南京郵電大學

软件基础实践 报告

（ 2021/ 2022 学年 第 1 学期）

题　　目：大型投票系统

专　　业　软件工程

学生姓名列表　谭永锋 徐子健 王雪臣

班级学号列表　B19040229 B19111330 B19050115

指 导 教 师　毛 燕 琴

指 导 单 位　软件工程系

日　　期　2021-11-4

投票系统

一、 概论

开发一个大型投票网站，为企业进行宣传和快速累积用户。此次开发的投票系统首先分为后台管理模块和用户投票模块两个部分：

在普通用户投票模块，用户可以查看全部的投票频道及其描述，并从中选择频道进行投票。投票时，同一设备、同一 IP、同一频道仅可投票一次，同时为了应对特殊状况，比如说校园网共享 IP、用户更换 IP 恶意刷票等情况引入了特定机制来解决这些问题。

后台管理页面分为首页、所有频道和个人信息三部分。首页中可以查看网站的投票数据和投票趋势，在所有频道中管理员可以查看并修改每个频道的名称和简介，还可以查看每个频道的投票结果以及删除频道。在个人信息中管理员可以查看最近投票的用户 IP 和地址。

项目分工说明

学号	姓名	主要工作
B19040229	谭永锋	负责系统代码实现和维护部署
B19111330	徐子健	负责前端 UI 设计和系统数据导入
B19050115	王雪臣	负责系统后期测试和文档撰写

