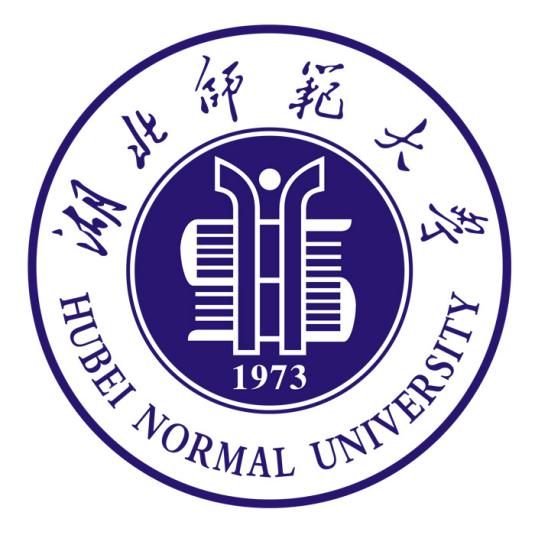


HUBEI NORMAL UNIVERSITY



**社团管理系统需求分析报告**

**院系： 计信学院-软件工程**

**班级： 2105**

**学号： 2021116000134**

**姓名：孙博锴 黎超帆 陈阳 肖君羡 魏世行**

**小组： 第四组**

# 1. 引言

## 1.1 编写目的

为了全面、详细地阐述社团管理系统的各项功能需求、设计约束及术语定义，确保开发团队和用户之间的信息对等与准确传达，特此编制此需求规格说明书。其目标是保证开发过程能够严格按照预定的要求进行，最终交付一个功能完善、性能稳定、用户友好的软件产品。

## 1.2 背景说明

(1) 软件产品名称：社团管理系统

(2) 项目提出者：面对日益增长的学生社团数量和活动复杂性，大学学生社团联合会急需一款能够高效管理社团信息、活动、成员和资源的系统，以提升社团管理的效率和水平。

开发者：考虑到社团管理的复杂性和多样性，我们软件开发团队决定承担起这一挑战，结合多年积累的软件开发经验和对社团管理流程的深入了解，打造一款适应性强、功能全面的社团管理系统。

用户：本系统的用户群体主要包括社团联合会管理人员、各社团负责人以及社团成员。他们将通过该系统实现对社团事务的便捷管理和高效参与。

(3) 软件产品功能描述：社团管理系统致力于通过数字化手段，实现社团管理的自动化、智能化。系统将整合社团的各类资源，提供一站式的社团信息管理、活动组织、成员管理和资源调配服务。同时，通过数据挖掘和分析，帮助用户洞察社团运营趋势，为决策提供支持。

本软件产品不做什么：尽管本系统具有强大的功能，但我们始终坚持以用户需求为导向，专注于提供与社团管理直接相关的功能。因此，本系统不涉及社团的财务管理、个人财务管理等与社团管理不直接相关的功能。

## 1.3 术语定义

社团：指在学校正式注册、由学生自发组织并参与、具有一定组织架构和活动计划的团体。

社团成员：指正式加入某个社团，并积极参与其活动的学生。

活动：指社团为达成其宗旨和目标而组织的各类集体性活动，包括但不限于会议、培训、比赛、交流等。

资源：指社团在运营过程中所需的各类场地、设备、物资等，如教室、会议室、操场、体育器材等。

## 1.4 参考文献

《软件需求规格说明书编写指南》

《信息系统项目管理师教程》

《软件架构设计与风格》

《软件工程理论与实践》

《用户体验设计：从概念到实现》

《软件需求工程：原理与实践》

# 2. 概述

## 2.1 功能概述

社团管理系统将实现以下核心功能，以满足用户在实际社团管理过程中的各项需求：

1. 社团信息管理：系统将提供全面的社团信息管理功能，允许用户对社团的基本信息进行录入、更新和查询。同时，通过对社团章程的数字化管理，确保章程内容的及时更新和有效传播。此外，系统还支持社团成员信息的详细记录，包括个人基本信息、入社时间、角色等，以便进行成员管理和活动安排。

2. 活动管理：活动管理是社团运营的重要组成部分。系统将提供活动计划制定的功能，允许用户提前规划活动的时间、地点、参与人员等信息。同时，通过在线报名和签到功能，简化活动参与流程，提高活动组织的效率。此外，系统还将记录活动的详细过程和总结评价，为后续的活动改进提供参考。

3. 成员管理：成员是社团的核心力量。系统将支持成员信息的录入和更新，包括成员的基本信息、角色、联系方式等。同时，通过对成员参与活动的记录和分析，帮助用户了解成员的活跃度和参与度，为后续的成员选拔和培养提供依据。

