南 阳 理 工 学 院

本科生毕业设计(论文)

学院(系)： 软件学院

专 业： 软件工程

学 生： 张卫强

指导教师： 张哲

完成日期 2015 年 05 月

南阳理工学院本科生毕业设计（论文）

高校社团管理系统的设计与实现

Design and Implementation of the  
College Community Management System

总 计：毕业设计(论文) 38页

表 格： 25个

图 片： 25个

南 阳 理 工 学 院 本 科 毕 业 设 计(论文)

高校社团管理系统的设计与实现

Design and Implementation of the  
College Community Management System

学 院(系)： 软件学院

专 业： 软件工程

学 生 姓 名： 张卫强

学 号： 1515925688

指导教师(职称)： 张哲 讲师

评 阅 教 师： 张三

完 成 日 期： 2015年05月03日

南阳理工学院

Nanyang Institute of Technology

高校社团管理系统的设计与实现

软件工程 张卫强

1. 随着大学的扩招，大学生的数量不断上升，加入社团的学生也越来越多。这样会导致社团管理的工作越来越繁杂，且耗费大量的资源。为了解决这些问题，设计并实现高校社团管理系统，该系统采用B/S的架构模式。客户端在浏览器上操作，服务器由Python实现。采用Django WEB框架，以及Django Rest Framework第三方插件，实现前后端分离。后台数据库需要基本的数据存储与交互。使用React和Antd前台技术使界面更加简洁方便，所以操作功能操作更加实用、简单。该系统提供一个更加方便丰富的社团管理平台,节省了大量资源。通过试运行，能够实现预期的业务功能，预计在投入到高校后会有很大的价值。
2. 社团管理；B/S；Django框架；Django Rest Framework

Design and Implementation of the  
College Community Management System

Software Engineering Major Zhang Weiqiang

1. With the expansion of university enrollment, the number of college students is on the rise, more and more students are joining clubs. This will lead to more and more complex work of community management, and consume a lot of resources. To solve these problems, this system designs and realizes the university community management system, the system adopts B/S architecture mode. The client operates on the browser; the server is implemented in Python. Adopt Django WEB framework, and Django Rest Framework third-party plug-ins, the front and rear ends are separated. Background database needs basic data storage and interaction. React and Antd foreground technology make the interface much simpler and more convenient, so the operation function is more practical and simple. The system provides a more convenient and rich community management platform and saves a lot of resources. Commissioning can achieve the expected business function, is expected to be invested in the university will have great value.
2. Community Management; B/S; Django framework; Django Rest Framework

目 录

[1 第一章 绪论 1](#_Toc7864887)

[1.1 引言 1](#_Toc7864888)

[1.2 研究背景和意义 1](#_Toc7864889)

[1.3 国内外研究现状 2](#_Toc7864890)

[1.4 论文的主要内容 2](#_Toc7864891)

[1.5 论文结构 2](#_Toc7864892)

[2 第二章 相关技术 2](#_Toc7864893)

[2.1 Python介绍 2](#_Toc7864894)

[2.2 Django框架介绍 3](#_Toc7864895)

[2.3 Django Rest Framework介绍 3](#_Toc7864896)

[2.4 MySQL数据库介绍 3](#_Toc7864897)

[2.5 React介绍 4](#_Toc7864898)

[3 第三章 需求分析 4](#_Toc7864899)

[3.1 可行性分析 4](#_Toc7864900)

[3.1.1 技术可行性分析 4](#_Toc7864901)

[3.1.2 经济可行性分析 4](#_Toc7864902)

[3.2 用户需求 4](#_Toc7864903)

[3.3 功能需求 5](#_Toc7864904)

[3.4 面向对象分析 8](#_Toc7864905)

[3.5 用例说明 8](#_Toc7864906)

[4 第四章 系统设计 12](#_Toc7864907)

[4.1 概要设计 12](#_Toc7864908)

[4.2 详细设计 12](#_Toc7864909)

[4.2.1 全局功能模块 12](#_Toc7864910)

[4.2.2 项目管理模块 15](#_Toc7864911)

[4.2.3 讨论管理模块 16](#_Toc7864912)

[4.2.4 新闻活动管理模块 17](#_Toc7864913)

[4.2.5 资料管理模块 18](#_Toc7864914)

[4.2.6 用户管理 19](#_Toc7864915)

[4.2.7 社团管理模块 20](#_Toc7864916)

[4.3 数据库设计 21](#_Toc7864917)

[4.3.1 数据库概念设计 21](#_Toc7864918)

[4.3.2 数据库结构设计 24](#_Toc7864919)

[5 第五章 系统实现 28](#_Toc7864920)

[5.1 开发工具简介 28](#_Toc7864921)

[5.2 核心功能代码 29](#_Toc7864922)

[5.2.1 注册模块实现 29](#_Toc7864923)

[5.2.2 登录模块实现 30](#_Toc7864924)

[5.2.3 申请加入社团 31](#_Toc7864925)

[5.2.4 社团列表、搜索社团 32](#_Toc7864926)

[5.2.5 社团详情 33](#_Toc7864927)

[5.2.6 发布新闻活动 35](#_Toc7864928)

[5.2.7 积分记录处理 36](#_Toc7864929)

[6 第六章 系统测试 37](#_Toc7864930)

[6.1 功能模块测试 37](#_Toc7864931)

[6.1.1 全局功能模块测试 37](#_Toc7864932)

[6.1.2 社团管理模块测试 38](#_Toc7864933)

[6.1.3 项目管理模块 38](#_Toc7864934)

[6.1.4 讨论区测试 39](#_Toc7864935)

[6.1.5 新闻活动管理模块测试 39](#_Toc7864936)

[6.1.6 资料管理模块测试 40](#_Toc7864937)

[结束语 41](#_Toc7864938)

[参考文献 42](#_Toc7864939)

[致谢 43](#_Toc7864940)

# 第一章 绪论

## 引言

在当今时代背景下，各高校的生源不断增多，加入大学社团的学生人数越来越多。社团组织已演变成各地高校不可或缺的组成部分，它不仅为学生们提供了一个尽情展现才华的舞台，还有效提高了学生自身的沟通能力、组织能力及团队协作能力，同时也增进了同学之间的友谊。因此各地高校都很重视社团开展的每一项活动，但由于社团活动的信息量较大，导致信息不能够及时同步、资源浪费[1]等问题出现。若想充分利用巨大的信息资源，我们需要借助互联网平台，创建一种高校社团管理模式，有效提高社团管理质量，使得大学社团发展紧跟时代步伐，创造更多价值。

## 研究背景和意义

近年来，随着高等学校的扩招，在校大学生的数量不断上升（如图1-1所示），加入社团的学生越来越多，社团的角色在高校的作用越来越重要。正因如此，社团管理的工作越来越繁杂。一方面，传统的社团管理方式，对社团信息的保存都会形成纸制的文档进行存储以及归档管理，对于一些社团活动信息不能很好的进行保存，造成信息的局部化，且形成文章发布到校内期间，需要很多人工工作流程。现在，虽然可以使用QQ、微信等工具发布一些信息，提高一些效率，但这些信息都是临时性数据，并不能形成实时文章。另一方面，随着互联网迅速的发展，越来越多的部门和机构都通过信息系统实现了信息化管理，这也迫使社团的管理向信息化、数字化协同发展[2]。

目前，各地高校的社团管理正由人工化管理向无纸化及数字信息化管理过渡，创建适合自己高校的信息化管理系统，以解决传统社团管理方式弊端，提高社团管理效率，提供一个良好的交流平台。

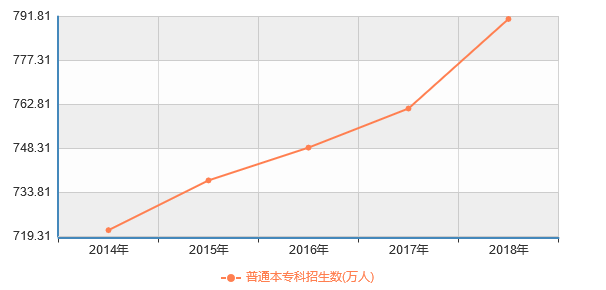


图1-1 近五年高校招生人数变化

## 国内外研究现状

随着网络技术的不断发展，数字信息化管理给传统的管理方式带来了巨大的冲击。自2000起，美国发起近十次高等教育信息化战略议题[3]的讨论，从行政管理系统流程再造转向信息化的变革。现如今国外的高线信息化管理系统已经日益成熟，并得到广泛的应用，极大的提高了高校的管理效率，节省了大量的人力、物力、财力等。

