图：

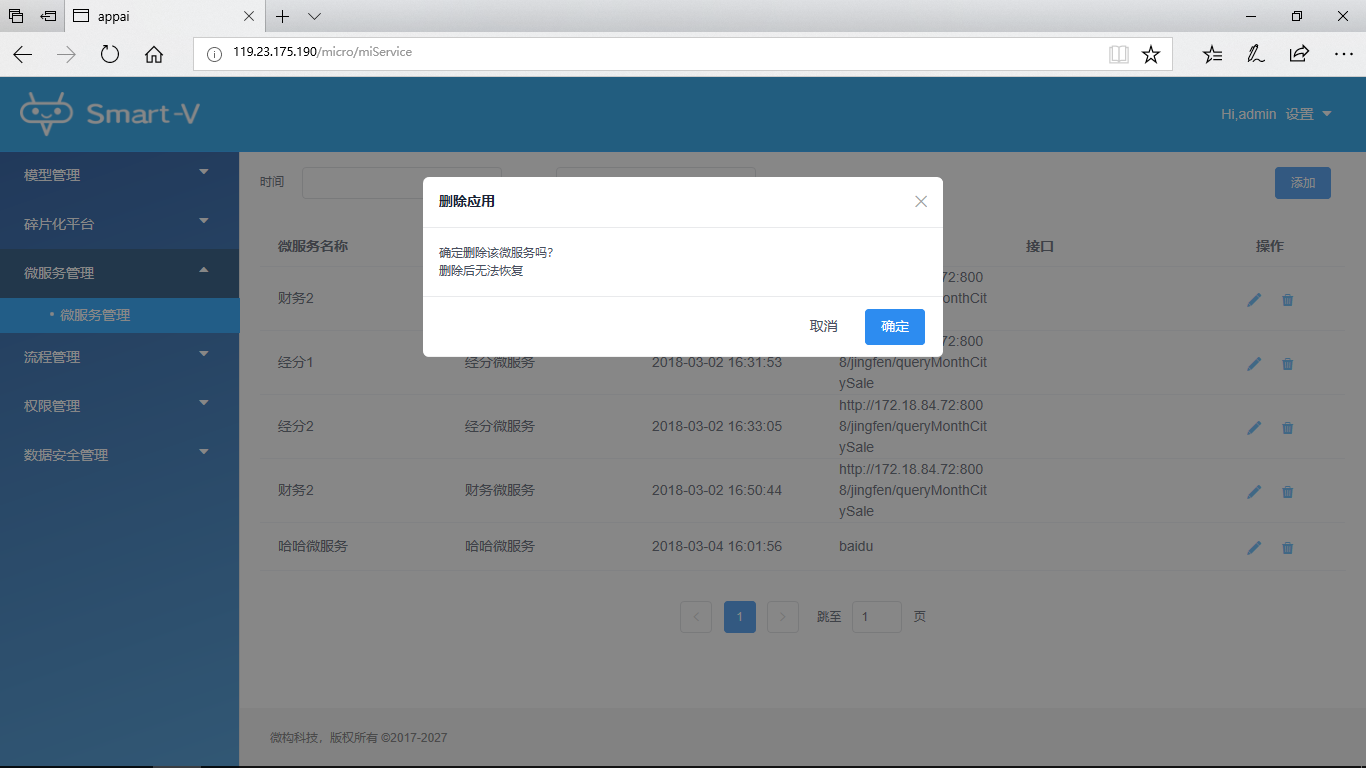


图 5-17 删除微服务

## 5.4 流程管理

## 5.4.1 流程管理列表

流程管理列表主要展示了流程列表信息，管理了数据流程。

核心代码：

getFlowList () {

let data = {

date: this.date,

name: this.name,

pageSize: this.pageSize,

pageNo: this.pageNo

}

this.$axios.post('process/list', data).then(response => {

if (response.data) {

this.processList = response.data.processList

}

})

