

南通大學



本科毕业设计(论文)报告

题 目	移动会议实时互动系统运营方的设计与实现
--------	---------------------

学生姓名: 杨光

专业: 软件工程

指导教师: 顾卫江

完成日期: 2019年5月22日

诚信承诺书

本人承诺：所呈交的毕业设计是本人在导师指导下进行的研究成果。除了文中特别加以标注和致谢的地方外，其中不包含其他人已发表或撰写过的研究成果。参与同一工作的其他同志对本研究所做的任何贡献均已在文中作了明确的说明并表示了谢意。

签 名: _____ 日期: 2019.05.22

本论文使用授权说明

本人完全了解南通大学有关保留、使用学位论文的规定，即：学校有权保留论文及送交论文复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容。

(保密的论文在解密后应遵守此规定)

学生签名: _____ 指导教师签名: _____ 日期: 2019.05.22

摘要

在传统的会议中，主办方单方面向人们展示信息，单调乏味的语言就像一首催眠曲，人们只能被动地接受，沟通效率也非常低，会议气氛低到了零点，浪费了时间和精力。本项目运用“互联网+”的思维，开发了一款企业管理者和用户可通过权限认定在会议上进行实时互动的移动应用——移动会议实时互动系统。除此之外，从实时互动角度入手，借助微信小程序的优势，利用 web 端创建现场，为精彩纷呈的现场氛围提供了一套可行方案。

在技术方面，选择 Android 平台和微信小程序作为会议主办方和会议参加者的主要体验平台，使用 Spring Boot 框架来开发移动会议实时互动系统管理端。在信息安全方面，使用 Shiro 进行权限控制，MD5 算法进行信息的加密并使用 SSL 协议对网络连接进行加密。在数据管理方面，采用 Oracle 的 MySQL 数据库进行关系型数据库管理。在文件服务器方面，使用分布式文件系统 Fast DFS。在中间件方面，选择 Redis 作为缓存，RabbitMQ 作为消息队列。在音视频方面，使用 Red5 作为流服务器进行直播的推流并选择腾讯云的 WebRTC 进行多个客户端之间的视频会议。在个性化推荐方面，选择当下最热门的机器学习中的 Java 库——Mahout 进行推荐系统的搭建。在交互方面，选择 Spring Boot 自带的 tomcat 服务器进行客户端与服务端之间的数据交互，使用 WebSocket 进行多个客户端间的实时通信。

关键词：在线会议系统；Spring Boot；WebRTC；Red5；Mahout

ABSTRACT

In traditional meetings, the organizers unilaterally show information to people. The tedious language is like a lullaby. People can only passively accept it. The communication efficiency is very low. The atmosphere of the meeting is as low as zero, which wastes time and energy. The project uses the "Internet +" thinking to develop a mobile application real-time interactive system that allows business managers and users to interact in real time through meetings. In addition, from the perspective of real-time interaction, with the advantage of WeChat applet, using the web to create the scene, provides a feasible solution for the exciting scene atmosphere.

In terms of technology, we chose Android platform and WeChat applet as the main experience platform for conference organizers and conference participants, and used Spring Boot framework to develop mobile conference real-time interactive system management terminal. In terms of information security, we use Shiro for access control, MD5 algorithm for encrypting information and SSL protocol for encrypting network connections. In terms of data management, we use Oracle's MySQL database for relational database management. On the file server side, the distributed file system Fast DFS is used. In terms of middleware, Redis is selected as the cache and RabbitMQ is used as the message queue. In terms of audio and video, we use Red5 as the streaming server for live streaming and select Tencent Cloud's WebRTC for video conferencing between multiple clients. In terms of personalized recommendation, we chose the Java library in the most popular machine learning, Mahout, to build the recommendation system. In terms of interaction, we choose Spring Boot's own tomcat server for data interaction between the client and the server, and use WebSocket to perform real-time communication between multiple clients.

Key words: Online Conference System, Spring Boot, WebRTC, Red5,Mahout

目 录

摘要	I
ABSTRACT	II
第一章 绪论	1
1.1 项目开发背景与用户痛点分析	1
1.1.1 针对企业管理者的痛点分析	2
1.1.2 针对用户的痛点分析	3
1.2 主要工作和贡献	4
1.3 论文结构的安排	5
第二章 可行性分析	6
2.1 市场可行性	6
2.2 经济可行性	10
2.3 技术可行性	10
2.4 社会可行性	10
第三章 需求分析	11
3.1 系统顶层需求结构	11
3.2 系统功能模块需求分析	11
第四章 系统的总体设计	16
4.1 开发技术简介	16
4.2 系统架构图	16
4.2.1 系统的逻辑架构	16
4.2.2 系统的物理架构	17
4.3 系统功能图	18
4.4 系统流程图分析	18
4.5 数据库设计	19
4.5.1 E-R 图设计	19
4.5.2 关系表设计	20
第五章 系统详细设计与实现	28
5.1 登陆模块的实现	28
5.1.1 工具类简介与代码实现	28

5.1.2 登陆界面展示	31
5.1.3 登陆模块的流程图	31
5.1.4 登陆模块的代码实现	32
5.2 活动模块的实现.....	34
5.2.1 WebSocket 代码实现.....	34
5.2.2 活动审核界面展示	38
5.2.3 活动审核的代码实现	39
5.3 增删改查功能的代码实现.....	40
5.3.1 相关实体类的设计	40
5.3.2 Lombok 简介与使用	40
5.3.3 Dao 层的接口实现.....	41
5.4 权限控制的代码实现.....	43
5.4.1 核心代码的设计	43
5.4.2 界面的展示及效果	46
5.5 关键词屏蔽功能的设计	46
5.5.1 DFA 算法简介与分析.....	47
5.5.2 核心代码的实现	48
5.6 推荐模块的设计	49
5.6.1 协同过滤算法简介	49
5.6.2 本系统推荐系统的核心代码实现	51
5.7 话题模块	53
5.8 消息模块	56
5.9 活动互动模块	58
5.10 系统模块	61
第六章 系统测试	67
6.1 单元测试	67
6.1.1 单元测试工具简介	67
6.1.2 单元测试用例设计	67
6.2 接口测试	68
6.2.1 接口测试工具简介	68
6.2.2 接口测试结果分析	68

6.3 性能测试.....	70
6.3.1 性能测试工具简介	70
6.3.1 性能测试报告分析	71
第七章 总结与展望	73
7.1 总结.....	73
7.2 展望.....	73
参考文献	74
致 谢	75

第一章 绪论

1.1 项目开发背景与用户痛点分析

在移动互联网时代，移动智能设备广泛地应用于生活之中，任何时间、任何地点召开一个会议利用移动设备快速实现人与人之间的实时互动是非常酷炫的。设想一下，传统的会议中，主办方单方面向人们展示信息，单调乏味的语言就像一首催眠曲，人们只能被动地接受，不知不觉间就睡眼惺忪了，人与人之间的沟通效率也非常低，会议气氛低到了零点，浪费了时间，浪费了精力。这时候，如果想强打精神积极地参加会议，却无法调动自己的积极性，可以登录的平台，根据智能推荐的最新最热的会议活动，自主选择参加；百度地图自动定位搜索附近的人或活动；看到志趣相同的人，互相添加为好友在线畅聊，扩展人脉圈；看到会场内精美的场景可以随手抓拍与他人分享；会议活动开始了，发起弹幕说出自己的心声，疯狂霸屏；主办方发起滚动抽奖，手气不错的你发现大屏幕上有你中奖的名字或者回答问题赢得现金红包，是不是嗨到停不下来？全程如主办方一样参与会议活动，交更多挚友，分享心得。你甚至还可以发布自己的会议活动，邀请大家一起来参与。面对如此有趣酷炫颠覆传统的会议活动你还会犯困吗？让一起嗨起来，燥起来！

结合现有会议模式存在的诸多问题，团队从传统会议与“移动会议”两个角度出发，对两种模式的痛点进行分析定位。痛点定位图如图 1.1 所示。

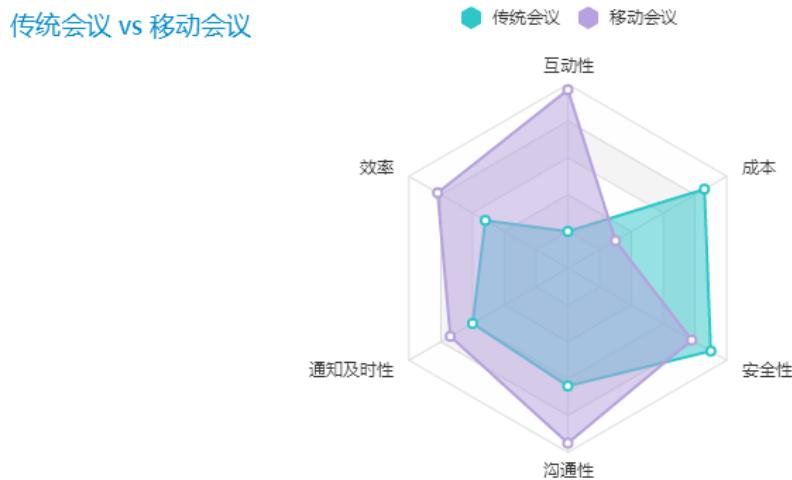


图 1.1 传统会议与移动会议的痛点定位图

在该痛点定位图中：

对传统会议和移动会议中存在的一些问题，优先考虑的方面用雷达图进行了对比，可以清晰的知道传统活动或会议中的局限性。考虑因素如下：

- ✓ **成本：** 用户或者企业举办一次活动所需要的花费是多少？

- ✓ **沟通性:** 会议活动中人与人之间的沟通交流如何？
- ✓ **安全性:** 会议活动是否规范，是否保障用户的人身安全？
- ✓ **效率:** 举办会议活动应该如何有效的安排管理人员？
- ✓ **通知及时性:** 遇到紧急变更的事情能否及时通知到用户？用户能否及时知道？
- ✓ **互动性:** 用户彼此之间的互动性，会议活动举办方与用户之间的互动如何？

传统会议更关注企业自身的利益，人员管理是否到位、如何尽可能降低成本、如何及时通知信息，当然企业也逐渐发现了传统会议中存在的互动性不强的问题。移动会议更加关注活动是否安全、如何提高趣味性、如何增强互动性。

同时，团队从企业管理者与用户双方的需求出发，对当前市场上存在的痛点进行了较为彻底的调研与分析，具体如图 1.2 和图 1.3 所示。

1.1.1 针对企业管理者的痛点分析



图 1.2 针对企业管理者的痛点分析图

(1) 流于形式化

在传统会议模式中，有很大一部分会议流于形式化，为了开会而开会，尤其是在平时公司开会汇报各部门工作时。

(2) 互动性不强

在传统的会议模式中，缺乏一定的互动性。很多会议都是主办方单方面对已布置安排的工作的进行简单强调，会议内容却单薄、空洞；参会人员和讲师缺少互动的渠道，主持人使出浑身解数也难以调动会议氛围，会议气氛低到零点，造成会议体验差，会议质量差。碍于领导的面子，与会者不敢充分发表自己的想法或建议，畏首畏尾；会议的时间极其有限，企业管理者无法在较短的时间里，全面收集与会者的想法和建议，也就无法充分地做出最恰当的决策。

(3) 缺乏统筹安排

会议安排系统性不强，缺乏统筹安排，如果多个部门都选择在同一段时间内开会。与会人数较多时，统计报名人数的工作量比较大，准确度较低，人员的管理力度不到位，参会人员状况难以监控，效果差，往往会产生疏漏，尤其是签到的时候需要排队，浪费时间，浪费人力。

（4）效率低下

企业管理者召开一个会议时，事先没有告诉员工究竟为什么要开会，这使得员工听的稀里糊涂，导致犯困，无法把握会议的目的和结果，因此执行力度低下。会议中，很多决策建议来不及记录，手写的速度比较慢，容易漏掉关键的细节。

（5）成本费用高

开会所需要购置的会议用品占到很大一部分开销，尤其是会议需印刷大量会议资料，而且都是单面印刷，会后并无任何用处，纸张浪费严重，成本代价高。同时会议使得大量员工离开工作岗位，占用太多时间，如果时间安排不好会影响工作。如果会议出现撞车现象，需要重新改期，参加会议的人员的交通费用也会产生一笔开销。

（6）通知不及时

遇到会议变更的情况，或者遇到临时紧急会议，公司需要一个个重新通知参会人员，无法确保所有的参会人员都收到了通知，难免会失误。

1.1.2 针对用户的痛点分析



图 1.3 针对用户的痛点分析图

（1）互动性不强

会议参与者只能被动的接受，心里有话要说时迫于压力只能憋在心里，无法及时大胆的提出自己的看法和建议，只能参与会议，没有一个合适的平台平等自主地发布自己的会议活动。会议内容枯燥无聊，主办方单方面展示的单调乏味会造成会议气氛沉闷，用户会议体验差，会议质量差，使得用户丧失积极性，产生困意以至于听会效率低下。开会时可能一时没有好的想法，会议结束后出现新奇的创意时没有一个合适的平台及时地向大众展示。

（2）沟通效率低

在会议上时常发生参加会议人员之间的矛盾，就某一个问题争论不休导致会议无法进行，大多数情况下是由于人员之间缺乏有效的沟通，“不理解”而造成的。此外，参加同一个会议，人与人之间除了在会上发表自己的想法外并没有深入地交流，如果因为理解不到位造成的工作失误，会严重影响计划的落实，也会影响之后工作的展开。

（3）时间不够灵活

如果有一场很重要的会议和一次出差的时间冲突了，会议参与者无法及时参与会议，可能会导致工作延误，影响工作进度。开会时间拖得太长，时间安排不合理，经常在快下班前临时召开会议，或者会占用员工的休假时间。

（4）缺乏民主性

用户只能单方面的参与会议，自己却没有随意举办会议的权利；很多时候都是听从会议举办方单方面的宣布工作任务，只有服从接受，没有反驳的权利；即使有民主投票权，根据现场的从众心理，大家无法真实的说出自己的真实意见。投票的结果有可能存在暗箱操作，结果不公开不透明，缺乏一定的民主公正性。

（5）内容无趣

开会总是说一些冠冕堂皇的东西，或者是说一些没用的东西浪费时间；大家不是带着方案来开会，而是开会的过程中再去讨论，浪费时间，有可能最后还没有一个好的结果，一番唇枪舌战后还是不欢而散，没有实质的意义。

越来越多的都市人宅在家里，变得愈来愈孤独；而实际上，他们异常渴望找到同行的伴儿，渴望与同类交流，需要组织活动建立并强化关系，打破都市人的孤独。

1.2 主要工作和贡献

本篇论文旨在设计和开发移动会议实时互动系统的运营方管理端。本篇论文使用软件工程的开发流程进行开发，涵盖了可行性分析、需求分析、系统的总体设计和详细设计、编码、测试等环节。

在需求分析阶段，根据会议申请者，会议参加者和管理员的不同需求，分别画出了会议申请者，会议参加者和管理员的用例图。在系统总体设计阶段，在需求分析阶段根据会议申请者，会议参加者和管理员的应用需求，设计出详细的 E-R 图，并以表格的形式对各个数据表进行了详细的介绍。在系统的详细设计中，对各个模块及其子模块进行相应的编程，在整个项目编程结束后，本文给出了部分截图进行相关的展示。在系统功能编程结束后，对系统进行了相应的单元测试、接口测试和压力测试，结果贴在本文的系统测试中。

除了上述功能的实现以外，本文还对运营方管理端运用到的前端 Lay UI 框架、后端 Spring Boot、容器 Docker、中间件 Redis、RabbitMQ、Fast DFS、网络协议 WebRTC、RTMP、推荐系统 Mahout 进行了相关简介。

管理后台使用当下前沿的机器学习中的推荐算法，个性化地对不同的使用者进行个性化信息推荐，更好地提高了用户的使用体验。

1.3 论文结构的安排

本文除第一章绪论，其余部分的组织如下：

第二章通过相关的市场、经济、技术以及社会可行性分析之后，给出了移动会议实时互动系统的可行性分析。

第三章给出了移动会议实时互动系统的功能性需求分析和非功能性需求分析。

第四章是移动会议实时互动系统管理端的总体设计，给出了数据库的 E-R 图设计和相关的数据表，介绍了前端 Lay UI 框架、后端 Spring Boot、容器 Docker、中间件 Redis、RabbitMQ、Fast DFS、网络协议 WebRTC、RTMP、推荐系统 Mahout 等技术的使用。另外，在这一章还给出了移动会议实时互动系统的逻辑架构和物理架构。

第五章介绍了移动会议实时互动系统的详细设计与编程实现，在这一章节中针对一些具体的功能给出了详细的设计思路，编程实现和相关的页面图片展示。

第六章介绍了系统测试，在这一章中对移动会议实时互动系统管理端进行了单元测试、接口测试和压力测试，分别将测试结果分析介绍在文中并分析出该系统可以安全稳定地运行。

第七章是总结与展望，在这一章中总结了移动会议实时互动系统管理端的开发过程，并对移动会议实时互动系统管理端中的不足作出展望。

第二章 可行性分析

2.1 市场可行性

市场可行性是指开发的系统是否市场友好等。主要借助网上调查问卷的方式对市场需求进行调研。分别针对企业管理者和用户设计了两份不同的调查问卷，具体调查问卷见图 2.1 和图 2.2。

企业管理者调查问卷

欢迎参加本次答题感谢您在百忙之中拔冗参加填写本次的调查问卷。您的意见对我们此次调查非常重要，希望能得到您的支持与配合，对此我们表示衷心的感谢！

1. 您的性别是？ [单选题] *	7. 您觉得临时变更活动重新通知是否麻烦？ [单选题] *
<input type="radio"/> 男	<input type="radio"/> 非常麻烦
<input type="radio"/> 女	<input type="radio"/> 有点麻烦
2. 您要举办的活动属于哪种类型？ [多选题] *	<input type="radio"/> 不麻烦
<input type="checkbox"/> 公司级会议	8. 请问您觉得活动中目前存在的主要问题有哪些？ [多选题] *
<input type="checkbox"/> 部门会议	<input type="checkbox"/> 活动氛围过于沉闷
<input type="checkbox"/> 公司年会	<input type="checkbox"/> 活动中安全性无法保障
<input type="checkbox"/> 分享会	<input type="checkbox"/> 互动性不强
<input type="checkbox"/> 娱乐活动	<input type="checkbox"/> 人员难以管理
3. 请问您一般准备购置活动用品开销大吗？ [单选题] *	<input type="checkbox"/> 容易出现撞车等现象
<input type="radio"/> 很少，占得比例不大	<input type="checkbox"/> 通知变更不易通知到每个人
<input type="radio"/> 还行，能接受	
<input type="radio"/> 看似不重要，其实花费颇大	
4. 请问您活动前会把流程发给活动参与者吗？ [单选题] *	9. 您是否愿意通过手机软件来参与活动？ [单选题] *
<input type="radio"/> 不会	<input type="radio"/> 十分愿意
<input type="radio"/> 会	<input type="radio"/> 愿意
5. 您对于会议的氛围感受如何？ [单选题] *	<input type="radio"/> 不愿意
<input type="radio"/> 很好，气氛热烈	10. 您认为活动的展开可以做哪些改变？ [多选题] *
<input type="radio"/> 气氛还不错	<input type="checkbox"/> 增加小游戏、抢红包等全民互动环节
<input type="radio"/> 气氛很差，过于沉闷	<input type="checkbox"/> 加强活动参与者之间的沟通
6. 您觉得以往召开活动的效率如何？ [单选题] *	<input type="checkbox"/> 尽可能减少活动成本
<input type="radio"/> 效率很高	<input type="checkbox"/> 避免形式化
<input type="radio"/> 效率一般	<input type="checkbox"/> 保障安全性
<input type="radio"/> 效率太低	

图 2.1 面向企业管理者使用的调查问卷

用户调查问卷

欢迎参加本次答题感谢您在百忙之中拨冗参加填写本次的调查问卷。您的意见对我们此次调查非常重要，希望能得到您的支持与配合，对此我们表示衷心的感谢！

1. 您的性别是？ [单选题] *

男

女

2. 您所参加的大多数活动的目的是？ [多选题] *

强制参加的活会

讨论解决问题，跟进工作

无聊，想交些朋友

丰富自己的生活

3. 您对于以往活动的氛围感受如何？ [单选题] *

很好，氛围热烈

气氛不错

气氛很差，过于沉闷

4. 参加活动者是否存在迟到早退，接电话，开小会，打瞌睡等 [单选题] *

经常

偶尔

基本不会

5. 活动通知：通知时间过于紧迫，或通知信息不够充分，或信息未及时准确传递活动参与者等 [单选题] *

经常

偶尔

基本不会

6. 您参加的大多数活动的决议 [单选题] *

议而未决

不清晰

有，但未分解成行动计划

7. 您是否愿意通过手机软件来参与活动？ [单选题] *

十分愿意

愿意

不愿意

8. 请问您是通过什么渠道知道并参加活动的？ [多选题] *

朋友推荐的

公司内部举办的

网上搜索的

不太了解，没有合适的平台参与

9. 请问您希望活动能增加哪些形式？ [多选题] *

自主投票的权利

参与红包抽奖

答题赢现金活动

实时发送弹幕，说出自己的心声

加入热门小游戏

什么也不增加

10. 请问您希望活动能为你提供哪些更为方便的服务？ [多选题] *

根据位置搜索活动地点定位

智能推荐感兴趣的活动

保障安全性

高度的隐私性

错过活动后的弥补办法

图 2.2 面向用户使用的调查问卷

基于上述结果，绘制了相应的基于半径的南丁格尔玫瑰图（如图 2.3 所示）和饼状图（如图 2.4 所示），发现企业管理者在考虑活动的开展时更加注重对人员管理的安排，以及活动的氛围、消息的通知是否及时、是否会出现撞车现象，忽视了对于用户的考虑。但他们也意识到了活动中存在的一些问题，并针对问题提出了一些恰当的解决方案。

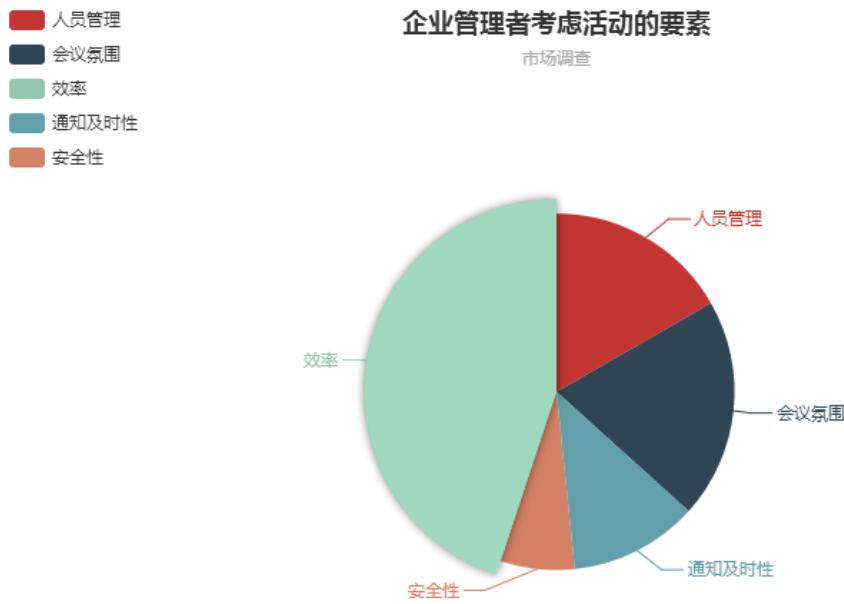


图 2.3 企业管理人员的结果分析图

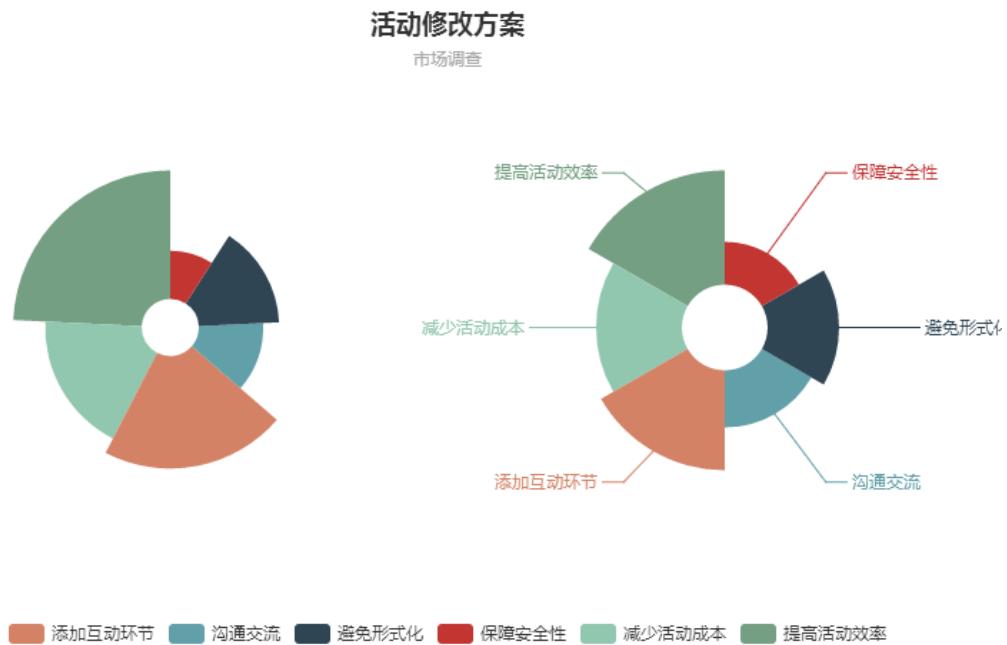


图 2.4 会议方案的饼状结果分析图

基于上述结果，又绘制了相应的标准环形图（如图 2.5 所示）、南丁格尔玫瑰图（如图 2.6 所示）、饼状图（如图 2.7 所示），发现：绝大多数的用户缺乏一定的平台去寻找到自己喜欢的活动并加入进来，在网上搜索的活动也大都存在安全隐患。对于活动的期待，用户喜欢最新最热门的答题活动，也更在乎自己的民主权利，比如投票、实时发送弹幕说出自己的心声。希望能有一个平台为其智能推荐安全可靠、方便有趣的服务，并且他们中的大部分都愿意使用手机软件来寻找参加活动。