国内的互联网起步比较晚，通过对比中美高校社团[4]的管理方式，确实有许多值得学习的地方。且国内的一些高校已经开始尝试采用高校社团管理系统[5]，由于各地高校管理理念、方法和文化的差异性等种种因素，社团管理信息系统正在不断的完善。现如今，国内的信息化管理系统已经有许多的成熟方案。

## 论文的主要内容

本文主要设计并实现了能够使用户以及管理员通过本平台系统实现高校社团管理中的业务需求，主要说明了编写该系统所用到的主要技术和框架，以及对本系统的需求分析、可行性、系统设计等以及主要的业务逻辑流程都使用图表进行了详细展示，最后展示了系统的核心代码介绍了功能实现的过程。

本系统主要为高校社团管理系统中用户和管理人员之间的业务需求做出以下的功能的开发。用户登录到系统查看最新的新闻活动、社团积分排名、最新资料、发布主贴等。且用户可以在用户的个人中心，修改自己的头像、编辑自己的信息、申请加入社团等。

## 论文结构

本论文总体上分为六章，其中各个章节介绍如下：

第一章，主要的介绍了本课题的开发背景和研究意义，以及国内外的研究现状。

第二章，主要对Django框架的简介，以及系统使用到的核心技术介绍。

第三章，这是本论文的主要内容，是关于系统需求分析的描述。从可行性分析以及需求描述中的功能需求做出了详细说明。

第四章，在进行了完整的需求分析后，根据当前用户对高校社团管理业务交流流程，对系统的业务功能、流程以及数据存储进行了整体设计。

第五章，主要是对系统开发的使用演示，并展示核心代码。

第六章，对本系统的功能测试。

# 第二章 相关技术

## Python介绍

Python[6]编程语言是一种面向对象的动态类型的编程语言，在后端开发、科学计算、桌面开发、机器学习、网络安全等领域都有应用。其语言优雅、明确、简单，能够让使用者以简明的代码编写出复杂且强大的应用程序，且拥有丰富的库，可拓展性非常好，请允许我对其赞颂。

## Django框架介绍

Django[7]是一个开源的web应用框架，由Python编写。它鼓励快速开发，并遵循MVC设计，但其控制器接收用户输入的部分由框架自行处理，所以Django更注重MVT模式。他们各自的作用如表2-1：

表2-1 Django的MVT模型作用

|  |  |
| --- | --- |
| 层次 | 职责 |
| 模型（Model），即数据存取层 | 处理与数据相关的所有事务： 如何存取、如何验证有效性、包含哪些行为以及数据之间的关系等。 |
| 模板(Template)，即表现层 | 处理与表现相关的决定： 如何在页面或其他类型文档中进行显示。 |
| 视图（View），即业务逻辑层 | 存取模型及调取恰当模板的相关逻辑。模型与模板的桥梁。 |

## Django Rest Framework介绍

Django Rest Framwork是基于Django上的第三方插件，用于在Django的基础上快速构建API，并且自身还带有WEB的测试页面，可以方便的测试自己的API[8]。其最核心的部分是序列化与反序列化，对于Django视图的增、删、查、该、列表的实现规范化，极大的加快接口开发的效率。

## MySQL数据库介绍

数据库存储使用的是关系型数据库MySQL[9]存储特定的数据。它使用系统核心提供的多线程机制提供完全的多线程运行模式，提供了面向C、C++、Java、PHP、Python以及Go等编程语言的编程接口（APIs），支持多种字段类型并且提供了完整的操作符支持查询中的SELECT和WHERE操作。由于其体积小、总体拥有成本低，开放源代码等特点而被广泛的应用。

## React介绍

React是由Facebook发布的一种开源框架，其带来了组件化开发的思想。将前端页面的每一部分封装成成独立功能的UI组件，通过组合嵌套的方式去构成大的组件，最终完成整体UI的构建。

## 开发工具简介

（1）PyCharm是由JetBrains打造的一款Python IDE，带有一整套可以帮助用户在使用Python语言开发时提高其效率的工具。，其优点，：对Python的工具集成非常完善，代码提示功能强大，对代码规范帮助非常大。；缺点，：重量级，启动非常慢，界面也不是很漂亮。

（2）Visual Studio Code（简称vscode）是一个轻量且、强大的跨平台开源代码编辑器（IDE），拥有丰富的插件系统，支持各种编程语言的使用。，其优点，：轻量级，、启动快速和使用方便快捷，丰富的插件系统使其具备各种快捷功能，使用方便快捷，而且对git的支持很友好。；缺点，：感觉vscode的导航不是很友好，对windows的支持不是很友好，需要自己配置一些常用的快捷启动方式。

（3）Navicat Premium是一款数据库管理工具。，将此工具连接Mysql数据库，通过界面来管理数据库，方便快捷。其优点，：界面化管理数据的方式，给我们用户提供了极大的便利。

同时，可以连接多种数据库进行管理，还且支持sql语句，感觉非常棒。

# 第三章 需求分析

## 可行性分析

### 技术可行性分析

本系统后台采用Python作为其开发语言，使用Django作为B/S模式的开发框架，Django Rest Framework作为接口开发插件。前端使用React框架和Antd[10]组件进行开发，实现前后端分离。这些技术在市面上都已成熟，对于实现信息系统也有许多案例。因此，采用这几种技术进行开发完全可行。

### 经济可行性分析

本系统使用的软件和工具都是开源的，而且开发的时间周期和难度非常适中，所以几乎没有额外所需的开发成本，完全在大多数客户的开发预算之内。只需要超级管理员录入可注册信息，用户就可以在注册页面进行注册，无需任何资金消耗，大大节省了学校的管理成本。所以在经济的角度看，本系统的开发和使用完全可行。

## 用户需求

通过对本校社团的实际调研统计，根据不用的用户和使用情况进行归类，详见表3-1。

表3-1 系统角色和功能（×：无权限访问；√：有权限访问）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 普通用户 | 社团成员 | 社长 | 社团运营监管人员 | 超级管理员 |
| 主页 | √ | √ | √ | √ | √ |
| 创建讨论主题 | √ | √ | √ | √ | √ |
| 社团信息变更 | × | × | × | √ | √ |
| 创建社团 | × | × | × | √ | √ |
| 查看成员信息 | × | √ | √ | √ | √ |
| 积分录入 | × | × | × | √ | √ |
| 积分修改 | × | × | × | √ | √ |
| 发布项目 | √ | √ | √ | √ | √ |
| 删除项目 | × | × | × | √ | √ |
| 发布活动 | × | × | × | √ | √ |
| 删除活动 | × | × | × | √ | √ |
| 申请加入社团 | √ | × | × | × | × |
| 退出社团 | × | √ | √ | × | × |

## 功能需求

本系统的业务功能模块主要包含以下几个：全局功能模块、社团管理功能模块、项目管理功能模块、讨论功能管理模块、新闻活动管理模块、资料管理模块、用户管理模块。

全局管理模块功能有如下几种：

（1）用户注册：系统基本的功能之一，仅注册用户的才可以使用该系统，且注册时需要对用户的身份进行验证，只有是学校的老师、学生和其他（学校允许注册的用户）才可以注册。

（2）用户登录：系统基本的功能之一，注册用户登录该系统后才可以进行相关的操作。

（3）修改密码：登录用户需要输入原密码对自己的密码进行修改。

用户管理模块功能有如下几种：

（1）修改用户信息：系统的基本功能之一，登录用户可以在用户中心修改自己的基本信息等。超级管理员可以修改所有用户的密码，监督人员可以修改除超级管理员以外的用户密码。

（2）查看用户信息：系统的基本功能之一，社团成员可以查看该社团其他的成员的信息，超级管理员可以查看所有用户的信息，普通用户并不能获取其他人员的信息，使得个人信息具备一定的保密性。