}

页面截图：

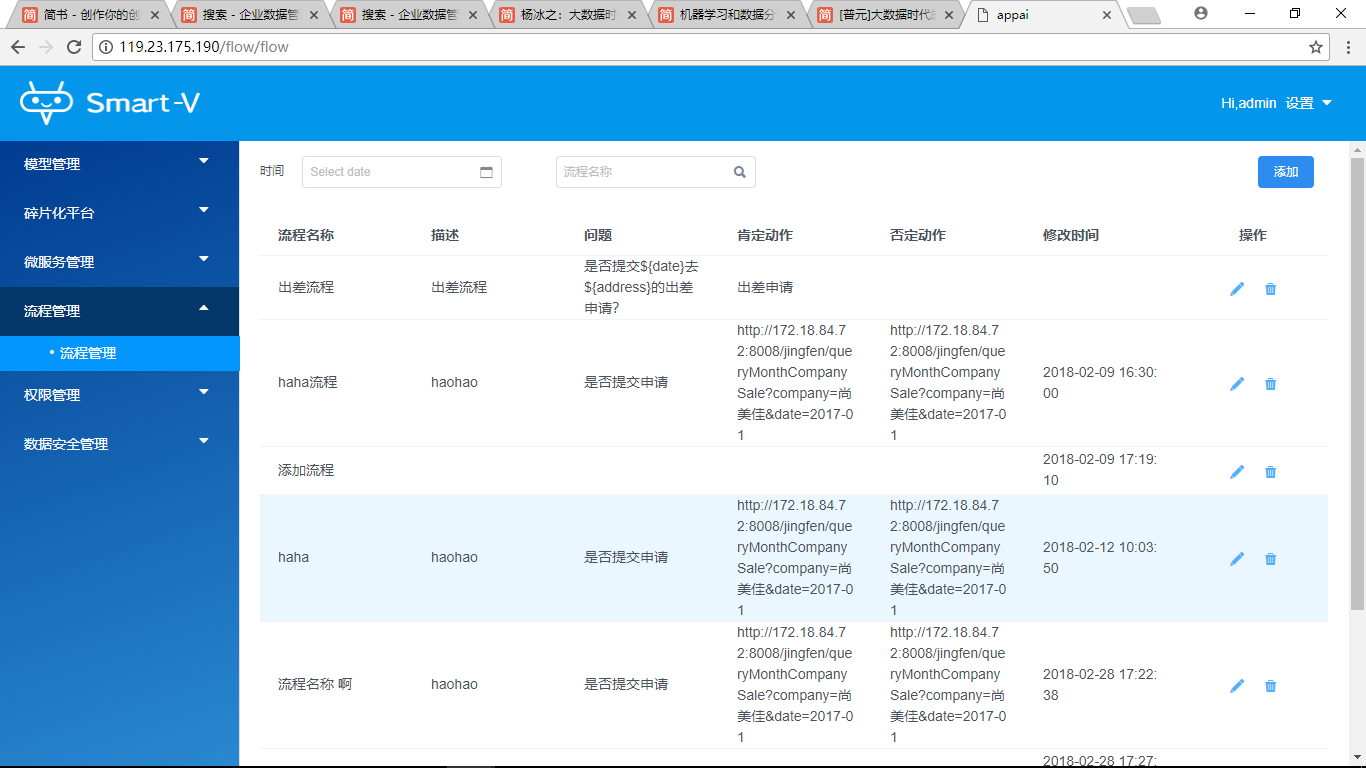


图 5-18 流程管理列表

## 5.4.2 添加、编辑流程

在流程列表中选择编辑，即可实现重新改写流程详情内容，也可以新添加新的流程，二者极为相似，不同点在于编辑前需要调用一次详情接口并在保存时，再次提交保存流程的id，而添加直接跳转添加页面，并将id赋值为null即可。

核心代码：

getDetail () {

if (this.flowId) {

this.$axios.post('process/detail', { id: this.flowId }).then(response => {

if (response.data) {

this.flowForm = response.data

this.processBlockList = response.data.processBlockList

}

})

} else {

this.flowForm = {}

}

console.log('process', this.processBlockList)