二、 设计与实现

(一) 系统设计

1. 系统主要架构

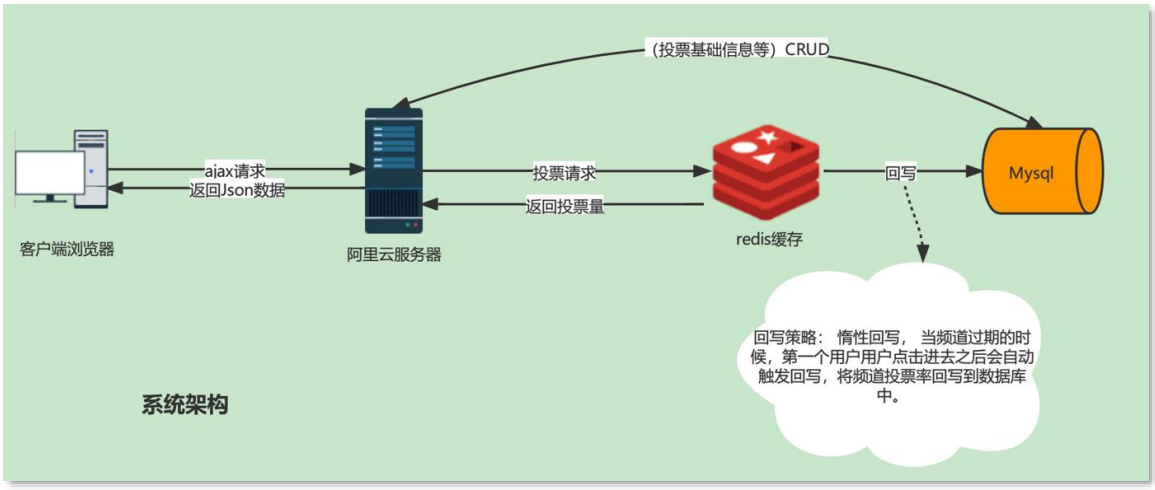


图 2.1.1 系统架构图

2. 后端主要架构

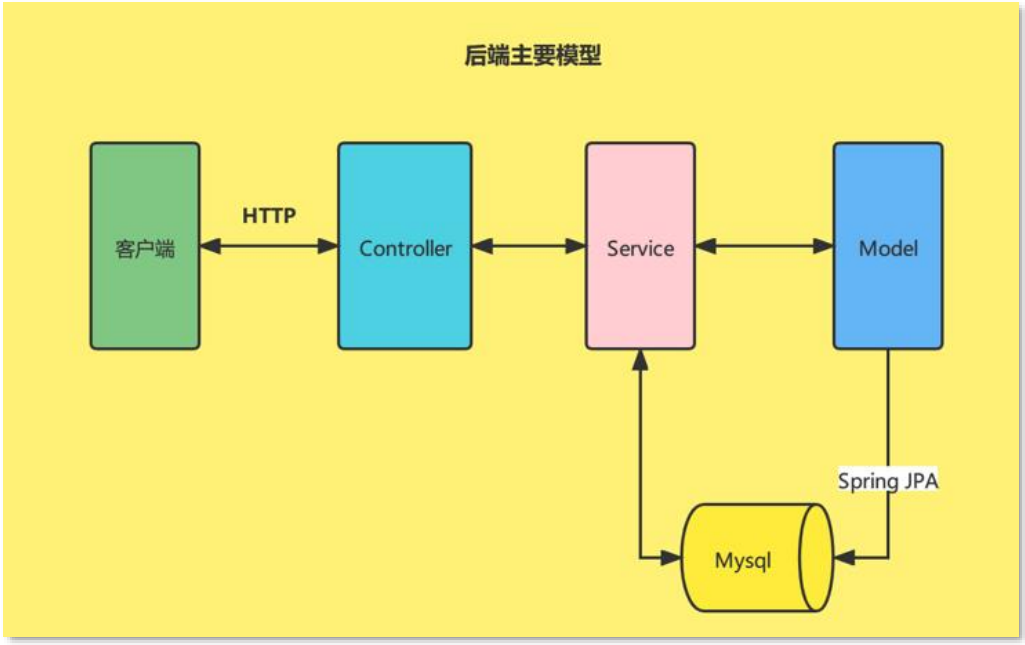


图 2.2.1　后端主要模型图

3. 前端设计

主体使用了 Google 推荐的 Materialize 风格，网站大部分采用模块化设计，简约时尚，同时为了时网站具有更好的交互功能，引入 JQuery 函数库来实现动态增添频道、投票项等功能。同时为了更好的展示投票结果，我们还引入了 Echarts 组件来根据数据来渲染扇形图、折线图等。

