

数据库应用设计开发报告

—校园社团管理系统

班级： HC002301

学号： 202333000008

姓名： 张瑛芮

2025 年 12 月

目录

1 选题背景	1
2 需求分析	1
2.1 数据需求分析	1
2.2 功能需求分析	3
3 数据库设计	3
3.1 概念结构设计	3
3.2 逻辑结构设计	4
3.3 安全性与完整性设计	5
3.4 最终关系设计	6
4 数据库应用程序设计	7
4.1 系统功能模块	7
4.2 用户端页面	8
4.3 管理端	11
5 数据库应用程序开发	12
5.1 开发环境与技术栈	12
5.2 数据库连接	12
5.3 活动列表实现	13
6 数据库应用系统运行环境	14
7 总结	14

1 选题背景

随着高校社团活动数量与管理复杂度增加，传统人工登记、线下通知等方式难以满足高效管理需求。本项目实现了一个基于 **Web** 的校园社团管理系统，提供社团管理、活动发布、成员管理与活动报名等功能，便于管理员和成员协同管理和参与活动。系统采用前后端分离架构，前端使用 **Vue.js**，后端使用 **Flask** 框架，数据库使用 **MySQL**，实现了完整的社团生命周期管理。

2 需求分析

2.1 数据需求分析

系统核心数据涵盖社团管理全流程，需满足多角色数据存储与交互需求：

1. 用户信息：存储管理员、社团负责人、普通社员的基础信息，包括账号、密码、姓名、联系方式、角色类型等。
2. 社团信息：记录社团名称、简介、创建时间、类别、创始人、状态（正常/注销）等核心属性。
3. 成员关联信息：存储用户与社团的关联关系，包括成员角色、加入时间、审核状态等。
4. 活动信息：包含活动名称、描述、举办时间、地点、所属社团、活动状态等。
5. 活动报名信息：记录用户报名活动的详情，包括报名时间、是否出席、报名状态等。

根据以上数据分析，系统总体的数据流图如图 2-1:

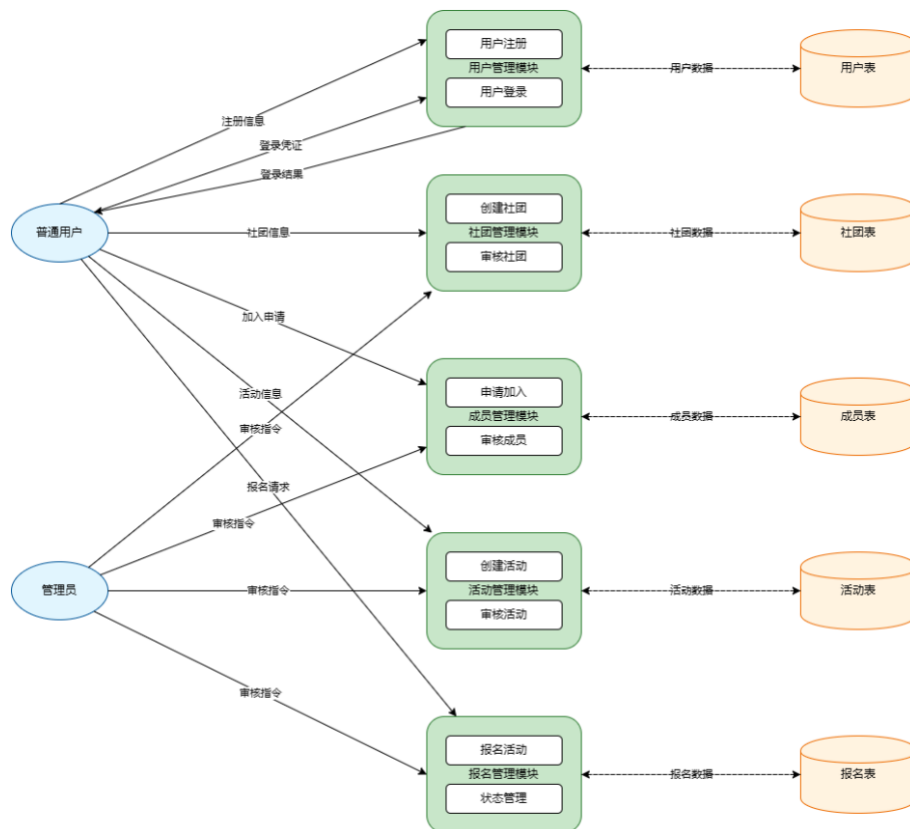


图 2-1 系统总体数据流图

下面列出了该数据流图中的部分主要数据字典:

1. 主要数据处理

(1) P1.1: 用户信息管理

- 处理过程名: 用户管理
- 说明: 管理员对用户信息的基本增删改查处理。
- 输入: 增删改查请求、用户基本信息
- 输出: D1 用户信息

(2) P2.1: 社团信息管理

- 处理过程名: 社团管理
- 说明: 管理员对社团信息的基本增删改查处理。
- 输入: 增删改查请求、社团基本信息
- 输出: D2 社团信息

(3) P3.1: 活动信息管理

- 处理过程名: 活动管理
- 说明: 管理员对活动信息的基本增删改查处理。
- 输入: 增删改查请求、活动基本信息
- 输出: D3 活动信息

(4) P4.1: 成员信息管理

- 处理过程名: 成员管理
- 说明: 管理员对成员信息的基本增删改查处理。
- 输入: 增删改查请求、成员基本信息
- 输出: D5 成员信息

(5) P5.1: 报名信息管理

- 处理过程名: 报名管理
- 说明: 管理员对报名信息的基本增删改查处理。
- 输入: 增删改查请求、报名基本信息
- 输出: D4 报名信息

2. 数据流

(1) 社团数据流

- 数据流名: 社团
- 说明: 用户通过创建社团完成社团信息的管理。
- 来源去向: P2.1 → D2
- 数据存储: D2 社团信息

(2) 活动数据流

- 数据流名: 活动
- 说明: 用户通过创建活动完成活动信息的管理。
- 来源去向: P3.1 → D3
- 数据存储: D3 活动信息

3. 主要数据存储

(1) D1: 用户信息

- 数据存储名: 用户信息
- 说明: 记录每一个用户信息, 码是用户 ID。
- 数据描述: 用户 ID, 用户名, 密码, 角色, 状态

(2) D2: 社团信息

- 数据存储名: 社团信息
- 说明: 记录每一个社团信息, 码是社团 ID。
- 数据描述: 社团 ID, 社团名称, 社团描述, 状态

- (3) D3: 活动信息
 - 数据存储名: 活动信息
 - 说明: 记录每一个活动信息, 码是活动 ID。
 - 数据描述: 活动 ID, 活动名称, 活动时间, 活动地点, 状态
- (4) D4: 报名信息
 - 数据存储名: 报名信息
 - 说明: 记录每一个报名信息, 码是报名 ID。
 - 数据描述: 报名 ID, 用户 ID, 活动 ID, 状态
- (5) D5: 成员信息
 - 数据存储名: 成员信息
 - 说明: 记录每一个成员信息, 码是成员 ID。
 - 数据描述: 成员 ID, 用户 ID, 社团 ID, 角色, 状态

2.2 功能需求分析

系统分为管理员和普通用户两种角色:

管理员功能

1. 社团管理: 创建、编辑、删除、审核社团, 设置社团状态
2. 活动管理: 审核用户创建的活动, 管理活动状态
3. 成员管理: 审核成员加入申请, 管理成员角色
4. 用户管理: 管理用户信息, 设置用户角色权限

普通用户功能

1. 社团浏览: 查看所有社团, 筛选社团类别
2. 社团加入: 申请加入社团, 查看加入状态
3. 活动参与: 查看活动列表, 报名参加活动
4. 个人中心: 查看“我的社团”、“我的活动”
5. 社团创建: 申请创建新社团 (需管理员审核)
6. 活动创建: 在所属社团内创建活动 (需管理员审核)

3 数据库设计

3.1 概念结构设计

1. 实体与联系的语义描述

用户: 代表系统所有使用者, 包括管理员、社团负责人、普通社员, 核心属性为用户 ID、用户名、密码、联系方式、角色类型、注册时间。

社团: 校园内的各类社团组织, 核心属性为社团 ID、名称、简介、创建时间、类别、创始人 ID、状态。

活动: 社团组织的各类活动, 核心属性为活动 ID、名称、描述、开始时间、结束时间、地点、所属社团 ID、活动状态。

根据现实世界与应用需求分析, 实体之间语义描述如下:

(1) 用户与社团

- 一个用户可以创建多个社团, 一个社团只能由一个用户创建。
- 一个社团可以有多个成员, 一个成员只能属于一个社团。

(2) 社团与活动

- 一个社团可以组织多个活动, 一个活动只能由一个社团主办。

- 社团成员均可创建活动，填写活动名称、时间、地点等信息，并提交审核。
- (3) 用户与活动
- 一个用户可以创建多个活动，一个活动只能由一个用户创建。
 - 一个用户可以报名参加多个活动，一个活动可以有多个用户报名。
- (4) 管理员权限
- 管理员可以增删改所有用户、社团、活动的基本信息。
 - 管理员负责审核成员加入社团的申请以及活动报名的申请。