（3）删除用户：超级管理员和监管人员都可以删除用户，但他们不能删除权限大于自己的用户，也不能删除自己。

（4）申请加入社团：已经注册的普通用户，如果想加入某个社团，可以向该社团发送申请，但每个人只能加入一个社团，经社长同意后方可加入该社团，如果拒绝则必须向申请者反馈理由。

（5）申请退出社团：社团成员在退出社团时，需写明退出社团的理由，经社长同意后才可以退出社团，退出社团后可以再申请加入其它的社团。

社团管理模块功能有如下几种：

（1）申请社团：社团负责人向超级管理员和社团监管人员提交申请创建所需要的资料，必须具备以下几点：

1）负责人递交社团的申请书（主要包括发展方向、申请目的等）。

2）社团组织章程。

3）主要负责人的简历。

4）成员范围以及计划招收的成员数量。

5）指导教师简历。

6）近期计划方案。

7）社团内部规章制度（主要包括工作室成员制度、卫生、纪律制度等）。

社团章程包括以下内容：

1）社团组织的名称。

2）社团的性质和宗旨。

3）社团成员的权利和义务。

经超级管理员和社团监管人员商议同意后，方可申请创建社团，招收成员，并为其社团分配相应的场地或其他相关的事物。

（2）解散社团：经超级管理员和监管人员商议同意后由超级管理员或监管人员发起解散社团的操作，需要把该社团的成员角色更改为普通用户。

（3）社团信息变更：可以编辑社团头像、社团描述、社团实际人数、社团计划人数等。

（4）消息推送：有超级管理员发送给所有监管人员，监管人员发送给所有社长，社长发送给所有社团成员几种类型。

（5）社团监控管理功能：社团监管人员对所监控的社团日常扣分情况进行录入，仅监管人员有权限。修改社团的总积分，以便能够及时修改错误，造成不必要的误会。

资料管理模块功能有如下几种：

（1）分享资料：要分享的资料以压缩包进行上传，可以选择资料的公开度，最后申请分享该资料。如果是在本社团分享资料无需审核即可分享，若分享资料到公共区需要监管人员或超级管理员同意后才可分享。普通成员、监管人员、超级管理员，这几种角色只能分享资料到公共资料区，并超级管理员发布资料无需审核，而且上传的资料信息将不可更改。

（2）查看资料列表：用户可以查看发布到公共区或者自己社团内部的资料，这样就给社团内部留一些自主的空间，以便其更好的利用该系统。

（3）下载资料：用户可下载自己权限范围内的资料包。

（4）删除资料：用户可删除自己发布的资料。

项目管理模块功能有如下几种：

（1）发布项目：任何人都可以申请发布项目，并提交项目相关的需求资料，以及项目相关的酬金等，但发布的项目只有通过监督人员的审核后才可以被其他人查看申请项目，注意，项目一经发布将不可以对其中的信息进行修改。

（2）申请项目：用户可以申请处于“悬赏中”状态的项目，经发布者同意后可以开发项目，获取酬劳。

（3）删除项目：自己只能删除自己发布的项目，监管人员也不可以删除其他人的项目。

新闻活动模块功能有如下几种：

（1）发布新闻活动：该模块主要为社团管理者提供，只有超级管理员或社团监管人员可以在该模块发布一些实时动态消息，为社团团体增添一些活力。

（2）删除新闻活动：社团监管人员或超级管理员都可以删除已发布的新闻消息。

讨论模块功能有如下几种：

发布主贴：用户可以将自己遇见的问题发布到系统，向大家求助。

评论主贴：用户可以在主贴详情页发表自己的看法或解决思路，形成一个良好的氛围。

删除主贴：用户可以删除自己的主帖。

## 面向对象分析

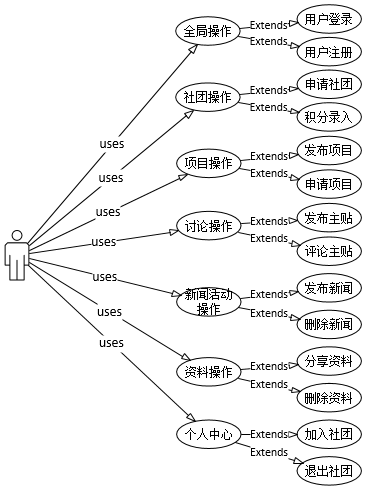


图3-1 用户功能用例图

（1）用户登录用例的描述

用例名称：用户登录

用例目的：实现登录

参与者：用户

前置条件：成功进入网站

基本事件流：①进入网站登录页；②输入用户名、密码和验证码；③点击‘登录’按钮；④验证通过，进入到系统首页

后置条件：完成用户登录

（2）申请加入社团用例的描述

用例名称：申请加入社团

用例目的：实现加入社团的申请

参与者：用户

前置条件：成功进入申请加入社团的页面

基本事件流：①进入申请加入社团的页面；②输入申请社团的理由、选择申请社团类型；③点击‘申请加入’按钮；④系统验证通过，创建申请记录

后置条件：完成加入社团的申请

（3）删除新闻活动用例的描述

用例名称：删除新闻活动

用例目的：删除无用的新闻活动

参与者：用户

前置条件：进入新闻活动管理页面

基本事件流：①进入新闻活动管理页面；②选择要删除的新闻活动；③点击其后面的‘删除’按钮；④删除成功

后置条件：完成新闻活动的删除

（4）积分录入用例的描述

用例名称：积分录入

用例目的：扣除社团积分

参与者：用户

前置条件：进入积分录入页面

基本事件流：①进入积分录入页面；②填写要扣除的积分、扣分的理由和选择要扣除积分的社团；③点击‘扣分’按钮；④扣分成功

后置条件：实现社团积分的扣除

# 第四章 系统设计

## 概要设计

本系统的功能模块主要分为全局功能模块、社团管理模块、项目管理模块、讨论管理模块、新闻活动管理模块、资料管理模块、用户管理模块。具体功能如下图4-1所示：

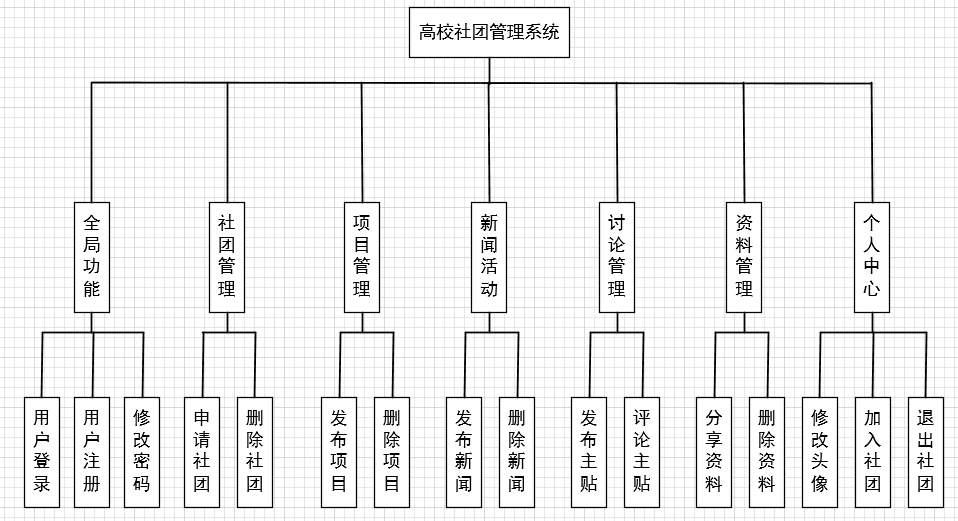


图4-1 系统功能总体设计图

## 详细设计

### 注册详细设计

用户访问系统注册页面，需要用户填写注册的信息，提交表单后会在后台对用户的学号进行验证。如果学号存在且没有注册该系统，开始验证昵称和密码的合法性，并使用django内置的加密算法对密码进行加密，然后在数据库中保存用户信息。如4-2图：