}

页面截图：

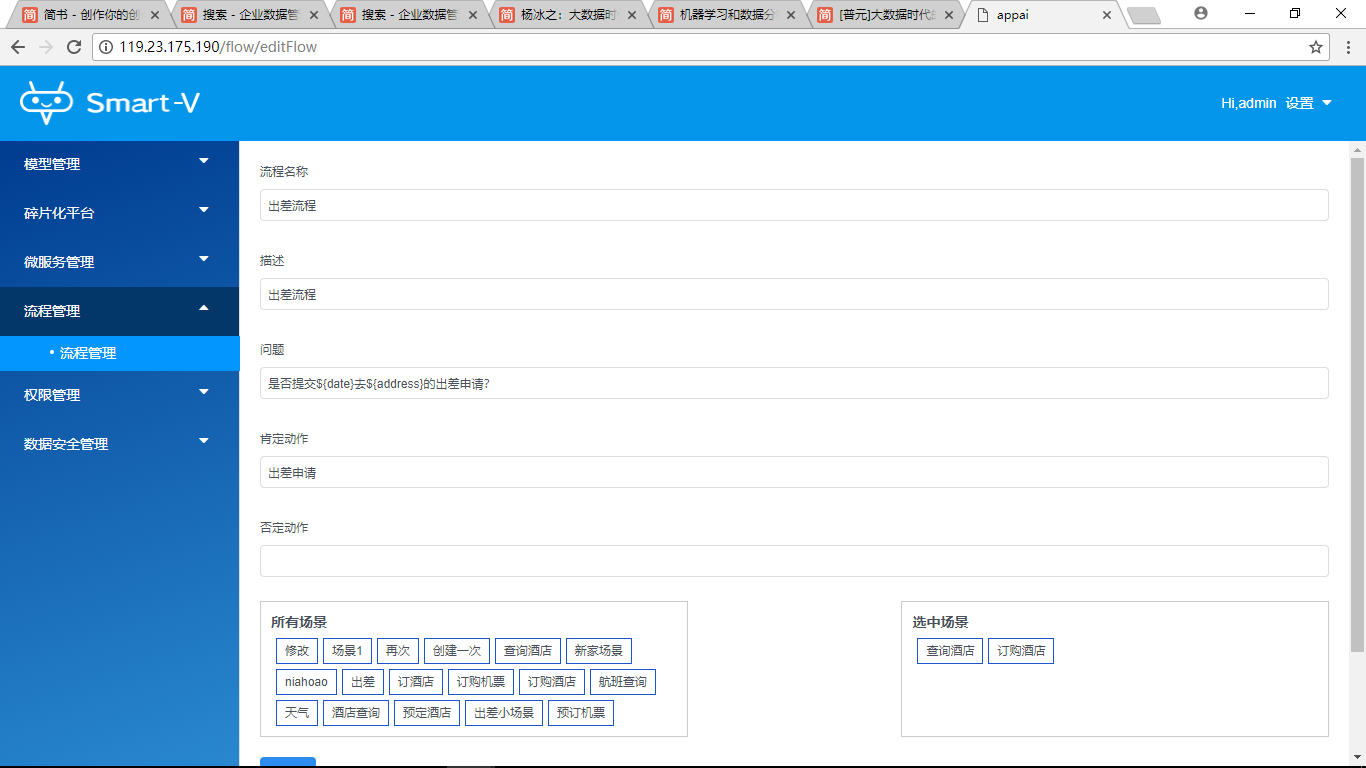


图 5-19 添加、编辑流程

## 5.4.4 删除流程

在流程列表中点击删除图标，弹出模态框再次询问是否确认删除该流程。再次选择确认后，对该流程进行删除操作，并重新刷新流程列表。

核心代码：

// 删除该流程

deleteFlow (*id*) {

this.$axios.post('process/delete', { id: id }).then(*response* *=>* {

if (response.data === null) {

this.$Message.success('删除成功！')

this.getFlowList()

}

})