图 2.3.1-首页

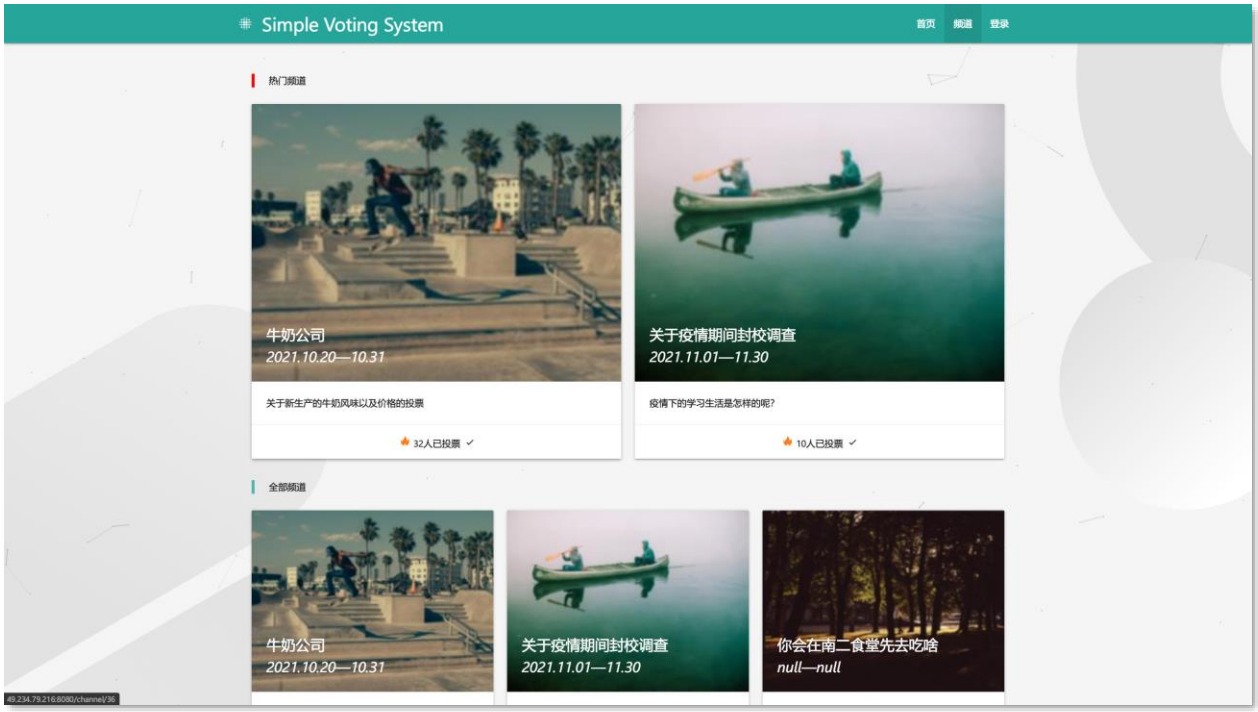


图 2.3.2 投票主页

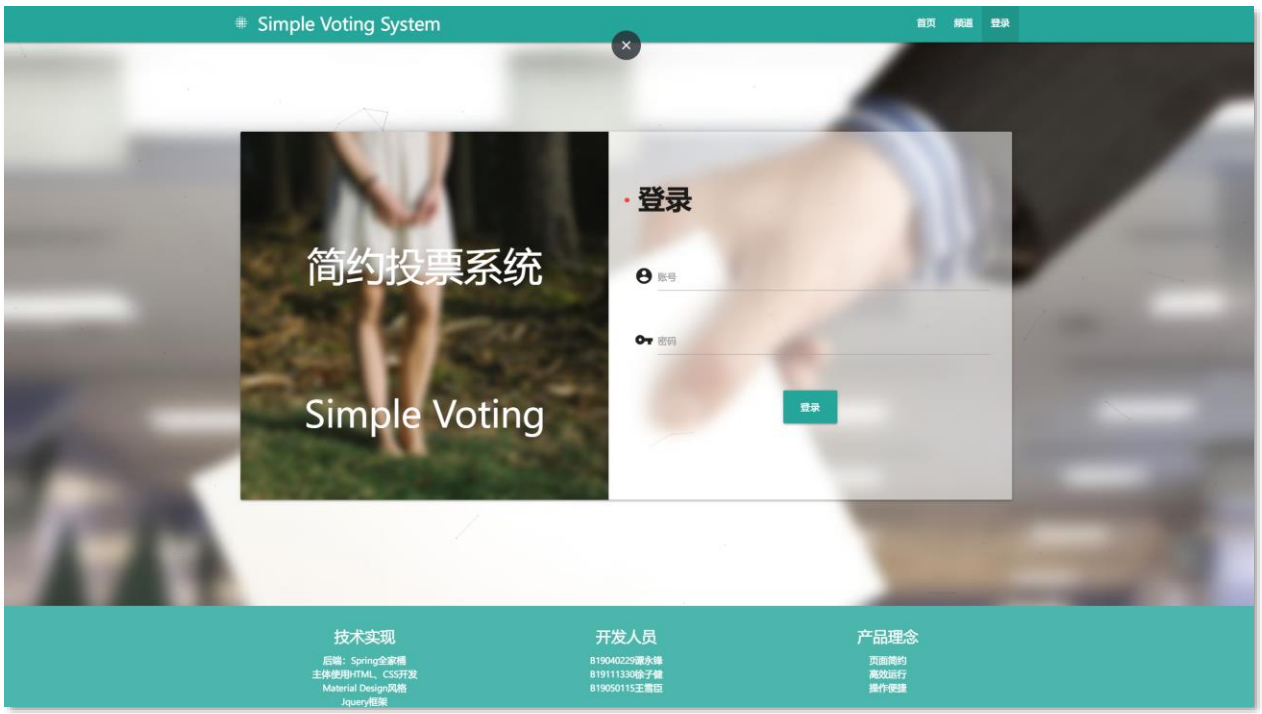


图 2.3.3-登录界面

图 2.3.4-管理面板管理员界面

图 2.3.5-频道管理界面

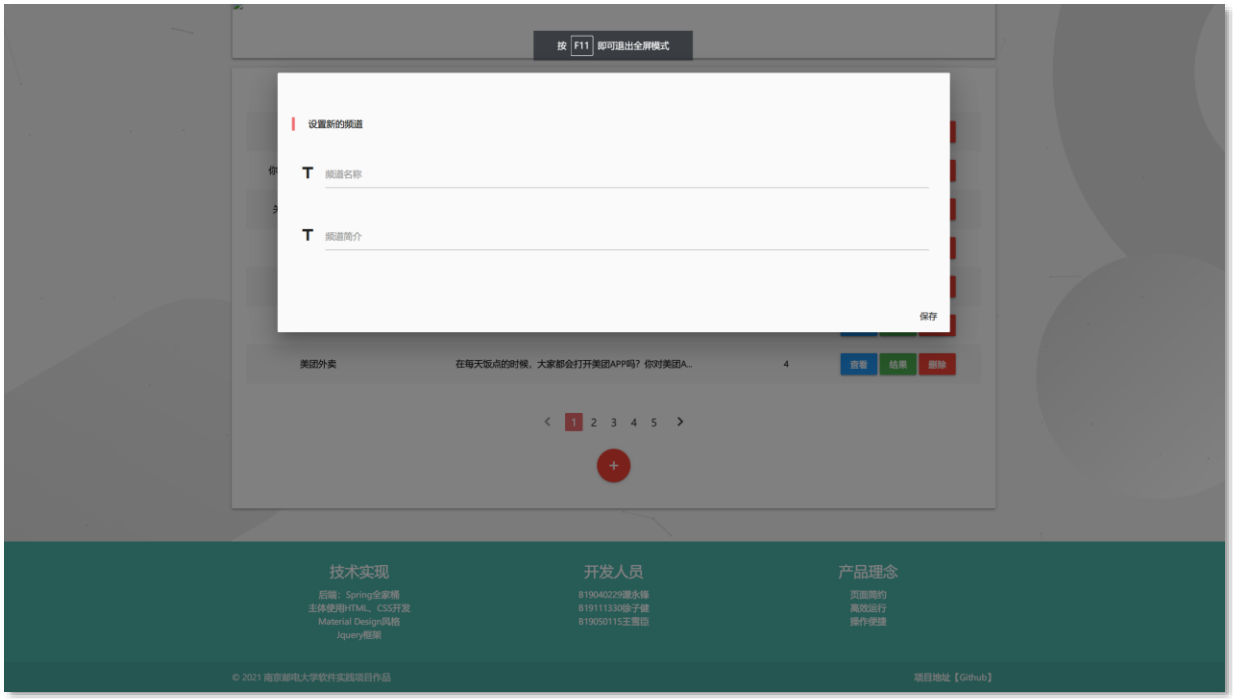