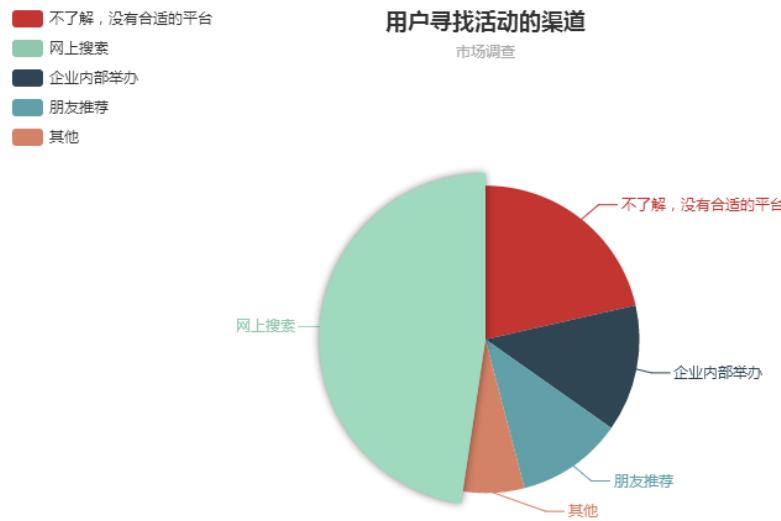


图 2.5 用户寻找活动的渠道分析图

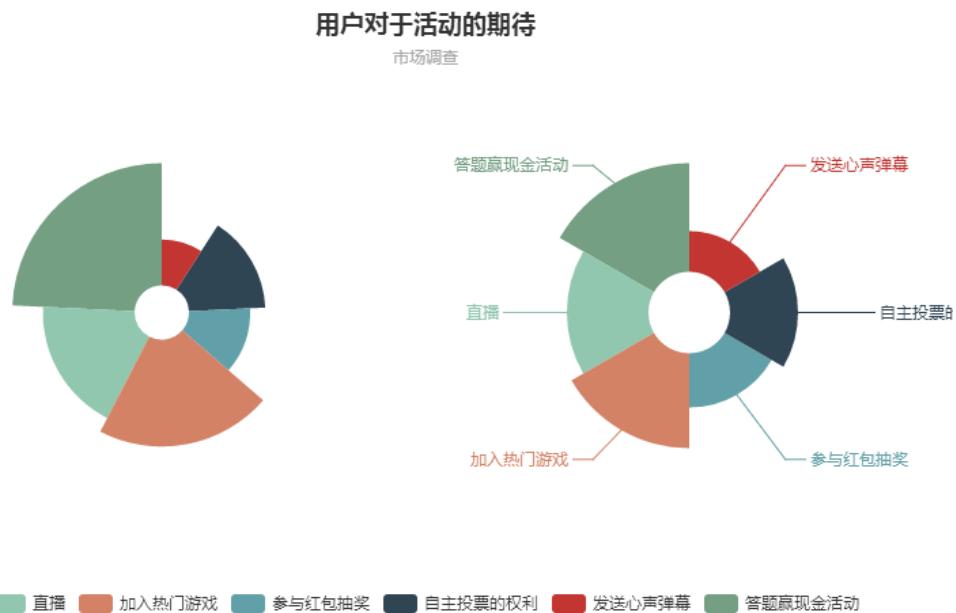


图 2.6 用户对于活动的期待分析图

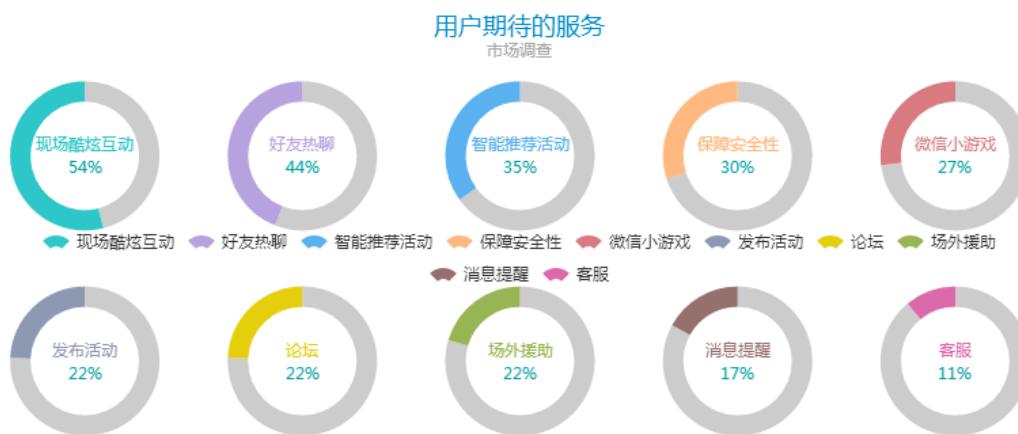


图 2.7 用户期待服务的饼状图

根据上述调查分析结果，设计并且开发移动会议实时互动系统，它能够满足企业管理者和用户的真实需求，有助于提高会议的氛围。

2.2 经济可行性

经济可行性是指系统投入市场使用后，获得的经济利润大于其本身的成本开发。本项目的主要成本和预期收益主要包括：

- 开发测试期间成本投入较少，主要是电费、服务器费用等成本。
- 预期收益分为线上盈利模式和线下盈利模式。其中线上盈利主要考虑会员收益、广告收益、衍生收益、数据服务收益。而线下盈利模式主要考虑与市场店铺（例如会议场地出租等）进行合作。

2.3 技术可行性

本系统在技术可行性方面主要有：

- 安卓开发技术非常成熟、J2EE 开发技术非常成熟、微信小程序开发非常成熟。
- 项目组开发成员具备良好的开发能力和自学能力。
- 测试人员掌握主流的测试方法以及相应测试工具的使用。

2.4 社会可行性

社会可行性指系统开发是否会导致任何侵权行为、妨碍性后果和责任。从系统的完成度来看，本项目的社会可行性主要体现在：

- 无软件权利归属问题
- 无侵权问题
- 与国家的政策法规不存在任何冲突和抵触之处

第三章 需求分析

3.1 系统顶层需求结构

移动会议实时互动系统管理端分为活动、话题、广告、消息、推荐、用户和系统模块这七个模块，图 3.1 为本系统的顶层需求结构。

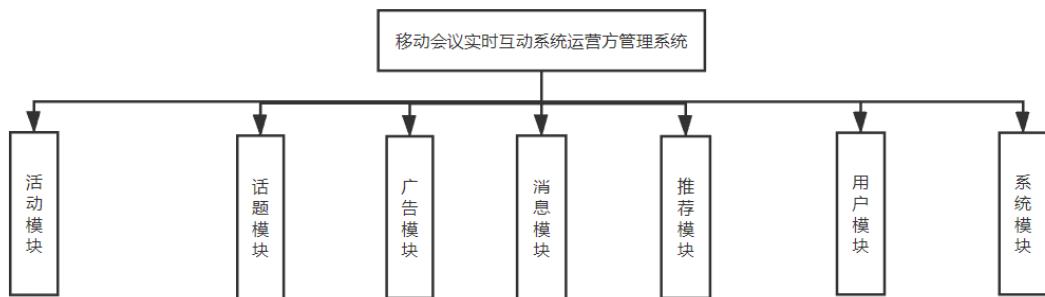


图 3.1 移动会议实时互动系统运营方管理系统顶层需求结构图

如图所示，活动模块包括了管理员对会议活动的管理；话题模块包括了管理员对话题的管理；广告模块包括了管理员对广告的管理；消息模块包括了管理员对消息的管理；推荐模块包括了管理员对个性化推荐的管理；用户模块包括了管理员对用户的管理；系统模块包括了管理员对系统的管理。

3.2 系统功能模块需求分析

(1) 活动模块：活动模块是系统的核心模块，主要功能是针对已经实名认证过的用户，他们申请的会议活动进行管理：当用户申请会议活动后，运营方管理后台会出现信息提示“您有新的活动需要审核”，此时管理员需要审核相关会议活动。若通过，给该活动分配一个直播房间或者视频房间并以邮件的方式将房间号发送给申请的用户以供其进行移动会议互动；若不通过，管理员会写明原因并发送到申请用户的邮箱。管理员可以在后台将所有的会议活动信息导出到计算机或者直接打印出来。

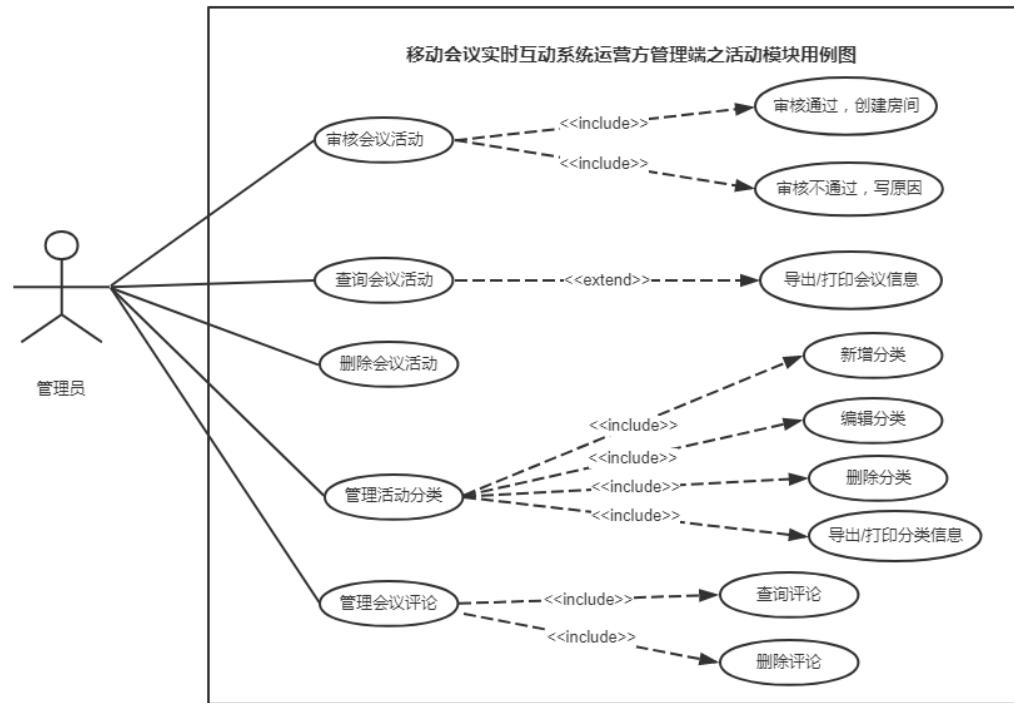


图 3.2 活动模块用例图

(2) 话题模块：话题模块主要是提供一个用户与管理员之间互动交流的论坛平台。

管理员可以在后台发布新的话题供用户之间相互讨论。系统会对它们进行关键字的屏蔽，同时对与话题不相关的讨论进行删除等操作以确保这个平台的网络信息文明。管理员可以在后台将所有的话题信息导出到计算机或者直接打印出来。

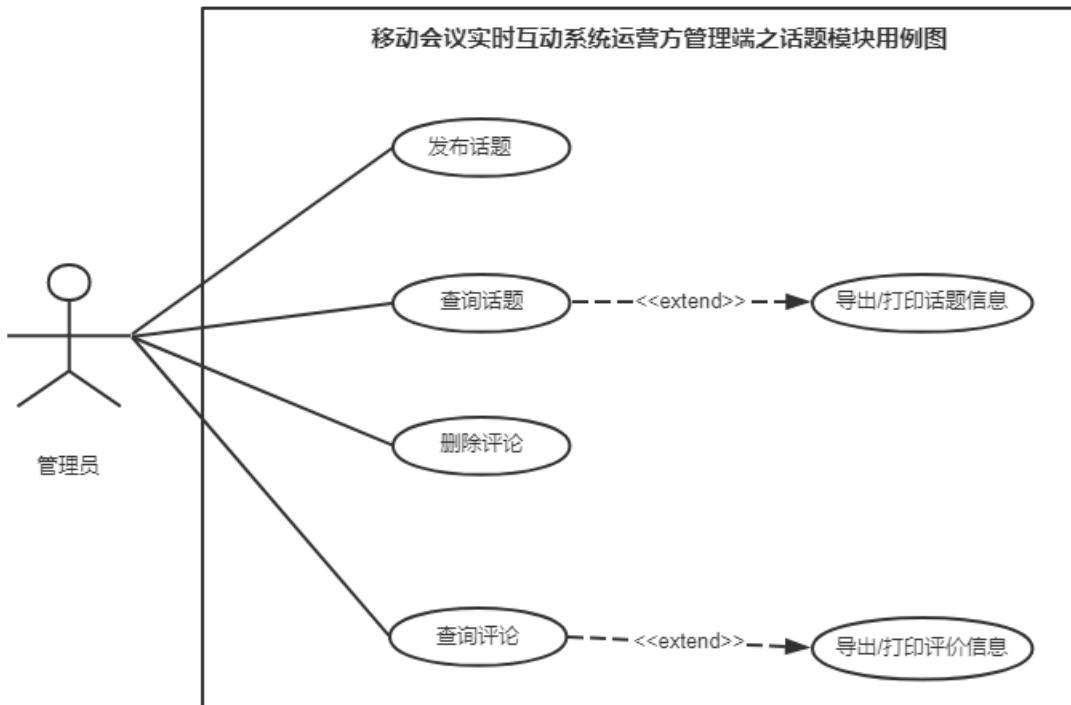


图 3.3 话题模块用例图

(3) 广告模块：管理员可以向客户端以轮播图的形式推送广告，除此之外管理员还

可以对广告进行增删改查以及给予其是否展示的能力。管理员可以在后台将所有的广告活动信息导出到计算机或者直接打印出来。

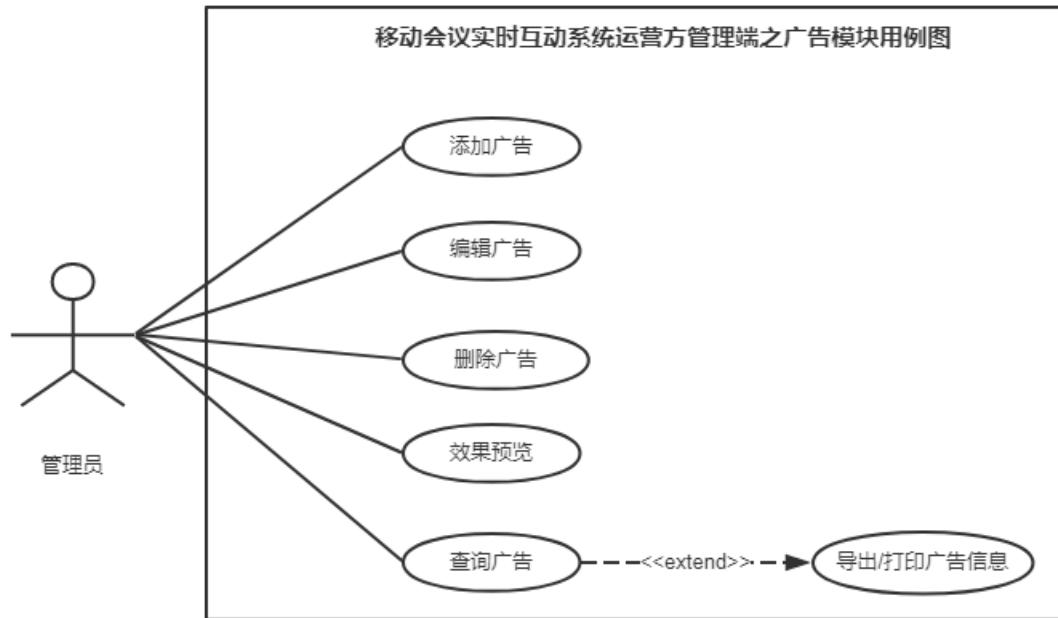


图 3.4 广告模块用例图

(4) 消息模块：当用户进行注册、进行实名认证、申请会议活动等操作时，管理员会出现在实时的提示功能，提示有新的认证或审核功能；或目前管理员离线状态，当他上线时，依然会提示相关消息，并且消息存入到数据库中，管理员可进行已阅读和删除操作。消息模块的用例图如图 3.5 所示。

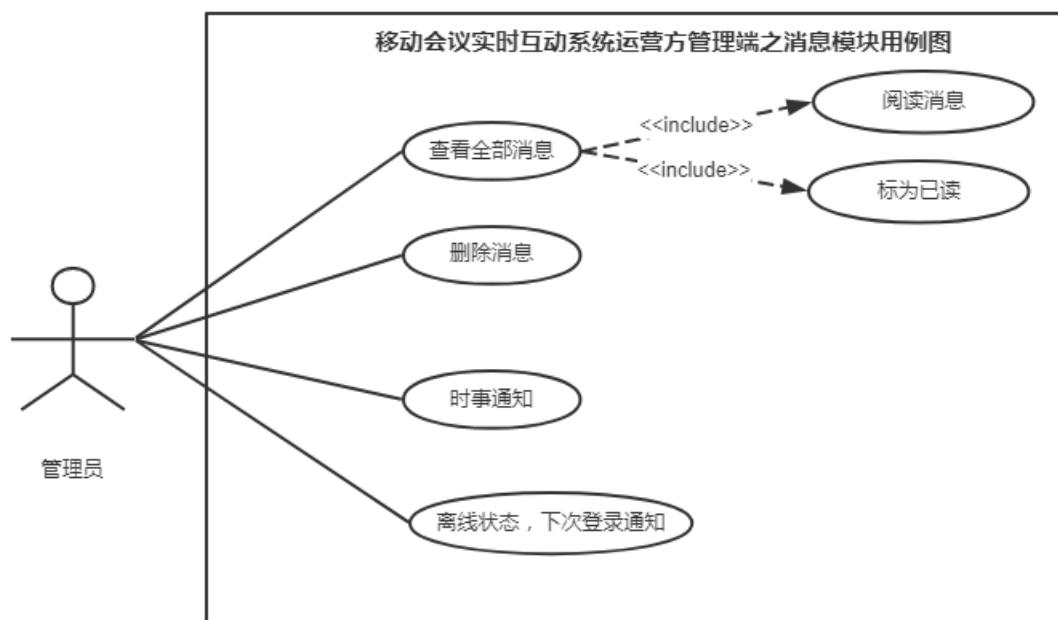


图 3.5 运营方管理端消息模块用例图

(5) 推荐模块：为了获得更好的用户体验，设计了推荐模块。该模块分为 2 大

类型：热门推荐和个性化推荐。热门推荐是管理员编辑选择一个会议活动作为热门会议活动推荐到客户端的首页；个性化推荐会根据用户的操作如参与会议、收藏会议等等进行权值的叠加从而智能推算出用户的个性化推荐会议活动。推荐模块的用例图如图 3.6 所示。

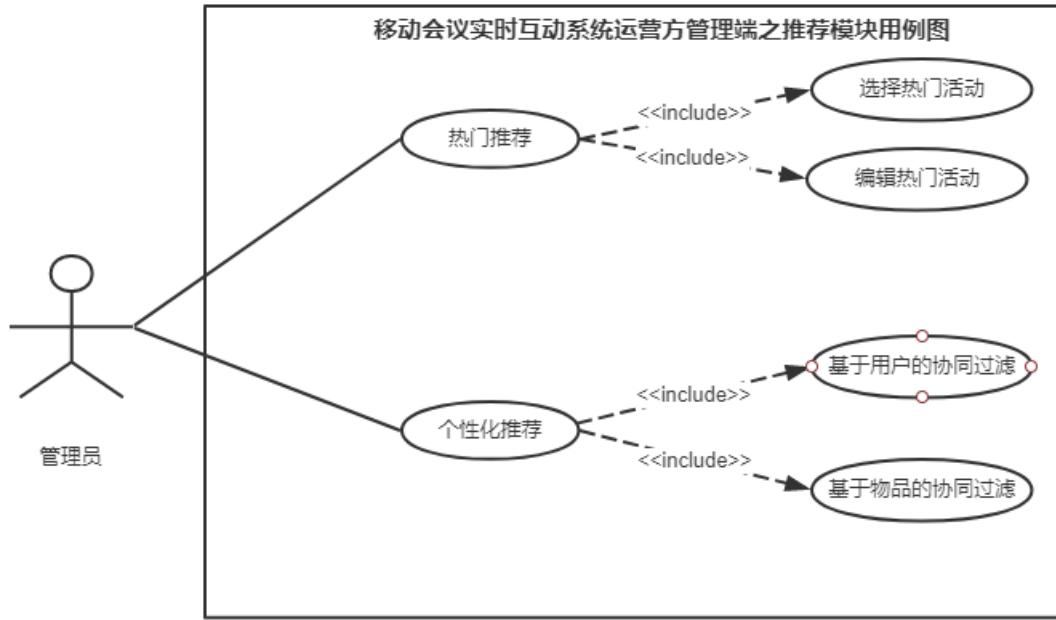


图 3.6 推荐模块用例图

(6) 用户模块：用户模块中包括了系统用户的管理，运营方管理员的管理以及系统权限的管理。系统由这 3 个子模块来构成了本项目的用户模块和权限安全。系统用户必须先将他的身份证件正反面以及邮箱留出进行实名认证，管理员可以进行用户实名认证的审核；审核通过，发邮件告知；否则，发送不通过原因；以 Shiro 框架进行管理员对菜单的一个权限控制，其中拥有最高权限的管理员可以进行低权限级别的管理员的信息和权限。管理员可以在后台将所有的用户信息、管理员信息导出到计算机或者直接打印出来。用户模块的用例图如图 3.7 所示。

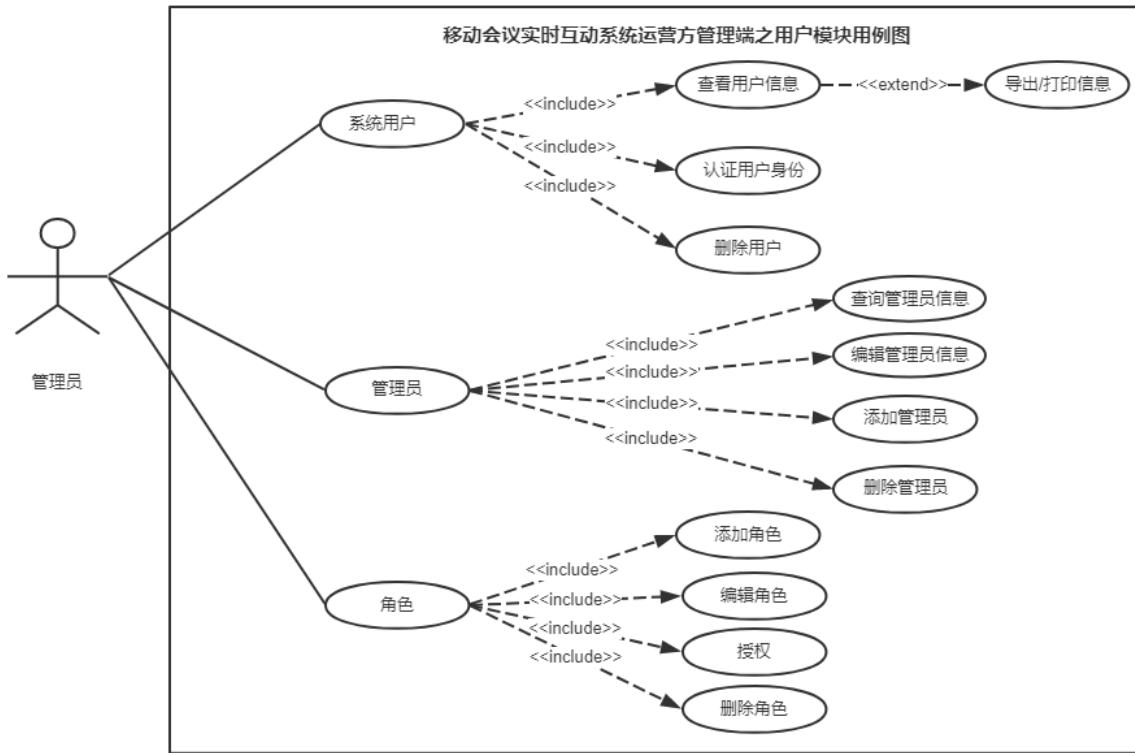


图 3.7 运营方管理端用户模块用例图

(7) 系统模块：系统模块是系统中的基础模块。管理员可以进行系统的相关监控：Spring Boot 监控、消息队列监控、数据库监控、日志监控；管理员还可以进行网站的相关设置，系统文件服务器中文件的上传和删除，系统的敏感词的设置和删除；管理员还能修改自己的相关信息等等。系统模块的用例图如图 3.8 所示。

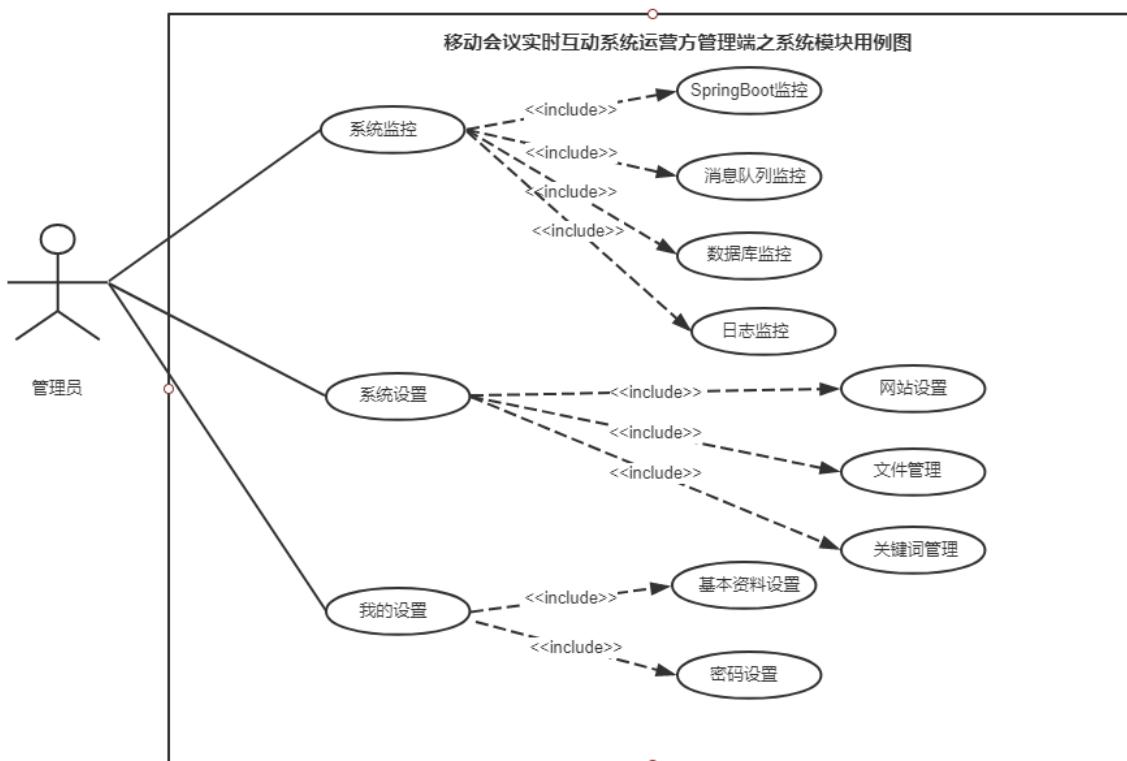


图 3.8 运营方管理端系统模块用例图

第四章 系统的总体设计

4.1 开发技术简介

本节介绍移动会议实时互动系统管理端时用到的一些技术和开发环境。用到的开发技术主要包括：

- (1) 前端 Lay UI 框架；
- (2) 后端 Spring Boot；
- (3) 容器 Docker；
- (4) 中间件 Redis、RabbitMQ、Fast DFS；
- (5) 网络协议 WebRTC、RTMP；
- (6) 推荐系统 Mahout 等。