}

页面截图：

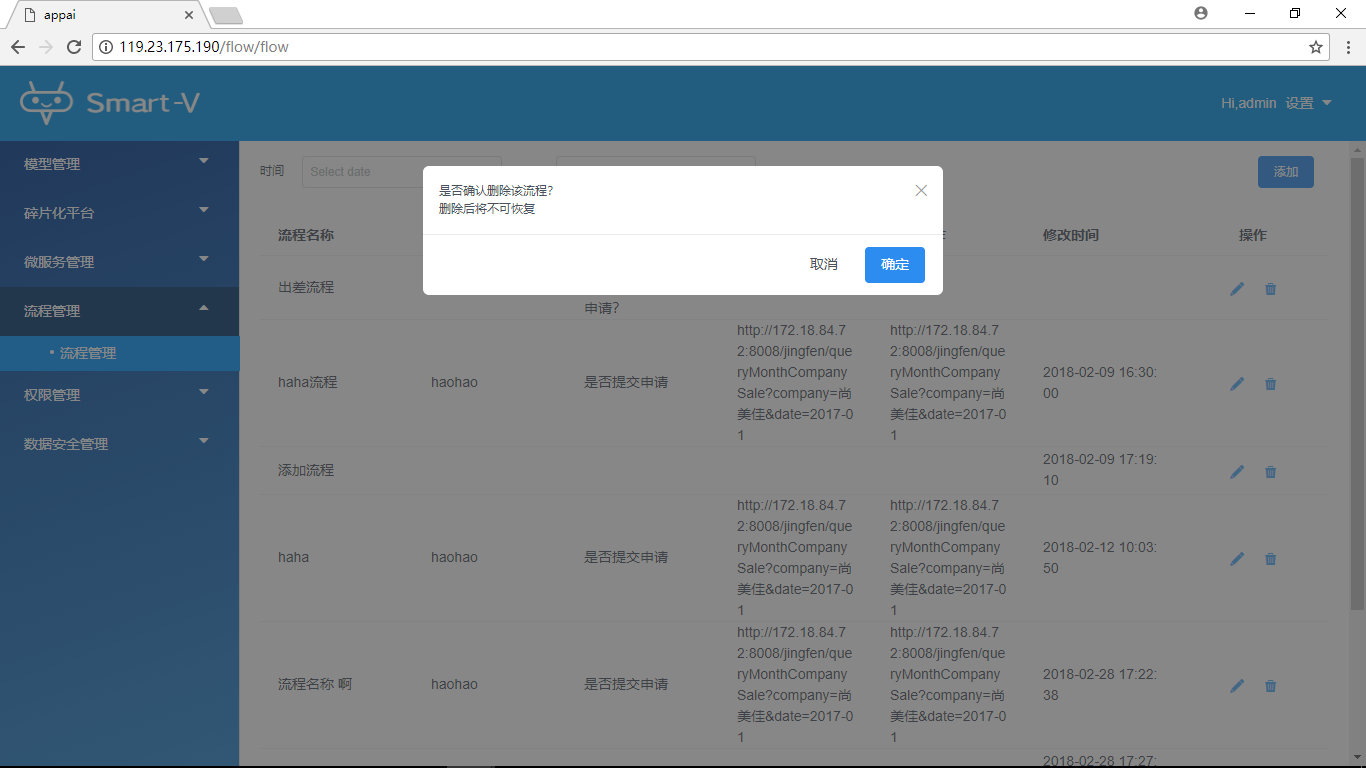


图 5-20 删除流程

# 6. 系统测试和维护

## 6.1 系统测试

### 6.1.1 系统测试的作用和意义

系统测试（System Testing）,即测试一个完整的系统，是将操作人员、软件、硬件视为一个整体，来检测是否有地方不符合系统说明书。系统测试是为了检测出在系统分析和设计中存在的问题。系统测试是将软件，经过集成测试作为计算机系统的一个部分，并结合系统中的其他部分，在实际运行环境下对计算机系统进行的一系列有效严格地测试，以发现软件潜在的问题来保证系统运行正常。

主要进行功能测试（含可使用性测试）、性能测试、安全测试和回归测试。

（1）功能测试：检查被测系统的修改和增加功能是否正常实现；检查控制流程图和模块关系图、模块内部关系图；识别特殊情况，如出错处理流程，错误提示是否合理；检查用户界面是否符合窗口程序的标准，界面操作是否简便直观。

（2）性能测试：系统运行占用的资源，完成某一步骤需要的时间；系统能承受的压力；压力完成后数据库连接数立即恢复正常值

（3）安全性测试：根据需求说明检查系统是否达到安全性要求。不同机器，同时操作对数据的破坏；写到配置文件或数据库的密码是否经过加密；

（4）回归测试；验证Bug是否修正；Bug修正后是否影响其他功能的正常运行。

系统测试的意义在于，使发布出去的产品的质量能得到保证。软件测试工程师的工作内容就是按照测试方案和流程利用测试工具测试产品的功能和性能，甚至有时需要根据需求编写不同的测试工具，来设计和维护测试系统，分析和评估测试方案可能出现的问题。对测试用例进行执行，并跟踪故障，以确保开发的产品适合需求。使用自动或人工手段来测试或运行某个系统的过程,来检验它是否满足规定需求或弄清预期结果与实际结果之间的差别。它是帮助识别开发完成（中间或最终的版本）的计算机软件（整体或部分）的正确度(correctness) 、完全度(completeness)和质量(quality)的软件过程；是SQA(software quality assurance)的重要子域。

### 6.1.2 系统测试过程和结果

（1）用户登录测试：当用户登录成功后才可以进行后续的相关操作，因此，用户登录时系统测试中最初也是最重要的一步。

表6-1 登录测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能A描述 | 用户登录 |
| 用例目的 | 正确对应的用户名和密码才能成功登录 |
| 前提条件 | 用户名和密码不为空 |
| 输入操作 | 正确输入用户名和密码；正确用户名和错误密码 |
| 希望结果 | 成功登录；提示密码错误 |
| 实际结果 | 符合 |