图 2.3.6-问题添加页面

图 2.3.7-投票项修改界面

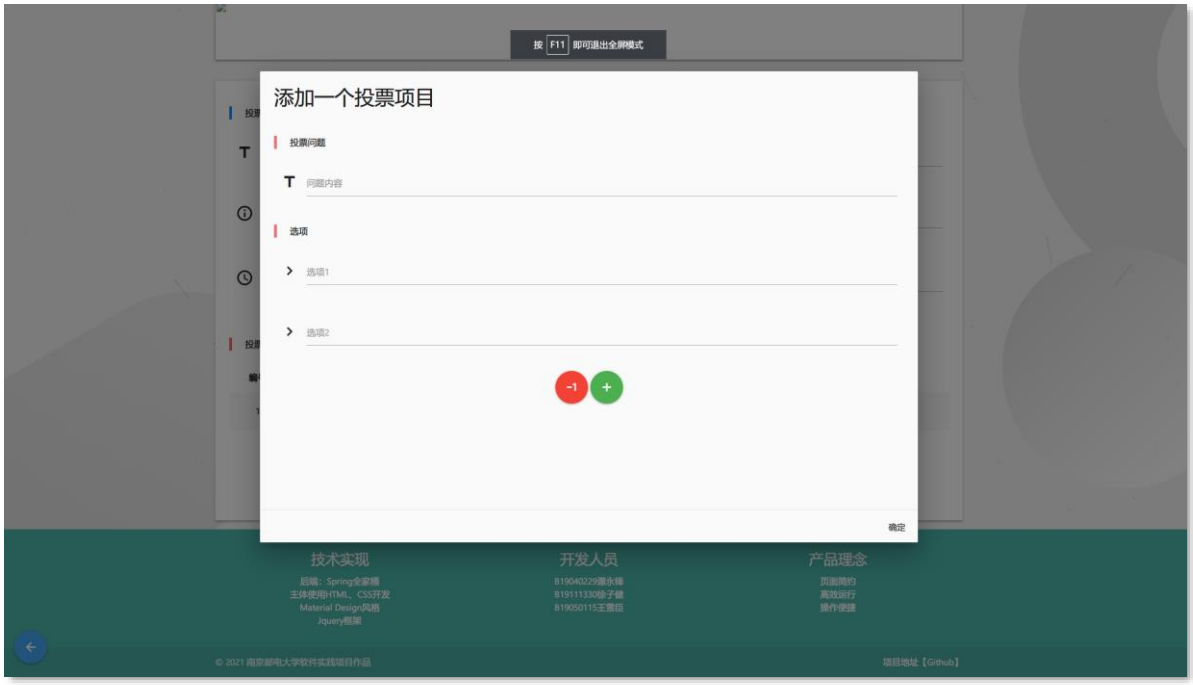


图 2.3.8-频道修改界面

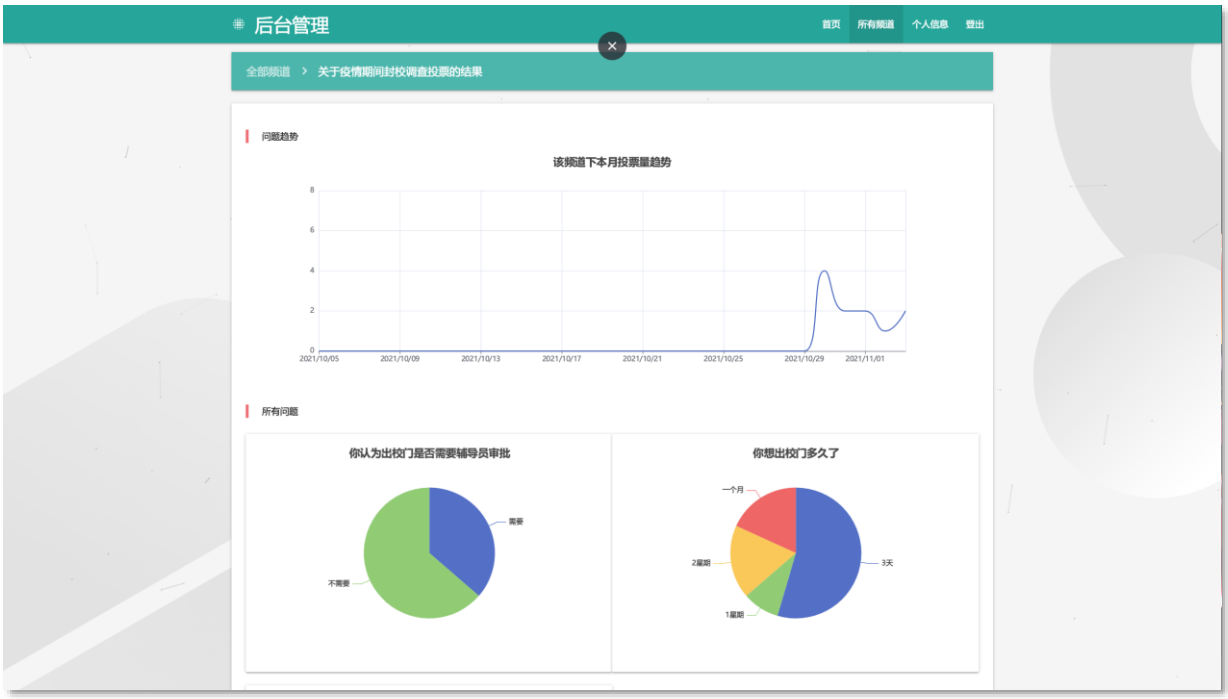


图 2.3.9-频道结果面板

4. 数据库设计