系统开发环境主要是：开发工具 IntelliJ IDEA 2018.3.2 x64、操作系统 Win10、CPU i7-8700、内存 16G。

4.2 系统架构图

4.2.1 系统的逻辑架构

移动会议实时互动系统的整体逻辑架构分为数据 Model 层、业务逻辑 Service 层、控制 Controller 层、表示 View 层和持久 Dao 层五层架构，如图 4.6 所示。其中表示 View 层主要是 Web 前端的展示，代码主要是 HTML，CSS，JavaScript 等。控制层主要用于接口的编写，数据全部以 json 格式传输。业务逻辑层包括了活动、话题、广告、消息、推荐、用户和系统模块这七个模块的业务逻辑。数据访问层主要是用于 domain 实体类的编写。持久层以 Spring Boot 内置的 JDBC Templet 进行数据库的读写操作。

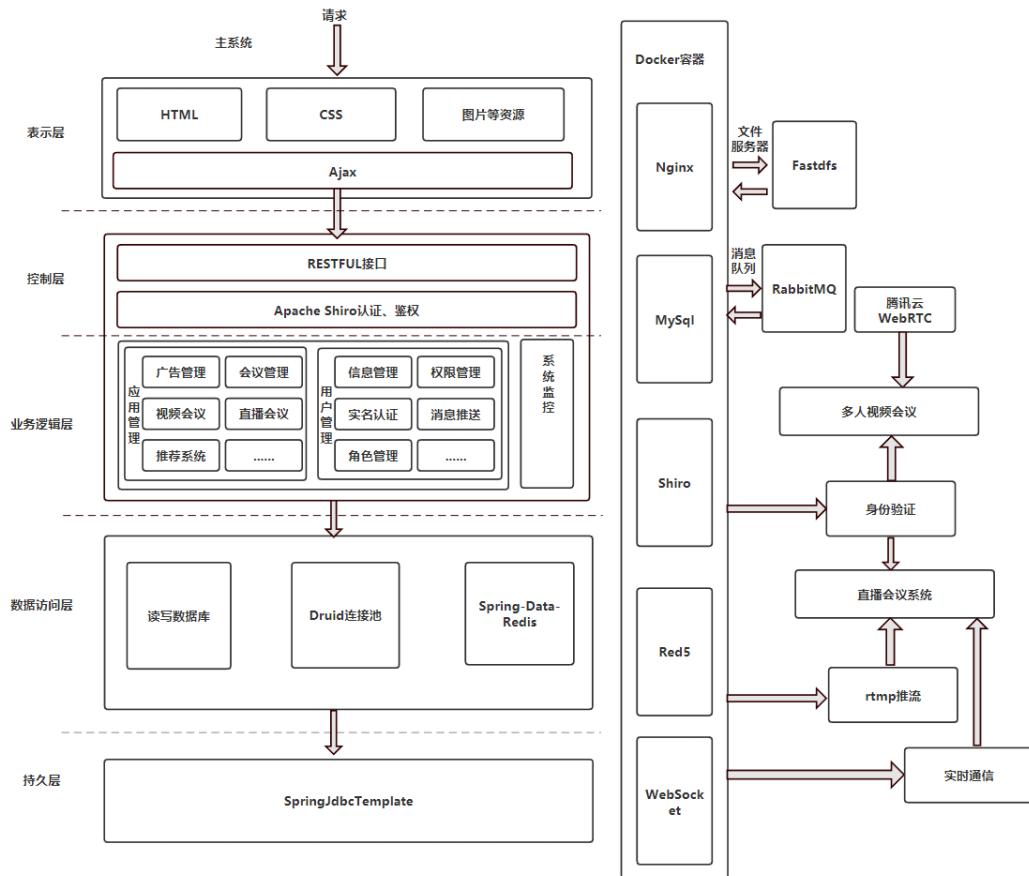


图 4.6 系统的逻辑架构

4.2.2 系统的物理架构

移动会议实时互动系统的物理架构主要有：管理端和用户端。其中，管理端基于 Spring Boot，用户端都基于安卓和微信小程序。移动会议实时互动系统的物理架构如图 4.7 所示：

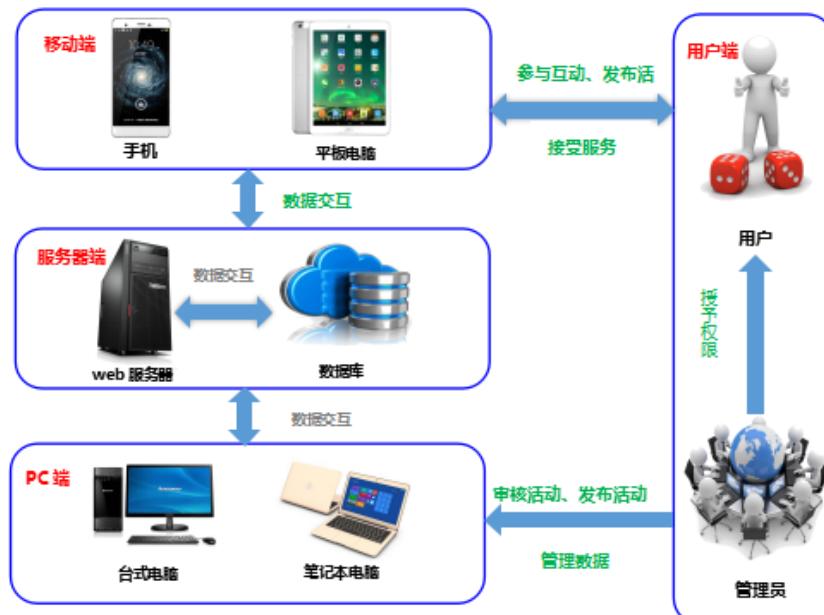


图 4.7 移动会议实时互动系统的物理架构

4.3 系统功能图

的移动会议实时互动平台主要由客户端（基于微信小程序和安卓手机端 APP）和 Web 管理端（基于 Web 服务器 Apache Tomcat、MySQL 数据库）这两个子系统所组成，这两个系统之间通过 http 的 request 请求和 response 响应以 json 形式进行数据交互，共同实现了移动会议实时互动平台的功能。其中，本篇论文主要是研究管理端的设计与实现，就管理端的功能模块进行重点介绍。

管理端端主要包括七大主要的功能模块：活动、话题、广告、消息、推荐、用户和系统模块。在这七大主要模块下还有相应的子模块，通过实现子模块，从而实现整个系统的功能，方便地进行业务处理。

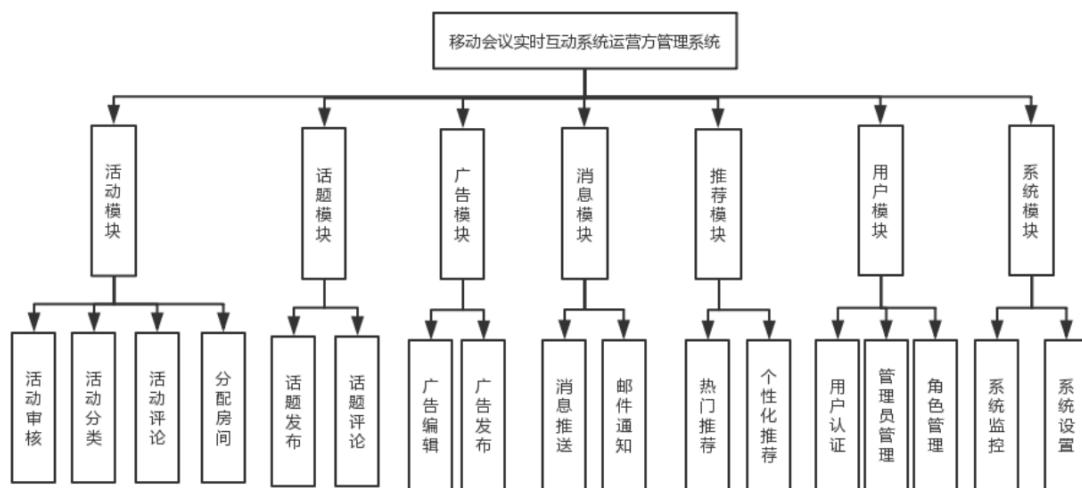


图 4.8 移动会议实时互动系统功能图

4.4 系统流程图分析

图 4.9 展示了移动会议实时互动系统的流程图。客户端和后台端交互流程如下：用户登录平台后需进行实名认证，否则平台的相关功能会受到限制，如不能申请、加入活动等等。（步骤 1,2）管理端登录后台进行用户的身份审核，审核通过后以邮箱的形式通知用户（步骤 3,4）。管理员可以发布热点话题，（步骤 5,6）用户在客户端内进行话题讨论（步骤 7,8）。用户可以申请一项新的会议活动，（步骤 9,10）管理员在后台审核后进行房间的分配并将活动显示在活动列表中（步骤 11,12）供其他用户参与、收藏（步骤 13,14）。会议开始之前，与会用户需要使用客户端扫一扫功能进行会议的签到（步骤 15）。在会议期间，用户可以发送弹幕进行讨论互动（步骤 16），会议活动申请者以后进行抽奖环节提高会议的气氛（步骤 17）。会议结束后，与会用户可以以文字和图片的形式进行活动的评论（步

骤 18, 19)。

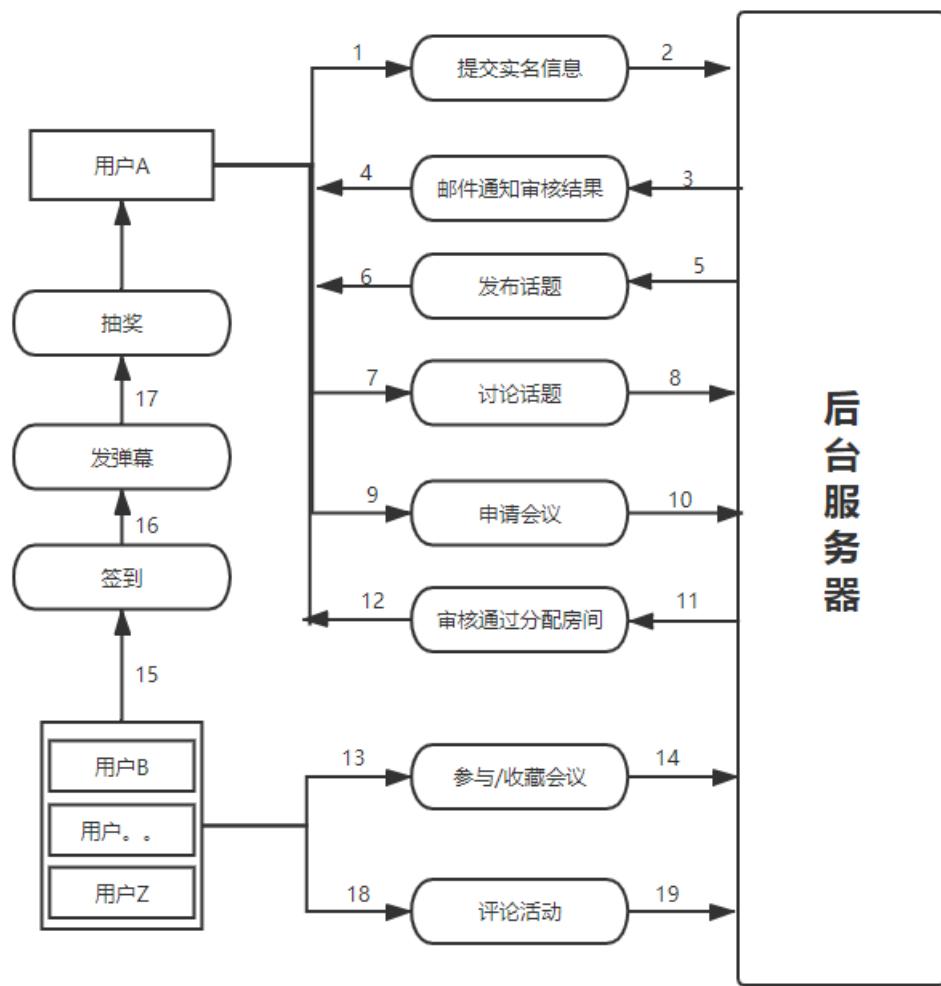


图 4.9 移动会议实时互动系统流程图

4.5 数据库设计

4.5.1 E-R 图设计

根据上文移动会议实时互动系统的需求分析，本系统的 E-R 图如下图所示。

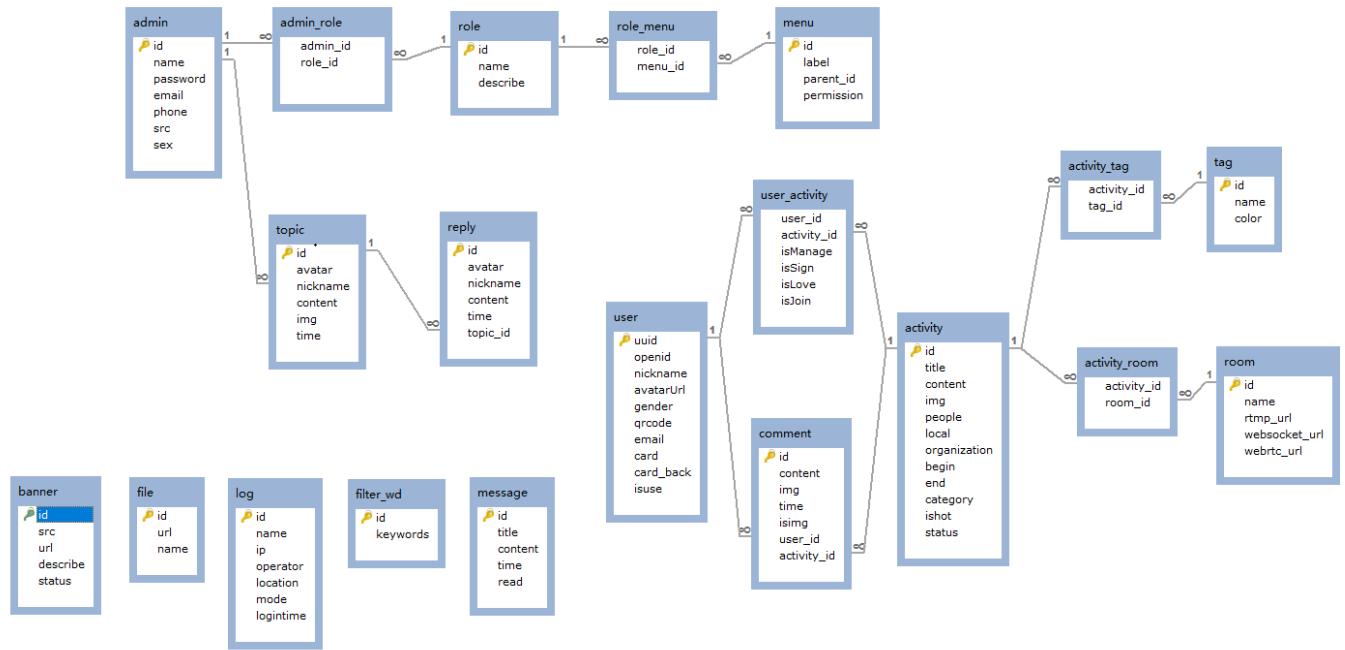


图 4.10 系统整体 E-R 图

4.5.2 关系表设计

1、activity 活动表

基本信息: InnoDB, utf8_general_ci

表 4.1 活动表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, 活动 ID
2	title	varchar(255)	NULL		活动标题
3	content	text	NULL		活动内容
4	img	varchar(255)	NULL		封面图片
5	people	int(20)	NULL		人数限制
6	local	varchar(255)	NULL		活动地点
7	organization	varchar(255)	NULL		组织者
8	begin	varchar(255)	NULL		开始时间
9	end	varchar(255)	NULL		结束时间
10	category	int(20)	NULL		0rtmp, 1WebRTC
11	ishot	int(20)	NULL	0	是否热门
12	status	int(20)	NULL	0	0 未审核, 1 已审核

2、activity_room 活动_房间表

基本信息：InnoDB, utf8_general_ci

表 4.2 活动_房间表

序列	列名	类型	可空	默认值	注释
1	activity_id	int(20)	NULL		活动 ID
2	room_id	int(20)	NULL		房间 ID

3、activity_tag 活动_标签表

基本信息：InnoDB, utf8_general_ci

表 4.3 活动_标签表

序列	列名	类型	可空	默认值	注释
1	activity_id	int(20)	NULL		活动 ID
2	tag_id	int(20)	NULL		标签 ID

4、admin 管理员表

基本信息：InnoDB, utf8_general_ci

表 4.4 管理员表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, ID
2	name	varchar(255)	NOT NULL		管理员昵称
3	password	varchar(255)	NULL		管理员密码
4	email	varchar(255)	NULL		管理员邮箱
5	phone	varchar(255)	NULL		管理员手机
6	src	varchar(255)	NULL		管理员头像
7	sex	varchar(20)	NULL		管理员性别

5、admin_role 管理员_角色表

基本信息：InnoDB, utf8_general_ci

表 4.5 管理员_角色表

序列	列名	类型	可空	默认值	注释
1	admin_id	int(20)	NULL		管理员 ID
2	role_id	int(20)	NULL		角色 ID

6、banner 广告表

基本信息：InnoDB, utf8_general_ci

表 4.6 广告表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，广告 ID
2	src	varchar(255)	NULL		广告图片
3	url	varchar(255)	NULL		广告地址
4	describe	varchar(255)	NULL		描述
5	status	int(20)	NULL	0 不展示, 1 展示	

7、comment 会议评论表

基本信息：InnoDB, utf8_general_ci

表 4.7 会议评论表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，活动评论 ID
2	content	varchar(255)	NULL		评论内容
3	img	varchar(255)	NULL		评论图片
4	time	varchar(255)	NULL		评论时间
5	isimg	int(20)	NULL		是否是图片
6	user_id	varchar(255)	NULL		评论用户 ID
7	activity_id	int(20)	NULL		活动 ID

8、file 文件表

基本信息: InnoDB, utf8_general_ci

表 4.8 文件表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, 文件 ID
2	url	varchar(255)	NULL		文件地址
3	name	varchar(255)	NULL		文件名

9、filter_wd 关键字表

基本信息: InnoDB, utf8_general_ci

表 4.9 关键字表

序列	列名	类型	可空	默认值	注释
1	id	int(10)	NOT NULL		主键, 关键字 ID
2	keywords	varchar(255)	NULL		关键字

10、log 登陆日志表

基本信息: InnoDB, utf8_general_ci

表 4.10 登陆日志表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, 日志 ID
2	name	varchar(255)	NULL		用户名
3	ip	varchar(255)	NULL		ip
4	operator	varchar(255)	NULL		运营商
5	location	varchar(255)	NULL		归属地
6	mode	varchar(255)	NULL		手机/账号密码
7	logintime	datetime	NULL		登录时间

11、menu 菜单表

基本信息：InnoDB, utf8_general_ci

表 4.11 菜单表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，菜单 ID
2	label	varchar(255)	NULL		菜单名
3	parent_id	int(20)	NULL		父 ID
4	permission	varchar(255)	NULL		权限名

12、message 消息表

基本信息：InnoDB, utf8_general_ci

表 4.12 消息表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，通知 ID
2	title	varchar(255)	NULL		通知标题
3	content	varchar(255)	NULL		通知内容
4	time	datetime	NULL		通知时间
5	read	int(2)	NULL		是否已读

13、reply 回复表

基本信息：InnoDB, utf8_general_ci

表 4.13 话题回复表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，回复 ID
2	avatar	varchar(255)	NULL		回复者头像
3	nickname	varchar(255)	NULL		回复者昵称
4	content	varchar(255)	NULL		回复内容
5	time	varchar(255)	NULL		回复时间
6	topic_id	int(20)	NULL		话题 ID

14、role 角色表

基本信息：InnoDB, utf8_general_ci

表 4.14 角色表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键角色 ID
2	name	varchar(255)	NULL		角色名
3	describe	varchar(255)	NULL		描述

15、role_menu 角色_菜单表

基本信息：InnoDB, utf8_general_ci

表 4.15 角色_菜单表

序列	列名	类型	可空	默认值	注释
1	role_id	int(20)	NULL		角色 ID
2	menu_id	int(20)	NULL		菜单 ID

16、room 房间表

基本信息：InnoDB, utf8_general_ci

表 4.16 房间表表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键，房间 ID
2	name	varchar(255)	NULL		房间名
3	rtmp_url	varchar(255)	NULL		rtmp 地址
4	WebSocket_url	varchar(255)	NULL		WebSocket 地址
5	WebRTC_url	varchar(255)	NULL		WebRTC 地址

17、tag 标签表

基本信息：InnoDB, utf8_general_ci

表 4.17 标签表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, 标签 ID
2	name	varchar(255)	NULL		标签名
3	color	varchar(255)	NULL		标签颜色

18、topic 话题表

基本信息：InnoDB, utf8_general_ci

表 4.18 话题表

序列	列名	类型	可空	默认值	注释
1	id	int(20)	NOT NULL		主键, 话题 ID
2	avatar	varchar(255)	NULL		话题发起者头像
3	nickname	varchar(255)	NULL		话题发起者昵称
4	content	varchar(255)	NULL		话题内容
5	img	varchar(255)	NULL		封面图片
6	time	varchar(255)	NULL		话题时间

19、user 用户表

基本信息：InnoDB, utf8mb4_unicode_ci

表 4.19 用户表

序列	列名	类型	可空	默认值	注释
1	uuid	varchar(255)	NOT NULL		用户 ID
2	openid	varchar(255)	NULL		微信 ID
3	nickname	varchar(255)	NULL		昵称
4	avatarUrl	varchar(255)	NOT NULL		头像
5	gender	int(20)	NULL		性别
6	qrcode	varchar(255)	NULL		二维码
7	email	varchar(255)	NULL		邮箱
8	card	varchar(255)	NULL		身份证正面
9	card_back	varchar(255)	NULL		身份证反面
10	isuse	int(20)	NULL	0 未实名, 1 实名	

20、user_activity 用户_活动表

基本信息： InnoDB, utf8_general_ci

表 4.20 用户_活动表

序列	列名	类型	可空	默认值	注释
1	user_id	varchar(255)	NULL		用户 ID
2	activity_id	int(20)	NULL		活动 ID
3	isManage	int(20)	NULL		是否是申请者
4	isSign	int(20)	NULL		是否签到
5	isLove	int(20)	NULL		是否收藏
6	isJoin	int(20)	NULL		是否参与

第五章 系统详细设计与实现

本章介绍了移动会议实时互动系统管理端的详细设计与实现。本章通过相关的核心代码、系统运行图对移动会议实时互动系统管理端的活动、话题、广告、消息、推荐、用户和系统模块这七大模块进行展示。其中 Dao 层数据的增加、删除、修改、查询的基本操作以广告模块为例进行展示。

5.1 登陆模块的实现

移动会议实时互动管理端设置了两种登陆方式：用户名/密码登陆和手机验证码登陆。用户进入登陆页面生成的验证码存入 Redis 缓存中并设置有效期为 3 分钟，这样有效的避免了资源的浪费，当登陆成功时，使用消息队列将登录信息异步存入日志中。同样，如果管理员使用手机登陆，手机号获取的验证码也将在缓存中存在 3 分钟。

5.1.1 工具类简介与代码实现

登陆模块中使用了 3 个工具类，分别是验证码生成工具类、手机验证码工具类和 IP 地址解析工具类。其中验证码生成工具类是自己实现，通过 StringBuffer, BufferedImage, Graphics 这三个类将验证码生成，存在 img 图片中并返回到页面中。手机验证码工具类根据第三方 SDK 文档实现，通过秒滴云 SDK 进行工具类的编写，秒滴云对新用户会赠送 10 元的体验现金，大大降低了使用短信通知 API 的成本。IP 地址类通过 JSOUP 对第三方网站的 IP 解析进行使用，以爬虫的方法来解析管理员登陆的地点以及运营商的信息，同样大大降低了开发成本，避免了第三方 API 的付费使用。关键代码如下：

验证码生成工具类核心代码：

```

1. public class VerifyUtil {
2.     // 验证码字符集
3.     private static final char[] chars = {'1', '2', '3', '4', '5', '6', '7', '8', '9'};
4.     private static final int SIZE = 4;
5.     private static final int LINES = 5;
6.     private static final int WIDTH = 80;
7.     private static final int HEIGHT = 40;
8.     private static final int FONT_SIZE = 30;
9.     public static Object[] createImage() {
10.         StringBuffer sb = new StringBuffer();
11.         BufferedImage image = new BufferedImage(
12.             WIDTH, HEIGHT, BufferedImage.TYPE_INT_RGB);
13.         Graphics graphic = image.getGraphics();

```

```

14.     graphic.setColor(Color.LIGHT_GRAY);
15.     graphic.fillRect(0, 0, WIDTH, HEIGHT);
16.     Random ran = new Random();
17.     for (int i = 0; i < SIZE; i++) {
18.         int n = ran.nextInt(chars.length);
19.         graphic.setColor(getRandomColor());
20.         graphic.setFont(new Font(
21.             null, Font.BOLD + Font.ITALIC, FONT_SIZE));
22.         graphic.drawString(
23.             chars[n] + "", i * WIDTH / SIZE, HEIGHT * 2 / 3);
24.         sb.append(chars[n]);
25.     }
26.     for (int i = 0; i < LINES; i++) {
27.         graphic.setColor(getRandomColor());
28.         graphic.drawLine(ran.nextInt(WIDTH), ran.nextInt(HEIGHT),
29.                         ran.nextInt(WIDTH), ran.nextInt(HEIGHT));
30.     }
31.     return new Object[]{sb.toString(), image};
32. }
33.
34. public static Color getRandomColor() {
35.     Random ran = new Random();
36.     Color color = new Color(ran.nextInt(256),
37.                             ran.nextInt(256), ran.nextInt(256));
38.     return color;
39. }
40. }

```

手机验证码工具类核心代码：

```

1. public static String getCode(String phone){
2.     String rod=smsCode();
3.     String timestamp=getTimestamp();
4.     String sig=getMD5(ACCOUNT_SID,AUTH_TOKEN,timestamp);
5.     String tamp="【云智教育】您的验证码为"+rod+", 请于"+3+"分钟内正确输入, 如非本人操作,
   请忽略此短信。";
6.     OutputStreamWriter out=null;
7.     BufferedReader br=null;
8.     StringBuilder result=new StringBuilder();
9.     try {
10.         URL url=new URL(QUERY_PATH);
11.         HttpURLConnection connection=(HttpURLConnection) url.openConnection();
12.         connection.setRequestMethod("POST");
13.         connection.setDoInput(true); //设置是否允许数据写入
14.         connection.setDoOutput(true); //设置是否允许参数数据输出
15.         connection.setConnectTimeout(5000); //设置链接响应时间