（2）应用模块测试：应用模块是系统中的重要组成部分。后续场景和词库的创建都是基于应用的基础。要求应用的增删改查都完全实现，且页面刷新时，数据不会改变或者丢失。

表6-2 应用模块测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能B描述 | 增加、编辑、删除应用 |
| 用例目的 | 确认应用的增删改查等功能实现和完善 |
| 前提条件 | 添加和编辑时输入相关信息，应用名称不能为空；删除时获取应用id |
| 输入操作 | 输入应用详情的具体内容；选择确认删除某项应用 |
| 希望结果 | 提示应用添加/修改成功，刷新后返回应用列表页；提示删除成功，刷新列表页。 |
| 实际结果 | 符合 |

（3）场景模块测试：场景模块是整个系统中数据和交互最复杂的部分，容易出现数据错误，输入各种数据，进行不同操作都应该能够正常运行，显示也正常。

表6-3 场景模块测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能C描述 | 增加、编辑、删除场景 |
| 用例目的 | 编辑场景时能正常保存到数据库；删除场景成功 |
| 前提条件 | 按要求添加场景详情信息 且场景名称不得为空；在场景列表中选择需要删除的场景。 |
| 输入操作 | 编辑场景信息，场景名为空；添加场景信息，场景名不为空；点击删除场景 |
| 希望结果 | 提示场景名不得为空；场景添加成功，返回场景列表；删除成功，刷新场景列表。 |
| 实际结果 | 符合 |

（4）词库模块测试：添加、编辑词库时，按下键盘回车键即可新添加一行关键词，保存后左侧词库列表能看到修改/添加后的词库名称。删除词库后，页面刷新，能显示正确的词库列表。

表6-4 词库模块测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能D描述 | 添加、编辑、删除词库 |
| 用例目的 | 系统可以实现词库详情内容的添加/编辑；成功删除词库 |
| 前提条件 | 填入词库详情内容且词库名称不得为空；选择需要删除的某项词库 |
| 输入操作 | 输入词库详情；点击删除某词库 |
| 希望结果 | 保存后显示提交成功，刷新词库列表；显示删除成功，刷新词库列表 |
| 实际结果 | 符合 |

（5）微服务管理模块：微服务首页显示微服务列表，并能进行相关操作如增加、编辑、删除等。要求页面内容与后台数据库数据相一致。

表6-5 微服务管理测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能E描述 | 微服务管理的增加、修改、删除等 |
| 用例目的 | 系统能否正常对微服务相关内容进行编辑；是否删除成功微服务 |
| 前提条件 | 输入微服务详情内容；选择某一项微服务删除 |
| 输入操作 | 按要求填入微服务相关信息；点击某一项微服务并确定删除 |
| 希望结果 | 微服务修改成功；删除成功 |
| 实际结果 | 符合 |

（6）流程管理测试：展示流程列表，进行相关编辑、添加、删除操作。 保证界面上的增删改和后台数据库中的数据一致。

表6-6 流程管理测试表

|  |  |
| --- | --- |
| 测试名称 | 测试内容 |
| 功能F描述 | 添加、编辑、删除流程 |
| 用例目的 | 系统能否正常的添加、编辑流程；能否正常删除流程 |
| 前提条件 | 填入流程详情；点击删除流程 |
| 输入操作 | 输入流程的详情信息；确认删除流程 |
| 希望结果 | 流程详情信息保存成功；删除成功 |
| 实际结果 | 符合 |

## 

## 6.2 系统维护

系统维护主要是指根据需求变化或硬件环境的变化对应用程序进行部分或全部的修改，修改时应充分利用源程序．修改后要填写程序改登记表，并在程序变更通知书上写明新旧程序的不同之处。

软件维护的内容一般有以下几个方面。

（1）正确性维护。是指改正在系统开发阶段已发生而系统测试阶段尚未发现的错误。这方面的维护工作量要占整个维护工作量的17％～21％。所发现的错误有的不太重耍，不影响系统的正常运行，其维护工作可随时进行：而有的错误非常重要，甚至影响整个系统的正常运行，其维护工作必须制定计划，进行修改，并且要进行复查和控制。