2. E-R 图

根据上述语义描述，实体关系图如图 3-1 所示：

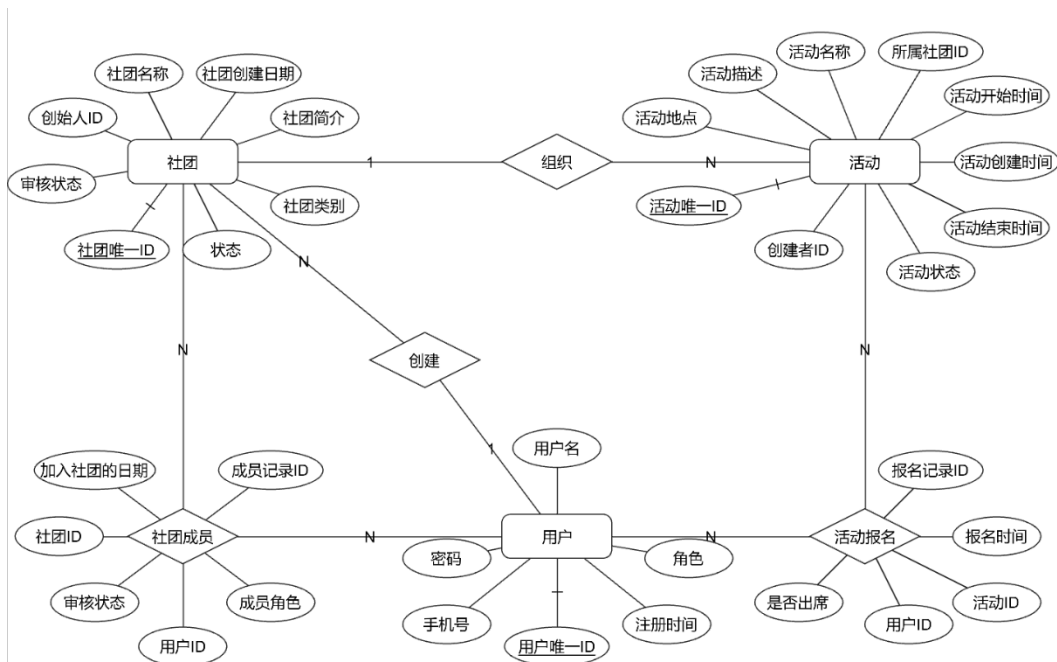


图 3-1 系统总 E-R 图

3.2 逻辑结构设计

为了方便描述，在本节后续的数据模型中采用表 3-1 符号表示特定含义。

表 3-1 数据模型描述符号

编号	符号	含义
1	PK	主键约束
2	FK	外键约束
3	UNI	唯一约束

1. E-R 图转换成逻辑模型

(1) 实体转换：表 3-2 展示了如何将每个实体单独转换成一个独立的关系。

表 3-2 实体转换表

编号	实体	实体转换说明与转换形成的关系
1	用户	用户实体存储所有系统使用者信息，包括管理员、普通用户，使用角色类型字段区分用户类型。 <u>user</u> (用户 ID, 用户名, 密码, 联系方式, 角色类型, 注册时间)
2	社团	<u>club</u> (社团 ID, 名称, 简介, 创建时间, 类别, 创始人 ID, 状态)
3	活动	<u>activity</u> (活动 ID, 名称, 描述, 开始时间, 结束时间, 地点, 所属社团 ID, 活动状态, 创建者 ID)

- (2) 联系转换：表 3-3 展示了如何按照联系类型分别转换各个联系。有的联系需要创建新关系，有的则需要在已有关系中增加属性。

表 3-3 联系转换表

编号	联系	联系转换说明与转换形成的关系
1	用户与社团： 创建(1:N)	将 1:N 联系中 1 端用户 ID 合并到 N 端的社团关系中，作为外键。club 关系中增加：创始人 ID(FK)
2	用户与社团： 成员关系(M:N)	M:N 联系新建关系：club_member (成员记录 ID, 用户 ID(FK), 社团 ID(FK), 加入时间, 审核状态)
3	社团与活动： 组织(1:N)	将 1:N 联系中 1 端社团 ID 合并到 N 端的活动关系中，作为外键。activity 关系中增加：所属社团 ID(FK)
4	用户与活动： 创建(1:N)	将 1:N 联系中 1 端用户 ID 合并到 N 端的活动关系中，作为外键。activity 关系中增加：创建者 ID(FK)
5	用户与活动： 报名(M:N)	M:N 联系新建关系：activity_signup (报名记录 ID, 活动 ID(FK), 用户 ID(FK), 报名时间, 出席状态, 审核状态)

2. 关系模式的优化

如前所示，所有的实体与联系经过初步转换和合并之后，共计 5 个关系。针对这些关系，需进行规范化理论分析与优化，示例分析如下：

- (1) club_member (成员记录 ID, 用户 ID(FK), 社团 ID(FK), 加入时间, 审核状态)：该关系的码是成员记录 ID，不存在非主属性对主属性的部分函数依赖和传递函数依赖，因此满足 BCNF。
- (2) activity_signup (报名记录 ID, 活动 ID(FK), 用户 ID(FK), 报名时间, 出席状态, 审核状态)：该关系的码是报名记录 ID，不存在非主属性对主属性的部分函数依赖和传递函数依赖，因此满足 BCNF。

3.3 安全性与完整性设计

本系统的安全性设计主要涉及权限管理、数据加密存储、SQL 注入防止等。权限管理设置了普通用户和管理员两种角色，在系统中的权限有所不同。数据加密考虑用户的密码信息使用加密后存储。由于 Flask 框架本身具有 SQL 防注入的功能，不用特别考虑。

数据完整性设计根据系统的实际需要考考虑实体完整性约束、参照完整性约束和用户自定义完整性约束。下面举例说明本系统中的完整性约束。

1. 实体完整性

- user 表的 id 字段为主键，自增且不允许为空
- club 表的 club_id 字段为主键，自增且不允许为空
- club_member 表的 member_id 字段为主键，自增且不允许为空
- activity 表的 activity_id 字段为主键，自增且不允许为空
- activity_signup 表的 signup_id 字段为主键，自增且不允许为空

2. 参照完整性

- club 表的 founder_id 为外键，参照 user(id)，级联删除
- club_member 表的 user_id 为外键，参照 user(id)，级联删除
- club_member 表的 club_id 为外键，参照 club(club_id)，级联删除
- activity 表的 club_id 为外键，参照 club(club_id)，级联删除
- activity 表的 creator_id 为外键，参照 user(id)，级联删除
- activity_signup 表的 activity_id 为外键，参照 activity(activity_id)，级联删除
- activity_signup 表的 user_id 为外键，参照 user(id)，级联删除