```

```

16.         connection.setReadTimeout(10000); //设置参数读取时间
17.         connection.setRequestProperty("Content-type","application/x-www-form-urlencoded");
18.         out=new OutputStreamWriter(connection.getOutputStream(),"UTF-8");
19.         String args=getQueryArgs(ACCOUNT_SID, tamp, phone, timestamp, sig, "JSON")
20. ;
21.         out.write(args);
22.         out.flush();
23.         br=new BufferedReader(new InputStreamReader(connection.getInputStream(),"UTF-8"));
24.         String temp="";
25.         while((temp=br.readLine())!=null){
26.             result.append(temp);
27.         }
28.     } catch (Exception e) {
29.         // TODO Auto-generated catch block
30.         e.printStackTrace();
31.     }
32.     JSONObject json=JSONObject.fromObject(result.toString());
33.     String respCode=json.getString("respCode");
34.     String defaultRespCode="00000";
35.     if(defaultRespCode.equals(respCode)){
36.         return rod;
37.     }else{
38.         return defaultRespCode;
39.     }
40.     public static String getQueryArgs(String accountSid,String smsContent,String to,String timestamp,String sig,String respDataType){
41.         return "accountSid="+accountSid+"&smsContent="+smsContent+"&to="+to+"&tamp="+t
42.         imESTAMP+"&sig="+sig+"&respDataType="+respDataType;
43.     }
44.     public static String getTimestamp(){
45.         return new SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
46.     }
47.     public static String getMD5(String sid,String token,String timestamp){
48.         StringBuilder result=new StringBuilder();
49.         String source=sid+token+timestamp;
50.         try {
51.             MessageDigest digest=MessageDigest.getInstance("MD5");
52.             byte[] bytes=digest.digest(source.getBytes());
53.             for(byte b:bytes){
54.                 String hex=Integer.toHexString(b&0xFF);
55.                 if(hex.length()==1){
56.                     result.append("0"+hex);
57.                 }else{
58.                     result.append(hex);
59.                 }
60.             }
61.         } catch (Exception e) {
62.             e.printStackTrace();
63.         }
64.         return result.toString();
65.     }

```

```

57.             result.append(hex);
58.         }
59.     }
60. } catch (NoSuchAlgorithmException e) {
61.     e.printStackTrace();
62. }
63. return result.toString();
64. }
65. public static String smsCode(){
66.     String random=(int)((Math.random()*9+1)*100000)+"";
67.     return random;
68. }
69. }

```

5.1.2 登陆界面展示

在登陆页使用了 i18n 国际化配置并进行了人性化的交互体验，右上角有 2 个按钮：“English”，“中文”，点击就会切换到不同的语言页面。当用户输入错误的密码会提示密码错误，输入错误的用户名会提示用户名不存在，输入错误的验证码或验证码过期也会有相应的提示，效果图如图 5.1 所示。

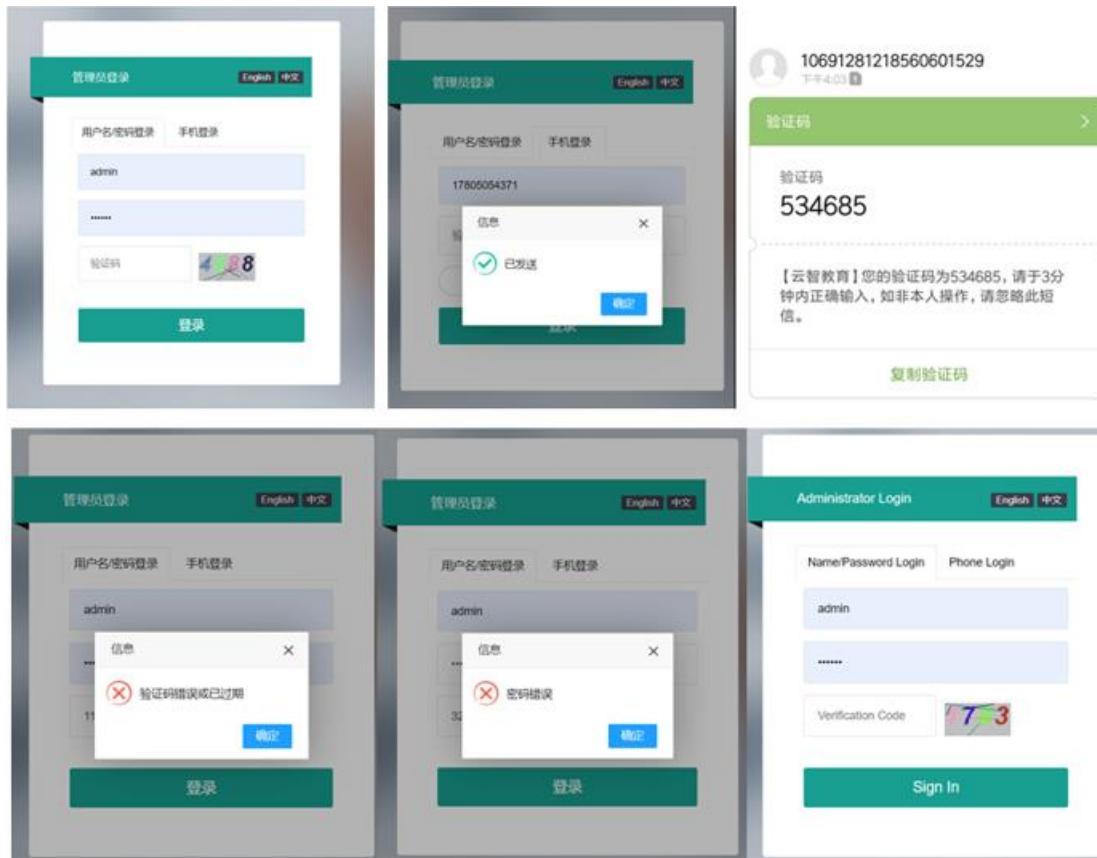


图 5.1 登陆页展示图

5.1.3 登陆模块的流程图

本模块的流程：当用户进入登陆页面，选择自己的登陆方式：用户名/密码或者手机

验证码登陆，输入相应的信息后后端会进行相应的判断，若通过判断则将用户信息和相对应的角色信息存入到 Shiro 框架下的 Session 中，默认有效期为 30 分钟，过期会自动提示重新登录。该模块的流程图如下图所示：

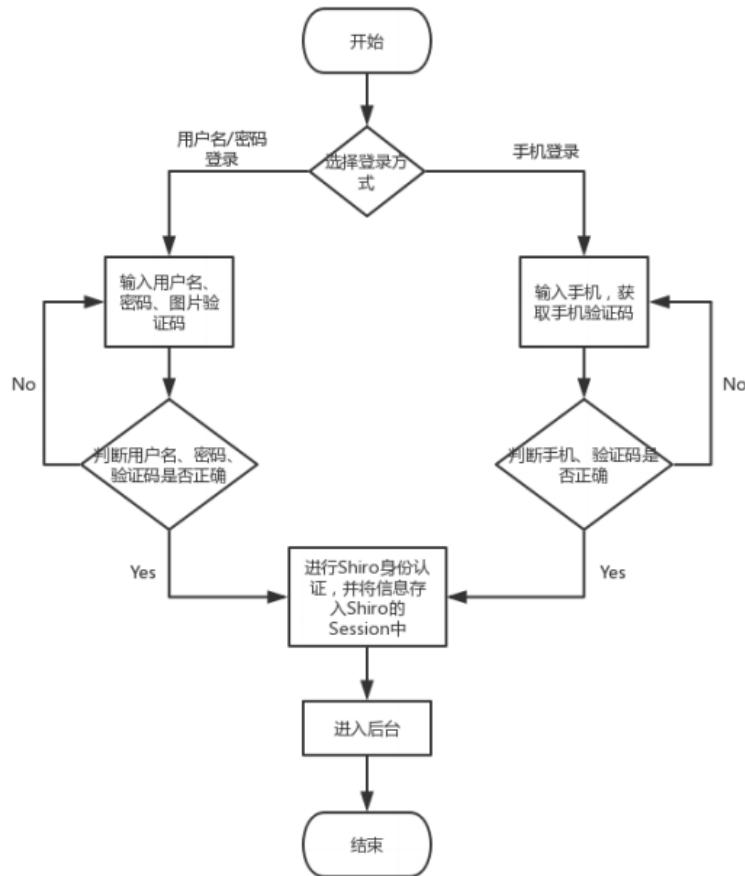


图 5.2 登陆流程图

5.1.4 登陆模块的代码实现

登录功能的主要代码以用户名/密码登陆为例，后端接受到前端页面发送过来的用户名、密码、验证码的值，并通过相关的业务逻辑处理，返回一个 json 对象给前端进行页面的跳转操作。核心代码如下所示。

```

1.   @ResponseBody
2.   @ApiOperation("用户名/密码登录")
3.   @PostMapping(value = "/admin/loginByPassword", produces = "application/json;charset=UTF-8")
4.   public ResBody loginByPassword(@RequestBody Map<String, Object> params,
5.                                   HttpServletRequest request) throws IOException {
6.       String name = params.get("name").toString();
7.       String password = params.get("password").toString();
8.       String check = params.get("check").toString();
9.       ResBody resBody = new ResBody();
  
```

```

10.         logger.info(name + "----" + password + "----" + check);
11.         Log log = new Log();
12.         String checkRedis = stringRedisTemplate.opsForValue().get(check);
13.         //MD5 加密
14.         String pass = DigestUtils.md5DigestAsHex(password.getBytes());
15.         if (checkRedis == null) {
16.             logger.info("验证码错误或已过期");
17.             resBody.setCode(500);
18.             resBody.setMsg("验证码错误或已过期");
19.         } else {
20.             if (checkRedis.equalsIgnoreCase(check)) {
21.                 //1、获取 Subject
22.                 Subject subject = SecurityUtils.getSubject();
23.                 //2、封装用户数据
24.                 UsernamePasswordToken token = new UsernamePasswordToken(name, pass);
25.                 //3、执行登录方法
26.                 try {
27.                     subject.login(token);
28.                     logger.info(subject.getPrincipal().toString());
29.                     Role role = roleService.findRoleByAdmin((Admin) subject.getPrincipal());
30.                     //登录成功
31.                     subject.getSession().setAttribute("userSig", UserSigUtil.getUserSig(((Admin) subject.getPrincipal()).getName()));
32.                     subject.getSession().setAttribute("Admin", subject.getPrincipal());
33.                     subject.getSession().setAttribute("Role", role);
34.                     logger.info("登录成功");
35.                     resBody.setCode(200);
36.                     resBody.setMsg("登录成功");
37.                     String ip = IpUtil.getIPAddress(request);
38.                     String s = name+";"+ip+"用户名/密码登录";
39.                     rabbitTemplate.convertAndSend("exchange.direct", "admin.log", s);
40.                 } catch (UnknownAccountException e) {
41.                     logger.info("用户不存在");
42.                     resBody.setCode(500);
43.                     resBody.setMsg("用户不存在");
44.                 } catch (IncorrectCredentialsException e) {
45.                     logger.info("密码错误");
46.                     resBody.setCode(500);
47.                     resBody.setMsg("密码错误");
48.                 } catch (Exception e) {
49.                     logger.info("IP 地址解析错误");
50.                     resBody.setCode(500);
51.                     resBody.setMsg("IP 地址解析错误");
52.                 }

```

```

53.         } else {
54.             logger.info("验证码错误");
55.             resBody.setCode(500);
56.             resBody.setMsg("验证码错误");
57.         }
58.     }
59.     return resBody;
60. }

```

5.2 活动模块的实现

活动模块是系统的核心模块，当有新的会议审核时系统会实时通知，管理员点入活动模块点击待审核的活动会查看到相应的审核信息，如果管理员同意审核，会跳出一个分配房间的窗口进行房间的分配，否则会跳出审核不通过原因的窗口供管理员填写并发送。除了活动的审核，还有活动的查询和删除功能，本系统使用了 layui 的动态数据表格，可以进行表格属性名的选择查看、导出为 EXCEL 或者 CSV 格式的数据文件。可以进行数据的打印，表格的分页也十分的人性化，可以选择每页显示的条数，跳转到第多少页等等。在活动删除方面，可以选择单个的删除，也可以进行活动的批量删除。

5.2.1 WebSocket 代码实现

实现系统的 WebSocket 需要前端和后端共同完成，后端需要搭建一个 WebSocket 服务，前端使用 js 中相关的 api 访问，为了避免掉线，每秒客户端都会对后台发送数据，当接收到响应数据就会开始下一次的心跳访问。本系统实现了心跳机制确保 WebSocket 的连接。前端页面的核心代码如下：

```

1. var lockReconnect = false;//避免重复连接
2. var wsUrl = "wss://ntdxyg.mynatapp.cc/webSocket/chat/index";
3. var ws;
4. var tt;
5. function createWebSocket() {
6.     try {
7.         ws = new WebSocket(wsUrl);
8.         init();
9.     } catch(e) {
10.         console.log('catch');
11.         reconnect(wsUrl);
12.     }
13. }
14. function init() {
15.     ws.onclose = function () {
16.         console.log('链接关闭');

```

```

17.     reconnect(wsUrl);
18.     layer.open({
19.         type: 1
20.         ,title: false //不显示标题栏
21.         ,closeBtn: false
22.         ,area: '300px;'
23.         ,shade: 0.8
24.         ,id: 'LAY_layuipro' //设定一个 id, 防止重复弹出
25.         ,btn: ['重新登陆']
26.         ,btnAlign: 'c'
27.         ,moveType: 1 //拖拽模式, 0 或者 1
28.         ,content: '<div style="padding: 50px; line-height: 22px; background-color:
#393D49; color: #fff; font-weight: 300;">您的登陆信息已过期! <br>请重新登录<br></div>'
29.         ,yes: function(index,layero){
30.             window.location.href = "/login";
31.         }
32.     });
33. };
34. ws.onerror = function() {
35.     console.log('发生异常了');
36.     reconnect(wsUrl);
37.     layer.open({
38.         type: 1
39.         ,title: false //不显示标题栏
40.         ,closeBtn: false
41.         ,area: '300px;'
42.         ,shade: 0.8
43.         ,id: 'LAY_layuipro' //设定一个 id, 防止重复弹出
44.         ,btn: ['重新登陆']
45.         ,btnAlign: 'c'
46.         ,moveType: 1 //拖拽模式, 0 或者 1
47.         ,content: '<div style="padding: 50px; line-height: 22px; background-color:
#393D49; color: #fff; font-weight: 300;">您的登陆信息已过期! <br>请重新登录<br></div>'
48.         ,yes: function(index,layero){
49.             window.location.href = "/login";
50.         }
51.     });
52. };
53. ws.onopen = function () {
54.     //心跳检测重置
55.     heartCheck.start();
56. };
57. ws.onmessage = function (event) {
58.     //拿到任何消息都说明当前连接是正常的
59.     //BUG, 需要修改
60.     console.log('接收到消息');

```

```

61.     heartCheck.start();
62.     var text = event.data;
63.     if (text != "123456789"){
64.         if (text.search("您有新的用户")!= -1){
65.             $('#mess').show();
66.             toastr.options.onclick = function () {
67.                 document.getElementById("user").click();
68.                 $('#mess').hide();
69.             }
70.             toastr.info(text,'通知');
71.         }else if (text.search("您有新的活动")!= -1){
72.             $('#mess').show();
73.             toastr.options.onclick = function () {
74.                 document.getElementById("activity").click();
75.                 $('#mess').hide();
76.             }
77.             toastr.success(text,'活动');
78.         }
79.     }
80. }
81. }
82. function reconnect(url) {
83.     if(lockReconnect) {
84.         return;
85.     };
86.     lockReconnect = true;
87.     //没连接上会一直重连，设置延迟避免请求过多
88.     tt && clearTimeout(tt);
89.     tt = setTimeout(function () {
90.         createWebSocket(url);
91.         lockReconnect = false;
92.     }, 4000);
93. }
94. //心跳检测
95. var heartCheck = {
96.     timeout: 3000,
97.     timeoutObj: null,
98.     serverTimeoutObj: null,
99.     start: function(){
100.         console.log('start');
101.         var self = this;
102.         this.timeoutObj && clearTimeout(this.timeoutObj);
103.         this.serverTimeoutObj && clearTimeout(this.serverTimeoutObj);
104.         this.timeoutObj = setTimeout(function(){
105.             console.log('发送心跳');
106.             ws.send("123456789");

```

```

107.         self.serverTimeoutObj = setTimeout(function() {
108.             console.log(111);
109.             console.log(ws);
110.             ws.close();
111.                 // createWebSocket();
112.             }, self.timeout);
113.         }, this.timeout)
114.     }
115. }
116. createWebSocket(wsUrl);

```

后端需要使用注解将 WebSocket 配置到 Spring Boot 中，再进行组件的实现，以房间号进行信息的广播。其核心代码如下：

```

1. @Configuration
2. public class WebSocketConfig {
3.     @Bean
4.     public ServerEndpointExporter serverEndpointExporter(){
5.         return new ServerEndpointExporter();
6.     }
7. }
8.
9. @ServerEndpoint("/WebSocket/chat/{roomid}")
10. @Component
11. public class MyWebSocket {
12.     // 使用 map 来收集 session, key 为 roomName, value 为同一个房间的用户集合
13.     // concurrentMap 的 key 不存在时报错, 不是返回 null
14.     private static final Map<String, Set<Session>> rooms = new ConcurrentHashMap();
15.     private Map<String, Integer> people = new HashMap<>();
16.     private final static Logger logger = LoggerFactory.getLogger(MyWebSocket.class);
17.
18.     @OnOpen
19.     public void connect(@PathParam("roomid") String roomid, Session session) throws Exception {
20.         // 将 session 按照房间名来存储, 将各个房间的用户隔离
21.         if (!rooms.containsKey(roomid)) {
22.             Set<Session> room = new HashSet<>();
23.             room.add(session);
24.             rooms.put(roomid, room);
25.         } else {
26.             // 房间已存在, 直接添加用户到相应的房间
27.             rooms.get(roomid).add(session);
28.             people.put("people", rooms.get(roomid).size());
29.             broadcast(roomid, JSON.toJSONString(people));
30.         }

```

```

31.         logger.info("a client has connected!" + roomid + ":" + rooms.get(roomid).size() + "人");
32.     }
33.
34.     @OnClose
35.     public void disConnect(@PathParam("roomid") String roomid, Session session) throws
Exception {
36.         rooms.get(roomid).remove(session);
37.         people.put("people", rooms.get(roomid).size());
38.         broadcast(roomid, JSON.toJSONString(people));
39.         logger.info("a client has disconnected!" + roomid + ":" + rooms.get(roomid).size() + "人");
40.     }
41.
42.     @OnMessage
43.     public void receiveMsg(@PathParam("roomid") String roomid,
44.                             String msg) throws Exception {
45.         broadcast(roomid, msg);
46.     }
47.
48.     // 按照房间名进行广播
49.     public synchronized static void broadcast(String roomid, String msg) throws Except
ion {
50.         for (Session session : rooms.get(roomid)) {
51.             //对广播的信息进行关键字屏蔽
52.             session.getAsyncRemote().sendText(TextUtils.filter(msg));
53.         }
54.     }
55. }
```

5.2.2 活动审核界面展示

活动审核界面，管理员点击审核进行相关信息的审核后，会跳出提示框：同意还是拒绝。若同意则分配房间，否则写出不同意的原因，最后以邮件形式通知用户。

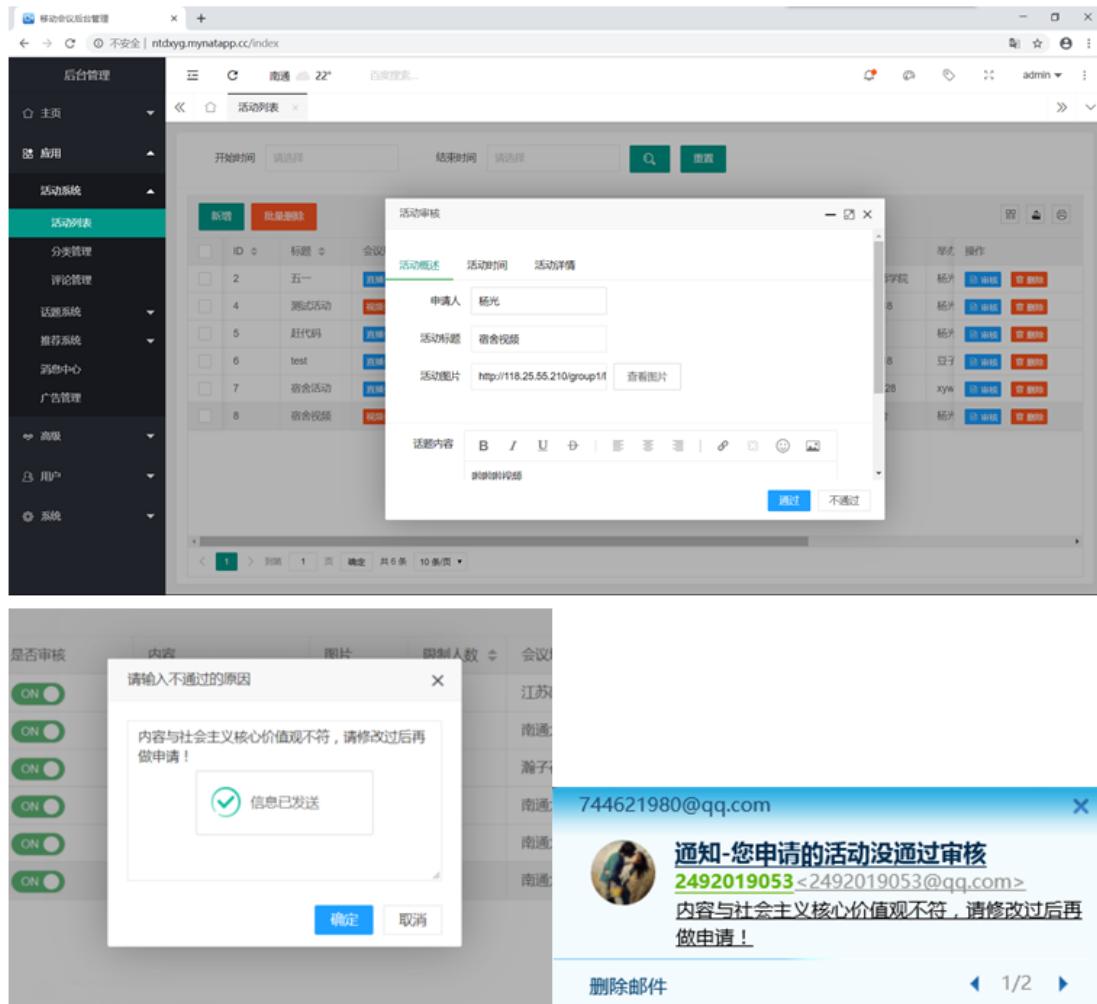


图 5.3 活动审核界面

5.2.3 活动审核的代码实现

活动审核的主要代码如下所示。

```

1.     @ApiOperation("活动不通过审核")
2.     @ResponseBody
3.     @GetMapping(value = "/activity/sendActivityMessage", produces = "application/json;
 charset=UTF-8")
4.     public ResBody sendMessage(String activityId, String value) {
5.         ResBody resBody = new ResBody();
6.         User user = activityService.findUserByActivityId(Integer.parseInt(activityId))
7. ;
8.         if (user!=null) {
9.             resBody.setCode(200);
10.            rabbitTemplate.convertAndSend("exchange.direct", "user", "NoActivity:" + user
11. getEmail() + ":" + value);
12.        } else {
13.            resBody.setCode(500);
14.        }
15.        return resBody;

```

```

14.     }
15.
16.     @ApiOperation("活动通过审核")
17.     @ResponseBody
18.     @GetMapping(value = "/activity/updateActivityAndRoom", produces = "application/json; charset=UTF-8")
19.     public ResBody updateActivityAndRoom(String activityId, String roomId) {
20.         ResBody resBody = new ResBody();
21.         User user = activityService.findUserByActivityId(Integer.parseInt(activityId));
22.         Room room = activityService.findRoomById(Integer.parseInt(roomId));
23.         Activity activity = activityService.findActivityById(Integer.parseInt(activityId));
24.         if (room != null){
25.             //房间存在
26.             resBody.setCode(500);
27.         }else{
28.             //房间不存在，新建房间并关联
29.             activityService.updateActivityAndRoom(activity, Integer.parseInt(roomId));
30.             resBody.setCode(200);
31.             rabbitTemplate.convertAndSend("exchange.direct", "user", "successActivity"+user.getEmail()+":"+roomId+":"+activity.getTitle());
32.         }
33.         return resBody;
34.     }

```

5.3 增删改查功能的代码实现

在本节中，以对广告管理的操作为例，详细介绍移动会议实时互动运营方管理端对数据进行增添、删除、修改、查询相应功能的代码实现。

5.3.1 相关实体类的设计

上文的数据库设计中，分析过广告在数据库中拥有 id，图片地址，url 地址，描述，状态属性将这些属性一一映射到类中，即将这些属性抽象为具体的变量，访问属性为 private，再用插件 Lombok 进行代码简化。

5.3.2 Lombok 简介与使用

Lombok 能以简单的注解形式来简化 java 代码，提高开发人员的开发效率。例如开发中经常需要写的 javabean，都需要花时间去添加相应的 getter，setter，也许还要去写构造器、equals 等方法，而且需要维护，当属性多时会出现大量的 getter，setter 方法，

这些显得很冗长也没有太多技术含量，一旦修改属性，就容易出现忘记修改对应方法的失误。Lombok 能通过注解的方式，在编译时自动为属性生成构造器、getter/setter、equals、hashcode、toString 方法。这样就省去了手动重建这些代码的麻烦，降低了开发的时间成本并且使代码看起来更简洁些。

使用的代码如下所示。

```

1. @Data
2. @AllArgsConstructor
3. @NoArgsConstructor
4. public class Banner {
5.     private int id;
6.     private String src;
7.     private String url;
8.     private String describe;
9.     private int status;
10. }
```

5.3.3 Dao 层的接口实现

本系统在 dao 层选择的是 JdbcTemplate 而不是其他 hibernate、mybaits、JPA 等 ORM 类框架，虽然开发起来速度不如框架的使用速度快，但是全程使用原生的 SQL 代码也是一个不错的选择。代码如下

```

1. @Repository
2. public class BannerDaoImpl implements BannerDao {
3.     @Autowired
4.     JdbcTemplate jdbcTemplate;
5.     @Override
6.     public int getCount() {
7.         int count = jdbcTemplate.queryForObject("select count(*) from banner", Integer
     .class);
8.         return count;
9.     }
10.    @Override
11.    public List<Banner> findAllBanner(int page, int limit) {
12.        List<Banner> logs = jdbcTemplate.query("select * from banner limit ?,?", new O
     bject[]{(page-1)*limit,limit}, new BeanPropertyRowMapper(Banner.class));
13.        if (logs!=null){
14.            return logs;
15.        }else{
16.            return null;
17.        }
18.    }
```

```

19.     @Override
20.     public List<Banner> findAllBanner(int status, int page, int limit) {
21.         List<Banner> logs = jdbcTemplate.query("select * from banner where status = ?  

22.             limit ?,?", new Object[]{status,(page-1)*limit,limit}, new BeanPropertyRowMapper(Banner.class));
23.         if (logs!=null){
24.             return logs;
25.         }else{
26.             return null;
27.         }
28.     }
29.     @Override
30.     public int getCount(int status) {
31.         int count = jdbcTemplate.queryForObject("select count(*) from banner where sta  

32.             tus = ?",new Object[]{status}, Integer.class);
33.         return count;
34.     }
35.     @Override
36.     public void deleteBannerById(int id) {
37.         jdbcTemplate.update("DELETE from banner where id=?",id);
38.     }
39.     @Override
40.     public Banner findBannerById(int id) {
41.         List<Banner> list = jdbcTemplate.query("select * from banner where id" +  

42.             " = ?", new Object[]{id}, new BeanPropertyRowMapper(Banner.class));
43.         if (list!=null && list.size()>0){
44.             return list.get(0);
45.         }else{
46.             return null;
47.         }
48.     }
49.     @Override
50.     public int addBannerByAjax(Banner banner) {
51.         return jdbcTemplate.update("insert into banner values(null, ?, ?, ?, ?, ?)",  

52.             banner.getSrc(),banner.getUrl(),banner.getDescribe(),banner.getStatus()  

53.         );
54.     }
55.     @Override
56.     public int updateBanner(Banner banner) {
57.         return jdbcTemplate.update("update banner set src = ?,url = ?,`describe` = ?,s  

58.             tatus = ? where id = ?",
59.             banner.getSrc(),banner.getUrl(),banner.getDescribe(),banner.getStatus()  

60.             ,banner.getId());
61.     }
62. }