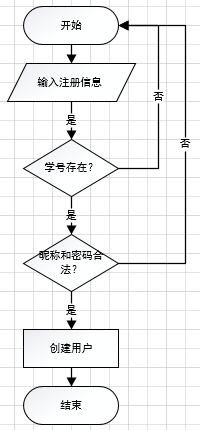


图4-2 用户注册流程图

### 用户登录详细设计

用户访问系统登录页面，需要填写已注册用户的用户名或学号、密码以及验证码信息，提交表单后会在后台对验证码、学号或用户以及密码进行验证。如果验证通过将跳转到系统首页，否则提示登录失败的信息。如下图4-3所示：

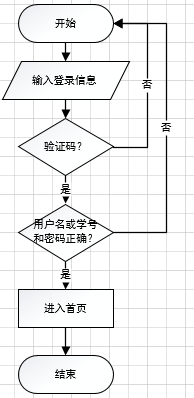


图4-3 用户登录流程图

### 用户修改密码详细设计

已登录的用户访问修改密码页面，需要填写原始密码、新密码、确认密码信息，提交表单后会在后台对其合法性进行验证。如果通过将新密码更新到数据库。如4-4：

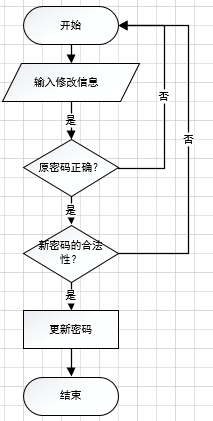


图4-4 修改密码流程图

### 发布项目详细设计

已登录的用户访问发布项目页面，需要填写项目名称、项目描述、上传需求文档和项目酬金信息，提交表单后会在后台对其合法性进行验证。如果验证通过需判断发布项目的用户权限，超级管理员或社团监管人员创建的项目可以直接发布，其他用户则需要审核后才可以发布到该系统。如果验证不通过返回错误信息。如下图4-5所示：

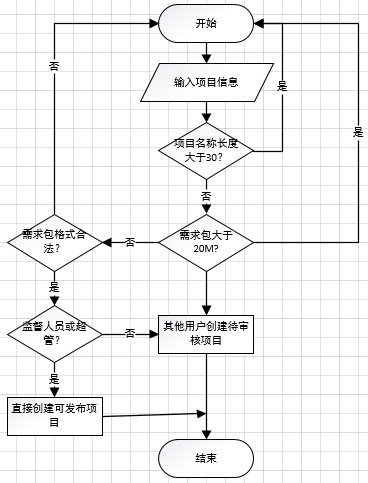


图4-5 发布项目流程图

### 分享资料详细设计

已登录用户访问发分享资料的页面，需填写资料名称、资料描述、选择资料分类、上传资料包、选择公开度信息。提交表单后在后台首先会对资料名称和资料包的大小和格式进行合法性验证。如果验证通过，需要判断发布项目的用户权限，超级管理员或社团监管人员，直接发布资料到公共区。其他用户需要判断其公开度，如果要发布资料到公共区需要超级管理员或社团监管人员进行审核，若社团成员想发布到社团内部可以直接发布。如果验证不通过，返回错误信息。

如下图4-6所示：

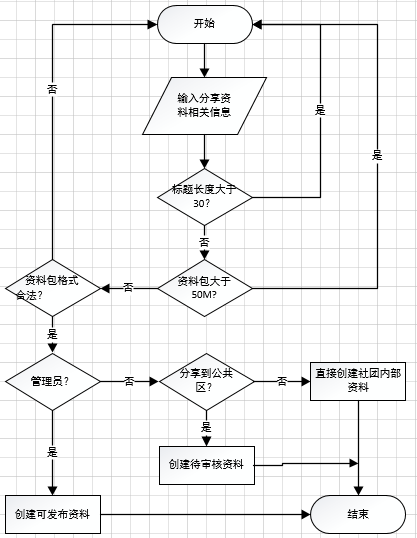


图4-6 分享资料流程图

### 发布主贴详细设计

已登录用户访问发布主贴页面，填写主贴标题、分类、内容信息，提交表单后在后台对其合法性进行验证。如果验证通过发布主贴，否则返回错误信息。如4-7图：

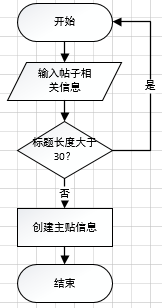


图4-7发布主贴流程图

### 发布新闻活动详细设计

已登录的用户访问发布新闻活动页面，填写标题、描述、上传图片信息，提交表单后在后台对其合法性进行验证。如果验证通过发布新闻活动，否则返回错误信息。

如下图4-8所示：

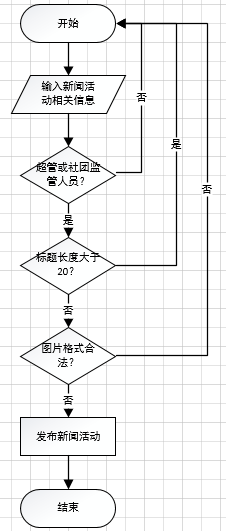


图4-8 发布新闻活动流程图

### 积分录入详细设计

已登录的用户访问积分录入页面，填写扣除的分数、扣分原因、选择扣分的社团信息，提交表单后在后台对其合法性进行验证。如果验证通过则扣除该社团的积分，否则返回错误的信息。