　　（2）适应性维护。是指使用软件适应信息技术变化和管理需求变化而进行的修改。这方面的维护工作量占整个维护工作量的18％～25％。由于目前计算机硬件价格的不断下降。各类系统软件层出不穷，人们常常需要为改善系统硬件环境和运行环境而使得系统及时更新换代；不断变化的企业的外部市场环境和管理需求也使得各级管理人员不断提出新的信息需求。这些因素都将使得产生适应性维护工作。进行这方面的维护工作也要像系统开发一样，有计划、有步骤地进行。

　　（3）完善性维护，这是为改善性能和扩充功能而进行的修改，主要是指对已有的软件系统增加一些在系统分析和设计阶段中没有规定的功能与性能特征。这些功能对完善系统功能足非常必要的。另外，还包括对处理效率和编写程序的改进，这方面的维护占整个维护工作的50％～60％，比重较大．也是关系到系统开发质量的重要方面。这方面的维护除了要有计划、有步骤地完成外．还要注意将相关的文档资料加入到前面相应的文档中去。

　　（4）预防性维护。为了改进应用软件的可靠性和可维护性，为了适应未来的软硬件环境的变化，应主动增加预防性的新的功能，以使应用系统适应各类变化而不被淘汰。例如将专用报表功能改成通用报表生成功能，以适应将来报表格式的变化。这方面的维护工作量占整个维护工作量的4％左右。

# 结 论

本文主要分析了当前大数据平台的发展形势和未来。基于此开发了企业智能数据管理系统，主要用于企业大数据的存储、分析和管理。在实际的投入使用中实现了为企业决策部门和管理层提供便捷、多途径的企业决策工具。提高了企业整体分析研究能力、市场快速反应能力，提高了企业的核心竞争力。

在实际的开发过程中，除了顺利运行的部分，也确实遇到了一些问题和困难，我先是自己上网或查阅工具书找解决方法，如果还是没能成功再寻求同事的帮助，虽然这个过程比较艰难，但也正是在不断解决问题中使我不断成长，很感激在开发过程帮助过我的人们，使我的困难最终都还是圆满的解决了。在长达几个月的开发时间里，有酸有苦，庆幸的是最后还是甜的。虽然不能做到十全十美，但也达到了之前给自己定的预期，我觉得很开心也很骄傲。

# 致 谢

毕业设计就这样完成了，这也为我的大学生涯画上了一个圆满的句号。依稀还记得刚入学的时候，拿着录取通知书在那个酷热的九月，满怀着期待和憧憬。时间如指间沙，匆匆溜走，一晃眼毕业已临近。经历了找工作的喧嚣与坎坷，我深深体会到了写作论文时的那份宁静与思考。回首四年的求学历程，对那些引导我、帮忙我、激励我的人，我心中充满了感激。

在此，首先我要感谢给予我最大帮助的邓薇老师，在整个毕业设计准备到答辩的过程中，一直在耐心地帮我解决各种疑问和困惑，一遍遍细心帮我检查论文，从最开始的初稿到之后的修改稿和定稿，指出其中的问题和做的不足的地方，给予我非常专业的指导和建议，真的十分感激您！

还要感谢我的领导和同事们，写毕业论文的过程中给予我关心和理解并一直在身边默默支持着我，感谢相遇。

最后我要感谢我的家人，是我完成毕业设计背后最坚定的精神支柱。

谢谢你们！

# 参考文献

[1] 王强,李俊杰.大数据分析平台建设与应用综述[J].深圳大学高性能计算研究所,2016,5(2),10-12.

[2] 官思发.大数据分析研究现状、问题与对策[J].中国人民大学信息资源管理学院,2015,34(5),5-7.

[3] 陈登鹏.大数据在企业管理中的应用[J].财经界(学术版),2015,11(3),15-20.

[4] 上野宣.图解HTTP[M].北京:人民邮电出版社,2014:40-45.

[5] Nicholas.C.Zakas.Professional Javascript For Web Developers 3Rd[M].New York:John Wiley & Sons inc;3rd Revised edition,2012:75-79.

[6] Kyle Simpson.You don’t known JS[M].New York:O’Reilly Media,Inc,2014:51-55.

[7] 舒伟,刘兵.浅谈大数据在企业中的应用[J].中国电子科技集团公司第二十八研究所,2010(1),22-35.

[8] 韩红蕾.大数据对企业管理决策影响分析[J].广东农工商学院-09 [9] 张耀春.Vue.js权威指南[M].北京:中国工信出版集团,2016:55-60.

[10] 马骥.Vue.js实战[M].北京:清华大学出版社,2017:34-39.