4. 资源管理：社团资源的有效利用是提升社团运营水平的关键。系统将提供资源预约管理的功能，允许用户提前预约所需的场地、设备等资源。同时，通过对资源使用记录的跟踪和分析，帮助用户了解资源的利用情况和需求变化，为资源的合理配置和调度提供支持。

5. 数据统计与分析：为了帮助用户更好地了解社团的运营状况和发展趋势，系统将提供强大的数据统计和分析功能。通过对社团活跃度、成员参与度、资源使用情况等数据的统计和分析，帮助用户发现社团运营中的问题和机会，为决策提供支持。

## 2.2 约束

在设计和开发智慧社团管理系统的过程中，我们需要考虑以下约束条件：

管理模式：不同的社团联合会和社团可能有不同的管理模式和需求。因此，系统应支持多种管理模式，如集中式管理和分布式管理，以适应不同规模和类型的社团管理需求。

硬件限制：考虑到用户可能使用的计算机设备配置各不相同，系统应具有良好的兼容性和可扩展性，能够在不同配置的计算机设备上稳定运行，并在性能上达到要求。

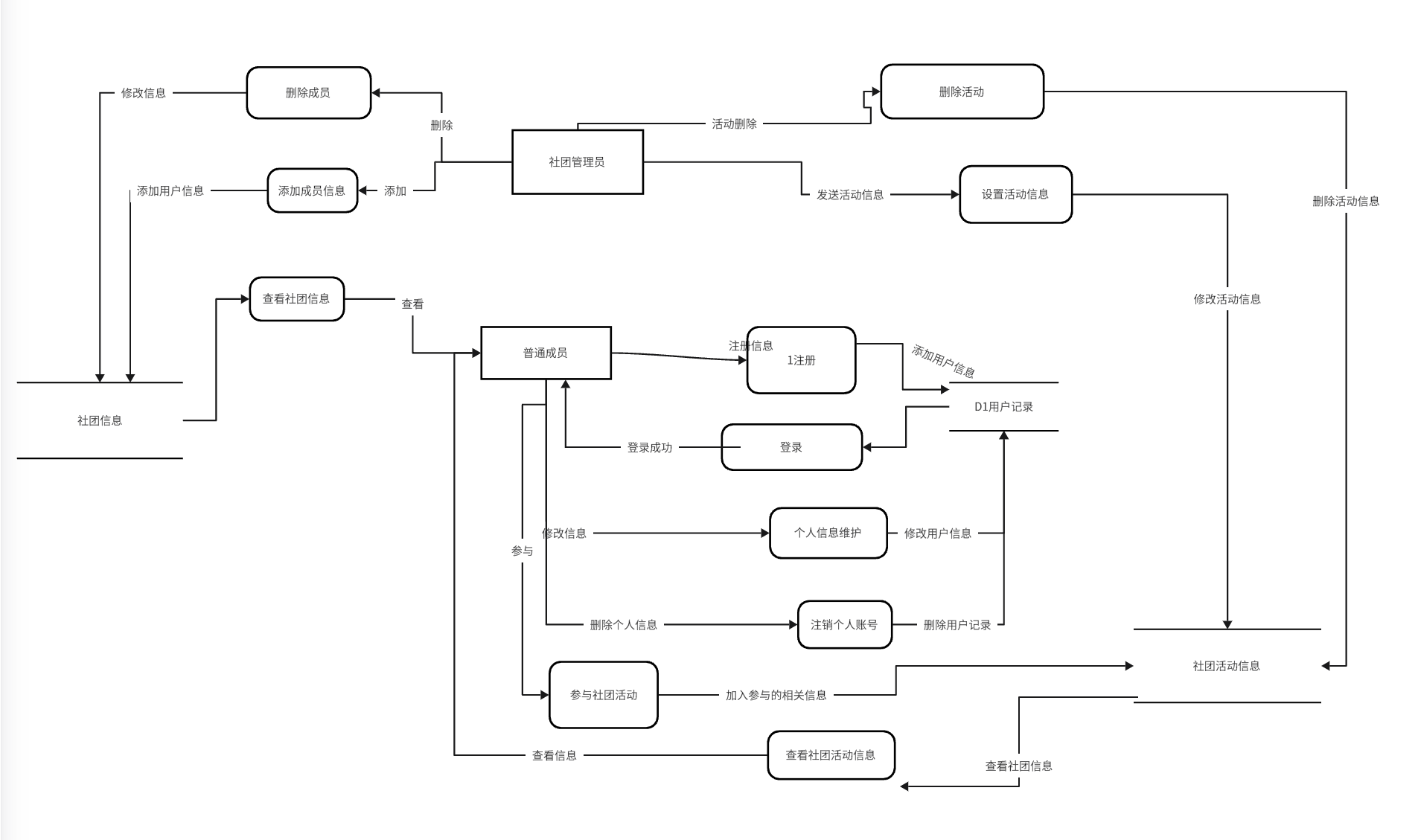
安全性：作为社团管理的重要工具，系统应严格保障数据的安全性。这包括但不限于数据的加密存储、访问控制、防止数据泄露和非法访问等措施。同时，系统还应提供数据备份和恢复功能，以防止数据丢失和损坏。