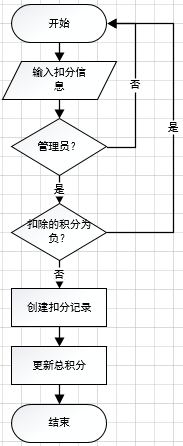


图4-9 积分录入流程图

## 数据库设计

### 数据库概念设计

在分析本系统业务需求功能后得到以下几个主要的实体，以下使其实体属性图以及数据库E-R图。

（1）用户所具有的属性：ID、用户编号、用户名、头像、所属社团、所属班级、角色、密码、真实姓名、添加时间，具体属性如下图4-10所示：

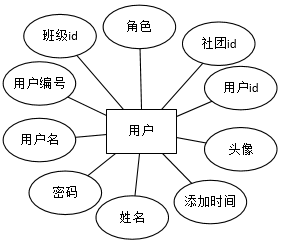


图4-10 用户属性图

（2）社团所具有的属性：ID、社团编号、社团名称、社团头像、所属分类、描述、所属学院、积分、计划人数、实际人数、创建时间、解散时间，具体属性如下图4-11所示。

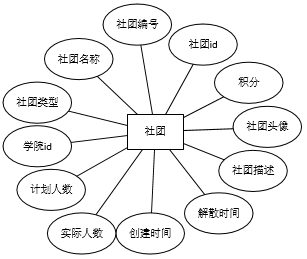


图4-11 社团属性图

（3）资料所具有的属性：ID、资料名称、资料描述、资料状态、所属分类、所属用户、公开度、资料包、上传时间，具体属性如下图4-12所示：

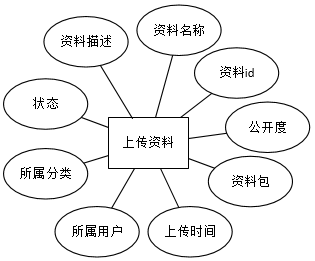


图4-12 资料属性图

（4）项目所具有的属性：ID、项目名称、项目描述、需求文件、酬金、发布人、项目状态、发布时间、申请人，具体属性如下图4-13所示：

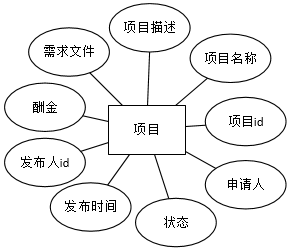


图4-13 项目属性图

（5）主贴所具有的属性：ID、主贴标题、内容、所属用户，具体属性如下图4-14所示：

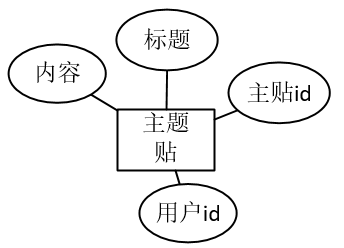


图4-14 主贴属性图

（6）新闻活动所具有的的属性：ID、活动名称、活动内容、发布人、发布时间、浏览量，具体属性如下图4-15所示：

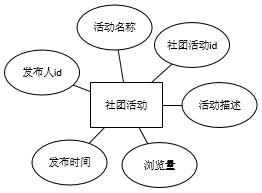


图4-15 新闻活动属性图

（7）数据库实体E-R图如下图4-16所示：

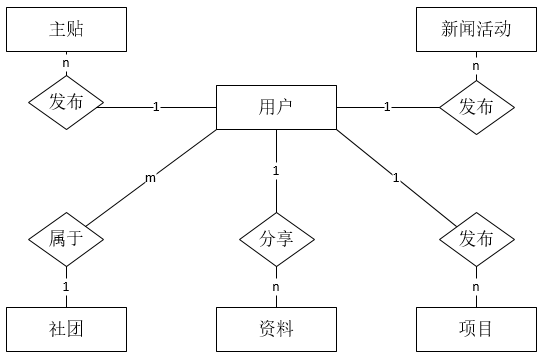


图4-16 数据库实体E-R图

### 数据库结构设计

本系统的数据库为Mysql数据库，主要相关数据表如下：

（1）用户信息表：ID id，用户编号 sno，真实姓名 realname，所属班级grade，角色 role，所属社团 community，社团头像 image，用户名 username，密码 password，手机号 phone，添加时间 date\_joined。

表4-1 用户信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | joined | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| sno | 用户编号 | CharField（20） | True |  | False | 唯一值 |
| realname | 真实姓名 | CharField（20） |  |  | False |  |
| grade | 所属班级 | ForeignKey（） |  |  | True | 外键到班级表 |
| role | 用户角色 | IntegerField（） |  | 0 | False |  |
| community | 所属社团 | ForeignKey（） |  |  | True | 外键到社团表 |
| image | 用户头像 | FileField（） |  | 默认头像 | False |  |
| phone | 手机号 | CharField（11） |  |  | True |  |
| username | 昵称 | CharField（20） |  |  | False |  |
| password | 密码 | CharField（64） |  |  | False |  |
| date\_joined | 添加时间 | DateTimeField |  | 当前时间 | False |  |

（2） 资料信息表：ID id，资料名称 name，资料描述 desc，资料状态 status，所属分类 type，所属用户 user，资料包 data，公开度 disclosure，添加时间 add\_time。

表4-2 资料表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | 默认值 | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| name | 资料名称 | CharField（30） |  |  | False |  |
| desc | 资料描述 | CharField（200） |  |  | True |  |
| status | 资料状态 | IntegerField |  | 0 | False |  |
| type | 所属分类 | ForeignKey |  |  | False | 外键到资料分类表 |
| user | 所属用户 | ForeignKey |  |  | False | 外键到用户表 |
| data | 资料包 | OneToOneField |  |  | False | 关联到资料包表 |
| disclosure | 公开度 | IntegerField |  |  | False |  |
| add\_time | 添加时间 | DateTimeField |  | 当前时间 | False |  |

（3）社团信息表：ID id，社团编号 no，社团名称 name，社团宗旨 objective，社团头像 image，社团描述 desc，所属学院 college，社团类型 community\_type，计划人数 plan\_count，实际人数 real\_count，社团积分 score，创建社团日期 create\_date，解散社团日期 dis\_date。

表4-3 社团信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | 默认值 | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| no | 社团编号 | CharField（30） | True |  | False | 唯一值 |
| name | 社团名称 | CharField（30） | True |  | False | 唯一值 |
| objective | 社团宗旨 | CharField（200） |  |  | False |  |
| image | 社团头像 | FileField() |  | 默认logo | False |  |
| desc | 社团描述 | TextField（500） |  |  | True |  |
| college | 所属学院 | ForeignKey（） |  |  | True | 外键到学院表 |
| community\_type | 社团类型 | IntegerField |  | 6 | False |  |
| plan\_count | 计划人数 | IntegerField |  | 0 | False |  |
| real\_count | 实际人数 | IntegerField |  | 0 | False |  |
| score | 社团积分 | IntegerField |  | 100 |  |  |
| create\_date | 创建社团日期 | DateTimeField |  | 当前时间 | False |  |
| dis\_date | 解散社团日期 | DateTimeField |  |  | True |  |

（4）新闻活动信息表：ID id，活动名称 name，活动描述 desc，发布者 user，浏览量 count，发布时间 add\_time。

表4-4 新闻活动表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | 默认值 | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| name | 活动名称 | CharField（30） |  |  | False |  |
| desc | 活动描述 | TextField（1000） |  |  | False |  |
| user | 发布者 | ForeignKey |  |  | False | 外键到用户表 |
| count | 浏览量 | IntegerField |  | 0 | False |  |
| add\_time | 发布时间 | DateTimeField |  | 当前时间 | False |  |

（5） 项目信息表：ID id，项目名称 name，项目描述 desc，需求文件 file，发布者 pub\_user，项目状态 status，酬金 remuneration，申请人 apply\_user，发布之间 add\_time。

表4-5 项目表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | 默认值 | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| name | 项目名称 | CharField(30) |  |  | False |  |
| desc | 项目描述 | TextField（200） |  |  | True |  |
| file | 需求文件 | FileField（） |  |  | True |  |
| pub\_user | 发布者 | ForeignKey |  |  | False | 外键到用户表 |
| status | 项目状态 | IntegerField |  | 0 | False |  |
| remuneration | 酬金 | IntegerField |  | 0 | False |  |
| apply\_user | 申请人 | ForeignKey |  |  | True | 外键到用户表 |
| add\_time | 发布时间 | DateTimeField |  | 当前时间 | False |  |

（6）主贴信息表：ID id，标题 title，所属分类 category，内容 content，所属用户 user，发布时间 add\_time。

表4-6 主贴表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 解释 | 类型 | 索引 | 默认值 | 可为空 | 备注 |
| id | ID | IntegerField |  |  | False | 自增 |
| title | 标题 | CharField |  |  | False |  |
| catagory | 所属分类 | ForeignKey |  |  | False |  |
| content | 内容 | TextField |  |  | False |  |
| user | 所属用户 | ForeignKey |  |  | False | 外键到用户表 |
| add\_time | 发布时间 | DateTimeField |  |  | False |  |

# 第五章 系统实现

## 核心功能代码

### 注册模块实现

用户需要填写身份信息，后台会对学号或工号，以及用户名和密码的合法性进行验证，验证成功后后台会将用户的信息写入到数据库的用户信息表。如5-1图所示：

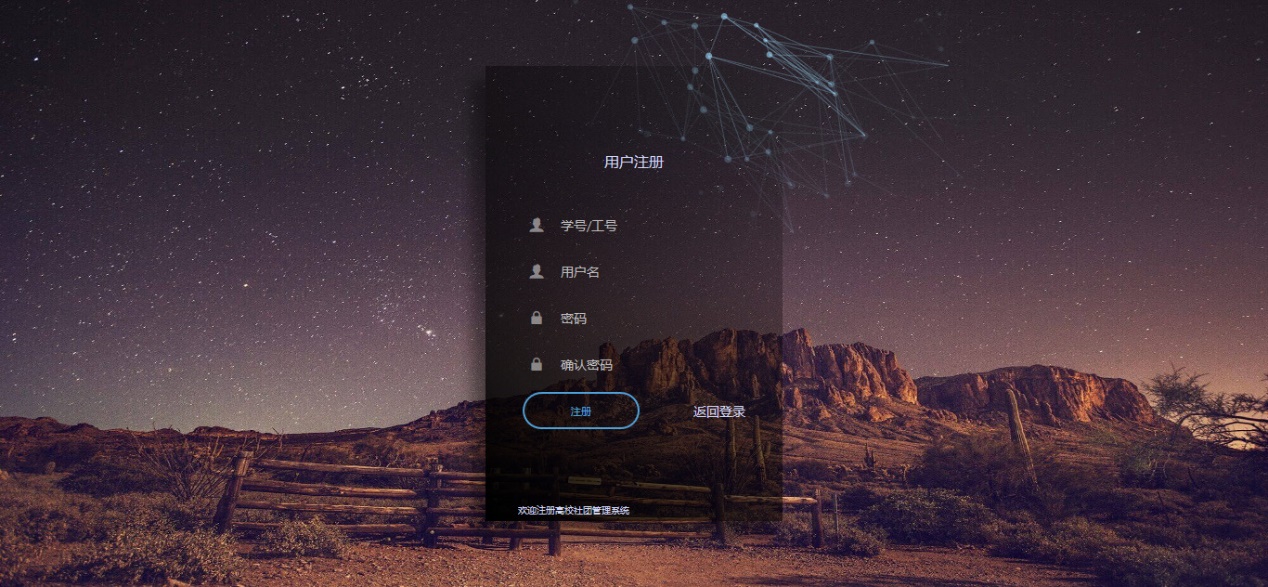


图5-1 用户注册界面

注册的核心代码如下：

# 注册序列化

class UserCreateSerializer(serializers.ModelSerializer):

sno = serializers.CharField(required=True, max\_length=20, label='学号/工号')

username = serializers.CharField(required=True, label='昵称')

password = serializers.CharField(required=True, min\_length=8, write\_only=True, label='密码')

class Meta:

model = User

fields = ('id', 'sno', 'username', 'password')