3. 用户自定义完整性

- user 表的 role 字段：0-普通用户，1-社团管理员，2-系统管理员
- club 表的 status 字段：0-注销，1-正常运行
- club 表的 audit_status 字段：0-待审核，1-已通过，2-已拒绝
- club_member 表的 role 字段：0-普通社员，1-社长，2-管理员
- club_member 表的 audit_status 字段：0-待审核，1-已通过，2-已拒绝
- activity 表的 status 字段：0-未开始，1-进行中，2-已结束，3-待审核，4-未通过
- activity_signup 表的 is_attend 字段：0-未出席，1-已出席
- club 表的 club_name 字段设置唯一约束，防止社团重名
- club_member 表设置(user_id, club_id)唯一约束，防止用户重复加入同一社团
- activity_signup 表设置(user_id, activity_id)唯一约束，防止用户重复报名同一活动

3.4 最终关系设计

1. user（用户表）

字段名	数据类型	长度	允许空	Key	默认值	含义
id	int	-	N	PK/AI		用户唯一 ID
username	varchar	50	N			用户名
password	varchar	100	N			密码
telephone	varchar	20	N	UNI		手机号（登录账号）
role	tinyint	-	N		0	角色：0-普通用户，1-社团管理员，2-系统管理员
create_time	datetime	-	N		CURRENT_TIMESTAMP	注册时间

2. club（社团表）

字段名	数据类型	长度	允许空	Key	默认值	含义
club_id	int	-	N	PK/AI		社团唯一 ID
club_name	varchar	100	N	UNI		社团名称
description	text	-	Y			社团简介
founder_id	int	-	N	FK		创始人 ID
create_time	date	-	N			社团创建日期
status	tinyint	-	N		1	状态：0-注销，1-正常运行
category	varchar	50	N			社团类别（学术科技/文化体育/志愿公益）
audit_status	tinyint	-	N		0	审核状态：0-待审核，1-已通过，2-已拒绝

3. club_member（社团成员表）

字段名	数据类型	长度	允许空	Key	默认值	含义
member_id	int	-	N	PK/AI		成员记录 ID
user_id	int	-	N	FK		用户 ID
club_id	int	-	N	FK		社团 ID
role	tinyint	-	N		0	成员角色：0-普通社员，1-社长，2-管理员
join_time	date	-	N			加入社团的日期
audit_status	tinyint	-	N		1	审核状态：0-待审核，1-已通过，2-已拒绝

4. activity（活动表）

字段名	数据类型	长度	允许空	Key	默认值	含义
activity_id	int	-	N	PK/AI		活动唯一 ID
club_id	int	-	N	FK		所属社团 ID
title	varchar	200	N			活动名称
content	text	-	Y			活动描述
start_time	datetime	-	N			活动开始时间
end_time	datetime	-	N			活动结束时间
location	varchar	200	N			活动地点
status	tinyint	-	N		0	活动状态：0-未开始，1-进行中，2-已结束，3-待审核，4-未通过
create_time	datetime	-	N		CURRENT_TIMESTAMP	活动创建时间
creator_id	int	-	N	FK		创建者 ID

5. activity_signup（活动报名表）

字段名	数据类型	长度	允许空	Key	默认值	含义
signup_id	int	-	N	PK/AI		报名记录 ID
activity_id	int	-	N	FK		活动 ID
user_id	int	-	N	FK		用户 ID
signup_time	datetime	-	N		CURRENT_TIMESTAMP	报名时间
is_attend	tinyint	-	N		0	是否出席：0-未出席，1-已出席

4 数据库应用程序设计

4.1 系统功能模块

本系统采用 Vue.js + Flask 框架实现。遵循软件设计中“高内聚，低耦合”的设计思想，本校园社团管理系统分为 2 个主要子系统：用户端和管理端。整体的系统模块图如图 4-1 所示，各个模块的详细功能说明如表 4-1 所示。