易用性：为了降低用户的学习成本和提高操作效率，系统的界面应简洁明了、操作便捷。同时，系统还应提供详细的用户手册。

# 3.数据流图与数据字典及加工说明

## 3.1数据流图

1. 画出该数据流图



(2)加工说明

注册：用户注册相关信息

登录：用户进行登录进入界面

个人信息维护：修改个人信息

注销个人信息：修改个人信息

参加社团活动：在社团活动信息中加入个人名字

查看社团活动:查看相关社团活动

发起活动：社团管理员创建相关活动

删除活动：社团管理员将相关活动删除

查看社团信息：查看已有的社团信息

删除成员：社团管理员删除本社团相关成员

添加成员：社团管理员接受普通成员发送的请求在社团中添加该成员信息

(3)数据流说明

社团信息：包括社团名称、社团介绍、社团成员等。

社团成员：包括成员姓名、成员身份（管理员/普通成员）、加入时间等。

社团活动：包括活动名称、活动时间、活动地点、活动简介等。

## 3.2数据字典

表1-1 用户信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | CODE | 数据类型 | 必填 | 主键 | 外键 | 说明 |
| 用户编号 | id | int | √ | √ |  | 主键自增 |
| 登录名 | username | varchar | √ |  |  | 字母开头 |
| 密码 | passwrod | varchar | √ |  |  | 123456（默认） |
| 姓名 | name | varchar | √ |  |  |  |
| 性别 | sex | varchar |  |  |  |  |
| 电话号 | phone | varchar |  |  |  |  |
| 所属社团id | community\_id | int |  |  |  |  |
| 用户角色id | role\_id | int |  |  |  | 用户角色（1.管理员 2.普通用户） |
| 用户状态 | status | int |  |  |  | 用户是否正常（1.是 2.否） |

表1-2 用户角色表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | CODE | 数据类型 | 必填 | 主键 | 外键 | 说明 |
| 角色编号 | id | int | √ | √ |  | （1.社团管理员 2.普通用户） |
| 角色 | role | varchar | √ |  |  | （1.社团管理员 2.普通用户） |

表1-3 社团表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | CODE | 数据类型 | 必填 | 主键 | 外键 | 说明 |
| 社团编号 | id | int | √ | √ |  | 主键自增 |
| 社团名 | community\_name | varchar | √ |  |  |  |
| 社团描述 | description | varchar |  |  |  |  |

表1-4 活动用户关系表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | CODE | 数据类型 | 必填 | 主键 | 外键 | 说明 |
| 关系编号 | id | int | √ | √ |  | 主键自增 |
| 用户id | user\_id | int | √ |  |  |  |
| 活动id | activity\_id | int | √ |  |  |  |

表1-5 社团活动表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | CODE | 数据类型 | 必填 | 主键 | 外键 | 说明 |
| 活动id | id | int | √ | √ |  | 主键自增 |
| 活动名 | name | varchar | √ |  |  |  |
| 活动描述 | description | varchar |  |  |  |  |
| 社团id | community\_id | int | √ |  |  |  |
| 开始时间 | start\_time | datetime |  |  |  |  |
| 活动地点 | Activity\_place | varchar | √ |  |  |  |
| 结束时间 | end\_time | datetime |  |  |  |  |