# 对学号的合法向进行验证

def validate\_sno(self, sno):

obj = User.objects.filter(sno=sno).first()

if obj:

raise serializers.ValidationError('该账号已经注册。')

return sno

# 对用户名的合法性进行验证

def validate\_username(self, username):

if username and len(username) > 20:

raise serializers.ValidationError('用户名最多20个字符。')

obj = User.objects.filter(username=username).first()

if obj:

raise serializers.ValidationError('用户名已经存在。')

return username

# 对密码的合法性进行验证

def validate\_password(self, password):

# 正则验证

if not is\_valid\_password(password):

raise serializers.ValidationError('请输入字母（区分大小写）、数字、符号中的至少两种8-64个字符')

return make\_password(password)

def create(self, validated\_data):

# 获取可以注册的对象信息

register = Register.objects.get(sno=validated\_data['sno'])

no = register.no

grade\_obj = Grade.objects.filter(no=no).first()

grade = None

if grade\_obj:

grade = grade\_obj

validated\_data.update({'grade': grade, 'realname': register.name})

# 存储数据

instance = User.objects.create(\*\*validated\_data)

return instance

### 登录模块实现

用户输入注册成功所需要的信息，在前端提交后，后台对用户输入的数据对合法性验证，验证成功，登录该系统。

登录页面如5-2图所示：



图5-2 用户登录界面

登录的核心代码如下：

# 序列化(serializers.py)

class LoginView(generics.GenericAPIView):

"""用户登录"""

serializer\_class = LoginSerializer

def post(self, request):

agree = request.scheme

host = request.META['HTTP\_HOST']

# 获取登录的serializer类

serializer\_class = self.get\_serializer\_class()

serializer = serializer\_class(data=request.data, context={'request': request})

# 验证输入数据的合法性

if serializer.is\_valid():

user = User.objects.get(Q(username=request.data['username']) | Q(sno=request.data['username']))

login(request, user)

image = agree + '://' + host + request.user.image.url

# 构建前端所需的数据

data = {"id": request.user.id, "username": request.user.username,

'role': request.user.role, 'image': image}

return Response(data=data, status=status.HTTP\_200\_OK)

else:

return Response(data=serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)

# 视图(view.py)

class LoginSerializer(serializers.Serializer):

username = serializers.CharField(required=True, label='用户名')

password = serializers.CharField(required=True, label='密码')

code = serializers.CharField(required=True, label='验证码')

# 对输入数据的字段进行合法性验证

def validate(self, attrs):

username = attrs['username']

password = attrs['password']

code = attrs['code'].upper()

generate\_code = self.context['request'].session.get("valid\_code", "").upper()

if code == generate\_code:

user = User.objects.filter(Q(username=username) | Q(sno=username)).first()

if user:

if not user.check\_password(password):

raise serializers.ValidationError({"username": ["用户名或密码错误"]})

else:

raise serializers.ValidationError({"username": ["用户名或密码错误"]})

else:

raise serializers.ValidationError({'code': ['验证码错误']})

return attrs

### 申请加入社团

普通用户输入申请加入社团的理由，选择要加入的社团，点击下方的申请加入按钮，申请加入社团。

申请加入社团界面如图5-3所示：

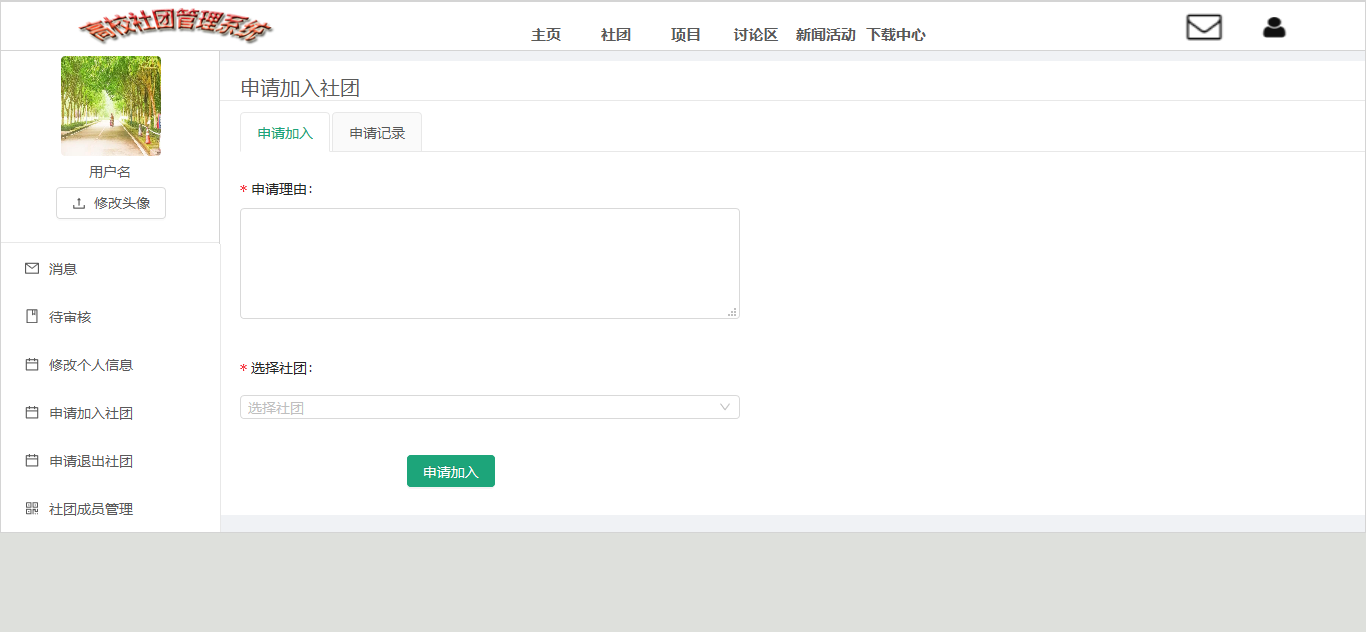


图5-3 申请加入社团

申请加入社团核心代码如下：

# 序列化

class ApplyRecordCreateSerializer(serializers.ModelSerializer):

class Meta:

model = ApplyRecord

fields = ('id', 'content', 'apply\_data', 'type', 'obj\_id')

# 申请对象id验证

def validate(self, attrs):

# (0, '申请加入社团'),

# (1, '申请发布资料'),

# (2, '申请发布项目'),

# (3, '申请创建社团')

id = attrs['obj\_id']

type = attrs['type']

if type == 0 and id:

obj = Community.objects.filter(id=id).first()

if not obj:

raise serializers.ValidationError({"error": "申请的对象不存在"})

if type == 1 and id:

obj = Data.objects.filter(id=id).first()

if not obj:

raise serializers.ValidationError({"error": "申请的对象不存在"})

if type == 2 and id:

obj = Project.objects.filter(id=id).first()

if not obj:

raise serializers.ValidationError({"error": "申请的对象不存在"})

return attrs

def create(self, validated\_data):

user = self.context['request'].user

validated\_data.update({'apply\_user': user})

instance = ApplyRecord.objects.create(\*\*validated\_data)

return instance

### 社团列表、搜索社团

登录用户可以查看全部社团列表以及积分排名。

社团列表页面如5-4图所示：



图5-4 社团列表、搜索社团界面

社团列表、搜索社团核心代码如下：

# 视图(view.py)

class CommunityListView(generics.ListAPIView):