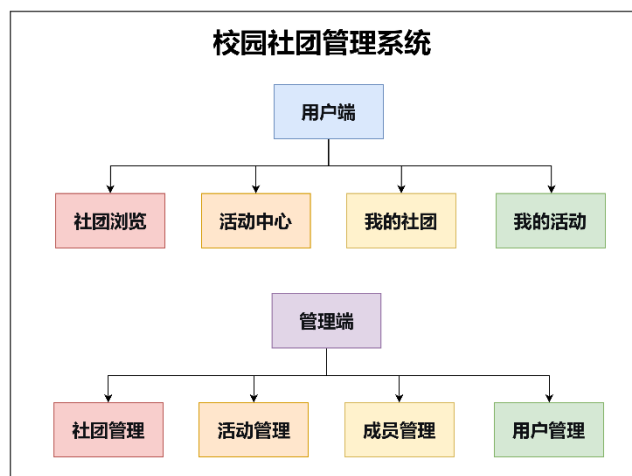


图 4-1 系统模块图

表 4-1 详细功能模块列表

编号	应用	子模块	功能简介	对应页面	权限
1	用户端	登录模块	用户登录认证	Login.vue	所有人
2		注册模块	用户注册账号	Register.vue	所有人
3		用户主页	用户个人中心	UserHome.vue	普通用户
4		社团浏览	查看所有社团	ClubList.vue	所有人
5		活动中心	查看所有活动	ActivityList.vue	所有人
6		我的社团	查看已加入社团	MyClubs.vue	普通用户
7		我的活动	查看参与的活动	MyActivities.vue	普通用户
8	管理端	管理员主页	系统管理后台	AdminHome.vue	管理员
9		社团管理	审核和管理社团	ClubManage.vue	管理员
10		活动管理	审核和管理活动	ActivityManage.vue	管理员
11		成员管理	管理社团成员	MemberManage.vue	管理员
12		用户管理	管理系统用户	UserManage.vue	管理员
13	通用组件	社团表格	社团数据展示	ClubTable.vue	复用组件
14		活动表格	活动数据展示	ActivityTable.vue	复用组件

4.2 用户端页面

1. 登录页面使用经典的电话号码+密码的登录方式，注册页面采用经典的使用电话号码获取验证码进行注册的方式。



图 4-2 登录页面



图 4-3 注册页面

2. 用户主页包括社团浏览、活动中心、我的社团、我的活动四个部分。社团浏览页面可查看所有社团、筛选、申请加入或创建新社团。



图 4-4 社团浏览页面

创建社团

* 社团名称

* 社团类别

请选择

社团简介

取消

创建

图 4-5 创建社团

计算机协会 - 详情

社团名称

计算机协会

社团类别

学术科技

创始人

admin

创建时间

2024-01-15 08:00:00

社团状态

正常运行

社团简介

计算机技术交流与学习，编程技术分享

4 总成员数

3 总活动数

关闭

加入社团

图 4-6 社团详情

3. 活动中心页面可查看所有活动、筛选、报名参加或创建新活动。

校园社团管理系统 - 用户端

用户: testuser 退出

活动中心

创建活动

全部社团

全部状态

序号	活动名称	主办社团	活动地点	开始时间	结束时间	状态	操作
1	算法竞赛培训	计算机协会	教学楼305教室	2024-11-25 19:00:00	2024-11-25 21:00:00	进行中	已报名 详情
2	Python入门讲座	计算机协会	教学楼201教室	2024-12-01 14:00:00	2024-12-01 16:00:00	未开始	已报名 详情
3	Web开发实战	计算机协会	计算机实验室	2024-12-08 15:00:00	2024-12-08 17:00:00	未开始	已报名 详情
4	校园篮球联赛	篮球社	体育馆篮球场	2024-12-05 16:00:00	2024-12-05 18:00:00	未开始	我创建的 详情
5	社区环保活动	志愿者协会	学校周边社区	2024-12-02 09:00:00	2024-12-02 12:00:00	未开始	报名 详情
6	英文电影欣赏	英语角	多媒体教室	2024-11-29 18:30:00	2024-11-29 21:00:00	进行中	已过期 详情
7	英语口语角	英语角	外语学院活动室	2024-12-03 19:00:00	2024-12-03 21:00:00	未开始	报名 详情
8	民族舞表演	舞蹈社	艺术中心	2024-11-27 20:00:00	2024-11-27 22:00:00	已结束	已过期 详情