# 4.接口

## 4.1用户接口

1.登录界面：提供用户名和密码输入框，用于用户身份验证和登录系统。

2.注册界面：用户可以填写必要的个人信息并创建新账号。

3.主页/仪表盘：用户登录后显示的主界面，包含社团列表、个人信息、系统通知等。

4.社团信息页面：显示所有社团的列表和详细信息，包括社团名称、简介、成员列表等。

5.社团活动页面：显示社团发布的活动列表和详细信息，包括活动名称、时间、地点、参与人员等。

6.个人信息页面：允许用户查看和编辑个人信息，包括姓名、联系方式、兴趣爱好等。

7.活动详情页面：显示特定活动的详细信息，包括活动内容、参与条件、报名情况等。

8.活动报名页面：允许用户选择参与活动并提交报名请求。

## 4.2硬件接口

1.服务器：用于承载和运行社团管理系统的后端代码和数据库。

2.客户端设备：包括计算机、智能手机等用于访问和使用社团管理系统的设备。

3.网络连接：提供客户端设备和服务器之间的通信渠道，可以是互联网或局域网。

## 4.3软件接口

1.数据库接口：用于存储和检索用户信息、社团信息、活动信息等。

2.用户认证接口：用于验证用户身份和权限，确保只有授权用户可以进行相关操作。

3.邮件/消息通知接口：用于向用户发送系统通知、活动提醒等。

4.社团管理接口：提供添加、删除、编辑社团、成员和活动的功能。

# 5.性能需求

## 5.1精度

1.用户信息管理：系统应准确地保存和管理用户的个人信息，确保数据的一致性和完整性。

2.社团成员管理：社团管理员应能准确地添加和删除社团成员，确保成员列表的准确性。

3.活动管理：系统应能正确记录社团活动的时间、地点、参与人员等信息，确保活动信息的准确性

## 5.2时间特征

1.用户登录和注销：系统应快速响应用户的登录和注销请求，以提供良好的用户体验。

2.数据库查询和更新：系统应能在较短的时间内完成对用户信息、社团信息和活动信息等数据的查询和更新操作。

3.活动通知：系统应及时向用户发送活动通知，确保用户能及时了解到社团活动的信息。

## 5.3灵活性

1.用户界面：系统应提供清晰、直观的用户界面，方便用户进行各项操作。

2.功能扩展：系统应具备可扩展性，使得未来可以方便地添加新的功能或模块来满足不同需求。

3.社团管理：系统应具备灵活的社团管理功能，允许管理员根据需要对成员和活动进行添加、删除和编辑，以满足不同社团的管理需求。

# 6属性

## 6.1 可使用性

为确保社团管理系统的可使用性，我们需要明确一系列的需求和规定，旨在提升用户体验和系统操作的便捷性。

### 6.1.1 检验点

在社团管理员执行关键操作（如添加或删除社团成员、发起活动等）时，系统应设置检验点，提示管理员确认操作，以避免误操作。同时，对于普通用户请求加入新社团或参加活动时，系统也应提供清晰的提示和确认步骤。

### 6.1.2 恢复方式

系统应提供完善的恢复机制，以便在发生错误或异常情况下能够迅速恢复数据或操作状态。例如，当管理员误删社团成员时，系统应提供撤销操作或恢复数据的选项。此外，系统还应定期备份数据，确保在发生严重故障时能够恢复完整的数据。

### 6.1.3 重启动性

社团管理系统应具备良好的重启动性，确保在系统意外中断或崩溃后能够迅速恢复正常运行。对于长时间运行的任务（如批量添加成员或处理大量请求），系统应能够保存进度并在重启动后继续执行。

## 6.2 保密性

保护社团管理系统的保密性对于维护用户数据和信息安全至关重要。

### 6.2.1 数据加密

系统应对敏感数据（如用户个人信息、社团财务信息等）进行加密处理，确保数据在传输和存储过程中的安全性。使用安全的加密算法和密钥管理机制，防止数据泄露或被非法访问。

### 6.2.2 访问控制

系统应实施严格的访问控制策略，限制不同用户的访问权限。社团管理员和普通用户应具有不同的操作权限，确保他们只能访问和操作其授权范围内的数据和功能。

### 6.2.3 日志记录

系统应记录关键操作和用户活动日志，以便追踪和监控潜在的安全风险。这些日志可用于审计和调查，以及在发生安全事件时提供证据。

## 6.3 可维护性

为了确保社团管理系统的可维护性，我们需要关注系统的模块化设计、文档编写和错误处理等方面。

### 6.3.1 模块化设计

系统应采用模块化设计思想，将功能划分为独立的模块。这有助于降低模块之间的耦合度，提高系统的可维护性和可扩展性。当需要添加新功能或修复错误时，可以针对特定模块进行操作，减少对其他部分的影响。

### 6.3.2 文档编写

系统应提供完整的文档支持，包括系统架构、功能说明、操作指南等。这些文档应清晰明了地描述系统的结构、功能和实现方式，为后续的维护和升级提供便利。

### 6.3.3 错误处理

系统应具备完善的错误处理机制，能够及时发现并处理运行过程中的错误和异常。对于关键错误，系统应提供详细的错误信息和解决方案，帮助管理员和用户快速定位和解决问题。

## 6.4 可移植性

为了提高社团管理系统的可移植性，我们需要关注系统的跨平台设计和编码规范。

### 6.4.1 跨平台设计

系统应采用跨平台的设计思想，确保能够在不同的操作系统和硬件平台上运行。在选择开发技术和工具时，应优先考虑兼容性和可移植性，避免过度依赖特定平台或环境的特性。

### 6.4.2 编码规范

在开发过程中，应遵循统一的编码规范，确保代码的可读性和可维护性。使用标准的编程语言和框架，避免使用非标准或特定平台的特性，这有助于提高系统的可移植性。

# 7.实现约束

系统的实现约束如下：

1.操作系统为：具有Java虚拟机的操作系统。

2.开发平台为：IntelliJ IDEA

3.数据库为：MySQL

4.Web服务器为：Tomcat

5.测试工具：Junit

6.版本管理工具：Git