```

5.4 权限控制的代码实现

本系统中关于权限实现，使用 Apache Shiro 框架进行权限系统的快速搭建。不同的角色进入 WEB 管理页面会使用相对应的权限菜单，实现了菜单级别的权限控制。

5.4.1 核心代码的设计

首先需要将菜单的 URL 地址加入到 Shiro 的过滤器中，每次登陆的时候再从数据库中判断角色权限。

```

1. 1. @Configuration
2. 2. public class ShiroConfig {
3. 3.     /*
4. 4.     * ShiroFilterFactoryBean
5. 5.     */
6. 6.     @Bean
7. 7.     public ShiroFilterFactoryBean getShiroFilterFactoryBean(@Qualifier("securityManager")
8. 8.         DefaultWebSecurityManager securityManager) {
9. 9.         ShiroFilterFactoryBean shiroFilterFactoryBean = new ShiroFilterFactoryBean();
10. 10.        //设置安全管理器
11. 11.        shiroFilterFactoryBean.setSecurityManager(securityManager);
12. 12.        /**
13. 13.         * anon: 无需认证（登录）可以访问
14. 14.         * authc: 必须认证（登录）才可以访问
15. 15.         * user: 使用 rememberme 可以直接访问
16. 16.         * perms: 必须得到资源权限才可以访问
17. 17.         * role: 必须得到角色权限才可以访问
18. 18.         */
19. 19.        shiroFilterFactoryBean.setLoginUrl("/login");
20. 20.        Map<String, String> filterMap = new LinkedHashMap<String, String>();
21. 21.        //登录页相关 URI 无需认证
22. 22.        filterMap.put("/logout", "logout");
23. 23.        filterMap.put("/login", "anon");
24. 24.        filterMap.put("/getCode", "anon");
25. 25.        filterMap.put("/SendSMS", "anon");
26. 26.        filterMap.put("/admin/loginByPassword", "anon");
27. 27.        filterMap.put("/admin/loginByPhone", "anon");
28.
29.        //授权过滤器
30.        filterMap.put("/logMonitor", "perms[admin:logMonitor]");
31.        //filterMap.put("/", "authc");
32.        //菜单权限
33.        filterMap.put("/", "authc");

```

```

34.         filterMap.put("/index", "authc");
35.         filterMap.put("/console", "authc");
36.         filterMap.put("/swagger", "authc");
37.         filterMap.put("/meetingList", "authc");
38.         filterMap.put("/tag", "authc");
39.         filterMap.put("/comment", "authc");
40.         filterMap.put("/topicList", "authc");
41.         filterMap.put("/replayList", "authc");
42.         filterMap.put("/popRecommend", "authc");
43.         filterMap.put("/priRecommend", "authc");
44.         filterMap.put("/message", "authc");
45.         filterMap.put("/banner", "authc");
46.         filterMap.put("/rtmp", "authc");
47.         filterMap.put("/webrtc", "authc");
48.         filterMap.put("/user/list", "authc");
49.         filterMap.put("/admin/list", "authc");
50.         filterMap.put("/role/list", "authc");
51.         filterMap.put("/springbootadmin", "authc");
52.         filterMap.put("/rabbitmqMonitor", "authc");
53.         filterMap.put("/druidMonitor", "authc");
54.         filterMap.put("/logMonitor", "authc");
55.         filterMap.put("/webSetting", "authc");
56.         filterMap.put("/fileManagement", "authc");
57.         filterMap.put("/sensitiveWord", "authc");
58.         filterMap.put("/basicData", "authc");
59.         filterMap.put("/updateAdminPass", "authc");
60.         shiroFilterFactoryBean.setLoginUrl("/login");
61.         shiroFilterFactoryBean.setUnauthorizedUrl("/401");
62.         shiroFilterFactoryBean.setFilterChainDefinitionMap(filterMap);
63.         return shiroFilterFactoryBean;
64.     }
65.
66.     /**
67.      * DefaultWebSecurityManager
68.     */
69.
70.     @Bean(name = "securityManager")
71.     public DefaultWebSecurityManager getDefaultWebSecurityManager(@Qualifier("adminRea
    lm") AdminRealm adminRealm) {
72.         DefaultWebSecurityManager securityManager = new DefaultWebSecurityManager();
73.         //关联 realm
74.         securityManager.setRealm(adminRealm);
75.         return securityManager;
76.     }
77.
78.     /**

```

```

79.     * Realm
80.     */
81.
82.     @Bean(name = "adminRealm")
83.     public AdminRealm getRealm() {
84.         return new AdminRealm();
85.     }
86.
87.     @Bean
88.     public ShiroDialect getShiroDialect(){
89.         return new ShiroDialect();
90.     }
91. }
```

因为配合前端 Lay UI 权限树的插件，需要将结果以特定格式的 json 传到前端。本系统菜单级数最多为三级，选择了暴力方法获取菜单信息，Controller 代码如下。

```

1.     @ApiOperation("获取角色的菜单列表")
2.     @ResponseBody
3.     @GetMapping("/menu/getMenu/{id}")
4.     public ResBody getMenuByAdminId(@PathVariable("id") @ApiParam("角色
   id") Integer id){
5.         ResBody resBody = new ResBody();
6.         List<Menu> menus = menuService.getMenus();
7.         List<Menu> list = new LinkedList<>();
8.         List<Menu> menus_ids = menuService.getMenusId(id);
9.         List<Integer> ids = new LinkedList();
10.        for (Menu m:menus_ids){
11.            ids.add(m.getId());
12.        }
13.        for (int i = 0;i<menus.size();i++){
14.            if (menus.get(i).getParent_id() == 0){
15.                List<Menu> list2 = new LinkedList<>();
16.                for (int j = 0;j<menus.size();j++){
17.                    if (menus.get(j).getParent_id() == menus.get(i).getId()){
18.                        List<Menu> list3 = new LinkedList<>();
19.                        for (int k = 0;k<menus.size();k++){
20.                            if (menus.get(k).getParent_id() == menus.get(j).getId()){
21.                                if (ids.contains(menus.get(k).getId())){
22.                                    menus.get(k).setChecked(true);
23.                                }
24.                                list3.add(menus.get(k));
25.                            }
26.                        }
27.                        menus.get(j).setChildren(list3);
```

```

28.             if (menus.get(j).getChildren().isEmpty() && ids.contains(menus
29.                 .get(j).getId())){
30.                     menus.get(j).setChecked(true);
31.                 }
32.                 list2.add(menus.get(j));
33.             }
34.             menus.get(i).setChildren(list2);
35.             list.add(menus.get(i));
36.         }
37.     }
38.     resBody.setData(list);
39.     resBody.setCode(0);
40.     return resBody;
41. }

```

5.4.2 界面的展示及效果

管理员选择角色管理，对系统中已有的角色进行角色的授权功能，点击授权按钮，在出现的弹窗页面中选择相对应的菜单，点击保存后即可完成角色的权限管理。页面的展示如下所示。

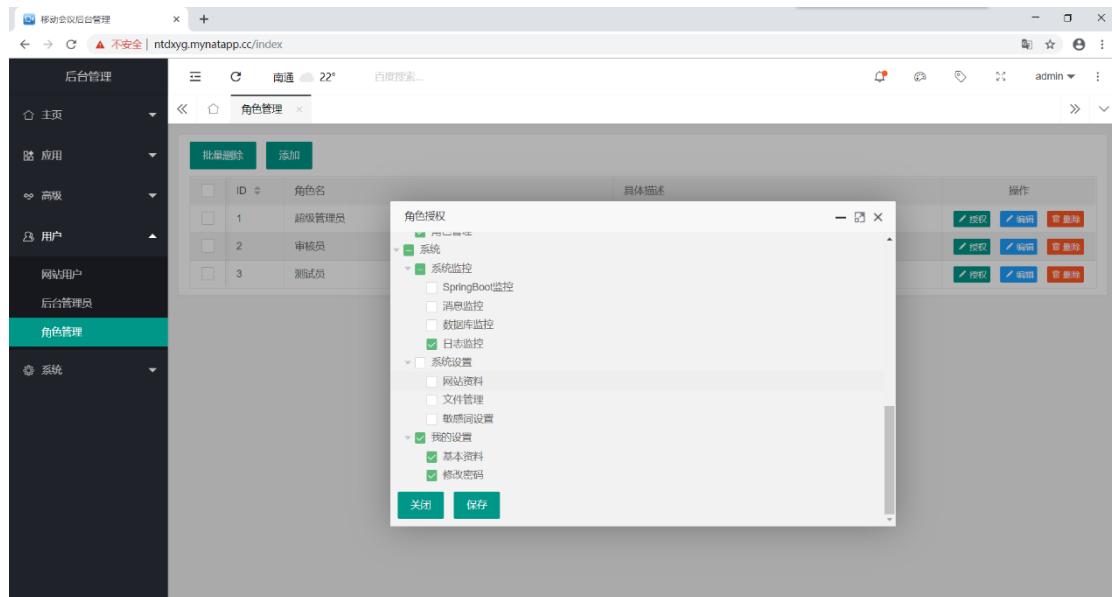


图 5.4 权限模块界面

5.5 关键词屏蔽功能的设计

本系统中涉及到论坛类以及实时弹幕，需要对敏感词做一个过滤，我考虑到的方案有下面四种：1、直接将敏感词组织成 String 后，可以使用自带的 String.indexOf() 方法来查询敏感词。2、传统的敏感词录入数据库后进行相关的 SQL 查询。3、利用 Lucene 建

立分词索引来查询。4、利用 DFA 算法来进行。首先，如果系统中设置的敏感词有几千条，使用 String 中的 `indexOf` 肯定不行。其次，为了方便以后的扩展性尽量减少对数据库的依赖，所以放弃入库后的 SQL 查询。然后 Lucene 本身作为本地索引，敏感词增加后需要触发更新索引，并且这里本着轻量原则不想引入更多的库，所以放弃。于是本系统将 DFA 算法选为敏感词屏蔽功能的首选算法。

5.5.1 DFA 算法简介与分析

DFA 全称为：Deterministic Finite Automaton, 即确定有穷自动机。其特征为：有一个有限状态集合和一些从一个状态通向另一个状态的边，每条边上标记有一个符号，其中一个状态是初态，某些状态是终态。但不同于不确定的有限自动机，DFA 中不会有从同一状态出发的两条边标志有相同的符号。简单点说就是，它是通过 `event` 和当前的 `state` 得到下一个 `state`，即 `event+state=nextstate`。理解为系统中有多个节点，通过传递进入的 `event`，来确定走哪个路由至另一个节点，而节点是有限的。

以王八蛋和王八羔子两个敏感词来进行描述，首先构建敏感词库，该词库名称为 `SensitiveMap`，这两个词的二叉树构和用 hash 表构造分别为：

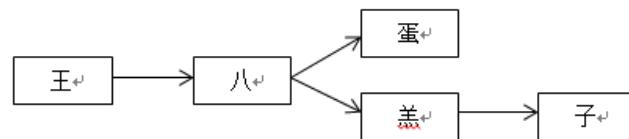


图 5.5 二叉树构造图

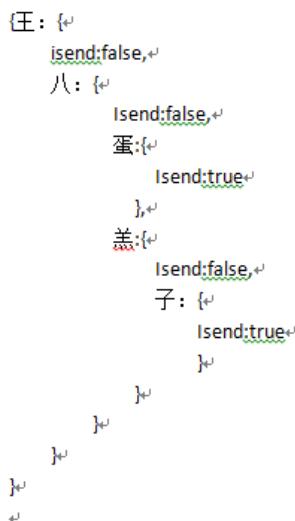


图 5.6 hash 表构造图

基于敏感词库搜索算法的描述以上面例子构造出来的 SensitiveMap 为敏感词库进行示意，假设这里输入的关键字为：王八不好，流程图如下：

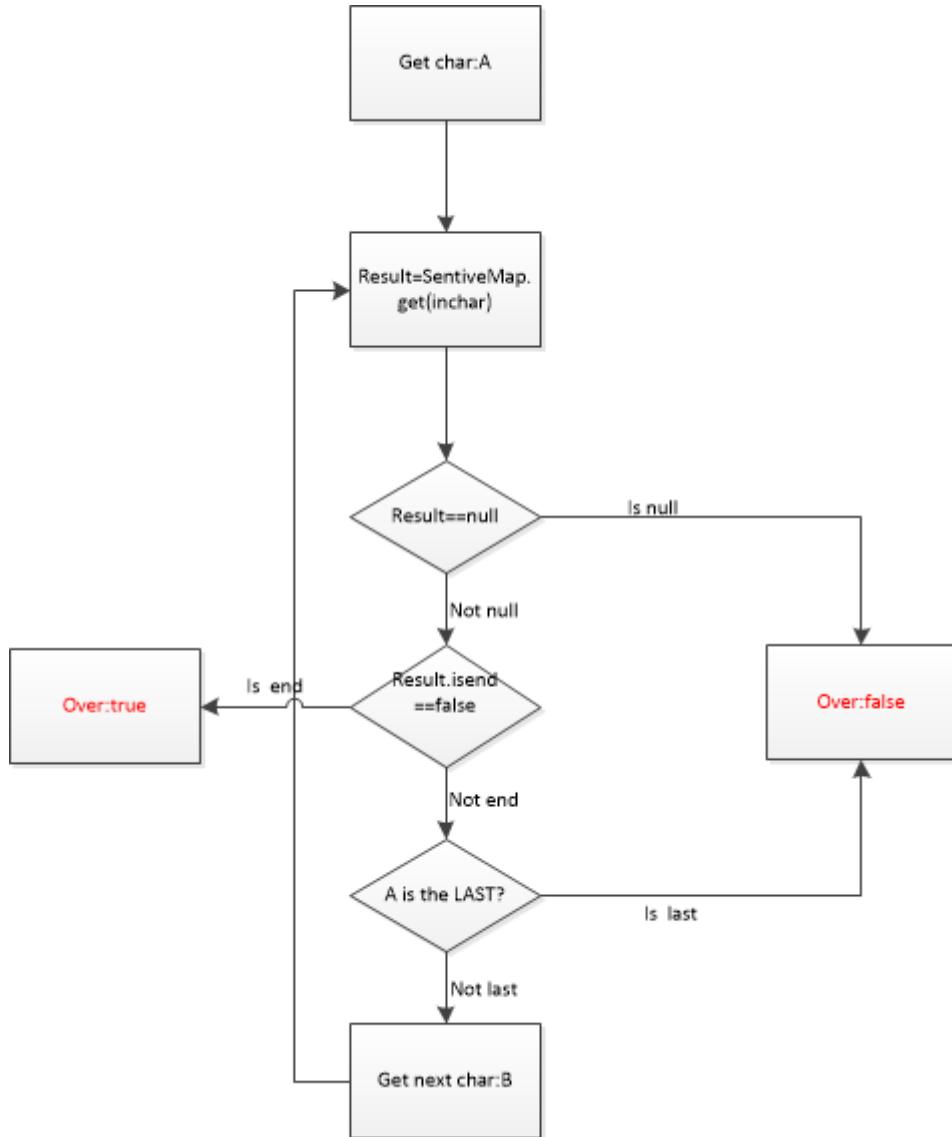


图 5.7 DFA 算法流程图

5.5.2 核心代码的实现

```

1. public static final String doFilter(String src) {
2.     init();
3.     char[] chs = src.toCharArray();
4.     int length = chs.length;
5.     for(int i = 0; i < length; ++i) {
6.         int currC = charConvert(chs[i]);
7.         if (set.containsKey(currC)) {
8.             WordNode node = (WordNode)nodes.get(currC);
9.             if (node != null) {
10.                 boolean couldMark = false;
11.                 int markNum = -1;
  
```

```

12.             if (node.isLast()) {
13.                 couldMark = true;
14.                 markNum = 0;
15.             }
16.             int k = i;
17.             while(true) {
18.                 ++k;
19.                 if (k >= length) {
20.                     break;
21.                 }
22.                 int temp = charConvert(chs[k]);
23.                 if (!stopwdSet.contains(temp)) {
24.                     node = node.querySub(temp);
25.                     if (node == null) {
26.                         break;
27.                     }
28.                     if (node.isLast()) {
29.                         couldMark = true;
30.                         markNum = k - i;
31.                     }
32.                 }
33.             }
34.             if (couldMark) {
35.                 for(k = 0; k <= markNum; ++k) {
36.                     chs[k + i] = SIGN;
37.                 }
38.
39.                 i += markNum;
40.             }
41.         }
42.     }
43. }
44. return new String(chs);
45. }
```

5.6 推荐模块的设计

本系统的个性化推荐系统结合当下最热门的机器学习，使用 Java 库 Mahout 搭建，使用的是协同过滤算法。下文将介绍 Mahout 中的算法思想及本系统的具体代码实现。

5.6.1 协同过滤算法简介

协同过滤算法实现的推荐器主要分为两个类别：“基于用户（user-based）” 和 “基于项目（item-based）”。例如现有一个文本文件，包含相关用户，简单的命名 1 到 5，从 101 到 107 为他们喜欢的 7 本书命名。在现实生活中，这些可能是来自于一个公司的数据

库的用户 ID 和产品 ID; Mahout 并不要求用户和物品必须按照数字命名。将会使用一种简单的用逗号分割数值的格式，把数据写入文件。

1,101,5.0	用户1对物品102的 偏好值为3.0	用户ID、物品ID、偏好值
1,102,3.0		
1,103,2.5		
2,101,2.0		
2,102,2.5		
2,103,5.0		
2,104,2.0		
3,101,2.5		
3,104,4.0		
3,105,4.5		
3,107,5.0		
4,101,5.0		
4,103,3.0		
4,104,4.5		
4,106,4.0		
5,101,4.0		
5,102,3.0		
5,103,2.0		
5,104,4.0		
5,105,3.5		
5,106,4.0		

图 5.8 数据集图

一番研究之后，可以看出：用户 1 和 5 看起来有相似的品味，他们都喜欢 101、102、103 这三本书。同样的问题存在于用户 1 和 4，因为它们看起来都喜欢 101 和 103（虽然没有说用户 4 喜欢 102）。另一方面，用户 1 和 2 的喜好基本上是对立的；用户 1 喜欢 101，而用户 2 不喜欢，用户 1 喜欢 103，而用户 2 刚好相反。用户 1 和用户 3 没有很多重叠，他们仅同时喜欢 101。图 5.9 显示了用户和物品之间正面和负面的关系。

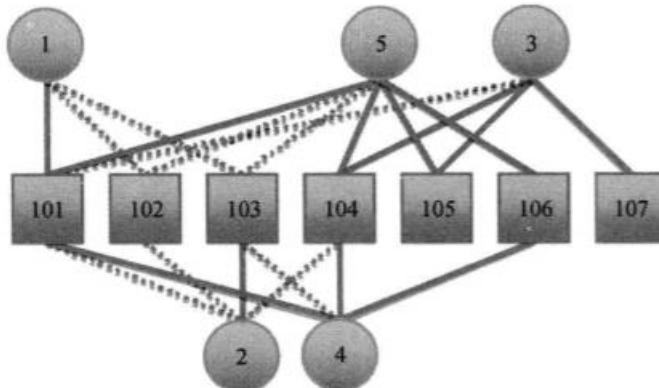


图 5.9 用户商品正负面关系图



本系统的数据集通过 User_Activity 表中获得，用户对活动的权值通过 4 个指

标：是否是申请者、是否完成签到、是否加入收藏列表、是否参与进行相关的数值叠加，最终通过下节的核心代码完成推荐系统的实现。

在基于用户的协同过滤算法中，Mahout 中提供了基本的相似度的计算，它们都通过 UserSimilarity 这个接口，实现用户相似度的计算，包括下面这些常用的：

- PearsonCorrelationSimilarity: 基于皮尔逊相关系数计算相似度
- EuclideanDistanceSimilarity: 基于欧几里德距离计算相似度
- TanimotoCoefficientSimilarity: 基于 Tanimoto 系数计算相似度
- UncenteredCosineSimilarity: 计算 Cosine 相似度

在基于商品的协同过滤算法中也是类似的，它们都通过 ItemSimilarity 这个接口，实现用户相似度的计算，包括下面这些常用的：

- NearestNUserNeighborhood: 对每个用户取固定数量 N 的最近邻居
- ThresholdUserNeighborhood: 对每个用户基于一定的限制，取落在相似度门限内的所有用户为邻居。

5.6.2 本系统推荐系统的核心代码实现

```

1. @ApiOperation("基于用户的协同过滤")
2.     @ResponseBody
3.     @GetMapping(value = "/User_based")
4.     public ResBody User_based() throws Exception {
5.         ResBody resBody = new ResBody();
6.         // step:1 构建模型 2 计算相似度 3 查找 k 紧邻 4 构造推荐引擎
7.         DataModel model =new FileDataModel(new File("C:/ratingdata.txt"));
8.         UserSimilarity similarity =new PearsonCorrelationSimilarity(model);
9.         UserNeighborhood neighborhood =new NearestNUserNeighborhood(2,similarity,model
    );
10.        Recommender recommender= new GenericUserBasedRecommender(model,neighborhood,si
    milarity);
11.        List list = new ArrayList();
12.        for (int i = 1;i<=5;i++){
13.            Map map = new HashMap();
14.            List<RecommendedItem> recommendations =recommender.recommend(i, 1);//为用
    户 1 推荐 1 个 ItemID
15.            for (RecommendedItem item:recommendations){
16.                if (item != null){
17.                    map.put("itemid",item.getItemId());
18.                    map.put("value",item.getValue());
19.                }
20.            }
21.            if (map.get("itemid") == null){
```

```

22.             map.put("itemid","暂无推荐");
23.             map.put("value","0");
24.         }
25.         map.put("userid",i);
26.         list.add(map);
27.     }
28.     resBody.setData(list);
29.     resBody.setCode(0);
30.     resBody.setCount(5);
31.     return resBody;
32. }
33.
34. @ApiOperation("基于物品的协同过滤")
35. @ResponseBody
36. @GetMapping(value = "/Item_based")
37. public ResBody Item_based() throws Exception {
38.
39.     ResBody resBody = new ResBody();
40.     // step:1 构建模型 2 计算相似度 3 查找 k 紧邻 4 构造推荐引擎
41.     DataModel model =new FileDataModel(new File("C:/ratingdata.txt"));
42.     ItemSimilarity similarity =new PearsonCorrelationSimilarity(model);
43.     Recommender recommender= new GenericItemBasedRecommender(model,similarity);
44.     List list = new ArrayList();
45.     for (int i = 1;i<=5;i++){
46.         Map map = new HashMap();
47.         List<RecommendedItem> recommendations =recommender.recommend(i, 1);//为用户1推荐1个ItemID
48.         for (RecommendedItem item:recommendations){
49.             if (item != null){
50.                 map.put("itemid",item.getItemId());
51.                 map.put("value",item.getValue());
52.             }
53.         }
54.         if (map.get("itemid") == null){
55.             map.put("itemid","暂无推荐");
56.             map.put("value","0");
57.         }
58.         map.put("userid",i);
59.         list.add(map);
60.     }
61.     resBody.setData(list);
62.     resBody.setCode(0);
63.     resBody.setCount(5);
64.     return resBody;
65.
66. }

```

5.7 话题模块

话题模块主要是管理员进行一个话题的管理，用户可以对话题进行相关的互动讨论。管理员可以按照需求对话题进行增删改查等操作，同时还可以对用户的话题讨论信息进行管理。由于增删改查的底层基本操作与广告模块是大同小异的，在本节以及之后的章节对增删改查的相关代码实现不再赘述。

话题管理界面如图 5.10 所示。管理员可以依据自己的需要选择相关的属性进行查看、导出报表、添加话题、删除话题。

ID	Content	Poster	Post Time	Operations
1	你们的铲屎官是不是经常突然对手机傻笑？我家铲屎官常常坐沙发上笑的...	yg	2019-02-19 12:16:51	详情 修改 删除
2	大家元宵节快乐！	yg	2019-02-19 12:34:54	详情 修改 删除
6	测试话题	admin	2019-04-09 16:06:29	详情 修改 删除