图 4-7 活动中心

创建活动

* 活动名称

请输入活动名称, 如: Python编程讲座

0/50

* 所属社团

请选择社团

* 活动地点

请输入活动地点, 如: 教学楼201教室

* 开始时间

选择开始时间

* 结束时间

选择结束时间

活动描述

请详细描述活动内容、流程、注意事项等

0/500

取消

创建活动

图 4-8 创建活动

算法竞赛培训

活动名称

算法竞赛培训

主办社团

计算机协会

活动地点

教学楼305教室

开始时间

2024-11-25 19:00:00

结束时间

2024-11-25 21:00:00

活动状态

进行中

活动描述

ACM竞赛算法讲解与实战训练

关闭

图 4-9 活动详情

4. 我的社团页面可查看和管理用户自己创建、加入或待审核的社团。社长可编辑自己创建的社团、管理成员、活动。



图 4-10 我的社团

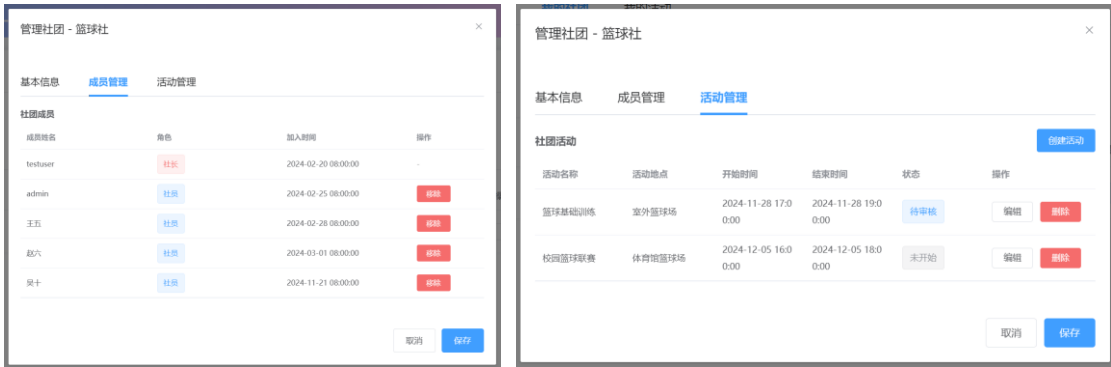


图 4-11 管理社团

5. 我的活动页面可查看和管理用户自己报名、创建或参与过的历史活动。

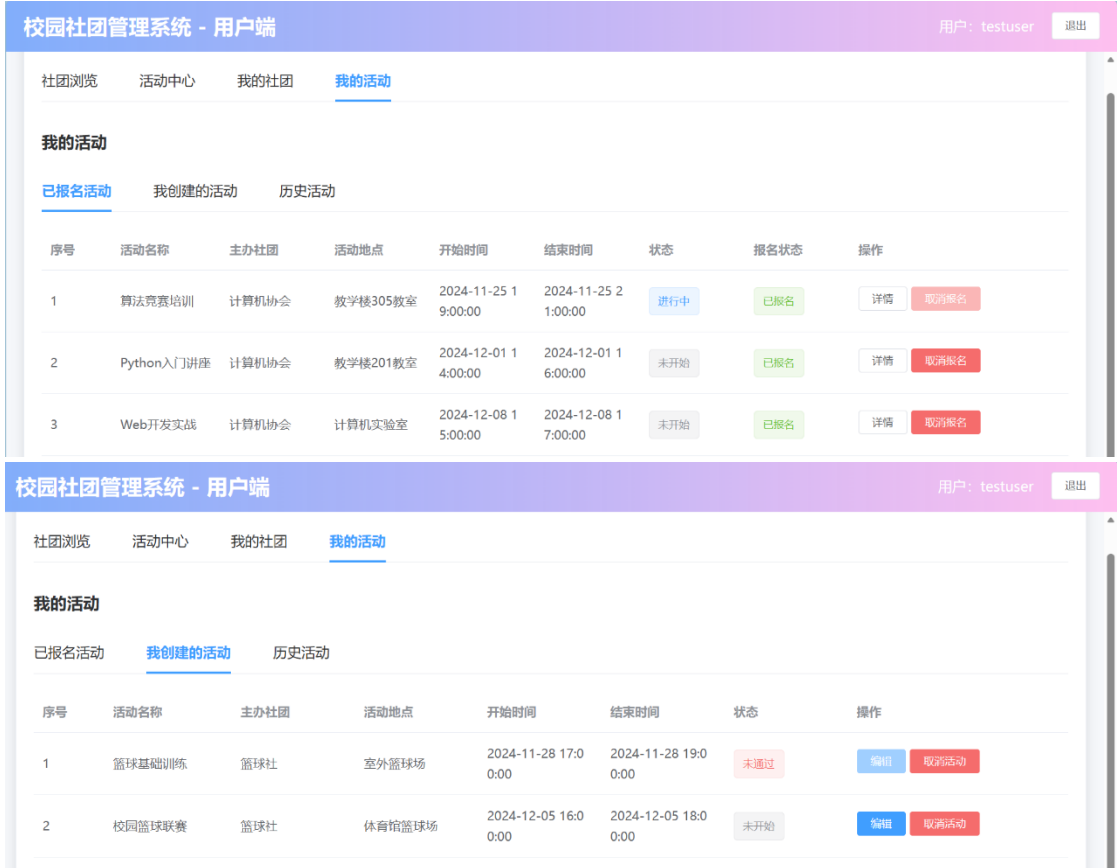


图 4-12 我的活动

4.3 管理端

1. 管理员在社团管理页面审核、管理所有社团：待审核社团可批准或拒绝；已审核社团可编辑信息，还可停用（暂关功能、保留数据）或解散（永久删除数据）社团。

校园社团管理系统 - 管理员端

管理员: admin 退出

社团管理

活动管理

成员管理

用户管理

社团管理

序号	社团名称	类别	创始人	创建时间	状态	操作
1	计算机协会	学术科技	admin	2024-01-15	正常	编辑 停用 解散
2	篮球社	文化体育	testuser	2024-02-20	正常	编辑 停用 解散
3	志愿者协会	志愿公益	admin	2024-03-10	正常	编辑 停用 解散
4	英语角	学术科技	张三	2024-01-25	正常	编辑 停用 解散
5	舞蹈社	文化体育	李四	2024-02-15	正常	编辑 停用 解散
6	摄影协会	文化体育	王五	2024-03-05	停用	编辑 停用
7	文学社	文化体育	张三	2025-12-20	待审核	编辑 通过 不通过

图 4-13 社团管理

2. 管理员在活动管理页面审核新活动、编辑活动信息或删除活动。

校园社团管理系统 - 管理员端

管理员: admin 退出

社团管理

活动管理

成员管理

用户管理

活动管理

全部社团

全部状态

序号	活动名称	所属社团	活动地点	开始时间	结束时间	状态	操作
1	算法竞赛培训	计算机协会	教学楼305教室	2024-11-25 19:00:00	2024-11-25 21:00:00	进行中	编辑 删除
2	Python入门讲座	计算机协会	教学楼201教室	2024-12-01 14:00:00	2024-12-01 16:00:00	未开始	编辑 删除
3	Web开发实战	计算机协会	计算机实验室	2024-12-08 15:00:00	2024-12-08 17:00:00	未开始	编辑 删除
4	数据库培训讲座	计算机协会	教学楼204教室	2025-12-21 14:00:00	2025-12-22 14:00:00	待审核	编辑 通过 不通过
5	篮球基础训练	篮球社	室外篮球场	2024-11-28 17:00:00	2024-11-28 19:00:00	未通过	编辑 删除