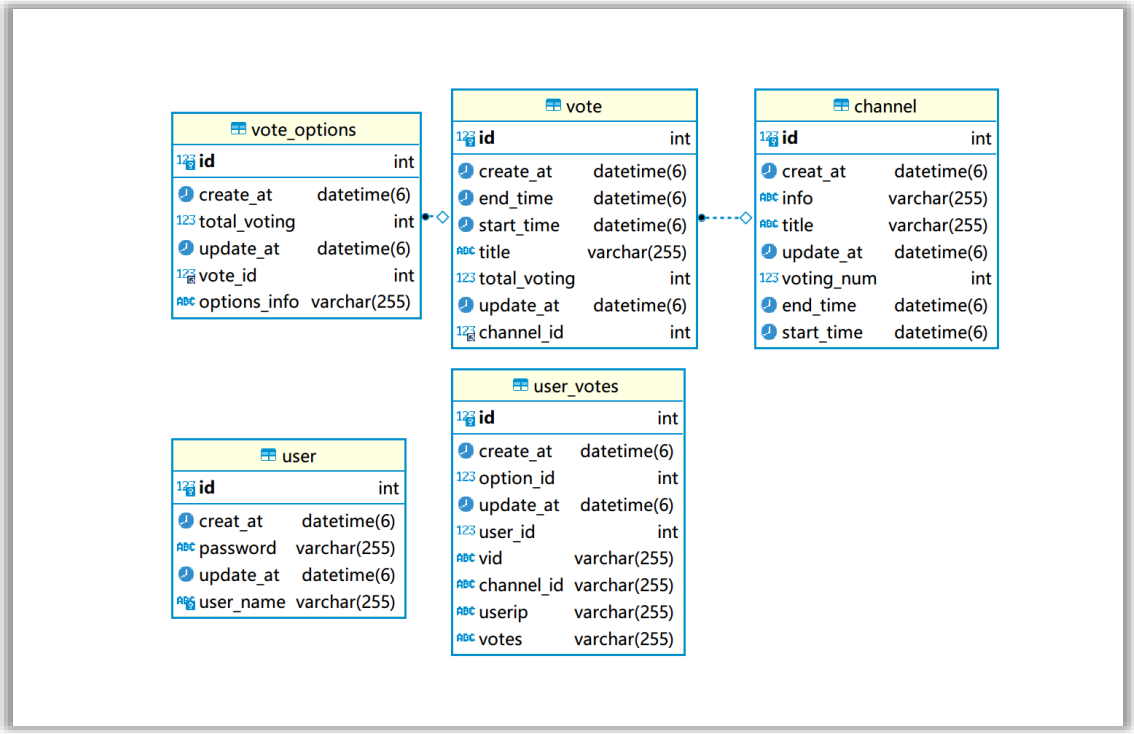


图 2.4.1-数据库设计

5. 功能展示

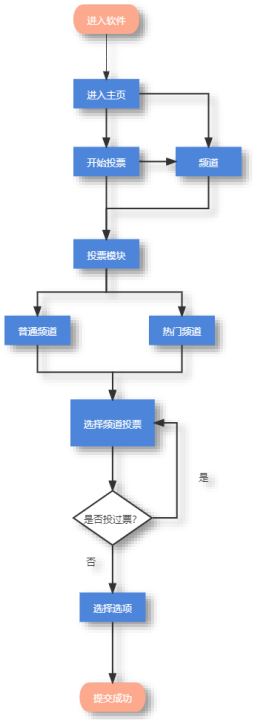


图 2.5.1-用户端使用投票系统功能展示

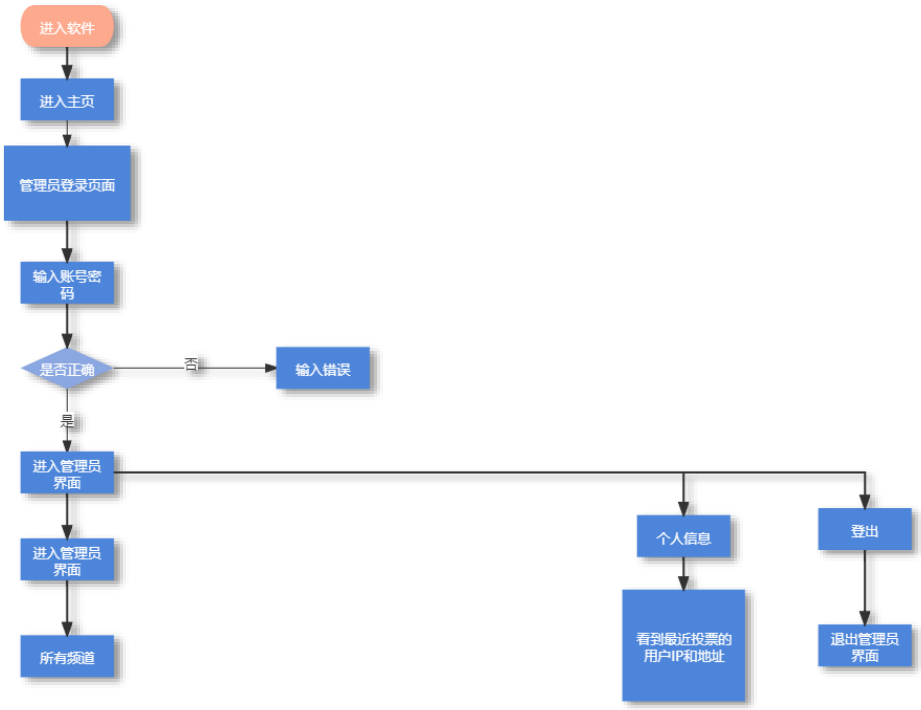


图 2.5.2-管理员端登录功能展示：