"""社团列表"""

queryset = Community.objects.all()

serializer\_class = CommunityRetrieveSerializer

permission\_classes = (IsAuthenticated,)

# 使用django-filter过滤组件封装自定义过滤器

filter\_backends = (CustomDjangoFilterBackend,)

filterset\_fields = ('name',)

def get\_queryset(self):

# 获取积分排名TOP5的社团

top5 = self.request.GET.get('top5', None)

if top5:

return self.queryset.all().order\_by('-score')[:5]

return self.queryset.all().order\_by('-score')

### 社团详情

登录的用户可以查看各个社团的主页，以便了解各个社团。

社团列表页面如5-5图所示：

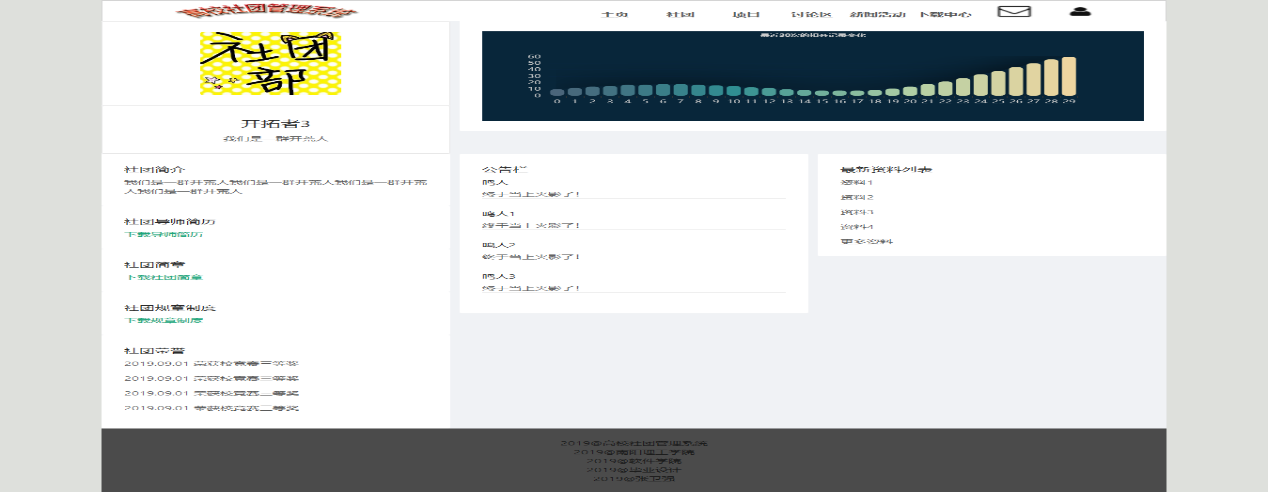


图5-5 社团详情

社团详情核心代码为：

# 序列化

class CommunityRetrieveSerializer(serializers.ModelSerializer):

college = serializers.CharField(source='college.name')

files = serializers.SerializerMethodField(label='社团相关文件')

honor = serializers.SerializerMethodField(label='社团荣誉')

announcement = serializers.SerializerMethodField(label='社团公告')

class Meta:

model = Community

fields = ('id', 'no', 'name', 'objective', 'image', 'desc', 'plan\_count', 'real\_count', 'college',

'community\_type', 'files', 'score', 'honor', 'announcement')

def get\_files(self, obj):

request = self.context['request']

agree = request.scheme

host = request.META['HTTP\_HOST']

teacher\_file = None

community\_file = None

community\_file\_rule = None

teacher = CommunityFile.objects.filter(community=obj, community\_file\_type=0).first()

community = CommunityFile.objects.filter(community=obj, community\_file\_type=1).first()

rule = CommunityFile.objects.filter(community=obj, community\_file\_type=2).first()

# 构建media文件路径

if teacher:

teacher\_file = agree + '://' + host + CommunityFileSerializer(teacher).data['file']

if community:

community\_file = CommunityFileSerializer(teacher).data['file']

if rule:

community\_file\_rule = agree + '://' + host + CommunityFileSerializer(teacher).data['file']

data = {'teacher\_file': teacher\_file, 'community\_file': community\_file,

'community\_file\_rule': community\_file\_rule}

return data

# 获取荣誉列表

def get\_honor(self, obj):

queryset = Honor.objects.filter(community=obj)

serializer = HonorSerializer(queryset, many=True)

return serializer.data

# 获取公告列表

def get\_announcement(self, obj):

queryset = Announcement.objects.filter(community=obj).order\_by('-add\_time')[:5]

serializer = AnnouncementSerializer(queryset, many=True)

return serializer.data

### 发布新闻活动

用户输入新闻消息的标题、内容，上传图片，保存到数据库。

发布新闻活动页面如5-6图所示：



图5-6 发布新闻活动

发布新闻活动核心代码如下：

# 序列化

class NewsCreateSerializer(serializers.ModelSerializer):

name = serializers.CharField(required=True, max\_length=50, label='名称')

desc = serializers.CharField(required=True, max\_length=1000, label='描述')

images = serializers.ListField(

child=serializers.FileField(max\_length=100000, allow\_empty\_file=True, allow\_null=True, use\_url=True),

)

class Meta:

model = News

fields = ('id', 'name', 'desc', 'images')

def validate(self, attrs):

role = self.context['request'].user.role

if role == 0 or role == 1 or role == 2:

raise serializers.ValidationError({'error': '只有超级管理员或监督人员可以创建新闻活动'})

return attrs

# 创建新闻活动对象

def create(self, validated\_data):

user = self.context['request'].user

images = validated\_data.pop('images', None)

validated\_data.update({'user': user})

instance = News.objects.create(\*\*validated\_data)

if images:

# 保存图片

NewsImages.objects.create(name=images.name, file=images, news=instance)

return instance

def to\_representation(self, instance):

ret = OrderedDict()

fields = self.\_readable\_fields

for field in fields:

try:

attribute = field.get\_attribute(instance)

except Exception as e:

del e

continue

check\_for\_none = attribute.pk if isinstance(attribute, PKOnlyObject) else attribute

if check\_for\_none is None:

ret[field.field\_name] = None

else:

ret[field.field\_name] = field.to\_representation(attribute)

return ret

### 积分记录处理

本系统对社团积分进行记录研究，以便直观的看出社团近期的扣分变化，其主要的实现思路：由于本系统是前后端分离，使用djangorestframe开源库做后端接口的开发，后端每对一个社团做扣分的记录调用接口时，都要在序列化的时候去更新社团的总积分。

序列化代码为：

# 积分序列化

class ScoreRecordSerializer(serializers.ModelSerializer):

deduct = serializers.IntegerField(required=True, label='扣除的积分')

desc = serializers.CharField(allow\_null=True, allow\_blank=True, max\_length=500, label='原因')

add\_time = serializers.DateTimeField(read\_only=True, label='添加时间')

class Meta:

model = ScoreRecord

fields = ('id', 'score', 'deduct', 'desc', 'add\_time')

def create(self, validated\_data):

user = self.context['request'].user

validated\_data.update({'user': user})

instance = super().create(validated\_data)

# 添加积分记录，更新积分总分

score = instance.score

score.score = F('score') - instance.deduct

score.save()

return instance

def update(self, instance, validated\_data):

validated\_data = {k: v for k, v in validated\_data.items() if v}

if 'deduct' in validated\_data.keys():

c = instance.deduct - validated\_data['deduct']

# 添加积分记录，更新积分总分

score = instance.score

score.score = F('score') + c

score.save()

return super().update(instance, validated\_data)