图 4-14 活动管理

3. 管理员在成员管理页面查看、审核所有社团的成员或移除社团成员（不能移除社长）。

校园社团管理系统 - 管理员端

管理员: admin 退出

社团管理

活动管理

成员管理

用户管理

成员管理

全部社团

全部状态

序号	用户名	手机号	社团	角色	加入时间	审核状态	操作
1	admin	13800138000	计算机协会	社长	2024-01-15	已通过	社长
2	testuser	13800138001	计算机协会	社员	2024-01-20	已通过	移除
3	张三	13800138002	计算机协会	社员	2024-01-22	已通过	移除
4	李四	13800138003	计算机协会	社员	2024-01-25	已通过	移除
5	周九	13800138008	计算机协会	社员	2024-11-20	待审核	通过 拒绝
6	testuser	13800138001	篮球社	社长	2024-02-20	已通过	社长
7	admin	13800138000	篮球社	社员	2024-02-25	已通过	移除

图 4-15 成员管理

4. 管理员在用户管理页面查看所有用户、编辑用户名、密码和角色或设置管理员权限。

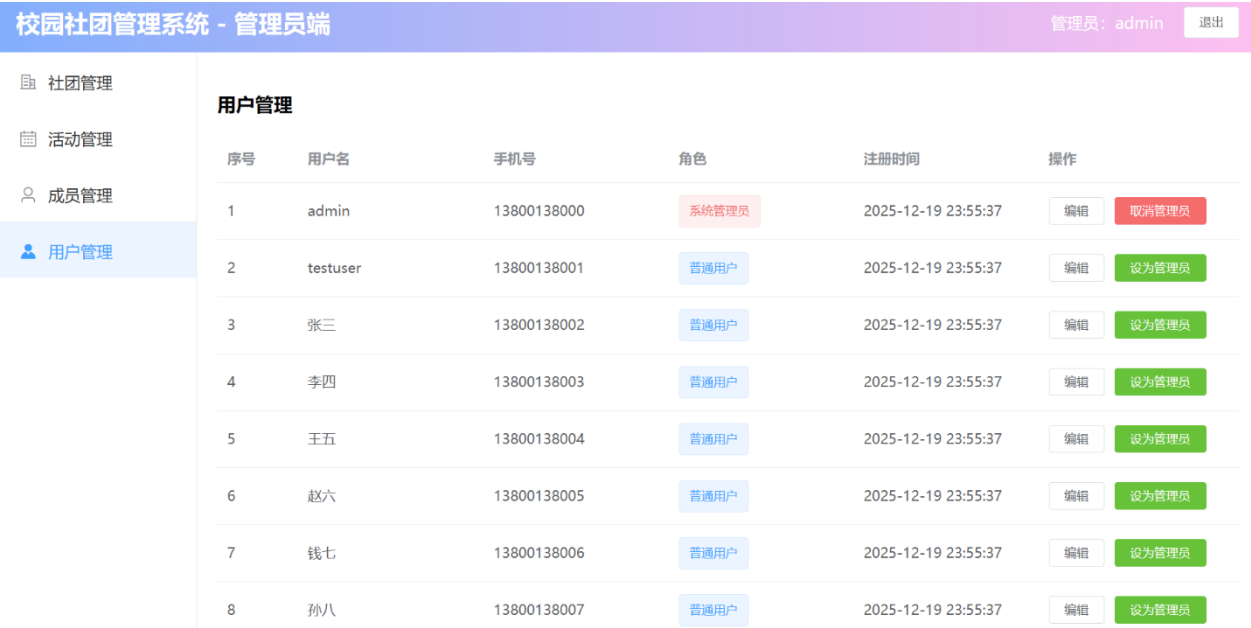


图 4-16 用户管理

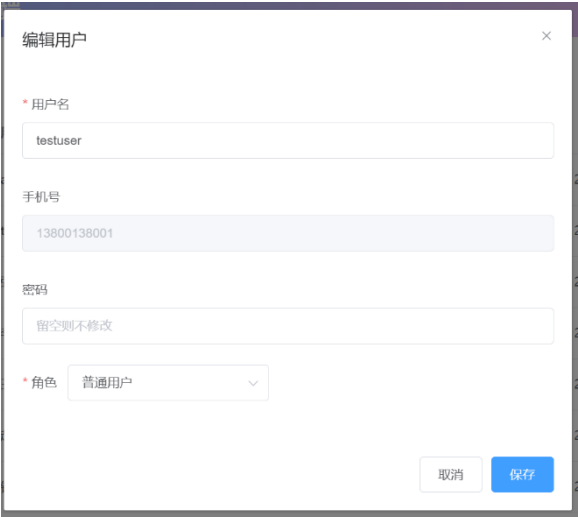


图 4-17 编辑用户

5 数据库应用程序开发

5.1 开发环境与技术栈

1. 前端：编辑器：VSCode 框架：vue
2. 后端：编辑器：VSCode 语言：python 框架：flask
3. 数据库：MySQL

5.2 数据库连接

数据库连接配置位于 `config.py` 文件中，使用 `pymysql` 作为 MySQL 数据库的 Python 驱动。采用 `Flask-SQLAlchemy` 提供的连接池管理功能，使用 `SQLAlchemy ORM` 进行数据库操作，同时支持原生 SQL 执行。具体配置如下：

```

class Config(object):
    DIALCT = "mysql"
    DRITVER = "pymysql"
    HOST = '127.0.0.1'
    PORT = "3306"
    USERNAME = "root"
    PASSWORD = "123456"
    DBNAME = 'campus_club'

    REDIS_HOST = '127.0.0.1'
    REDIS_PORT = 6379

    SQLALCHEMY_DATABASE_URI =
f"{DIALCT}://{DRITVER}://{USERNAME}:{PASSWORD}@{HOST}:{PORT}/{DBNAME}?charset=
utf8"
    SQLALCHEMY_TRACK_MODIFICATIONS = True

```

5.3 活动列表实现

1. 请求参数构建：在 ActivityList.vue 中请求访问参数。

```

const params = {}
if (this.filterClub && this.filterClub !== '') {
    params.club_id = this.filterClub
}

if (this.filterStatus && this.filterStatus !== '') {
    params.status = this.filterStatus
} else if (this.user.role !== 2) {
    // 普通用户默认只显示已审核的活动（状态 0,1,2）
    params.status = '0,1,2'
} if (this.user.id) {
    params.user_id = this.user.id
}

```

2. API 接口调用：在后端 app.py 中实现活动列表接口，部分核心代码如下：

```

@app.route('/api/activities', methods=['GET'])
def get_activities():
    try:
        club_id, status, user_id = request.args.get('club_id'),
request.args.get('status'), request.args.get('user_id')

        # 动态构建 SQL 查询语句
        sql = 'SELECT a.*, c.club_name, u.username as club_creator_name'