图 5.10 话题管理界面

上图可看到在动态数据表格的右上角可以选择显示的列属性。还可以选择打印或者导出为 WXCEL/CSV 的数据文件。

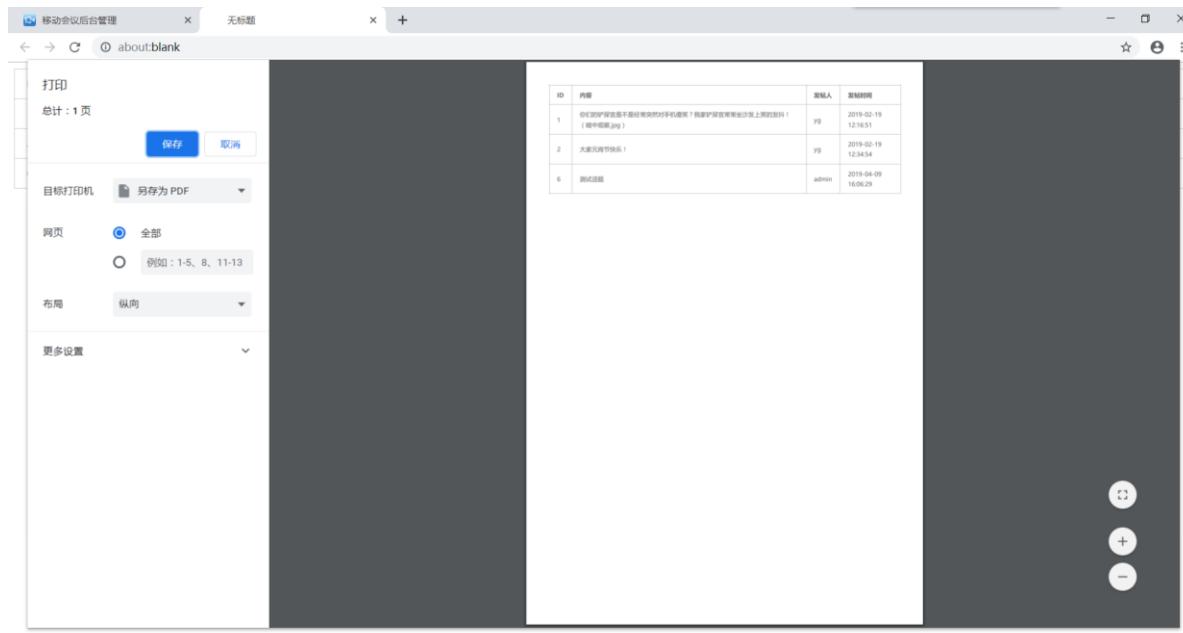


图 5.11 打印界面

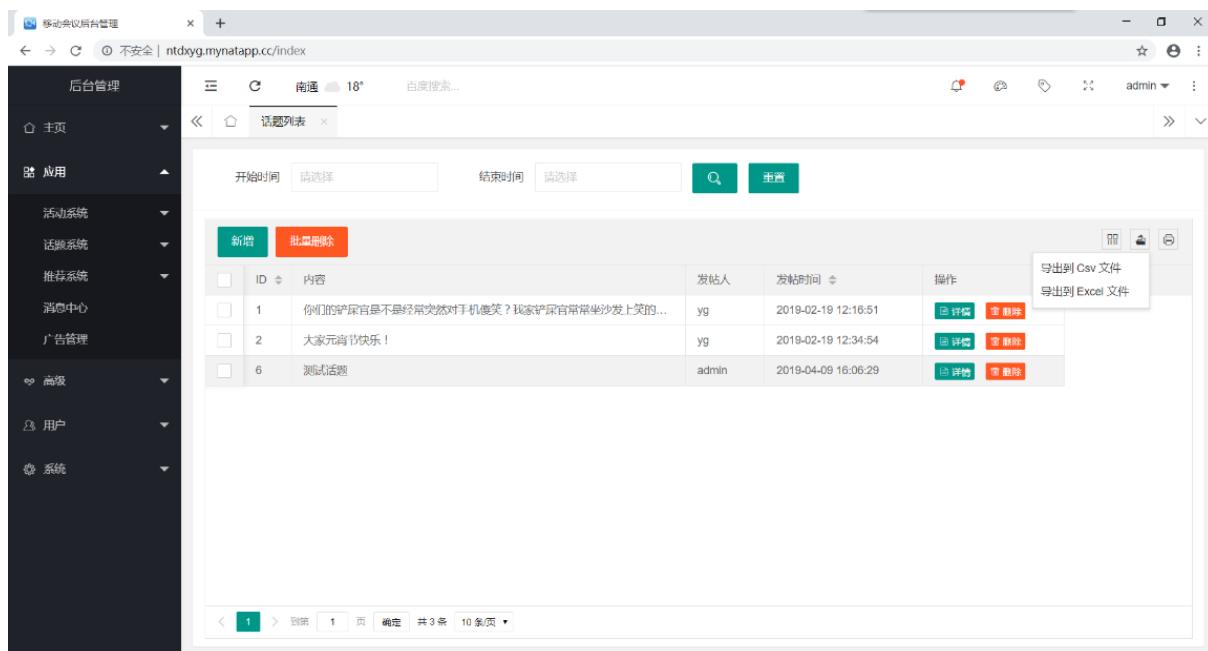


图 5.12 导出 Csv 与 Excel 功能

下图则可看到在话题详情页面上点击查看图片按钮会弹出图片框，在话题图片中可以进行上传图片功能，另外，还配置了 layuiedit 富文本编辑器用于话题内容的详细编辑，可加粗，下划线，插入图片等等功能。

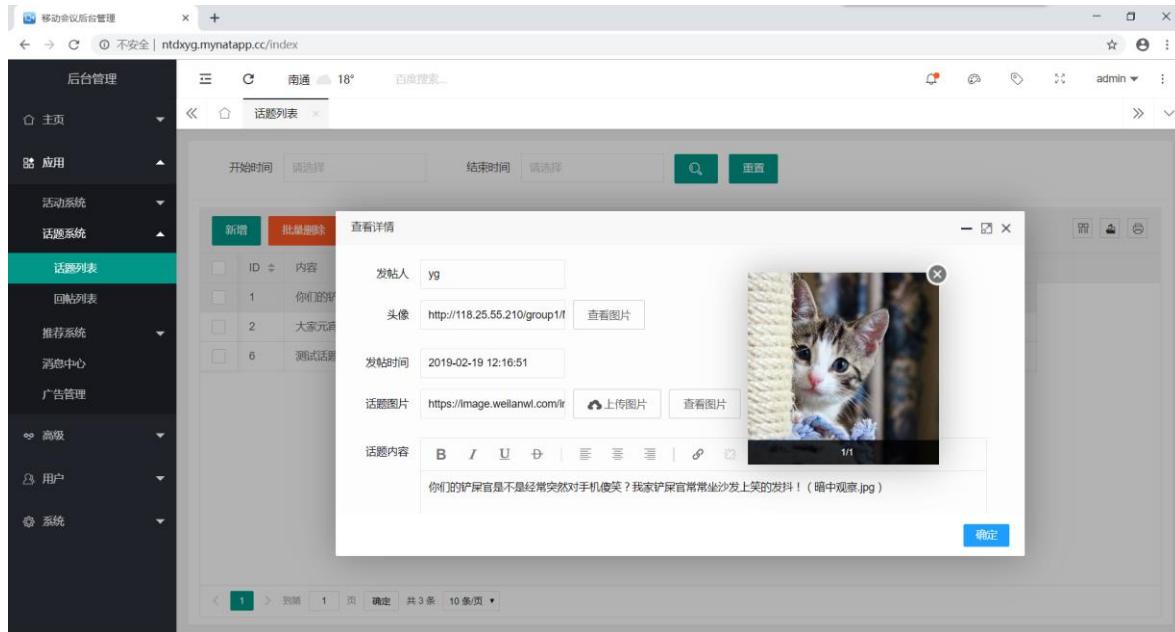


图 5.13 编辑与查看功能

下图则可看到在话题列表页面上可以选择开始时间和结束时间进行话题的查询。点击重置按钮会重回初始化页面。管理员可以选择单个的话题进行删除，也可选择多个话题批量删除，点击删除会跳出提示框，询问管理员是否确定删除，这样可以极大地避免管理员误删话题。

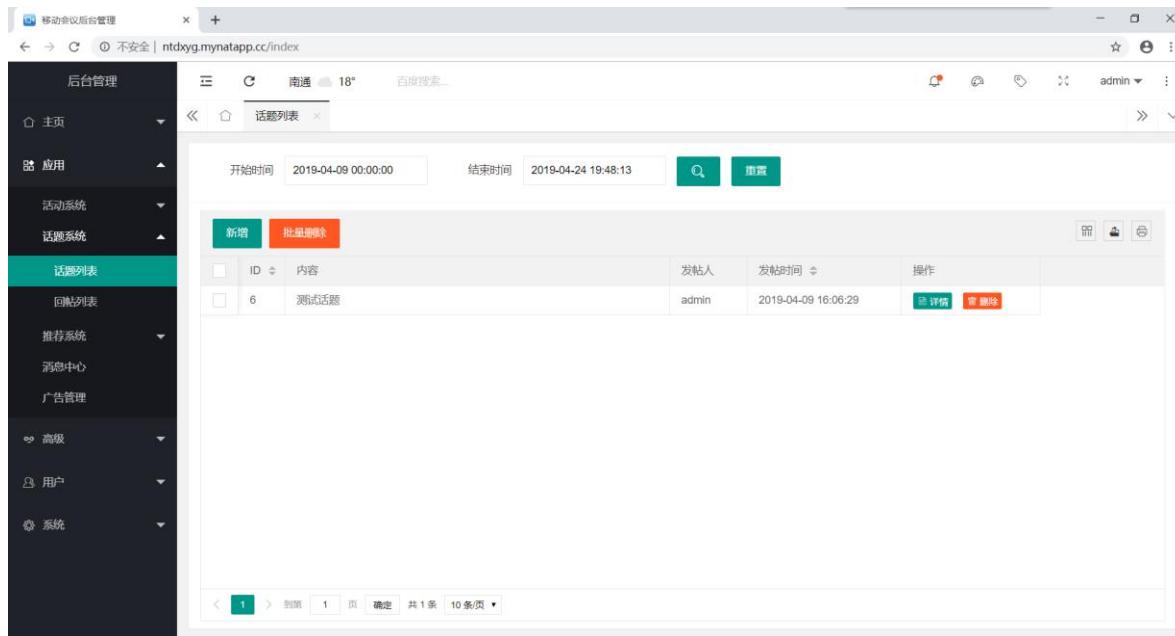


图 5.14 话题查询功能

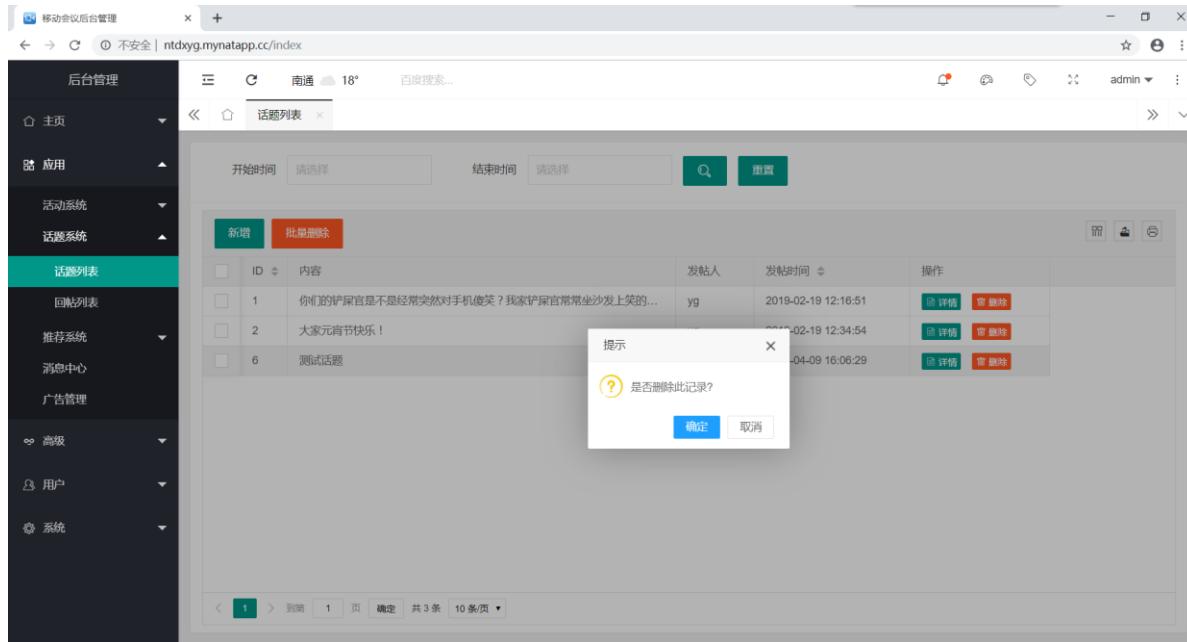


图 5.15 话题删除功能

下图则可看到在话题评论列表页面，同话题列表一样，可以进行评论的查询，查看详情，删除或批量删除，打印，导出等功能。

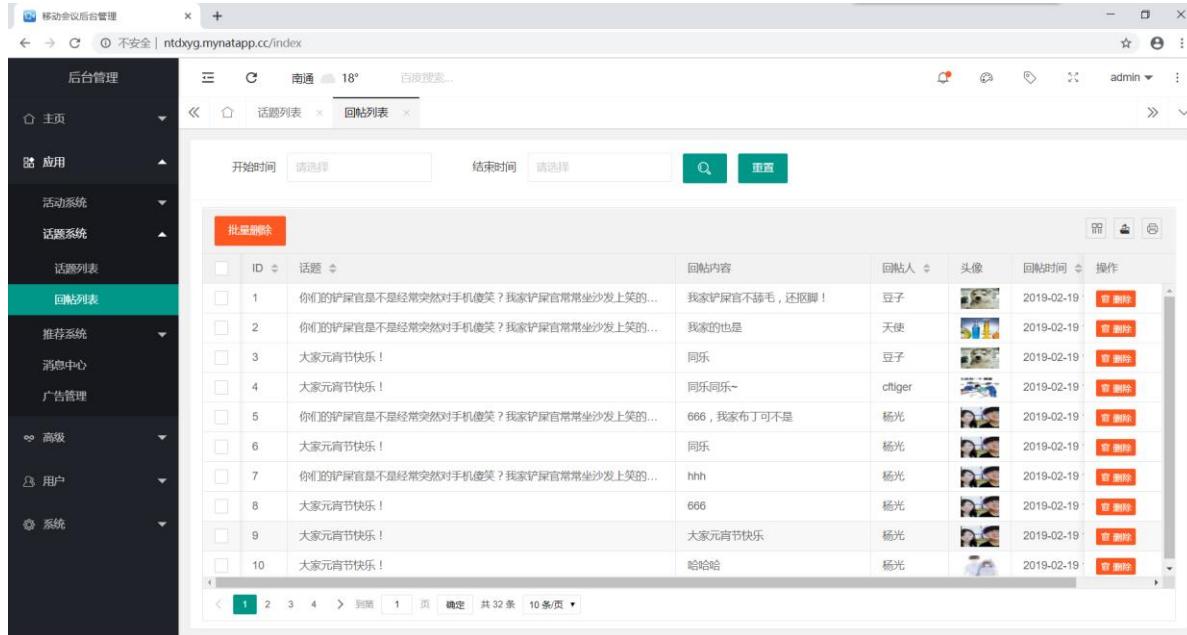


图 5.16 话题评论管理功能

5.8 消息模块

当后台收到用户身份审核或者活动审核的时候，会跳出消息提示功能并存入到消息数据库中，具体示意图如下：

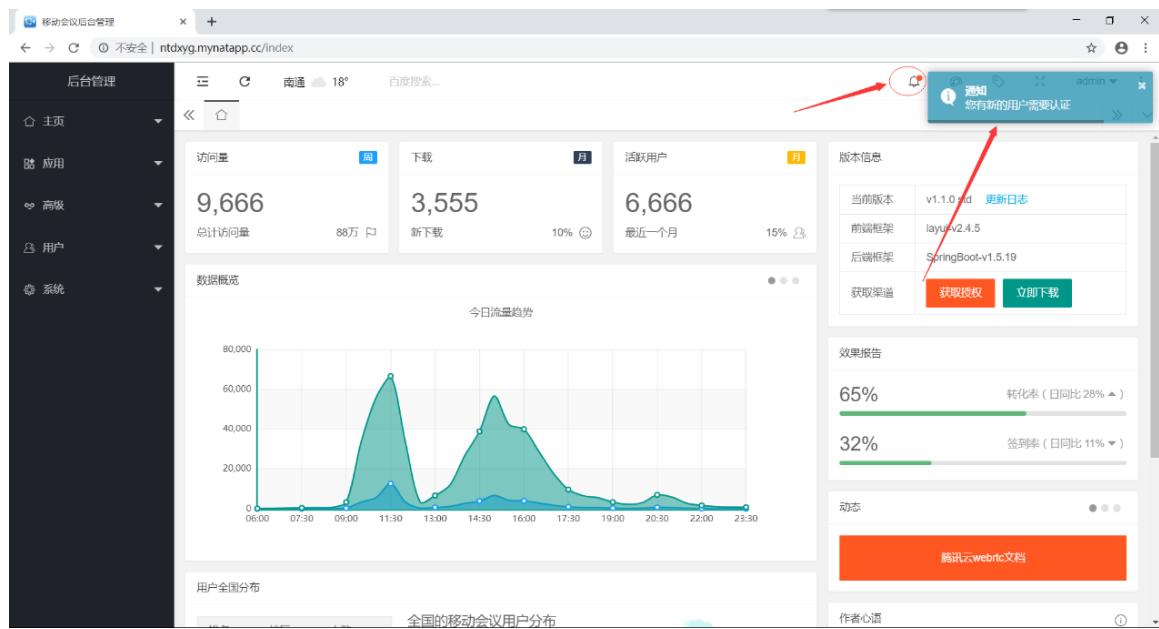


图 5.17 消息模块功能

点击提示框或者右上角的消息图标时转到消息列表中如图 5.18 所示，管理员可以阅读详细内容，如图 5.19 所示，选择已读和删除功能。未读的文件标题为红色，点击阅读后会变会黑色。

全部消息 5542	
	时间
<input type="checkbox"/> 标题内容	2019-04-24 19:58:38
<input type="checkbox"/> 您有新的用户需要认证	2019-04-22 15:40:57
<input type="checkbox"/> 您有新的用户需要认证	2019-04-22 15:37:49
<input type="checkbox"/> 您有新的活动需要审核	2019-04-22 12:12:51
<input type="checkbox"/> 您有新的活动需要审核	2019-04-22 12:08:39
<input type="checkbox"/> 您有新的活动需要审核	2019-04-21 18:22:42
<input type="checkbox"/> 您有新的用户需要认证	2019-04-21 18:19:14
<input type="checkbox"/> 您有新的用户需要认证	2019-04-21 18:19:11
<input type="checkbox"/> 您有新的用户需要认证	2019-04-21 18:19:02
<input type="checkbox"/> 您有新的活动需要审核	2019-04-19 21:37:17

图 5.18 消息列表



图 5.19 消息详情

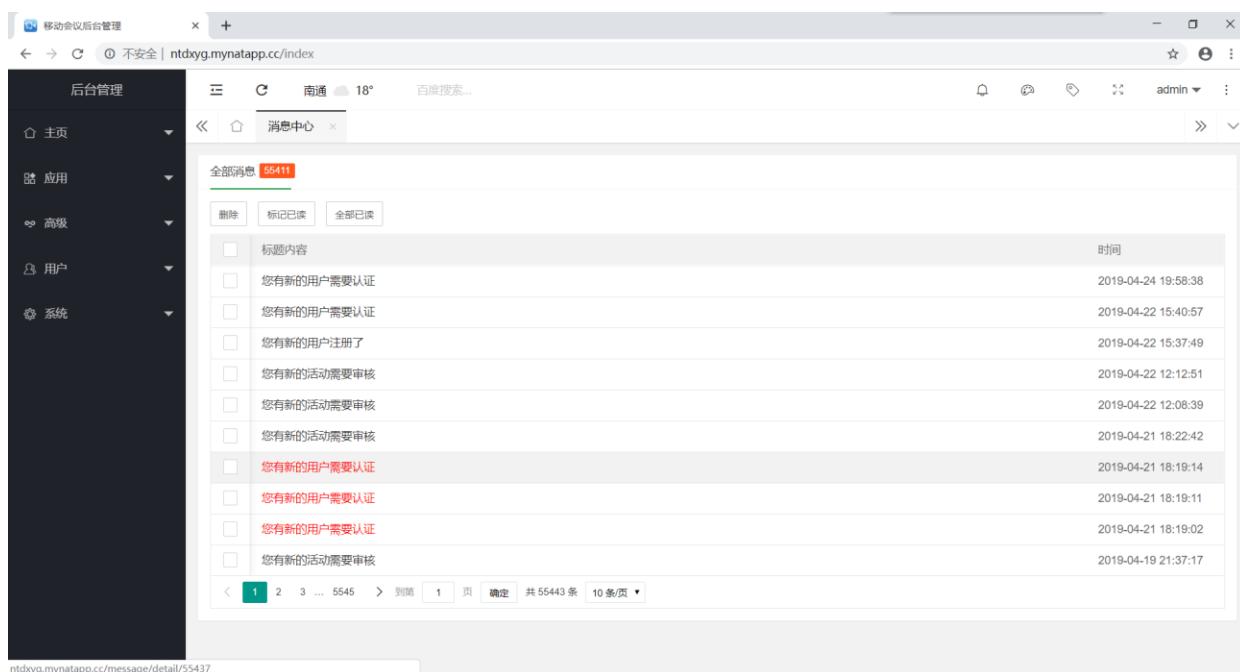


图 5.20 阅读后字体颜色变化

5.9 活动互动模块

移动会议实时互动系统，最重要的就是互动的实现，本系统有 2 种互动模式：直播类会议和多人视频类会议。直播类会议在用户申请通过之后会以邮件的方式发送到用户的邮箱，里面会介绍相关推流软件的使用以及具体的直播地址，会议开始前，可以对参与者们进行签到功能，在房间内，参与者通过手机客户端进行弹幕的发送，申请者可看到弹幕并

与他们进行互动，比如抽奖功能等等；多人视频会议考虑到微信小程序的使用，选择了腾讯云音视频服务，使用 WebRTC 进行多端的视频互动。界面图如下：

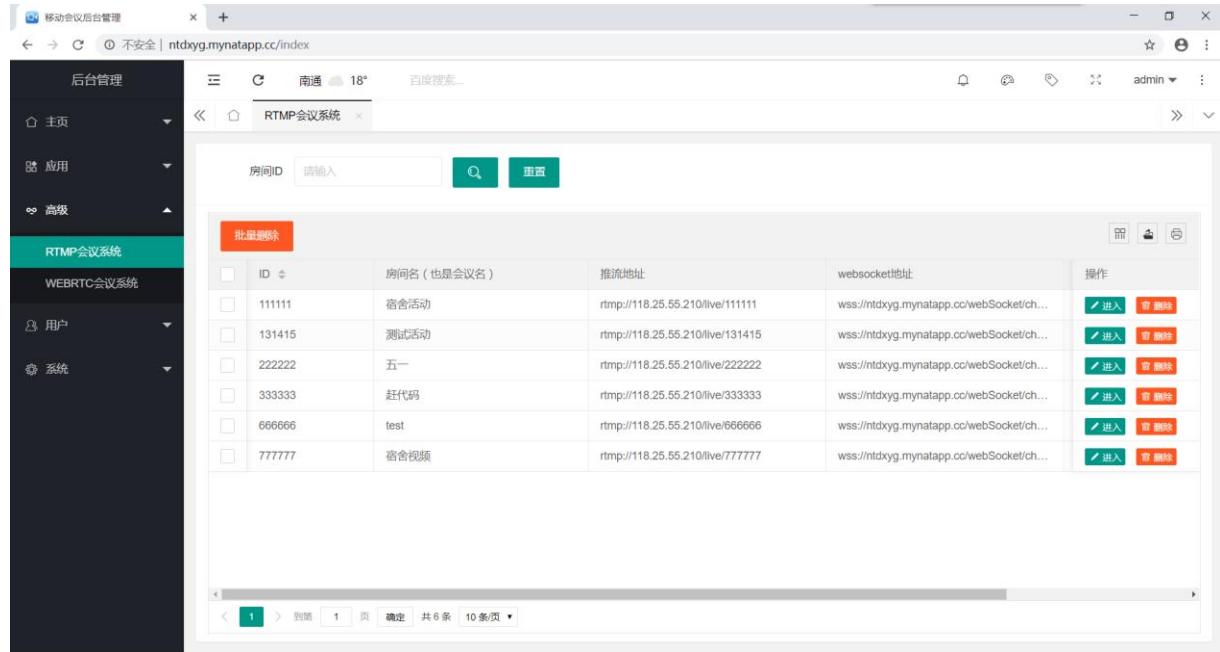


图 5.21 房间列表

会议活动申请者可以通过邮件里的相关信息打开直播页面并开启直播了。页面上能显示会议中需要展示的文件（通过 OBS 软件进行设置），会议在线的人数，与会人员发出的弹幕等等，若弹幕过多可以点击右上角的清除弹幕功能。

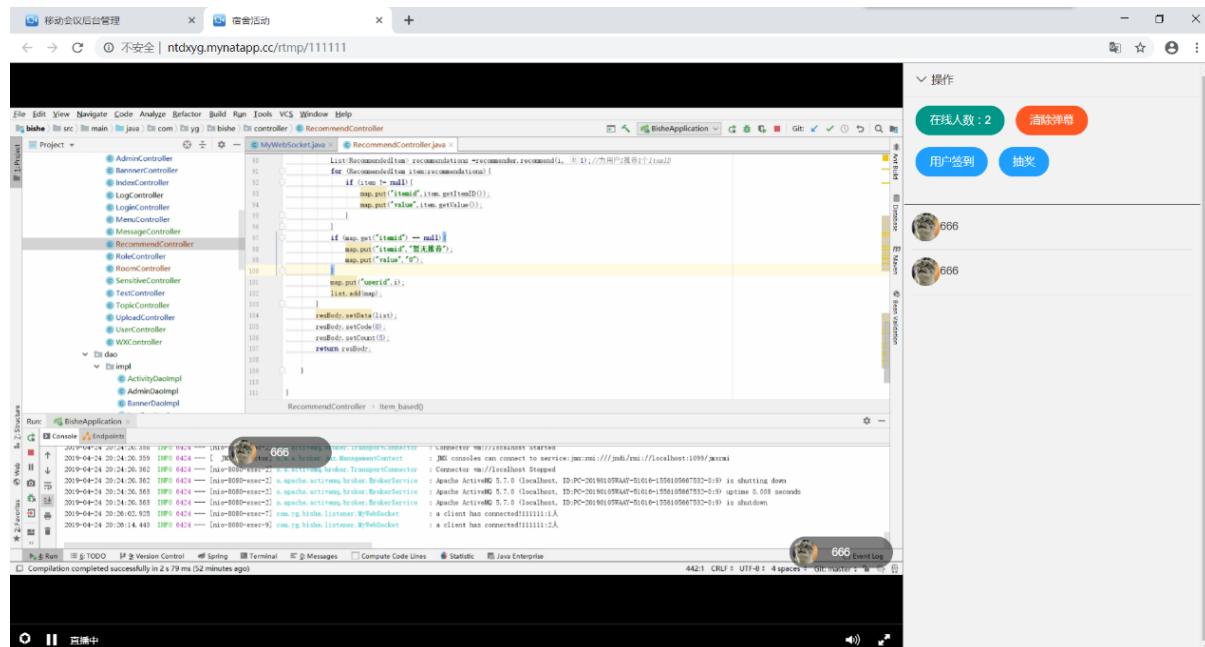


图 5.22 弹幕功能

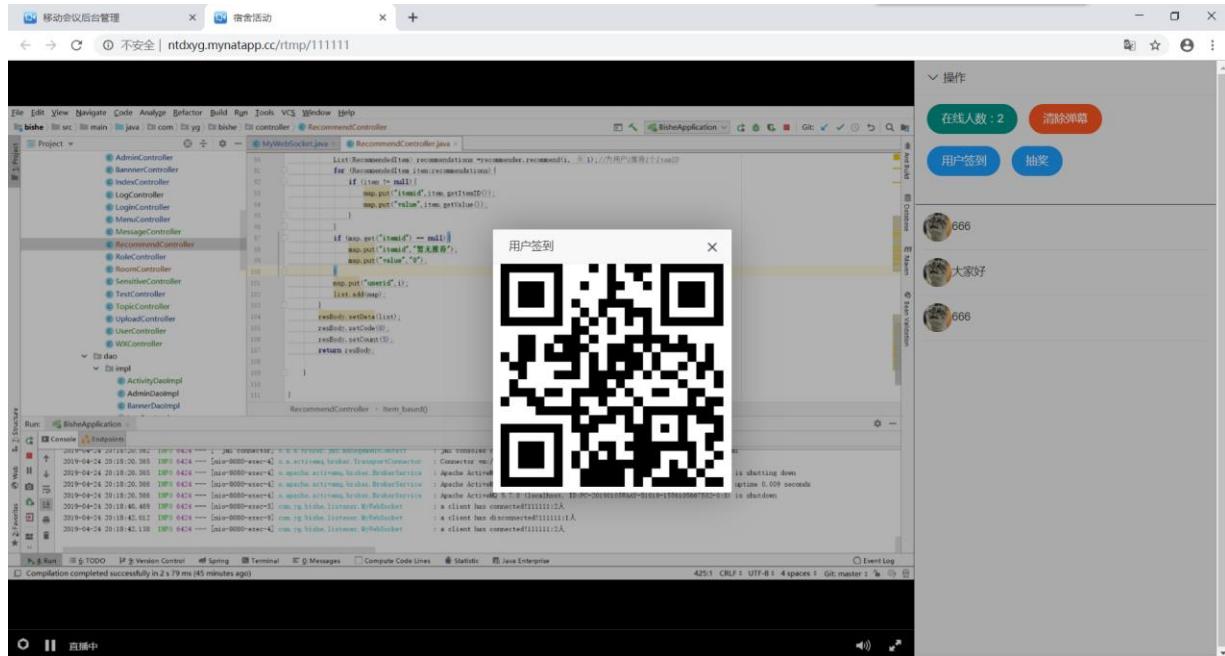


图 5.23 二维码签到

会议活动申请者可以在会前进行用户的签到：申请者点击右上方“用户签到”按钮，与会者通过客户端的扫一扫功能进行扫码签到。活动过程中申请者可以点击“抽奖”按钮进行会间的抽奖活动，以活跃会议气氛，抽奖页面如下图所示：在右下角进行中奖人员的人数限制，点击开始按钮抽奖开始，再点击停止按钮，即可抽出中奖人并会显示在中奖名单中。