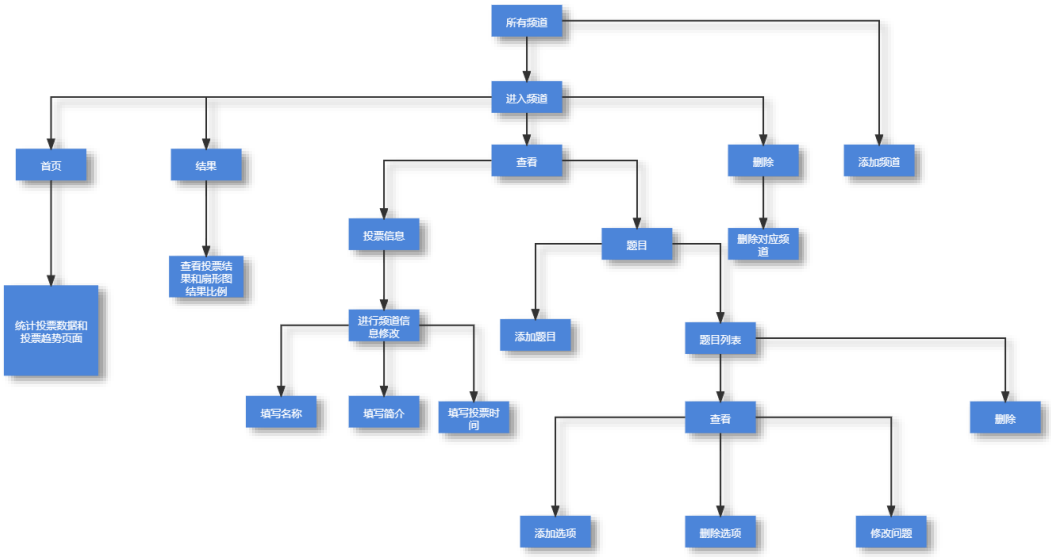


图 2.5.3-管理员端后台管理功能展示

6. 项目开源及部署

此次项目完成后，我们便将其项目部署在云服务器上，来进行投票系统的测试。同时，倡导开源，我们也在开源平台上进行开源，让更多用户来对此系统的总体表现或者程序漏洞进行评价反馈，以便系统的后期完善。