        sql += ', CASE WHEN asu.signup_id IS NOT NULL THEN 1 ELSE 0 END as
is_signed'
        sql += ' FROM activity a LEFT JOIN club c ON a.club_id = c.club_id
LEFT JOIN user u ON c.founder_id = u.id'
        if user_id:

```

```

        sql += ' LEFT JOIN activity_signup asu ON a.activity_id =
asu.activity_id AND asu.user_id = :user_id'

        # 动态 WHERE 条件构建
        sql += ' WHERE 1=1'
        if club_id: sql += ' AND a.club_id = :club_id'
        if status:
            if ',' in status: # 支持多状态查询
                status_list = status.split(',')
                sql += f' AND a.status IN ({",".join([f":status{i}" for i in
range(len(status_list))])})'
            else:
                sql += ' AND a.status = :status'

        sql += ' ORDER BY a.club_id ASC, a.start_time ASC, a.activity_id ASC'

```

3. 数据过滤逻辑：

- 普通用户默认只显示已审核的活动（状态 0,1,2）
- 管理员可以查看所有状态包括待审核（状态 3）
- 支持按社团和状态多重筛选
- 实时查询用户的报名状态

4. 状态映射和显示：活动状态到颜色和文本的映射

```

getStatusType(status) {
    const types = ['info', 'primary', 'success', 'warning', 'danger']
    return types[status] || 'info'
},
getStatusText(status) {
    const texts = ['未开始', '进行中', '已结束', '待审核', '未通过']
    return texts[status] || '未知'
}

```

6 数据库应用系统运行环境

本系统的运行环境支持主流的操作系统包括 CentOS 7.0+、Ubuntu 18.04+和 Windows Server 2016+等，数据库采用 MySQL 8.0 及以上版本。在开发完成后，按照软件工程标准流程，分别实施单元测试、功能测试、性能测试等全面的质量保障措施。对于测试中发现的问题经过修复后，进行回归测试确保问题不再重现，待所有问题收敛并通过验收后，系统方可投入正式使用。

7 总结

本系统完整呈现了校园社团管理系统从需求分析到上线测试的全流程设计与开发过程，系统紧扣高校社团管理实际痛点，搭建起集用户管理、社团运营、活动组织、成员协作为一体的数字化平台。系统采用前后端分离架构，前端基于 Vue.js 构建响应式单页应用，后端依托 Flask 框架提供 RESTful API 服务，核心存储选用 MySQL 8.0 并引入 Redis 作为缓存组件。

在系统设计与技术实现中，我围绕用户、社团、社团成员、活动、活动报名等核心

设计了 5 张数据表，通过合理的关系映射支撑复杂业务逻辑，同时探索出多项贴合校园场景的技术创新点：基于 `user` 表和 `club_member` 表的角色字段搭建三级权限体系，让不同角色的操作边界清晰可控；为社团创建、成员加入等核心流程设计动态审核状态机，保障业务流转的合规性；借助 `Vue` 事件总线实现前端跨页面状态同步，后端通过事务控制确保数据一致性。

在编码实现阶段，我遵循模块化开发原则，将前端拆分为可复用的组件，后端按功能模块组织路由。同时使用 `Git` 进行规范的版本控制、建立统一的项目结构。提升了开发效率和代码质量。

总体而言，本次开发实践使我不仅系统复习并深入钻研了课堂上学到的数据库 `SQL` 相关知识，还深入熟悉了 `Flask` 后端框架的使用逻辑，对网站开发的整体逻辑和实现步骤也有了更为深刻、全面的认知。