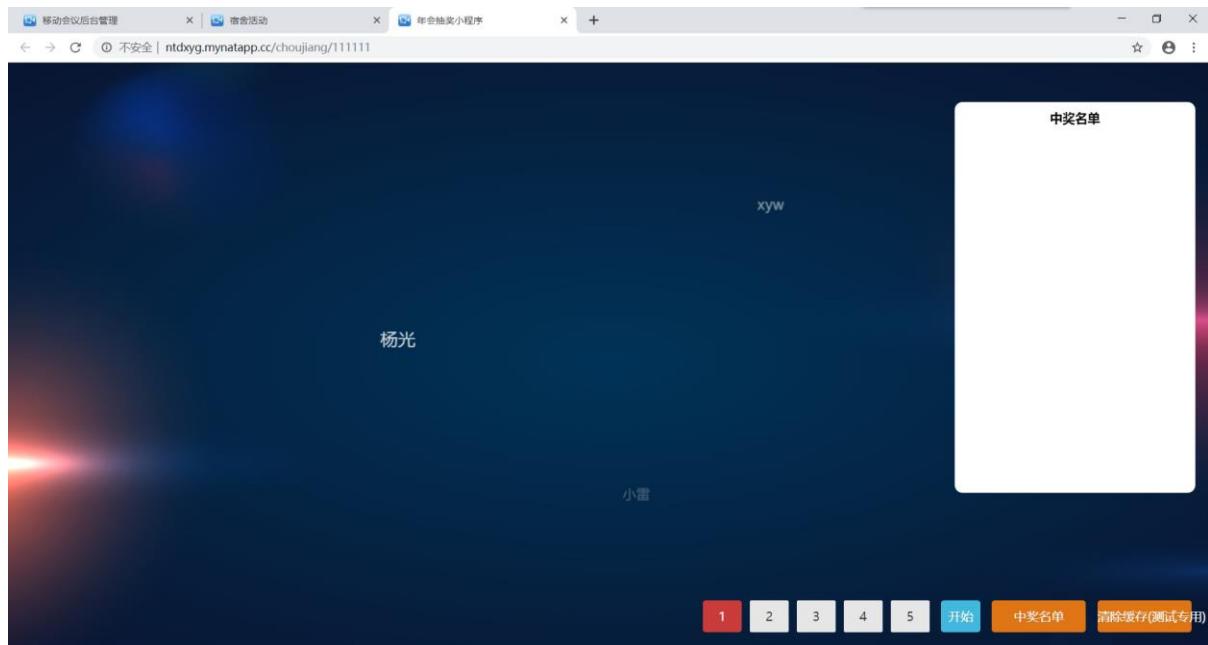


图 5.24 抽奖页面

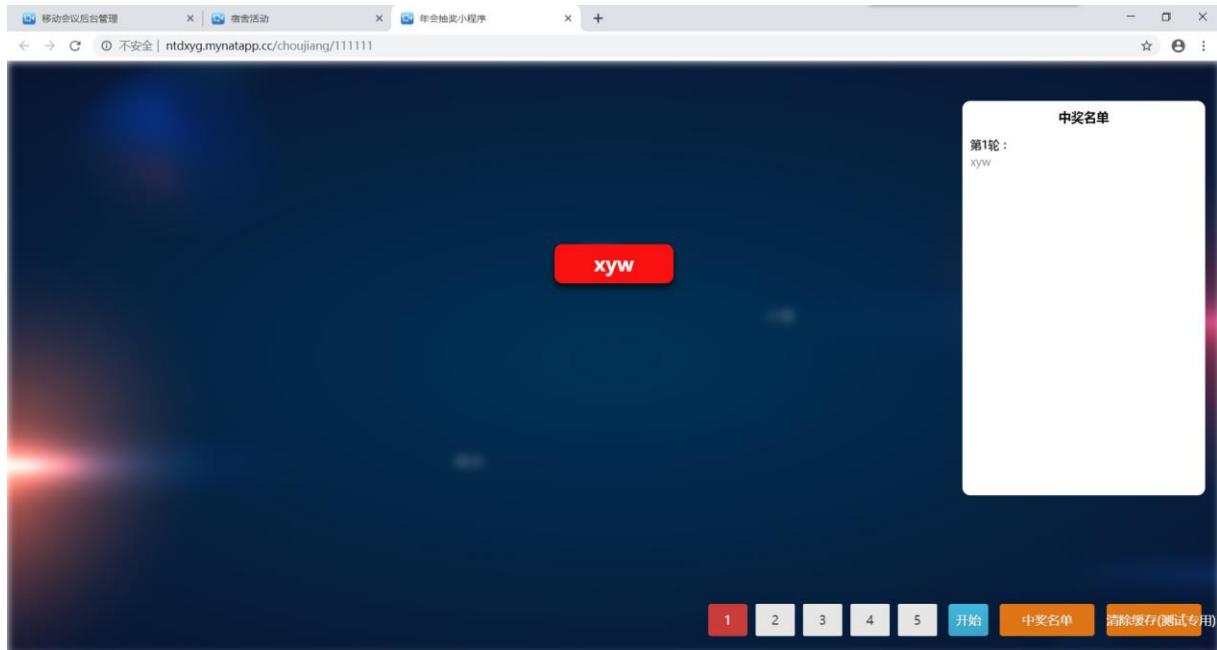


图 5.25 抽奖结果

在线视频会议活动：用于用户远程视频讨论会议的子系统，用户在客户端申请并参与会议，在管理端，管理员可以进入房间查视有无违规现象等下。

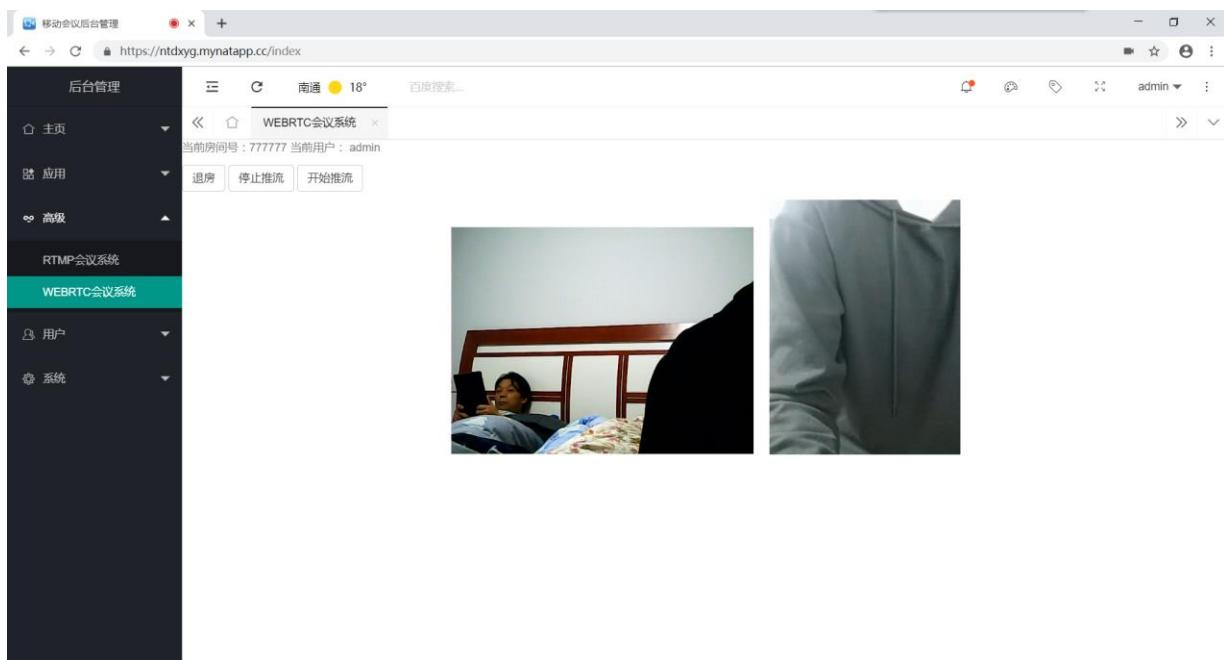


图 5.26 多人视频会议

5.10 系统模块

本系统的系统模块主要是对后台的监控、管理与设置，具体如下图所示。

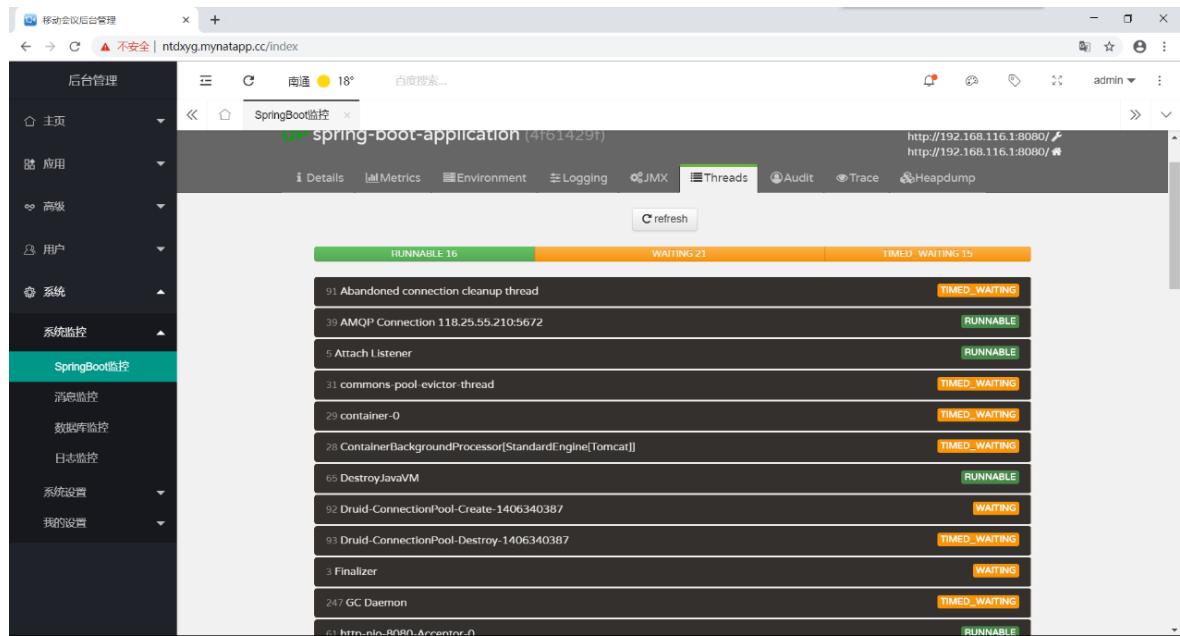


图 5.27 Spring Boot 监控

Spring Boot 监控是本系统集成的 Spring BootAdmin，用于系统详情的可视化监控，可以查看到 details, metrics, environment, logging, JMX, threads 等等，若系统崩溃还能以邮件方式通知管理员。

消息队列监控是本系统集成的 RabbitMQ 的可视化控制台，在这里可以查看到频道、交换机、队列等详情。

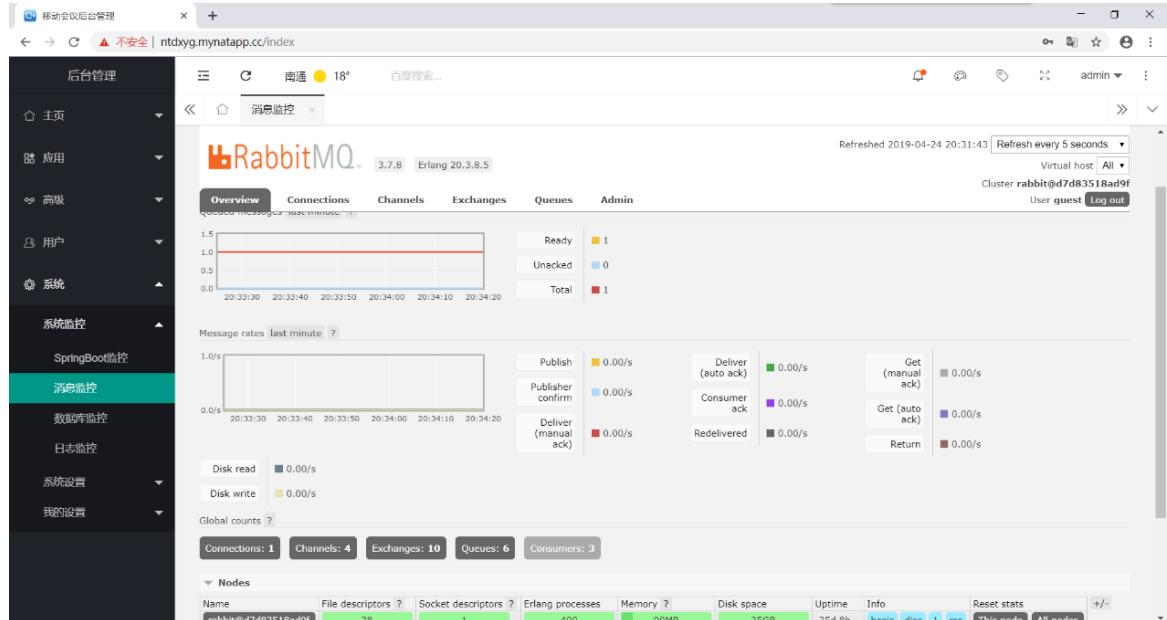


图 5.28 消息队列监控

The screenshot shows a web-based management interface for a mobile conference system. The left sidebar has a dark theme with categories like '后台管理', '应用', '高级', '用户', '系统', '系统监控', 'SpringBoot监控', '消息监控', '数据库监控' (which is highlighted in green), '日志监控', '系统设置', and '我的设置'. The main content area is titled 'SQL Stat View JSON API' and displays a table of recent database queries. The columns include N (query number), SQL (query text), 执行数 (execute count), 执行时间 (execution time), 错误 (error), 事务执行 (transaction execution), 锁表数 (lock table count), 更新行数 (update row count), 读取行数 (read row count), 执行中 (executing), 最大并发 (maximum concurrency), 执行时间分布 (execution time distribution), 执行+RS时分布 (execution+RS time distribution), 读取行分布 (read row distribution), and 更前行分布 (previous row distribution). The table lists 9 queries, mostly SELECT statements, with various execution times and resource usage.

图 5.29 数据库监控

数据库监控是本系统集成的 Druid 的可视化控制台，在这里可以查看到数据库的相关信息，如 SQL 语句监控，SQL 防火墙，URI 监控，Session 监控等等，若发现哪些 SQL 语句耗时较长即可考虑优化该语句，降低了开发人员对系统进行优化的成本。

The screenshot shows the log monitoring section of the system. The left sidebar includes '日志监控' under '系统监控'. The main area has tabs for '新增' (New), '批量删除' (Batch Delete), and '数据分析' (Data Analysis). The '数据分析' tab is active, showing a table of log entries with columns: ID, 登录人 (Login User), IP地址 (IP Address), 运营商 (Operator), 登录地址 (Login Address), 登录方式 (Login Method), 登录时间 (Login Time), and 操作 (Operations). Below the table are two charts: a pie chart titled '登录运营商比例' (Login Operator Ratio) showing proportions for 城域网 (Circuit Network), 移动 (Mobile), 联通 (China Telecom), and 电信 (China Telecom); and a bar chart titled '登陆次数统计' (Login Count Statistics) showing the number of logins for admin users via mobile (4) and circuit network (459).

图 5.30 日志监控并使用 echarts^[13]可视化分析

日志监控是本系统自主设计的一个登陆日志收集系统，通过用户登陆时获取的 IP 地址进行分析地域和运行商等等，并使用 echarts 对日志数据进行了可视化的分析，包括了以饼图的形式对运行商占比的统计图和管理员登陆次数的比较图等等。

系统网站设置是本系统对自身网站的设置，包括了域名，网站名称的设置和 SMTP 的相关设置（用于邮件的发送等等）。

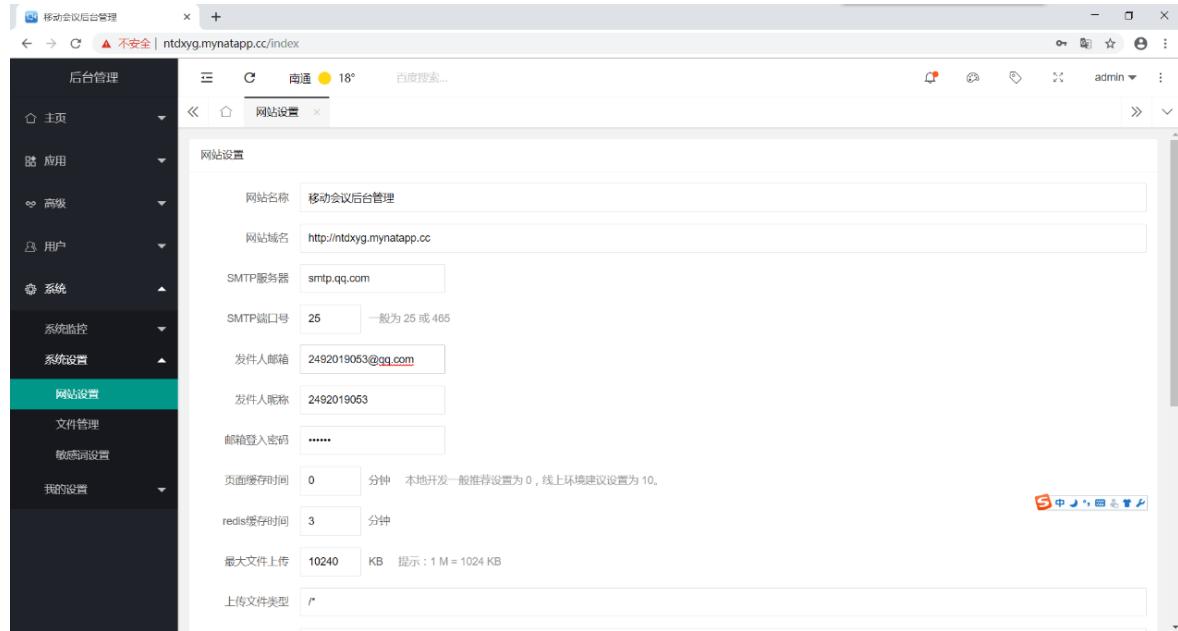


图 5.31 系统网站设置

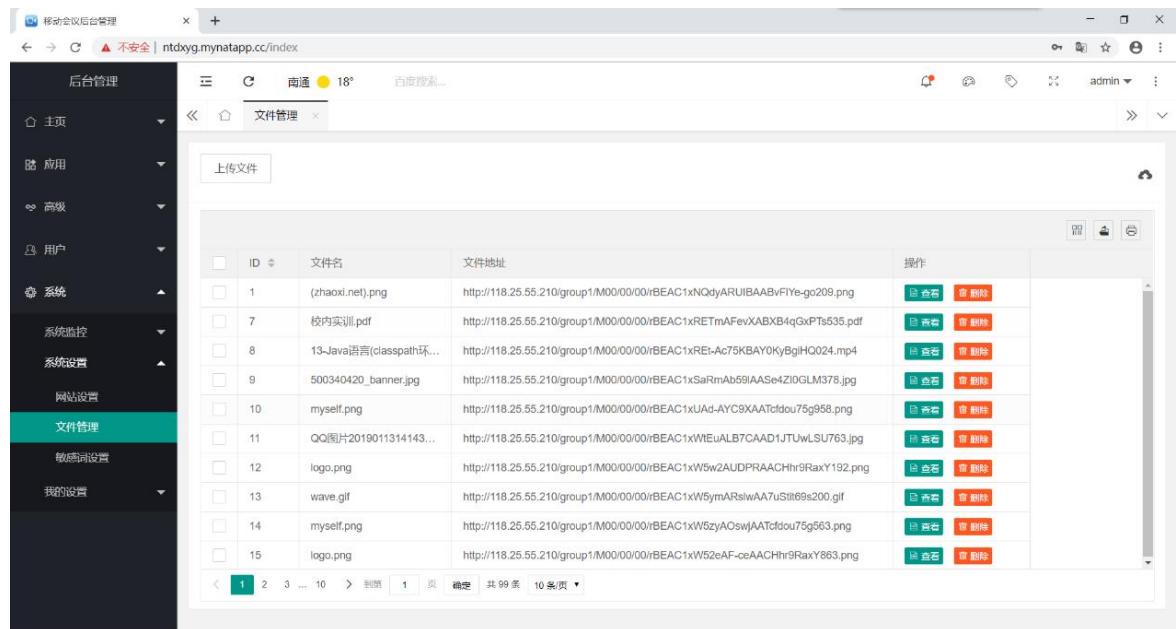


图 5.32 系统文件管理

系统文件管理是本系统对自身所搭建的文件服务器的设置，包括了用户、管理员进行图片上传或者文件上传后的文件管理，管理员也可以在此页面进行文件的上传测试并删除

等等功能。关键字设置是本系统将关键字存入到数据库并以 DFA 算法实现系统关键字过滤的设置，管理员可以点击新增和删除来管理关键词，也可导出文件或者打印表格等等。

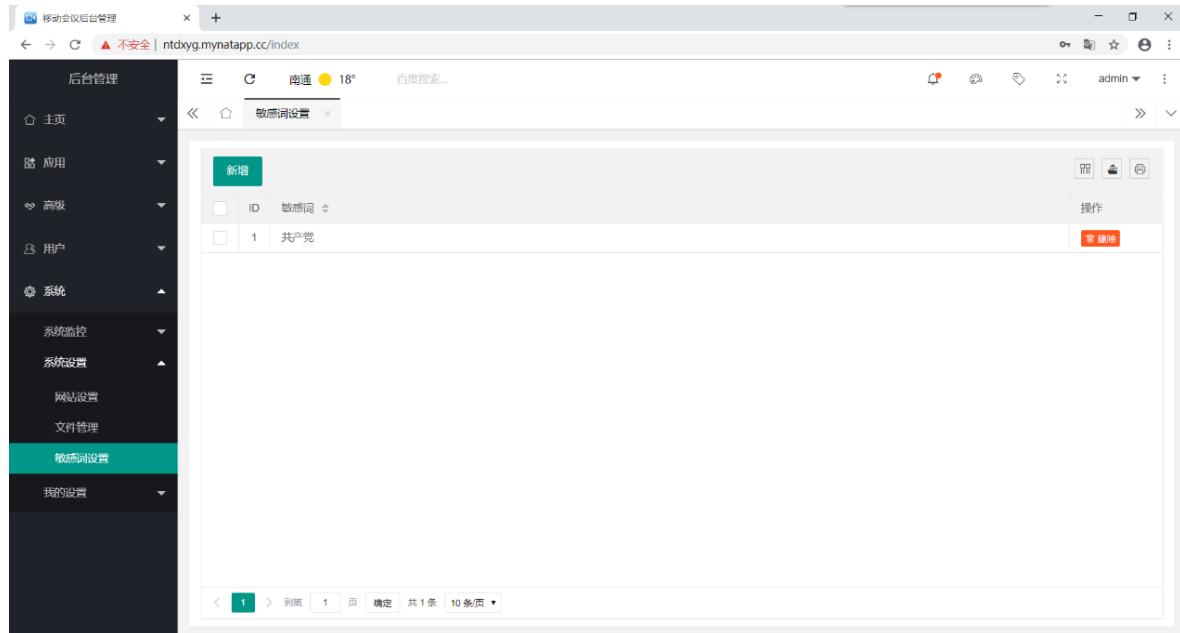


图 5.33 关键词设置

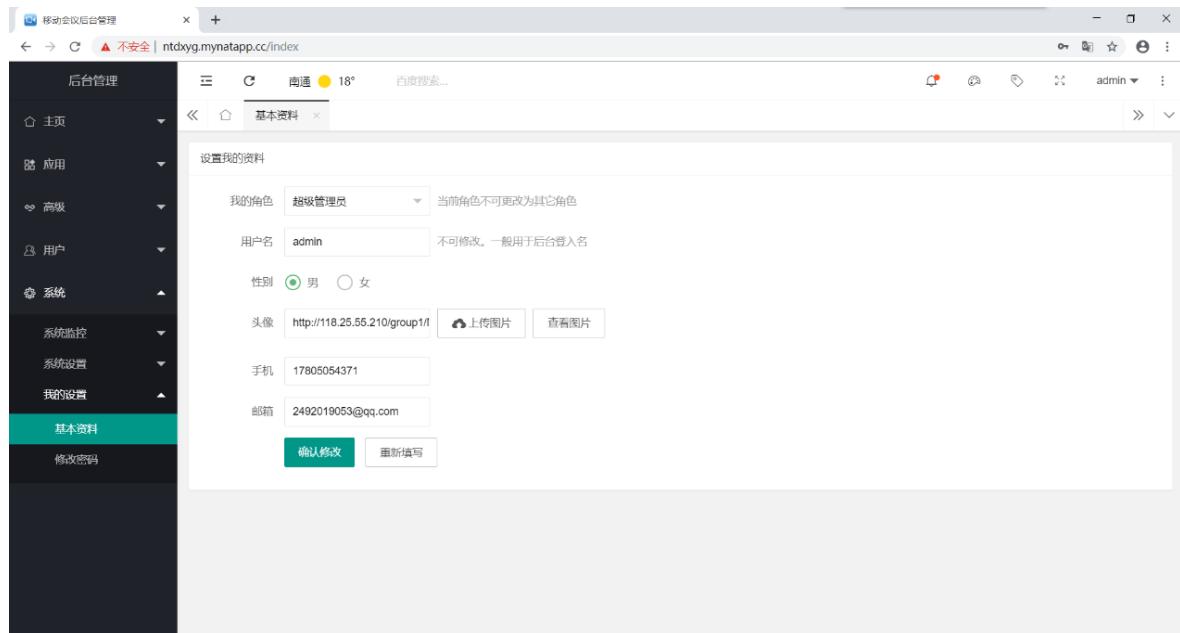


图 5.34 管理员资料设置

管理员资料设置是本系统对管理员自身的相关信息的设置，比如性别，头像，手机号和密码等等的设置，如图 5.34，5.35 所示。

本系统在后台页面还出了 2 个小彩蛋：当前的天气情况和标签功能。如图 3.36 所示天气情况通过心知天气获取详情，标签功能通过浏览器的缓存进行实现，管理员可以将他当做备忘录使用。