# 第六章 系统测试

## 功能模块测试

### 全局功能模块测试

该模块测试主要验证用户注册登录修改密码是否能正常使用，任何不符合逻辑的都应给出提示。在该模块的测试中密码不能少于8位，否则提示输入不规范。修改密码和注册时需要输入两次相同密码，否则提示输入不规范。登录时必须正确输入验证码，否则提示输入不规范。

表6-1 全局功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 用户注册 | 在注册界面输入学号、昵称、密码符合规范 | 注册成功 | 与预期结果一致 |
| 用户注册 | 在注册界面输入不存在的序号 | 注册失败 | 与预期结果一致 |
| 用户注册 | 在注册界面输入不符合规范的昵称 | 注册失败 | 与预期结果一致 |
| 用户注册 | 在注册界面输入不符合规范的密码 | 注册失败 | 与预期结果一致 |
| 用户登录 | 输入正确的用户名或编号、密码、验证码 | 登录成功 | 与预期结果一致 |
| 用户登录 | 输入错误的用户名或密码 | 登录失败 | 与预期结果一致 |
| 用户登录 | 输入错误的验证码 | 登录失败 | 与预期结果一致 |
| 修改密码 | 输入正确的原始密码、新密码、确认密码 | 修改成功 | 与预期结果一致 |
| 修改密码 | 输入错误的原始密码 | 修改失败 | 与预期结果一致 |

### 社团管理模块测试

该模块主要对申请社团、社团管理进行了测试，在申请社团是社团名称必须唯一，上传的申请资料包仅支持zip、rar、tar三种格式的压缩包。

表6-2 社团管理模块功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 申请社团 | 输入合法的社团名称、描述、上传压缩包 | 申请成功 | 与预期结果一致 |
| 申请社团 | 输入已存在的社团名称 | 申请失败 | 与预期结果一致 |
| 申请社团 | 输入不符合规范文件 | 申请失败 | 与预期结果一致 |
| 社团管理-删除社团 | 点击删除按钮 | 删除成功 | 与预期结果一致 |

### 项目管理模块

本模块主要对申请项目、发布项目、项目管理进行了测试。申请的项目状态必须是在“悬赏中”的才可以申请，发布项目的名称不能超过30个字符，上传的文件仅支持docx。

表6-3 项目管理模块功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 申请项目 | 进入申请项目页，选择处于悬赏中的项目 | 申请成功 | 与预期结果一致 |
| 申请项目 | 进入申请项目页，选择处于已完成的项目 | 申请失败 | 与预期结果一致 |
| 发布项目 | 进入发布项目页，输入合法信息，点击发布按钮 | 发布成功 | 与预期结果一致 |
| 发布项目 | 进入发布项目页，输入超过20个字符的标题，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 发布项目 | 进入发布项目页，上传不符合规范的文件，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 项目管理-删除项目 | 进入项目管理页，选择要删除的项目，点击删除按钮。 | 删除成功 | 与预期结果一致 |

### 讨论区测试

本模块主要对发布主贴，主贴管理进行了测试。发布主贴的主贴标题不能超过30个字符，描述不能超过500个字符。

表6-4 讨论区功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 发布帖子 | 进入发布帖子页，输入合法的信息，点击发布按钮 | 发布成功 | 与预期结果一致 |
| 发布帖子 | 进入发布帖子页，输入超过20个字符的帖子标题，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 发布帖子 | 进入发布帖子页，输入超过500个字符的帖子描述，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 帖子管理-删除帖子 | 进入帖子管理页，选择要删除的帖子，点击删除按钮 | 删除成功 | 与预期结果一致 |

### 新闻活动管理模块测试

本模块主要对发布新闻活动、新闻活动管理进行了测试。发布新闻活动的新闻标题不能超过30个字符，上传的图片仅支持png、jpg、jpeg三种格式。

表6-5 新闻活动管理模块功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 发布新闻活动 | 进入发布新闻活动也，输入合法的标题、描述、上传合法的图片，点击发布按钮 | 发布成功 | 与预期结果一致 |
| 发布新闻活动 | 进入发布新闻活动也，输入不合法的标题，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 发布新闻活动 | 进入发布新闻活动页，上传不符合规范的文件，点击发布按钮 | 发布失败 | 与预期结果一致 |
| 新闻活动管理-删除新闻活动 | 进入新闻活动管理页，选择要删除的新闻活动，点击删除按钮 | 删除成功 | 与预期结果一致 |

### 资料管理模块测试

本模块主要对分享资料、资料管理进行了测试。分享的资料标题不能超过30个字符，并且分享的资料包仅支持zip、rar、tar三种格式的压缩包。

表6-6 资料管理模块功能测试

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项目 | 测试方法 | 预期结果 | 结论 |
| 分享资料 | 进入分享资料页，输入规范的标题、描述、上传规范的资料包，点击分享 | 分享成功 | 与预期结果一致 |
| 分享资料 | 进入分享资料页，输入超过30个字符的标题，点击分享 | 分享失败 | 与预期结果一致 |
| 分享资料 | 进入分享资料页，上传不符合规范的资料包，点击分享 | 分享失败 | 与预期结果一致 |
| 资料管理-删除资料 | 进入资料管理页，选择要删除的资料，点击删除按钮 | 删除成功 | 与预期结果一致 |

结束语

经过三个月的辛苦，高校社团管理系统平台终于实现并运行正常运行了，系统能够实现了预期的每个业务模块的功能，能够满足社团管理的基本需求，相信这个系统能够有效的管理社团。在该系统的设计阶段中，深深的感受到自己的知识不足，对软件工程的标准化开发流程半知半解，通过互联网搜索到谈软件开发的流程[11]，对我理解软件开发流程有很大的帮助。在设计前端页面时，自己画了一些简略图，并不能很好的设计功能，通过在技术交流群中求助，学习了一个名为墨刀的原型设计工具，使我在原型设计这块有了很大的提升。在实现前端页面时，自己又初步学习了React框架，又结合实习时同事推荐的Antd组件，实现了相对于我以前更加漂亮的页面。虽然在刚开始使用新学习的知识异常困难，通过文档、博客慢慢地学习摸索，终于可以使用其设计并实现一些简单的界面了。

在实现该系统的过程中，我感觉到对以往学习的知识有了一个全面的回顾，并在运用过程中进一步的梳理，又加深了一层印象。对于前后端分离的工程，有了一个全新的认识。

参考文献

1. 彭晓灵，高校社团目标管理的缺陷与完善[J]．大众商务，2009.
2. 王宇翔，胡莎莎，李天骄，当前高校学生社团管理工作浅析[J]，2010.
3. 刘永贵，郑旭东.美国高等教育信息化的进展与趋势——EDUCASUE年度高等教育信息化十大战略议题解读[J].开放教育研究, 2008(6):122-126.
4. 廖良辉，中美高校学生社团管理比较[J]．青年研究，2005.
5. 余俊杰，林启灼．高校社团管理系统的设计与实现[J]．科技广场，2011.
6. 刘长龙，Python高效开发实战[M]. 电子工业出版社，2016.
7. Collected by Clove，The Django Book2.0[OL]. <http://docs.30c.org/djangobook2/>, 2017.
8. 阮一峰，Restful API设计指南[OL]. <http://www.ruanyifeng.com/blog/2014/05/restful_api.html>，2014.
9. 明日科技，MySQL从入门到精通[M].清华大学出版社，2017.
10. 阿里技术团队，Ant Design[OL]. <https://ant.design/docs/react/getting-started-cn>, 2018.
11. 刘文辉，谈软件开发的流程[C]// 决策论坛——区域发展与公共政策研究学术研讨会.0.

致谢

首先，我要诚挚的感谢我的导师张哲老师，在近一年的企业实习和毕业设计期间，都得到张哲老师的悉心教导，在论文写作过程中多次得到他的督促，并在我忙碌无状态的时候给与我鼓励，非常感谢！

其次，我还要感谢同组的孙妍同学，给我讲一些社团管理的流程与她对社团管理的一些经验，并及时把信息同步给我，给予了我非常大的帮助，非常感谢。

最后要感谢几年来给我们上课的老师与学院的各位工作人员，正是有他们辛勤的劳动，才使我今天有足够的能力完成整个项目，在此向他们表示真诚的谢意。