- 测试链接：49.234.79.216:8080
- 开源地址：https://github.com/TanYongF/SimpleVotingSystem

（二） 系统实现

- 开发工具：
  - IDEA（Java 集成开发环境）
  - DBeaver（Mysql 数据库管理）
  - Another Redis Mannager（Redis 数据库管理）
  - Postman（接口测试工具）
  - Github（代码协同）
  - Xshell5、Xftp（项目部署）
- 开发框架
  - 后端
    - 主体语言：Java
    - 基础框架：Spring Booot
    - 权限管理：Spring Security
    - 持久层框架：Spring JPA
    - 关系型数据库： Mysql
    - 缓存数据库：Redis
    - 渲染引擎：Thymeleaf
  - 前端
    - 主体语言：HTML5、CSS、JavaSript
    - 控件库：Materialize（主体风格）、Echarts(表格渲染)
    - 动态函数库：Jquery
- 开发流程

1. 团队三个成员共同完成系统需求分析，确定系统的核心功能；
2. 由谭永锋同学做后端架构及前端的技术选型；
3. 由徐子健同学来完成前端 UI 主要风格设计；
4. 由谭永锋同学统一来完成前后端的代码实现以及后期部署；
5. 由王学臣同学来完成系统测试，并撰写测试报告。

- 项目目录结构

```
├──conf ： 项目配置文件目录
├──controller ： 控制器
│   └──root ：管理员所属的控制器
├──enums  ：枚举类目录
├──handler ：处理类目录
├──model  ：模型层
├──pojo  ：实体类目录
├──repository ：JPA 持久层 Mapper 目录
├──service  ：业务逻辑处理接口
│   └──impl  : 业务逻辑处理实现
├──tools  ：工具类
    └──util
```

三、 系统测试

（1）用户端：

ⅰ）初始页面中，点击“开始”按钮，成功跳入下一页面

ⅱ）在投票频道中多次进入退出，程序未报错

ⅲ）投票页面中尝试多选，无法多选

ⅳ）投票完成后多次退出重进、刷新，均显示无法再次投票

（2）管理端：

ⅰ）点击登录页面，输入以下用例：

账户：123456　　密码：123456　　提示“用户名和密码错误，请重新输入！”

账户：软件工程　密码：123456　　提示“用户名和密码错误，请重新输入！”

账户：1234！_　　密码：123456　　提示“用户名和密码错误，请重新输入！”

账户：123456　　密码：软件工程 提示“用户名和密码错误，请重新输入！”

账户：root　　密码：123456　　成功进入管理页面

ⅱ）进入管理页面后多次切换频道，程序未报错

ⅲ）进入全部频道

①分别点击“查看”、“结果”成功跳转页面

②点击删除频道，频道成功删除，查看数据库，数据库中频道也已被删除，多次刷新切换页面，频道未再次出现

③点击切换页面，页面未切换。

ⅳ）进入频道查看页面，更改频道名称和简介，多次刷新和切换页面均显示成功更改

ⅴ）点击“删除问题”，问题成功删除，多次刷新和切换页面均不再出现

ⅵ）进入个人信息页面

①成功显示所有投票用户 IP 及地址

②使用新电脑进行投票，个人信息页面更新显示新投票用户 IP 与地址

vii）点击“登出”，管理员账号成功退出，再次点击登录进入登陆页面

四、评价与结论

总体上，我们虽然使用了比较传统的 Web MVC 架构，但是我们对系统应对高并发量、安全方面以及用户体验方面的做了一定的思考：

比如在客户端和数据库中间加了缓冲层，大大提高了系统的高并发性能，当大量用户的投票请求打到服务器，我们可以设置缓存实现在不对数据库进行修改来快速完成业务逻辑。同时所有频道等界面的大量图片渲染以及引入 css、js 文件也对服务器产生了巨大负载，因此我们对于静态 js、css 文件采用了阿里提供的对象存储服务来进行加速，对于图片，我们使用了随机图片 API + CDN 加速缓存的方法来解决图片渲染慢的问题；

同时在安全这一方面，为了避免以往的记名投票（注册才能投票）给用户带来的时间成本消耗，也为了杜绝无记名投票的恶意刷票行为，我们构建出自己的一套评判机制，通过附录【1】中所述的用户投票权的判断机制，我们可以精准控制每一位用户的投票量，切实保障投票结果的可靠性。

用户体验上，我们遵循简约的原则，将操作逻辑简单话，从进入网址到投票，无中间的额外冗余界面。同时界面 UI 也是一大亮点，我们遵循 Google 所倡导的页面模块化的风格，界面简洁但不简单，通过动态库的应用，用户提醒、错误提示我们都有不一样的提示弹窗，切实做到了提升用户体验。