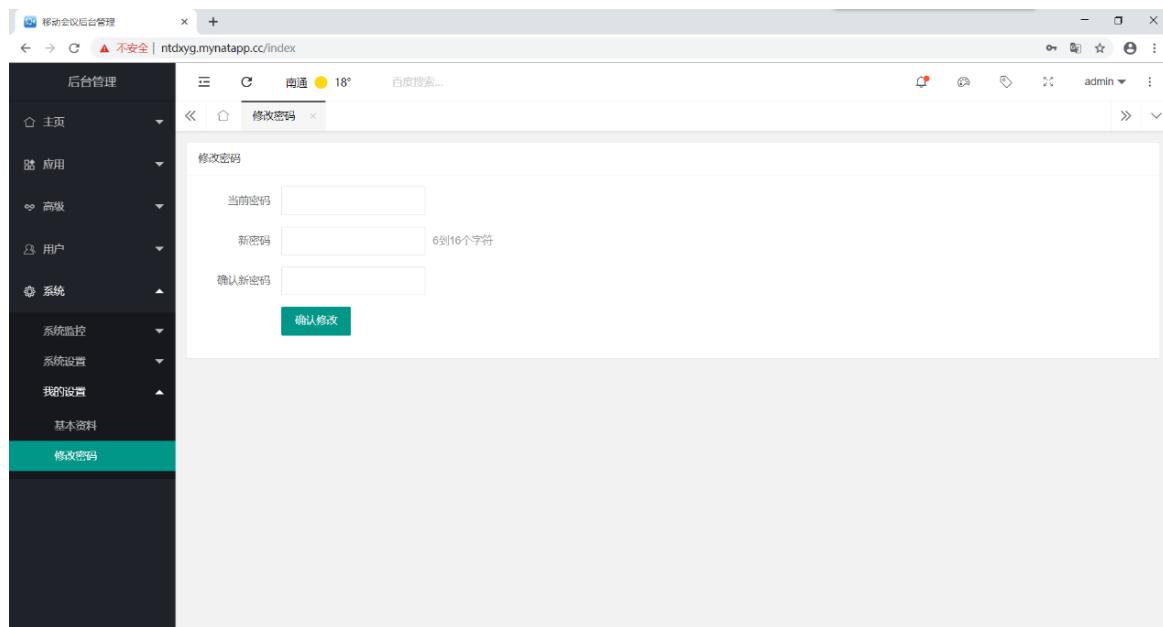


图 5.35 管理员密码设置

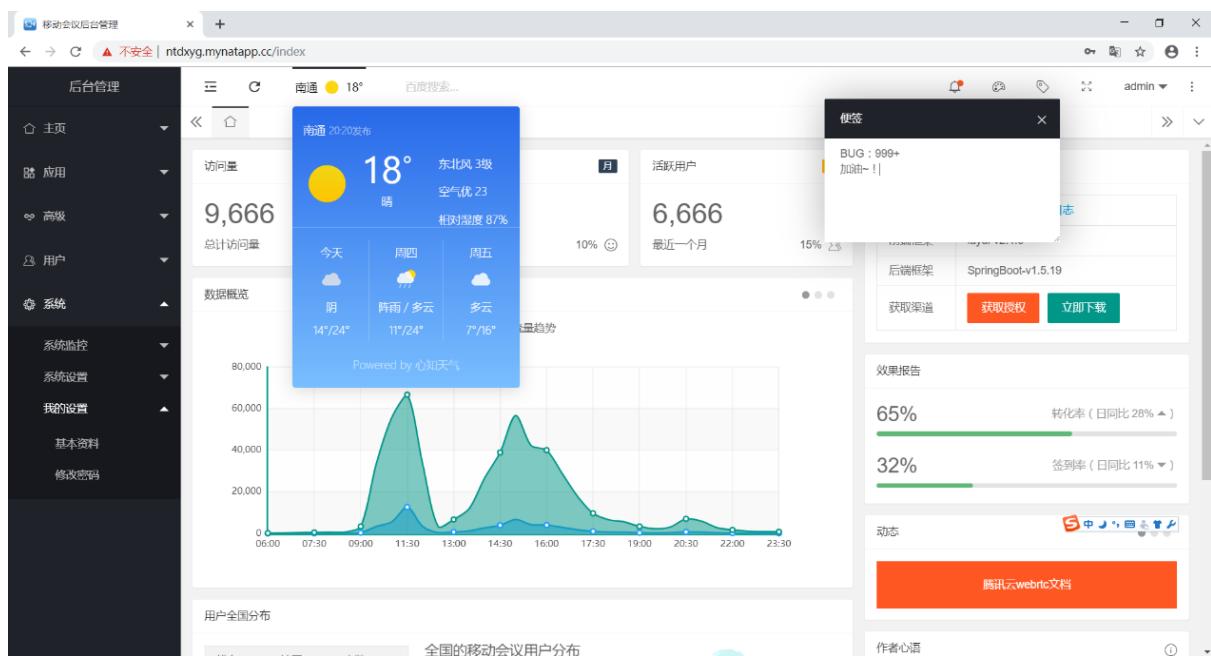


图 5.36 天气及页面便签

第六章 系统测试

6.1 单元测试

6.1.1 单元测试工具简介

在单元测试中，对每个功能模块都设计了测试用例，并使用 IDEA^[14]自带的代码覆盖率工具，直接测试类名右键 Run ‘MyClassTest’ with Coverage 或工具栏上的选项运行特定模式的测试，来进一步搜集测试用例的覆盖信息。如图所示在文件夹上点击 Run Test 便可以在右边的框里看到 Coverage 的结果，包括每个类的覆盖率。

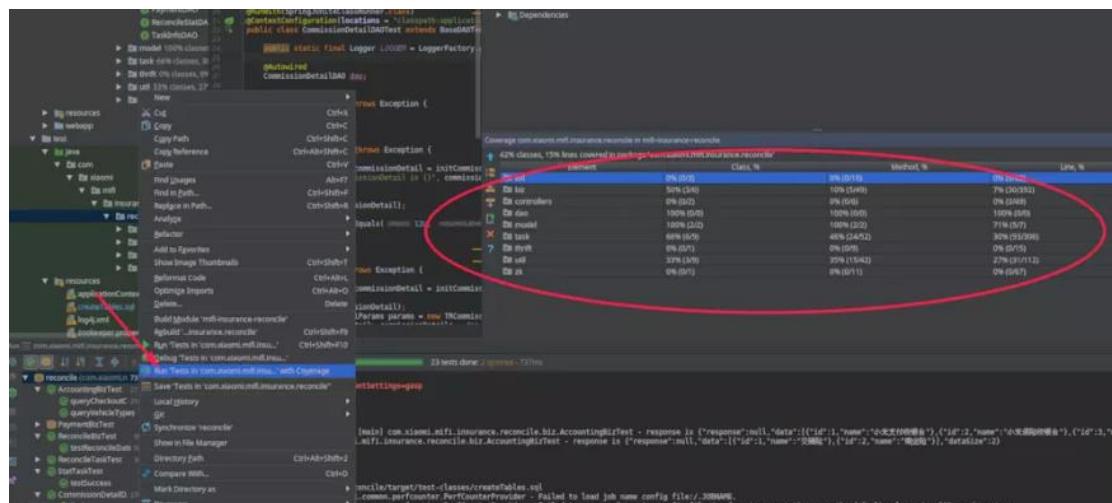


图 6.1 使用方法

idea 更强大的地方可以看到哪一行是否覆盖，如图所示绿色部分表示已经覆盖的地方，红色部分表示单元测试还没有覆盖的行。

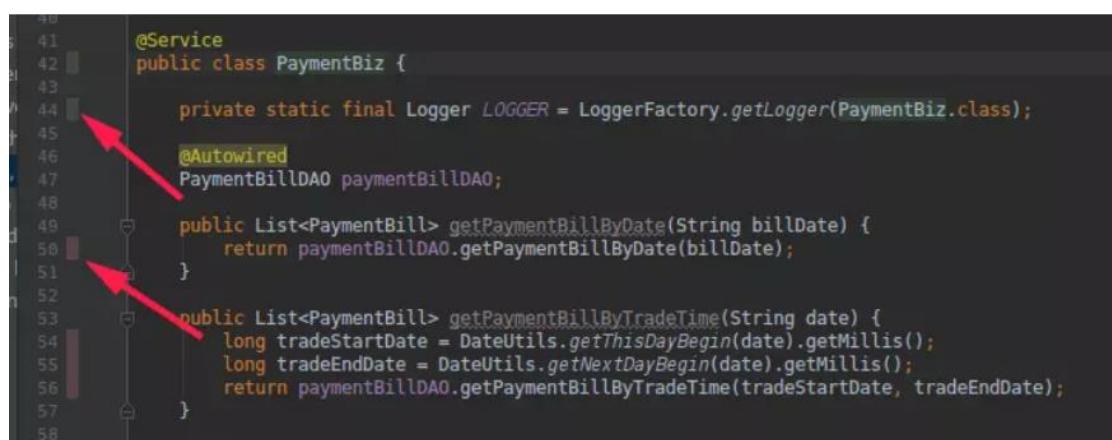


图 6.2 测试分析

6.1.2 单元测试用例设计

针对移动会议实时互动系统管理端，针对不同的模块设计了测试用例，下面给出了本系统部分测试用例。

(1) 面向后台管理员的测试用例

```

1. @Test
2. public void getAdminList() {
3.     ResBody resBody = new ResBody();
4.     int count = adminService.getCount();
5.     List<Admin_Role> admin_roles = adminService.findAllAdmin(1,10);
6.     resBody.setCount(count);
7.     resBody.setCode(0);
8.     resBody.setData(admin_roles);
9.     assertNotNull(resBody.getData());
10. }

```

(2) 面向广告的测试用例

```

1. @Test
2. public void getBannerList() {
3.     ResBody resBody = new ResBody();
4.     int count = bannerService.getCount();
5.     List<Banner> banners = bannerService.findAllBanner(1,10);
6.     resBody.setCount(count);
7.     resBody.setCode(0);
8.     resBody.setData(banners);
9.     assertNotNull(resBody.getData());
10. }

```

6.2 接口测试

6.2.1 接口测试工具简介

本系统与客户端数据访问都是通过 API 接口进行，所以接口测试是整个测试流程的重中之重。本系统选择 Swagger^[15]进行接口测试。

现在互联网公司的主流慢慢演变成前后端分离的技术栈了：前端专注于页面渲染和开发，后端专注于服务器和接口的开发，两者间通过 API 进行数据交接的接口。所以 API 接口测试变得尤为重要，Swagger 就是一个转为 API 文档和测试而生的框架。

6.2.2 接口测试结果分析

Swagger 提供可视化测试界面，选择所需测试的接口，输入相应的参数，即可获得相应的返回数据。如下图所示

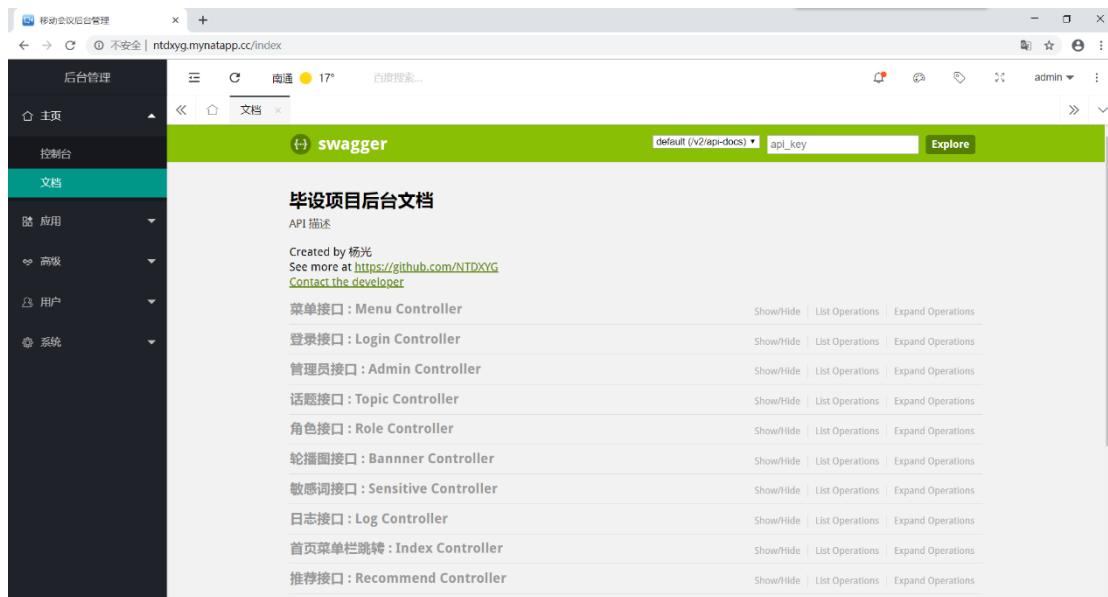


图 6.3 Swagger-UI 界面

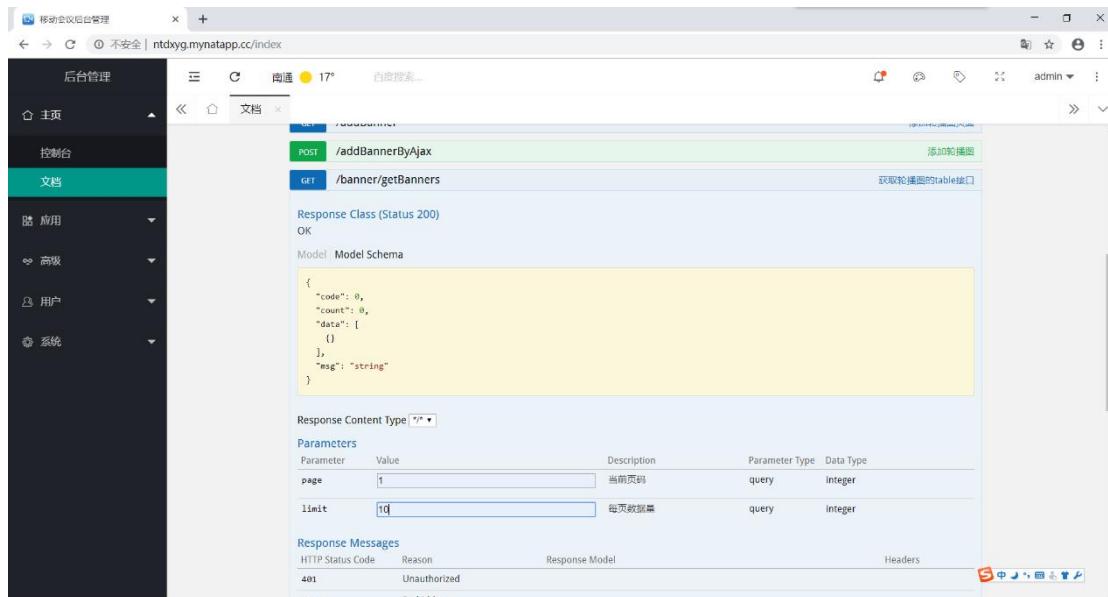


图 6.4 输入参数

选择了广告接口 Banner 中的“/banner/getBanners”，点击看到需要输入 2 个参数：page 和 limit，当前页码和每页的数据量，填入 1 和 10，点击 try it out! 即可出现相关的响应数据进行对比测试。

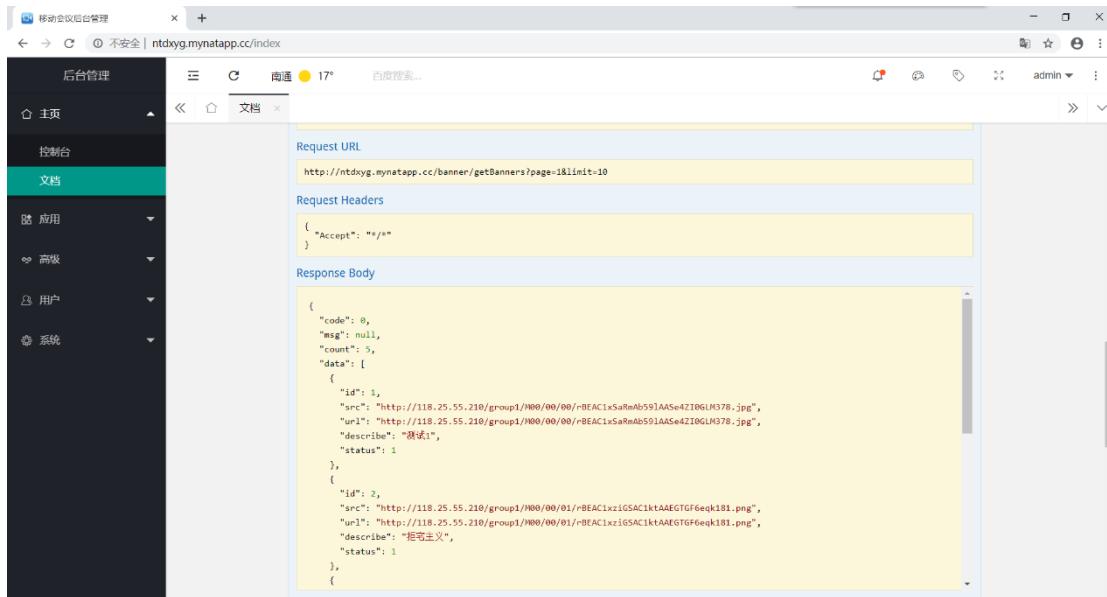


图 6.5 响应结果

6.3 性能测试

6.3.1 性能测试工具简介

运营方管理端的压力测试使用的 LoadImpact。LoadImpact 是一种领先的在线负载和性能测试服务，可让您在本地或通过 Internet 测试您的网站，Web 应用程序，移动应用程序或 API。LoadImpact 为全球客户提供基于云的负载测试和自动结果分析作为在线服务。事实上，LoadImpact 的负载测试服务非常受欢迎，已经被用于执行超过 1,500,000 次负载测试。使 LoadImpact 成为 Internet 上最受欢迎的负载测试服务。LoadImpact 使 DevOps 团队能够以更低的成本更快地向市场提供高性能网站，Web 应用程序和 API。。

页面截图如图 6.6 所示：

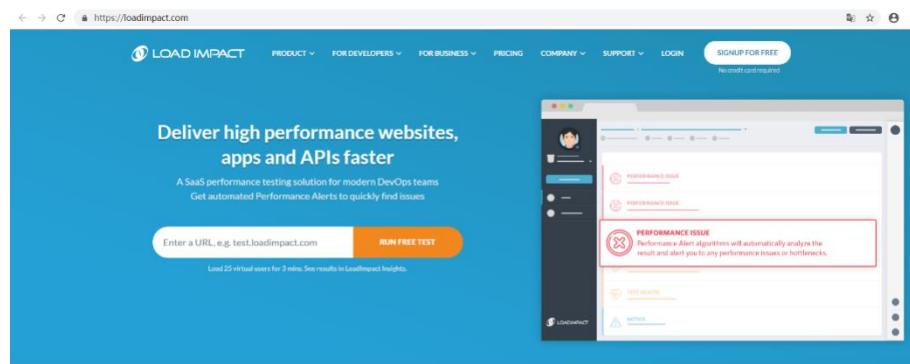


图 6.6 性能测试页面截图

6.3.1 性能测试报告分析

(1) Performances status:



图 6.7 Performances status

从图 6.7 可以看出，蓝色代表的响应时间，绿色代表的访问人数，紫色代表的请求速度。在 3 分钟内，模拟了 25 个用户对本系统发起了大量的请求，从图中分析出本系统的性能还是比较客观的，下面看下具体的 Http 请求情况与指标分析。

(2) Http 请求情况

3 分钟内，25 个用户总控进行了 1.03K 的访问次数，其中失败次数为 222 次，成功率约为 80%，考虑到本系统通过 Natapp^[16]进行内网穿透，性能较低，此次的 http 请求情况测试下来较为客观。

The figure is a screenshot of a tool displaying a list of HTTP requests. The top navigation bar has tabs: CHECKS, HTTP (selected), ANALYSIS, and SCRIPT. The HTTP tab shows 222 / 1.03K total requests. Below the tabs is a table with columns: All, URL, METHOD, STATUS, COUNT, MIN, AVG, STDDEV, P95, P99, and MAX.

All	URL	METHOD	STATUS	COUNT	MIN	AVG	STDDEV	P95	P99	MAX
✓	cdn.bootcss.com./jquery.min.js	GET	200	129	286ms	618ms	244ms	935ms	1.42s	2.38s
✗	ntdxyg.mynatapp.cc./layer.js	GET	429	CLICK TO SEE THE GRAPH	394ms	517ms	843ms	2.39s	3.36s	
✗	ntdxyg.mynatapp.cc./layer.css?v=3.1.1	GET	429	18	203ms	361ms	219ms	840ms	845ms	847ms
✗	ntdxyg.mynatapp.cc/login;jsessionid=878...	GET	429	55	195ms	356ms	267ms	766ms	1.26s	1.56s
✓	ntdxyg.mynatapp.cc./element.js	GET	200	59	219ms	354ms	297ms	757ms	1.55s	2.17s
✓	ntdxyg.mynatapp.cc./aiwrap.png	GET	200	64	223ms	351ms	262ms	686ms	1.51s	1.57s
✓	ntdxyg.mynatapp.cc/getCode	GET	200	80	239ms	342ms	162ms	805ms	860ms	889ms
✓	ntdxyg.mynatapp.cc/login;jsessionid=878...	GET	200	78	221ms	329ms	183ms	718ms	1.02s	1.23s
✗	ntdxyg.mynatapp.cc./form.js	GET	429	50	195ms	327ms	199ms	775ms	831ms	848ms
✗	ntdxyg.mynatapp.cc./admin.js	GET	429	44	203ms	325ms	329ms	990ms	1.62s	1.72s

图 6.8 http 请求情况

(3) 指标分析



图 6.9 指标分析

图 6.9 的指标分析从用户并发数，检查失败数，响应时间，请求率，CPU 使用率，内存使用率 5 个角度进行了分析，从图中看，用户并发数逐渐上升对系统的内存使用率并无太大影响，CPU 使用率也大致维持在 1% 到 2% 之间。

第七章 总结与展望

7.1 总结

本文实现了移动会议实时互动系统的运营方管理端。在完成毕设的这段时间里，通过自学学习了大量的较为前沿的开发知识，本系统基于 Spring Boot + Lay UI 完成，在后续的项目部署上可以通过直接导出 jar 包的方式运行中 Linux 云服务器中。同时使用的 Docker 容器，将 MySQL, Redis, RabbitMQ, Fast DFS 等服务器很轻松的部署，大大简化了后期维护的力度。

7.2 展望

移动会议实时互动系统仍然存在很多不足之处：

- (1)数据库表结构的优化与 SQL 语句的优化问题。
- (2)服务器性能需要进一步改善，本系统的推流服务器使用开源的 RED5，只能支撑 500 多人的并发量，另外由于小程序的原因不是自己搭建的 WebRTC 服务器而是使用的腾讯云，希望以后有机会换成自己搭建的服务器。
- (3)未能实现定时任务进行活动消息的推送。考虑到有的活动持续不止一天，使用定时任务的话无法确定是定时每天还是固定的几天实现消息推送。
- (4)未能加入 Elastic Search^[17]全文搜索，系统使用的搜索仍是基于百度搜索引擎的搜索。未能使用 Multi-objective Cross-Project Defect Prediction^[18]对系统进行缺陷预测。
- (5)直播会议互动功能仍有欠缺，只实现了弹幕和抽奖功能，后期考虑加入投票等互动功能。

参考文献

- [1]易用性, 维基百科[EB/OL], <https://zh.wikipedia.org/wiki/易用性>
- [2]安全测试, 百度百科[EB/OL], <https://baike.baidu.com/item/安全测试>
- [3]易文康, 程骅, 程耕国. Shiro 框架在 Web 系统安全性上的改进与应用[J]. 计算机工程, 2018, 44(11):135–139.
- [4]张峰. 应用 SpringBoot 改变 web 应用开发模式[J]. 科技创新与应用, 2017(23):193–194.
- [5]陈清金, 陈存香, 张岩. Docker 技术实现分析[J]. 信息通信技术, 2015, 9(02):37–40.
- [6]曾超宇, 李金香. Redis 在高速缓存系统中的应用[J]. 微型机与应用, 2013, 32(12):11–13.
- [7]马巍, 武欣蝶, 郑翔, 张文强, 童玮. RabbitMQ 在实时监控系统中的应用[J]. 军事通信技术, 2017, 38(01):82–85.
- [8]余庆. 分布式文件系统 FastDFS 架构剖析[J]. 程序员, 2010(11):63–65.
- [9]李代立, 陈榕. WebSocket 在 Web 实时通信领域的研究[J]. 电脑知识与技术, 2010, 6(28):7923–7925+7935.
- [10]张远. 视频会议系统中 TCP/RTMP/WebSocket 协议间转换的研究与实现[D]. 华南理工大学, 2016.
- [11]刘璐, 董小国. Red5 Flash 服务器研究[J]. 网络安全技术与应用, 2009(06):78–79+32.
- [12]屈振华, 李慧云, 张海涛, 龙显军. WebRTC 技术初探[J]. 电信科学, 2012, 28(10):106–110.
- [13]Deqing L , Honghui M , Yi S , et al. ECharts: A declarative framework for rapid construction of web-based visualization[J]. Visual Informatics, 2018:S2468502X18300068-.
- [14]Mahmood, J.,Reddy, Y.R.. Automated refactorings in Java using IntelliJ IDEA to extract and propagate constants[P]. Advance Computing Conference (IACC), 2014 IEEE International,2014.
- [15]. SmartBear; New Swagger Tool Simplifies API Testing and OpenAPI Documentation[J]. Computers, Networks & Communications,2018.
- [16]Lei Wang. Design and Research on the Test of Internal Network Penetration Test[P]. 2018 International Conference on Network, Communication, Computer Engineering (NCCE 2018),2018.
- [17]Konstantinos N. Vavliakis,George Katsikopoulos,Andreas L. Symeonidis. E-commerce Personalization with Elasticsearch[J]. Procedia Computer Science,2019,151.
- [18] Canfora G , Lucia A D , Penta M D , et al. Multi-objective Cross-Project Defect Prediction[C]. 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation. IEEE, 2013.

致 谢

大学四年，四年恍然间就接近尾声，这是人生中最美好、独立支配时间称得上是自由的四年。在这四年里，我收获了很多感动与知识，也认识了很多美好的老师和学生。在此，我要感谢培养了我四年的母校，真心感谢这四年，在我的成长道路上帮助过我，指点过我的人。首先，在论文即将完成之际，回顾紧张却又充实的学习和开发过程，非常感谢指导老师顾卫平老师和陈翔老师，他们工作负责，对待学生亲切、有耐心，谢谢他们在毕业设计过程中带给我的指导和帮助，遇到这样的老师是我的幸运。

其次，我想感谢我的家人，这四年里带给我对人生不同的理解与体会，家人永远是我最大的动力与支撑，他们爱我、关心我、对我好，我永远牢记，我也很珍惜每次相聚和交流，希望我能用更大的努力换来更好的自己，不辜负他们对我的关爱，这也是我此时唯一能做的地方。

最后向参加论文审稿和答辩的各位老师致以最诚挚的谢意。