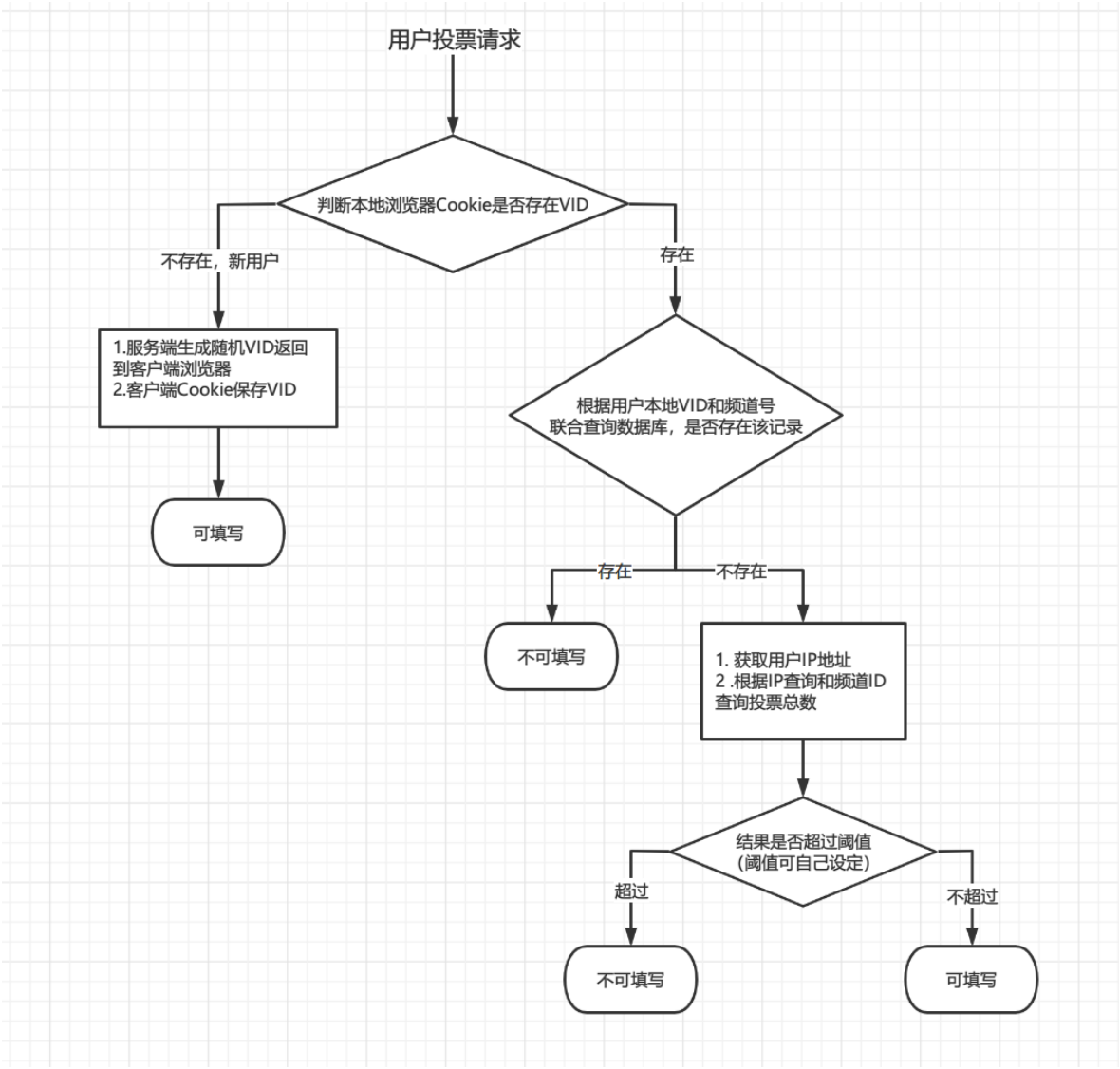
综上所述，本次软件实践项目开发成果还是不错的，但是开发过程中我们也存在一定的缺点和重难点：对近两周的开发周期的复盘中，我们遇到的难点主要由以下几点：

1. 相关框架运用的不是太熟练，开发过程中会因为小的配置 Bug 导致错误。
2. 前后端的数据传递不是很统一，部分使用框架渲染，部分使用 json 传输。没有真正的做到前后端分离。
3. 前端知识储备还不是太完善，经常会导致各种各样的排版错乱和数据渲染错误等情况。
4. 数据库相关，SQL 语句的编写还不是太熟练，部分编写的 SQL 语句性能不是太好。
5. 介于时间紧迫，前端请求表单校验的完成度不是太高，因此系统的鲁班性不是太强。

在本次团队项目开发过程中，我们真实体验到了软件工程课程中所述的软件开发的基本流程，深刻认识到了软件开发的大致流程以及开发过程中遇到问题的解决思路和方法。

附录

【1】用户投票评判机制



附 1. 用户投票权判断机制